World Journal of Advanced Research and Reviews
eISSN: 2581-9615 CODEN (USA): WJARAI
Cross Ref DOI: 10.30574/wjarr
Journal homepage: https://wjarr.com/

(Review Article)

Designing Recommender Systems at Scale: Multi-Stage Architecture and Cold-Start Optimization

Aniruddha Zalani *

Indian Institute of Technology, Kanpur, India.

World Journal of Advanced Research and Reviews, 2025, 26(03), 1097-1102

Publication history: Received on 31 March 2025; revised on 08 June 2025; accepted on 10 June 2025

Article DOI: https://doi.org/10.30574/wjarr.2025.26.3.2050

Abstract

Recommender systems form the backbone of digital platforms, facilitating content discovery in increasingly crowded online ecosystems. Large social media platforms face the dual challenge of processing vast data volumes while maintaining relevance across diverse user preferences. The cold-start problem—effectively recommending content with minimal historical data—remains a persistent challenge in these environments. This article examines major platforms' architecture and optimization techniques to address these challenges. The multi-stage recommendation architecture combines diverse candidate generation methods with deep neural ranking, supported by sophisticated caching and feature retrieval systems. For cold-start scenarios, specialized techniques, including exploration pipelines, freshness boosting, and contextual matching, have significantly improved engagement. Real-world implementations show that properly designed recommender systems can improve relevance metrics while reducing computational latency. Integrating robust system design with cold-start optimization provides a blueprint for recommendation systems that maintain high relevance from a user's first interaction onward, balancing the competing demands of scale, speed, and personalization quality.

Keywords: Recommender Systems; Cold-Start Problem; Multi-Stage Architecture; Real-Time Feature Engineering; Personalization Acceleration; Content Exploration

1. Introduction

Recommender systems have become central to digital platforms, facilitating the discovery of relevant content in an increasingly crowded online ecosystem. For large social media platforms serving hundreds of millions of users daily, these systems face dual challenges: They must efficiently process massive amounts of data while maintaining high relevance across diverse user preferences and content types. One particularly persistent challenge is the "cold-start problem"—effectively recommending new items or serving new users with minimal historical interaction data.

Reels recommendation system demonstrates the immense scale of modern content discovery platforms, processing over 95 billion Reels daily while serving more than 2 billion monthly active users, as Ayushi Jain (2023) reported. The platform's multi-layered recommendation architecture implements a sophisticated candidate generation system that evaluates thousands of potential videos per user session, with the initial retrieval phase narrowing approximately 70,000 candidates to around 500 for deeper evaluation before final presentation to users [1]. This approach has enabled achieving a reported 44% year-over-year increase in time spent on Reels while maintaining recommendation generation times under 150 milliseconds.

The cold-start problem represents a significant barrier to recommendation performance in platforms of this scale. According to Pasrija and Pasrija (2024), new users typically require 8-10 meaningful interactions before personalized

* Corresponding author: Aniruddha Zalani

recommendations begin showing substantial relevance improvements, with standard collaborative filtering approaches suffering from a 37% reduction in recommendation precision for users with fewer than 5 interactions [2]. Their analysis of e-commerce platforms revealed that new products experience an average of 43% fewer impressions during their first 24 hours despite potentially high relevance to user interests, creating a significant barrier to content discovery.

Drawing from implementations across major platforms, including Instagram Reels, it demonstrates how specialized cold-start techniques improved overall engagement by 15% across core metrics. Instagram's approach leverages a sophisticated embedding-based ranking system that utilizes over 50 signals, including content consumption patterns, creator relationships, and visual similarity features, to overcome initial data sparsity for new content [1]. Similarly, techniques analyzed by Pasrija and Pasrija (2024) show that hybrid content-based filtering solutions incorporating demographic data can reduce new user recommendation error rates by up to 18.7% compared to pure collaborative approaches [2].

Our approach provides actionable insights for practitioners seeking to build recommendation systems that maintain high relevance from a user's first interaction onwards, balancing the competing demands of scale, speed, and personalization quality.

2. Multi-Stage Recommender Architecture

The foundation of scalable recommendation systems is a multi-stage architecture designed to handle the computational challenges of serving over 100 million daily users. This architecture follows a progressive filtering approach, beginning with broad candidate generation and culminating in precise personalized ranking.

The first stage employs diverse candidate generation methods, including collaborative filtering-based factorization machines, graph neural networks for user-item embeddings, and sequence models capturing temporal patterns in user behavior. According to Covington et al. from Google, their American video-sharing company recommendation system initially retrieves between 10,000 and 50,000 video candidates from various sources, with collaborative filtering approaches alone generating approximately 15,000 potential recommendations [3]. Their multi-stage architecture handles more than 400 hours of video uploaded every minute while serving recommendations to over 2 billion monthly active users. Similarly, Smith and Linden documented that Amazon's product recommendation engine processes 66.5 billion observations daily while maintaining latency requirements below 100 milliseconds [4].

The second stage implements a lightweight scoring mechanism that rapidly narrows this candidate pool to several hundred items. This intermediate filtering uses simplified versions of ranking models, primarily focused on user-item affinity signals and basic content quality metrics. According to Smith and Linden, an American e-commerce company's second-stage filtering typically reduces an initial pool of 20,000 candidate products to approximately 500, using a simplified model with only 50-100 features that can be computed in under 10 milliseconds per request [4]. An American video-sharing company recommendation system similarly narrows candidates to "hundreds" using what they term "lightweight scoring functions" that achieve 95% of full model accuracy while requiring only 1/100th of the computational resources [3].

The final ranking stage applies deep neural networks to score the remaining candidates comprehensively. Production models incorporate over 1,000 features spanning user demographics, historical engagement patterns, content characteristics, and contextual factors. An American video-sharing company recommendation system employs a deep neural network with three hidden layers containing 256, 512, and 1024 nodes, processing approximately 200 features per video-user pair [3]. This architecture is supported by specialized feature retrieval and caching layers that dramatically reduce latency.

Performance measurements indicate that this architecture improved overall engagement by 15% while reducing recommendation latency by 30% compared to previous monolithic approaches. Google's implementation achieved a 40% increase in watch time metrics through its multi-stage architecture. Smith and Linden report that Amazon's tiered approach reduced average recommendation generation time from 150ms to 35ms while improving purchase conversion rates by 29% [4].

Table 1 Multi-Stage Recommender Architecture Components [3, 4]

Component	Function	Implementation Examples
Candidate Generation	Creates an initial pool of potential recommendations using diverse methods	American video-sharing company (collaborative filtering, embedding models), Amazon (factorization machines)
Intermediate Filtering	Narrows candidate pool using lightweight scoring	Amazon (simplified models with focused features), American video-sharing company (lightweight scoring functions)
Deep Ranking	Comprehensive scoring of remaining candidates	American video-sharing company (neural network with three hidden layers), Amazon (tiered ranking approach)
Feature Retrieval	Supports ranking with cached and computed features	Chinese internet company (multi-tiered caching), Amazon (feature service infrastructure)

3. Real-Time Feature Engineering and Serving Infrastructure

Supporting the multi-stage architecture is a sophisticated feature engineering and serving infrastructure designed to handle real-time recommendations at scale. This infrastructure addresses the critical challenge of retrieving and processing features with millisecond-level latency requirements while serving millions of concurrent requests.

The feature service layer implements a three-tiered caching strategy: an in-memory cache for frequently accessed features, a distributed Redis cluster for medium-access features, and a fallback to the primary database for rare feature lookups. According to Chen et al., their recommendation platform at a Chinese internet company implements a similar multi-tiered caching strategy that handles over 10 billion user requests daily with an average latency of 30 milliseconds [5]. Their system employs a distributed in-memory cache spanning 200 machines with 256GB RAM, achieving a 97.2% hit rate for high-frequency features. A Redis cluster containing 500 nodes provides secondary caching with 2.3% of requests for medium-frequency features, while only 0.5% require database access. This architecture helped them reduce feature retrieval latencies from an average of 120ms to just 30ms, representing a 75% improvement critical for maintaining user engagement on their platform, which serves over 600 million daily active users [5].

For real-time features based on recent user actions, a stream processing pipeline using Apache Kafka and Flink processes user interactions as they occur. Agarwal et al. detail an American service provider company's LASER platform, which ingests approximately 1 terabyte of event data daily through a Kafka cluster capable of processing over 100,000 events per second [6]. This system updates user-item affinity scores within 10-30 seconds of user actions, incorporating these fresh signals into recommendation generations. Their platform maintains a continuous processing pipeline that handles 120 million daily user interactions, generating real-time updates to a feature space containing over 100 million items and 259 million users. This low-latency processing proved crucial for American service provider companies' advertising response prediction models, where incorporating fresh user activity signals improved click-through rates by 5.92% compared to models using only batch-computed features [6].

To accommodate the resource-intensive nature of deep learning models in the final ranking stage, an adaptive batching mechanism dynamically adjusts inference batch sizes based on the current system load. Chen et al. describe a Chinese internet company's implementation that varies batch sizes between 8 and 512 requests depending on traffic conditions, achieving 4.2x higher throughput during peak periods while keeping latencies within acceptable bounds [5]. Their system dynamically provisions between 80 and 320 GPU servers during different traffic conditions, with each NVIDIA V100 GPU handling approximately 300 requests per second at optimal batch sizes of 128 inference requests.

The resulting infrastructure supports over 1 million recommendation requests per second during peak hours with 99th percentile latency under 100ms. American service provider company's LASER system maintains p95 latencies of 65ms when serving 35,000 queries per second, achieved through a distributed architecture of 20 serving nodes with request-level parallelization. This enables American service provider companies to meet strict service level agreements for their advertising customers [6].

Table 2 Real-Time Feature Engineering Infrastructure [5, 6]

Component	**Function**	**Implementation Examples**
Multi-Tiered Caching	Optimizes feature retrieval latency	Chinese internet company (in-memory cache, Redis cluster, database fallback)
Stream Processing	Enables real-time feature updates	American service provider company LASER (Kafka/Flink pipeline)
Adaptive Batching	Optimizes inference throughput	Chinese Internet company (dynamic batch size adjustment)
Distributed Serving	Supports high request volume	American service provider company (request-level parallelization)

4. Cold-Start Optimization for New Content

A critical challenge in recommender systems is providing adequate visibility for new content with minimal historical engagement data—the "cold-start problem." To address this, specialized techniques for content-side cold-start optimization have been developed.

The approach begins with a dedicated exploration pipeline that selectively exposes new content to a small, diverse subset of users. According to Bressan et al., from an American video streaming platform, their production system implements an exploration strategy that initially exposes new titles to approximately 1-2% of their user base, carefully selected to represent diverse viewing preferences [7]. Their exploration algorithm dynamically adjusts exposure rates based on early performance signals, with exploration pools ranging from 450,000 to 1.2 million users depending on the content category and predicted popularity. This controlled exposure strategy allows an American video streaming platform to gather statistically significant engagement data within 24-48 hours of content release while minimizing potential negative impacts on overall user experience. Their implementation leverages a Thompson sampling multi-armed bandit framework that balances exploration with exploitation. It incorporates 84 content features extracted using deep neural networks that predict engagement potential from visual, textual, and audio signals [7].

Table 3 Cold-Start Content Optimization Techniques [7, 8]

Technique	**Function**	**Implementation Examples**
Exploration Pipeline	Selectively exposes new content	An American video streaming platform (controlled exposure strategy)
Freshness Boost	Temporarily increases ranking scores	American social media service (time-aware boosting factor)
Content Quality Prediction	Estimates engagement potential	An American video streaming platform (neural networks for engagement prediction)
Real-Time Monitoring	Tracks early performance signals	American social media service (multi-metric evaluation)

A "freshness boost" re-ranking strategy within the recommendation pipeline has proven effective for social media video content. As detailed by Eksombatchai et al. from an American social media service, their "PinnerSage" recommendation system implements a time-aware boosting factor that temporarily increases ranking scores for new content [8]. This boost factor begins at 2.7x for newly uploaded content and decays logarithmically over a 72-hour period, with a half-life typically set at 24 hours. The boost magnitude also varies based on creator authority metrics—with established creators receiving an additional 15-30% initial boost—and projected content quality as determined by a specialized "cold-start quality classifier" that analyzes 128 content features. This approach smooths from cold-start to normal recommendation patterns while ensuring new content receives adequate exposure.

Real-time monitoring tools track the performance of new content across key engagement dimensions. American social media service system evaluates early signals across six core engagement metrics: click-through rate, save rate, and session duration impact [8]. When content demonstrates unusually positive early signals (exceeding category benchmarks by 1.8 standard deviations), their system automatically increases exploration rates from the default 1.5% to up to 4.5% of eligible users.

Implementing these techniques on social media platforms has yielded measurable improvements: American social media service reported a 0.5% increase in content saves, a 0.8% increase in click-through rates, and a 0.3% increase in session durations for fresh content [8]. These seemingly modest percentage improvements translate to millions of additional daily interactions platform-wide, significantly enhancing content creator satisfaction and platform engagement metrics.

5. User Cold-Start and Personalization Acceleration

While content cold-start represents one dimension of the challenge, the user cold-start problem—providing relevant recommendations to new users—presents another critical area for optimization. Techniques to rapidly build personalized user models from minimal interaction data have shown significant promise in production environments.

For new users, modern recommendation systems leverage a combination of contextual signals and demographic similarities to existing users. According to Volkovs et al. from Koho Financial and the University of Toronto, their cold-start recommendation system analyzes multiple contextual features, including geographic location, device type, and referral source, to create initial user profiles. Their approach, implemented in a commercial recommendation system serving millions of users, demonstrated that properly constructed content-based neighbor models can achieve up to 90% of the performance of fully developed collaborative filtering models for new users [9]. Their implementation extracted 1,024-dimensional embeddings from content metadata to build feature vectors, allowing for effective user-item matching even without explicit interaction history. This novel technique leveraged an attentional neural network architecture (CBNN-a) trained on a dataset of 7.08 million users with 478 million interactions, which outperformed traditional matrix factorization methods by 12% for completely new users and 7.2% for users with very limited history [9].

As users interact with content, an incremental learning approach heavily weights these first interactions. Aharon et al. from Yahoo Research describe an "OFF-Set" algorithm designed for the persistent cold-start problem in online recommendation systems [10]. Their approach employs a factorization technique that processes feature sets in a single pass, making it particularly well-suited for environments where new users and items constantly enter the system. Their model processes approximately 100,000 new features daily while maintaining a vocabulary of 100 million features in production. For new users, they implement specialized feature integration that can generate effective user representations after as few as 3-5 interactions, compared to the significantly larger interaction histories typically required by standard factorization approaches. Their approach showed a 13.7% improvement in recommendation accuracy compared to baseline methods when tested on a dataset containing 234 million events from Yahoo's front page [10].

Table 4 User Cold-Start Techniques [9, 10]

Technique	Function	Implementation Examples
Contextual Matching	Creates initial user profiles	Koho/University of Toronto (content-based neighbor models)
Fast Embedding	Generates provisional user representations	Yahoo (OFF-Set algorithm)
Progressive Introduction	Balances exploration and exploitation	Koho/University of Toronto (diversity-aware exploration)
Incremental Learning	Heavily weights initial interactions	Yahoo (specialized feature integration)

To further enhance cold-start performance, progressive introduction strategies have proven effective. Volkovs et al. demonstrate a diversity-aware exploration approach that presents users with recommendations spanning multiple content categories [9]. Their system strategically samples from different content clusters while maintaining recommendation quality, gradually shifting from exploration to exploitation as user preferences become clearer. This approach effectively balances immediate engagement with long-term preference learning, particularly important for the critical first session, where user retention decisions are often made.

These techniques significantly reduce the "cold-start gap"—the performance difference between recommendations for new versus established users. Aharon et al. reported that their approach reduced this gap by approximately 30% in production environments at Yahoo, with particularly strong improvements in user engagement metrics during the first

24 hours after sign-up [10]. This improvement addressed one of the most challenging periods in the user lifecycle, where abandonment rates are typically highest due to less personalized experiences.

6. Conclusion

The design and implementation of large-scale recommender systems require balancing multiple competing objectives: computational efficiency, personalization quality, and effective handling of cold-start scenarios. The multi-stage architecture described throughout this article demonstrates how major platforms have successfully addressed these challenges through tiered processing pipelines that progressively narrow recommendation candidates while increasing scoring sophistication. This approach, implemented by companies, yielded substantial improvements in recommendation quality and system performance. The sophisticated feature engineering infrastructure supporting these architectures, incorporating multi-tiered caching, stream processing, and adaptive inference optimization, enables real-time personalization at an unprecedented scale. Perhaps most significantly, the specialized techniques for cold-start optimization provide pathways to overcome one of recommendation systems' most persistent challenges. For content cold-start, controlled exploration strategies and dynamic boosting mechanisms create visibility opportunities for new items, while user cold-start techniques leverage contextual signals and accelerated embedding generation to rapidly personalize experiences. These approaches, demonstrated through implementations at an American video streaming platform, American social media service, and other platforms, significantly reduce the quality gap between new and established entities in the recommendation ecosystem. These architectural patterns and optimization techniques form a comprehensive blueprint for building recommendation systems that maintain high relevance from the first interaction onward while scaling to serve hundreds of millions of users.

References

[1] Ayushi Jain, "Decoding Instagram System Design & Architecture (And How Reels Recommendation Works?)," TechAhead, Nov 26, 2024. [Online]. Available: https://www.techaheadcorp.com/blog/decoding-instagram-system-design-architecture-and-how-reels-recommendation-works/

[2] Vatesh Pasrija, Supriya Pasrija, "The Cold-Start Problem In Recommender Systems: Challenges And Mitigation Techniques," International Research Journal of Modernization in Engineering Technology and Science, Volume:06/Issue:05/May-2024. [Online]. Available: https://www.irjmets.com/uploadedfiles/paper//issue_5_may_2024/55701/final/fin_irjmets1715656884.pdf

[3] Paul Covington et al., "Deep Neural Networks for YouTube Recommendations," RecSys '16: Proceedings of the 10th ACM Conference on Recommender Systems, Pages 191 - 198, 2016. [Online]. Available: https://dl.acm.org/doi/10.1145/2959100.2959190

[4] Brent Smith and Greg Linden, "Two Decades of Recommender Systems at Amazon.com," IEEE Internet Computing (Volume: 21, Issue: 3, May-June 2017), 2017. [Online]. Available: https://ieeexplore.ieee.org/document/7927889

[5] Hao Chen et al., "Macro Graph Neural Networks for Online Billion-Scale Recommender Systems," arXiv:2401.14939v2 [cs.IR] 8 May 2024. [Online]. Available: https://arxiv.org/pdf/2401.14939

[6] Deepak Agarwal et al., "LASER: A Scalable Response Prediction Platform for Online Advertising," WSDM '14: Proceedings of the 7th ACM international conference on Web search and data mining, Pages 173 - 182, 2014. [Online]. Available: https://dl.acm.org/doi/10.1145/2556195.2556252

[7] Marco Bressan et al., "The Limits of Popularity-Based Recommendations, and the Role of Social Ties," KDD '16: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Pages 745 - 754, 2016. [Online]. Available: https://dl.acm.org/doi/10.1145/2939672.2939797

[8] Chantat Eksombatchai et al., "Pixie: A System for Recommending 3+ Billion Items to 200+ Million Users in Real-Time," WWW '18: Proceedings of the 2018 World Wide Web Conference, Pages 1775 - 1784, 2018. [Online]. Available: https://dl.acm.org/doi/10.1145/3178876.3186183

[9] Maksims Volkovs et al., "Content-based Neighbor Models for Cold Start in Recommender Systems," RecSys Challenge '17: Proceedings of the Recommender Systems Challenge 2017, Article No.: 7, Pages 1 - 6, 2017. [Online]. Available: https://dl.acm.org/doi/10.1145/3124791.3124792

[10] Michal Aharon et al., "OFF-set: one-pass factorization of feature sets for online recommendation in persistent cold start settings," RecSys '13: Proceedings of the 7th ACM conference on Recommender systems, Pages 375 - 378, 2013. [Online]. Available: https://dl.acm.org/doi/10.1145/2507157.2507221