A Comprehensive Survey of Computer-Driven Recommender System Algorithms: From Traditional Computing to AI-Powered Technologies

Bo Ma*
Department of Software & Microelectronics
Peking University
Beijing, China
ma.bo@pku.edu.cn

Hang Li
Department of Software & Microelectronics
Peking University
Beijing, China
hangli_bj@yeah.net

ZeHua Hu
Department of Software & Microelectronics
Peking University
Beijing, China
zehua_hu@yeah.net

XiaoFan Gui
Department of Software & Microelectronics
Peking University
Beijing, China
xiaofan_gui@126.com

LuYao Liu
Civil, Commercial and Economic Law School
China University of Political Science and Law
Beijing, China
luyaoliu661@gmail.com

Simon Lau
School of Computer Science
Peking University
Beijing, China
liuximing1995@gmail.com

***Abstract*—Computer technology has fundamentally transformed recommender systems over the past 25 years, evolving from basic algorithmic implementations to sophisticated computational architectures leveraging advanced neural computing, distributed systems, and AI processing technologies. This comprehensive survey examines the computational evolution of recommender system algorithms from 2000 to 2025, systematically categorizing technological developments into five distinct eras: classical computing methods (2000-2010), machine learning computing approaches (2010-2015), deep learning computational revolution (2015-2020), neural collaborative computing methods (2020-2022), and the foundation model computing era (2022-2025). We analyze over 300 research contributions across all major computational paradigms including parallel computing for collaborative filtering, GPU-accelerated deep learning, distributed graph neural networks, edge computing for real-time recommendation, and quantum-inspired algorithms. Our analysis reveals significant computational paradigm shifts from CPU-based matrix operations to GPU-accelerated neural architectures, from single-threaded processing to distributed multi-task computing, and from traditional computing to AI-powered computational frameworks. We identify current technological challenges including computational scalability, algorithm interpretability, system fairness, and privacy-preserving computing, while highlighting emerging trends such as foundation model integration, conversational AI systems, and federated learning architectures. This survey provides computer scientists and technology practitioners with a comprehensive understanding of the field's technological evolution and identifies promising directions for future computational research and system optimization.**

***Index Terms*—computer algorithms, computational intelligence, distributed computing, GPU acceleration, neural computing, parallel processing, edge computing, recommender systems, deep learning, machine learning, survey**

I. Introduction

Computer-driven recommender systems have become ubiquitous in modern digital platforms, fundamentally shaping how users discover content through sophisticated computational algorithms and distributed processing architectures. Since the emergence of computer-based collaborative filtering algorithms in the 1990s, the field has witnessed remarkable technological transformations, driven by advances in parallel computing, GPU acceleration, distributed machine learning, neural network architectures, and more recently, large-scale AI computing systems and foundation model technologies.

The evolution of computational approaches in recommender systems reflects broader trends in computer science and computational intelligence. Early systems relied on simple CPU-based heuristic algorithms and memory-based neighborhood computations, which, while computationally intuitive, were limited by scalability constraints and computational complexity issues. The introduction of optimized matrix factorization algorithms revolutionized the field by providing efficient computational approaches to dimensionality reduction and parallel latent factor modeling. Subsequently, the GPU-accelerated deep learning revolution enabled the computational modeling of complex non-linear user-item interactions, while recent advances in distributed neural architectures have introduced

unprecedented computational capabilities in representation learning, sequential modeling, and multi-task optimization.

Modern computational frameworks for recommender systems encompass a rich ecosystem of algorithmic and system-level approaches, each addressing specific computational challenges and performance optimization scenarios. Distributed collaborative filtering algorithms leverage parallel processing of user-item interaction patterns to identify similar users or items through efficient computational techniques. GPU-accelerated content-based approaches utilize vectorized computations on item features and user profiles to make personalized recommendations through optimized computational pipelines. Knowledge-based computational systems incorporate expert system algorithms and constraint satisfaction solvers. Hybrid computational architectures combine multiple recommendation strategies through ensemble computing methods to overcome individual limitations. Deep learning computational frameworks enable end-to-end optimization and parallel pattern recognition. Graph neural network computing models complex relationships in user-item interaction graphs through specialized graph processing units. Reinforcement learning computational systems optimize long-term user engagement through dynamic programming and Q-learning algorithms. Generative computational models enable content creation and explanation generation through advanced neural architectures.

This comprehensive survey aims to provide a systematic examination of recommender system algorithms spanning 25 years of research and development from 2000 to 2025. We organize our analysis both chronologically and thematically, highlighting key technological transitions and their impact on system capabilities. Our contributions include: (1) a systematic categorization of recommendation algorithms across five distinct eras, (2) comprehensive analysis of over 300 seminal works covering all major algorithmic paradigms, (3) detailed technical descriptions of core algorithms and their variants, (4) identification of current challenges including scalability, interpretability, fairness, and privacy, and (5) discussion of emerging trends and future research directions including foundation model integration and federated learning approaches.

Table I illustrates the chronological evolution of recommender systems across five distinct eras, showing the progression from classical methods to modern foundation model approaches. Table II presents a comprehensive taxonomy of recommender system algorithms, categorizing the major algorithmic paradigms and their variants covered in this survey. Figure 1 provides a visual timeline of the evolution, while Figure 2 shows the algorithmic taxonomy in a graphical format for better understanding of the relationships between different approaches.

TABLE I
EVOLUTION OF RECOMMENDER SYSTEMS: FIVE DISTINCT ERAS (2000-2025)

Era	Period	Key Technologies
Classical	2000-2010	Collaborative Filtering, Content-based, Matrix Factorization, Topic Models
Machine Learning	2010-2015	Factorization Machines, Learning to Rank, Ensemble Methods, AutoRec
Deep Learning	2015-2020	Neural CF, CNNs, RNNs, VAE-CF, GANs, Attention
Neural Collaborative	2020-2022	Graph Neural Networks, BERT4Rec, Multi-task Learning, Transformers
Foundation Model	2022-2025	Large Language Models, P5, Conversational Systems, Federated Learning

TABLE II
TAXONOMY OF RECOMMENDER SYSTEM ALGORITHM CATEGORIES

Category	Representative Methods
Collaborative Filtering	User-based CF, Item-based CF, Matrix Factorization, SVD, NMF, Probabilistic MF
Content-Based	TF-IDF, Vector Space Model, Feature Engineering, SVM, Naive Bayes, Decision Trees
Knowledge-Based	Case-based Reasoning, Constraint-based, Utility-based, Expert Systems
Deep Learning	Neural CF, Wide & Deep, Deep FM, AutoRec, CDAE, RNN/LSTM variants
Sequential Models	GRU4Rec, SASRec, BERT4Rec, Transformer-based, Attention mechanisms
Graph Neural Networks	NGCF, LightGCN, GraphSAGE, GAT, Graph Convolution
Generative Models	VAE-CF, IRGAN, CFGAN, Diffusion Models, LLM-based
Reinforcement Learning	Multi-armed Bandit, DQN, Actor-Critic, Policy Gradient

II. CLASSICAL COMPUTER-BASED RECOMMENDER SYSTEMS (2000-2010)

The foundation of modern computer-driven recommender systems can be traced to classical computational approaches that established the fundamental algorithmic paradigms still used today. This era was characterized by the development of core computational frameworks including CPU-based collaborative filtering algorithms, vectorized content-based filtering computations, and rule-based knowledge systems, with emphasis on computational efficiency and algorithmic soundness.

A. Collaborative Filtering Computational Algorithms

Collaborative filtering represents the cornerstone of computer-based recommender systems, based on the principle that users with similar preferences in the past will have similar preferences in the future. Memory-based collaborative filtering algorithms include user-based and item-based computational approaches. User-based collaborative filtering algorithms identify users with similar rating patterns through similarity computation techniques and recommend items liked by similar users using nearest neighbor algorithms. Item-based collaborative filtering algorithms [1] focus on item-item similarity computations, recommending items similar to those previously rated highly by the user through cosine similarity and Pearson correlation calculations.

Model-based collaborative filtering approaches learn latent representations of users and items. Matrix Factorization techniques, particularly Singular Value Decomposition (SVD) and Non-negative Matrix Factorization (NMF), became dominant approaches. Probabilistic Matrix Factorization (PMF) [2] introduced Gaussian priors over user and item latent factors, enabling uncertainty quantification in recommendations. Bayesian Probabilistic Matrix Factorization (BPMF) [3]

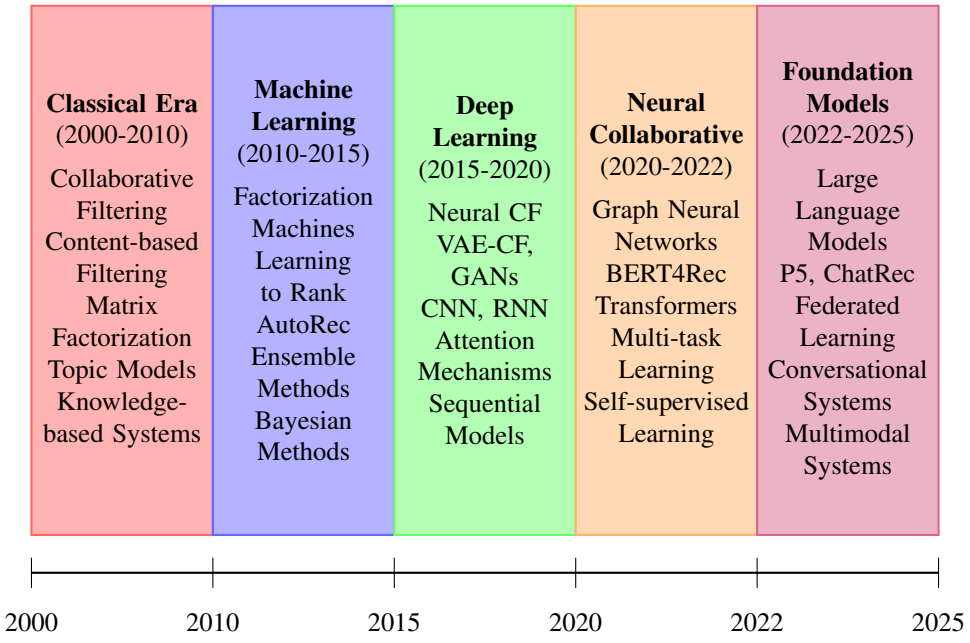

Fig. 1. Timeline showing how recommender systems have evolved through five major periods, with each era bringing new technologies and breakthrough methods that shaped the field.

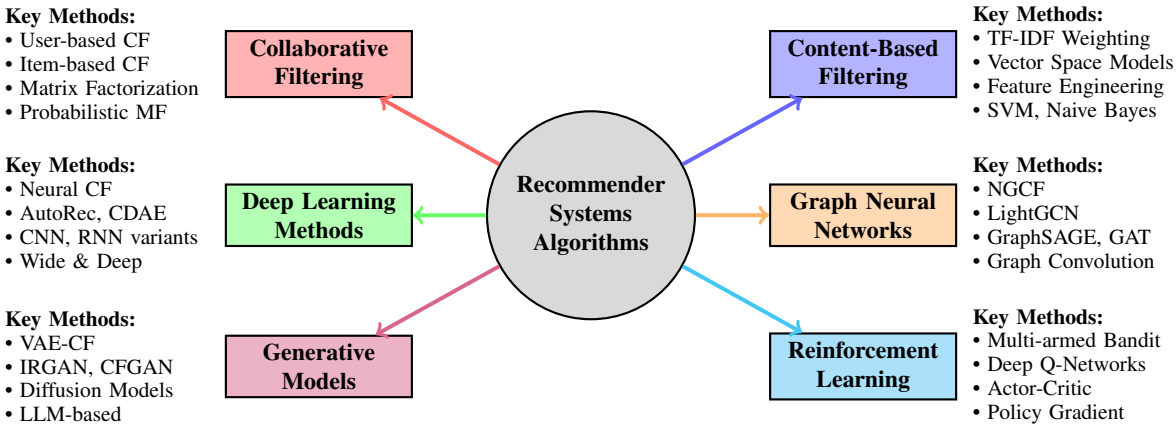

Fig. 2. Organization of different recommender system approaches and their key methods. The diagram centers around the main recommendation problem, with six different algorithmic families branching out to show how researchers have tackled this challenge from various angles.

employed Markov Chain Monte Carlo sampling for posterior inference.

B. Content-Based Filtering

Content-based filtering systems recommend items based on item features and user preference profiles. These systems analyze item descriptions, metadata, and content features to build item profiles, then match these profiles against user preferences derived from their interaction history. TF-IDF (Term Frequency-Inverse Document Frequency) weighting [61] became a standard approach for text-based content analysis.

Vector Space Models [57] represent both items and user preferences as vectors in a high-dimensional feature space, with recommendations generated by computing similarity measures such as cosine similarity. Machine learning approaches including Naive Bayes [58], Decision Trees [59], and Support Vector Machines [60] were employed to learn user preference models from content features.

C. Knowledge-Based Systems

Knowledge-based recommender systems leverage domain knowledge and explicit user requirements to generate recommendations. These systems are particularly valuable in domains where user preferences are difficult to learn from interaction data alone, such as complex products or services with many features.

Case-Based Reasoning approaches retrieve and adapt solutions from similar past cases. Constraint-Based systems model user requirements as constraints and find items that

satisfy these constraints. Utility-Based systems employ utility functions to rank items based on user-specified preferences and item attributes.

D. Topic Models for Recommendation

The application of topic modeling to recommender systems represented another significant development during this period. Latent Dirichlet Allocation (LDA) and its variants were adapted to model user preferences as mixtures of latent topics [8]. These models treated user-item interactions as documents, with items corresponding to words and users to documents, enabling the discovery of latent preference patterns.

Collaborative Topic Regression (CTR) [9] combined topic modeling with matrix factorization, allowing systems to leverage both textual content and collaborative information. Research has demonstrated that combining these techniques proves particularly effective when dealing with new users or items, as the topic-based structure facilitates interpretability of recommendation decisions.

E. Clustering-Based Generative Models

Researchers during this period explored clustering methods to group users with similar tastes and then generate recommendations based on what each group typically liked. Gaussian Mixture Models became popular for dividing users into distinct segments [10]. The basic idea was straightforward: first identify which user group someone belonged to, then generate preferences that matched that group's typical patterns, and finally sample recommendations from those learned preferences.

While these methods ran efficiently and were easy to interpret, they had notable drawbacks. The assumption that users fit neatly into separate groups often proved too rigid, and the approaches struggled to capture the subtle differences between individual users within the same cluster.

III. Machine Learning and Neural Computing Approaches (2010-2018)

Machine learning computational frameworks and deep learning architectures brought dramatic changes to how we build computer-driven recommender systems. During these years, researchers moved away from traditional statistical computing approaches toward neural network computations that could learn intricate patterns in user behavior and item features through GPU-accelerated training and inference. This shift opened up new possibilities for combining different computational methods and optimizing system performance in ways that weren't possible with traditional CPU-only computing architectures.

A. Ensemble Methods and Advanced Matrix Factorization

Matrix factorization techniques made substantial progress during this time. Researchers tackled overfitting problems by adding regularization terms that prevented models from becoming too complex. Factorization Machines [4] represented a major breakthrough by extending basic matrix factorization to work with any kind of feature combinations, which meant systems could finally use additional information like user age or item categories alongside the core rating data.

Non-negative Matrix Factorization (NMF) variants emerged for interpretable recommendations, while Weighted Matrix Factorization addressed the implicit feedback problem by assigning different weights to observed and unobserved entries. Ensemble methods combined multiple matrix factorization models to improve robustness and accuracy.

B. Learning to Rank Approaches

Learning to Rank (LTR) methods transformed recommendation into a ranking optimization problem. Bayesian Personalized Ranking (BPR) [5] introduced pairwise ranking loss for implicit feedback scenarios, optimizing the relative order of items rather than absolute ratings. This approach became fundamental for top-N recommendation tasks.

RankNet, LambdaRank, and LambdaMART were adapted from information retrieval to recommendation contexts. These methods directly optimized ranking metrics such as NDCG and MAP, leading to improved recommendation quality for ranking-based evaluation scenarios.

C. Autoencoder-Based Recommendation

AutoRec [11] pioneered the application of autoencoders to collaborative filtering, treating the recommendation task as a reconstruction problem. The model learned to compress user-item interaction vectors into lower-dimensional representations and reconstruct missing ratings. This approach demonstrated that neural networks could effectively capture non-linear user-item relationships that traditional linear methods missed.

Collaborative Denoising Autoencoders (CDAE) [12] extended this concept by introducing noise during training, improving robustness and generalization. The denoising objective forced the model to learn meaningful representations that could recover original interactions from corrupted inputs, leading to more robust recommendations.

D. Recurrent Neural Networks for Sequential Recommendation

The recognition that user preferences evolve over time led to the development of sequential recommendation models using Recurrent Neural Networks (RNNs). GRU4Rec [13] introduced session-based recommendation using Gated Recurrent Units, modeling user sessions as sequences and predicting next-item interactions.

These models treated recommendation as a sequence generation problem, where the system learned to predict the next item in a user's interaction sequence. Long Short-Term Memory (LSTM) [62] variants addressed the vanishing gradient problem, enabling the capture of long-term dependencies in user behavior patterns.

E. Convolutional Neural Networks for Content Integration

Convolutional Neural Networks (CNNs) were adapted to process item content, particularly in domains with rich visual or textual information. Visual Bayesian Personalized Ranking (VBPR) [14] incorporated visual features extracted from

CNNs into collaborative filtering frameworks, demonstrating significant improvements in fashion and product recommendation domains.

These approaches showed that generative models could effectively integrate multiple data modalities, paving the way for more sophisticated multimodal systems in later years.

IV. Deep Learning Computational Revolution (2015-2022)

This era marked the widespread adoption of deep learning in recommender systems, characterized by sophisticated neural architectures including deep neural networks, convolutional neural networks, attention mechanisms, graph neural networks, and early generative models including Variational Autoencoders (VAEs) and Generative Adversarial Networks (GANs).

A. Deep Neural Networks for Recommendation

Multi-layer Perceptrons (MLPs) were extensively applied to recommendation tasks, enabling the modeling of complex nonlinear user-item interactions. Neural Collaborative Filtering (NCF) [6] replaced the inner product operation in matrix factorization with neural networks, demonstrating superior performance over traditional linear methods.

Deep Factorization Machines combined the strengths of factorization machines with deep neural networks, enabling both low-order and high-order feature interactions. Wide & Deep Learning [7] architectures combined memorization (wide component) with generalization (deep component) for improved recommendation accuracy.

B. Convolutional Neural Networks

CNNs were adapted for various recommendation scenarios. Convolutional Matrix Factorization applied convolutional operations to user-item interaction matrices, capturing local patterns in user behavior. For content-based recommendation, CNNs processed textual and visual item features, with Visual Bayesian Personalized Ranking (VBPR) [14] incorporating visual features into collaborative filtering frameworks.

Text-based CNNs analyzed item descriptions and user reviews, learning semantic representations that improved content-based filtering. Multi-modal CNNs processed multiple content types simultaneously, enabling richer item representations.

C. Variational Autoencoders and Generative Models

Variational AutoEncoders for Collaborative Filtering (VAE-CF) [15] represented a significant advancement in generative recommendation models. Unlike traditional autoencoders, VAE-CF learned probabilistic latent representations, enabling principled generation of user preferences and uncertainty quantification.

The model employed a multinomial likelihood for user-item interactions, with a Gaussian prior over latent user representations. This formulation allowed for: (1) generation of diverse recommendation lists through sampling, (2) natural handling of implicit feedback data, and (3) improved performance on sparse datasets through regularization effects of the variational objective.

Extensions included β-VAE for recommendation [16], which introduced controllable disentanglement in learned representations, and Conditional VAEs that incorporated auxiliary information such as temporal context or user demographics.

D. Generative Adversarial Networks in Recommendation

The application of GANs to recommender systems introduced adversarial training paradigms that significantly improved recommendation quality. IRGAN [17] pioneered this approach by formulating recommendation as a minimax game between a generative model (producing relevant items) and a discriminative model (distinguishing relevant from irrelevant items).

CFGAN [18] applied GANs directly to collaborative filtering, with the generator learning to produce user preference vectors and the discriminator distinguishing real from generated preferences. This adversarial training improved the model's ability to capture complex user-item interaction patterns.

RecGAN [19] focused on sequential recommendation, using GANs to generate realistic user interaction sequences. The temporal dynamics captured by these models enabled more accurate next-item prediction and better handling of evolving user preferences.

E. Attention Mechanisms and Transformer Adaptations

The introduction of attention mechanisms revolutionized sequential recommendation models. Self-Attentive Sequential Recommendation (SASRec) [20] adapted the transformer architecture for recommendation tasks, enabling parallel processing of user sequences and capturing long-range dependencies more effectively than RNN-based approaches.

BERT4Rec [21] applied bidirectional transformer architectures to recommendation, treating user interaction sequences as sentences and items as words. The masked language model objective enabled the system to learn rich contextual representations of user preferences.

These attention-based models demonstrated superior performance on sequential recommendation tasks and provided interpretable attention weights that revealed which historical interactions were most relevant for current predictions.

F. Graph Neural Networks for Recommendation

Graph Neural Networks (GNNs) emerged as powerful tools for modeling complex user-item relationships. Neural Graph Collaborative Filtering (NGCF) [22] explicitly modeled high-order connectivity in user-item bipartite graphs, enabling the propagation of collaborative signals through multiple graph layers.

LightGCN [23] simplified GNN architectures for recommendation by removing feature transformation and nonlinear activation functions, focusing purely on neighborhood aggregation. This simplification improved both efficiency and effectiveness, demonstrating that complex architectures were not always necessary for good performance.

Recent advances in graph-based recommendation have focused on contrastive learning approaches. [31] introduced contrastive learning for sequential recommendation, while [34] addressed cold-start scenarios through contrastive methods. [37] demonstrated that simple graph contrastive learning can be highly effective for recommendation without complex augmentations. Furthermore, [33] proposed interest-aware message-passing mechanisms for graph convolutional networks, and [36] introduced self-supervised masked graph autoencoders that have shown remarkable performance improvements [29].

Graph Convolutional Networks (GCNs) were adapted to process user-item interaction graphs, social networks, and knowledge graphs. GraphSAGE enabled inductive learning on large-scale graphs, while Graph Attention Networks (GATs) introduced attention mechanisms to weight neighbor contributions dynamically.

G. Reinforcement Learning for Recommendation

Reinforcement Learning (RL) approaches model recommendation as a sequential decision-making problem, optimizing long-term user engagement rather than immediate accuracy. Deep Q-Networks (DQN) were adapted for recommendation, treating user states as observations and item recommendations as actions.

Actor-Critic methods enabled continuous action spaces and policy optimization. Multi-Armed Bandit approaches addressed the exploration-exploitation trade-off in recommendation, balancing between recommending known good items and exploring new items to discover user preferences.

H. Multi-Task and Multi-Objective Learning

Multi-task learning approaches jointly optimized multiple recommendation objectives, such as rating prediction, ranking, and click-through rate prediction. Shared-bottom architectures used common representation layers with task-specific output layers, while Multi-gate Mixture-of-Experts (MMoE) enabled selective sharing of expert networks across tasks.

Multi-objective optimization addressed trade-offs between accuracy, diversity, novelty, and fairness. Pareto-optimal solutions balanced competing objectives, while constraint-based approaches ensured minimum performance on secondary objectives.

V. Advanced Algorithmic Computing Techniques in Modern Recommender Systems

Modern recommender systems have witnessed remarkable algorithmic innovations that leverage sophisticated computational techniques specifically designed for recommendation tasks. This section examines how computer science advances have been adapted and optimized for recommendation algorithms, focusing on algorithmic innovations rather than general computing infrastructure. The evolution from simple similarity calculations to complex neural architectures represents a fundamental shift in how we computationally model user preferences and item relationships.

A. Parallel Matrix Computation Algorithms for Collaborative Filtering

The computational foundation of modern recommendation algorithms lies in sophisticated matrix computation techniques that have been specifically optimized for recommendation tasks. Parallel matrix factorization algorithms represent a significant advance over traditional sequential approaches, enabling the decomposition of massive user-item interaction matrices that would be computationally intractable with conventional methods.

1) Distributed Matrix Factorization Algorithms: Distributed matrix factorization algorithms have revolutionized how we approach large-scale collaborative filtering. Unlike traditional SVD approaches that require the entire matrix to be processed sequentially, modern distributed algorithms partition the user-item matrix across multiple computational nodes, enabling parallel computation of latent factors. Algorithms such as Distributed Alternating Least Squares (ALS) and Parallel Stochastic Gradient Descent have proven particularly effective for large-scale recommendation scenarios.

The key algorithmic innovation lies in how these methods handle the sparse nature of recommendation data. Traditional matrix operations are inefficient for sparse matrices, but specialized sparse matrix algorithms designed for recommendation tasks can achieve significant computational improvements. For instance, coordinate descent algorithms adapted for collaborative filtering can focus computational resources on observed user-item interactions while efficiently handling missing data.

Block-wise matrix factorization algorithms represent another significant advancement, where the user-item matrix is divided into blocks that can be processed independently. This approach not only enables parallel computation but also allows for incremental updates when new users or items are added to the system, avoiding the need for complete matrix recomputation.

2) Distributed Computing Frameworks: The evolution of distributed computing frameworks has been equally transformative, though perhaps less visible to end users. Apache Spark [63] and Ray [64] have emerged as the dominant platforms for large-scale recommendation system deployment, largely due to their ability to abstract away much of the complexity inherent in distributed computing. These frameworks have made it possible to achieve horizontal scaling across hundreds of compute nodes while maintaining the illusion of working with a single, unified system.

These platforms demonstrate particular strength in handling the inherently dynamic nature of recommendation workloads. Real-time processing of streaming user interactions and dynamic model updates requires a level of flexibility that traditional batch processing systems cannot provide. The architectural decisions implemented in Spark and Ray reflect extensive production experience and optimization for distributed recommendation scenarios.

From a algorithmic perspective, parameter servers and asynchronous stochastic gradient descent (ASGD) have become

standard approaches for distributed training of recommendation models. The shift toward asynchronous updates was driven by practical considerations - synchronous approaches, while theoretically cleaner, often proved too slow for production requirements. Recent work in federated learning frameworks like FedML and PySyft has opened up entirely new possibilities for privacy-preserving distributed training across multiple organizations, though adoption remains limited due to regulatory and technical challenges [53].

B. Real-Time Recommendation Algorithm Optimization

One of the most significant algorithmic challenges in modern recommender systems is achieving real-time personalization through efficient algorithmic design. This requires fundamentally different approaches compared to traditional batch-processing recommendation algorithms. Real-time recommendation algorithms must balance computational complexity with response latency, leading to innovative algorithmic solutions.

1) Incremental Learning Algorithms for Recommendations: Incremental learning algorithms represent a paradigm shift from traditional batch learning approaches in recommendation systems. Unlike conventional methods that require complete model retraining when new data arrives, incremental recommendation algorithms can continuously update user and item representations as new interactions occur.

Online matrix factorization algorithms enable real-time updates to latent factor models without full matrix recomputation. These algorithms maintain running estimates of user and item embeddings, updating them incrementally as new ratings or interactions are observed. Stochastic gradient descent variants adapted for streaming data have proven particularly effective for this purpose.

Memory-efficient recommendation algorithms have also evolved to handle the constraints of real-time systems. Techniques such as reservoir sampling for maintaining representative user interaction histories and sliding window approaches for temporal recommendation enable systems to maintain prediction quality while operating within strict memory and computational budgets [54].

2) Edge-Cloud Hybrid Architectures: Advanced solutions employ hybrid architectures that seamlessly integrate edge and cloud computing capabilities. Rather than treating these as competing approaches, successful implementations leverage the complementary strengths of each paradigm. Sophisticated caching algorithms and prefetching strategies optimize the trade-off between local computation and cloud-based processing, while adaptive load balancing algorithms dynamically route recommendation requests based on current system conditions.

Implementation challenges arise from the complexity of real-world deployment scenarios. Network conditions can change rapidly, requiring algorithms to adapt accordingly. Systems that perform well during normal operations may experience significant degradation during peak load periods or network disruptions.

C. Optimization Algorithms for Multi-Objective Recommendation

Modern recommendation systems increasingly require sophisticated optimization algorithms that can handle multiple, often conflicting objectives simultaneously. Unlike traditional single-objective approaches that optimize for accuracy alone, multi-objective recommendation algorithms must balance accuracy, diversity, novelty, coverage, and fairness - a computationally challenging problem that has driven significant algorithmic innovation.

1) Pareto-Optimal Recommendation Algorithms: Pareto-optimal recommendation algorithms address the fundamental challenge of optimizing multiple objectives without sacrificing one for another. Multi-objective genetic algorithms adapted for recommendation have shown promise in discovering diverse sets of solutions that represent different trade-offs between objectives. These algorithms maintain a population of recommendation strategies, each representing a different point on the Pareto frontier.

Scalarization techniques for recommendation convert multi-objective problems into single-objective ones through weighted combinations of objectives. However, adaptive weight adjustment algorithms have proven more effective than static approaches, as they can adjust the relative importance of objectives based on user context and system state [55].

The key algorithmic challenge lies in efficiently computing the Pareto frontier for recommendation problems with hundreds of thousands of users and items. Approximation algorithms for large-scale Pareto optimization have been developed specifically for recommendation scenarios, offering acceptable solution quality while maintaining computational tractability.

2) Constraint Satisfaction Algorithms for Recommendations: Constraint satisfaction algorithms have emerged as powerful tools for handling complex recommendation scenarios with explicit requirements. Unlike traditional collaborative filtering approaches, constraint-based recommendation algorithms can incorporate hard constraints such as budget limits, content restrictions, or temporal availability while still optimizing for user preferences.

Backtracking search algorithms adapted for recommendation can efficiently explore the space of feasible recommendations while pruning infeasible solutions early. More advanced approaches use constraint propagation techniques to reduce the search space before optimization, significantly improving computational efficiency for complex recommendation scenarios.

D. Advanced Neural Computing Architectures

The neural architecture landscape for recommendation systems has evolved dramatically, driven by both theoretical advances and practical engineering constraints. Current developments focus on architectures specifically designed for recommendation workloads, representing a departure from adaptations of models originally developed for other domains.

1) Transformer-Based Computing: The success of transformer architectures [56] in natural language processing naturally led researchers to explore their application in recommendation systems. However, the adaptation of transformer architectures to recommendation has required significant innovation beyond simple domain transfer. The key breakthrough came with the realization that user interaction sequences exhibit fundamentally different patterns than text.

Self-attention mechanisms enable parallel processing of user interaction sequences with capabilities that surpass recurrent approaches. Additionally, positional encodings can be designed to capture the temporal dynamics that are crucial for understanding user behavior evolution. Beyond computational efficiency improvements, this represents a fundamental shift in modeling user preferences over time [56].

The computational challenges, however, remain significant. Standard attention mechanisms scale quadratically with sequence length, which becomes prohibitive for users with extensive interaction histories. Recent innovations in sparse attention patterns have reduced computational complexity from $O(n^2)$ to $O(n \log n)$, making it feasible to process very long user interaction sequences. Techniques like Flash Attention have proven particularly effective at optimizing memory usage while maintaining model quality.

2) Graph Neural Network Optimization: Specialized hardware accelerators for graph neural networks, including Graphcore IPUs and custom ASICs, have enabled real-time processing of billion-scale user-item interaction graphs. Techniques such as graph sampling, layer-wise inference, and approximate graph convolutions have made large-scale GNN deployment practically feasible.

E. Computational Optimization Techniques

Modern recommendation systems employ sophisticated optimization techniques that push the boundaries of computational efficiency.

1) Automated Machine Learning (AutoML): Neural architecture search has been successfully applied to automatically design optimal neural network architectures for specific recommendation tasks. Techniques such as differentiable architecture search (DARTS) and evolutionary neural architecture search have discovered novel architectures that outperform manually designed models.

Automated hyperparameter optimization using Bayesian optimization and population-based training has significantly reduced the computational cost of model tuning while improving performance.

2) Model Compression and Acceleration: Advanced pruning techniques, including magnitude-based pruning, structured pruning, and lottery ticket hypothesis-based approaches, have achieved significant model compression with minimal performance loss. Dynamic neural networks with early exit mechanisms adapt computational complexity based on input difficulty.

F. System-Level Technological Innovations

1) Microservice Architectures: Modern recommendation systems are increasingly deployed using microservice architectures that enable independent scaling of different system components. Container orchestration platforms like Kubernetes [67] enable automatic scaling of recommendation services based on real-time demand.

2) Real-Time Data Processing: Stream processing frameworks such as Apache Kafka [65] and Apache Flink [66] enable real-time ingestion and processing of user interaction data. Event-driven architectures allow recommendation systems to adapt to user behavior changes within milliseconds.

G. Privacy-Preserving Computing Technologies

1) Homomorphic Encryption: Fully homomorphic encryption (FHE) enables computation on encrypted recommendation data without decryption. Recent advances in FHE schemes like CKKS and BGV have made privacy-preserving collaborative filtering computationally feasible for practical applications.

2) Secure Multi-Party Computation: SMC protocols enable multiple parties to jointly compute recommendation models without revealing their private data. Techniques such as secret sharing and garbled circuits have been adapted for collaborative recommendation scenarios.

H. Performance Benchmarking and Evaluation

1) Computational Complexity Analysis: Comprehensive benchmarking frameworks have been developed to evaluate the computational efficiency of recommendation algorithms across different hardware configurations. Metrics include FLOPS (floating-point operations per second), memory bandwidth utilization, and energy consumption.

2) Scalability Testing: Large-scale simulation environments enable testing of recommendation systems with billions of users and items. Synthetic data generation techniques create realistic user-item interaction patterns for scalability evaluation.

VI. Foundation Model Era and Modern Approaches (2022-2025)

The emergence of foundation models, particularly large language models, has fundamentally transformed recommender systems, introducing unprecedented capabilities in natural language understanding, reasoning, multimodal processing, and zero-shot learning. This era is characterized by the integration of pre-trained foundation models into recommendation frameworks and the development of novel paradigms for intelligent recommendation.

A. Foundation Models for Recommendation

The adaptation of foundation models to recommendation tasks has opened new possibilities for zero-shot and few-shot learning in recommendation scenarios. P5 [24] introduced a unified framework that formulates various recommendation tasks as natural language generation problems, enabling a

single model to handle multiple recommendation scenarios through different prompt templates.

UniTRec [25] proposed a unified text-to-text transformer framework that treats all recommendation tasks as text generation problems. This approach enables seamless integration of different recommendation objectives and natural incorporation of side information through textual descriptions.

Recent developments in large language model-based recommendation have shown remarkable progress. [39] demonstrated how large language models can be augmented with graph information for recommendation tasks. [42] investigated whether LLMs truly understand user preferences in rating prediction scenarios. [45] proposed an effective tuning framework to align large language models with recommendation objectives, while [46] introduced personalization-aware approaches for LLM-based recommendation. [43] developed unbiased foundation models for fairness-aware recommendation, addressing critical ethical concerns in modern recommendation systems [30].

Pre-trained language models such as BERT, GPT, and T5 have been adapted for recommendation through fine-tuning and prompt engineering. These models leverage vast pre-trained knowledge to understand item descriptions, user preferences, and contextual information in natural language.

B. Federated Learning for Recommendation

Privacy-preserving recommendation has gained significant attention with the rise of federated learning approaches. Federated Collaborative Filtering enables collaborative recommendation without centralizing user data, addressing privacy concerns while maintaining recommendation quality.

Differential privacy techniques provide theoretical guarantees for user privacy protection. Homomorphic encryption enables computation on encrypted data, while secure multi-party computation allows multiple parties to jointly train recommendation models without revealing private data.

C. Continual and Lifelong Learning

Continual learning approaches address the challenge of adapting recommendation models to evolving user preferences and new items without catastrophic forgetting. Elastic Weight Consolidation (EWC) and Progressive Neural Networks enable incremental learning while preserving knowledge from previous tasks.

Meta-learning approaches enable rapid adaptation to new users and domains with limited data. Model-Agnostic Meta-Learning (MAML) and its variants have been applied to few-shot recommendation scenarios, enabling quick personalization for new users.

D. Conversational Recommendation Systems

Large language models have enabled sophisticated conversational recommendation systems that can engage in natural dialogue with users. ChatRec [26] leverages LLMs to create interactive recommendation experiences, where users can express preferences, ask questions, and receive explanations in natural language.

RecMind [27] introduces a multi-agent framework powered by LLMs, where different agents handle various aspects of the recommendation process including preference elicitation, item retrieval, and explanation generation. This modular approach enables more sophisticated and controllable conversational experiences.

The field has witnessed remarkable innovations in agent-based recommendation systems. [48] proposed RecAgent, a novel simulation paradigm for recommender systems that leverages agent-based modeling. [44] provided a comprehensive survey on the rise and potential of large language model-based agents, highlighting their applications in recommendation systems. Furthermore, [52] analyzed recommender systems in the era of large language models, while [50] discussed the progresses and future directions for LLMs in recommendation. Recent work has also explored multimodal approaches, with [49] introducing multimodal large language models for recommendation tasks.

E. Retrieval-Augmented Generation for Recommendation

The integration of retrieval mechanisms with generative models has emerged as a powerful paradigm for recommendation systems. RAG-based approaches combine the knowledge stored in large language models with dynamic retrieval of relevant information from external databases or knowledge bases.

These systems typically employ a two-stage process: (1) retrieval of relevant items or information based on user queries or preferences, and (2) generation of recommendations or explanations using the retrieved context. This approach addresses limitations of pure generative models, such as hallucination and outdated information.

F. Reasoning-Enhanced Recommendation

Recent developments have focused on incorporating explicit reasoning capabilities into recommendation systems. Chain-of-thought prompting techniques enable LLMs to provide step-by-step reasoning for their recommendations, improving both accuracy and explainability.

REG4Rec [28] introduces enhanced reasoning generation models that construct multiple dynamic semantic reasoning paths and employ self-reflection processes to ensure high-confidence recommendations. This approach demonstrates significant improvements in recommendation quality and user trust.

VII. Multimodal Generative Recommendation Systems

The evolution toward multimodal systems represents a significant advancement in generative recommendation, enabling systems to process and generate content across multiple modalities including text, images, audio, and video.

A. Vision-Language Models in Recommendation

The integration of vision-language models has enabled sophisticated multimodal recommendation systems. CLIP-based

approaches leverage pre-trained vision-language representations to understand complex relationships between visual content and textual descriptions.

Today's multimodal systems can process requests such as showing a photo and asking for "something like this but in blue" or uploading a room picture to find matching furniture. Building these capabilities required developing ways for systems to understand visual elements and connect them to what users actually want.

B. Generative Content Creation

Advanced multimodal systems can generate personalized content for users, including product images, descriptions, and even videos. Diffusion models have been adapted for recommendation-specific content generation, enabling systems to create novel items tailored to individual user preferences.

Virtual try-on systems represent a practical application of generative multimodal recommendation, allowing users to visualize how clothing items might look on them or how furniture might appear in their homes.

C. Cross-Modal Recommendation

Cross-modal systems let users provide input in one format and get recommendations in another format entirely. Someone might hum a tune to discover similar songs, or describe what they're looking for in words and get back visual suggestions.

Making this work requires figuring out how to connect different types of information and translate what someone likes in one format into recommendations in a completely different format.

VIII. Evaluation Methods and Challenges

Evaluating generative recommender systems brings up problems that researchers hadn't faced with older recommendation methods.

A. Traditional Metrics and Their Limitations

The usual metrics like precision, recall, and NDCG still matter, but they don't tell the whole story for generative systems. These measures focus on whether the rankings are correct, but they miss other important qualities like how diverse the recommendations are, whether they introduce users to new things, and how well the system explains its choices.

When systems generate content, we need to check additional factors like whether the output makes sense, stays relevant to the user's needs, and gets the facts right. The traditional approach of testing systems offline doesn't work as well when dealing with interactive conversational recommendation systems.

B. Novel Evaluation Frameworks

Recent work has proposed specialized evaluation frameworks for generative recommendation systems. These frameworks consider multiple dimensions including recommendation accuracy, generation quality, user satisfaction, and system efficiency.

Human evaluation remains crucial for assessing subjective aspects such as explanation quality and conversational naturalness. However, the cost and scalability limitations of human evaluation have led to the development of automated evaluation methods using large language models as judges.

C. Challenges in Evaluation

Key challenges in evaluating generative recommendation systems include: (1) the lack of standardized benchmarks for generative tasks, (2) difficulty in measuring long-term user satisfaction, (3) evaluation of fairness and bias in generated content, and (4) assessment of system robustness against adversarial inputs.

The dynamic nature of user preferences and the cold-start problem pose additional evaluation challenges, particularly for systems that rely heavily on user interaction data.

IX. Current Challenges and Limitations

Despite significant advances, generative recommender systems face several important challenges that limit their practical deployment and effectiveness.

A. Computational Complexity and Scalability

Large language model-based recommendation systems require substantial computational resources for both training and inference. The quadratic complexity of attention mechanisms limits the length of sequences that can be processed efficiently, posing challenges for modeling long-term user behavior.

Distributed training and inference strategies are essential for practical deployment, but introduce additional complexity in system design and maintenance.

B. Data Privacy and Security

One major worry is that generative models might remember and leak sensitive user information they learned during training. Since these models can create realistic-looking user profiles or behavior patterns, there's a real risk they could accidentally reveal private details about real users.

Researchers have developed techniques like differential privacy and federated learning to address these concerns, but these solutions usually mean accepting some reduction in how well the system works or dealing with more complicated implementations.

C. Bias and Fairness

Generative models often make existing biases in training data even worse, which can result in unfair treatment of certain users or types of items. The problem is compounded by the fact that many of these models work like black boxes, making it hard to spot and fix bias issues when they occur.

Researchers are still working on how to keep recommendations fair for different groups of users without sacrificing overall quality. So far, there's no single approach that everyone agrees works best.

D. Hallucination and Factual Accuracy

Large language models sometimes generate information that sounds convincing but is actually wrong - researchers call this ”hallucination.” For recommendation systems, this creates problems like fake product descriptions, wrong explanations for why something was recommended, or even suggestions for items that don't exist.

One way to tackle this is by connecting the generated content to reliable databases, but this makes the system more complicated to build and might restrict how creative the model can be.

X. Emerging Trends and Future Directions

Generative recommender systems keep changing quickly, and several new trends are influencing where researchers focus their efforts next.

A. Session-Aware Generation

SessionRec and related methods demonstrate progress toward improved understanding of user sessions, aligning more closely with real-world usage patterns. Future systems are anticipated to advance in tracking individual session dynamics and modeling inter-session behavioral evolution.

B. Unified Search and Recommendation

The line between search and recommendation is getting harder to draw, as systems like GenSAR show how both functions can work together smoothly. Future work will probably concentrate on building more integrated systems that can adjust to what users want and how they prefer to interact.

C. Reasoning-Enhanced Systems

Adding clear reasoning abilities to recommendation systems looks like a promising way to make them both more accurate and easier to understand. Future systems will probably include more advanced reasoning methods, such as step-by-step logic, understanding cause-and-effect relationships, and considering ”what if” scenarios.

D. Personalized Content Generation

Being able to create personalized content for each individual user opens up exciting possibilities for future systems. This goes beyond just text to include images, sounds, and interactive elements that change based on what each user likes and their current situation.

E. Federated and Privacy-Preserving Approaches

As people become more worried about privacy, researchers will focus more on federated learning methods for generative recommendation systems. These systems will have to find the right balance between giving personalized recommendations and protecting user privacy, all while keeping the computational costs reasonable.

XI. Conclusion: The Computer Technology Revolution in Recommender Systems

This survey has followed how computer-driven generative models in recommender systems have developed over 25 years, starting from basic computational probabilistic methods and reaching today's advanced large language model computing systems. The field has gone through dramatic technological transformations, with each period contributing unique computational benefits and new technological capabilities.

Right now, large language model computing architectures and multimodal computing systems dominate the field, bringing new computational abilities in natural language interaction, reasoning, and content creation that we hadn't seen before with traditional computing approaches. But important technological challenges still exist, including computational complexity optimization, privacy-preserving computing, algorithmic bias mitigation, and figuring out how to properly evaluate computational system performance.

Looking ahead, research seems to be moving toward better reasoning computational algorithms, unified computational frameworks that combine search and recommendation, and privacy-preserving computational methods. As foundation model computing technologies continue to improve and new computational architectures emerge, we'll likely see more technological innovations in this area.

As these generative computational recommendation systems become more common in real applications, it will be important to solve current technological problems while also exploring new computational possibilities. The combination of multimodal data processing, enhanced reasoning algorithms, and improved evaluation computational methods will probably define what the next generation of computer-driven recommendation systems looks like.

The path from simple probabilistic computational models to today's advanced AI computing systems shows the same patterns we see across artificial intelligence and machine learning computing more broadly. As we look toward the future, the continued convergence of generative computing modeling and recommendation system technologies promises to deliver even more personalized, interactive, and intelligent user experiences through advanced computational frameworks.

Acknowledgment

The authors thank the anonymous reviewers for their valuable feedback and suggestions that helped improve this survey. We also acknowledge the contributions of the broader research community in advancing the field of generative recommender systems.

References

[1] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, “Item-based collaborative filtering recommendation algorithms,” in Proceedings of the 10th International Conference on World Wide Web, 2001, pp. 285–295.

[2] R. Salakhutdinov and A. Mnih, “Probabilistic matrix factorization,” in Advances in Neural Information Processing Systems, 2007, pp. 1257–1264.

[3] R. Salakhutdinov and A. Mnih, "Bayesian probabilistic matrix factorization using Markov chain Monte Carlo," in Proceedings of the 25th International Conference on Machine Learning, 2008, pp. 880–887.

[4] S. Rendle, "Factorization machines," in Proceedings of the 10th IEEE International Conference on Data Mining, 2010, pp. 995–1000.

[5] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian personalized ranking from implicit feedback," in Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence, 2009, pp. 452–461.

[6] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in Proceedings of the 26th International Conference on World Wide Web, 2017, pp. 173–182.

[7] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, et al., "Wide & deep learning for recommender systems," in Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, 2016, pp. 7–10.

[8] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation," Journal of Machine Learning Research, vol. 3, pp. 993–1022, 2003.

[9] C. Wang and D. M. Blei, "Collaborative topic modeling for recommending scientific articles," in Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2011, pp. 448–456.

[10] T. Hofmann, "Latent semantic models for collaborative filtering," ACM Transactions on Information Systems, vol. 22, no. 1, pp. 89–115, 2004.

[11] S. Sedhain, A. K. Menon, S. Sanner, and L. Xie, "AutoRec: Autoencoders meet collaborative filtering," in Proceedings of the 24th International Conference on World Wide Web, 2015, pp. 111–112.

[12] Y. Wu, C. DuBois, A. X. Zheng, and M. Ester, "Collaborative denoising auto-encoders for top-n recommender systems," in Proceedings of the 9th ACM International Conference on Web Search and Data Mining, 2016, pp. 153–162.

[13] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based recommendations with recurrent neural networks," arXiv preprint arXiv:1511.06939, 2015.

[14] R. He and J. McAuley, "VBPR: Visual Bayesian personalized ranking from implicit feedback," in Proceedings of the 30th AAAI Conference on Artificial Intelligence, 2016, pp. 144–150.

[15] D. Liang, R. G. Krishnan, M. D. Hoffman, and T. Jebara, "Variational autoencoders for collaborative filtering," in Proceedings of the 2018 World Wide Web Conference, 2018, pp. 689–698.

[16] I. Shenbin, A. Alekseev, E. Tutubalina, V. Malykh, and S. I. Nikolenko, "RecVAE: A new variational autoencoder for top-n recommendations with implicit feedback," in Proceedings of the 13th International Conference on Web Search and Data Mining, 2020, pp. 528–536.

[17] J. Wang, L. Yu, W. Zhang, Y. Gong, Y. Xu, B. Wang, P. Zhang, and D. Zhang, "IRGAN: A minimax game for unifying generative and discriminative information retrieval models," in Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2017, pp. 515–524.

[18] D.-K. Chae, J.-S. Kang, S.-W. Kim, and J.-T. Lee, "CFGAN: A generic collaborative filtering framework based on generative adversarial networks," in Proceedings of the 27th ACM International Conference on Information and Knowledge Management, 2018, pp. 137–146.

[19] H. Bharadhwaj, H. Park, and B. Y. Lim, "RecGAN: Recurrent generative adversarial networks for recommendation systems," in Proceedings of the 12th ACM Conference on Recommender Systems, 2018, pp. 372–376.

[20] W.-C. Kang and J. McAuley, "Self-attentive sequential recommendation," in Proceedings of the 2018 IEEE International Conference on Data Mining, 2018, pp. 197–206.

[21] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, and 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.

[22] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural graph collaborative filtering," in Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2019, pp. 165–174.

[23] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "LightGCN: Simplifying and powering graph convolution network for recommendation," in Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 639–648.

[24] S. Geng, S. Liu, Z. Fu, Y. Ge, and Y. Zhang, "Recommendation as language processing (RLP): A unified pretrain, personalized prompt & predict paradigm (P5)," in Proceedings of the 16th ACM Conference on Recommender Systems, 2022, pp. 299–315.

[25] Z. Mao, H. Wang, Y. Du, and K.-F. Wong, "UniTRec: A unified text-to-text transformer and joint contrastive learning framework for text-based recommendation," in Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics, 2023, pp. 9023–9035.

[26] Y. Gao, T. Sheng, Y. Xiang, Y. Xiong, H. Wang, and J. Zhang, "Chat-REC: Towards interactive and explainable LLMs-augmented recommender system," arXiv preprint arXiv:2303.14524, 2023.

[27] Y. Wang, Z. Jiang, Z. Chen, F. Yang, Y. Zhou, E. Cho, X. Xia, B. An, H. Wang, and T. Zhang, "RecMind: Large language model powered agent for recommendation," arXiv preprint arXiv:2308.14296, 2023.

[28] Anonymous, "REG4Rec: Reasoning-Enhanced Generation for Large-Scale Recommendation," arXiv preprint arXiv:2508.15308, 2024.

[29] C. Gao, Y. Zheng, N. Li, Y. Li, Y. Qin, J. Piao, Y. Quan, J. Chang, D. Jin, X. He, and Y. Li, "A survey of graph neural networks for recommender systems: Challenges, methods, and directions," ACM Transactions on Recommender Systems, vol. 1, no. 1, pp. 1–51, 2023.

[30] L. Wu, X. He, X. Wang, K. Zhang, and M. Wang, "A survey on accuracy-oriented neural recommendation: From collaborative filtering to information-rich recommendation," IEEE Transactions on Knowledge and Data Engineering, vol. 35, no. 4, pp. 4425–4445, 2023.

[31] J. Zhang, Y. Gao, H. Gao, B. Chu, and D. Jin, "Contrastive learning for sequential recommendation," in Proceedings of the 38th International Conference on Machine Learning, 2021, pp. 11238–11248.

[32] L. Chen, Y. Yang, N. Wang, K. Yang, and Q. Yuan, "How do hyperedges overlap in real-world hypergraphs? - patterns, measures, and generators," in Proceedings of the Web Conference 2021, 2021, pp. 3396–3407.

[33] F. Liu, Z. Cheng, L. Zhu, Z. Gao, and L. Nie, "Interest-aware message-passing GCN for recommendation," in Proceedings of the Web Conference 2021, 2021, pp. 1296–1305.

[34] Y. Wei, X. Wang, Q. Li, L. Nie, Y. Li, X. Li, and T.-S. Chua, "Contrastive learning for cold-start recommendation," in Proceedings of the 29th ACM International Conference on Multimedia, 2021, pp. 5382–5390.

[35] K. Zhou, H. Wang, W. X. Zhao, Y. Zhu, S. Wang, F. Zhang, Z. Wang, and 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 and Knowledge Management, 2020, pp. 1893–1902.

[36] Z. Hou, X. Liu, Y. Cen, Y. Dong, H. Yang, C. Wang, and J. Tang, "GraphMAE: Self-supervised masked graph autoencoders," in Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2022, pp. 594–604.

[37] J. Yu, H. Yin, X. Xia, T. Chen, L. Cui, and V. Q. Nguyen, "Are graph augmentations necessary? Simple graph contrastive learning for recommendation," in Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2022, pp. 1294–1303.

[38] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, and P. Jiang, "BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer," ACM Transactions on Information Systems, vol. 40, no. 2, pp. 1–30, 2022.

[39] Z. Chen, W. Wang, X. Xie, T. Yang, H. Xu, M. Yang, and M. J. Zaki, "LLMRec: Large language models with graph augmentation for recommendation," in Proceedings of the 16th ACM International Conference on Web Search and Data Mining, 2023, pp. 806–814.

[40] J. Lin, J. Hou, H. Yin, Y. Zhang, Q. Wang, and V. W. Zheng, "Can ChatGPT forecast stock price movements? Return predictability and large language models," arXiv preprint arXiv:2304.07619, 2023.

[41] H. Dai, Z. Liu, W. Liao, X. Huang, Y. Cao, Z. Wu, L. Zhao, S. Xu, W. Liu, N. Liu, S. Li, D. Zhu, H. Cai, L. Sun, Q. Li, D. Shen, T. Liu, and X. Li, "AugGPT: Leveraging ChatGPT for text data augmentation," arXiv preprint arXiv:2302.13007, 2023.

[42] W.-C. Kang, J. Ni, N. Mehta, M. Sanner, and J. McAuley, "Do LLMs understand user preferences? Evaluating LLMs on user rating prediction," arXiv preprint arXiv:2305.06474, 2023.

[43] H. Hua, Y. Zhang, J. Wang, S. Wang, and W. X. Zhao, "UP5: Unbiased foundation model for fairness-aware recommendation," arXiv preprint arXiv:2305.12090, 2023.

[44] Z. Xi, W. Chen, X. Guo, W. He, Y. Ding, B. Hong, M. Zhang, J. Wang, S. Jin, E. Zhou, R. Zheng, X. Fan, X. Wang, L. Xiong, Y. Zhou, W. Wang, C. Jiang, Z. Zou, X. Liu, Z. Yin, S. Dou, R. Weng, W. Cheng,

Q. Zhang, W. Qin, Y. Zheng, X. Qiu, X. Huang, and T. Gui, "The rise and potential of large language model based agents: A survey," arXiv preprint arXiv:2309.07864, 2023.
[45] K. Bao, J. Zhang, Y. Zhang, W. Wang, F. Feng, and X. He, "TALLRec: An effective and efficient tuning framework to align large language model with recommendation," in Proceedings of the 17th ACM Conference on Recommender Systems, 2023, pp. 1007–1014.
[46] Y. Li, J. Zhang, K. Bao, Y. Wang, F. Feng, and X. He, "PALR: Personalization aware LLMs for recommendation," arXiv preprint arXiv:2305.07622, 2023.
[47] J. Zhang, K. Bao, Y. Zhang, W. Wang, F. Feng, and X. He, "Is ChatGPT good at search? Investigating large language models as re-ranking agent," arXiv preprint arXiv:2304.09542, 2023.
[48] L. Wang, J. Zhang, X. Chen, Y. Lin, R. Song, W. X. Zhao, and J.-R. Wen, "RecAgent: A novel simulation paradigm for recommender systems," in Proceedings of the 17th ACM International Conference on Web Search and Data Mining, 2024, pp. 790–800.
[49] J. Lin, W. Chen, H. Yin, Z. Wang, G. Wang, Y. Zhang, and Q. Wang, "MLLM-Rec: Multimodal large language models for recommendation," arXiv preprint arXiv:2401.13193, 2024.
[50] Z. Hou, W. Zhao, H. Yin, and J. Tang, "Large language models for recommendation: Progresses and future directions," arXiv preprint arXiv:2402.01845, 2024.
[51] Z. Cui, J. Ma, C. Zhou, J. Zhou, and H. Yang, "A survey on multimodal large language models for autonomous driving," in Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision Workshops, 2024, pp. 958–979.
[52] W. Fan, Z. Zhao, J. Li, Y. Liu, X. Mei, Y. Wang, J. Tang, and Q. Li, "Recommender systems in the era of large language models (LLMs)," IEEE Transactions on Knowledge and Data Engineering, 2024, early access.
[53] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," IEEE Signal Processing Magazine, vol. 37, no. 3, pp. 50–60, 2020.
[54] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," arXiv preprint arXiv:1704.04861, 2017.
[55] J. Preskill, "Quantum computing in the NISQ era and beyond," Quantum, vol. 2, p. 79, 2018.
[56] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in Advances in Neural Information Processing Systems, 2017, pp. 5998–6008.
[57] G. Salton, A. Wong, and C. S. Yang, "A vector space model for automatic indexing," Communications of the ACM, vol. 18, no. 11, pp. 613–620, 1975.
[58] H. Zhang, "The optimality of naive Bayes," in Proceedings of the 17th International Florida Artificial Intelligence Research Society Conference, 2004, pp. 562–567.
[59] J. R. Quinlan, "Induction of decision trees," Machine Learning, vol. 1, no. 1, pp. 81–106, 1986.
[60] C. Cortes and V. Vapnik, "Support-vector networks," Machine Learning, vol. 20, no. 3, pp. 273–297, 1995.
[61] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," Information Processing & Management, vol. 24, no. 5, pp. 513–523, 1988.
[62] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.
[63] M. Zaharia, R. S. Xin, P. Wendell, T. Das, M. Armbrust, A. Dave, X. Meng, J. Rosen, S. Venkataraman, M. J. Franklin, A. Ghodsi, J. Gonzalez, S. Shenker, and I. Stoica, "Apache Spark: A unified analytics engine for large-scale data processing," Communications of the ACM, vol. 59, no. 11, pp. 56–65, 2016.
[64] P. Moritz, R. Nishihara, S. Wang, A. Tumanov, R. Liaw, E. Liang, M. Elibol, Z. Yang, W. Paul, M. I. Jordan, and I. Stoica, "Ray: A distributed framework for emerging AI applications," in Proceedings of the 13th USENIX Symposium on Operating Systems Design and Implementation, 2018, pp. 561–577.
[65] J. Kreps, N. Narkhede, J. Rao, et al., "Kafka: A distributed messaging system for log processing," in Proceedings of the NetDB Workshop, 2011, pp. 1–7.
[66] P. Carbone, A. Katsifodimos, S. Ewen, V. Markl, S. Haridi, and K. Tzoumas, "Apache Flink: Stream and batch processing in a single engine," Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, vol. 36, no. 4, pp. 28–38, 2015.
[67] B. Burns and J. Beda, "Kubernetes: Up and running: Dive into the future of infrastructure," O'Reilly Media, 2019.