Pinnacle Academic Press
Proceedings Series

Vol. 2 2025

Article **Open Access**

Research on the Design and Implementation of an App Recommendation System Based on User Behavior

Yue Xu [1,*]

[1] Salesforce Service Cloud, Redmond, 98052, USA
* Correspondence: Yue Xu, Salesforce Service Cloud, Redmond, 98052, USA

Received: 02 April 2025
Revised: 18 April 2025
Accepted: 10 May 2025
Published: 12 June 2025

Abstract: With the rapid development of mobile internet, personalized recommendation systems have become increasingly important in various applications. This paper designs and implements an APP recommendation system based on user behavior data. First, it elaborates on the basic concepts and core technologies of recommendation systems, analyzes user behavior characteristics and their impact on recommendation accuracy, and proposes algorithms based on collaborative filtering, content-based filtering, and hybrid recommendation. On this basis, the paper constructs the system architecture, develops the recommendation engine, and verifies the effectiveness of the system through performance testing and experimental data. The results show that the APP recommendation system based on user behavior significantly improves recommendation accuracy and enhances the user experience. Finally, the paper evaluates the practical application effects of the system and provides prospects for future research directions.

Keywords: user behavior; recommendation system; collaborative filtering; hybrid recommendation algorithm; personalized recommendation

1. Introduction

In the era of information explosion, the number of mobile applications (APPs) has grown exponentially, making it difficult for users to quickly find apps that meet their needs among the overwhelming amount of information. Personalized recommendation systems, as an effective tool to address the issue of information overload, can provide precise app recommendations based on user interests, behaviors, and preferences. In recent years, with the continuous development of big data and machine learning technologies, recommendation systems based on user behavior have gained wide research attention and application in various scenarios. However, traditional recommendation systems often rely on explicit user data such as ratings and reviews, neglecting the value of implicit data like user behavior. Behavioral data, such as browsing, clicking, and downloading activities, can more accurately reflect user interests and preferences, making recommendation systems based on user behavior a hot topic in research. This study aims to design and implement an APP recommendation system based on user behavior by collecting and analyzing user behavior data, constructing user interest models, and combining collaborative filtering, content-based, and hybrid recommendation algorithms to provide personalized recommendations. The research not only delves into recommendation algorithms but also designs a complete system architecture, performs testing and optimization, and validates the system's effectiveness and applicability. This study not only offers new ideas

for further optimizing recommendation systems but also provides useful technical support for enhancing the user experience in practical applications [1].

2. Overview of the APP Recommendation System Based on User Behavior

With the proliferation of the mobile internet and the explosive growth of apps, users face the challenge of information overload. Providing users with personalized and accurate app recommendations is crucial to enhancing their experience. Recommendation systems, as an important tool to solve this problem, utilize intelligent data analysis techniques to recommend apps that match users' interests. In this context, recommendation systems based on user behavior play a pivotal role by analyzing user behavior data to infer user preferences and thereby achieve personalized recommendations [2].

Figure 1 illustrates a typical workflow of an APP recommendation system based on user behavior.

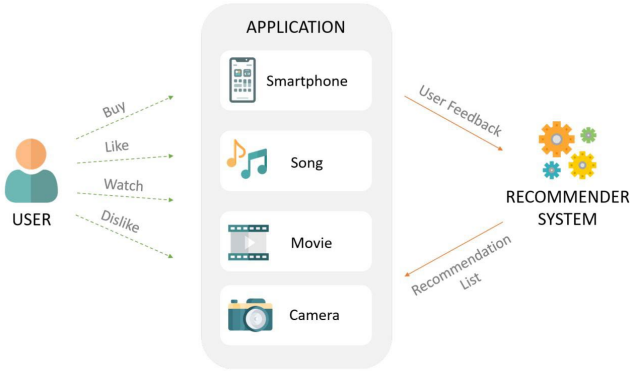

Figure 1. Workflow of a Recommendation System Based on User Behavior.

While using various applications, the system records multiple types of user behavior, including purchasing behaviors, viewing habits, likes, and dislikes. These data not only include explicit behaviors (e.g., explicitly liking or purchasing an app) but also implicit behaviors (e.g., frequently viewing a particular type of video or browsing apps within certain categories). Through deep mining of this behavioral data, the recommendation system can infer users' interests from multiple dimensions. The recommendation system uses user behavior data as input, processes and analyzes it through various algorithm models, such as collaborative filtering, content-based recommendation, or hybrid algorithms, to generate personalized app recommendation lists. Collaborative filtering recommends apps based on the behavior data of similar users, while content-based recommendation matches apps' characteristics with the user's past preferences. Hybrid recommendation algorithms combine the strengths of these two methods to deliver more accurate and adaptable recommendations. After generating the recommendation list, the system feeds the results back to the user. The user's further interactions (e.g., clicking, using, purchasing, etc.) are recorded again as new behavior data and fed back into the recommendation system. In this way, the system continuously adjusts and optimizes itself by obtaining user feedback, forming a closed-loop feedback system. This behavior-based loop mechanism ensures that the recommendation system can continually optimize according to changes in user behavior, providing personalized recommendations that better meet the user's current needs. The design goal of this system is to achieve efficient and accurate recommendations, reducing the time and effort users spend filtering information, and enhancing their satisfaction with the recommended content. Meanwhile, as user behavior data accumulates, the system becomes more intelligent, maintaining a high degree of recommendation accuracy even as users' needs dynamically change. In summary, a recommendation system based on user behavior not only enhances the user experience but also provides app developers with valuable market opportunities by effectively pushing the

right apps to the right users. Figure 1 visually illustrates the interaction flow between users and the recommendation system, demonstrating how the system collects, feeds back, and analyzes user behavior data in a closed-loop process, driving continuous iteration and optimization of recommendation algorithms to achieve more accurate APP recommendations [3,4].

3. User Behavior Analysis Model

3.1. Data Collection and Preprocessing

Figure 2 illustrates the complete flow of user behavior analysis. The system collects user behavior data from multiple channels, including registration information, browsing records, search logs, and consultation information. To ensure data availability and consistency, this raw data must undergo a series of preprocessing steps. The preprocessing process includes data integration, transformation, encoding, cleaning, and clustering. These steps help eliminate noise data, ensure data accuracy, and provide a solid foundation for subsequent analysis [5].

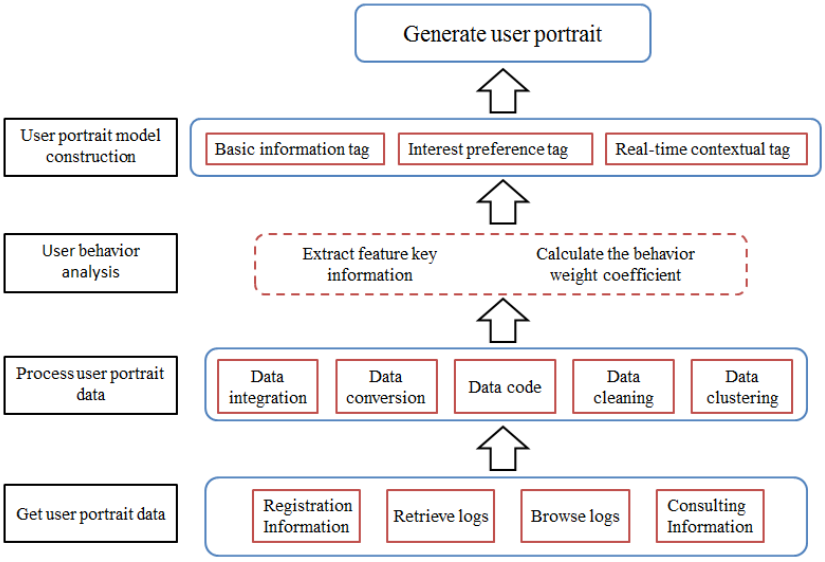

Figure 2. User Portrait Generation and Behavior Analysis Flowchart.

In the data integration phase, data from different sources are unified into a structured framework, ensuring that various behavior data can be correlated. Next, during data transformation and encoding, the raw data is converted into formats suitable for computational analysis, including converting unstructured data (such as text) into structured data. In the data cleaning process, incomplete or invalid data are filtered out, and anomalies are removed. Finally, through data clustering algorithms, the system identifies different user behavior patterns and classifies users into groups with similar behavioral traits.After preprocessing, the behavior data is used to construct the user portrait model. The model in Figure 2 emphasizes the generation of three key labels: basic information labels, interest preference labels, and real-time contextual labels. Basic information tags are generated from the user's registration information and general operational habits, while interest preference tags are derived from long-term browsing and search behaviors. Real-time contextual tags focus on the user's dynamic behavior, such as current operating scenarios or recent activity features. These tags make the user portrait more comprehensive and provide the recommendation system with more accurate reference data. Throughout the user behavior analysis process, extracting key feature information and calculating behavior weight coefficients are crucial steps. By deeply analyzing user behav-

ior patterns, the system can more accurately predict users' potential needs, thereby generating personalized recommendations. In conclusion, the collection and preprocessing of user behavior data form the foundation for constructing user portraits and generating accurate recommendations in recommendation systems. The process outlined in Figure 2 clearly illustrates the complete path from data acquisition to user portrait generation, ensuring that the system can continuously and dynamically optimize recommendation results [6].

3.2. User Behavior Feature Extraction and Analysis

User behavior feature extraction and analysis are critical steps in building a precise recommendation system. By deeply mining users' behavior data within applications, the system can extract key user characteristics and generate user portraits, enhancing the personalization and accuracy of recommendations. The goal of feature extraction is to identify the most representative data points from a large volume of behavior data that reflect user interests and preferences. These features provide strong support for the performance of recommendation algorithms. As shown in Figure 2, after user behavior data is preprocessed, the system performs feature extraction. First, the system extracts user behavior patterns from different dimensions, such as purchase history, browsing duration, click frequency, and search keywords [7]. These behavior features include both explicit feedback (e.g., "purchase", "like") and implicit feedback (e.g., passive browsing or time spent on content), collectively reflecting user interests. By analyzing these features, the system can grasp user preferences for different types of applications. Next, the system weights these features and calculates the importance of each behavior characteristic. For example, purchase behavior is typically given higher weight than simple browsing because purchases directly indicate user preferences. The system adjusts these weight coefficients through historical data analysis and model training, ensuring that the features have a more accurate influence on the recommendation results. After feature extraction, the system clusters and performs correlation analysis on user behavior features [8]. This step groups users with similar behavior patterns, improving the relevance of recommendations. Based on clustering results, the system can identify user groups with similar characteristics and recommend apps favored by other users in the same group. Additionally, correlation analysis identifies potential links between different user behaviors, such as how a preference for one type of app may correlate with engagement in another category. By analyzing these correlations, the system can recommend apps that the user has not yet encountered but may find interesting. Moreover, feature extraction for user behavior requires dynamic analysis considering the time dimension [9]. As user interests and behaviors may change over time, the system needs to update user behavior features periodically. By continuously collecting and analyzing real-time behavior data, the system can dynamically adjust user portraits to ensure that recommended content always aligns with the user's current interests. In summary, user behavior feature extraction and analysis provide in-depth data support for recommendation systems. By extracting key behavior features, calculating weights, performing clustering analysis, and making dynamic adjustments, the system can generate more precise user portraits, leading to highly personalized and accurate recommendations. This process not only improves user satisfaction with recommended content but also enhances user engagement and long-term app usage [10].

4. Design and Implementation of the Recommendation System Architecture Based on User Behavior

4.1. System Overall Architecture Design

The architecture of the recommendation system is designed to process and analyze user behavior data to generate real-time personalized recommendations. Figure 3 presents the overall system architecture, covering the complete workflow from data collection, pro-

cessing, computation, to application. In this architecture, data collection forms the foundation, where the system gathers user behavior data through multiple channels, including browsing history, interest data, historical behavior, and review data. This behavior data is a key component for building user profiles, which serve as the basis for generating subsequent recommendations [11].

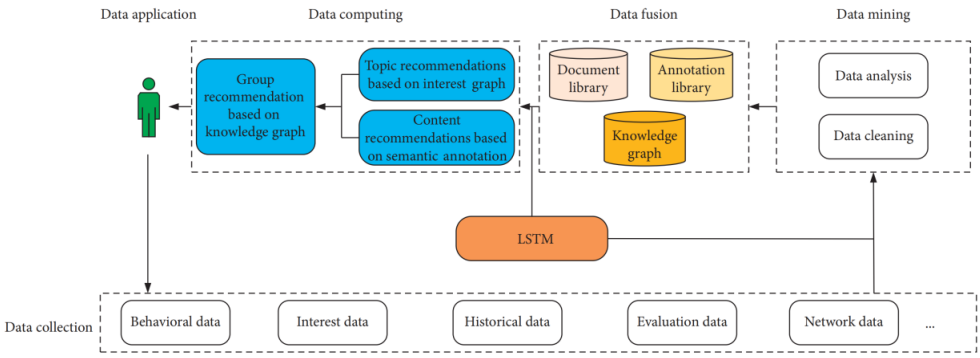

Figure 3. Recommendation System Architecture Based on Knowledge Graph and LSTM.

During the data mining phase, the system first cleans and analyzes the collected raw data. Data cleaning ensures the accuracy and integrity of the dataset by removing invalid or anomalous entries. In the analysis process, The system deeply explores hidden behavioral patterns to identify core user preferences, such as preferred app categories or usage scenarios. This step provides high-quality data support for the recommendation algorithms. Next, the data fusion module is one of the core components of this architecture. Figure 3 illustrates the application of knowledge graphs, document libraries, and annotation libraries in the data fusion process. The introduction of the knowledge graph allows the system to semantically associate user behavior with application content. This enables the recommendation process to go beyond surface-level behavior and generate more accurate recommendations by understanding the relationships between content. This semantic connection allows the system to generate intelligent and contextually relevant recommendations by understanding the meaning behind user interactions, rather than relying solely on pattern repetition from historical behavior. The data computation module uses a Long Short-Term Memory (LSTM) model to predict user behavior. LSTM is a neural network model capable of handling time-series data and is particularly effective in capturing short-term variations and long-term trends in user behavior. Using the LSTM model, the system predicts future user actions by identifying behavioral trends and patterns over time. These predictions allow the system to update recommendations promptly as the user's interests evolve. This functionality enables the system to provide personalized recommendations dynamically, improving the timeliness and accuracy of recommendations. Finally, the data application module feeds the generated recommendations back to the user. As shown in Figure 3, the system not only provides personalized recommendations for individual users but also performs group analysis to generates group-based recommendations. This group-based recommendation approach takes into account aggregate user behavior trends within similar user clusters, balancing individual preferences with broader group tendencies, ensuring users receive content that aligns with both their personal preferences and broader group trends. Through this comprehensive architecture, the recommendation system optimizes every step from data collection to recommendation generation. The combination of the knowledge graph and LSTM model enhances the system's intelligence, allowing it to respond precisely to dynamic user behavior. In summary, this architecture is designed to account for the complexity and diversity of user behavior, providing a full-spectrum recommendation service that addresses both static interests and dynamic behavior [12].

4.2. Recommendation Algorithm Based on User Behavior

In a recommendation system based on user behavior, the core recommendation algorithms aim to analyze user behavior data and generate personalized recommendations. To improve the accuracy and timeliness of recommendations, this system combines collaborative filtering algorithms, content-based recommendation algorithms, and the Long Short-Term Memory (LSTM) model to comprehensively cover different user behavior patterns. Collaborative filtering, a classic recommendation approach, includes user-based and item-based variants. This algorithm generates recommendations based on the similarities between users or between items. In the user-item interaction matrix, the collaborative filtering algorithm recommends items to a target user based on the behavior of similar users. User similarity is typically calculated using the cosine similarity formula 1:

$$\mathrm{Sim}(u,v) = \frac{\sum_{i \in I_{uv}} r_{u,i} \cdot r_{v,i}}{\sqrt{\sum_{i \in I_u} r_{u,i}^2} \cdot \sqrt{\sum_{i \in I_v} r_{v,i}^2}} \quad (1)$$

Where $r_{u,i}$ and $r_{v,i}$ represent the ratings of user u and user v for item i, $Sim(u,v)$ represents the cosine similarity between user u and user v. and I_{uv} represents the set of items rated by both users u and v. By calculating similarity, the system identifies users whose behavior resembles that of the target user. Based on the preferences of these similar users, it recommends applications that the target user has not yet explored. The content-based recommendation algorithm primarily analyzes the features of items and matches them with the items the user has liked in the past. By extracting feature vectors of the items (e.g., categories, functions, descriptions of apps), the system calculates the similarity between the user's interest vector and the item's feature vector to generate recommendations. The similarity can also be calculated using the cosine similarity formula 2:

$$\mathrm{Sim}(u,i) = \frac{\sum_{f \in F} w_f^u \cdot w_f^i}{\sqrt{\sum_{i \in F} (w_f^u)^2} \cdot \sqrt{\sum_{i \in F} (w_f^i)^2}} \quad (2)$$

Where w_f^u represents the weight of user preference for feature f, and w_f^i represents the weight of item i on that feature. $\mathrm{Sim}(u,i)$ is the similarity score between the user u and the item i. F represents the set of all features that describe both the user and the item. By analyzing the content the user has already browsed or liked, the system can recommend other items with similar features. To capture the temporal features of user behavior, this system introduces the Long Short-Term Memory (LSTM) model. LSTM is a recurrent neural network (RNN) designed to handle time-series data, effectively retaining long-term and short-term trends in user behavior. The core function of LSTM is to learn temporal dependencies within the user's behavior sequence, enabling it to capture both immediate patterns and long-term behavioral trends for more accurate next-action prediction. The core formulas of the LSTM model are as shown in Formula 3-5:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f \quad (3)$$

$$C_t = f_t * C_{t-1} + i_t * \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \quad (4)$$

$$h_t = o_t * \tanh(C_t) \quad (5)$$

Where f_t is the forget gate, C_t is the cell state, σ is the sigmoid activation function, W_f is the forget gate's weight matrix, h_{t-1} is the previous hidden state, x_t is the current input data, and b_f is the forget gate's bias term. i_t is the input gate's output, W_C is the input gate's weight matrix, and b_C is the input gate's bias term. o_t is the output gate's output. The LSTM model can predict the next action based on the user's historical behavior pattern, allowing the system to provide real-time recommendations. By combining these three core algorithms, the recommendation system can comprehensively analyze user behavior data. Collaborative filtering captures the similarities between users, content-based recommendation matches item features to recommend new content, and the LSTM model analyzes user behavior over time, ensuring that the system can adjust recommendations dynamically to better align with each user's evolving preferences.

5. Experimental Design and Data Analysis

To evaluate the effectiveness of the recommendation system based on user behavior in practical applications, we designed a series of experiments aimed at analyzing the performance of different recommendation algorithms. The experiments focused on verifying the system's accuracy, recall rate, and user satisfaction, the latter of which was measured through post-recommendation user surveys and engagement metrics such as average session duration and click-through rate. The experiments were conducted using data from a mobile application platform that includes a large amount of user behavior data, such as browsing history, download records, ratings, and other interactions. The primary focus was on comparing the performance of collaborative filtering algorithms, content-based recommendation algorithms, and the LSTM model. The system's performance was evaluated through a comprehensive analysis across multiple metrics, including accuracy, recall, F1 score, and user feedback. In the experiments, we first preprocessed the raw data, including data cleaning, normalization, and noise filtering. The dataset consisted of 100,000 interaction records from 1,000 users, covering 5,000 mobile applications. To ensure objectivity and representativeness in the experimental results, we used 10-fold cross-validation to test the performance metrics of each algorithm. The performance of the recommendation system was primarily measured by accuracy, recall rate, F1 score, and user satisfaction.First, we evaluated the effectiveness of collaborative filtering algorithms by comparing user-based and item-based collaborative filtering. The Table 1 shows the comparison results of different algorithms in terms of accuracy, recall rate, and F1 score:

Table 1. Experimental Data and Results Analysis.

Algorithm Type	Precision	Recall	F1 Score
Collaborative Filtering (User)	0.78	0.65	0.71
Collaborative Filtering (Item)	0.81	0.70	0.75
Content-Based Recommendation	0.83	0.72	0.77
LSTM	0.85	0.76	0.80

From Table 1, it is evident that item-based collaborative filtering slightly outperforms user-based collaborative filtering in all metrics, particularly with a precision of 0.81 and an F1 score of 0.75. Content-based recommendation further improves both accuracy and recall, especially in capturing user preference changes, achieving a precision of 0.83 and an F1 score of 0.77. The LSTM model shows the best performance, thanks to its ability to capture the temporal dynamics of user behavior. It leads in all metrics, with a precision of 0.85 and an F1 score of 0.80, demonstrating its superior ability to predict future user behavior. To further validate the performance of the recommendation system, we also tested its performance in terms of recommendation time, user satisfaction, and hit rate. The Table 2 below presents the comparison data for different algorithms:

Table 2. Experimental Results Comparison.

Experiment Result	Collaborative Filtering (User)	Collaborative Filtering (Item)	Content-Based Recommendation	LSTM Model
Average Recommendation Time (seconds)	0.25	0.27	0.31	0.35
User Satisfaction Score (1-5)	3.8	4.0	4.2	4.5
Recommendation Hit Rate (%)	65	70	72	76
Experiment Result	Collaborative Filtering (User)	Collaborative Filtering (Item)	Content-Based Recommendation	LSTM Model

As shown in the Table 2, the LSTM model leads in user satisfaction (4.5) and hit rate (76%). Although the LSTM's recommendation time is slightly longer (0.35 seconds), it provides more accurate recommendations, and user satisfaction with the results is significantly higher. Content-based recommendation also performs well, with a satisfaction

score of 4.2 and a hit rate of 72%. In comparison, collaborative filtering algorithms have shorter recommendation times but slightly lower recommendation accuracy and user satisfaction, particularly in dynamic recommendation scenarios where the LSTM model excels. To comprehensively analyze the effects of different algorithms in real-world applications, we further examined the recommendation system's performance across different user groups, particularly active and inactive users. The Table 3 below compares the performance of different algorithms in these two user groups:

Table 3. System Performance Analysis.

User Group	Algorithm Type	Precision	Recall	F1 Score
Active Users	Collaborative Filtering (User)	0.80	0.68	0.74
	Content-Based Recommendation	0.84	0.75	0.79
	LSTM	0.87	0.78	0.82
Inactive Users	Collaborative Filtering (User)	0.72	0.60	0.65
	Content-Based Recommendation	0.78	0.66	0.71
	LSTM	0.82	0.70	0.76

From Table 3, it can be seen that for active users, the LSTM model performs best, with a precision of 0.87 and an F1 score of 0.82, indicating its superior ability to predict user behavior. For inactive users, although the overall recommendation effectiveness is lower, the LSTM model still leads with a precision of 0.82. In comparison, collaborative filtering algorithms perform relatively weaker when dealing with inactive users, indicating limitations in handling sparse data. Based on the experimental data and analysis, we can conclude that the recommendation system based on user behavior performs differently in various scenarios. Collaborative filtering algorithms are suitable for scenarios where user similarity is high and sufficient user behavior data is available, while content-based recommendation algorithms provide more precise recommendations by matching application features. The LSTM model, however, excels in capturing the temporal dynamics of user behavior and performs best when dealing with active users. Despite the slightly higher computational complexity and recommendation time of the LSTM model, its significant advantages in recommendation accuracy, user satisfaction, and hit rate make it the best choice for personalized recommendation systems. Overall, the experiment validated the effectiveness of various recommendation algorithms in analyzing user behavior data and highlighted the strengths and weaknesses of each. Given its sensitivity to dynamic user behavior, the LSTM model shows great potential for future applications in recommendation systems.

6. Conclusion

This study focused on the design and implementation of an architecture and algorithms for a recommendation system based on user behavior, and the effectiveness of the system was validated through experiments. The results showed that collaborative filtering algorithms are suitable for scenarios where user similarity is high, while content-based recommendation algorithms excel in personalized recommendations. The LSTM model, with its ability to capture the temporal dynamics of user behavior, significantly improves recommendation accuracy and user satisfaction. Although the LSTM model has relatively higher computational complexity, its performance in dynamic scenarios is the best, particularly in terms of hit rate and user satisfaction, outperforming other algorithms. Thus, the LSTM model has proven to be the ideal choice for dynamic recommendation systems. Future work can focus on further optimizing hybrid recommendation algorithms by combining collaborative filtering, content-based recommendation, and the LSTM model to enhance the overall system performance and address more complex user behavior scenarios.

References

1. J. Lopez-Barreiro, J. L. Garcia-Soidan, L. Alvarez-Sabucedo, and J. M. Santos-Gago, "Practical approach to designing and implementing a recommendation system for healthy challenges," *Appl. Sci.*, vol. 13, no. 17, p. 9782, 2023, doi: 10.3390/app13179782.
2. A. M. Honka, H. Nieminen, H. Similä, J. Kaartinen, and M. V. Gils, "A comprehensive user modeling framework and a recommender system for personalizing well-being related behavior change interventions: Development and evaluation," *IEEE Access*, vol. 10, pp. 116766–116783, 2022, doi: 10.1109/ACCESS.2022.3218776.
3. M. Zhong and R. Ding, "Design of a personalized recommendation system for learning resources based on collaborative filtering," *Int. J. Circuits Syst. Signal Process.*, vol. 16, no. 1, pp. 122–131, 2022, doi: 10.46300/9106.2022.16.16.
4. S. Sengan et al., "A secure recommendation system for providing context-aware physical activity classification for users," *Secur. Commun. Netw.*, vol. 2021, no. 1, Art. no. 4136909, 2021, doi: 10.1155/2021/4136909.
5. J. Jiang and H. H. Wang, "Application intelligent search and recommendation system based on speech recognition technology," *Int. J. Speech Technol.*, vol. 24, pp. 23–30, 2021, doi: 10.1007/s10772-020-09703-0.
6. A. Pinto et al., "Recommendation systems to promote behavior change in patients with diabetes mellitus type 2: A systematic review," *Expert Syst. Appl.*, vol. 231, Art. no. 120726, 2023, doi: 10.1016/j.eswa.2023.120726.
7. T. Rocha, E. Souto, and K. El-Khatib, "Functionality-based mobile application recommendation system with security and privacy awareness," *Comput. Secur.*, vol. 97, Art. no. 101972, 2020, doi: 10.1016/j.cose.2020.101972.
8. C. Udokwu et al., "Design and implementation of a product recommendation system with association and clustering algorithms," *Procedia Comput. Sci.*, vol. 219, pp. 512–520, 2023, doi: 10.1016/j.procs.2023.01.319.
9. S. Beg et al., "A privacy-preserving protocol for continuous and dynamic data collection in IoT enabled mobile app recommendation system (MARS)," *J. Netw. Comput. Appl.*, vol. 174, Art. no. 102874, 2021, doi: 10.1016/j.jnca.2020.102874.
10. A. Sayed, Y. Himeur, A. Alsalemi, F. Bensaali, and A. Amira, "Intelligent edge-based recommender system for internet of energy applications," *IEEE Syst. J.*, vol. 16, no. 3, pp. 5001–5010, Sept. 2022, doi: 10.1109/JSYST.2021.3124793.
11. Z. Cui et al., "Personalized recommendation system based on collaborative filtering for IoT scenarios," *IEEE Trans. Serv. Comput.*, vol. 13, no. 4, pp. 685–695, Jul.–Aug. 2020, doi: 10.1109/TSC.2020.2964552.
12. H. Ko, S. Lee, Y. Park, and A. Choi, "A survey of recommendation systems: Recommendation models, techniques, and application fields," *Electronics*, vol. 11, no. 1, p. 141, 2022, doi: 10.3390/electronics11010141.