A Large-Scale Web Search Dataset for Federated Online Learning to Rank

Gregoriadis, Marcel; Kang, Jingwei; Pouwelse, Johan

DOI
10.1145/3746252.3761651

Publication date
2025

Document Version
Final published version

Published in
CIKM 2025 - Proceedings of the 34th ACM International Conference on Information and Knowledge Management

Citation (APA)
Gregoriadis, M., Kang, J., & Pouwelse, J. (2025). A Large-Scale Web Search Dataset for Federated Online Learning to Rank. In M. D. Jones, C. Lallemand, A. Karahanoğlu, A. Rapp, R. van den Heuvel, A. Balasubramaniam, & J. Dawson (Eds.), *CIKM 2025 - Proceedings of the 34th ACM International Conference on Information and Knowledge Management* (pp. 6387-6391). ACM. https://doi.org/10.1145/3746252.3761651

A Large-Scale Web Search Dataset for Federated Online Learning to Rank

Marcel Gregoriadis
m.gregoriadis@tudelft.nl
Delft University of Technology
Delft, The Netherlands

Jingwei Kang
j.kang@uva.nl
University of Amsterdam
Amsterdam, The Netherlands

Johan Pouwelse
j.a.pouwelse@tudelft.nl
Delft University of Technology
Delft, The Netherlands

Abstract

The centralized collection of search interaction logs for training ranking models raises significant privacy concerns. Federated Online Learning to Rank (FOLTR) offers a privacy-preserving alternative by enabling collaborative model training without sharing raw user data. However, benchmarks in FOLTR are largely based on random partitioning of classical learning-to-rank datasets, simulated user clicks, and the assumption of synchronous client participation. This oversimplifies real-world dynamics and undermines the realism of experimental results. We present AOL4FOLTR, a large-scale web search dataset with ≈ 2.6 million queries from 10,000 users. Our dataset addresses key limitations of existing benchmarks by including user identifiers, real click data, and query timestamps, enabling realistic user partitioning, behavior modeling, and asynchronous federated learning scenarios.

CCS Concepts

• **Information systems** → **Learning to rank**; **Distributed retrieval**; **Peer-to-peer retrieval**.

Keywords

Learning to rank, Federated learning, Web search

ACM Reference Format:
Marcel Gregoriadis, Jingwei Kang, and Johan Pouwelse. 2025. A Large-Scale Web Search Dataset for Federated Online Learning to Rank. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management (CIKM '25), November 10–14, 2025, Seoul, Republic of Korea.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3746252.3761651

1 Introduction

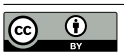
Online Learning to Rank (OLTR) is a widely used technique that aims to learn a ranker from users' interactions with search results. The centralized data collection, however, exposes users to privacy risks, as query and interaction logs reveal sensitive information, such as demographic attributes or political views [1, 3, 32]. Federated learning approaches have been explored to address user privacy concerns [16, 33, 34, 36]. In Federated Online Learning to Rank (FOLTR), clients train a local ranking model on their personal interactions with search results, and collaboratively update a global model via privacy-preserving protocols. While FOLTR presents a promising approach for developing privacy-preserving ranking models, its evaluation is constrained by the absence of publicly available datasets. In order to simulate client behavior, existing work relies on random partitioning of classical offline learning-to-rank datasets [33], and the simulation of user interactions based on click models [16, 33, 34]. This is inadequate as users have different document and click preferences [4, 35], giving rise to the non-IID problem in federated learning [42]. Clients also vary in usage frequency, i.e., the data quantity they contribute to the global model. As Wang and Zuccon [35] verified, this heterogeneity in client data poses a threat to FOLTR, as models learn less effectively. Moreover, existing work considers synchronous federated learning settings, which are inflexible and do not scale [39]. Realistically, individual client updates arrive with varying frequency and burst patterns. This further impedes model convergence through issues related to staleness [37] and fairness [23]. Despite the heterogeneous nature of real client data, and the asynchronicity of search interactions, FOLTR is commonly simulated in synchronous settings and with IID data. Accurate simulations demand a dataset with real user profiles and query timestamps. Existing datasets typically aggregate data across large user populations to preserve individual privacy [6, 28].

In this work, we present AOL4FOLTR [11], the first real-world dataset for FOLTR. It contains more than 2.5 million search interactions from 10 thousand users, including raw queries and documents, user IDs, timestamps, clicked and non-clicked documents (i.e., result lists). We base our dataset in the AOL query logs released in 2006 [2]. We scraped the original website content at query time using the Internet Archive, recovering more than 420 thousand websites. Furthermore, we used this collection of websites as a basis for reconstructing top-20 result sets for each query. Finally, we encoded query-document pairs using 103 features, following conventions from popular learning-to-rank datasets. We believe that our dataset positions itself as an important baseline for evaluating both synchronous and asynchronous FOLTR scenarios. Our dataset and code are made publicly available, along with documentation to facilitate reproducibility[1].

2 Background and Related Work

Traditional (offline) Learning to Rank (LTR) datasets include Microsoft's WEB10k/30k [28], as well as datasets from Yahoo [5] and Istella [7]. In these datasets, each query-document pair is annotated by humans with relevance scores from 0 (irrelevant) to 4 (highly relevant), which is used to optimize the ranking model. In OLTR, the ranking model is continuously updated using real-time user clicks, which serve as implicit signals.Unfortunately, there is no publicly available LTR dataset containing real user interaction data

[1] https://github.com/mg98/aol4foltr



CIKM '25, Seoul, Republic of Korea

ACM ISBN 979-8-4007-2040-6/2025/11
https://doi.org/10.1145/3746252.3761651

(i.e., click/no-click) [34]. As a result, researchers often resort to LTR datasets with simulated user interactions generated by click models [13, 15, 26]. This approach is also commonly found in the evaluation of FOLTR [16, 33, 34]. In FOLTR, clients train local ranking models on local data, and occasionally synchronize with a global model on a centralized server, where client updates are aggregated through methods like FedAvg [22]. A common challenge in federated learning is client heterogeneity, as it complicates model convergence [25, 38, 40]. Nonetheless, this issue remains a blind spot in current FOLTR research. Evaluations are typically based on IID random splits of traditional offline LTR datasets [16, 33, 34, 36]. However, this oversimplifies real-world settings where clients are heterogeneous [4, 35].

Public click datasets are rare. Prior studies have shown that "anonymized" user IDs can easily be deanonymized [1, 24]. Since then, companies have become more cautious about releasing new datasets. For example, when Microsoft released ORCAS [6], a click dataset derived from search interactions on Bing[2], they only included queries submitted by at least k users, and removed user IDs and timestamps. Researchers at Yandex [17] took a different approach by masking query terms and URLs, replacing them with numeric IDs. Obfuscation techniques like this are effective at preserving user privacy but make it difficult to extract meaningful features for LTR. The AOL dataset [27], to the best of our knowledge, remains the only publicly available source of raw query logs. This work is not the first to attempt to reconstruct search result lists from AOL query logs. Our method builds on the work of Guo et al. [14], who introduced AOL4PS. Their method was to leverage BM25 rankings over the document corpus in order to simulate result lists. This approach was later picked up in simulations of OLTR in decentralized (peer-to-peer) settings [10, 12]. We extend this method with a random offset to debias the results, as well as the inclusion of "natural candidates", which we explain in Section 3.1.1. Furthermore, we use the Internet Archive[3] to retrieve websites approximately at the time of the query logs (mid-2006). Our corpus surpasses AOL4PS by 271,392 documents, whilst also more faithfully reflecting the original state of the websites.

3 Dataset Creation

In 2006, AOL released a dataset of query logs from users of their web search engine [27]. To this date, it remains the only publicly available dataset that combines user identifiers, raw search queries, and the URL of the clicked document [20]. More recently, MacAvaney et al. [19] successfully restored the contents of the majority of documents using the Internet Archive. Their approach ensured that the recovered content approximates the state of the documents at the time of the query. Furthermore, they parsed the HTML, extracting title and body as plain text. This forms the basis for our dataset. Training and evaluation of a LTR model requires a set of candidate documents for each query, i.e., the set of search results from which the user could choose. In OLTR, this means that for every clicked document, we must also know the documents the user has decided *against*. This data is missing from the original dataset. The reconstruction of result lists forms the core of our methodology. After reconstructing result lists, we filter our dataset to only include the top 10,000 users by number of query logs.

[2]https://bing.com
[3]https://archive.org

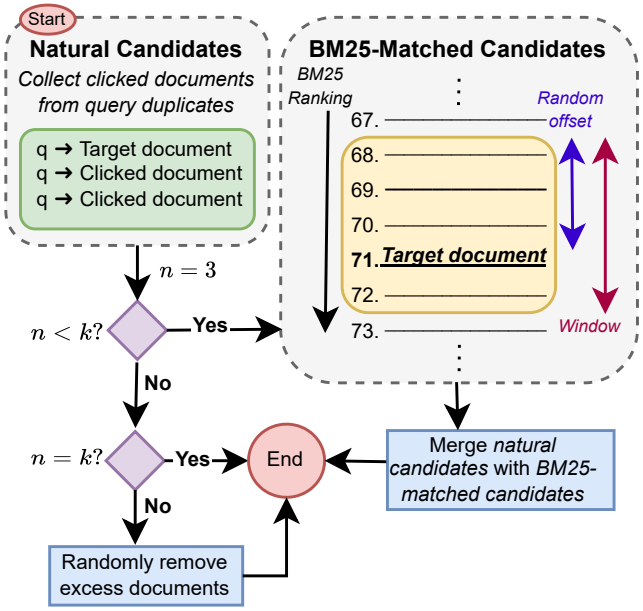


Figure 1: Example run of our top-k result list generation method. Natural candidates are extracted from activities with redundant queries, and then supplemented with BM25-matched candidates.

3.1 Reconstructing Result Lists

It is not possible to restore the result lists originally presented to users for each query. To address this limitation, we simulate top-20 result lists for each query log based on the corpus of documents in the dataset. Our approach exploits redundant queries to extract *natural candidates* and then supplements them with a BM25-based matching strategy inspired by Guo et al. [14]. We illustrate our approach in Figure 1, and detail its components in the following.

3.1.1 Natural candidates. Each *query log* comprises the raw search query, and the clicked document (or *target document*), among other metadata. By the raw search query, it is often possible to identify duplicates across multiple query logs where the same search query was used and distinct documents were clicked. For 9.3 % of queries, we could identify at least five distinct clicked documents; for 44.3 % of queries, it was at least two. We define *natural candidates* for a given query log as all documents clicked in any other query log with the same query, including the target document of the current query log. This definition is based on the assumption that if a document was clicked in one query log, it was also included in the candidate set for *all* query logs corresponding to the same query. We acknowledge that this is a strong assumption, as search engines often vary result lists based on factors such as user location, language preferences, personalization signals, and temporal dynamics. However, we believe these issues are mitigated by the fact that the query logs were collected over a relatively short period of three months and exclusively from users located in the United

States. Moreover, personalized search in 2006 was only beginning to emerge around that time, and was far less advanced than it is today [8].

3.1.2 BM25-matched candidates. We aimed for exactly $k = 20$ candidates for each query log. Usually, the number of natural candidates $n < 20$. In case of $n > 20$, we randomly remove excess documents from the candidate set, but never the target document of the query log itself. For the missing $k - n$ candidates, our approach leverages BM25 retrieval[4]. We used `pyserini` [18] to build an index of documents based on their title, body text, and URL. Based on the BM25 search utility in this library, we generated a top-1000 ranking of documents matching our query. We explicitly excluded natural candidates from this list, except for the target document. A naive approach would be to select the top-ranking items to supplement the candidate list. We observed that, most of the time, the target document is not within the top-k documents. This could create an unintended bias that the ranker might learn during training. To avoid such bias, therefore, we apply a window around the target document's position within the top-1000 ranking. We set the window size $w = k - n + 1$, to account for the missing candidates and the target document itself. The windowing approach is inspired from Guo et al. [14], who also used it to reconstruct result lists in the AOL dataset. Rather than centering the target document within the window, however, we placed it at a random offset between 0 and $w - 1$. This strategy is again intended to mitigate potential sources of bias.

3.2 Feature Selection

After reconstructing result lists, we compiled query-document feature vectors according to the standard LETOR format [28]. For each candidate document in each query log, we created a training record consisting of a query ID, a binary relevance label (1 if the candidate corresponds to the target document, 0 otherwise), and the feature vector. The feature vector encodes all information used by the ranker to make relevance decisions. Consequently, identifying the most informative features is a critical aspect of LTR [9]. For AOL4FOLTR, we followed the conventions established by classic LTR datasets. Specifically, we replicated all features from WEB30k [28] that could be derived from our data. This excludes features like PageRank or dwell time, over which we have no information. In total, we employ 103 features. Our feature selection is documented in our code repository. Furthermore, our open-source approach and provision of raw data allow researchers to experiment by creating and adding new features.

4 Dataset Analysis

Our dataset comprises 2,594,705 query logs (637,996 unique queries) from 10,000 unique users, who clicked on 428,157 distinct documents.

Data quantity. As is typical in such datasets, click activity exhibits power-law characteristics, with most clicks generated by a few users. We display this in Figure 2.

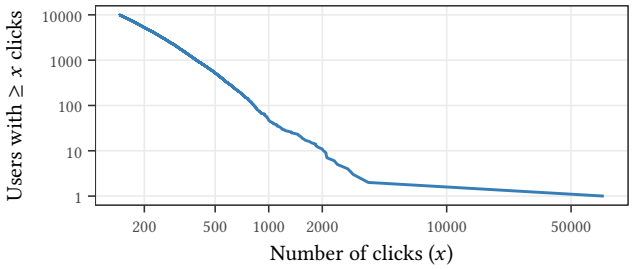


Figure 2: Cumulative number of users by minimum click count (log-log). A minority of users account for most clicks.

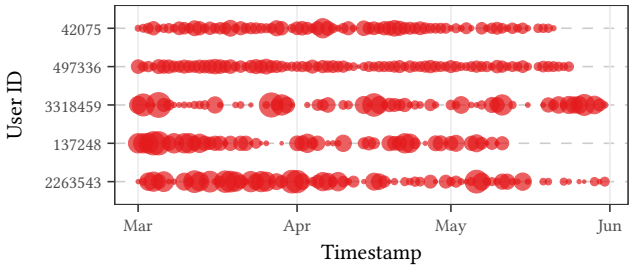


Figure 3: Temporal activity patterns for a sample of users. User activities are bursty and irregular. Each dot represents activity on a given day, with dot size indicating the volume of activity.

Temporal patterns. User activity over time is highly variable and irregular. Typically, users engage in short periods of concentrated interactions (often conceptualized as *sessions* [19]), resulting in activity bursts. In Figure 3, we visualize these bursts for the five most active users in our dataset[5]. Each dot represents user activity (i.e., clicks) on a given day, with the dot size indicating the number of clicks.

Feature heterogeneity. Local data heterogeneity is a known and well-studied problem in federated learning [41, 42]. Specifically, in the case of online learning to rank, it implies collaborative learning from clients with dissimilar click preferences with regards to the features of a search result [4, 35]. We measured this divergence by comparing the feature-wise probability distribution of clicked search results between all users. Results are shown in Figure 4. Wasserstein distance (also known as Earth Mover's Distance) is a standard algorithm to measure feature skew in non-IID federated learning [31, 41].

5 Experimental Evaluation

We evaluate our dataset for its application in both synchronous and asynchronous FOLTR. To this end, we employ 100 clients corresponding to the top 100 users by number of query logs in the dataset. As a benchmark, we construct an IID variant by randomly distributing the query logs of the 100 users across all clients. Finally, we use a temporal split, holding out the latest 20 % for testing.

[4]BM25 is a standard ranking function based on keyword matching [29, 30].

[5]User 71845 was excluded due to anomalously high activity.

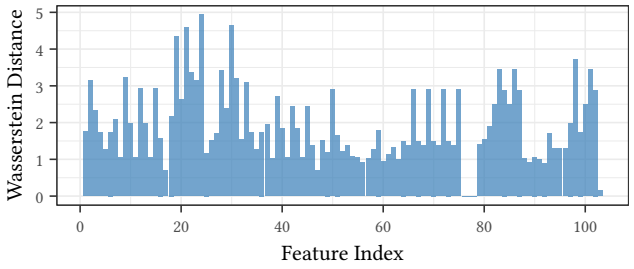


Figure 4: Feature distribution divergence of clicked documents across clients.

Our experiment uses FPDGD [33], the state-of-the-art algorithm in FOLTR. Each client c trains a local model θ^c on every personal search interaction, i.e., each click results in a local model update. After a certain number of model updates n_c, the updated model θ^c_{t+1} is sent to the server, where model updates from all $|C|$ clients are received synchronously. The server then performs a weighted averaging of all updates, as shown in Equation (1). The weight is determined by the number of queries each client has processed relative to other clients. In our experiment, we use the original implementation and hyperparameters used by the authors of FPDGD [33], including added noise for differential privacy and a constant number of queries per update $n_c = 4$.

$$\theta_{t+1} = \sum_{c=1}^{|C|} \frac{n_c}{n} \theta^c_{t+1}, \quad \text{where} \quad n = \sum_{c=1}^{|C|} n_c \tag{1}$$

Evaluations of FOLTR in the literature have focused on the setting of synchronous federated learning. In real systems, which are asynchronous, users send updates at different frequencies and at different times. Specifically, in FOLTR applications, client model updates are expected once a client has completed a batch of interactions [16, 33]. Asynchronous FOLTR, therefore, must be able to handle stale updates, as outdated gradients may not align with the current global model. To this end, we employ FedAsync [37], a standard algorithm for dealing with staleness in asynchronous federated learning. In FedAsync, received local updates are weighted according to staleness, as shown in Equation (2). Staleness is measured by the number of rounds r since the client has synchronized with the global model.

$$\theta_{t+1} = \frac{1}{1+r} \theta^c_{t+1} \cdot \left(1 - \frac{1}{1+r}\right) \theta_t \tag{2}$$

In Figure 5, we present results for both synchronous and asynchronous learning of the global model, as measured by Mean Reciprocal Rank (MRR)[6] and evaluated on the test set after each round. We stopped the experiment after 10,000 rounds. In both settings, updates of individual clients are processed in chronological order, and each client update represents a batch of the client's n_c "next" queries. The *synchronous* setting, however, does not respect the global order of client updates. This is because each round t processes the tth batch of all clients, despite timelines across clients not aligning. When a client runs out of batches, it gets skipped. In the *asynchronous* setting, a round processes a single client update, and client updates are processed in the chronological order given by the timestamp of the last query in the batch. A client's model only synchronizes with the global model after it sent its update. That is, in the synchronized setting, the client models are updated after every round; in the asynchronized setting, a client's model remains stale until the same client sends their next update.

Our experimental results indicate major performance instability in the asynchronous setting when simulating real user profiles. This effect is entirely absent from our IID benchmark, which also exhibits higher overall MRR. Further, we notice only minor differences with the IID benchmark in the synchronous setting, with convergence occurring after around 1,000 rounds. We hypothesize that incorporating features reflecting document content, beyond abstract keyword-matching metrics, may yield more pronounced differences, as suggested by the findings of Wang and Zuccon [35].

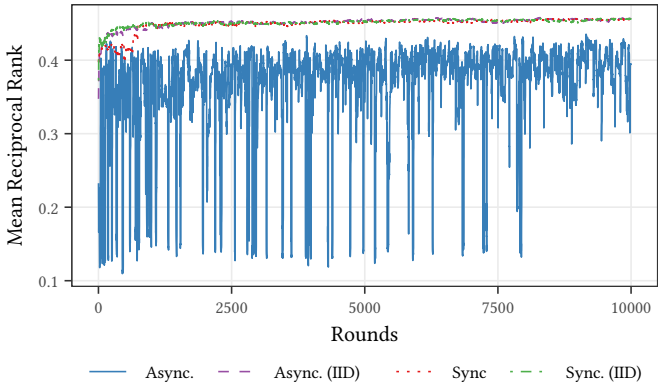


Figure 5: Experimental evaluation of our dataset with 100 clients simulating users with the most queries in our dataset.

6 Conclusion

We introduced AOL4FOLTR, a novel online learning-to-rank dataset based on real user clicks, encompassing user IDs, query timestamps, and raw query and document contents. This resource sets a new benchmark for the simulation of heterogeneous and asynchronous federated learning settings. Our experiments demonstrated the importance of simulations with real data rather than IID data, as is found in current literature. Nevertheless, we believe the true implications extend beyond the results presented here. By releasing raw queries and document contents, we empower researchers to derive new LTR features. Because of the availability of our data and methods, this resource offers broader relevance, with utility in the study of LTR feature selection, personalization techniques, and federated or decentralized information retrieval.

Acknowledgments

This work was funded by the Dutch National NWO/TKI Science Grant BLOCK.2019.004 and NWO Grant KICH3.LTP.20.006.

[6] MRR is a standard metric when evaluating ranking quality [21].

GenAI Usage Disclosure

Generative AI tools were used to assist with writing and coding for this project. All outputs were thoroughly reviewed by the authors, who take full responsibility for the content and integrity of this work.

References

[1] Michael Barbaro, Tom Zeller, and Saul Hansell. 2006. A face is exposed for AOL searcher no. 4417749. *New York Times* 9, 2008 (2006), 8.

[2] David J Brenes and Daniel Gayo-Avello. 2009. Stratified analysis of AOL query log. *Information Sciences* 179, 12 (2009), 1844–1858.

[3] Claudio Carpineto and Giovanni Romano. 2016. A Review of Ten Year Research on Query Log Privacy. *IIR* (2016).

[4] Jacopo Cecchetti, Nicola Tonellotto, and Raffaele Perego. 2024. Learning to Rank for Non Independent and Identically Distributed Datasets. In *Proceedings of the 2024 ACM SIGIR International Conference on Theory of Information Retrieval*. 71–79.

[5] Olivier Chapelle and Yi Chang. 2011. Yahoo! learning to rank challenge overview. In *Proceedings of the learning to rank challenge*. PMLR, 1–24.

[6] Nick Craswell, Daniel Campos, Bhaskar Mitra, Emine Yilmaz, and Bodo Billerbeck. 2020. Orcas: 18 million clicked query-document pairs for analyzing search. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2983–2989.

[7] Domenico Dato, Sean MacAvaney, Franco Maria Nardini, Raffaele Perego, and Nicola Tonellotto. 2022. The istella22 dataset: Bridging traditional and neural learning to rank evaluation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 3099–3107.

[8] Zhicheng Dou, Ruihua Song, and Ji-Rong Wen. 2007. A large-scale evaluation and analysis of personalized search strategies. In *Proceedings of the 16th international conference on World Wide Web*. 581–590.

[9] Xiubo Geng, Tie-Yan Liu, Tao Qin, and Hang Li. 2007. Feature selection for ranking. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. 407–414.

[10] Marcel Gregoriadis, Rowdy Chotkan, Petru Neague, and Johan Pouwelse. 2025. SwarmSearch: Decentralized Search Engine with Self-Funding Economy. In *2025 IEEE 50th Conference on Local Computer Networks (LCN)*. IEEE.

[11] Marcel Gregoriadis, Jingwei Kang, and Johan Pouwelse. 2025. *AOL4FOLTR*. https://doi.org/10.5281/zenodo.15678397

[12] Marcel Gregoriadis, Quinten Stokkink, and Johan Pouwelse. 2025. Decentralized Adaptive Ranking using Transformers. In *Proceedings of the 5th Workshop on Machine Learning and Systems*. 12–18.

[13] Artem Grotov and Maarten De Rijke. 2016. Online learning to rank for information retrieval: Sigir 2016 tutorial. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. 1215–1218.

[14] Qian Guo, Wei Chen, and Huaiyu Wan. 2021. AOL4PS: a large-scale data set for personalized search. *Data Intelligence* 3, 4 (2021), 548–567.

[15] Katja Hofmann, Anne Schuth, Shimon Whiteson, and Maarten De Rijke. 2013. Reusing historical interaction data for faster online learning to rank for IR. In *Proceedings of the sixth ACM international conference on Web search and data mining*. 183–192.

[16] Eugene Kharitonov. 2019. Federated online learning to rank with evolution strategies. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. 249–257.

[17] Eugene Kharitonov, Pavel Serdyukovpv, and Will Cukierski. 2013. Personalized Web Search Challenge. https://kaggle.com/competitions/yandex-personalized-web-search-challenge. Kaggle.

[18] Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: A Python Toolkit for Reproducible Information Retrieval Research with Sparse and Dense Representations. In *Proceedings of the 44th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2021)*. 2356–2362.

[19] Sean MacAvaney, Craig Macdonald, and Iadh Ounis. 2022. Reproducing personalised session search over the AOL query log. In *European Conference on Information Retrieval*. Springer, 627–640.

[20] Sean MacAvaney, Andrew Yates, Sergey Feldman, Doug Downey, Arman Cohan, and Nazli Goharian. 2021. Simplified Data Wrangling with ir_datasets. In *SIGIR*.

[21] Brian McFee and Gert Lanckriet. 2010. Metric learning to rank. (2010).

[22] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*. PMLR, 1273–1282.

[23] Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. 2019. Agnostic federated learning. In *International conference on machine learning*. PMLR, 4615–4625.

[24] Arvind Narayanan and Vitaly Shmatikov. 2006. How to break anonymity of the netflix prize dataset. *arXiv preprint cs/0610105* (2006).

[25] Takayuki Nishio and Ryo Yonetani. 2019. Client selection for federated learning with heterogeneous resources in mobile edge. In *ICC 2019-2019 IEEE international conference on communications (ICC)*. IEEE, 1–7.

[26] Harrie Oosterhuis and Maarten de Rijke. 2018. Differentiable unbiased online learning to rank. In *Proceedings of the 27th ACM international conference on information and knowledge management*. 1293–1302.

[27] Greg Pass, Abdur Chowdhury, and Cayley Torgeson. 2006. A picture of search. In *Proceedings of the 1st international conference on Scalable information systems*. 1–es.

[28] Tao Qin and Tie-Yan Liu. 2013. Introducing LETOR 4.0 Datasets. *CoRR* abs/1306.2597 (2013). http://arxiv.org/abs/1306.2597

[29] Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends® in Information Retrieval* 3, 4 (2009), 333–389.

[30] Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. 1995. Okapi at TREC-3. *Nist Special Publication Sp* 109 (1995), 109.

[31] David Solans, Mikko Heikkila, Andrea Vitaletti, Nicolas Kourtellis, Aris Anagnostopoulos, Ioannis Chatzigiannakis, et al. 2024. Non-IID data in Federated Learning: A Survey with Taxonomy, Metrics, Methods, Frameworks and Future Directions. *arXiv preprint arXiv:2411.12377* (2024).

[32] Samuel Sousa, Christian Guetl, and Roman Kern. 2021. Privacy in open search: A review of challenges and solutions. *arXiv preprint arXiv:2110.10720* (2021).

[33] Shuyi Wang, Bing Liu, Shengyao Zhuang, and Guido Zuccon. 2021. Effective and privacy-preserving federated online learning to rank. In *Proceedings of the 2021 ACM SIGIR international conference on theory of information retrieval*. 3–12.

[34] Shuyi Wang, Shengyao Zhuang, and Guido Zuccon. 2021. Federated online learning to rank with evolution strategies: a reproducibility study. In *European Conference on Information Retrieval*. Springer, 134–149.

[35] Shuyi Wang and Guido Zuccon. 2022. Is Non-IID Data a Threat in Federated Online Learning to Rank?. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2801–2813.

[36] Yansheng Wang, Yongxin Tong, Dingyuan Shi, and Ke Xu. 2021. An efficient approach for cross-silo federated learning to rank. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*. IEEE, 1128–1139.

[37] Cong Xie, Sanmi Koyejo, and Indranil Gupta. 2019. Asynchronous federated optimization. *arXiv preprint arXiv:1903.03934* (2019).

[38] Chenhao Xu, Youyang Qu, Yong Xiang, and Longxiang Gao. 2023. Asynchronous federated learning on heterogeneous devices: A survey. *Computer Science Review* 50 (2023), 100595.

[39] Liu Yang, Ben Tan, Vincent W Zheng, Kai Chen, and Qiang Yang. 2020. Federated recommendation systems. In *Federated Learning: Privacy and Incentive*. Springer, 225–239.

[40] Mang Ye, Xiuwen Fang, Bo Du, Pong C Yuen, and Dacheng Tao. 2023. Heterogeneous federated learning: State-of-the-art and research challenges. *Comput. Surveys* 56, 3 (2023), 1–44.

[41] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. 2018. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582* (2018).

[42] Hangyu Zhu, Jinjin Xu, Shiqing Liu, and Yaochu Jin. 2021. Federated learning on non-IID data: A survey. *Neurocomputing* 465 (2021), 371–390.