Design a Recommendation System in Real Estate Investment Based on Context Approach

Tinh T. Nguyen1,2, Sang Vu1,2,*, Truc Nguyen1,2, Vuong T. Pham2,3,4 and Hien D. Nguyen1,2

1University of Information Technology, Ho Chi Minh City, Vietnam
2Vietnam National University, Ho Chi Minh City, Vietnam
3University of Science, Ho Chi Minh City, Vietnam
4Institute of Data Science and Artificial Intelligence, Sai Gon University, Ho Chi Minh City, Vietnam

Keywords: Recommendation System, Consultation, Many-Cold-Start-Users, Nearest-Neighbors Approach, Intelligent System.

Abstract: The real estate investment industry has experienced a significant increase in user participation over the years, with individuals keen on registering concurrent interests in both recent and prior projects. This growing trend necessitates the development of an approach that can recommend real estate items in a simultaneous manner. However, the presence of unrequired memberships and stop-by behaviors has introduced several challenges, resulting in numerous cold-start scenarios for new users. This study proposes a recommendation system tailored specifically for real estate, designed to offer warm-start item recommendations of cold-start users using a content-based approach and a session-based recommendation system. Herein, a system for real estate recommendation with acceptable warm-start item recommendations is proposed in the many-cold-start-users scenario. The session-based recommendation system is adapted and made use of pre-existing methods to effectively handle sequential and contextual data for the encoded attribute prediction of the next-interacted item. Then, the nearest-neighbors method is employed weighted cosine similarity to identify conforming candidates. The results demonstrate the effectiveness of efficiently integrating the information and the difficulty in performing well in item recommendations simultaneously.

1 INTRODUCTION

Among the crucial challenges in any e-commerce is maintaining the existing users while attracting new ones and recommendation systems play a crucial role in enhancing the user experience and improving the overall functionality of a real estate investment platform. The common recommendation system uses historical records as prior knowledge to choose candidates and performs most effectively with adequate records (Roy and Dutta, 2022, Nguyen et al, 2020). However, the recommendation task becomes complex due to no clear data of new users which inevitably leads to more problems for the recommendation/consultation system. Therefore, addressing this cold-start problem is essential for enhancing the overall functionality and usability of real estate investment platforms.

The item cold-start problem occurs when a recommendation system cannot recommend new items due to record deficiencies and new listing omissions. Cold-start items are new items with few or no interactions (Wei et al., 2017), whereas the rest of the items are warm-start items. As new items are added continuously in practical applications, this problem can cause missed opportunities for recommendations, particularly in real estate recommendations wherein users can register a concurrent interest in recent and prior projects. Specific attributes of real estate, such as location, developer brand, and living space, can influence user behavior when searching and buying properties (Chia et al, 2016). Thus, a recommendation approach using these attributes to real-estate items simultaneously must be determined.

Context information is useful for recommendation tasks (Adomavicius and Tuzhilin,
2011). This information is also significant for real estate recommendation because user interests can vary according to the context. For example, users searching from urban areas may be more interested in condominiums than users searching from rural areas. Using context information is an approach to achieve better real estate recommendations. The context information is also represented by ontology (Bouihi and Bahaj, 2018, Nguyen et al., 2021) and graph information (Yan et al., 2019, Nguyen et al., 2023).

Deep learning techniques can be used to learn item attributes and predict the representation of real-estate items for the corresponding factorization machine (Wei et al., 2017). Meta-learning can also be applied (Vartak et al., 2017). However, these approaches utilize the factorization machine, which interprets the engagement or rating prediction to the recommendation task. This requires user identifiers and sufficient records for efficiency. Hence, this characteristic is also observed in other real estate search engines and e-commerce systems with unrequired membership, which encourages the use of a content-based approach and session-based recommendation system.

A session-based recommendation system (Hidasi et al., 2016) uses sequential behavior without relying on user identifiers. This paper proposes a real estate recommendation approach for solving the item cold-start problem with acceptable warm-start item recommendations in the cold-start-users scenario. It modifies a session-based recommendation system and employs existing mechanisms to efficiently deal with sequential and context information for the next-interacted item’s encoded attribute prediction. The nearest-neighbors approach is also used with weighted cosine similarity to determine conforming candidates. The experimental results show that the proposed method is not only among different applied mechanisms but also against baselines using the top-n recommendation with the real estate dataset.

2 PREMILARIES

2.1 Content-Based Recommendation System

Content refers to the attributes of an item; this can take the form of different data types, such as metadata and text description. The content-based recommendation system comprises a profile learner and a filtering component when working with structured item representations (Lops et al., 2011). The profile learner predicts the user profile from interacted item attributes in a similar representation to the item profile, after which the filtering component determines the relevant items using the matching algorithm. As it relies only on item attributes, it can constantly recommend cold-start items. This study tends to use the nearest-neighbors approach with weighted cosine similarity as a filtering component. Weighted cosine similarity is selected as the similarity function owing to its efficiency and flexibility with our user and item profiles, which are high-dimensional vectors. Weighted cosine similarity is defined as follows:

\[
\text{Similarity} = \frac{\sum w_{uv} u_i v_i}{\sqrt{\sum w_{uu} u_i^2} \sqrt{\sum w_{vv} v_i^2}}
\]

where \( u_i \) and \( v_i \) are components of vector \( u \) and \( v \), resp., \( w_i \) is the weight for both components.

2.2 Recurrent Recommendation System Without User Identifiers

Without a user identifier, the task of recommendation is underappreciated owing to the sparsity of training data (Zhang et al., 2019). This sparsity leads the recommendation system to learn from sequential interactions without using user identifiers. Many previous works (Li et al., 2017, Liu et al., 2018) relied only on the sequence of interactions in each session. Such a system is known as a session-based recommendation system and uses RNN (Recurrent Neural Network) as a core layer of the model owing to its capability for capturing sequential patterns. The system operates by receiving the click sequence of the session, \([e_1, e_2, \ldots, e_{n-1}, e_n]\) and predicting the next click \( e_{n+1} \) where \( e_i \) is the \( i \)th event of the session. The output of the system lists the scores for each item, after which the system recommends only the top-\( n \) highest-scored items to the user. The profile learner utilizes this structure to predict the user profile from the sequential patterns.

The process of splitting the click sequence into training sequences working with the corresponding structure is proposed in (Hidasi, 2016). Each training sequence contains the input sequences and ground truths. Input sequences use every possible prefix within the training sequence and subsequent clicks as ground truths (Figure 1). This generates an adequate number of training sequences for deep learning.

2.3 Attention Mechanism in the Current Recommendation System

A click sequence used in a recurrent recommendation system is implicit feedback. It is indirect feedback
implied from the user behavior and requires careful consideration due to its characteristics of being very noisy and providing no negative feedback (Hu et al., 2008). It is impossible to determine whether the users like or dislike the item on which they clicked, nor whether a click is a miss click. The profile learner uses the attention mechanism to deal with noise and capture the purpose of the sequence, giving precedence to each click differently.

Li et al. (2017) proposed the encoder portion of the neural attentive recommendation machine (NARM). NARM is an encoder-decoder session-based recommendation system with an attention mechanism. Its encoder portion incorporates two encoders, the global encoder, and the local encoder. The former represents the entirety of user behavior in the click sequence, i.e., the last hidden state of the RNN as follows:

$$c_g = h_t$$  \hspace{1cm} (2)

where, \(c_g\) is the output of the global encoder and \(h_t\) is the last hidden state of RNN. The local encoder represents the main purpose of the click sequence, defined as the sum of weighted hidden states from every time step as follows (3):

$$c_l = \sum_{j=1}^{n} a_j h_j$$  \hspace{1cm} (3)

where, \(c_l\) is the output of the local encoder, \(h_j\) is the hidden state of RNN at time step \(j\) and \(a_j\) is the weighted factor, which is defined as:

$$a_j = \frac{e^{score(h_j, h_j)}}{\sum_{k=1}^{n} e^{score(h_k, h_j)}}$$  \hspace{1cm} (4)

$$score(h_i, h_j) = A_1 \sigma(A_1 h_i + A_2 h_j)$$  \hspace{1cm} (5)

where, \(\sigma\) is an activation function, \(A_1\) is a weighting vector, and \(A_1\) and \(A_2\) are the learned weights of \(h_i\) and \(h_j\), respectively. As a result, both outputs from the global and local encoders are concatenated and used in the computation of the subsequent layers.

### 2.4 Application of Context Information in the Recurrent Recommendation System

Context information, such as the time and location of the requested service, is useful when applied to the recommendation task (Adomavicius and Tuzhilin, 2011). The profile learner uses the LC technique (Beutel et al., 2018) to efficiently incorporate context features, thereby overcoming difficulties inherent in increasing the dimension of inputs that entail more hidden units in the model. It works by determining the elements-wise product of all embedded context features in hidden states as follows:

$$h_j = (1 + \sum w_j) * h_j$$  \hspace{1cm} (6)

where \(h_j\) is the hidden state of RNN at time step \(j\) and \(w_j\) is the embedded context feature.

The embedding layer of each context feature is initialized by a 0-mean Gaussian distribution to ensure that the multiplicative term has a mean of 1. This initialization causes the multiplicative term to act like an attention mechanism in the hidden state. The element-wise product is performed both before and after passing through the RNN. These multiplications are considered as prefusion and post fusion, consecutively.

### 3 CONTENT-BASED APPROACH FOR RECOMMENDATION SYSTEM IN REAL ESTATE

Using the content-based approach having a profile learner and a filtering component, the item cold-start problem has been solved. The profile learner is a modified session-based recommendation system with an attention mechanism to predict user profiles using sequential and context information. The filtering component uses the nearest-neighbors approach to determine the most relevant items.

#### 3.1 Profile Learner

The profile learner predicts a user profile composed of the encoded attributes of the next-interacted item. It utilizes the click sequence and context information. Let \([e_1, e_2, ..., e_n]\) denote a click sequence wherein \(e_i\)
is the \(i\)th event of the sequence and \([c_1, c_2, \ldots, c_n, e_{n+1}]\) are context features where \(c_j\) corresponds to \(e_j\). The profile learner predicts \([f_1, f_2, \ldots, f_n, f_{n+1}]\) where \(f_i\) is the \(i\)th encoded feature of \(e_{n+1}\) determined from the click sequence and context features. The encoded feature is either one-hot or binary encoding depending on the possible number of classes. Each \(f_i\) prediction is either a multiclass or binary classification depending on ground truth encoding. For example, real estate projects have the number of bedrooms as a feature. This is reflected by one possible class among three: one, two, or three bedrooms. Therefore, predicting this feature is a multiclass classification problem. Furthermore, it is a multilabel classification problem when grouping such predictions as a feature prediction. Particularly, we predict the possibilities of all classes for each feature of the next-interacted item used them as a user profile.

The structure of the session-based recommendation system is modified its task to the objectives. Furthermore, the attention mechanism of the encoder portion of NARM is used and adopted the LC in the profile learner to efficiently deal with sequential and context information, consecutively. Thus, this efficiency should provide better user profile prediction results and both warm-start and cold-start item recommendations. Our profile learner received an embedded item identifier, numerical features, embedded categorical features, and the embedded context features of a click sequence as inputs and then predicted the user profile (Figure 2).

In the attention layer, global and local encoders are used similar to the encoder portion of NARM but using postfusion products instead of the original hidden states. This replacement includes the effect of context information when calculating the attention score. The output of the attention layer is a concatenated vector from the local and the global encoders, which is used by the fully connected layers to calculate the scores of the classes of all features. Each fully connected layer is responsible for only one encoded feature prediction; thus, its number of units is equal to the number of corresponding classes. Its activation function is either the softmax or sigmoid function for multiclass and binary classification.

### 3.2 Filtering Components

The filtering component is responsible for determining the candidates conforming to the predicted user profile through the matching algorithm. Herein, we used the nearest-neighbors approach to gather top-\(n\) related items by calculating the scores of all items using weighted cosine similarity, which considers the numerical values of every possibility in the user profile. It is also suitable for high-dimensional vectors, which are similar to both our user and item profiles. The representation of the item and user profiles must be the same to compute the similarity score. Thus, the concatenated vector of all encoded features is used. Regarding these profiles, multiclass and multilabel encoded features have different influences on the similarity score calculation owing to the different sum of values within the vector. To deal with this, we selected weighted cosine similarity over cosine similarity because it is flexible to assign different weights to each component. Each \(w_i\) for (1) is defined as follows:

\[
 w_i = \begin{cases} 
 \frac{1}{n} & \text{if } i = 1 \\
 1 & \text{otherwise} 
\end{cases} 
\] (7)

Where, \(n\) is the number of classes of the corresponding multilabel feature. The structure of our filtering component is shown in Figure 3. It was used to calculate the similarity score between all item profiles and the predicted user profile using (1) and (7) together as the similarity function. This study designs a method recommending the top-\(n\) items with the highest similarity score to the user.

### 3.3 Real-Estate Pricing System

A real estate pricing system is a tool or model used to estimate the value or price of properties in the real estate market. It utilizes various factors and data points to provide insights into property valuation, helping buyers, sellers, and real estate professionals make informed decisions. A typical real estate pricing system includes:

- **Data collection**: A real estate pricing system collects a wide range of data related to the properties being analyzed. The system may also consider additional data sources such as demographic information, economic indicators, and infrastructure development plans.

- **Feature selection**: Once the data is collected, the pricing system identifies the most relevant features that significantly impact property prices. This process involves statistical analysis and machine learning techniques to determine which variables have the strongest correlation with property values.

- **Model development**: The pricing system utilizes various statistical and machine learning models to build a prediction model. Commonly used models include multiple linear regression, random forests, or neural networks. The choice of model depends on the complexity of the data and the accuracy required for price estimation.
3.4 Regression Tree

A regression tree is a machine learning algorithm used for solving regression problems. It is a type of decision tree where each internal node performs a feature or an attribute, and each leaf node represents a prediction or an estimated value for the target variable.

The process of building the regression tree involves recursively partitioning the feature space based on the values of different input features. The goal is to create homogeneous regions or subsets of the data that share similar characteristics in terms of the target variable. This partitioning is done by selecting the feature and the corresponding threshold that results in the best split, often measured using metrics like mean squared error (MSE) or mean absolute error (MAE).

3.5 Linear Regression

For the dataset of house prices and their corresponding sizes, a linear regression model is built to predict the price of a house based on its size, number of bedrooms, number of toilets, number of floors as Table 1.

<table>
<thead>
<tr>
<th>Order</th>
<th>Price</th>
<th>Area</th>
<th>Bedrooms</th>
<th>Toilets</th>
<th>Floors</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10</td>
<td>126</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>15</td>
<td>152</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>17</td>
<td>116</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>11</td>
<td>133</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 1: Some information of a house.

The linear relationship between the area, number of bedrooms/toilets/floors of the house (variables $x_1, x_2, x_3, x_4$) and its price (dependent variable $y$) was found. It can be represented by the equation:

$$y' = f(x) = w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + c \quad (8)$$

where, $w = \{w_1, w_2, w_3, w_4\}$ is the slope (regression coefficient) and $c$ is the intercept ($y$-intercept).

The linear regression model provides a linear approximation of the relationship between house size and price. It assumes a constant slope, indicating that the price increases linearly with the size of the house. However, it's important to note that linear regression may not capture complex nonlinear relationships or interactions between variables.

Figure 2: The structure of the proposed profile leaner.

Figure 3: The structure of the proposed filtering component.
4 EXPERIMENTAL RESULTS

This section details the dataset and data preparation. It also explains how the experiment was conducted and evaluated to compare the performance of the proposed approach and selected baselines.

4.1 Dataset and Data Preparation

The dataset used herein comprised one year of website records captured in 2020 obtained from website batdongsan.com.vn, a popular real estate search engine website, which included 13,425,274 interactions between 305,019 users and 6,849 items. It contains 6,917 items with metadata that were used as candidates; these metadata were consequently processed as the item profile for each corresponding item. It means each interaction has its context features.

First, the item profiles from its metadata are provided the ground truth of the user profile prediction and the participant in the similarity score calculation. These metadata comprised 38 features, some of which have missing and unique values. Features, which have many unique values or missing values of more than half, are removed. Those are not effective to interpret. Besides, performing data imputation to fill the remaining missing values based on other feature values: the type of real estate, location, price level, etc.

Thereafter, all values are assigned into classes depending on their corresponding feature characteristic. Each value is assigned into a class with an upper bound for discrete numerical features. Values greater than or equal to this upper bound were categorized equally. For example, the number of bedrooms is a discrete numerical feature with five classes: one, two, three, four, and five or more bedrooms. This approach entailed the limitation of the number of classes by grouping several sample classes.

After performing classifications, using either one-hot or binary encoding encodes assigned class for one possible class and many possible classes, respectively. As a result, we generated item profiles composed of 15 encoded features for 6,917 items, the information relating to which is shown in Table 2.

Finally, sets of sequences of context features is divided into the training and testing sets. These included context features for the prefusion. Seven context features were used: n_click, requested device, requested province, requested country, user agent, page reference, and delta time. These features and outline their participation are described in Table 3.

4.2 Evaluation

The top-n recommendation task is used for evaluation because it is practical for real usage. This task evaluates performance based on the recommendation list provided by the system and the actual click of the user. The appropriate metrics are Recall@K and Mean Reciprocal Rank@K (MRR@K), where K is the number of items in the recommendation list. Recall@K measures the model performance whether the actual click is on the K-items recommendation list.

\[
\text{Recall@K} = \frac{n_{hit}}{N}
\]

where, \(n_{hit}\) is the number of cases having the actual click and \(N\) is the number of all cases.

MRR@K measures the ranking performance of the model as an average of reciprocal ranks of the actual click within the recommendation list as follows:

\[
\text{MRR@K} = \frac{1}{N} \sum_{c \in C} \frac{1}{\text{rank}(c)}
\]

where, \(c\) is the actual click, \(C\) is a set of cases having the actual click, \(N\) is the number of all cases.

The model performance is evaluated in terms of two aspects: warm-start and cold-start item recommendations. We assigned test cases to each perspective using the type of their actual click item. We defined new items appearing only in the testing set and the top 100 most recently introduced items in the training set.

Figure 4 is the results of items ranking in real-estate dataset by their price for inputted requirements.

![Figure 4: The results for ranking of items by their price.](image)

4.3 Comparison Against Other Methods

Baselines were categorized by their strengths in terms of two aspects: warm-start and cold-start item recommendations. There are warm-start item baselines:
• S-Pop: A sequence popularity predictor that recommends items ranked by their number of interactions in the current sequence.

• Item-KNN (Sarwar et al., 2001): It uses the nearest-neighbors approach with cosine similarity between the recently interacted item vector in the current sequence and other item vectors to obtain the most relevant items.

• NARM (Li et al., 2017): An encoder-decoder GRU-based session-based recommendation system with an attention mechanism.

• STAMP (Liu et al., 2018): An MLP-based session-based recommendation system with an attention mechanism. It creates a recommendation list based on sequential clicks.

The content-based approach (Lops e al., 2011) with different profile learners and similar filtering components for the cold-start item baselines. They were as follows:

• CB (S-Pop): A sequence popularity predictor that predicts the user profile using the item profile of the most interacted item in the current sequence.

• CB (Mean): A model that predicts the user profile from the meaning of the item profiles in the current sequence.

Table 2: Description, type, and post-processed information of the encoded attributes within the item profiles.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Encoded Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>detail_Id</td>
<td>Address</td>
<td>One-hot</td>
<td>Unique real-estate ID</td>
</tr>
<tr>
<td>address</td>
<td>number</td>
<td>One-hot</td>
<td>Unique province/city ID</td>
</tr>
<tr>
<td>square</td>
<td>number</td>
<td>One-hot</td>
<td>Total space of real-estate</td>
</tr>
<tr>
<td>numBedroom</td>
<td>number</td>
<td>Binary</td>
<td>Number of bedrooms</td>
</tr>
<tr>
<td>numBathroom</td>
<td>number</td>
<td>Binary</td>
<td>Number of bathrooms</td>
</tr>
<tr>
<td>numFloor</td>
<td>number</td>
<td>Binary</td>
<td>Number of floors</td>
</tr>
<tr>
<td>unitPrice</td>
<td>string</td>
<td>One-hot</td>
<td>Price per unit (m2)</td>
</tr>
<tr>
<td>totalPrice</td>
<td>string</td>
<td>One-hot</td>
<td>Total price of real-estate</td>
</tr>
<tr>
<td>legalInfo</td>
<td>string</td>
<td>One-hot</td>
<td>Legal documents information</td>
</tr>
<tr>
<td>directOfHouse</td>
<td>string</td>
<td>One-hot</td>
<td>Direction of real-estate</td>
</tr>
<tr>
<td>directOfBalcony</td>
<td>string</td>
<td>One-hot</td>
<td>Direction of balcony real-estate</td>
</tr>
<tr>
<td>furnitureInfo</td>
<td>string</td>
<td>Binary</td>
<td>Furniture information</td>
</tr>
</tbody>
</table>

Table 3: Description and participation of context features in prefusion and postfusion.

<table>
<thead>
<tr>
<th>Feature Name</th>
<th>Description</th>
<th>Prefusion</th>
<th>Postfusion</th>
</tr>
</thead>
<tbody>
<tr>
<td>n_click</td>
<td>Number of consecutive clicks with the same item</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>requested_device</td>
<td>Type of the used device (Desktop and Mobile)</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>requested_province</td>
<td>Province ID where user uses the service</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>requested_country</td>
<td>Country ID where user uses the service</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>user_agent</td>
<td>Operation system of the used device (i.e. Android, Windows, etc.)</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>page_referrer</td>
<td>Previous page before the current browsing</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>delta_time</td>
<td>Time between current click and last click in hour</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 4: Comparison with warm-item test cases.

<table>
<thead>
<tr>
<th>Method</th>
<th>Recall@1</th>
<th>Recall@5</th>
<th>MRR@5</th>
<th>Recall@10</th>
<th>MRR@10</th>
<th>Recall@15</th>
<th>MRR@15</th>
<th>Recall@20</th>
<th>MRR@20</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-Pop</td>
<td>6.49</td>
<td>18.35</td>
<td>11.07</td>
<td>21.02</td>
<td>11.44</td>
<td>21.88</td>
<td>11.51</td>
<td>22.43</td>
<td>11.54</td>
</tr>
<tr>
<td>Item – KNN</td>
<td>8.95</td>
<td>25.64</td>
<td>14.8</td>
<td>36.56</td>
<td>16.25</td>
<td>43.49</td>
<td>16.8</td>
<td>48.49</td>
<td>17.08</td>
</tr>
<tr>
<td>NARM</td>
<td>13.75</td>
<td>32.79</td>
<td>20.62</td>
<td>43.39</td>
<td>21.93</td>
<td>49.91</td>
<td>22.45</td>
<td>54.59</td>
<td>22.71</td>
</tr>
<tr>
<td>STAMP</td>
<td>14.09</td>
<td>33.14</td>
<td>20.88</td>
<td>43.55</td>
<td>22.27</td>
<td>49.98</td>
<td>22.78</td>
<td>54.62</td>
<td>23.04</td>
</tr>
<tr>
<td>CB (S-Pop)</td>
<td>6.49</td>
<td>13.96</td>
<td>9.12</td>
<td>18.83</td>
<td>9.76</td>
<td>22.16</td>
<td>10.02</td>
<td>24.77</td>
<td>10.17</td>
</tr>
<tr>
<td>CB (Mean)</td>
<td>6.03</td>
<td>19.85</td>
<td>11.14</td>
<td>26.81</td>
<td>12.06</td>
<td>31.45</td>
<td>12.43</td>
<td>35.05</td>
<td>12.63</td>
</tr>
<tr>
<td>Proposed approach</td>
<td>11.97</td>
<td>24.21</td>
<td>16.29</td>
<td>31.90</td>
<td>17.31</td>
<td>37.13</td>
<td>17.72</td>
<td>41.23</td>
<td>17.95</td>
</tr>
</tbody>
</table>
Table 4 is the results of warm-start item recommendation through 12,066 test cases. Moreover, to identify conforming candidates, the nearest-neighbors method is also combined with weighted cosine similarity. In terms of Recall@K, the proposed approach is mostly fourth following Item-KNN, NARM, and STAMP, respectively. The only exception in terms of Recall@1 is that our approach is at the third place by outperforming Item-KNN. In terms of MRR@K, our approach is at the third place behind NARM and STAMP, consecutively. Meanwhile, our approach yields better performance in both terms compared to cold-start item baselines. These results are better in terms of ranking and one-item recommendation. The proposed approach cannot beat Item-KNN, NARM, and STAMP in overall warm-start item recommendation owing to two causes. The first is having more candidates. There are 6,917 considered items when calculating the similarity score, out of which not all participate in the interaction logs. Conversely, these three baselines consider only 6,105 items found in the training set. The second is the disadvantage of using the only item attributes to determine the candidates. This results in retrieving only items like to the predicted user profile while users can register their interests with different attributes.

After weighing every feature equally when computing the similarity score, this might not match with user’s attributes priority. There could be user specific requirements when searching for real estate. Hence, making this approach more personalized by incorporating different weights for each feature can improve the recommendation performance.

5 CONCLUSION

In this paper, an approach for building a recommendation system in real-estate is proposed. In the case of numerous cold-start customers, this method resolves the item cold-start problem with considerable cold-start item recommendations. It adapts a session-based recommendation system and makes use of already in place methods to effectively handle sequential and contextual data for the encoded attribute prediction of the next interacted item. The experimental results demonstrate that this method is superior to baselines utilizing the top-n recommendation with the dataset from the real estate search engine as well as other used methods.

Based on the idea, people in the same group should react similarly to similar items, the recommendation approach improves search results by using customer demographic data (Matuszelański and Kopczewska, 2022). In the future, the proposed method will be combine the knowledge base of real-estate investment (Nguyen et al., 2022) for recommending more accuracy based on customers’ behaviors, which will be aimed at the demographic profile of customer.

ACKNOWLEDGMENT

This research was supported by The VNUHCM-University of Information Technology’s Scientific Research Support Fund.

REFERENCES


