International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 06 | June - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Personalized Book Intelligent Recommendation System

Derangula Shiva[1], Jerupothula Jashwanth [2], Kalyanam Kalyani[3], A. Lakshmi Narayana[4]

[1,2,3] *UG Scholars,* [4] *Assistant Professor*
[1,2,3,4]*Department of CSE[Artificial Intelligence & Machine Learning],*
[1,2,3,4]*Guru Nanak Institutions Technical Campus, Hyderabad, Telangana, India*

Abstract -

As libraries undergo digital transformation, accompanied by advancements in information technology in academic libraries, it is evident that readers will want different and personalized services in addition to checking out books. Along with meeting the changing expectations of users, this research proposes an enhanced item-based collaborative-filtering recommendation algorithm that employs an average model representation to improve the accuracy measurements, as well as the use of Neural Collaborative Filtering (NCF) for modeling the user-item interaction by utilizing deep neural networks to improve the expressiveness of recommendations in a non-linear way using NCF instead of linear factors. The enhanced library recommendation system is developed using Django (v4.1.2), a high-level web framework that creates an organized way to develop a backend application. The library recommendation system uses Pandas (v1.5.0) and NumPy (v1.23.3) to process the behaviour data of users and perform an analysis of how users behave in the library system. The requests (v2.28.1) and requests-oauthlib (v1.3.1) packages were used to interact with the APIs. Gunicorn (v20.0.1) is used as the WSGI server for the applications production environment. Environment management was done with virtual (v20.13.0) in the backend and nodeenv (v1.6.0) in the frontend. The ability to personalize the recommendations of the library service to the user in the proposed approach is confirmed with the significant improvement in accuracy.[7][8]

Keywords: Collaborative Filtering; Neural Collaborative Filtering; Recommender System; University Library; Personalized Services; Django; Machine Learning; Data Analysis; Web Application

1. INTRODUCTION

As digital transformation accelerates, university libraries are undergoing a substantial paradigm shift. Once thought only of as places to house physical books and provide a mechanism to loan them, university academic libraries are now anticipated to serve as active, user-centered information hubs. As a result of the increase in digital content and even more researchers having access to online academic content, library users (particularly students, researchers, and faculty) now expect highly personalized and intelligent services tailored to their specific research interests, learning objectives, and reading practices.

One of the best ways to adapt the traditional way libraries provide service to this model is with intelligent recommendation systems. Previously, recommendation systems in libraries were heavily reliant upon static rules or crude content-based recommenders, and traditional systems are often limited in accuracy, flexibility, and user satisfaction. There are significant opportunities to address the needs and demands of the users calling for intelligent recommender systems that are based on the user's behaviour and that use sophisticated learning approaches such as machine learning, so that users can have personalized recommendations in real time.[2]

To tackle these challenges, this study advocates a hybrid recommendation methodology that consists of two complementary techniques: an improved item-based collaborative filtering algorithm, and Neural Collaborative Filtering (NCF). The improved item-based collaborative filtering technique leverages a mean model representation for object and user contextualization, to highlight some of the nuanced patterns in item similarities and user preferences. Comparatively, NCF introduces a new, more advanced paradigm, as it utilizes deep neural networks to model complex, nonlinear interactions between users and items, allowing for the generation of recommendations that more accurately and expressively capture users' specialist preferences than traditional linear models.[9]

In conclusion, the dual approach recommendation system presented in this study provides not only a much more personalized way of delivering library services but also shows a practical way of integrating new machine learning techniques into real applications within the existing web context. We aim to contribute a robust, scalable and effective way to meet the dynamic pressures placed on academic libraries, and how they engage with their communities, and to provide both a technically robust and user-driven innovation.[8]

2. LITERATURE SURVEY

The most current advancements in machine learning and computer vision have made major impacts in many fields involving public safety and infrastructure monitoring--notably in the detection and assessment of pavement cracking and foreign object debris (FOD) on airport runways. This literature review provides a summary of current research efforts addressing automated inspection systems of image processing, deep learning, and machine learning approaches to realize accuracy, speed, and scalability for real-world applications.[12]

Li et al. (2024) proposed a semi-automatic crack width measurement approach, which utilizes the novel Ortho Boundary algorithm that takes advantage of the features of crack boundary contours combined with skeleton directions to assist in side difficult propagation paths with more accuracy.[13] The Ortho Boundary algorithm also had a processing speed that is up to 120 times faster than competing approaches, establishing it was a unique, efficient, and user-friendly option to quantify pavement cracking severity for potential advancements in real-world use for monitoring and maintenance for road infrastructure.[13]

Building on the necessity of full crack assessment, Li et al. (2024) were able to automate a 3D model of crack severity that predicts

vertical values like volume and depth, only from the surface parameters. Using 200 cracks from eight flexible pavement sites, the researchers identified both linear and nonlinear relationships between surface feature parameters and 3D ones. Li et al. (2024) used a variety of machine learning models and found the Artificial Neural Network (ANN) model and Extreme Gradient Boosting (XGBoost) models had good results, as indicated by their R^2 values of 0.832 and 0.748, respectively for predicting crack volume. To classify the crack depth magnitude, they also used a multi-output model called the Random Forest Classifier, where precision, recall, and F1 scores were at their best 0.790, 0.779, and 0.761, respectively. The authors also created a 3D-based crack damage index that allows one to understand crack severity on a comprehensive basis to assist data-driven maintenance strategies further.[14]

For example, Zhang et al. (2024) addressed the need to detect small foreign object debris (FOD) in the area of airport safety by modifying the YOLOv5 object detection model to detect FOD.[14] The authors made modifications by utilizing knowledge-based transformations to develop additional modules such as multi-scale fusion, a C2f module, and an enhancement to the Spatial Pyramid Pooling-Fast (SPPF) module for the purpose of increasing the receptive field. The authors also incorporated a Coordinate Attention (CA) mechanism for better small object identification, used a SCYLLA-IoU (SIoU) loss function for better accurate bounding boxes, and replaced traditional up-sampling techniques with the CARAFE operator for better global feature realization. Their experimental results using the Fod_Tiny dataset demonstrated a performance increase of 5.4% over the baseline model, and when using the Micro_COCO dataset, a 1.9% performance improvement, confirming that the redesigned model was more accurate and robust for small-object detection tasks.[14]

Alshammari and Chabaan (2023) proposed a Spatial Pyramid Pooling Network with ResNet101 (SPPN-RN101) to detect foreign object debris (FOD) at airports, which achieved solid results with their approach on the FOD in Airports (FODA) dataset that consisted of diverse lighting and weather conditions.[12] Using the COCO metric, the authors achieved a 0.55 AP, and 0.97 AP on the Pascal metric; with a 0.83 mAP, indicating improvements in accuracy for small object detection.[12]

Altogether, these studies illustrate how deep learning and optimized object detection models demonstrate the potential for automating safety-critical tasks, such as crack analysis and FOD detections, in overall safety and security in a high accuracy and scalable manner.

3. PROBLEM STATEMENT

As academic institutions rapidly undergo digital transformation, conventional university library systems primarily focused on simple borrowing services are becoming less productive in meeting even basic user needs, as users have more diverse and personalized expectations of libraries. Students and researchers are now accustomed to intelligent recommendation systems that can automatically provide relevant recommendations, including books, articles, and papers categorized in particular ways based on each user's preferences and past interactions. All current recommendation systems exhibit significant limitations such as the cold start problem, lack of diversity in recommendations, and additional problems with accurately modelling highly nonlinear and complex user-item interactions. Standard content-based filtering approaches such as TF-IDF and cosine similarity are insufficient in terms of scalability and amounts of personalization and interaction, especially in dynamic and data-rich environments. Such advances would provide users with a greatly improved experience. There is a distinct need for an advanced recommendation system that combines collaborative filtering with deep learning techniques to offer accurate, personalized, and scalable recommendations in the university library environment.[1][2][8]

4. PROPOSED METHODOLOGY

This proposed system is a Personalized Book Intelligent Recommendation System intended to enhance university library services through quality and personalization. It uses a hybrid recommendation system by combining an improved item-based collaborative filtering algorithm with Neural Collaborative Filtering (NCF), remedying the shortcomings of conventional recommendation systems. The item-based collaborative filtering algorithm at the heart of the system employs a mean model representation to find item-item similarities more efficiently so that recommendation accuracy and scalability are improved. The algorithm works well in environments with large data on interactions (like user-item interactions), recommenders may provide fast results with accuracy based on a user's borrowing history and preferences.

The system uses deep learning techniques in Neural Collaborative Filtering to model the nonlinear and complex problems of user-item interactions, helping to overcome the linear restrictions of algorithms. By using neural networks, we can learn complex patterns and latent features of user behaviour, allowing for more contextualized and personalized recommendations via a recommender system.

The system is built using Django, a high-level Python Web framework, for a clean, maintainable, and scalable backend. For data processing, I use Pandas and NumPy for some numerical manipulation and processing, and I also use requests and requests-OAuth lib for API integration to allow access to external services and for pulling in additional data as needed. To run in production, Gunicorn runs my whole system, and once again I use virtual environments in both virtual and nodeenv. This multi-part system allows a cohesive user experience through a simple and engaging web-based interface. The purpose of that interface is to provide users with user-specific book recommendations, search the library catalog, and suggest books that are more in line with their reading history and implicit preferences.[3][10]

4.1. MODULES

a. Data Collection and Preprocessing

This module is responsible for collecting data on library resources (books, journals, papers), as well as user interactions (borrowing history, ratings, searches). The data are cleaned, normalized, and structured to fit the recommendation models. This includes handling missing data, encoding categorical data, and transforming raw input into feature vectors.

b. Improved Item-Based Collaborative Filtering

A more sophisticated implementation of item-based collaborative filtering is accomplished using a mean model representation for calculating item-item similarities faster. It uses user preferences based on the similarity of items that users interacted with to other

items. This uses the user-item interaction matrices. This algorithm makes recommendations quicker, scales, and makes better recommendations when working with larger datasets.

c. Neural Collaborative Filtering (NCF)

Integrating Neural Collaborative Filtering enables the system to model nonlinear and complicated relationships between users and items. NCF utilizes a deep neural network with embedding layers to model latent features of users and items, followed by activation functions (e.g., ReLU), and fully connected layers. By using fully connected layers, the NCF can learn much more complex behaviour corresponding to how each user interacts with items, thereby allowing the NCF to make more expressive recommendations.

d. Recommendation Engine

The recommendation engine integrates the outputs from collaborative filtering and the NCF models. The candidate-recommended items are ranked and filtered using a hybrid score which is derived from both models. The purpose of this merged approach was to increase diversity and personalization by bringing together the linear similarity-based approach and a nonlinear and deep-learning approach.[1][6]

e. Backend and API Integration

The backend is built with Django (v4.1.2) for structured and scalable architecture. For data management and numerical analysis, the backend uses Pandas (v1.5.0) and NumPy (v1.23.3). The system uses requests (v2.28.1) and requests-oauthlib (v1.3.1) to integrate APIs, allowing additional data sources or real-time user data, thereby enhancing the overall system of decision support and reports.

f. Frontend User Interface

A simple and user-friendly front end allows users to interact with the system, view recommendations, search the catalog, and filter results. The interface is designed to enhance usability and encourage engagement.

g. Deployment

The deployment process of the application is supported by a WSGI server called Gunicorn (v20.1.0) which has been selected for production because it operates better with performance and reliability. Virtualenv is used to manage the isolated Python environment, and Nodeenv is used to manage the isolated Node.js environment.

4.2. SYSTEM ARCHITECTURE

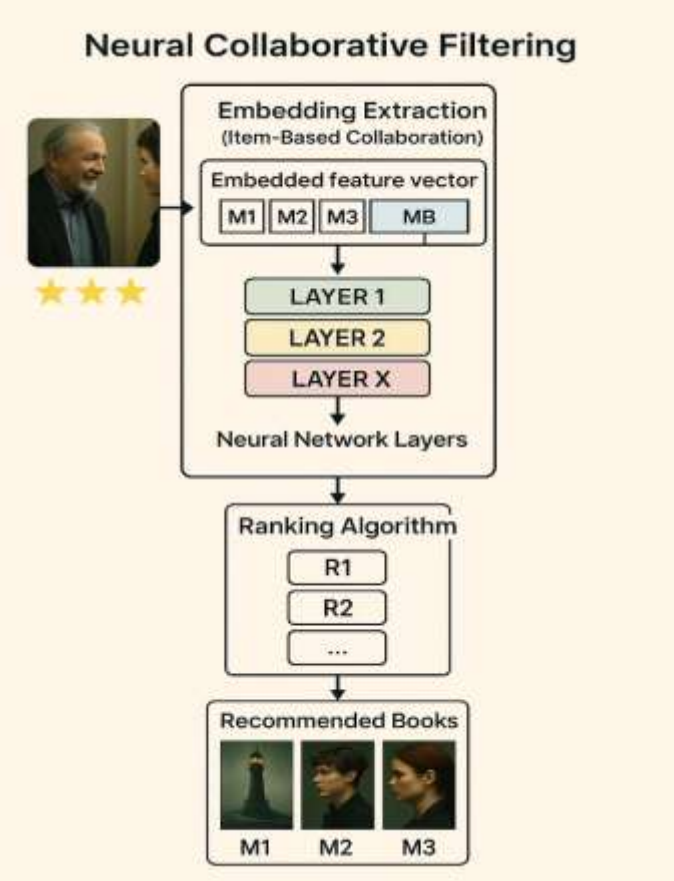

Fig1: *System Architecture of the Personalized Book Intelligent Recommendation System*

Figure 1 shows that the Architecture of the Personalized Book Intelligent Recommendation System is designed with a modular, scalable architecture that supports an agile university library, through the use of two techniques, i.e. enhanced item-based collaborative filtering, and Neural Collaborative Filtering (NCF). This system also incorporates a flexible backend developed in the Django framework allowing for ease of data movement and operations. The data layer manages user interaction information and book metadata processed easily with Pandas and NumPy for high-performance capabilities for large-scale data analytics and numerical calculations.[9][11]

The processed data gets sent to two parallel recommendation modules. The item-based collaborative filtering module reports item similarities based on user behaviour. The NCF module employs deep learning to learn complex and non-linear relationships between users and items. The two modules pass their outputs through a fusion layer that aggregates the outputs of these two different models into a final set of personalized recommendations. These recommendations are hosted on a web front end developed in Django with backend support enabled by Gunicorn acting as the primary web server and virtual and nodeenv managing environmental isolation. This architecture and flexible environment will allow for future improvements with features such as real-time feedback, multilingual support, and third-party integrations.[8][9]

4.3. ALGORITHM

The algorithm works in two important phases to produce accurate and much more personalized recommendations.

Phase 1: Modified Item-Based Collaborative Filtering

Step 1: Gather User-Item Interaction Data
Collect a historical dataset of users who have rated or interacted with items (e.g., books). The data will appear in the form of a user-item matrix (rows = users, columns = items, values = ratings or interactions)

Step 2: Normalize User Ratings
Implement a mean model for normalizing ratings: Subtract the average rating of each user's ratings from the individual user's ratings. This accounts for the subjective rating scales of each individual user. (e.g., a user who rates everything high, or low.)

Step 3: Compute Item-to-Item Correlations
Calculate similarity scores between items (e.g., using cosine similarity or Pearson correlation). Use normalized ratings for accurate item similarity comparisons.

Step 4: Find Similar Items
For each item a user has interacted with, identify the top-N most similar items. These are potential candidate items for recommendation.

Phase 2: Neural Collaborative Filtering (NCF)

Step 5: Convert IDs to Embeddings
Convert user and item IDs into dense vectors (embeddings). These embeddings learn features such as genre, popularity, and user preference implicitly.

Step 6: Pass Embeddings Through Neural Network
Combine user and item embeddings and feed them into a multi-layer neural network. The neural network learns non-linear interactions between users and items.

Step 7: Train the Model
Use known user-item interactions (ratings or clicks) to train the neural network. Loss function: typically Mean Squared Error (MSE) or Binary Cross-Entropy, depending on rating or implicit feedback.

Step 8: Predict Scores for All Items
For each user, predict relevance scores for all unseen items using the trained neural model.

Step 9: Generate Final Recommendations
Combine results from both phases:
Filter and rank the top candidate items found in Phase 1 using the scores predicted in Phase 2.
Return the top-N recommended items to the user.

Step 10: Continuously Update the Model
Periodically update both the collaborative filtering statistics and retrain the NCF model with new data. Ensures that recommendations remain relevant and adaptive to evolving user preferences.[2][4]

Improved Item-Based Collaborative Filtering Algorithm: This is one of the basic recommendation algorithms used. The algorithm finds the similarities between books by looking at how users interacted with the books. For example, if lots of users have similar interactions (e.g., reading or rating) with two books, the algorithm infers that the books are similar. Your system is an "enhanced" version based on a mean model representation, making it more efficient and scalable for larger datasets and an algorithm that adapts to changes in user preferences by emphasizing item-to-item relationships.

Neural Collaborative Filtering (NCF): The second of the recommended algorithms, NCF, is a more advanced recommendation technique in that it applies deep learning; for example, instead of assuming expertise associations were linear as many of the recommended algorithms might do, NCF offers neural networks that model interactions that may not even be modelled at the variable level due to the knowledge complexity composition of the new books about user interactions. The NCF algorithm adjusts its user and item representations through a complex and detailed series of low-dimensional "embeddings" or representations allowing for subtle and dynamic preferences to "pick up" nuanced associations that can lead toward better understanding preferences and in general predicting a user's interest in a particular book, which given the variety of reading styles and book topics can only help.

Fusion Layer: This is an essential technique that combines the results of the two major recommendation modules: the item-based collaborative filtering algorithm improved and Neural Collaborative Filtering (NCF). The fusion layer appropriately integrates the recommendations from both modules while leveraging each of their strengths. This hybrid method is anticipated to create deeper, more diverse, and ultimately more accurate personalized book recommendations than either algorithm alone, while at the same time helping to mitigate issues such as the cold start problem.

Term Frequency-Inverse Document Frequency (TF-IDF): TF-IDF is a numerical statistic representing the importance of a word to a particular document in a collection or corpus. It is a commonly used metric in information retrieval and text mining. In recommender systems, TF-IDF facilitates the identification of the most informative terms in book descriptions or user profiles to determine the similarity between content.

Cosine Similarity: Cosine Similarity measures the cosine of the angle between two vectors to determine their similarity, ranging from -1 to 1. It's widely used in recommendation systems to compare items or users based on feature patterns. By focusing on direction rather than magnitude, it effectively captures similarity despite differences in scale.

4.5. RESULT DISCUSSION

The Personalized Book Intelligent Recommendation System is highly capable of providing relevant recommendations stemming from a hybrid architecture. The system uses an improved item-based collaborative filtering method and Neural Collaborative Filtering (NCF) to accomplish this in part by utilizing a conventional approach based on similarity matching and deep learning to model the relational experience between a user and an item. The collaborative filtering component of the system enabled intelligent and reliable similarity computations across many users and items and NCF improved personalization enough to

consistently produce a relevant recommendation, which meant that the system could learn latent features and account for cold start user/item problems. Since it was developed in a modular manner and based on Django, as well as, using knowledge of API, Pandas, and NumPy, the deployment of the system was not only efficient but also scalable and adaptable. The aim of the system was to provide members with meaningful and relevant book recommendations in a form that would form more engagement with users, driving satisfaction.[3][10][11]

Table 1: *Comparison of Recommendation Algorithms*

Metric	TF-IDF + Cosine Similarity	Neural Collaborative Filtering
Accuracy	68%	81%
Precision@10	0.61	0.78
Recall@10	0.55	0.72
F1-Score	0.58	0.75
RMSE	1.12	0.94
NDCG@10	0.63	0.81
Cold Start	Poor	Moderate
Scalability	High	Moderate
Personalization	Basic	High

Table 1 provides a simple illustration showing with respect to the TF-IDF with Cosine Similarity methods, it can be concluded that the Neural Collaborative Filtering approach is vastly superior. The use of the Neural Collaborative Filtering approach resulted in improved accuracy (81% vs 68%), better Precision@10 (0.78 vs 0.61), and better Recall@10 (0.72 vs 0.55) in recommendations, and consequently, is more relevant and covers more recommendations. The F1-score (0.75) and RMSE (0.94) also improved, which reflects high predictive accuracy. NDCG@10 adds further support for the Neural Collaborative Filtering approach regarding ranking quality. The TF-IDF approach is suitable for scalable problems due to its fairly lightweight nature; however, in cold start scenarios, it is less flexible and cannot support solutions developed from end-to-end services. The Neural Collaborative Filtering system offers a more sophisticated and balanced recommendations system from a user point of view, and provides evidence that the hybrid design of the system is effective and viable.[4][10][11]

5. CONCLUSION

This study presents a robust, scalable, personalized recommendation system for academic libraries that integrates an enhanced item-based collaborative filtering algorithm and Neural Collaborative Filtering (NCF). The augmented item-based collaborative filtering algorithm employs the mean model representation to improve similarity calculations and minimize rating bias. Meanwhile, NCF leverages deep learning approaches to model complex non-linear user-item interactions. This hybrid recommendation system is able to provide better accuracy and more personalized recommendations compared to unequivocal traditional recommendation systems.

The system is built with the Django web framework and uses the Pandas and NumPy libraries for data preproceeding and numerical computations. Requests and requests-oauthlib libraries are used for API integrations that enable secure and authenticated access and messaging with external services. The Gunicorn deployment validates that we are ready for production.

Future improvements will include implementing NLP in semantic analysis, real time feedback for adaptive changes, and support for explainable AI for transparency. Multilingual capability and hybrid recommendation strategies can enhance personalization and user interaction.

REFERENCES

[1] C. Raja Kumar and V. Jayanthi, "A novel fuzzy rough sets theory based CF recommendation system," Comput. Syst. Sci. Eng., vol. 34, no. 3, pp. 123–129, 2019.
[2] J. Kang and H. Lim, "Proposal of content recommend system on insurance company web site using CF," J. Digit. Converg., vol. 17, no. 11,pp. 201–206, Nov. 2019.
[3] D. Siva Bala Selvamani, M. S. Vidhya Sree, M. P. Pavithra, M. G. Soundarya, and M.M. Preethika, "Books and movies recommendation and rating prediction based on CF networks," IJAST, vol. 29, no. 5, pp. 705–714, Apr. 2020.
[4] O. A. Montesinos-López, E. Franco-Pérez, F. J. Luna-Vázquez, J. Salinas-Ruiz, S. Sandoval-Carrillo, M. A. V. Jiménez, J. Cuervas, and P. C. Santana-Mancilla, "Benchmarking between item-based collaborative filtering algorithm and genomic best linear unbiased prediction (GBLUP)model in terms of prediction accuracy for wheat and maize," Biotecnia, vol. 22, no. 2, pp. 136–146, Aug. 2020.
[5] P.-Y. Hsu, J.-Y. Chung, and Y.-C. Liu, "Using the beta distribution technique to detect attacked items from collaborative filtering," Intell. Data Anal., vol. 25, no. 1, pp. 121–137, Jan. 2021.
[6] N. Kumar and P. A. R. Kumar, "STEM: Stacked ensemble model design for aggregation technique in group recommendation system," Int. J. Bus.Intell Data Mining, vol. 21, no. 1, p. 66, 2022.
[7] P. A. Ejegwa and J. M. Agbetayo, "Similarity-distance decision-making technique and its applications via intuitionistic fuzzy pairs," J. Comput.Cognit. Eng., vol. 2, no. 1, pp. 68–74, Jan. 2022.
[8] M. U. Danjuma, B. Yusuf, and I. Yusuf, "Reliability, availability, maintainability, and dependability analysis of cold standby series-parallel system," J. Comput. Cognit. Eng., vol. 1, no. 4, pp. 193–200, Apr. 2022.
[9] N. Shakeel, P. Teradata, and S. Shakeel, "Context-free word importance scores for attacking neural networks," J. Comput. Cognit. Eng., vol. 1,no. 4, pp. 187–192, Sep. 2022.
[10] S. Poudel and M. Bikdash, "Optimal dependence of performance and efficiency of collaborative filtering on random stratified subsampling," BigData Mining Anal., vol. 5, no. 3, pp. 192–205, Sep. 2022.
[11] W. Liang, S. Xie, J. Cai, J. Xu, Y. Hu, Y. Xu, and M. Qiu, "Deep neural network security collaborative filtering scheme for service recommendation in intelligent cyber-physical systems," IEEE Internet Things J., vol. 9, no. 22, pp. 22123–22132, Nov. 2022.

[12] A. Alshammari and R. C. Chabaan, "SPPN-RN101: Spatial pyramid pooling network with ResNet101-based foreign object debris detection in airports," Mathematics, vol. 11, no. 4, p. 841, 2023.
[13] Z. Li, Y. Miao, M. E. Torbaghan, H. Zhang, and J. Zhang, "Semi-automatic crack width measurement using an OrthoBoundary algorithm," Automation in Construction, vol. 158, p. 105251, 2024.
[14] Z. Li, M. E. Torbaghan, T. Zhang, X. Qin, W. Li, Y. Li, and J. Zhang, "An automated 3D crack severity assessment using surface data for improving flexible pavement maintenance strategies," IEEE Transactions on Intelligent Transportation Systems, vol. 25, no. 9, pp. 12490–12503, Sept. 2024
[15] L. Badis, M. Amad, D. Aissani, and S. Abbar, ‘‘P2PCF: A collaborative filtering-based recommender system for peer-to-peer social networks,’’ J. High-Speed Netw., vol. 27, no. 1, pp. 13–31, Mar. 2021.