GPT4Rec: A Generative Framework for Personalized Recommendation and User Interests Interpretation

Jinming Li[1], Wentao Zhang[2], Tian Wang[2], Guanglei Xiong[2], Alan Lu[2] and Gerard Medioni[2]

[1]University of Michigan, Ann Arbor
[2]Amazon, United States

Abstract

Recent advancements in Natural Language Processing (NLP) have led to the development of NLP-based recommender systems that have shown superior performance. However, current models commonly treat items as mere IDs and adopt discriminative modeling, resulting in limitations of (1) fully leveraging the content information of items and the language modeling capabilities of NLP models; (2) interpreting user interests to improve relevance and diversity; and (3) adapting practical circumstances such as growing item inventories. To address these limitations, we present GPT4Rec, a novel and flexible generative framework inspired by search engines. It first generates hypothetical "search queries" given item titles in a user's history, and then retrieves items for recommendation by searching these queries. The framework overcomes previous limitations by learning both user and item embeddings in the language space. To well-capture user interests with different aspects and granularity for improving relevance and diversity, we propose a multi-query generation technique with beam search. The generated queries naturally serve as interpretable representations of user interests and can be searched to recommend cold-start items. With GPT-2 language model and BM25 search engine, our framework outperforms state-of-the-art methods by 75.7% and 22.2% in Recall@K on two public datasets. Experiments further revealed that multi-query generation with beam search improves both the diversity of retrieved items and the coverage of a user's multi-interests. The adaptiveness and interpretability of generated queries are discussed with qualitative case studies.

Keywords

personalized recommendation, user interests modeling, generative language models, query generation, searching

1. Introduction

Recommender systems are information filtering systems that provide personalized suggestions for items that are relevant to a particular user, ubiquitously adopted in applications including E-commerce and social media services [1, 2, 3, 4]. With the development of Natural Language Processing (NLP), many NLP-based models are proposed for personalized recommendation by modeling the user-item interaction sequences [5, 6, 7]. The recent success of Large Language Models (LLM, [8, 9, 10, 11]) that achieved superior performances across various NLP tasks or diagolue (ChatGPT [12]), have spurred the research of LLM-based recommender systems[13, 14,

eCom'23: ACM SIGIR Workshop on eCommerce, July 27, 2023, Taipei, Taiwan
lijinmin@umich.com (J. Li); wentazha@amazon.com (W. Zhang); wangtan@amazon.com (T. Wang); glx@amazon.com (G. Xiong); alalu@amazon.com (A. Lu); medioni@amazon.com (G. Medioni)

15, 16, 17, 18]. In particular, BERT4Rec [13] adopted the bi-directional self-attention structure of BERT [9] and outperformed other NLP-based models and sequential models[19, 5, 15].

Despite improvement in model performances, existing NLP-based models [13, 14, 15, 16, 17, 18] typically treat items as mere IDs and adopt discriminative modeling, which have the following limitations: First, these models are incapable of fully leveraging the content information of items and the language modeling ability of NLP models. Second, these models are unable to accommodate changing and growing item inventories, both of which are important practical issues [20, 21]. Moreover, discriminative models are challenging to interpret user interests, which is essential to improve diversity and quality of recommendation [22, 23, 24, 25].

To address these limitations, we propose GPT4Rec, a novel and flexible generative framework for personalized recommendation that simultaneously offers interpretations of user interests. Our framework is inspired by search engines: GPT4Rec first generates hypothetical "search queries" with a generative language model, which takes the item titles in a user's history combined with a generation prompt as model input. Then, GPT4Rec retrieves items for recommendation by searching the generated queries with a search engine. The key component of GPT4Rec is a powerful generative language model that learns both user and item embeddings in the language space, allowing us to utilize the semantic information in item titles and capture user's diverse interests. To decode user's multi-interests and improve recommendation diversity, we adopt the multi-query beam search technique in query generation. These queries are human-understandable and hold standalone value for interpreting user interests. In addition, searching queries that represent user interests for recommendation can naturally solve the item cold-start problem and changing item inventory issue. Last but not least, GPT4Rec is favorable in practice as it is flexible to accommodate more advanced LLMs or search engines to improve performance.

In summary, our contributions include: (1) We propose GPT4Rec, a novel and flexible generative framework that regard recommendation task as query generation + searching; (2) We adopt the multi-query beam search strategy to produce diverse and interpretable user interests representations; (3) We conduct experiments on two public datasets, demonstrating substantial improvement of Recall@K against state-of-the-art methods; (4) With both quantitative and qualitative analysis, we investigate and show that the number of generated queries improves the diversity of retrieved items as well as the coverage of users' multi-interests.

2. Methodology

The architecture of the proposed framework is illustrated in Figure 1. Given a user's item interaction sequence, GPT4Rec formats the item titles with a prompt and uses a generative language model to learn both item and user embeddings in the language space. The model then generates multiple queries that represent user's interests, which will be fed to a search engine to retrieve items for recommendation. In this paper we choose the GPT-2 language model and BM25 search engine, while the framework is flexible to incorporate more advanced models.

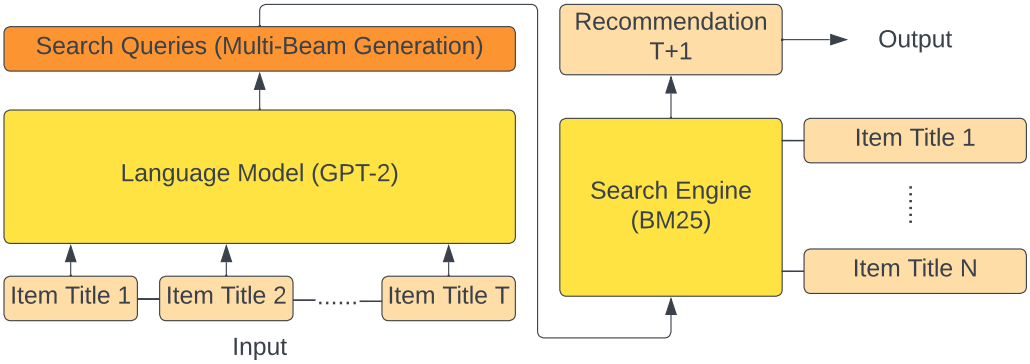

Figure 1: The architecture of the GPT4Rec framework.

2.1. Query Generation with the Language Model

The first component of GPT4Rec is a generative language model, whose goal is to learn the user representation in the language space from the item interaction sequence, and then generates multiple queries that represent user interests. Fine-tuning the chosen GPT-2 model, which has 117M parameters with sophisticated transformer-type architecture and is pre-trained on enormous language corpus, serves our purpose of capturing user interests and item content information. Through experiments we decide to use the following prompt to format model input:

Previously, the customer has bought:
<ITEM TITLE 1>. <ITEM TITLE 2>...
In the future, the customer wants to buy

which contains semantic information in item titles and is denoted as W^u for each user u. GPT-2 learns user representation from W_u and is then able to generate queries based on the conditional distribution $P(\cdot|W_u)$ in a sequential way.

In order to well-represent users' diverse interests and increase the diversity of recommendation results, we propose to generate not single but multiple queries with the beam search technique [19, 26]. Given beam size m, generation score function $S(\cdot)$, and the candidate queries $Q^l = (q_1^l, q_2^l, ...q_m^l)$ with length l, the beam search algorithm updates Q^{l+1} to be the top-m queries of length $l+1$ that maximize $\mathrm{S}(W^u, q), \quad q \in \{|q| = l+1, q_{[1:l]} \in Q^l\}$. Most importantly, such beam search strategy generates user interests representation that are diverse in different aspects and granularity.

2.2. Item Retrieval with the Search Engine

The second component of the proposed framework is a search engine that functions as a "discriminator". It takes each generated query as input and retrieves the most relevant items in

the inventory as output with matching score functions. The matching score functions measure the similarity on the language space, playing a similar role of inner-product similarity in vector-embedding methods [1, 2]. We choose to adopt BM25 matching score function [27], as it is one of the most widely used baseline search engines that accounts for the term frequency saturation and the document length with two corresponding parameters k_1 and b.

Let K represent the total number of recommendation items and m represent the number of generated queries, we propose a ranking-based strategy to combine searching results of each query. It first retrieves top-K/m items from searching results with the query having highest generation score, then sequentially adds K/m non-repeated items from the rest of queries by their score rankings. The proposed strategy is able to balance both relevance and diversity of the retrieved items.

2.3. Training Strategy

We propose a flexible two-step training procedure that optimizes the language model and the search engine separately, which is favorable in practice for model iteration. Given the item interaction sequence $i_1, i_2, ..., i_T$ of each user u, we take the first $T-1$ item titles in the prompt shown in Section 2.1, then concatenate it with the title of last item i_T to form training corpus to fine-tune the pre-trained GPT-2 model. This strategy is motivated by the contrastive learning literature [28], with the underlying idea that *the most fine-grained and accurate search query is the target item title itself.* After the language model is trained, we optimize the BM25 parameters by grid-searching parameters k_1 and b that gives the best performance when searching the generated queries to retrieve target items.

3. Experiments

3.1. Experiment Setup

3.1.1. Data

We conduct experiments on two public datasets, 5-core Amazon Review data[29] in categories Beauty and Electronics. Items with missing or noisy titles (more than 400 characters) are discarded. Item interaction sequence of each user is deduplicated and truncated at the maximum length of 15. Descriptive statistics of the preprocessed datasets are presented in Table 2. Following the next-item prediction task setup [13], users sequences are split into training, validation, and test sets by 0.8/0.1/0.1 proportions and the last item in each test sequence is treated as prediction target.

3.1.2. Evaluation Metrics

Recall@K is the primary metric of interests for next-item prediction, which measures whether the top-K recommended items contains the target item, averaged over all users. Besides Recall@K, we are also interested in:
Diversity@K [25], which measures dissimilarity if items as

Table 1
Overall performance of baseline methods and the proposed framework with different number of generated queries. The best performance is highlighted in bold font and the best baseline results is underlined.

Dataset	Recall@K	FM-BPR	ContentRec	YouTubeDNN	BERT4Rec	GPT4Rec			
						5 Queries	10 Queries	20 Queries	40 Queries
Beauty	K=5	0.0356	0.0254	<u>0.0376</u>	0.0355	**0.0653**	—	—	—
	K=10	0.0499	0.0440	<u>0.0549</u>	0.0513	0.0810	**0.1036**	—	—
	K=20	0.0716	0.0644	0.0753	<u>0.0816</u>	0.1027	0.1252	**0.1454**	—
	K=40	0.1040	0.0952	0.1066	<u>0.1161</u>	0.1297	0.1522	0.1743	**0.2040**
Electronics	K=5	0.0345	0.0241	0.0352	<u>0.0362</u>	**0.0434**	—	—	—
	K=10	0.0387	0.0307	0.0435	<u>0.0451</u>	0.0480	**0.0545**	—	—
	K=20	0.0441	0.0391	0.0539	<u>0.0573</u>	0.0524	0.0607	**0.0705**	—
	K=40	0.0505	0.0494	0.0684	<u>0.0751</u>	0.0574	0.0672	0.0788	**0.0918**

Table 2
Dataset statistics.

Name	#User	#Item	#Interaction	Ave. Length	Meta Info.
Beauty	22,254	11,778	190,726	7.439	Cate.
Electronics	728,719	159,456	6,724,382	7.797	Cate., Brand

$$\text{Diversity@}K = \text{Average}\left[1 - \frac{\sum_{i_1 \neq i_2} \text{Sim}(i_1, i_2)}{K(K-1)}\right],$$

where we adopt the *Jaccard similarity* [1] on item category or brand.
Coverage@K, which measures how well the recommended items R cover the user sequence U in terms of category or brand:

$$\text{Coverage@}K = \text{Average}\left[\frac{\left|\left(\bigcup_{i \in R} Cate_i\right) \cap \left(\bigcup_{i \in U} Cate_i\right)\right|}{\left|\bigcup_{i \in U} Cate_i\right|}\right],$$

which is a more reasonable metric for measuring the coverage of user's multiple interests as Diversity@K would favor totally random/irrelevant recommendation.

3.1.3. Baseline Methods

FM-BPR [3] is a Factorization Machine model with Bayesian Personalized Ranking criteria, which learns the collaborative information from user-item interaction matrix.
ContentRec is a content-based model widely used in industry, which learns bad-of-words embeddings from item titles and mean-pooling over user-interacted items for user embeddings.
YouTubeDNN [2] is one of the most popular deep-learning model used in industry, which adopts a neural network to learn user/item embeddings.
BERT4Rec [13] is a state-of-the-art BERT-based model, with bidirectional self-attention architecture and masked training strategy.

[1] https://en.wikipedia.org/wiki/Jaccard_index

3.1.4. Implementation Details

We implement the GPT-2 model released by HuggingFace [30] with 117M parameters and fine-tune the model for 20 epochs using the Adam optimizer with weight decay. We set the learning rate to be 0.0001 and initialize the optimizer with 2000 warm-up steps. BM25 is optimized over $k1 \in [0, 3], \quad b \in (0, 1)$ with previously discussed method. ContentRec is implemented using StarSpace [6] to learn item title embeddings and the other baseline methods are implemented with public available Github repositories [2]. For all the baseline methods, we select the optimal embedding dimensions from $64, 128, 256$ on the validation set while setting the rest of hyperparameters following the authors' suggestions in their original papers.

3.2. Quantitative Analysis

3.2.1. Overall Performance

Table 1 summarizes the Recall@K of GPT4Rec with varying numbers of generated queries compared to baseline methods on two datasets. Our proposed framework outperforms all baseline methods on both datasets, achieving a **75.7%** relative improvement on the smaller-scaled Beauty dataset in Recall@40, and **22.2%** on the larger-scaled Electronics dataset.

The comparison with baseline methods suggests that both item content information and modern language modeling are key ingredients for achieving superior performance. One the one hand, while BERT4Rec has the best performance among the baseline methods by leveraging modern language modeling techniques, it fails to fully utilize the item content information by treating items as IDs. On the other hand, ContentRec's use of item content information with bag-of-words embeddings and mean-pooling modeling is insufficient for achieving comparable performance.

3.2.2. Advantages of Multi-query Generation

When looking at the lower-triangle numbers of both datasets shown in Table 1, monotone increasing trends can be found in row-wise, column-wise, and diagonal-wise directions. These trends indicate that the multi-query beam search generation strategy greatly improves the relevance of recommended items. In particular, generating K queries and retrieving one item per query yields the best performance of Recall@K. This finding suggests that *each query contains enough detail to retrieve a relevant item.*

We further investigate the diversity and coverage of user interests in recommended items, as well as their relationship to the multi-query generation strategy. As reported in Table 3, for both Diversity@K and Coverage@K the best performance are achieved by generating K queries and retrieve one item per query, which aligns our finding for Recall@K in Table 1. Similar monotone increasing trends indicate that multi-query generation strategy produces more comprehensive user interests representations. On the other hand, as we see in columns

[2]LightFM: https://github.com/lyst/lightfm
StarSpace: https://github.com/facebookresearch/StarSpace
DeepMatch: https://github.com/shenweichen/DeepMatch
RecBole: https://github.com/RUCAIBox/RecBole

Table 3
Diversity and coverage of user interests versus the number of generated queries. Highest values with regard to category/brand information are highlighted in bold font for two datasets.

Dataset		Beauty (Category)				Electronics (Category)				Electronics (Brand)			
Number of Queries		5	10	20	40	5	10	20	40	5	10	20	40
Diversity@K	K=5	**0.679**	—	—	—	**0.671**	—	—	—	**0.534**	—	—	—
	K=10	0.654	**0.716**	—	—	0.617	**0.733**	—	—	0.529	**0.643**	—	—
	K=20	0.659	0.706	**0.749**	—	0.605	0.703	**0.778**	—	0.559	0.645	**0.717**	—
	K=40	0.679	0.715	0.749	**0.783**	0.601	0.696	0.762	**0.811**	0.604	0.669	0.724	**0.767**
Coverage@K	K=5	**0.417**	—	—	—	**0.321**	—	—	—	**0.173**	—	—	—
	K=10	0.472	**0.547**	—	—	0.340	**0.425**	—	—	0.177	**0.239**	—	—
	K=20	0.535	0.602	**0.674**	—	0.364	0.447	**0.537**	—	0.184	0.245	**0.317**	—
	K=40	0.614	0.669	0.726	**0.787**	0.389	0.474	0.562	**0.653**	0.197	0.255	0.324	**0.403**

Figure 2: Example of a test user with diverse interests in Beauty products, with target item circled in red.

of Diversity@K, increasing K without increasing the number of queries will not improve diversity. This shows searching doesn't assist in improving diversity.

3.3. Qualitative Analysis

In this section, we explore the effectiveness of generated queries in capturing user interests and their usefulness in interpreting user interests through case studies. We illustrate two test user sequences and model outputs in Figure 2 and Figure 3.

In the first example, the test user has diverse interests in multiple categories and brands of Beauty products. GPT4Rec in this case is able to generate diverse queries that not only cover some products or brands in user history, but also generates relevant items that never appeared in user history such as "makeup palette". This demonstrates that GPT4Rec is able to capture the associations among items based on user-item interaction and item title semantic information. In the second example, the test user has very a specific interest in Logitech wireless mouses. GPT4Rec manages to capture this feature and all generated queries are concentrated on the same brand, same categories but with different product details.

We have demonstrated the effectiveness of GPT4Rec in generating queries that capture user interests across various aspects and granularity levels, as shown in two case studies above. These queries directly serves the purpose of interpreting user interests. Moreover, comparison

Figure 3: Example of a test user with specific interests in Electronics products, with target item circled in red.

of the two examples indicates the diversity level of generated queries are adaptive to the level in user sequences, which further helps understand their behavior.

4. Conclusion

In this work, we propose GPT4Rec, a novel generative framework that produces personalized recommendation and interpretable user interests representation simultaneously. Leveraging advanced language models and item content information, GPT4Rec achieves superior performance and naturally solves practical issues such as item cold-start problem. The proposed multi-query beam search technique generates user interests representation with different aspects and granularity, yieldings improvement in relevance and diversity of recommendation results. Our framework is flexible to incorporate more advanced generative language models or search engines, as well as better generation and retrieval strategies, which are interesting directions for future exploration.

References

[1] S. Rendle, Factorization machines, in: 2010 IEEE International conference on data mining, IEEE, 2010, pp. 995–1000.

[2] P. Covington, J. Adams, E. Sargin, Deep neural networks for youtube recommendations, in: Proceedings of the 10th ACM conference on recommender systems, 2016, pp. 191–198.

[3] M. Kula, Metadata embeddings for user and item cold-start recommendations, in: T. Bogers, M. Koolen (Eds.), Proceedings of the 2nd Workshop on New Trends on Content-Based Recommender Systems co-located with 9th ACM Conference on Recommender Systems (RecSys 2015), Vienna, Austria, September 16-20, 2015., volume 1448 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2015, pp. 14–21. URL: http://ceur-ws.org/Vol-1448/paper4.pdf.

[4] A. Pal, C. Eksombatchai, Y. Zhou, B. Zhao, C. Rosenberg, J. Leskovec, Pinnersage: Multi-modal user embedding framework for recommendations at pinterest, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 2311–2320.

[5] D. Jannach, M. Ludewig, When recurrent neural networks meet the neighborhood for session-based recommendation, in: Proceedings of the eleventh ACM conference on recommender systems, 2017, pp. 306–310.
[6] L. Wu, A. Fisch, S. Chopra, K. Adams, A. Bordes, J. Weston, Starspace: Embed all the things!, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 32, 2018.
[7] K. Zhou, H. Wang, W. X. Zhao, Y. Zhu, S. Wang, F. Zhang, Z. Wang, J.-R. Wen, S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization, in: Proceedings of the 29th ACM international conference on information & knowledge management, 2020, pp. 1893–1902.
[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).
[9] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805 (2018).
[10] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al., Language models are unsupervised multitask learners, OpenAI blog 1 (2019) 9.
[11] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., Language models are few-shot learners, Advances in neural information processing systems 33 (2020) 1877–1901.
[12] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, et al., Improving language understanding by generative pre-training (2018).
[13] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, P. Jiang, Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer, in: Proceedings of the 28th ACM international conference on information and knowledge management, 2019, pp. 1441–1450.
[14] S. Geng, S. Liu, Z. Fu, Y. Ge, Y. Zhang, Recommendation as language processing (rlp): A unified pretrain, personalized prompt & predict paradigm (p5), arXiv preprint arXiv:2203.13366 (2022).
[15] W.-C. Kang, J. McAuley, Self-attentive sequential recommendation, in: 2018 IEEE international conference on data mining (ICDM), IEEE, 2018, pp. 197–206.
[16] D. Sileo, W. Vossen, R. Raymaekers, Zero-shot recommendation as language modeling, in: Advances in Information Retrieval: 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10–14, 2022, Proceedings, Part II, Springer, 2022, pp. 223–230.
[17] Y. Zhang, H. Ding, Z. Shui, Y. Ma, J. Zou, A. Deoras, H. Wang, Language models as recommender systems: Evaluations and limitations (2021).
[18] G. Penha, C. Hauff, What does bert know about books, movies and music? probing bert for conversational recommendation, in: Fourteenth ACM Conference on Recommender Systems, 2020, pp. 388–397.
[19] M. Ranzato, S. Chopra, M. Auli, W. Zaremba, Sequence level training with recurrent neural networks, arXiv preprint arXiv:1511.06732 (2015).
[20] X. N. Lam, T. Vu, T. D. Le, A. D. Duong, Addressing cold-start problem in recommendation systems, in: Proceedings of the 2nd international conference on Ubiquitous information management and communication, 2008, pp. 208–211.

[21] A. I. Schein, A. Popescul, L. H. Ungar, D. M. Pennock, Methods and metrics for cold-start recommendations, in: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval, 2002, pp. 253–260.
[22] Y. Zhang, X. Chen, et al., Explainable recommendation: A survey and new perspectives, Foundations and Trends® in Information Retrieval 14 (2020) 1–101.
[23] Y. Cen, J. Zhang, X. Zou, C. Zhou, H. Yang, J. Tang, Controllable multi-interest framework for recommendation, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 2942–2951.
[24] C. Li, Z. Liu, M. Wu, Y. Xu, H. Zhao, P. Huang, G. Kang, Q. Chen, W. Li, D. L. Lee, Multi-interest network with dynamic routing for recommendation at tmall, in: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, 2019, pp. 2615–2623.
[25] M. Chen, Y. Wang, C. Xu, Y. Le, M. Sharma, L. Richardson, S.-L. Wu, E. Chi, Values of user exploration in recommender systems, in: Fifteenth ACM Conference on Recommender Systems, 2021, pp. 85–95.
[26] A. K. Vijayakumar, M. Cogswell, R. R. Selvaraju, Q. Sun, S. Lee, D. Crandall, D. Batra, Diverse beam search: Decoding diverse solutions from neural sequence models, arXiv preprint arXiv:1610.02424 (2016).
[27] S. Robertson, H. Zaragoza, et al., The probabilistic relevance framework: Bm25 and beyond, Foundations and Trends® in Information Retrieval 3 (2009) 333–389.
[28] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, F. Makedon, A survey on contrastive self-supervised learning, Technologies 9 (2020) 2.
[29] R. He, J. McAuley, Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering, in: proceedings of the 25th international conference on world wide web, 2016, pp. 507–517.
[30] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, et al., Huggingface's transformers: State-of-the-art natural language processing, arXiv preprint arXiv:1910.03771 (2019).