[Array 19 (2023) 100299](https://doi.org/10.1016/j.array.2023.100299)

|  |  |  |
| --- | --- | --- |
|  | Contents lists available at [ScienceDirect](https://www.elsevier.com/locate/array) |  |
| Array |
| journal homepage: [www.elsevier.com/locate/array](http://www.elsevier.com/locate/array) |
|  | | |

A hybrid recommendation scheme for delay-tolerant networks: The case of digital marketplaces✩  
Victor M. Romero II∗, Bea D. Santiago, Jay Martin Z. Nuevo

*Division of Natural Sciences and Mathematics, University of the Philippines Visayas Tacloban College, Magsaysay Boulevard, Tacloban City, 6500, Leyte, Philippines*

|  |  |  |
| --- | --- | --- |
| A R T I C L E | I N F O | A B S T R A C T |
| *MSC:*  0000  1111 |  | Recommender systems are widely-adopted by numerous popular e-commerce sites, such as Amazon and E-bay, to help users find products that they might like. Although much has been achieved in the area, most recommender systems are designed to work on top of centralized platforms that are traditionally supported by fixed infrastructure like the Internet. Hence, additional work is warranted to examine the applicability and performance of recommender systems in challenging environments that are characterized by dynamic network topology and variable transmission delays. This study deals with the design of a recommender system that is compatible in a delay-tolerant network where communication is supported by opportunistic encounters between participating nodes. The proposed approach combines collaborative filtering and content-based filtering techniques to generate rating predictions for users. To make the system more tolerant against interruptions, each node maintains a local recommender that generates predictions using user profiles that are obtained through opportunistic exchanges over a clustered topology. Simulation results indicate that the proposed approach is able to improve coverage while alleviating the cold-start problem. |
| *Keywords:*  Delay-tolerant  Digital marketplace  Recommender systems |

**1. Introduction**

Delay-Tolerant Networks (DTN) are communication networks that are specifically created to operate in challenging environments lacking fixed infrastructure or where traditional networking is not feasible. These environments include isolated and remote areas as well as inter-planetary communication. In the absence of fixed infrastructures, DTNs leverage readily available wireless interfaces and opportunistic encoun-ters between mobile participating nodes. Consequently, these networks exhibit characteristics such as high and fluctuating transmission delays, absence or failure of end-to-end paths, and significant node mobil-ity [1,2]. In order to facilitate communication in such an unpredictable and highly dynamic environment, the network relies on a *store-carry-forward* mechanism to tolerate delay and achieve incremental progress until the message is received by the intended destination [3]. The process of how messages are propagated in a DTN is depicted in Fig. 1. When a source node needs to send a message, it first stores the message in its buffer until it encounters another node. If the second node meets the forwarding conditions and is capable of carrying the message, the source node forwards the message to the second node. The second node then carries the message until it encounters another node that

can further forward it. This series of forwarding continues through intermediate nodes until the message reaches the destination node. It has been proposed that electronic commerce can be supported effectively by delay-tolerant networking over an infrastructure charac-terized by high transmission delay [4]. This type of system could bring advantages to both consumers and entrepreneurs, even when compared to the traditional Internet. E-commerce, along with other Internet-based services like email, social network, and media streaming, naturally aligns with delay-tolerant networking because it does not require brief round trip time and is asynchronous. However, in order for e-commerce to be viable in a delay-tolerant environment, it is important to minimize the number of query/response interactions that occur while the user interacts with the server. To address this, more computational tasks must be transferred to the user devices, including relocation of code (such as rendering order forms) and data (such as shopping cata-logs). By adopting this approach, users can effortlessly browse products and compare prices without being dependent on server availability. Moreover, they become immune to delays and interruptions caused by congestion in the network infrastructure. Opportunistic encounters in the DTN-based digital marketplace can be focused instead on the prop-agation of product information towards the incremental completion

✩ This study was made possible through the support of the University of the Philippines Visayas Office of the Vice Chancellor for Research and Extension (Regular In-house Research Grant).

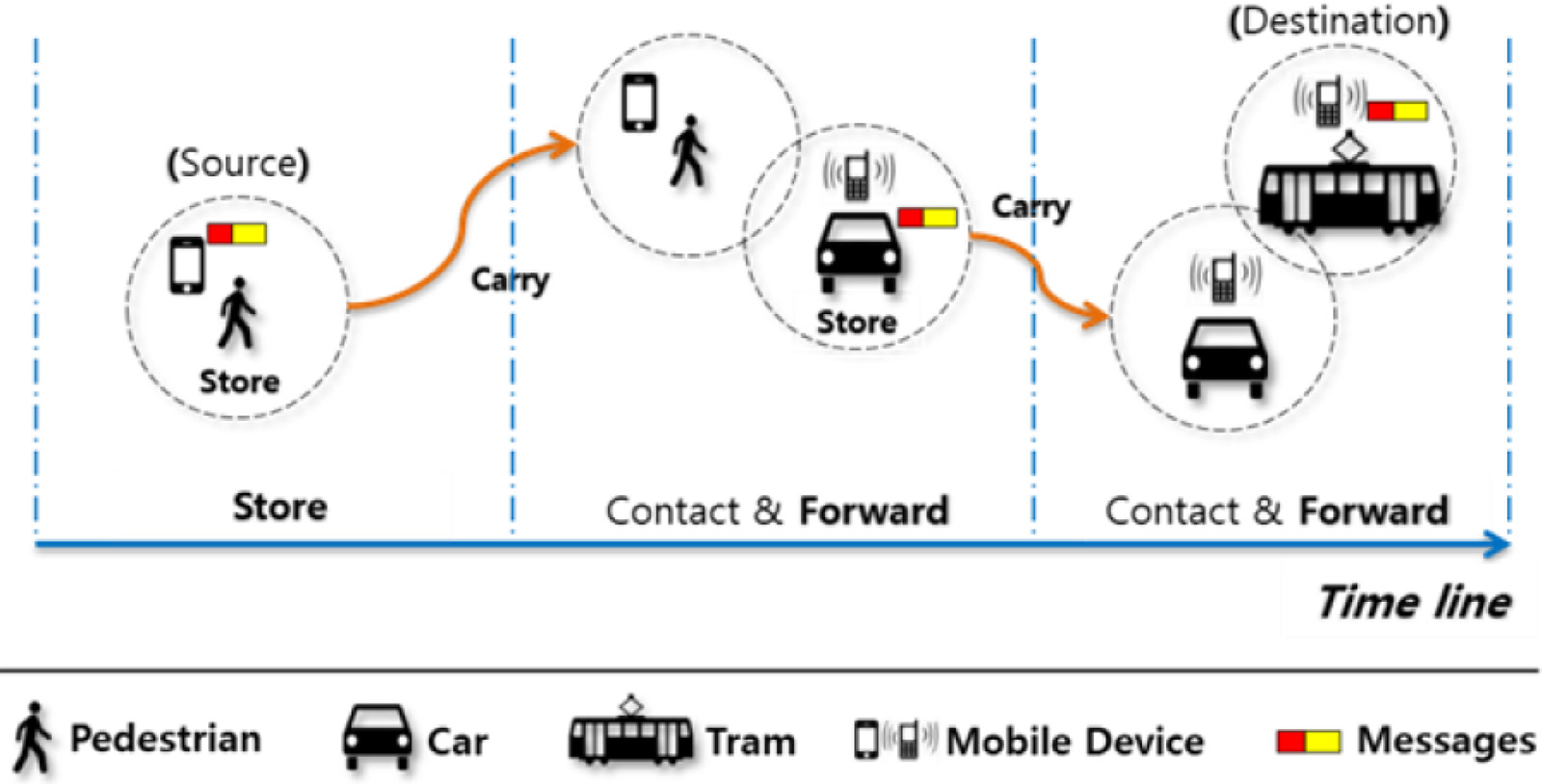
∗ Corresponding a[uthor.](mailto:vmromero@up.edu.ph)

*E-mail address:* [vmromero@up.edu.ph](mailto:vmromero@up.edu.ph) (V.M. Romero II).

<https://doi.org/10.1016/j.array.2023.100299>  
[Received 30 December 2022; Received in rev](https://doi.org/10.1016/j.array.2023.100299)ised form 4 June 2023; Accepted 11 June 2023   
Available online 15 June 2023   
[2590-0056](http://creativecommons.org/licenses/by-nc-nd/4.0/)/© 2023 The Authors. Published by Elsevier Inc. This is an open access article under the CC BY-NC-ND license ([http://creativecommons.org/licenses/by-](http://creativecommons.org/licenses/by-nc-nd/4.0/)

[nc-nd/4.0/](http://creativecommons.org/licenses/by-nc-nd/4.0/)).

*V.M. Romero II et al.*  *Array 19 (2023) 100299*



**Fig. 1.** Store-carry-forward mechanism in DTN taken from [3].

of locally stored product database and requests for order fulfillment. Although the envisioned system in [4] was not implemented, the author has highlighted key design considerations and advantages including exploitation of available computing power in end devices, reduction of server requirements, and potential to increase revenue by eliminating entry barriers. We independently developed a prototype of a DTN-based digital marketplace motivated by the potential of delay-tolerant networking in bridging the opportunity gap caused by the disparity in Internet access across different areas in the Philippines. The application was specifically designed for the Android operating system and utilized IBR-DTN’s Application Programming Interface (API) [5] to convert user actions, such as product registration and chat, into bundles that could be distributed within the network. The application provided a consistent set of functions to all users, enabling them to simultaneously assume the roles of both buyers and sellers. By capitalizing on existing network interfaces and leveraging opportunistic encounters, the result-ing delay-tolerant digital marketplace was able to operate without the need for additional hardware or network requirements.

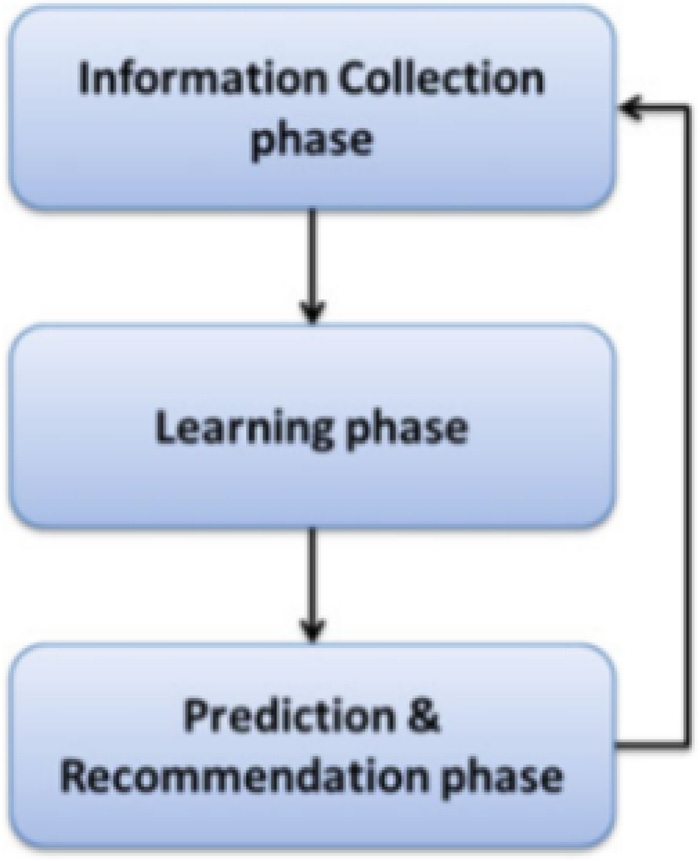
Similar to traditional online counterparts, the number of available products in a DTN-based digital marketplace is expected to grow as more users join the platform. This growth is amplified by the fact that different sellers can list similar products in their individual catalogs. Eventually, the product database outgrows the capacity of human agents to browse through and compare products, which could hinder meaningful interactions such as commitment of transactions [6]. Sev-eral approaches have been proposed to address information overload in digital marketplaces. Each approach can be characterized using a common framework that is based on two variables: personalization and automation. One example of an impersonal and automatic approach is the utilization of best seller lists generated from transaction reports. On the other hand, there are digital services that offer personalized filtering options, allowing users to navigate through large databases based on specific criteria, but requiring manual input from the users.

*1.1. Recommender systems*

Recommender Systems provide a personalized and automated so-lution to address the information overload challenge. These software tools assist users in navigating overwhelming product catalogs by ana-lyzing their preferences and recommending items that are more likely to be of interest to them. While recommender systems are commonly used in e-commerce, their application extends to various other fields

2

*V.M. Romero II et al.*  *Array 19 (2023) 100299*



**Fig. 2.** Phases of recommendation process taken from [14].

*1.1.2. Filtering techniques in recommender systems*   
 Various techniques can be used in the prediction and recommen-dation phase of a recommender system. Among these approaches, two well-known methods are content-based filtering and collaborative filtering.

Content-based filtering recommends items by analyzing the proper-ties of previously rated items to build individual user profiles [16]. The representation of a user’s level of interest in a new item is determined by measuring the similarity between the user’s profile and the features of the new item. Thus, the representation of user profiles plays a central role in content-based filtering.

The Vector-Space Model (VSM) is the simplest and most commonly used representation of user profiles in content-based filtering. The VSM is a statistical term-based technique widely employed in information retrieval literature. It represents the contents of documents as vectors of weighted terms. Similarly, user profiles are also represented as vectors of weighted terms that reflect the user’s preferences. The weight assigned to each keyword signifies its importance in representing the document or the user profile.

The adoption of content-based filtering in a recommender system presents several advantages [17]:

• User independence: Content-based filtering relies solely on the characteristics of previously rated products to construct user pro-files. Therefore, it does not require data from other users to make recommendations for a specific user.

• Transparency: Recommendations made by content-based filtering can be easily explained by examining the features that contribute to the inclusion of an item.

However, content-based filtering also has some drawbacks that should be taken into account:

• Limited content analysis: Content-based recommender techniques require sufficient information about the features or descriptions of items to accurately represent them. Therefore, the content of each item needs to possess a certain level of richness to enable precise user profile construction and subsequent recommendation.

• Overspecialization: Since content-based filtering suggests items similar to those previously rated, it may lack the ability to pro-vide unexpected or novel recommendations. This phenomenon, known as the *serendipity problem*, restricts the system’s capacity for generating highly diverse recommendations.

3

*V.M. Romero II et al.*  *Array 19 (2023) 100299*

Hybrid recommender systems aim to enhance performance and leverage the strengths of different recommendation techniques by com-bining them [19]. Kim et al. [20] proposed three hybridization mod-els: the linear combination model, the sequential combination model, and the mixed combination model. In the linear combination model, the output of multiple recommendation techniques is merged. Each technique generates a list of recommendations with assigned weights, which are then combined to produce a final recommendation list [11, 21]. The sequential combination model first applies content-based fil-tering to construct user profiles and then utilizes collaborative filtering to generate recommendations. Content-based filtering can be used to either enrich the user’s rating data [22,23] or construct user profiles considering the content of rated items [20]. The mixed combination model incorporates both semantic content and user ratings to gen-erate recommendations. This approach combines the advantages of both techniques in the recommendation process. Apart from these hy-bridization strategies, other studies have combined multiple techniques using different approaches. In [24], each user has two user profiles constructed using content-based and collaborative filtering. Similarity between users is computed by combining the similarity scores using both user profiles using a weighted hybridization technique. Turnip et al. [10] uses content-based filtering on top of collaborative filtering to refine the list of recommendations such that only items with a certain degree of content similarity as the user’s profile are retained. Logesh et al. [25] and Tran et al. [26] group users into clusters and employ user–user collaborative filtering within the clusters to predict missing ratings in the user–item rating matrix. Item–item collaborative filtering is then applied using the denser user-rating matrix.

*1.2. Recommender system in opportunistic networks*

Recommender systems designed for traditional environments face challenges when implemented in opportunistic networks, primarily due to the absence of a central entity. To overcome this limitation, the phases of recommendation need to be modified to accommodate the dynamic nature of the network. Existing literature on recommender systems in opportunistic networks distributes the responsibility of gen-erating recommendations across all the participating nodes [27–29]. Each node possesses a local recommender system that handles the prediction/recommendation phase for its active user. The information collection phase is also implemented in a distributed manner where each node collects the necessary data needed by the local recommender system and stores it in a local database. Each node also handles the processing of the learning phase of the active user using the available data in the local database.

This study focuses on designing a hybrid recommender system specifically for a delay-tolerant digital marketplace environment. The dynamic nature of the network poses challenges in implementing a recommender system, as acquiring the necessary data for user profiling and generating predictions must be done opportunistically. Addition-ally, in the absence of a central entity, recommendation processing needs to be distributed across mobile participating nodes. Despite the limitations in communication network, the recommender system is expected to generate meaningful recommendations. To address this, a hybrid approach is employed, combining both content-based and collaborative filtering techniques. This hybridization aims to strike a balance between recommendation accuracy and coverage.

**2. Materials and methods**

Mobile devices with the necessary computing resources (e.g., CPU, bandwidth, and storage) are ubiquitous in the real world. We imagine these devices to cooperate and collectively form the communication infrastructure necessary for supporting a delay-tolerant digital market-place. Using readily available network interfaces, each device defines a communication buffer. Whenever two devices come within the range

4

*V.M. Romero II et al.*  *Array 19 (2023) 100299*

for understanding their preferences. Additionally, the system aims to obtain user profiles of individuals with similar interests to the active user. These profiles are acquired through opportunistic exchanges of user information with other nodes within the cluster. By sharing user profiles, the system gathers valuable data that fills the local database, which subsequently supports the operations of the local recommender. The learning phase of the proposed hybrid recommender system operates in a distributed manner, with each node independently cal-culating the interests of its active user. The user’s interest profile is represented as a vector of keywords extracted from the descriptions of items that the user has previously rated. This process does not rely on external data sources and can be carried out autonomously by each individual node.

During the prediction phase of the proposed hybrid recommender system, each node operates its own local recommender to generate item predictions for its active user. The local recommender relies on the data stored in its local database to make accurate rating predictions. This decentralized approach allows the recommender system to continue functioning even when nodes are disconnected from other nodes in the network.

*2.1.1. Local database*   
 The local database of the proposed hybrid recommender system consists of three buffers, namely: (1) *RatedProductsBuffer*, (2) *Simi-larProfilesBuffer*, and (3) *ProductRecommendationsBuffer*.

The *RatedProductsBuffer* stores information about the items that have been previously rated by the target user. For each rated item, the buffer contains the product description and the corresponding rating provided by the user.

The *SimilarProfilesBuffer* is responsible for storing the user profiles obtained from opportunistic encounters with other nodes in the net-work. When a node receives a similar user profile, it compares the similarity between its own user profile and the received profile with the similarity with its current least similar neighbor (*lsn*). Initially, the *lsn* is set to −1. If the similarity between the node and the received user profile is higher than the current *lsn*, the least similar user profile in the *SimilarProfilesBuffer* is evicted to make space for the new profile. The *lsn* is updated to reflect this change and track the similarity of the least similar profile in the buffer.

The *ProductRecommendationsBuffer* contains the products that will be recommended to the target user. The specific algorithm for selecting and adding items to this buffer based on predicted ratings is beyond the scope of this study and is recommended for future development.

*2.1.2. Interest extraction*   
 Interest extraction in the proposed hybrid recommender system involves two phases: item profile creation and user profile creation. Each item *𝑖𝑗* has an item profile which is a vector of keywords extracted from its item description. However, item descriptions often contain unstructured text, including characters or texts that are irrelevant and introduce noise to the item profile creation process. To address this, a text pre-processing procedure is applied to sanitize the item descriptions. The pre-processing procedure is illustrated in Fig. 3. Using the sanitized item descriptions, item profiles are created using the algorithm used in [32]. Item profile *𝑣*(*𝑖𝑗*) is denoted as,

|  |  |
| --- | --- |
| *𝑣*(*𝑖𝑗*) = ((*𝑘*1 *𝑖𝑗, 𝑤*(*𝑘*1 *𝑖𝑗*))*,* (*𝑘*2 *𝑖𝑗, 𝑤*(*𝑘*2 *𝑖𝑗*))*,* … *,* (*𝑘𝑚 𝑖𝑗, 𝑤*(*𝑘𝑚 𝑖𝑗*))) | (1) |
| where *𝑘𝑙 𝑖𝑗*corresponds to a keyword, *𝑤*(*𝑘𝑙 𝑖𝑗*) denotes the weight of *𝑘𝑙 𝑖𝑗*,  and *𝑚* denotes the number of keywords in the description. The weight  *𝑤*(*𝑘𝑙 𝑖𝑗*) is computed using the following formula,  *𝑤𝑘𝑙*  where *𝑛*(*𝑘𝑙*   *𝑖𝑗*   =  ∑*𝑚 𝑞*=1(1 + *𝑙𝑜𝑔*(*𝑛*(*𝑘𝑙*  *𝑖𝑗*) corresponds to the number of occurrences *𝑘𝑙* 1 + *𝑙𝑜𝑔*(*𝑛*(*𝑘𝑙 𝑖𝑗*))  *𝑖𝑗*)))  *𝑖𝑗*has in the  (2)  description. | |

5

*V.M. Romero II et al.*  *Array 19 (2023) 100299*

• *pTime* — the time when the user profile was last updated locally by the cluster member.

Each cluster member actively communicates with its respective clus-ter head by sending a *BeaconMessage* to convey its current state. This message serves as a means of informing the cluster head of its presence and providing updates on its status. Upon receiving a *BeaconMessag*e from a cluster member, the cluster head compares the *pTime* value in the message with the corresponding record in the *MemberInfor-mationTable*. If the two values are not equal, indicating a change in the member’s profile, the cluster head sends a *ProfileRequestMessage* to request the updated profile information from the cluster member. Subsequently, the cluster member responds by sending a *ProfileUp-dateMessage* to the cluster head, ensuring that the profile information remains up to date within the cluster.

Upon receiving a *ProfileUpdateMessage* from a cluster member, the cluster head initiates an update procedure following the steps outlined in Algorithm 1. To optimize the update process and reduce the number of messages to be sent, a delayed broadcast mechanism is employed. When the cluster head receives a *ProfileUpdateMessage*, it starts an up-date timer. If subsequent *ProfileUpdateMessages* are received before the timer expires, the timeout period is reset. However, to prevent excessive waiting times, a maximum waiting time, denoted as *maxWaitTime*, is introduced. If the update timer reaches *maxWaitTime*, the update timer is no longer restarted, even if *ProfileUpdateMessages* are further received. This ensures that the system does not experience prolonged delays.

**Algorithm 1** Update Procedure of Cluster Head

**for** each received user profile from a cluster member **do**  addr = cluster member address   
 newProfiles.add(addr, user profile)   
 **if** updateTimer is running **then**   
 updateTimer.reset();   
 **end if**   
 updateTimer.start()   
**end for**   
**for** each elapsed updateTimer **do**   
 updateUserSimilarityMatrix(newProfiles);   
 membersSimilarProfiles = gatherSimilarProfiles(); sendSimilarProfiles(membersSimilarProfiles)   
**end for**

|  |
| --- |
| After the update timer expires, the cluster head proceeds to update the user–user similarity matrix, which contains the similarity values between each cluster member. User profiles received during the update procedure are incorporated into the matrix. For each new profile, the cluster head computes the similarity with existing user profiles and updates the corresponding entries in the matrix. If a cluster member that provided a new profile already has an entry in the matrix, its entry is updated accordingly.  User–user similarity matrix is illustrated in Table 1. Each cell *𝑠𝑖𝑚𝑣𝑖,𝑣𝑗* corresponds to the similarity between cluster members *𝑣𝑖* and *𝑣𝑗*. Simi-larity is calculated using cosine similarity, which determines similarity based on the angle between two vectors. The cosine-based similarity measure is computed as follows:  *𝑠𝑖𝑚*(*𝑣*1*, 𝑣*2) =  where *𝑚*′is the union of keywords from *𝑣*1 and *𝑣*2, and *𝑤*(*𝑘𝑎*√∑*𝑚*′*𝑎*=1(*𝑤*(*𝑘𝑎*∑*𝑚*′*𝑎*=1*𝑤*(*𝑘𝑎*  *𝑣*1))2 ∗*𝑣*1) ∗ *𝑤*(*𝑘𝑎*√∑*𝑚*′*𝑎*=1(*𝑤*(*𝑘𝑎*  *𝑣*2)  *𝑣*2))2  *𝑣*1) and  (6)  *𝑤*(*𝑘𝑎 𝑣*2) corresponds to the weights of the *𝑎*th keyword of *𝑚*′ in *𝑣*1and *𝑣*2, respectively.  Once the matrix is updated, the cluster head proceeds to aggregate similar profiles for each member within its cluster. For every cluster |

6

*V.M. Romero II et al.*  *Array 19 (2023) 100299*

**Table 3**

User–item Rating Matrix.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | *𝑖𝑅* 1 | *𝑖𝑅* 2 | *𝑖𝑅* 3 | ⋯ | *𝑖𝑅 𝑄* |
| *𝑢𝑡𝑎𝑟𝑔𝑒𝑡* | *𝑟𝑅 𝑢𝑡𝑎𝑟𝑔𝑒𝑡,𝑖*1  *𝑟𝑅 𝑢*2*,𝑖*1 | *𝑟𝑅 𝑢𝑡𝑎𝑟𝑔𝑒𝑡,𝑖*2  *𝑟𝑅 𝑢*2*,𝑖*2 | *𝑟𝑅 𝑢𝑡𝑎𝑟𝑔𝑒𝑡,𝑖*3  *𝑟𝑅 𝑢*2*,𝑖*3 | ⋯ | *𝑟𝑅 𝑢𝑡𝑎𝑟𝑔𝑒𝑡,𝑖𝑄*  *𝑟𝑅 𝑢*2*,𝑖𝑄*  *𝑟𝑅 𝑢*3*,𝑖𝑄* |
| ⋯ |
| *𝑢*2 |
| *𝑢*3 | *𝑟𝑅 𝑢*3*,𝑖*1 | *𝑟𝑅 𝑢*3*,𝑖*2 | *𝑟𝑅 𝑢*3*,𝑖*3 | ⋯ |
| ⋯ | ⋯ | ⋯ | ⋯ | ⋯ | ⋯ |
| *𝑢𝑃* | *𝑟𝑅 𝑢*4*,𝑖*1 | *𝑟𝑅 𝑢*4*,𝑖*2 | *𝑟𝑅 𝑢*4*,𝑖*3 | ⋯ | *𝑟𝑅 𝑢*4*,𝑖𝑄* |

**Table 4**

An example of a user–item rating matrix.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | *𝑖𝑅* 1 | *𝑖𝑅* 2 | *𝑖𝑅* 3 | *𝑖𝑅* 4 | *𝑖𝑅* 5 |
| *𝑢𝑡𝑎𝑟𝑔𝑒𝑡* | 5  5 | ?  2 | 3  4  3 | 4 | 4  4 |
| *𝑢*2 | 4  4  4 |
| *𝑢*3 | 4 | 3  2 |
| *𝑢*4 | 3 |
| *𝑢*5 |

**Table 5**

Item–item similarity matrix.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *𝑖𝑅* 1 | | *𝑖𝑅* 2 | *𝑖𝑅* 3 | ⋯ | *𝑖𝑅 𝑄* |
| *𝑖𝑁* 1 | *𝑠𝑖𝑚𝑖𝑁* 1*,𝑖𝑅* | *𝑠𝑖𝑚𝑖𝑁* 1*,𝑖𝑅* | *𝑠𝑖𝑚𝑖𝑁* 1*,𝑖𝑅* | ⋯ | *𝑠𝑖𝑚𝑖𝑁* 1*,𝑖𝑅 𝑄* |
| *𝑖𝑁* 2 | *𝑠𝑖𝑚𝑖𝑁* 2*,𝑖𝑅* | *𝑠𝑖𝑚𝑖𝑁* 2*,𝑖𝑅* | *𝑠𝑖𝑚𝑖𝑁* 2*,𝑖𝑅* | ⋯ | *𝑠𝑖𝑚𝑖𝑁* 2*,𝑖𝑅 𝑄* |
| *𝑖𝑁* 3 | *𝑠𝑖𝑚𝑖𝑁* 3*,𝑖𝑅* | *𝑠𝑖𝑚𝑖𝑁* 3*,𝑖𝑅* | *𝑠𝑖𝑚𝑖𝑁* 3*,𝑖𝑅* | ⋯ | *𝑠𝑖𝑚𝑖𝑁* 3*,𝑖𝑅 𝑄* |
| ⋯ | ⋯ | ⋯ | ⋯ | ⋱ | ⋯ |
| *𝑖𝑁 𝐾* | *𝑠𝑖𝑚𝑖𝑁 𝐾,𝑖𝑅* | *𝑠𝑖𝑚𝑖𝑁 𝐾,𝑖𝑅* | *𝑠𝑖𝑚𝑖𝑁 𝐾,𝑖𝑅* | ⋯ | *𝑠𝑖𝑚𝑖𝑁 𝐾,𝑖𝑅* |
| of the Item–Item similarity matrix where *𝑠𝑖𝑚𝑖𝑁 𝑘,𝑖𝑅𝑞* denotes the similarity between items *𝑖𝑁 𝑘*and *𝑖𝑅 𝑞*.  The local recommender checks the item–item similarity matrix to obtain the top *𝑙* most similar items in *𝐼𝑅* to *𝑖𝑁 𝑘*. To make rating prediction for item *𝑖𝑁 𝑘*, the ratings of *𝑢𝑡𝑎𝑟𝑔𝑒𝑡* on the top *𝑙* most similar items are considered using Eq. (8):  *𝑟𝑢𝑡𝑎𝑟𝑔𝑒𝑡,𝑖𝑁*  where *𝑠𝑖𝑚𝑖𝑁*   *𝑘*=∑*𝑄*  *𝑘,𝑖𝑅𝑞*   *𝑞*=1(*𝑠𝑖𝑚𝑖𝑁*  denotes the similarity between items *𝑖𝑁*∑*𝑄 𝑞*=1*𝑠𝑖𝑚𝑖𝑁*   *𝑘,𝑖𝑅𝑞* ∗ *𝑟𝑢𝑡𝑎𝑟𝑔𝑒𝑡,𝑖𝑅𝑞* )  *𝑘,𝑖𝑅𝑞*  *𝑘*  and *𝑖𝑅 𝑞*and   (8)  *𝑟𝑢𝑡𝑎𝑟𝑔𝑒𝑡,𝑖𝑅𝑞* denotes the rating of *𝑢𝑡𝑎𝑟𝑔𝑒𝑡* to *𝑖𝑅* In general, when making predictions, the approach depends on *𝑞*in the user–item rating matrix.  whether the item in question has already been rated by the target user and its similar users, which is implied by an entry in the user–item rating matrix. If an entry exists, the predicted rating generated through collaborative filtering is utilized as the predicted rating for that item. On the other hand, if there is no existing entry for the item, the content-based filtering approach is employed instead. Therefore, the content-based filtering approach is only utilized when collaborative filtering is unable to make a rating prediction for the specific product. | | | | | |

*2.2. Simulation environment*

The ONE (Opportunistic Network Environment) simulator [33] is a simulation engine developed in Java that enables the execution of large-scale experiments in opportunistic networks. It provides the capability to assign diverse mobility models to different groups of nodes, allowing for the generation of movement patterns that mimic real-world agent navigation. Additionally, ONE allows users to specify the routing protocol to be employed by each node for message routing within the network. The interactions between nodes as they traverse the simulation environment can be visualized through ONE’s graphical user interface or logged for future analysis. Leveraging ONE’s appli-cation programming interface (API), we have developed a prototype delay-tolerant digital marketplace to evaluate the effectiveness of our proposed recommender system.

7

*V.M. Romero II et al.*  *Array 19 (2023) 100299*

|  |  |
| --- | --- |
| **Table 7**  General parameters used for recommender system simulation. |  |
| Parameter | Value |
| Simulation update interval  Scenario end time  Request message size  Default message size  Message TTL  Routing protocol  Transmission range  Message buffer size  Item vector threshold  Item similarity threshold | 0.1 s  9000 s  1 KB  300 KB  2 min  Direct delivery 200 m  50 MB  0.01  0.3 |

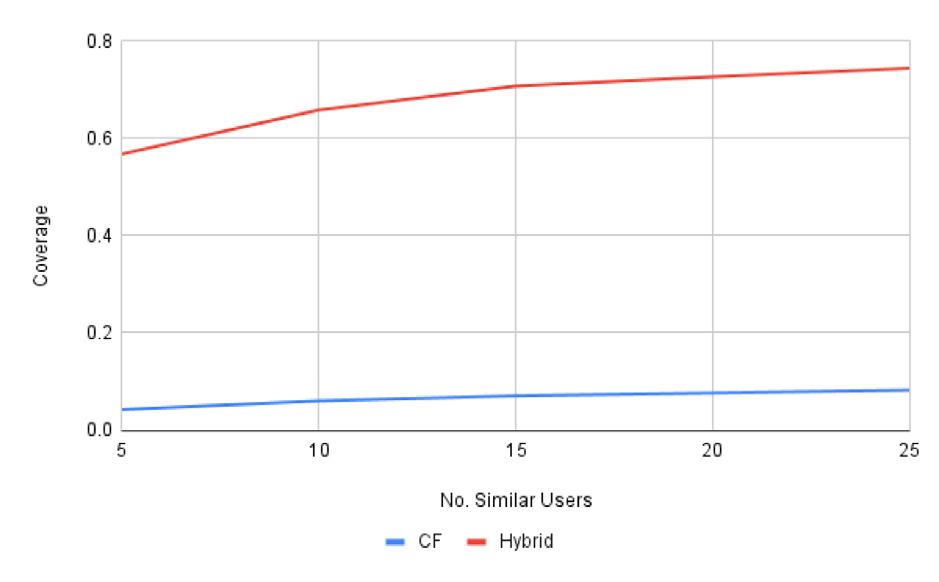
consistency of events across different simulation scenarios and facil-itates reproducibility. Once the simulation begins, nodes start moving according to their assigned mobility models. They establish connections with other nodes in close proximity, reflecting the opportunistic nature of the network. Throughout the simulation, reports are continuously updated to gather statistics that capture various aspects of the network’s state, including encounters, topology, and buffer occupancy.

*2.2.4. General simulation parameters*   
 The general parameters and their default values are summarized in Table 7. The update interval of a simulation denotes the discrete time interval between event processing. It is believed that no events are processed between two timestamps. This parameter is set to 0.1 s to allow for the recording of events that occur during possibly brief contact between nodes. Simulation time for all scenarios is set to 9000 s. The transmission speed is set to 11 Mbps, which is equal to the IEEE 802.11b standard’s theoretical maximum throughput. Messages received by each node are sent to adjacent neighbors using Direct Delivery as a routing protocol until they reach their destination or are dropped due to expired time-to-live values, which is set to 2 min. The parameters are configured in this manner because of the nature of our proposed approach which only sends messages to nodes that are in their respective transmission range. Therefore, the TTL of the messages only need to be long enough to be sent to the final recipient. When the messages are already received by their final recipient, they are deleted from the message buffer of the sender. At the same time, the messages are no longer added to the message buffer of the receiver. The message buffer size of each node is 50 MB and their corresponding transmission range is set to 200 m. The size of *RequestMessage* is 1 KB while the size of the other messages – *ProductMessage*, *SimilarProfilesMessage*, *ProfileUpdateMessage*, *BeaconMessage* – is set to 300 KB.

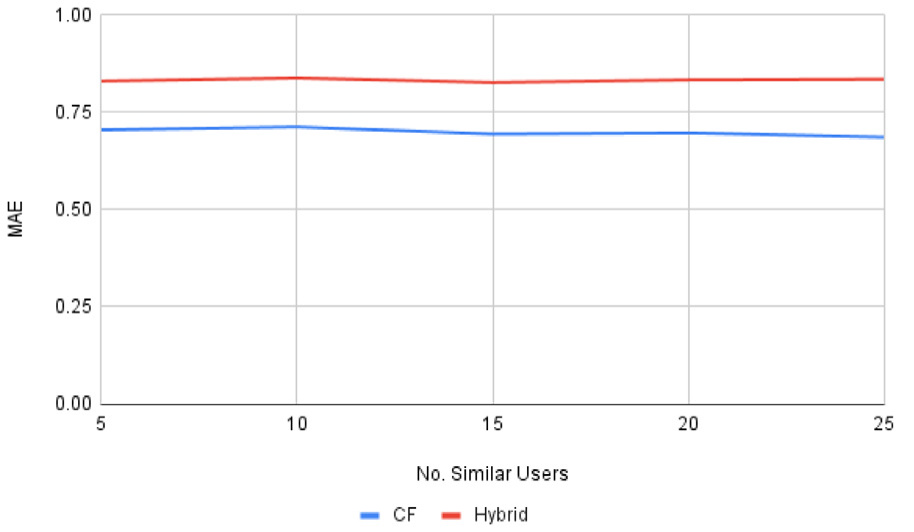
*2.2.5. Dataset filtering*   
 Users have different interests, which can be reflected on their purchase history. To emulate this, we use the user rating dataset extracted from Amazon website such that the nodes in the simulation will exhibit actual rating behaviors of real users in an e-commerce site. The dataset contains product metadata and reviews by actual users of the ecommerce platform from May 1996 to October 2018. Each product’s metadata include its description, category, information, and prices, among others. There are 233.1 million total number of reviews and 29 product categories in the dataset. However, due to constraints in computing resources, only products under the *Grocery and Gourmet* category are considered which contains 283,354 products, 4,887,517 ratings by 2,695,230 users. The dataset is further filtered to include only users with at least 20 ratings to make user profiles more representative.

Since the number of users in the reduced version of the dataset is 7689, exceeding the allotted number of nodes for the simulation, 170 users are randomly selected. The dataset contains the timestamp of the actual time the items were rated by the users in the website. Therefore, we split the ratings of each user into train and test set based on their

8



*V.M. Romero II et al.*  *Array 19 (2023) 100299*



**Fig. 4.** Performance of our proposed hybrid recommender system and pure collaborative filtering using Amazon dataset.

*2.3.3. DTN set-up*   
 The next set of experiments focus on how our proposed recom-mender system makes rating predictions on products in a delay-tolerant environment. Over time, the MAE and coverage may change as the nodes obtain the profiles of their most similar users. Therefore, we compute the MAE and coverage every 5.0 s from the start of the simulation to track the performance of the recommender system as it gathers the profiles of more similar users in the network. We compare the performance of the reference algorithm and our proposed hybrid recommender system in a DTN environment. This experiment deter-mines the viability of our proposed hybrid approach in improving the performance of our reference algorithm in building a recommender system in a delay-tolerant environment. Finally, we conduct a com-parison between the performance of the proposed hybrid approach in both a centralized setup and a DTN setup. This experiment aims to determine the degree of success achieved by the DTN version despite the challenging network conditions.

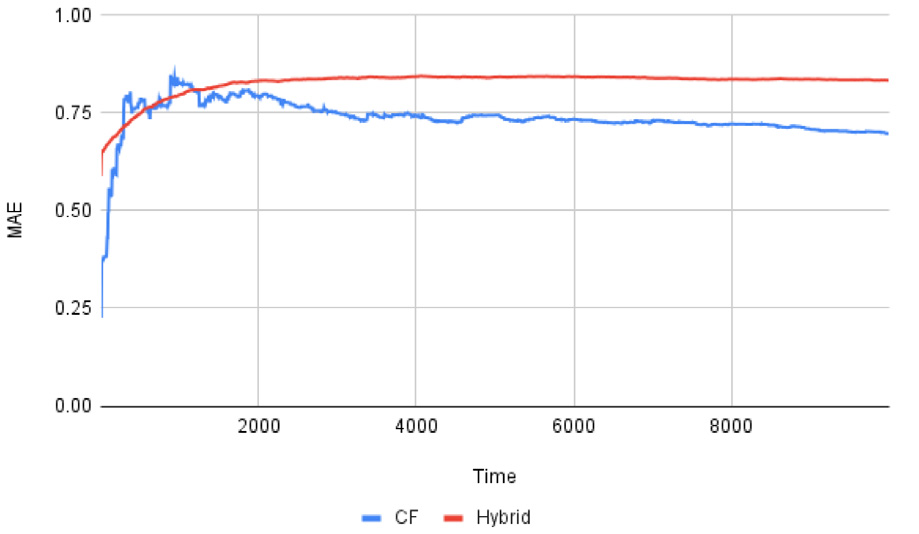
**3. Results and discussions**

Fig. 4 compares the performance of the reference algorithm and the proposed hybrid approach. The coverage of the proposed hybrid approach is higher than the coverage of the reference algorithm. Our proposed approach was able to perform well in terms of coverage despite the sparseness of the user–item rating matrix which signals that the proposed hybrid recommender system was able to alleviate the cold start problem for new items which is evident in this dataset.

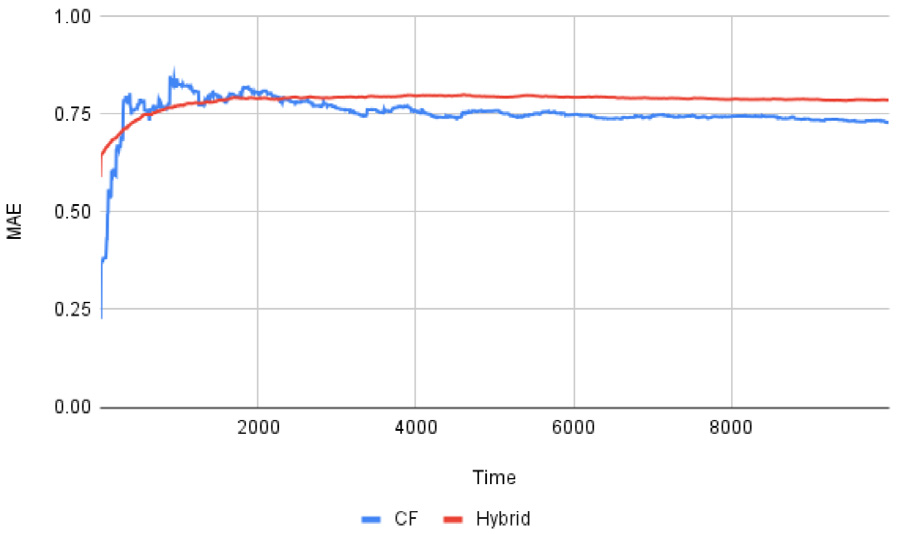
The MAE of both approaches did not increase nor decrease as it considers the opinions of more similar users. Because the nodes in this part of the experiment are chosen at random, the nodes have low similarity with each other. There are instances where the nodes cannot fill up their corresponding *SimilarProfilesBuffer*. Therefore, their list of similar users considered in making rating predictions is no longer increased despite increasing the size of the *SimilarProfilesBuffer*. The performance of the proposed hybrid approach and the reference algorithm are tested in a delay-tolerant environment where nodes no longer have global access to the profiles of all the users in the network. Instead, nodes obtain the profiles of similar users through opportunistic exchange over clustered topology. Both approaches are tested considering the ratings of different number of similar users in predicting the ratings of users on items. The results are shown in Figs. 5, 6, 7, 8, and 9.

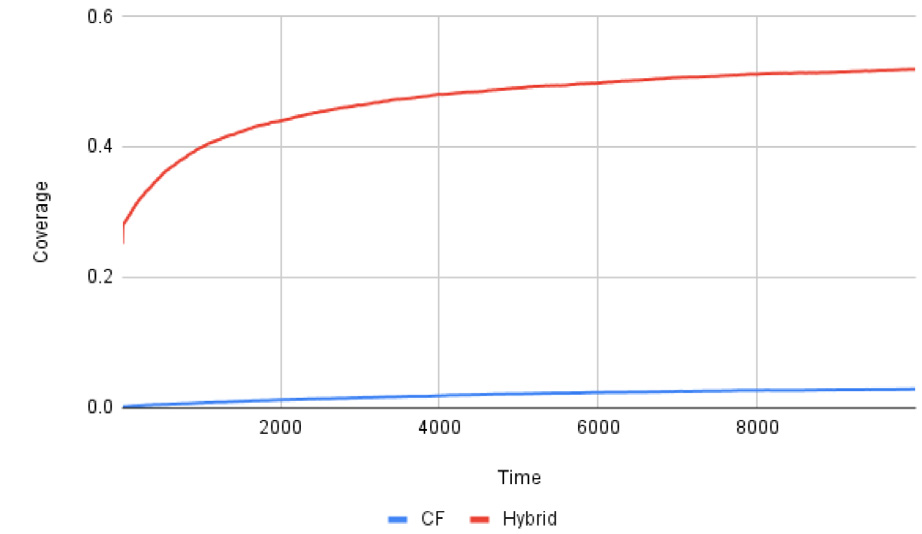
At the start of the simulation, the *SimilarProfilesBuffer* of all nodes are still empty because they have not had any contact with any other nodes in the network yet. The *SimilarProfilesBuffer* of each node is gradually filled over time. However, the similarity of the user profiles

9



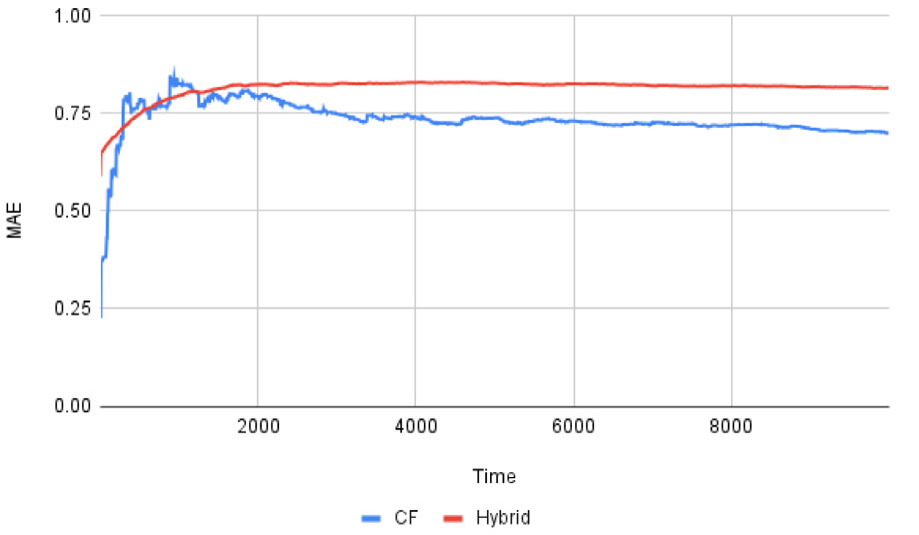
*V.M. Romero II et al.*  *Array 19 (2023) 100299*

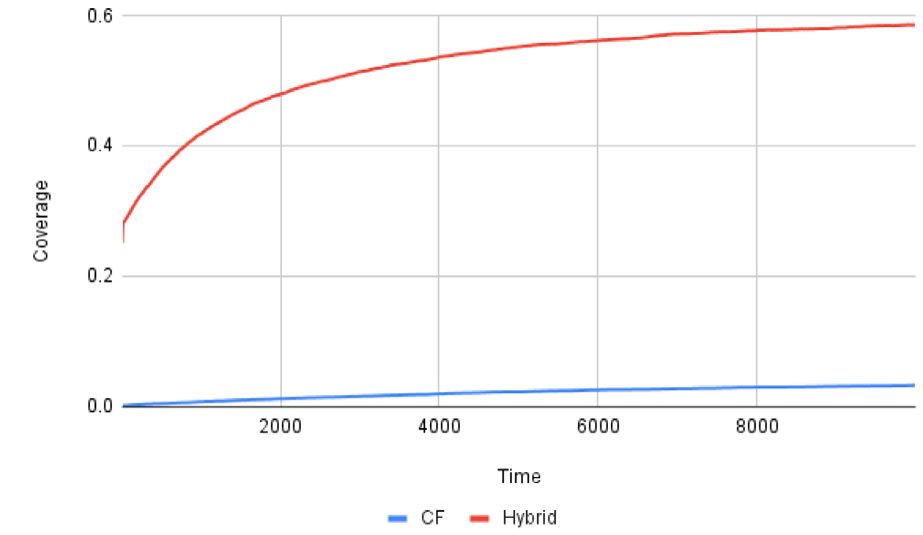




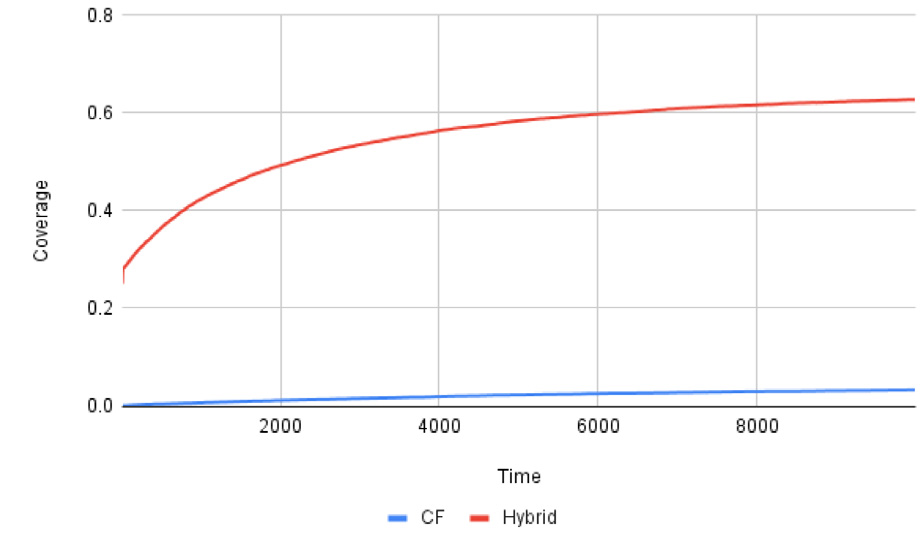
**Fig. 5.** Performance of proposed hybrid recommender system vs. pure collaborative filtering over delay-tolerant network considering the ratings of 5 similar users.





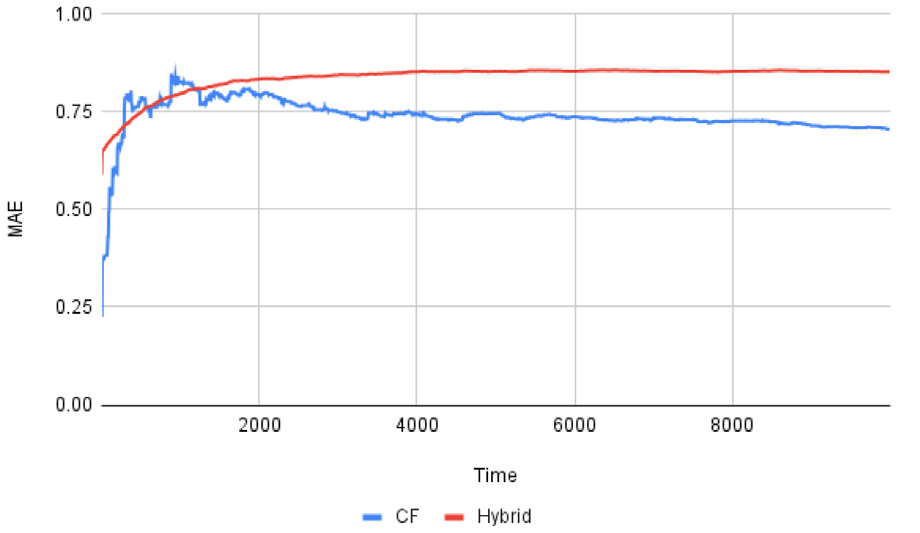
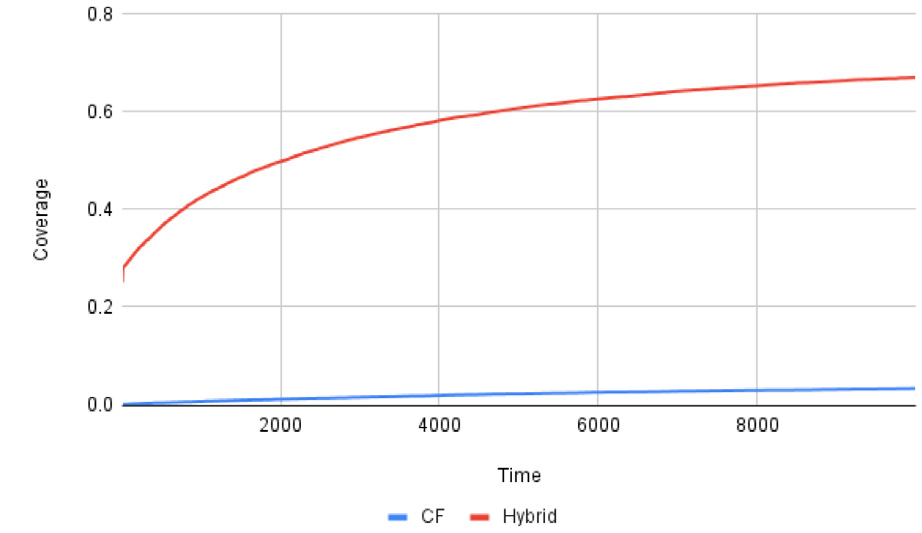
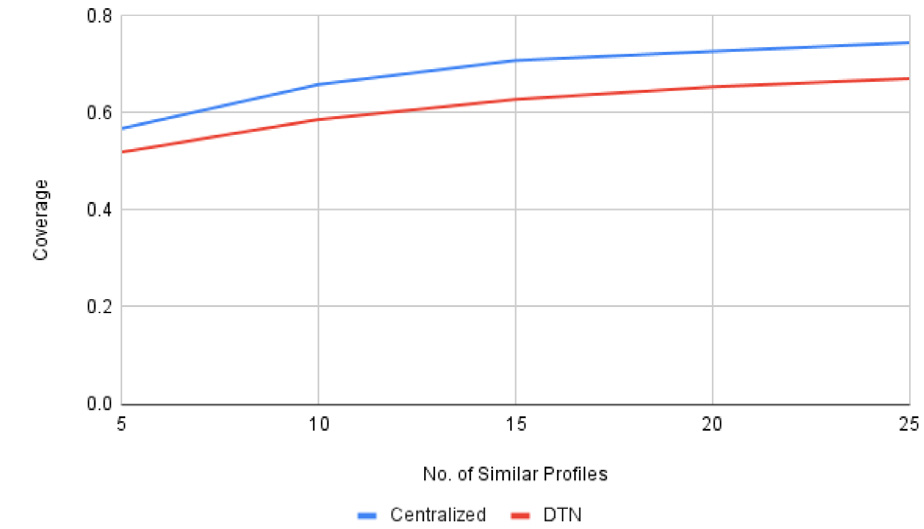
**Fig. 6.** Performance of proposed hybrid recommender system vs. pure collaborative filtering over delay-tolerant network considering the ratings of 10 similar users.



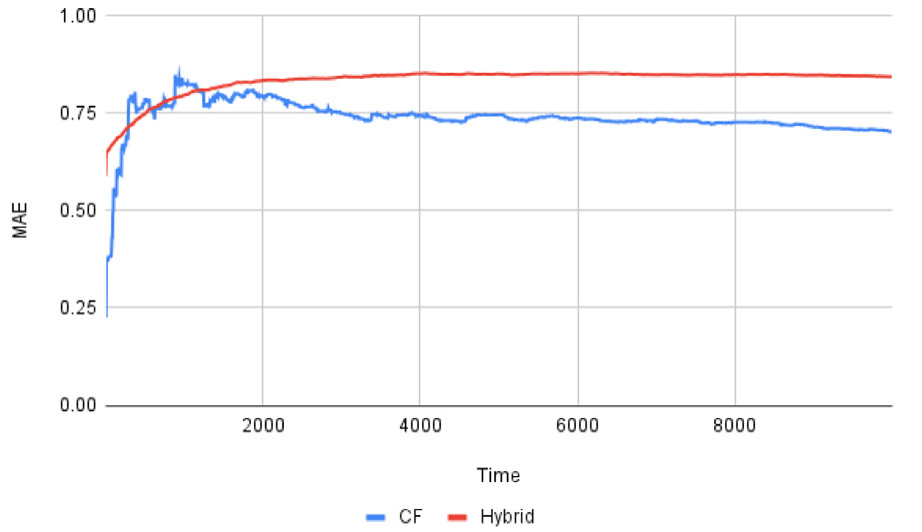
 

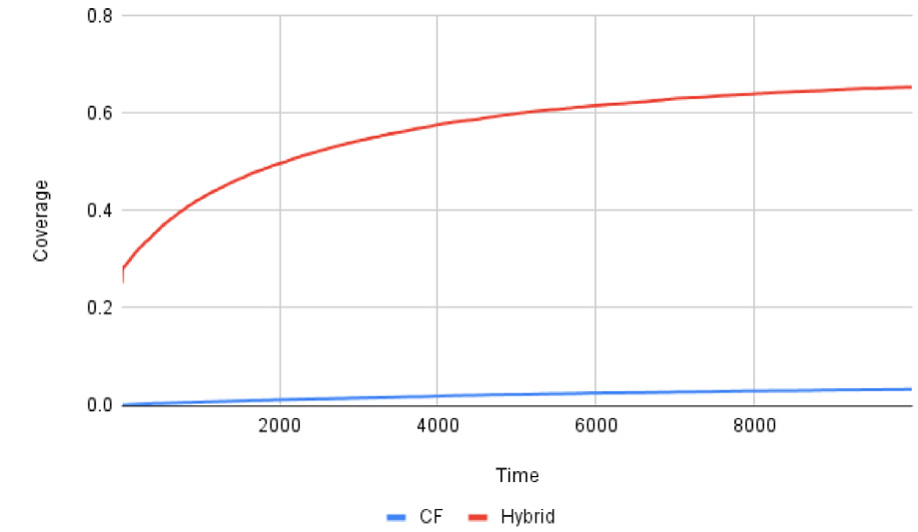
**Fig. 7.** Performance of proposed hybrid recommender system vs. pure collaborative filtering over delay-tolerant network considering the ratings of 15 similar users.

10



*V.M. Romero II et al.*  *Array 19 (2023) 100299*



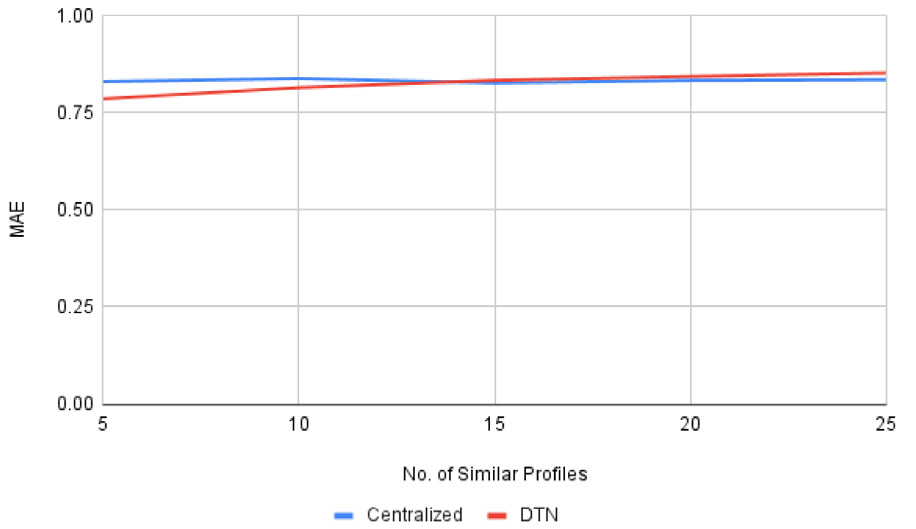


**Fig. 8.** Performance of proposed hybrid recommender system vs. pure collaborative filtering over delay-tolerant network considering the ratings of 20 similar users.

**Fig. 9.** Performance of proposed hybrid recommender system vs. pure collaborative filtering over delay-tolerant network considering the ratings of 25 similar users.



**Fig. 10.** Performance of proposed hybrid recommender system in centralized vs. DTN set-up.

revealed that the reference algorithm achieves better MAE, but at the expense of significantly lower coverage. In contrast, the proposed hybrid approach exhibits higher coverage while having a slightly higher MAE. The choice between these two approaches depends on the trade-off between higher coverage and a slightly higher MAE, or lower MAE with a smaller set of user–item pairs for rating predictions. Notably, the MAE of the proposed hybrid approach is expected to improve as more users provide ratings for items. In the delay-tolerant network environment, the proposed hybrid approach demonstrates its ability to make rating predictions even in the absence of user profiles from other network nodes. In such cases, the recommender system relies on pure content-based filtering. As nodes progressively obtain profiles of more similar users, the coverage of the system increases. This indicates that considering the ratings of users with higher similarity to the target

11

*V.M. Romero II et al.*  *Array 19 (2023) 100299*

Investigation, Data curation, Writing – original draft, Writing – re-view & editing, Visualization. **Jay Martin Z. Nuevo:** Conceptualiza-tion, Methodology, Software, Validation, Formal analysis, Investiga-tion, Data curation, Writing – original draft, Visualization.

**Declaration of competing interest**

The authors declare the following financial interests/personal rela-tionships which may be considered as potential competing interests: Victor M. Romero II reports financial support was provided by Uni-versity of the Philippines Visayas - Office of the Vice Chancellor for Research and Extension.

**Data availability**

Data will be made available on request.

**Acknowledgment**

This study was supported by the University of the Philippines Visayas - Office of the Vice Chancellor for Research and Extension through its Regular In-house Research grant.

**References**

[1] [Chenji H, Stoleru R. Delay-tolerant networks (DTNs) for emergency commu-nications. In: Advances in delay-tolerant networks (DTNs). Elsevier; 2021, p. 103–34.](http://refhub.elsevier.com/S2590-0056(23)00024-3/sb1)

[2] [Penning A, Baumgärtner L, Höchst J, Sterz A, Mezini M, Freisleben B. Dtn7: An open-source disruption-tolerant networking implementation of bundle protocol 7. In: Ad-hoc, mobile, and wireless networks: 18th International conference on ad-hoc networks and wireless, ADHOC-now 2019, Luxembourg, Luxembourg, October 1–3, 2019, Proceedings 18. Springer; 2019, p. 196–209.](http://refhub.elsevier.com/S2590-0056(23)00024-3/sb2)

[3] [Baek KM, Seo DY, Chung YW. An improved opportunistic routing](http://refhub.elsevier.com/S2590-0056(23)00024-3/sb2) [protocol based](http://refhub.elsevier.com/S2590-0056(23)00024-3/sb3)  [on context information of mobile nodes. Appl Sci 2018;8(8):1344.](http://refhub.elsevier.com/S2590-0056(23)00024-3/sb3)

[4] [Burleigh S. Delay-tolerant electronic commerce. In: 2015 Internation](http://refhub.elsevier.com/S2590-0056(23)00024-3/sb3)[al conference](http://refhub.elsevier.com/S2590-0056(23)00024-3/sb4)  [on wireless communications & signal processing (WCSP). IEEE; 2015, p. 1–4.](http://refhub.elsevier.com/S2590-0056(23)00024-3/sb4) [5] [Schildt S, Morgenroth J, Pöttner W-B, Wolf L. Ibr-dtn: a lightweight, modular](http://refhub.elsevier.com/S2590-0056(23)00024-3/sb5)  [and highly portable bundle protocol implementation. Electronic Communications](http://refhub.elsevier.com/S2590-0056(23)00024-3/sb5)  [of the EASST 2011;37.](http://refhub.elsevier.com/S2590-0056(23)00024-3/sb5)

[6] [Alamdari PM, Navimi](http://refhub.elsevier.com/S2590-0056(23)00024-3/sb5)[pour NJ, Hosseinzadeh M, Safaei AA, Darwesh A. A systematic study on the recommender systems in the E-commerce. IEEE Access 2020;8:115694–716.](http://refhub.elsevier.com/S2590-0056(23)00024-3/sb6)

[7] [Galeano D, Paccana](http://refhub.elsevier.com/S2590-0056(23)00024-3/sb6)[ro A. A recommender system approach for predicting drug side effects. In: 2018 International joint conference on neural networks (IJCNN). IEEE; 2018, p. 1–8.](http://refhub.elsevier.com/S2590-0056(23)00024-3/sb7)

[8] [Berkani L. A semantic and social-based collaborative recommendation of friends](http://refhub.elsevier.com/S2590-0056(23)00024-3/sb8)  [in social networks. Softw - Pract Exp 2020;50(8):1498–519.](http://refhub.elsevier.com/S2590-0056(23)00024-3/sb8)

[9] [Kbaier MEBH, Masri H, Krichen S. A personalized hybrid to](http://refhub.elsevier.com/S2590-0056(23)00024-3/sb8)[urism recommender system. In: 2017 IEEE/ACS 14th International conference on computer systems and applications (AICCSA). IEEE; 2017, p. 244–50.](http://refhub.elsevier.com/S2590-0056(23)00024-3/sb9)

[10] [Turnip R, Nurjanah D, Kusumo DS. Hybrid recom](http://refhub.elsevier.com/S2590-0056(23)00024-3/sb9)[mender system for learning material using content-based filtering and collaborative filtering with good learners’ rating. In: 2017 IEEE conference on e-learning, e-management and e-services (IC3e). IEEE; 2017, p. 61–6.](http://refhub.elsevier.com/S2590-0056(23)00024-3/sb10)

[11] [Wenzhen W. Personalized music reco](http://refhub.elsevier.com/S2590-0056(23)00024-3/sb10)[mmendation algorithm based on hybrid collaborative filtering technology. In: 2019 International conference on smart grid and electrical automation (ICSGEA). IEEE; 2019, p. 280–3.](http://refhub.elsevier.com/S2590-0056(23)00024-3/sb11)

[12] [Forouzandeh S, Aghdam AR, Barkhordari M, Fahimi SA, Vayqan](http://refhub.elsevier.com/S2590-0056(23)00024-3/sb11) [MK, Forouzan-deh S, Khani EG. Recommender system for users of internet of things (IOT). Int J Comput Sci Netw Secur 2017;17(8):46.](http://refhub.elsevier.com/S2590-0056(23)00024-3/sb12)

12