Further investigating transformer models in recommender systems

JAKUB FRĄC, Vrije Universiteit Amsterdam, Netherlands
DOMINYKAS SEPUTIS, University of Amsterdam, Netherlands
MARCELL SCHUH, Vrije Universiteit Amsterdam, Netherlands

Many advancements in recommender systems (RecSys), specifically sequential and session-based recommendations originated from Natural Language Processing (NLP). A recent NLP development is the Transformer architecture, which shows promise in RecSys applications with techniques like Bert4Rec. In 2021 NVIDIA introduced the Transformers4Rec (T4Rec) framework using the HuggingFace Transformers library to bridge NLP and the realm of RecSys. Their research focused on session-based next-click predictions on smaller e-commerce, and news datasets. We adapted T4Rec to the larger Ekstra Bladet News Recommendation Dataset, and explored model performance with different configurations, including the addition of textual embeddings, and metadata of the articles, as well as the impact of different dataset sizes.Our results showed significant improvements in model performance from the addition of vectorized textual representations and validated the assumptions regarding larger data volumes from NLP. In our comparison of model bases we found that GPT-2 outperforms XLNET. Lastly we identified several challenges when working with T4Rec.

1 INTRODUCTION

In the digital era of e-commerce and social media, personalizing the user journey is essential [19]. Although recommender systems have long existed, the recent surge in data volume and system complexity demands enhanced recommendation methods. Traditional models, such as factor models [9, 11] and neighborhood methods [24], decompose the user-item interaction matrix into d-dimensional vectors for each item and user. This transforms the recommendation problem into one of matrix completion, where missing entries are inferred using the dot product of corresponding user-item latent factors.

With the availability of more granular user behavior logging, session-based recommenders were introduced, especially for scenarios where users tend to produce more than a few interactions within their journey. This approach utilizes user behavior as a sequence, similar to the natural language processing (NLP) domain. Consequently, methods used in NLP have been adapted to session-based recommendation settings. Initially, Recurrent Neural Network (RNN) based methods were introduced [8]. Following the breakthrough in sequential data modeling by [30] with the introduction of Self-Attention based Transformers, these methods were naturally applied to session-based recommendations [26].

While transformer-based model architectures have enabled the development of higher-capacity models and improved data augmentation and training techniques for sequential recommendations, their primary applications continue to be in NLP and Computer Vision (CV). Recognizing the gap in tooling for applying transformers to recommendation use cases, [4] introduced Transformers4Rec. This open-source library, built on HuggingFace's Transformers library [31], aims to bring the advancements of NLP-based transformers to the recommender system community, making these techniques accessible for sequential and session-based recommendation tasks.

While the Transformers4Rec framework was originally evaluated in the e-commerce and news domains, where session-based recommendation is highly suitable, the authors primarily focused on simple features like tokens (item IDs), categorical, and continuous features representing item and time domains. They did not extensively utilize more expressive features, such as vectorized textual representations of news articles. Additionally, the datasets used had a relatively low number of sessions, with mean impressions per session ranging from 2.69 (ADRESSA news [7]) to 5.49 (REES46 eCommerce [6]).

To validate the relevance of the proposed framework, we apply it to the 2024 RecSys recommendation challenge. Specifically, we adapt the session-based recommendation model to the Ekstra Bladet News Recommendation Dataset (EB-NeRD) [29]. This dataset includes approximately twice as many articles and 80 times more impressions than the previously evaluated G1 news [3] dataset.

We expand the original framework by incorporating more expressive features, such as article textual embeddings, and examine the importance of features beyond the token level. Given that vector-type features have been successfully integrated into recommender systems with beneficial results [16, 34], exploring these capabilities is crucial for evaluating the framework's relevance in the current recommendation-systems landscape.

Given the larger dataset, we discuss the importance of dataset size for transformer-based model architectures by comparing model performance trained on different data subsample sizes. Additionally, we compare session-based recommender systems based on different pre-training strategies and contrast them with alternative deep neural network-based recommender architectures.

2 RELATED WORK

Our work primarily builds upon the Transformers4Rec framework introduced by [4]. Transformers4Rec excels at not just modelling, but pre-processing, and model serving by integrating with NVIDIA's Merlin stack [27], for GPU-accelerated tabular data processing with NVTabular [17] and the Triton inference server [18]. However, the framework mainly focuses on tabular data. There are also alternatives that offer comprehensive transformer-based recommendation frameworks with different feature support and extensive modularity.

SSLRec [22] is designed to provide a standardized, flexible, and comprehensive framework for assessing various Self Supervised Learning (SSL)-enhanced recommenders. Boasting a modular architecture, SSLRec enables users to effortlessly evaluate advanced models. It includes a complete suite of data augmentation and self-supervised learning tools, facilitating the creation of custom SSL recommendation models. Additionally SSLRec streamlines the training and evaluation processes, ensuring consistency and fairness across different models. The platform encompasses a wide array of state-of-the-art SSL-enhanced recommendation models applicable to diverse scenarios, thus empowering researchers to assess these advanced models and promote further progress in the field.

GPT4Rec [12] is a different recommendation framework that utilised generative models. It generates hypothetical "search queries" based on the titles of items in a user's history, and then retrieves items for recommendation by searching these queries. This framework enables model learning both user and item embeddings within the language space. To effectively capture user interests across different aspects and granularities, authors propose a multi-query generation technique utilizing beam search. These generated queries serve as interpretable representations of user interests and can be used to recommend cold-start items. Experiments conducted by authors demonstrated that multi-query generation with beam search enhances both the diversity of retrieved items and the coverage of a user's varied interests.

It is clear current state-of-the-art (SOTA) recommendation systems are not limited to tabular data and require extensive flexibility in their development. Evaluating any framework on more intricate data sizes and diverse feature scenarios is crucial for objectively assessing their capabilities.

3 METHOD

3.1 Using Transformers for next-click prediction

Transformers have shown great promise in recommendation systems by leveraging self-attention to catch complex, long-range dependencies in user-item interactions. This helps in understanding user preferences even in large-scale, sparse datasets. Unlike static models, transformers encode

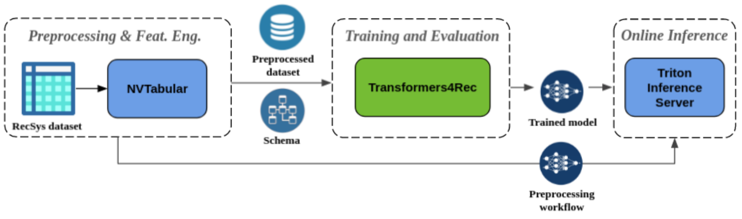

Fig. 1. Transformers4Rec pipeline overview [4].

sequential user behavior, capturing intricate patterns and contextual information, thus enabling more personalized and accurate recommendations.

Models like BERT4Rec [26], which use masked language modeling for next item prediction, demonstrate significant improvements over traditional methods like collaborative filtering and RNNs. Transformers' scalability and efficiency make them ideal for modern recommendation tasks.

Mathematically, the self-attention mechanism is defined for an input sequence $X = [x_1, x_2, \ldots, x_n]$. Attention scores are computed as $\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) V$, where d_k is the key vector dimension. This allows each sequence element to attend to all other elements, capturing dependencies regardless of their distance in the input.

In recommendation tasks, let $X_u = [e_{u_1}, e_{u_2}, \ldots, e_{u_n}]$ be the user's interaction embeddings. The transformer computes $Z_u = \text{Transformer}(X_u)$ to predict subsequent interactions or ratings. With positional encodings, layer normalization, and feed-forward networks, transformers learn nuanced user behaviors, enhancing recommendation accuracy.

3.2 Next-Click Prediction for Ranking Application

Ranking systems tailor item lists for users based on multiple factors, used in contexts like web search and decision support. While simpler methods like TF-IDF [25] or BM25 [23] exist, modern systems use advanced methods including neural networks [2, 13, 20].

These systems often involve multiple layers for precision and efficiency. Initially, a candidate list $N_{\text{candidates}}$ is retrieved using efficient methods from a larger database N_{database}, followed by complex methods to evaluate and rank these candidates.

Typically, relevancy is measured using a pairwise approach [10, 14, 33], requiring $1 \times C$ computations for each candidate C. Applying next-click item prediction for ranking is computationally expensive due to the need for softmax over all dictionary items D ($\text{Item}_{1..D}$), similar to challenges in NLP where negative sampling [15] mitigates this issue.

Transformer-based next-click models can consider in-view items with modifications, like serving as a base for a Two-Tower system [1] or masking non-candidates before applying softmax:

$$\forall \text{ item} \in \text{All Items}, \quad \text{if} \quad \text{item} \notin \text{Candidates}, \quad \text{then} \quad \text{set} \quad \text{candidate} = -\infty$$

In our experiments, we discarded the two-stage setting and evaluated the model by returning scores for top-k items within a next-click prediction setup.

3.3 Transformers4Rec Framework Overview

The Transformers4Rec framework is a comprehensive pipeline for creating transformer-based recommendation engines, optimized for NVIDIA's Merlin stack [27].

For data preprocessing, categorical features are assigned IDs and processed as one-hot vectors or embeddings. Both categorical and continuous features are normalized and combined into a single feature vector, aggregated via concatenation merge. Specifically, if each session $s^{(u)}$ is represented by n_u items, $x^{(u)} = x^{(u)}_{1:n_u}$, and I feature sequences $f^u = \{f^u_{i,1:n_u} : i \in 1, \ldots, I\}$, then:

Concatenation Merge: Concatenating the item ID feature $x^{(u)}_k$ with other features:

$$m_k = \text{concat}(x^{(u)}_k, f^{(u)}_{1,k}, \ldots, f^{(u)}_{I,k})$$

For the inclusion of pre-trained embeddings as a feature, we modify the provided preprocessing pipeline by adding textual embeddings as additional features to be considered by the model. Each item is assigned an index in a table of pre-trained embeddings. During forward propagation, the randomly initialized embedding values are replaced with the pre-trained ones. Specifically, if $E_{\text{pretrained}}$ represents the table of pre-trained embeddings, and E_{random} represents the table of randomly initialized embeddings, the embedding for item i, e_i, is given by:

$$e_i = E_{\text{pretrained}}[i], \quad \text{if pre-trained available}$$
$$e_i = E_{\text{random}}[i], \quad \text{otherwise}$$

To handle the continuous stream of user interactions, which increases dataset size and computational demands, incremental retraining is employed. Data is split into time windows T (one day for e-commerce, one hour for news). Evaluation for time window T_{i+1} uses training sessions from $[T_1, \ldots, T_i]$.

4 EXPERIMENTAL SETUP

4.1 Data

***Origin*.**
We utilize the Ekstra Bladet News Recommendation Dataset (EB-NeRD), a large-scale Danish dataset created by Ekstra Bladet to support advancements and benchmarking in news recommendation research. EB-NeRD contains over 2.3 million users and more than 380 million impression logs from Ekstra Bladet. Additionally, it includes a collection of more than 125,000 news articles, enriched with textual content features such as titles, abstracts, and bodies. This dataset provides text features in a low-resource language for context in recommender systems.

The dataset is derived from user impressions, where a list of in-view articles (candidate clicks) and actual clicks are recorded for each user. In addition, a history of user-clicked items is provided, along with impression metadata such as article read time, device type, and timestamp.

Dataset	days	items	sessions	interactions	sessions length (avg)
G1 news [3]	16	46,027	1,048,556	2,988,037	2.84
ADRESSA [7]	16	13,820	982,210	2,648,999	2.69
EB-NeRD [29]	35	78,552	12,687,583	245,408,020	19.34

Table 1. Comparison of dataset sizes previously used to evaluate the Transformers4Rec framework versus the novel dataset.

***Adaptation*.**
Since the Transformers4Rec framework requires sessions for training, with each session including at least two impressions, we had to preprocess the dataset into the correct format. We combined the impressions dataset with user history and then created artificial sessions by dividing all user impressions into smaller sessions, each containing at least two and at most 20 impressions. This resulted in 12,687,583 sessions, with an average of 19 impressions per session.

We then grouped each session by timestamp and split them into daily sessions. In total, the dataset spans 35 days of impressions, which we divided into training and validation sets with a 90%

and 10% split, respectively. Although the provided dataset offered a test split with the objective of ranking in-view articles, the correct labels were not available as we conducted our experiments during the ”RecSys Challenge 2024” competition. To enable more efficient iteration, we created our own test set. We trained our model on the first 34 days and reported the final metrics on the 35th day. While the original problem presented within the test set was to rank in-view items using user history, we modified the goal to next-click prediction to suit the newly created test set setting.

***Features used*.**
On top of the standard categorical/continuous features we also used article IDs to assign them trainable embeddings - similarly to assigning embedding for tokens in the NLP setting.

Recognizing that pre-trained embeddings have been successfully introduced as features to recommender systems [16, 34], we also extended the original Transformers4Rec framework by including pre-trained text embeddings constructed using the BERT language model [5] as an input feature.

4.2 Ablation Studies

To verify the importance of additional signals within the data, as opposed to training the model only on token-level information, we conducted several experiments. We evaluated models trained using different combinations of features: tokens (article IDs) alone (*tokens*), tokens combined with continuous and categorical features (*tokens+feats*), tokens combined with continuous and categorical features along with textual embeddings (*tokens+feats+embs*), and tokens combined with text embeddings (*tokens+embs*).

Furthermore, understanding that the performance of transformer-based models grows with the number of parameters, which commonly require a substantial amount of data for proper training [21, 28], we conducted experiments by training the model on different subsamples of the dataset (10%, 50%, and 100%) to validate whether the same assumptions from the natural language modeling domain hold true in the recommendation domain.

To identify the best training strategy for transformer architecture-based recommender systems, we also compared masked language modeling (MLM) in the encoder setting against next-token prediction in the decoder setting. Given that it is unclear if the MLM-based approach is optimal for our data, we experimented with both. Specifically, we compared two different model bases: XLNet [32] and GPT-2 [21], while keeping the rest of the modeling pipeline constant, with differences only in the head of the recommender model.

We conducted these experiments using multiple setups, utilizing NVIDIA K80 and NVIDIA A6000 GPUs. The memory in these machines ranged from 16GB to 60GB of RAM.

4.3 Evaluation Metrics

To evaluate the performance of recommender systems, we employ multiple metrics. Primarily, we use Normalized Discounted Cumulative Gain (NDCG) and recall at various top-k stages to assess the prediction of next-click. All experiments are conducted with multiple runs, and we report the mean of the results.

5 RESULTS AND DISCUSSION

5.1 Feature Impact on Model Performance

When evaluating the importance of different feature type combinations (Table 2), we find that adding article metadata features only improves metrics by approximately 2%. In contrast, the inclusion of pre-trained textual embeddings almost doubles the overall metrics. This significant improvement is because pre-trained text embeddings provide much richer information regarding article similarity and sentiment compared to randomly initialized embedding vectors.

Although adding pre-trained embeddings slightly increases the model's parameter size, the trade-off between the number of parameters and model performance is highly favorable compared to training on tabular features alone. This demonstrates the substantial impact that pre-trained textual embeddings have on enhancing model performance.

Features combination	Model size (parameters)	NDCG @ 10	Recall @ 10	NDCG @ 20	Recall @ 20
tokens	35,421,632	0.372	0.582	0.402	0.701
tokens+feats	35,427,803	0.379	0.587	0.409	0.706
tokens+emb	37,123,072	0.667	0.693	0.673	0.717
tokens+feats+emb	37,128,917	0.678	0.706	0.685	0.732

Table 2. Comparison of model performance trained on different feature combinations and model sizes. The models were trained using XLNet with 100% of the training set.

5.2 Dataset size importance studies.

To evaluate the importance of dataset size for transformer-based recommender systems, we compare different model setups on multiple dataset subsample sizes. Specifically, we use XLNet trained on *tokens+feats* and *tokens* with 10%, 50%, and 100% of the dataset subsamples. As shown in Figure 2, while the addition of more complex features results in slight improvements in metrics, using larger datasets leads to significant jumps in model performance.

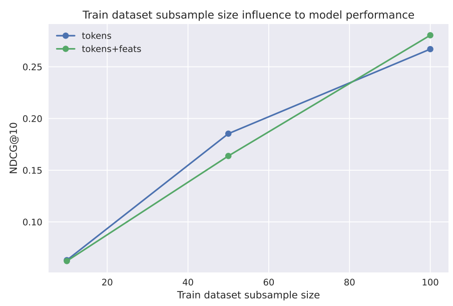

Fig. 2. Train dataset subsample size influence to model performance. The models were trained using XLNet with *tokens* and *tokens+feats* feature combinations.

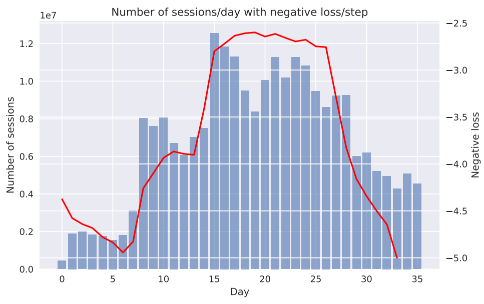

Fig. 3. Influence of sample size to the training loss of XLNet model trained on 100% of data using *tokens* as features.

This indicates that transformer-based recommender systems, much like transformer-based NLP systems, benefit considerably from larger datasets.

This relationship is further illustrated in Figure 3, where we depict the relationship between training loss per epoch and session count per epoch. As we train the model on continuous data binned into day-level timestamps, some days have less data than others. Since each day contains different items due to the nature of the news domain, this size difference impacts the model's performance.

5.3 Model's base influence to model's performance

The results in Table 3 show that GPT-2 based models outperform XLNet based models, despite having fewer overall parameters. Although XLNet surpasses GPT-2 in various NLP benchmarks [35],

Model	Model size (parameters)	NDCG @ 10	Recall @ 10	NDCG @ 20	Recall @ 20
XLNet tokens+embs	37,123,072	0.169	0.225	0.181	0.272
XLNet tokens+feats+embs	37,128,917	0.205	0.265	0.217	0.312
GPT-2 tokens+embs	36,338,944	0.259	0.333	0.273	0.392
GPT-2 tokens+feats+embs	36,344,789	0.341	0.467	0.362	0.551

Table 3. Comparison of model performance trained with different model heads (GPT-2 and XLNet). Evaluated on 10-th day of timesteps.

it is unclear whether the encoder vs. decoder-only training strategies or the model architectures are influencing this performance discrepancy. To make definitive claims, more extensive studies are required. However, based on these preliminary results, we can identify the clear advantages of GPT-2 based architectures over XLNet.

5.4 Challenges of working with Transformers4Rec library

During our studies, we encountered several challenges with the Transformers4Rec library, primarily due to insufficient documentation. While examples are provided, the overall guidance is lacking, making it difficult to translate domain knowledge from commonly used frameworks like PyTorch, Jax, or TensorFlow to NVIDIA's Merlin stack.

The data preprocessing with NVTabular posed significant challenges, particularly given its design assumption that computations occur on GPUs. Handling large datasets with limited computational resources necessitated either multiple GPUs or dataset partitioning, complicating the process and potentially affecting metric accuracy due to per-partition aggregations. Additionally, extending NVTabular and transformer models to include non-tabular features, such as pre-trained textual embeddings, proved to be complex. The library's nature as a wrapper over other frameworks further hinders rapid iteration, often requiring source code modifications.

6 CONCLUSION

In this paper, we explored the application of the Transformers4Rec framework to enhance recommender systems, specifically targeting the Ekstra Bladet News Recommendation Dataset, a significantly larger and more complex dataset compared to those used in prior work. Our experimental results validated several key insights:

- Including textual embeddings, considerably improved model performance. It nearly doubled the performance metrics compared to models trained on token-level information alone.
- Larger datasets substantively benefit transformer-based recommender systems. Our experiments indicated clear improvements in performance metrics as the training set size increased, reinforcing that Transformer architecture thrives in the large-volume-data regime.
- Contrary to the results from the original paper [4], we found that GPT2 architecture with NTP objective outperforms XLNet with MLM.
- While the Transformers4Rec framework holds promise, we identified several challenges, particularly around documentation, adaptability, and computational requirements.

Our findings suggest several directions for future work. First, more extensive studies should be conducted to confirm our results, as our evaluation was limited to a single novel dataset. Exploring different model configurations and performing comprehensive hyperparameter tuning could yield better results. Additionally, future work should consider incorporating pre-trained embeddings from domains other than text, such as vision, to fully evaluate the importance of expressive features in session-based recommender systems.

REFERENCES

[1] Fedor Borisyuk, Krishnaram Kenthapadi, David Stein, and Bo Zhao. 2016. CaSMoS: A framework for learning candidate selection models over structured queries and documents. In *Proceedings of the 22nd ACM SIGKDD International*

Conference on Knowledge Discovery and Data Mining. 441–450.

[2] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*. 89–96.

[3] Gabriel de Souza Pereira Moreira, Felipe Ferreira, and Adilson Marques Da Cunha. 2018. News session-based recommendations using deep neural networks. In *Proceedings of the 3rd workshop on deep learning for recommender systems*. 15–23.

[4] Gabriel de Souza Pereira Moreira, Sara Rabhi, Jeong Min Lee, Ronay Ak, and Even Oldridge. 2021. Transformers4rec: Bridging the gap between nlp and sequential/session-based recommendation. In *Proceedings of the 15th ACM conference on recommender systems*. 143–153.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR* abs/1810.04805 (2018). arXiv:1810.04805 http://arxiv.org/abs/1810.04805

[6] Martin Fridrich. 2023. *Machine Learning in Customer Churn Prediction*. Dissertation Thesis. Brno University of Technology, Faculty of Business and Management. Supervised by Petr Dostál.

[7] Jon Atle Gulla, Lemei Zhang, Peng Liu, Özlem Özgöbek, and Xiaomeng Su. 2017. The adressa dataset for news recommendation. In *Proceedings of the international conference on web intelligence*. 1042–1048.

[8] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939* (2015).

[9] Balázs Hidasi and Domonkos Tikk. 2012. Fast ALS-based tensor factorization for context-aware recommendation from implicit feedback. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2012, Bristol, UK, September 24-28, 2012. Proceedings, Part II 23*. Springer, 67–82.

[10] Amit Kumar Jaiswal. 2024. Towards a Theoretical Understanding of Two-Stage Recommender Systems. *arXiv preprint arXiv:2403.00802* (2024).

[11] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.

[12] Jinming Li, Wentao Zhang, Tian Wang, Guanglei Xiong, Alan Lu, and Gerard Medioni. 2023. GPT4Rec: A generative framework for personalized recommendation and user interests interpretation. *arXiv preprint arXiv:2304.03879* (2023).

[13] Weiwen Liu, Yunjia Xi, Jiarui Qin, Fei Sun, Bo Chen, Weinan Zhang, Rui Zhang, and Ruiming Tang. 2022. Neural re-ranking in multi-stage recommender systems: A review. *arXiv preprint arXiv:2202.06602* (2022).

[14] Zhuang Liu, Yunpu Ma, Yuanxin Ouyang, and Zhang Xiong. 2021. Contrastive learning for recommender system. *arXiv preprint arXiv:2101.01317* (2021).

[15] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[16] Cataldo Musto, Giovanni Semeraro, Marco De Gemmis, Pasquale Lops, et al. 2015. Word Embedding Techniques for Content-based Recommender Systems: An Empirical Evaluation. *Recsys posters* 1441 (2015).

[17] NVIDIA. 2023. NVIDIA Merlin NVTabular. https://developer.nvidia.com/nvidia-merlin/nvtabular Accessed: 2023-10-03.

[18] NVIDIA. 2023. NVIDIA Triton Inference Server. https://developer.nvidia.com/triton-inference-server Accessed: 2023-10-03.

[19] Yuanzhe Peng. 2022. A survey on modern recommendation system based on big data. *arXiv preprint arXiv:2206.02631* (2022).

[20] Alberto Purpura, Karolina Buchner, Gianmaria Silvello, and Gian Antonio Susto. 2021. Neural feature selection for learning to rank. In *Advances in Information Retrieval: 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28–April 1, 2021, Proceedings, Part II 43*. Springer, 342–349.

[21] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1, 8 (2019), 9.

[22] Xubin Ren, Lianghao Xia, Yuhao Yang, Wei Wei, Tianle Wang, Xuheng Cai, and Chao Huang. 2023. SSLRec: A Self-Supervised Learning Library for Recommendation. *arXiv preprint arXiv:2308.05697* (2023).

[23] 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.

[24] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*. 285–295.

[25] Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation* 28, 1 (1972), 11–21.

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

[27] NVIDIA Merlin Team. 2023. NVIDIA Merlin. https://github.com/NVIDIA-Merlin/Merlin GitHub repository.

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

[29] Unknown. 2023. RecSys Dataset. https://recsys.eb.dk/dataset/ Accessed: 2023-10-03.

[30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[31] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771* (2019).

[32] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems* 32 (2019).

[33] Yantao Yu, Weipeng Wang, Zhoutian Feng, and Daiyue Xue. 2021. A dual augmented two-tower model for online large-scale recommendation. *DLP-KDD* (2021).

[34] Xiangyu Zhao, Maolin Wang, Xinjian Zhao, Jiansheng Li, Shucheng Zhou, Dawei Yin, Qing Li, Jiliang Tang, and Ruocheng Guo. 2023. Embedding in Recommender Systems: A Survey. *arXiv preprint arXiv:2310.18608* (2023).

[35] Xuhui Zhou, Yue Zhang, Leyang Cui, and Dandan Huang. 2020. Evaluating commonsense in pre-trained language models. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 9733–9740.

A DATASET SIZE INFLUENCE ON MODEL'S PERFORMANCE

	Dataset subsample size		
Features combination	*10%*	*50%*	*100%*
tokens	0.063	0.185	0.267
tokens+feats	0.062	0.164	0.281

Table 4. Dataset size influence on model's performance. The model was trained using XLNet. Reported metric is NDCG@10.