Large Scale Deployment of BERT Based Cross Encoder Model for Re-Ranking in Walmart Search Engine

Ajit Puthenputhussery
Walmart Global Tech
Hoboken, New Jersey, USA

Changsung Kang
Walmart Global Tech
Sunnyvale, California, USA

Alessandro Magnani
Coupang
Mountain View, California, USA

Tian Zhang
Hongwei Shang
Nitin Yadav
Walmart Global Tech
Sunnyvale, California, USA

Prijith Chandran
Bhavin Madhani
Yuan-Tai Fu
He Wang
Walmart Global Tech
Sunnyvale, California, USA

Zbigniew Gasiorek
Walmart Global Tech
Hoboken, New Jersey, USA

Salvatore Tornatore
Srikanth Dasaka
Vivek Agrawal
Walmart Global Tech
Sunnyvale, California, USA

Michael Bowersox
Cun Mu
Walmart Global Tech
Hoboken, New Jersey, USA

Ciya Liao
Walmart Global Tech
Sunnyvale, California, USA

Abstract

Re-ranking plays a crucial role in product search by reassessing products from the primary retrieval system based on specific engagement and relevance criteria. While transformer-based models like the cross encoder have advanced the relevance of ranking models in recent years, a significant challenge arises from the high latency cost associated with running a cross encoder model at runtime. This challenge becomes more pronounced in the long-tail segment, where conventional techniques like caching prove ineffective. To tackle these issues, our paper introduces a scalable framework featuring a BERT-based cross encoder model for re-ranking, deployed in the Walmart search engine. We employ strategies such as intermediate representations, operator fusion, and vectorization to improve the inference latency of the cross encoder model. Furthermore, we provide a detailed discussion on the runtime implementation, highlighting key learnings and practical tricks that ensured minimal impact on response latency during production. Finally, we present the results of online experiments, including manual evaluation and interleaving test conducted on real-world e-commerce search traffic.

CCS Concepts

• **Information systems** → **Retrieval models and ranking**.

Keywords

product search, ranking optimization, e-commerce ranking

ACM Reference Format:
Ajit Puthenputhussery, Changsung Kang, Alessandro Magnani, Tian Zhang, Hongwei Shang, Nitin Yadav, Prijith Chandran, Bhavin Madhani, Yuan-Tai Fu, He Wang, Zbigniew Gasiorek, Salvatore Tornatore, Srikanth Dasaka, Vivek Agrawal, Michael Bowersox, Cun Mu, and Ciya Liao. 2025. Large Scale Deployment of BERT Based Cross Encoder Model for Re-Ranking in Walmart Search Engine. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '25), July 13–18, 2025, Padua, Italy*. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3726302.3731965

SIGIR '25, Padua, Italy

ACM ISBN 979-8-4007-1592-1/2025/07
https://doi.org/10.1145/3726302.3731965

1 Introduction

Product search in e-commerce is an emerging research area in information retrieval and is gaining increasing popularity in recent years [1, 3–5, 7–11, 13, 19, 21–24, 27, 28]. As many customers are switching to e-shopping, virtually all major retailers have their own product search engines, with some processing millions of queries per day. Given a user query, product search is the process of ranking a list of products satisfying multiple requirements such as improving user engagement, maximizing order likelihood, enhancing relevance, promoting diversity, among others. The process comprises retrieving a recall set of relevant products from the catalog and then re-ranking them to determine the best products for customer presentation. Previous studies [14, 16, 21] emphasizes the critical role of search result quality in ensuring user satisfaction and customer retention, highlighting the importance of developing product search engines that effectively balance relevance and user engagement criteria.

Many e-commerce companies are leveraging semantic search in their search engines to bridge vocabulary gaps and overcome token matching limitations [15, 17, 25, 26]. A common practice is to utilize a dual encoder architecture for capturing the relevance

between query and documents. The dual encoder models allow for more efficient inference since the document/product embeddings can be pre-computed. Recent research [6, 18] indicates that cross encoder models outperform dual encoder models, as they demonstrate better generalization to training data. A cross encoder model adopts a single-tower architecture by considering the joint sequence of query and product information. This allows interactions from the early stages of the model. However, a significant drawback is that during inference, all query-product pairs must pass through the model, resulting in a substantial increase in latency. Queries submitted to an e-commerce search engine follow a power law distribution, where infrequent (tail) queries have low occurrences, but collectively constitute a significant portion of the query volume [2, 20]. Traditional techniques like caching prove ineffective for tail queries due to this large volume, posing a challenge in deploying a cross encoder model for the long-tail segment.

To address the limitations highlighted above, we describe a scalable re-rank framework with the cross encoder model used at Walmart.com. We showcase the advantages of this system using online experiments, particularly for tail queries, and share insights gained during the production implementation. These insights encompass challenges related to enhancing the cross encoder model's performance and engineering considerations in deploying the model to production while maintaining a reasonable cost-to-serve. The main contributions of the paper are summarized as follows:

- We deploy a scalable framework featuring cross encoder BERT model for re-ranking in the Walmart search engine to handle the long-tail segment.
- We utilize techniques like intermediate representation, vectorization and fusing BERT layer operations for improving the performance of the cross encoder model.
- We share practical tricks for latency optimization from developing and deploying the framework in the e-commerce website that serves millions of online customers daily.
- We show the feasibility of the proposed framework using online experiments.

2 Framework Architecture and Implementation

In this section, we initially compare the dual encoder and cross encoder architecture. We then discuss the details of training and implementing the cross encoder model in production. Finally, we explore the architecture of our proposed framework for re-ranking.

2.1 Cross Encoder Model

The cross encoder model uses a single tower architecture and takes the concatenated sequence of both the query and the product information as input and performs dense cross interactions to predict the relevance score. The product information includes the title and attributes like product type, color, brand, and gender.

2.1.1 Dual Encoder vs Cross Encoder. The dual encoder adopts a two tower architecture, independently encoding the query and product information. It then calculates the similarity between the two independent embeddings to predict relevance. In contrast, the cross encoder integrates the query and product information, facilitating comprehensive interactions between their tokens from the model's early stages. While the dual encoder model theoretically boasts good capacity, it is susceptible to overfitting due to its factorized nature, leading to potential conflicts in the loss function. Consider a scenario with a query like "pokemon pikachu 10 inch", a positive product (product+) like "Squishmallows Plush 10 inch Pokemon Pikachu", and a negative product (product-) like "Squishmallows Plush 14 inch Pokemon Pikachu". In the dual encoder model, updating based on a positive pair (query, product+) may inadvertently raise the score for a negative pair (query, product-), especially when product+ bears superficial similarity to product-. This results in inefficient model training and a suboptimal score distribution, a challenge not encountered by the cross encoder.

However, numerous challenges arise during the deployment of the cross encoder model in production. In the case of the dual encoder, only the generation of the query embedding at runtime is required to compute similarity, while the product embeddings can be generated offline and stored in a key-value store. On the contrary, we need to trigger the cross encoder model for every query-product pair, introducing considerably more latency compared to the dual encoder.

2.1.2 Training Data and Model Details. The underlying model utilized is a BERT-base uncased model with 12 layers sourced from the HuggingFace repository. It was pre-trained on masked language modelling task using Walmart product catalog data and a binarized orders prediction task using past 1 year of Walmart engagement data. The primary objective of the pre-training phase was to enhance the BERT model's performance specifically for e-commerce product texts. Following pre-training, the model underwent fine-tuning using historical human relevance judgments obtained from past manual evaluation tests, employing a weighted cross-entropy loss function with 3 classes. Human evaluators, guided by well-defined criteria, assessed the relevance of query-product pairs on a 3-pt scale (rating 2 - exact match, rating 1 - partial match, rating 0 - irrelevant).

2.1.3 Runtime Implementation. Unlike a dual encoder model, a cross encoder model incurs a substantial latency overhead as it must run for all query-product pairs in production. To reduce this latency, we implemented an intermediate representation of the model. Employing the operator fusion technique, we merged multiple operators within the BERT layers into one to eliminate redundant memory access. Additionally, we utilized vectorization to process multiple elements contiguous in memory to reduce the I/O latency. Implementing these techniques allowed us to accelerate the model inference by 5x compared to other standard frameworks.

The cross encoder model is served by a remote model serving which has access to the GPU nodes as shown in figure 1. The primary components within this service involve data deserialization, I/O tensor creation, model inference, and model output generation. The payload undergoes deserialization and is loaded into GPU memory using pre-allocated I/O tensors. Subsequently, the model is applied to the input to generate the class prediction, which is then forwarded to the re-rank component.

2.2 Re-rank Architecture

The overall architecture for the proposed framework is shown in figure 1. When a user enters a query, it is directed to the Query

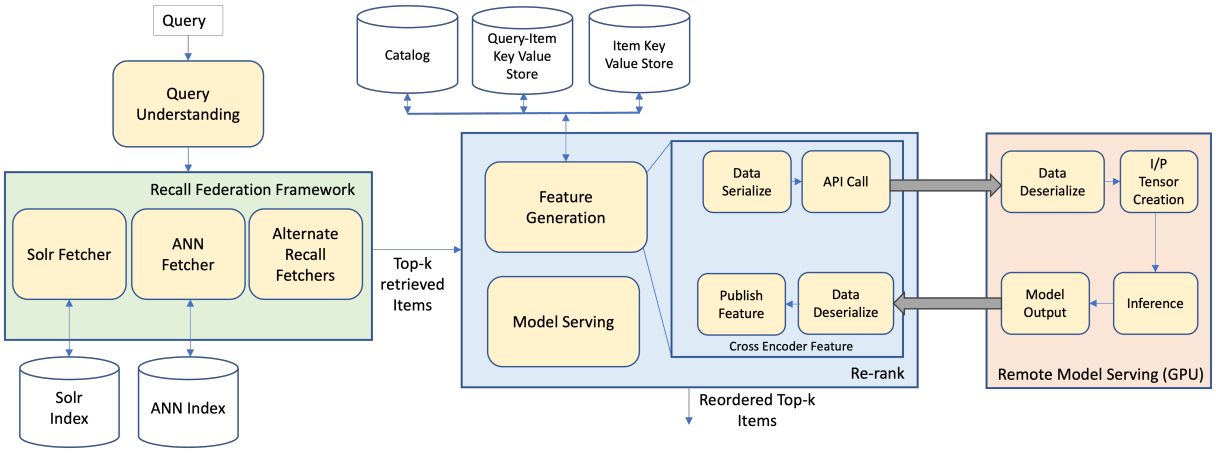

Figure 1: The overall framework architecture.

Understanding component. This component is responsible for determining query traffic type, understanding query intent, tagging query tokens to attributes, and generating query features, etc. Once the query traffic type has been identified, it generates a query plan for the Solr Fetcher and passes the query embedding to the approximate nearest neighbor (ANN) Fetcher. The Solr Fetcher retrieves items from the Solr index which is a traditional inverted index. The ANN Fetcher [15] retrieves the items that are closest to the query in the embedding space. There are additional recall fetchers that are activated based on certain conditions. The top items from all the recall are combined in the Recall Federation Framework and become the rerank recall set. This set is then ranked by the re-rank component. There are two main sub components in the re-rank component namely the feature generation and the model serving.

2.2.1 Feature Generation. The first subcomponent in re-rank is the feature generation step that is responsible for computing all features necessary for running the learning to rank model in the model serving stage. The features are either computed at runtime or they are pre-computed using a daily pipeline and saved to a key value store. The features used in the model can be classified into the following sub-categories:

- Semantic: these features capture the semantic relations between the query-item pair–e.g., dual encoder BERT embedding cosine similarity between query and item title [15], cross encoder feature.
- Text Match: these features are based on token match using the raw text of the query and the item descriptions–e.g., BM25 text match score, query-item token match ratio, etc.
- Query Attributes: these features capture the query intent and different attributes present in the query tokens–e.g., query attributes like size, color, product type, brand, etc.
- Item Attributes: these features capture the different item attributes computed at the item level–e.g., title attributes, title length, user ratings, user reviews, item department, etc.
- Engagement: these features capture the user interaction at the item, query-item and the query-item attribute level–e.g., query-item click rate, query-item add-to-cart rate [12].

2.2.2 Model Serving. The second component of the system utilizes a Gradient Boosted Decision Tree (GBDT) model to perform model inference, considering all features generated in the feature generation component. This process generates a ranking score, and the final list of items presented to customers is determined by sorting the items based on this re-rank score.

3 Latency Optimization

In this section, we discuss few practical tricks to minimize the response latency impact by the cross encoder model in production.

3.1 Using Key-Value Store for Product Tokens

In latency performance assessments, we observed that the BERT tokenization process accounted for roughly 30% of the total latency. This was primarily due to its application to all the products in the intermediate recall set during runtime. To address this, we implemented a strategy where we pre-compute the tokenized output of the product information and stored it in an Item key-value store. A daily pipeline was developed to compute the tokens for updated and new products. This optimization significantly decreased the tokenization latency from 30% to 3.5%.

3.2 Payload Compression

We noticed that the payload size of the request was considerably large since the product tokens from the entire intermediate recall set had to be forwarded to the remote model serving. This was contributing to the network latency, accounting for approximately 24% of the overall latency. To address this, we experimented with various compression schemes and determined that GZIP compression followed by base64 encoding offered the right balance between compression speed, size, and CPU usage. This optimization resulted in an average 10x reduction in payload size and a substantial improvement in network latency, reducing it from 24% to 1.5% of the overall latency.

3.3 Batching

To further minimize runtime latency impact, we segmented the intermediate product recall set into batches. The re-rank component initiated multiple parallel requests using these batches, which were then sent to the remote model serving. These batches were concurrently processed by the remote model serving cluster, and their outputs were combined in the re-rank component. We found that the batch size of 50 was optimal for our use case. Another important parameter for optimization was tuning the number of worker threads (num-workers) for the cross encoder model per GPU node. Each thread contains the model weights and pre-allocated I/O tensor allocations. The num-workers of the model depends on the model type, GPU memory and the GPU compute capacity. There is a trade-off between the num-workers and the GPU throughput. Too many workers could lead to inference latency degradation due to contention among the threads, while too few workers could reduce GPU throughput owing to higher queue latency in the GPU node. Batching and tuning the num-workers resulted in a 2x improvement in P50 and a 4x improvement in P95 overall latency.

4 Online Evaluation

In this section, we present the results of online experiments conducted on real-world e-commerce search traffic. Note that the control used in all experiments is the Walmart production system without the cross encoder feature.

4.1 Interleaving and A/B Test Results

We evaluated the user engagement performance of our proposed architecture in comparison to the current production system at Walmart using an interleaving technique. Interleaving is an online evaluation method where each user is exposed to a mix of ranking results from both the control and variation. The metric assessed is the number of items added to the cart at position 40 (ATC@40) for both the control and variation rankings. The outcomes presented in Table 1 and Figure 2 illustrate the improvements in user engagement performance, particularly in top positions within the long-tail segment.

We further tested the user engagement performance using the A/B testing technique. A/B testing is a method where users are split into two buckets, with one bucket exposed to the control and the other to variation. Metrics are then compared between these groups to evaluate the effect of the change. The assessment was done using the following metrics:

- Session with ATC: The number of sessions where at least one product was added to cart
- Total ATC: The number of products added to cart per user
- Session Abandonment Rate: The number of sessions where any product was not added to cart compared to the total sessions
- Clicks to ATC: The number of clicks conducted by a user before the product was added to cart

The A/B test was conducted for a period of two weeks. The results in table 1 demonstrate that the proposed framework increased the product add-to-carts improving the user engagement. In addition, we see a reduction in the sessions that are abandoned and fewer clicks before the product is added to cart.

Interleaving Test	Lift (P-value)
ATC@40 Lift	**+0.77%** (0.00)

AB Test	Lift (P-value)
Sessions with ATC	**+0.37%** (0.00)
Total ATC	**+0.52%** (0.00)
Session Abandonment Rate	**-0.67%** (0.00)
Clicks to ATC	**-0.54%** (0.00)

Table 1: Interleaving and A/B Test results.

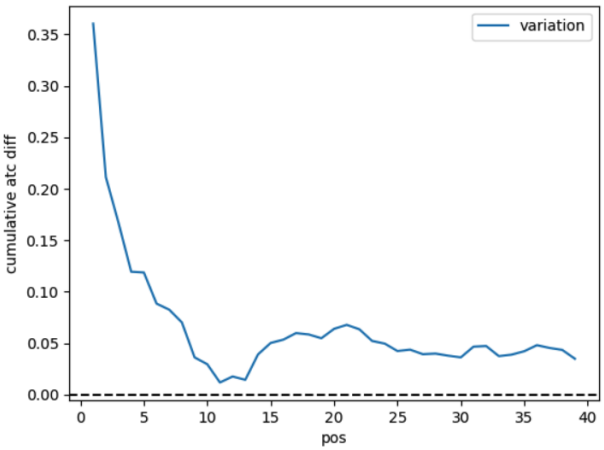

Figure 2: The interleaving ATC lift on the top-40 positions.

Method	NDCG@5 Lift (P-value)	NDCG@10 Lift (P-value)
Proposed Framework	**+4.79%** (0.00)	**+4.37%** (0.00)

Table 2: Human evaluation on the top-10 ranking results on a random sample of long-tail search traffic queries.

4.2 Manual Evaluation Results

We assessed the effectiveness of our proposed architecture through human evaluators tasked with evaluating the top-10 ranking results from both the proposed architecture and the current production system at Walmart. The item's relevance was rated using a 3-point scale (exact match, partial match, or non-relevant) based on well-defined guidelines and NDCG was computed on this rating. To receive an exact match rating, all attributes had to perfectly match. The queries were randomly selected from the tail segment of search traffic. As indicated in Table 2, our proposed architecture demonstrated a significant improvement in the relevance of long-tail queries.

5 Conclusion

In this paper, we introduced a scalable framework with a BERT-based cross encoder model deployed in the Walmart search engine. Techniques such as intermediate representation, operator fusion, and vectorization are employed to improve the model performance. The runtime implementation is detailed, including key learnings and practical tricks to minimize the response latency. Our system significantly improved the relevance of the search engine and user engagement, measured by online evaluations.

References

[1] David Carmel, Elad Haramaty, Arnon Lazerson, and Liane Lewin-Eytan. 2020. Multi-Objective Ranking Optimization for Product Search Using Stochastic Label Aggregation. In *Proceedings of The Web Conference 2020*. 373–383.

[2] Doug Downey, Susan Dumais, and Eric Horvitz. 2007. Heads and tails: studies of web search with common and rare queries. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. 847–848.

[3] Huizhong Duan, ChengXiang Zhai, Jinxing Cheng, and Abhishek Gattani. 2013. Supporting keyword search in product database: a probabilistic approach. *Proceedings of the VLDB Endowment* 6, 14 (2013), 1786–1797.

[4] Yukang Gan, Yixiao Ge, Chang Zhou, Shupeng Su, Zhouchuan Xu, Xuyuan Xu, Quanchao Hui, Xiang Chen, Yexin Wang, and Ying Shan. 2023. Binary embedding-based retrieval at Tencent. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 4056–4067.

[5] Malay Haldar, Mustafa Abdool, Prashant Ramanathan, Tao Xu, Shulin Yang, Huizhong Duan, Qing Zhang, Nick Barrow-Williams, Bradley C Turnbull, Brendan M Collins, et al. 2019. Applying deep learning to airbnb search. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1927–1935.

[6] Sebastian Hofstätter, Sophia Althammer, Michael Schröder, Mete Sertkan, and Allan Hanbury. 2020. Improving efficient neural ranking models with cross-architecture knowledge distillation. *arXiv preprint arXiv:2010.02666* (2020).

[7] Rishikesh Jha, Siddharth Subramaniyam, Ethan Benjamin, and Thrivikrama Taula. 2024. Unified Embedding Based Personalized Retrieval in Etsy Search. In *2024 IEEE International Conference on Future Machine Learning and Data Science (FMLDS)*. IEEE, 258–264.

[8] Shubhra Kanti Karmaker Santu, Parikshit Sondhi, and ChengXiang Zhai. 2017. On application of learning to rank for e-commerce search. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 475–484.

[9] Mingming Li, Chunyuan Yuan, Binbin Wang, Jingwei Zhuo, Songlin Wang, Lin Liu, and Sulong Xu. 2023. Learning query-aware embedding index for improving e-commerce dense retrieval. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 3265–3269.

[10] Sen Li, Fuyu Lv, Taiwei Jin, Guli Lin, Keping Yang, Xiaoyi Zeng, Xiao-Ming Wu, and Qianli Ma. 2021. Embedding-based product retrieval in taobao search. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 3181–3189.

[11] Juexin Lin, Sachin Yadav, Feng Liu, Nicholas Rossi, Praveen R Suram, Satya Chembolu, Prijith Chandran, Hrushikesh Mohapatra, Tony Lee, Alessandro Magnani, et al. 2024. Enhancing Relevance of Embedding-based Retrieval at Walmart. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*. 4694–4701.

[12] Qi Liu, Atul Singh, Jingbo Liu, Cun Mu, and Zheng Yan. 2024. Towards More Relevant Product Search Ranking Via Large Language Models: An Empirical Study. In *SIGIR eCom Workshop*.

[13] Bo Long, Jiang Bian, Anlei Dong, and Yi Chang. 2012. Enhancing product search by best-selling prediction in e-commerce. In *Proceedings of the 21st ACM international conference on Information and knowledge management*. 2479–2482.

[14] Eric Hsueh-Chan Lu, Wang-Chien Lee, and Vincent Shin-Mu Tseng. 2012. A Framework for Personal Mobile Commerce Pattern Mining and Prediction. *IEEE Transactions on Knowledge and Data Engineering* 24, 5 (2012), 769–782. doi:10.1109/TKDE.2011.65

[15] Alessandro Magnani, Feng Liu, Suthee Chaidaroon, Sachin Yadav, Praveen Reddy Suram, Ajit Puthenputhussery, Sijie Chen, Min Xie, Anirudh Kashi, Tony Lee, et al. 2022. Semantic retrieval at walmart. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 3495–3503.

[16] Wendy W Moe. 2003. Buying, searching, or browsing: Differentiating between online shoppers using in-store navigational clickstream. *Journal of consumer psychology* 13, 1-2 (2003), 29–39.

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

[18] Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage Re-ranking with BERT. *arXiv preprint arXiv:1901.04085* (2019).

[19] Nicholas Rossi, Juexin Lin, Feng Liu, Zhen Yang, Tony Lee, Alessandro Magnani, and Ciya Liao. 2024. Relevance filtering for embedding-based retrieval. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*. 4828–4835.

[20] Amanda Spink, Dietmar Wolfram, Major BJ Jansen, and Tefko Saracevic. 2001. Searching the web: The public and their queries. *Journal of the American society for information science and technology* 52, 3 (2001), 226–234.

[21] Ning Su, Jiyin He, Yiqun Liu, Min Zhang, and Shaoping Ma. 2018. User intent, behaviour, and perceived satisfaction in product search. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. 547–555.

[22] Andrew Trotman, Jon Degenhardt, and Surya Kallumadi. 2017. The architecture of ebay search. In *eCOM@ SIGIR*.

[23] Christophe Van Gysel, Maarten de Rijke, and Evangelos Kanoulas. 2016. Learning latent vector spaces for product search. In *Proceedings of the 25th ACM international on conference on information and knowledge management*. 165–174.

[24] Shitao Xiao, Zheng Liu, Weihao Han, Jianjin Zhang, Defu Lian, Yeyun Gong, Qi Chen, Fan Yang, Hao Sun, Yingxia Shao, et al. 2022. Distill-vq: Learning retrieval oriented vector quantization by distilling knowledge from dense embeddings. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1513–1523.

[25] Ji Yang, Xinyang Yi, Derek Zhiyuan Cheng, Lichan Hong, Yang Li, Simon Xiaoming Wang, Taibai Xu, and Ed H Chi. 2020. Mixed negative sampling for learning two-tower neural networks in recommendations. In *Companion Proceedings of the Web Conference 2020*. 441–447.

[26] Shaowei Yao, Jiwei Tan, Xi Chen, Keping Yang, Rong Xiao, Hongbo Deng, and Xiaojun Wan. 2021. Learning a product relevance model from click-through data in e-commerce. In *Proceedings of the Web Conference 2021*. 2890–2899.

[27] Jun Yu, Sunil Mohan, Duangmanee Putthividhya, and Weng-Keen Wong. 2014. Latent dirichlet allocation based diversified retrieval for e-commerce search. In *Proceedings of the 7th ACM international conference on Web search and data mining*. 463–472.

[28] Yuan Zhang, Dong Wang, and Yan Zhang. 2019. Neural IR meets graph embedding: a ranking model for product search. In *The World Wide Web Conference*. 2390–2400.

6 Author Bio

Ajit Puthenputhussery is a Principal Data Scientist in the Walmart Search team. He received his BE degree in CSE from Mumbai University and PhD degree in CS from New Jersey Institute of Technology. His research interests include large-scale machine learning with applications in search ranking and retrieval.