A Brief History of Recommender Systems

Zhenhua Dong¹, Zhe Wang²,³, Jun Xu⁴, Ruiming Tang¹, Jirong Wen⁴
¹ Huawei Noah’s Ark Lab; ² ByteDance Ltd; ³ Tsinghua University
⁴ Gaoling School of Artificial Intelligence, Renmin University of China, China

{dongzhenhua,tangruiming}@huawei.com; zhe.wangz@bytedance.com; zhewang06@tsinghua.edu.cn;
junxu,jrwen}@ruc.edu.cn

ABSTRACT

Soon after the invention of the World Wide Web, the recommender system emerged and related technologies have been extensively studied and applied by both academia and industry. Currently, recommender system has become one of the most successful web applications, serving billions of people in each day through recommending different kinds of contents, including news feeds, videos, e-commerce products, music, movies, books, games, friends, jobs etc. These successful stories have proved that recommender system can transfer big data to high values. This article briefly reviews the history of web recommender systems, mainly from two aspects: (1) recommendation models, (2) architectures of typical recommender systems. We hope the brief review can help us to know the dots about the progress of web recommender systems, and the dots will somehow connect in the future, which inspires us to build more advanced recommendation services for changing the world better.

CCS CONCEPTS

• Information systems → Recommender systems.

KEYWORDS

recommender system, recommendation model, architecture

ACM Reference Format:

Zhenhua Dong¹, Zhe Wang²,³, Jun Xu⁴, Ruiming Tang¹, Jirong Wen⁴. 2022. A Brief History of Recommender Systems. In Proceedings of ACM Conference (Conference'17). ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/nmnnmn.nnmmnn

1 INTRODUCTION

In 1992, Belkin and Croft [5] analyzed and compared information filtering and information retrieval, where the information retrieval is the fundamental technology of search engine, and the recommender system is mainly based on the technology of information filtering. In the same year, Goldberg et al. [25] proposed Tapestry system which is the first information filtering system based on collaborative filtering through human evaluation. Inspired by the study, some researchers from Massachusetts Institute of Technology (MIT) and University of Minnesota (UMN) developed the news recommendation service, named GroupLens [64], whose key component is a user-user collaborative filtering model. Prof. John Riedl founded a research lab at UMN, also named GroupLens, which is the pioneer of recommender system studies. For music and video, similar recommendation technologies have been applied by the Ringo system [75] and Video Recommender [35], respectively. Along with the emergence of e-commerce, the industry realized the business value of recommendation. Net Perceptions [89], the first company that focuses on offering the marketing recommender engine, was found in 1996. The customers include Amazon, Best Buy, and JC Penney etc. Schafer et al. [72] explained how recommender systems help E-commerce sites to increase sales through analyzing six web sites, from three aspects such as interfaces, recommendation model, and user inputs. Since then, the academic studies and industrial practical applications became the two wheels of the progress of recommender system technologies. In the fall of 1997, the GroupLens research lab launched the MovieLens [30] project and trained the first version of the recommender model with the EachMovie dataset. After that, several MovieLens datasets were continuously released during 1998 to 2019, and became one of the most popular datasets for recommendation studies.

From the perspective of recommendation models, collaborative filtering technologies dominated the recommender system applications and studies before 2005, such as user-user collaborative filtering [7, 34], item-item collaborative filtering [50, 69] and Singular Value Decomposition (SVD) based collaborative filtering [71]. Inspired by the Netflix Prize during 2006 to 2009, the matrix factorization models had been extensively studied [41, 42]. In the same period, some researchers began to propose informal arguments that the evaluation of the recommender systems should move from the conventional accuracy metrics to the user-centric evaluation [55]. Because of the fast development of basic research and commercial applications in recommender system, the community decided to hold the first ACM RecSys Conference [40] at UMN in 2007. Currently, ACM RecSys has become one of the most important annual academic conference that focuses on the study of recommender system. In the same year, Richardson et al. [67] presented a logistic regression (LR) model that achieved a 30% reduction in term of the error in click-through rate estimation. Since then, LR models were continuously improved from different aspects, including optimizing methods [26, 54], automatic feature engineering [33] etc. In 2010, Rendle [63] proposed the Factorization Machines (FMs) which combines the advantages of Support Vector Machines (SVM) and factorization models. Based on FMs, Juan et al. [38] proposed Field-aware Factorization Machines (FFMs) which considers the fields of features when modeling the weight of each feature pair. In the meantime, more and more studies pay attention to the user experiences in recommender system. Pu et al. [59, 60]
proposed a user-centric evaluation framework for recommender systems; Konstan and Riedl [39] appeal the evolution of the recommender system study from research concentrated purely on algorithms to research concentrated on user experience.

Since 2016, recommendation models based on deep neural networks have emerged in both academia and industry. As for the industrial recommendation models, the Wide&Deep model [11] and DeepFM [29] had been deployed for improving the App recommendations. YouTubeDNN [13] and correct-sf6 [94] were used for increasing the accuracy of video recommendation. DIN [105] and DIEN [104] were proposed for modeling the sequential information like user interests with attention mechanism. Wang et al. proposed DCN [83] and DCN V2 [84] to automatically and efficiently learn bounded-degree predictive feature interactions. In academia, researchers also proposed important deep recommendation models, such as FNN [100], PNN [61], NeuralCF [32], NFM [31], CVAE [46]. To address the issue of reproducibility in recommendation model studies [14, 48], [106] developed an open benchmark for CTR prediction, named FuxiCTR. Sun et al. [78] created benchmarks, like reproducible and fair evaluation metrics, for implicit-feedback based top-N recommendation algorithms. [102] proposed a unified framework to develop and reproduce recommendation algorithms for the research purpose. There are also some other open source recommendation models which largely advanced the progress of recommender system studies [22, 28, 90].

In recent years, for addressing the biases in recommender systems, there are increasingly studies on causal inference inspired recommendation [19, 96], Schnabel et al. [73] provided an approach to handling the selection biases by adapting models and estimations from causal inference. Thorsten taught a course named counterfactual machine learning [17] in 2018. Most of the course content was based on examples from information retrieval and recommender systems. In this paper, we do not attempt to give a comprehensive review of all aspects of recommender systems, such as human computer interaction [12, 79], evaluation [4, 15, 59], privacy [2, 98], attacking [44], user experiences [39, 59], fairness [21] etc. There have been a number of thorough surveys about these related research topics. In the rest of the studies, we mainly review two aspects of recommender systems based on our research and industrial experiences, including practical recommendation models and the architectures of typical recommender systems, corresponding to Section 2, 3. Finally, we will briefly discuss some ideas about the future recommender system.

2 RECOMMENDATION MODELS
There have been a number of surveys and books focusing on the recommendation models [23, 65, 99], in this section, we will introduce five kinds of practical recommendation models chronologically. The models are selected mainly based on two principles: the model has important effect on the progress of recommender system, and the model has been successfully deployed on the products.

2.1 Collaborative Filtering
Most of early recommendation models belong to collaborative filtering technologies. K-Nearest-Neighbor (KNN) models predict the user neighbors of a target user u through computing a similarity between u’s prior preferences and the preferences of other users, then the preferences of the neighbors assigned to a target item i can be as a prediction for u to i [34], this classic recommendation model is known as user-user based collaborative filtering. With the similar way, item-item based collaborative filtering predicts a user’s preference to an item i based on the user’s preferences on the neighbors of i. When the number of users is more than that of items, item-item based collaborative filtering model is faster than the user-user one [69]. Some studies [82] combine the two kinds of collaborative filtering approaches. To improve the efficiency of recommendation based on big data, the commercial recommender systems applied SVD [70] for reducing the dimensionality of user-item preference matrix to a latent taste dimensions. During the Netflix Prize, the Matrix Factorization (MF) models [43] had been deeply studied for rating prediction task, the MF is another kind of collaborative filtering technology, which is better than the classic KNN models by considering factors like implicit feedback and temporal information. It is interesting to find that the Netflix product did not really use the winner’s solution, which uses the ensemble approach for combining different models, too large engineering effort for implementation is one reason [66]. Many recommender systems can only collect positive user feedbacks, so several one-class collaborative filtering models [56, 76] are proposed for such recommendation scenarios, where some missing samples as negative. Yu et. al. [95] develop efficient optimization techniques to model “full” samples, where all missing entries as negative samples.

Collaborative filtering based recommendation model is easy to implement and deploy, but there are also some problems, like cold start issues for both user and item, low computing efficiency when the numbers of users and items are huge.

2.2 Linear models
Linear models can mitigate the cold start problem through considering the side information, such as user demographic, historic behaviors, item attributes, contextual information. Logistic regression is one of the most practical linear models for recommendation, especially for the click-through rate (CTR) task [33]. Since it can naturally model the value of CTR between 0 and 1. Given a data set with n instances (yi, xi), i = 1,...,m, where yi is the label and xi is the feature vector, and the model w can be obtained by learning the following optimization:

$$\min_w \frac{\lambda}{2} \|w\|^2_2 + \sum_{i=1}^{n} \log(1 + \exp(-y_i \phi(w, x_i))),$$  

(1)

where λ is the regularization parameter and the \|w\|^2_2 term is used to avoid overfitting data. The second term in (1) is an approximate sum of training errors by using the logistic loss. The \phi function is important. Traditionally, a linear model (LM) is considered with \phi(w, x) = w^T x. Since every feature corresponds to an individual weight of w, the jointly weight between features is not taken into account. So a sort of feature engineering technologies are proposed for modeling the feature pair conjunction. Beside artificial feature interactions, degree-2 polynomial (Poly2) [9] is a widely adopted model, which learns a weight for every possible feature pair. Decision tree plus LR is another practical automatic feature engineering approach, Facebook [33] designs a hybrid model structure, where
boosted decision tree transforms the input features, and the output of each individual tree can be as the input feature to a logistic linear classifier. There are some important progresses about optimizing algorithms for LR, [26] learns the LR with Bayes optimization for improving the Ads CTR estimator of Microsoft Bing. McMahan et al. [53] propose the Follow-the-Regularized-Leader (FTRL) algorithm, and then implement an FTRL-Proximal online learning algorithm for improving the accuracy of CTR prediction task [54].

Logistic regression model is practical for industry recommender system, since it is good at modeling the side information and easy to extend. But LR can not model features conjunction by itself, the conjunction operation always depends on artificial feature engineering or other models, like Poly2 or boost decision trees, so features conjunction and model optimization are two isolate stages.

2.3 Low rank models

Generalized low rank models [80] can approximate a tabular data set by a low rank representation. Matrix factorization and Factorization Machines (FMs) are two popular low rank models, the former one had been introduced in the previous section, while the FMs [63] provides another feature conjunction method, which can decomposes one weight of Poly2 for each feature pair into the inner product of two $k$ dimensional latent vectors corresponding to each feature in the pair, and the $\phi$ function in (1) is as following:

$$\phi_{\text{FMM}}(\mathbf{w}, \mathbf{x}) = \sum_{j_1 = 1}^{n} \sum_{j_2 = j_2 + 1}^{n} (\mathbf{w}_{j_1} \cdot \mathbf{w}_{j_2}) x_{j_1} x_{j_2}, \quad (2)$$

where $j_1$ (or $j_2$) are feature indexes, $\mathbf{w}_{j_1}$ (or $\mathbf{w}_{j_2}$) $\in \mathbb{R}^k$. Because $\mathbf{w}_{j_1}$ and $\mathbf{w}_{j_2}$ are not solely decided by the co-occurrence of feature pair $x_{j_1}$ and $x_{j_2}$, so FMs can mitigate the issue of insufficiently learning of rare feature pairs. Field-aware Factorization Machines [38] groups features to several fields, and each feature has different latent vector for each field, and the $\phi$ is as following:

$$\phi_{\text{FMM}}(\mathbf{w}, \mathbf{x}) = \sum_{j_1 = 1}^{n} \sum_{j_2 = j_2 + 1}^{n} (\mathbf{w}_{j_1, f_1} \cdot \mathbf{w}_{j_2, f_2}) x_{j_1} x_{j_2}, \quad (3)$$

where $f_1$ and $f_2$ are the fields of $j_1$ and $j_2$ respectively. When deciding the weight for $x_{j_1} x_{j_2}$, we use $j_1^{f_1}$s corresponding latent vector on $j_2^{f_2}$ field and $j_2^{f_2}$s corresponding latent vector on $j_1^{f_1}$ field. FMs won some CTR prediction competitions, like Criteo and Avazu Kaggle competitions. [8] indicated that Poly2 tends to be good at modeling the dense features, while the FM/FFM are useful for sparse ones, so a model-based feature conjunction method named AutoConjunction has been proposed.

Fig.1 summarizes the evaluation of the above classic recommendation models. Feature conjunction is important to the accuracy of recommendation estimator. Although the above models can automatically learn the first and second order feature conjunctions efficiently, it is still hard to extend them to model the combined features more than two. So researchers began to study the neural recommendation models, whose structure are more flexible for advanced feature conjunction.

2.4 Neural models

Since 2016, deep neural network based recommendation model has received rapidly growing attention. It not only became research hotspot in academia, but also dominated development of industry recommender system and brought huge commercial value. Starting from general Multi-layer Perceptron, Fig.2 illustrates main evolution process of deep learning recommendation model which includes the following directions.

Classic deep neural network. Deep Crossing [74] is an early-stage deep learning recommendation model, while its architecture is classic and has been inherited by many following models. Deep Crossing has three main components, such as embedding layer, stacking layer and residual layers. The function of embedding layer is to transform sparse categorical features into dense embeddings. Stacking layer is used for concatenating embedding and other continuous features. The last residual layers consisting of residual units automatically combine features to produce a superior model. YouTubeDNN [13] is another influential model with similar architecture as Deep Crossing. The main difference is that YouTubeDNN uses ReLU neurons instead of residual units as the feature crossing layers. The most valuable thing of YouTubeDNN is that it discloses some details of industry recommendation model, such as feature processing, specific methods of model training and serving.

Two-tower neural network. Since the nature of recommendation is to calculate a user interest to a specific item, two-tower neural network containing user tower and item tower becomes a straight forward solution for recommendation model. CF based matrix factorization is a classic and the simplest two-tower model using dot product to calculate similarity between item and user. NeuralCF [31] replaces dot product with MLP to enhance feature crossing capacity.
of the model. Some follow-up studies like YouTube Neural Retrieval Model [94] and Embedding-based Retrieval model of Facebook Search [36] add more features to each tower in the basic two-tower structure. The most important benefit of two-tower structure for industry application is highly efficient model serving. By making use of ANN (Approximate Nearest Neighbor) search in user and item embedding space, recommender system can fetch hundreds of relevant items from millions level candidates pool with a constant time complexity, which is necessary for industry online system.

**Shallow and deep model.** This type of model mainly refers to the Wide&Deep model [11] and its subsequent variants, such as Deep&Cross [83], DeepFM [29], AFM [92] etc. The main idea is to combine two or more deep learning networks with different advantages to enhance model capability. The original Wide&Deep model is composed of LR and DNN, so it has both advantages of strong memorization of LR and generalization of DNN. To strengthen capability of feature crossing, DeepFM replaces LR with FM as a new wide component, while cross layer takes the place of LR in Deep&Cross. On the other hand, AFM and NFM [31] improve deep component by adding attention net and bi-interaction layers.

**Natural Language Processing (NLP) inspired neural model.** There are some NLP key technologies have been applied for improving the performance of recommender system. DIN [105] introduces attention mechanism into recommendation model to learn relevance between user historically interactive items and target item. DIEN [104] uses sequence model to learn the evolution trend of user interest. Transformer models, like BERT [16], have achieved great successes for NLP tasks, they are also quickly applied in the field of sequential recommendation, like BERT4Rec [77] and SSE-PT [91].

**Deep Reinforcement learning.** The environment of online recommender system is a dynamic feedback loop. The user’s interactive behaviors are collected in real time and used to the model training process. Reinforcement learning is good at modeling this kind of continuously changing environment. Some early studies on Exploration and Exploitation (E&E), like LinUCB [45], try to use real-time rewards to improve model. Some studies combine deep learning and reinforcement learning, and enhances the recommendation performance by letting the recommendation model learn user rewards in real time. Among these studies, DRN (Deep Reinforcement Learning Network) [103] is a deep Q-Learning based recommendation framework, whose main innovation is to explore model parameter online by using dueling bandit gradient descent. NICF (Neural Interactive Collaborative Filtering) [107] is another typical model which can quickly catch the user’s interests through representing the exploration policy with a two-tower neural network.

**Graph neural network.** Much of internet data is in the form of graphs like knowledge graph, social network and user-item bipartite graph, so how to efficiently use graph structure data has become a key research direction for providing more accurate, diverse, and explainable recommendation. Some random walk based graph embedding approaches, like DeepWalk [58] and Node2vec [27] make initial trails on this direction. In recent years, different methods of graph neural network have been proposed to directly build a recommendation model on the graph structure. RippleNet [81] starts from selected seeding nodes and recursively propagates the embeddings from a node’s neighbors to refine the node’s embedding. Furthermore, KGAT (Knowledge Graph Attention Network) [87] adds user-item interaction links into graph and employs an attention mechanism to discriminate the importance of the neighbors based on RippleNet.

### 2.5 Causality inspired methods

Recently, the researchers in recommender system community have realized the importance of technology evolution from association learning to causal inference, which can not only improve the accuracy but also have potential to benefit the recommendation in a broader scope of perspectives, such as debias, transparency, fairness, evaluation, robustness, etc. Beyond modeling with the statically observed data, the causality based methods can actively explore new situations through intervention, or build and learn the unobserved world with counterfactual learning methods. For intervention technologies, [6] makes use of uniform data to handle the bias problems through Inverse Propensity Score (IPS), Doubly Robust (DR), joint learning. [37, 86] apply swap intervention approach to improve the ranking tasks. [1, 24] propose several debiasing methods with natural intervention data, and prove their efficiency for mitigating the position bias of information system. For counterfactual learning, [19] summarizes some counterfactual approaches for modeling three kinds of data, such as observed biased data, observed unbiased data and unobserved data. The counterfactual approaches mainly include counterfactual data learning, the correcting biased observed data and doubly robust methods. The basic idea of counterfactual data learning is that we counterfactually assume that each user expresses his or her preferences to each item, and construct the counterfactual samples. The full information data through combining the factual and counterfactual data has potential to learn a more unbiased model. [10, 96, 97] propose efficient methods to learn the parameters of recommendation model from the whole data (including all missing data). Yang et al. [93] counterfactually simulate user ranking-based preferences to handle the data scarce problem with Pearl’s causal inference framework. IPS [73] is the most popular method of the correcting biased data approach, it is defined as the conditional probability of receiving the treatment given pre-treatment covariate [68], but IPS method should satisfy the prerequisite about overlap and unconfoundedness [3], which are not easy to achieve in recommendation scenario when candidates number is large. The doubly robust methods have IPS part and direct method part, corresponding to the first two counterfactual approaches. Dudik et al. [20] prove the DR can yields accurate prediction when either part is a consistent estimator for policy evaluation and learning.

**Structural Causal Model (SCM) [57] and Potential Outcome Model (POM) [51]** are two mainstream causal frameworks, most of the current causality inspired recommendation studies [52, 85, 101] try to analyze the problems (e.g. biases) with SCM, and a few studies [88] begin to frame recommendation with POM for developing the deconfounded recommender.

### 3 TYPICAL RECOMMENDER SYSTEMS

Beyond models, the practical recommender systems need other important components. In this section, we will introduce typical recommender systems and their architectures.
Collaborative Filtering based Recommender System. Tapestry [25] is the earliest collaborative filtering based recommender system, which relied on the opinions of people in a small community. The Fig 2 in [25] describes the flows of information through the major architectural components of Tapestry, including Indexer, Document store, Annotation store, Filterer, Little box, Remailer, Appraiser, Reader/Browser. The collaborative filtering of Tapestry can not serve large communities, since we can not assume each user knows each others, so the neighbors mechanism had been implemented by GroupLens [64] news recommender system, whose architecture is illustrated by the Fig 2 in [64] and Fig 2 in [71], where the Better Bit Bureau is plugged entity to the netnews architecture, it can collect the user preferences from clients, predict the recommendation scores, and send the score to clients. The core components include WWW Server, Dynamic HTML Generator and Recommender System.

Two-layer Recommender System. Bambini et. al. [63] describes how to integrate a recommender system into a real production environment of one IP Television (IPTV) providers. The Fig 9.2 in [65] describes the architecture of the IPTV recommender system, and the Fig 9.3 illustrates the model training in batch stage and prediction in real-time stage. As far as we know, it is the first public recommender system whose architecture is two-stage architecture.

Three-layer Recommender System. Amatriain and Basilico [66] describe a generic three-layer architecture of recommender system at Netflix, where the Online Computation can respond better to recent events, user interaction, and requests in real-time, like generating the recommendation results. The Offline Computation has more time to execute more complex computation with more data, like model training. While the Nearline Computation is an intermediate between these two computations which can perform online-like computations but not in real-time, like user features updating. The main components of the three-layer architecture have been shown in Fig.11.10 in [66]. Dong [18] introduces a general three-layer architecture diagram for industrial recommender systems, and describes its key components. As shown in Fig.3, there are mainly three layers, named Offline Module, Online Service and Nearline Updating. The Offline Module is in charge of model training. It extracts features from Offline Data Service, then generate the samples through Feature Extractor, and the recommendation model can be learned through the Modeling component with batch learning mechanism. After the offline evaluation by Analyzer, we select the best model and upload it to the Predictor in Online Service. The extracted features are also synchronized to the Indexer of Online Service. The Online Service receives and distributes the user’s recommendation requirement through Router. After parsing the requirement and acquire the user ID, we can get the feature vector $x$ through the Indexer and Feature Extractor, finally, the Predictor computes the recommendation score and generate the recommendation list. The Nearline Updating is in charge of updating the features in real-time by Feature updating, and updating the recommendation model according to specific scenario through Model updating.

4 ABOUT FUTURE

The future is very hard to predict, especially for the fast developing recommender systems and their widely applications. However, we
hope to summarize some thoughts about the trends of recommendation based on the above history of web recommendation and our practical experiences. We think that the evolution of recommender systems mainly come from two driving forces, technological innovations and valuable industrial applications.

From the perspective of technological paradigm, the large pre-trained model is learned with big data based on self-supervised learning methods. It has potential to make use more past information to learn more efficient recommendation model. On the one hand, the recommender system can make use of the knowledge in the pre-trained model to improve the system efficiency; on the other hand, we also need to explore novel pre-training methods for recommendation tasks through improving the structure and objectives of the classic pre-train models. Human understands and interacts with the world through multimodal information, recommender system also needs to make use of the multimode technologies to model user, item and context. There are some studies [49, 62] about how to learn multimodal pre-train model. Most of the previous recommendation work studied how to learn the associations with delicately complicated models through fitting the observed data. Although the association based learning methods achieve many successes, the disadvantages gradually emerge, such as the performance is unstable in out-of-distribution conditions, the complex models are nontransparent, the prediction result is less responsible. We think the neural science, causal inference and knowledge-enhanced methods have potential to inspire new solutions to handle the problems.

From the perspective of applications, recommender systems serve users, so more intelligent and user-centric applications will emerge, like naturally conversational recommender system and general intelligent assistants, would be the next mainstream recommendation applications. On the one hand, the future recommender system likes the British butler, who is well trained and serves people based on his knowledge and long serving experiences with the master; on the other hand, it also likes a knowledgeable teacher, who can help us to discover new useful information and benefit our lifelong growth. In the mean time, the user-centric and responsible evaluations would be future studied, we hope the future recommender system can provide more accurate, fair, accountable, knowledgeable, transparent and useful services for improving the user experiences in both long and short term.

**ACKNOWLEDGMENTS**

Special thanks to Prof. John Riedl for his great and pioneering research in area of recommender systems, and his wonderful conversations with Zhenhua Dong about GroupLens, MovieLens, Net Perception during 2010 to 2011, there is no this work without him. Thanks to the colleagues of Huawei Noah’s Ark Lab, there is no this work without their fabulous work in both recommender system applications and academic studies in the past 10 years.
REFERENCES

[1] Aman Agarwal, Ivan Zaitsev, Xuanhui Wang, Cheng Li, Marc Najork, and Thorsten Joachims. 2019. Estimating position bias without intrusive interventions. In Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining. 474–482.

[2] Esma Aimeur, Gilles Brassard, José M Fernandez, and Flavien Serge Mani Onana. 2008. A lambda: a privacy-preserving recommender system for electronic commerce. International Journal of Information Security 7, 5 (2008), 307–334.

[3] Peter C Austin. 2011. An introduction to propensity score methods for reducing the effects of confounding in observational studies. Multivariate behavioral research 46, 3 (2011), 399–424.

[4] Joeran Beel, Stefan Langner, Marcel Genzmeir, Bela Gipp, Corinna Breitinger, and Andreas Nürnberger. 2013. Research paper recommender system evaluation: a quantitative literature survey. In Proceedings of the International Workshop on Reproducibility and Replication in Recommender Systems Evaluation. 15–22.

[5] Nicholas J. Belkin and W. Bruce Croft. 1992. Information filtering and information retrieval: two sides of the same coin? Communications of the ACM. 35, 12 (1992), 29–38.

[6] Stephen Bonner and Flavian Vasile. 2018. Causal embeddings for recommender systems. In Proceedings of the 12th ACM conference on recommender systems. 104–112.

[7] John S. Breese, David Heckerman, and Carl Kadie. 1998. Empirical analysis of collaborative filtering algorithms for recommendation. In UAI '98 Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence. 43–52.

[8] Chih-Yao Chang, Xing Tang, Bo-Wen Yuan, Jui-Yang Hsia, Zhong Liu, Zhenhua Dong, Xiuqiang He, and Chih-Jen Lin. 2020. AutoConjunction: Adaptive Model-based Feature Conjunction for CTR Prediction. In 2020 21st IEEE International Conference on Machine Learning and Applications (ICMLA). IEEE. 202–209.

[9] Yin-Wen Chang, Chou-Jui Hsieh, Kai-Wei Chang, Michael Ringgaard, and Chih-Jen Lin. 2010. Training and testing low-degree polynomial data mappings via linear SVM. Journal of Machine Learning Research 11, 4 (2010).

[10] Chong Cheng, Min Zhang, Yongfeng Zhang, Yiqun Liu, and Shaoping Ma. 2020. DeepFM: A hybrid deep and factorization model for CTR prediction. In Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval. 855–864.

[11] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Yuchin Juan, Yong Zhuang, Wei-Sheng Chin, and Chih-Jen Lin. 2016. Field-aware Factorization Machines for CTR prediction. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 1355–1364.

[12] Dan Cosley, Shyong K Lam, Istvan Albert, Joseph A Konstan, and John Riedl. 2016. Wide & Deep Learning for Recommender Systems. In Proceedings of the 8th International Workshop on Data Mining for Online Advertising. 1–9.

[13] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for recommender systems. In Proceedings of the 22nd International Conference on World Wide Web. 173–182.

[14] Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. 2011. Embedding-based retrieval in Facebook search. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. 194–201.

[15] Xiangnan He, Liao Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tian-Seng Chua. 2017. Neural Collaborative Filtering. In Proceedings of the 26th International Conference on World Wide Web. 585–592.

[16] Junyang Lin, Rui Men, An Yang, Chang Zhou, Ming Ding, Yichang Zhang, Peng Guibing Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, Xiuqiang He, and Zhenhua Dong. 2018. DeepFM: An End-to-End Wide & Deep Learning Framework for CTR Prediction. arXiv preprint arXiv:1804.04950 (2018).

[17] Yehuda Koren. 2010. Collaborative filtering with temporal dynamics. In Proceedings of the 19th international conference on World Wide Web. 661–670.

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

[19] Xiaopeng Li and James She. 2017. Collaborative Variational Autoencoder for Recommender Systems. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 305–314.

[20] Huaqiang Wang, Ang Zhang, Zhi Sun, and Neil Yorke-Smith. 2015. Librec: a Java library for recommender systems. In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. 855–864.

[21] Junyang Lin, Rui Men, An Yang, Chang Zhou, Ming Ding, Yichang Zhang, Peng Wang, Ang Wang, Le Jiang, Xianyan Jia, et al. 2021. M6: A chinese multimodal information retrieval system. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 2553–2561.

[22] Zhichong Fang, Aman Agarwal, and Thorsten Joachims. 2019. Intervention harvesting for context-dependent examination-bias estimation. In Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval. 825–834.

[23] David Goldberg, David Nichols, Brian M. Oki, and Douglass Terry. 1992. Using collaborative filtering to weave an information tapestry. Communications of the ACM. 35, 12 (1992), 61–70.

[24] Thore Graepel, Joaquin Q. Candela, Thomas Borchert, and Ralf Herbrich. 2010. Web-Scale Bayesian Click-Through rate prediction for Sponsored Search Advertising in Microsoft’s Bing Search Engine. In Proceedings of the 27th International Conference on Machine Learning. 15–20.

[25] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. 855–864.

[26] Shyong K Lam, Istvan Albert, Joseph A Konstan, and John Riedl. 1999. An algorithmic framework for performing collaborative filtering. In Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval. Vol. 51. 230–237.

[27] Shyong K Lam and John Riedl. 2004. Shilling recommender systems for user experience. User Modeling and User-adapted Interaction 12, 1 (2002), 101–123.

[28] Hideki Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. 2011. Embedding-based retrieval in Facebook search. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 2553–2561.

[29] Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. 2017. Unbiased learning-to-rank with biased feedback. In Proceedings of the Tenth ACM International Conference on Web Search and Data Mining. 781–789.

[30] Yuchin Juan, Yong Zhuang, Wei-Sheng Chin, and Chih-Jen Lin. 2016. Field-aware Factorization Machines for CTR Prediction. In Proceedings of the 10th ACM Conference on Recommender Systems. 43–50.

[31] Xiangnan He and Tat-Seng Chua. 2017. Neural Factorization Machines for CTR prediction. In Proceedings of the 19th international conference on World Wide Web. 474–482.

[32] Guibing Guo, Jie Zhang, Zhu Sun, and Neil Yorke-Smith. 2015. Librec: a Java library for recommender systems. In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. 855–864.

[33] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. 855–864.
[98] Justin Zhan, Chia-Lung Hsieh, I-Cheng Wang, Tsan-Sheng Hsu, Churn-Jung Liau, and Da-Wei Wang. 2010. Privacy-preserving collaborative recommender systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 40, 4 (2010), 472–476.

[99] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tai. 2019. Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys (CSUR)* 52, 1 (2019), 1–38.

[100] Wenan Zhang, Tianming Du, and Jun Wang. 2016. Deep learning over Multi-Field categorical Data - A case study on user response prediction. In *ECIR*. 45–57.

[101] Yang Zhang, Fuli Feng, Xiangnan He, Tianxin Wei, Chonggang Song, Guoshui Ling, and Yongdong Zhang. 2021. Causal Intervention for Leveraging Popularity Bias in Recommendation. *arXiv preprint arXiv:2105.06067* (2021).

[102] Wayne Xin Zhao, Shanlei Mu, Yu peng Hou, Zihan Lin, Kuiyuan Li, Yushuo Chen, Yujie Lu, Hui Wang, Changxin Tian, Xingyu Pan, Yingqian Min, Zhichao Feng, Xinyan Fan, Xu Chen, Pengfei Wang, Wendi Ji, Yaliang Li, Xiaolin Wang, and Ji-Rong Wen. 2020. RecBole: Towards a Unified, Comprehensive and Efficient Framework for Recommendation Algorithms. *arXiv: Information Retrieval* (2020).

[103] Guanjie Zheng, Fuzheng Zhang, Zihan Zheng, Yang Xiang, Nicholas Jing Yuan, Xing Xie, and Zhenhui Li. 2018. DRN: A deep reinforcement learning framework for news recommendation. In *Proceedings of the 2018 World Wide Web Conference*. 167–176.

[104] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaqiang Zhu, and Kun Gai. 2019. Deep Interest Evolution Network for Click-Through Rate Prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 5941–5948.

[105] Guorui Zhou, Xiaqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep Interest Network for Click-Through Rate Prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery; Data Mining*. 1059–1068.

[106] Jieming Zhu, Jinyang Liu, Shuai Yang, Qi Zhang, and Xiuqiang He. 2020. Fuxi-CTR: An Open Benchmark for Click-Through Rate Prediction. *arXiv preprint arXiv:2009.05794* (2020).

[107] Lixin Zou, Long Xia, Yulong Gu, Xiangyu Zhao, Weidong Liu, Jimmy Xiangu Jian Huang, and Dawei Yin. 2020. Neural interactive collaborative filtering. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 749–758.