Received March 27, 2021, accepted April 19, 2021, date of publication May 4, 2021, date of current version May 13, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3077564

Real Estate Recommendation Approach for Solving the Item Cold-Start Problem

JIRUT POLOHAKUL, EKAPOL CHUANGSUWANICH, ATIWONG SUCHATO, AND PROADPRAN PUNYABUKKANA

Department of Computer Engineering, Chulalongkorn University, Bangkok 10330, Thailand

Corresponding author: Proadpran Punyabukkana (proadpran.p@chula.ac.th)

This work was supported in part by Home Dot Tech Company Ltd.

ABSTRACT The item cold-start problem occurs when a recommendation system cannot recommend new items owing to record deficiencies and new listing omissions. When searching for real estate, users can register a concurrent interest in recent and prior projects. Thus, an approach to recommend cold-start and warm-start items simultaneously must be determined. Furthermore, unrequired membership and stop-by behavior cause real estate recommendations to have many cold-start and new users. This characteristic encourages the use of a content-based approach and a session-based recommendation system. Herein, we propose a real estate recommendation approach for solving the item cold-start problem with acceptable warm-start item recommendations in the many-cold-start-users scenario. We modify a session-based recommendation system and employ existing mechanisms to efficiently deal with sequential and context information for the next-interacted item's encoded attribute prediction. Subsequently, we use the nearest-neighbors approach using weighted cosine similarity to determine conforming candidates. We use Recall@K and MRR@K with the top-n recommendation to evaluate warm-start and cold-start item recommendations among different applied mechanisms and against the baselines. The results demonstrate the effectiveness of efficiently integrating the information and the difficulty in performing well in warm-start and cold-start item recommendations simultaneously. Our proposed approach illustrates the capability of solving the item cold-start problem while yielding promising results in both recommendations although neither result is the best. We believe that our approach provides a suitable compromise between both recommendations and that it will benefit recommendation tasks focusing on both recommendations.

INDEX TERMS Context awareness, machine learning, recommender systems, recurrent neural networks.

I. INTRODUCTION

Among the crucial challenges in e-commerce is maintaining the existing users while attracting new ones. While the abundance of information and choices can be a deterrent, a common approach is to provide recommendations to users to reduce their time and effort searching for information, with the hope of increasing satisfaction. An example of this situation is when users search for real estate on the Internet. Generally, a recommendation system uses historical records as prior knowledge to choose candidates and performs most effectively with adequate records. However, the recommendation task becomes complex for new items, which inevitably leads to the item cold-start problem.

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 [30], 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. For instance, users who concern about a location can be interested in many real estate projects at a particular place regardless of the property age. Specific attributes of real estate, such as location, developer brand, and living space, can influence user behavior when searching and buying properties [2], [7], [21], [26]. Thus, a recommendation approach using these attributes to recommend cold-start and warm-start items simultaneously must be determined.

Item attributes have been used in previous studies to mitigate the item cold-start problem. Deep learning techniques can be used to learn item attributes and predict the

The associate editor coordinating the review of this manuscript and approving it for publication was Xianzhi Wang.

representation of cold-start items for the corresponding factorization machine [30]. Meta-learning can also be applied [22], [28]. 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; both types of data are insufficient in the case of real estate recommendation. In such situations, unrequired membership and stop-by behavior cause the system to have many newcomers and a large number of users with few records. In other words, it has many cold-start users and much more new users. We believe that 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 a session-based recommendation system. Unlike the factorization machine, a content-based approach [19] can instantly solve the item cold-start problem corresponding to any number of records by relying on item attributes. A session-based recommendation system [12] can use sequential behavior without relying on user identifiers. Therefore, we follow a content-based approach to solve the item cold-start problem. Furthermore, we use a session-based recommendation system for efficient learning of the user profile.

Context information is useful for recommendation tasks [1]. We believe that 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. Thus, we apply context information to our approach to achieve better real estate recommendations.

Herein, we propose a real estate recommendation approach for solving the item cold-start problem with acceptable warm-start item recommendations in the many-cold-start-users scenario. We modify a session-based recommendation system and employ existing mechanisms to efficiently deal with sequential and context information for the next-interacted item's encoded attribute prediction. Subsequently, we use the nearest-neighbors approach using weighted cosine similarity to determine conforming candidates. Thereafter, we compare our proposed approach not only among different applied mechanisms but also against baselines using the top-n recommendation with the dataset from the real estate search engine. We evaluate recommendation systems with respect to two aspects: warm-start and cold-start item recommendations. This proposed approach, which addresses the item cold-start problem, will be beneficial for any recommendation system belonging to similar domains.

II. RELATED WORK

For real estate recommendations, Yuan *et al.* [32] employed a user-oriented recommendation system using ensemble techniques from case-based reasoning and ontological structures. Their system required user criteria and preferences as the knowledge from which the relevant items were found; however, this information was unavailable in our dataset. Furthermore, Yu *et al.* [31] proved that location is a significant feature via the addition of geographical proximity to the weighted-regularized factorization machine [23] using a method ensuring that real estate with geographical proximity has similar latent factors. However, their experiment used a dataset involving real estate from only one city. In this study, we use real estate information from a whole country. This is justified by the fact that users might be interested in various real estates from several distant locations. Badriyah *et al.* [3] proposed a property recommendation system based on content-based filtering and association rules. Their approach created user and item profiles from the collection of words in advertisements and performed term frequency-inverse document frequency (TF-IDF). Thereafter, it generated association rules using the user profiles as item sets via the apriori algorithm and recommended property products based on these rules. Knoll *et al.* [16] conducted an experiment on extracted real estate website data for comparing a deep learning approach with the factorization machine. They adapted neural collaborative filtering (NCF) [10] to consider item features together with user and item identifiers. Their results demonstrated that deep learning outperforms the factorization machine in both overall and cold-start results. Nevertheless, these methods omit sequential and context information and suffer from the item cold-start problem.

A deep learning approach has been applied to recommendation tasks for capturing sequential patterns [33]. It benefits from a nonlinear transformation, representation learning, and sequence modeling. Hidasi *et al.* [12] proposed the session-based recommendation system with the top-n recommendation task. It is a recurrent recommendation system without user identifiers using a recurrent neural network (RNN) with a gated recurrent unit (GRU) [8]. They designed a model capable of capturing sequential patterns from a click sequence within the session and predicting the next click. They also applied the long short-term memory (LSTM) [13]; however, this yielded discouraging results compared to the GRU. In their consequent work [11], they added another RNN to their recommendation system to incorporate item features into the model and thus proved the features' utility. Li *et al.* [17] applied the attention mechanism to session-based recommendation systems, thereby capturing the primary purpose of the session. Moreover, their model simultaneously captured global and local attentions, i.e., the final hidden state of RNN and the sum of weighted hidden states at every time step. Likewise, Liu *et al.* [18] used a multilayer perceptron (MLP) instead of an RNN. They employed the sum of weighted item representations as the global attention and the last item representation as the local attention. Their work demonstrated that MLP achieves computational efficiency and enhanced recommendations, particularly when considering a long sequence. Moreover, they proved that the last click in the user's session is the dominant control on the next click. Beutel *et al.* [5] found that

concatenating context features with the input is an inefficient means to incorporate them. A greater number of dimensions of concatenated input leads to requiring more units from the hidden layer to increase the efficiency of the model. Hence, they proposed a technique, latent cross (LC), performing element-wise products between embedded context features and the hidden states. Context information was incorporated both before and after being consecutively fed to the GRU as prefusion and postfusion, respectively. Although these works showed a capability of dealing with context information, item features, and sequential patterns, they suffered from the item cold-start problem.

To solve the item cold-start problem, Wei *et al.* [30] proposed a hybrid recommendation model combining a time-aware model, timeSVD++ [4], with a deep learning architecture and a stacked denoising autoencoder (SDAE) [29] for movie rating prediction. They used the descriptions of the cold-start item to predict its latent factor through the SDAE. Consequently, they found the top-n nearest warm-start items using Pearson's correlation coefficient and used the mean of their predicted ratings as the prediction. Vartak *et al.* [28] introduced a meta-learning perspective for solving the item cold-start problem in the recommendation system using a cold-start item representation together with user representation from a learned history to predict the engagement between the cold-start item and the user. Furthermore, they proposed a linear and nonlinear classifier with weight and bias adaptations, respectively. These two classifiers were separately used for each user with different weights or biases depending on specific histories. Pan *et al.* [22] used meta-embedding to assess the item cold-start problem in click-through rate prediction to make the model better at dealing with the cold-start and faster at warming-up. When a new item was detected, they used its features to learn the representation using a meta-embedding generator designed to update the weight with future interactions until it became warm. These works demonstrated that using item attributes enables solving the item cold-start problem. However, sequential information was omitted and the proposed systems were reliant on the factorization machines, which are considered improper for real estate recommendations.

Content-based recommendation systems [19] are another means of solving the item cold-start problem, and benefit from using only the attributes that can work with any number of records. Setiadi *et al.* [25] recommended scientific articles through content-based filtering using two matching algorithms, i.e., k-means clustering and cosine similarity. They used TF-IDF to represent both user and item profiles, and elucidated the advantage of using k-means clustering over cosine similarity to obtain relevant items. Luostarinen and Kohonen [20] used a form of topic modeling, latent dirichlet allocation (LDA) [6], to represent and recommend news through the naive Bayes classifier, nearest-neighbor regression, and linear regression with cosine similarity. Deldjoo *et al.* [9] integrated audio and visual descriptors to the hybrid recommendation system for solving the movie cold-start problem. They used metadata and extracted features to help recommend cold-start movies. The aforementioned works illustrated the advantages of content-based filtering for solving the item cold-start problem; however, they omitted sequential and context information, and their corresponding items do not include real estate.

Herein, we solve the item cold-start problem while using sequential and context information through a content-based approach by modifying a session-based recommendation system to be a profile learner. We also apply an attention mechanism and LC to efficiently deal with sequential and context information, respectively.

III. BACKGROUND

In this section, we provide knowledge for implementing our proposed approach. The proposed approach leverages a content-based recommendation system, a recurrent recommendation system without user identifiers, an attention mechanism in the recurrent recommendation system, and application of context information in the recurrent recommendation system.

A. 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 [19]. 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. Herein, we follow this approach to solve the item cold-start problem. We use the nearest-neighbors approach with weighted cosine similarity as a filtering component. We select weighted cosine similarity 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:

$$similarity = \frac{\sum_i w_i u_i v_i}{\sqrt{\sum_i w_i u_i^2}\sqrt{\sum_i w_i v_i^2}}, \tag{1}$$

where u_i and v_i are components of vector u and v respectively, and w_i is the weight corresponding to both components.

B. RECURRENT RECOMMENDATION SYSTEM WITHOUT USER IDENTIFIERS

Without a user identifier, the task of recommendation is underappreciated owing to the sparsity of training data [33]. This sparsity leads the recommendation system to learn from sequential interactions without using user identifiers. Many previous works [12], [17], [18], [27] relied only on the sequence of interactions in each session. Such a system is known as a session-based recommendation system and uses

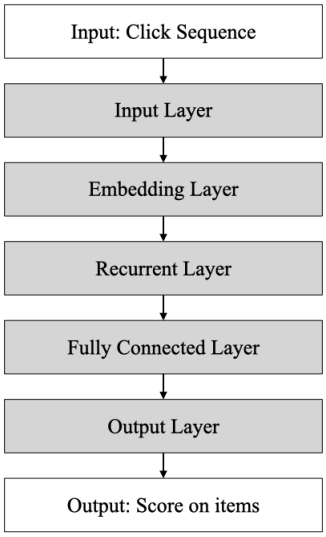

FIGURE 1. Structure of the session-based recommendation system.

RNN 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. This task is either a multiclass or binary classification treating each item as one class. 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 structure of the session-based recommendation system proposed in [12], as shown in Fig. 1, is used herein. Our profile learner utilizes this structure to predict the user profile from the sequential patterns.

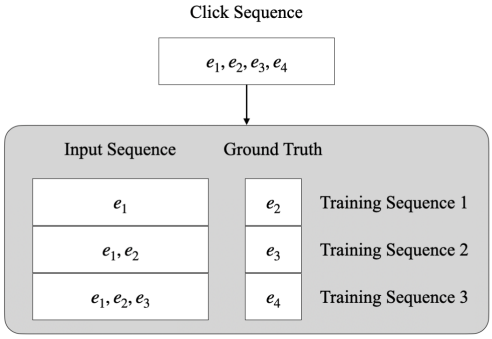

FIGURE 2. An example of splitting the click sequence into training sequences.

The process splitting the click sequence into training sequences working with the corresponding structure is proposed in [27]. Each training sequence contains the input sequences and ground truths. We obtain the input sequences using every possible prefix within the training sequence with the subsequent clicks as ground truths (Fig. 2). This enables us to generate an adequate number of training sequences for deep learning.

C. ATTENTION MECHANISM IN THE RECURRENT RECOMMENDATION SYSTEM

A click sequence used in a recurrent recommendation system is implicit feedback. It is an indirect feedback implied from the user behavior and requires careful consideration due to its characteristics of being very noisy and providing no negative feedback [14]. It is impossible to determine whether the users like or dislike the item on which they clicked, nor whether a click is a missclick. Our profile learner uses the attention mechanism to deal with noise and capture the purpose of the sequence, giving precedence to each click differently. Herein, we follow the encoder portion of the neural attentive recommendation machine (NARM) [17].

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:

$$c_g = h_t, \tag{2}$$

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:

$$c_l = \sum_{j=1}^{n} \alpha_j h_j, \tag{3}$$

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

$$\alpha_j = \frac{e^{score(h_t, h_j)}}{\sum_{j=1}^{n} e^{score(h_t, h_j)}}, \tag{4}$$

$$score(h_t, h_j) = A_3 \sigma(A_1 h_t + A_2 h_j), \tag{5}$$

where σ is an activation function, A_3 is a weighting vector, and A_1 and A_2 are the learned weights of h_t 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.

D. 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 [1]. Our profile learner uses the LC technique [5] 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:

$$h_j = (1 + \sum w_i) * h_j, \tag{6}$$

where h_j is the hidden state of RNN at time step j and w_i is the embedded context feature. The embedding layer of each

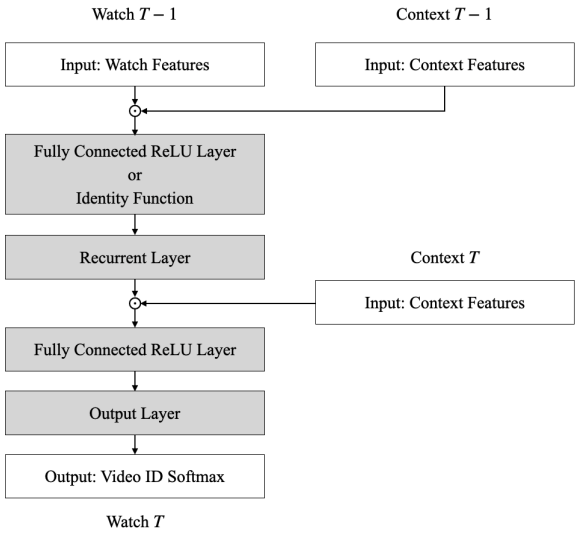

FIGURE 3. Structure of YouTube's recurrent recommendation system when applying the latent cross technique.

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 as shown in Fig. 3. These multiplications are considered as prefusion and postfusion, consecutively.

IV. PROPOSED METHOD

Our method follows the content-based approach having a profile learner and a filtering component to solve the item cold-start problem. Our 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. The following sections describe the implementation of these two parts.

A. 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 $[e_1, e_2, \ldots, e_{n-1}, e_n]$ denote a click sequence wherein e_i is the i^{th} event of the sequence and $[c_1, c_2, \ldots, c_n, c_{n+1}]$ are context features where c_i corresponds to e_i. The profile learner predicts $[f_1, f_2, \ldots, f_{m-1}, f_m]$ 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. Another feature is the unit type; real estate projects can have multiple unit types simultaneously, i.e., both a detached house and a semi-detached house in the same project. The prediction of each class is a binary 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 and use them as a user profile.

We leveraged the structure of the session-based recommendation system to modify its task to the objectives of this study. Furthermore, we used the attention mechanism of the encoder portion of NARM and adopted the LC in our 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, as shown in Fig. 4. We used GRU as a core layer because it uses sequential information without suffering a vanishing gradient problem and has advantageous features over LSTM. We performed prefusion and postfusion of the LC before and after passing through the GRU layer to efficiently incorporate context information into the model. In the attention layer, we used global and local encoders 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, respectively. As a result, all predicted encoded features are representative of the user profile.

B. FILTERING COMPONENT

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. Moreover, it is 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, we use the concatenated vector of all encoded features.

Regarding these profiles, multiclass and multilabel encoded features have different influences on the similarity score calculation owing to the different sum of values within

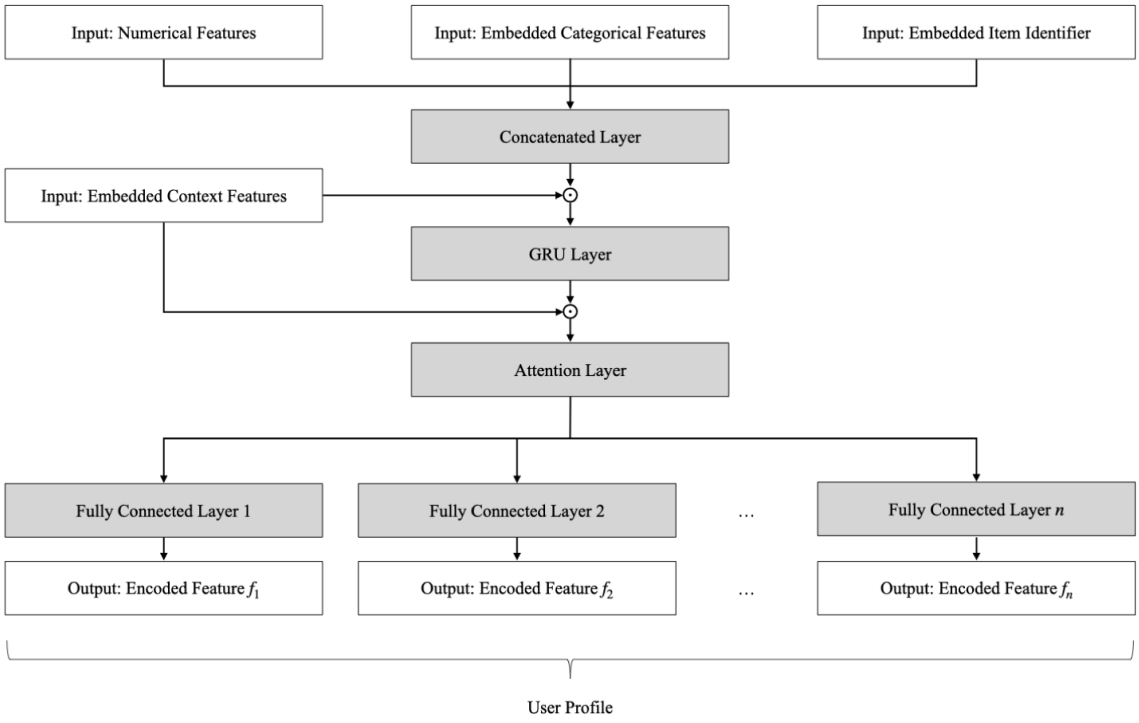

FIGURE 4. The structure of the proposed profile learner.

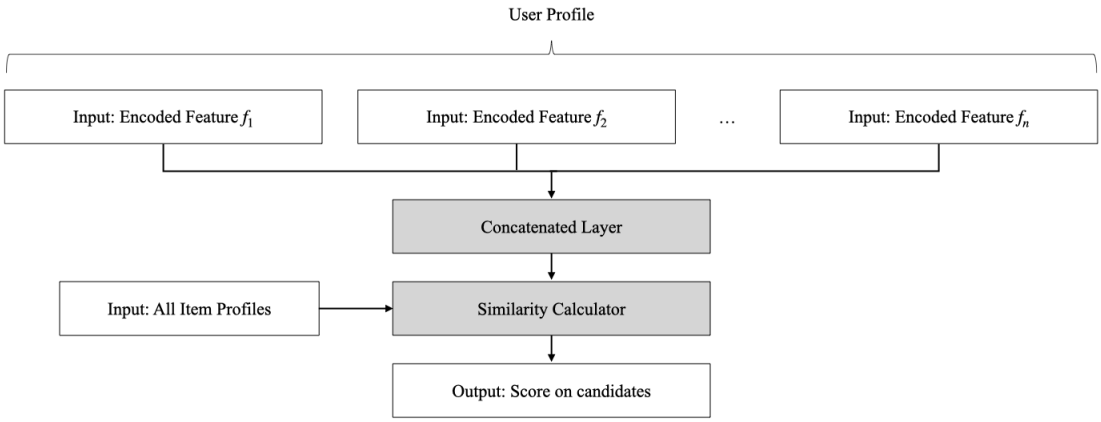

FIGURE 5. The structure of the proposed filtering component.

the vector. The former vector is guaranteed to have a sum of 1 whereas the latter's sum can be any value between 0 and the number of classes. To deal with this, we selected weighted cosine similarity over cosine similarity because it is flexible to assign different weights to each component. We reduced only the influence of multilabel feature component using one divided by its number of classes as a weight. In other words, we defined our w_i for (1) as follows:

$$w_i = \begin{cases} 1/n, & \text{if } u_i \text{ and } v_i \text{ are components} \\ & \text{of multilabel feature} \\ 1, & \text{otherwise,} \end{cases} \tag{7}$$

where n is the number of classes of the corresponding multilabel feature.

The structure of our filtering component is shown in Fig. 5. 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. However, because it is possible to have an equal score, this approach uses the overall number of interactions as the secondary score to break the equality. Particularly, we used popularity to order items with equal similarity scores. Our method thus recommends the top-n items with the highest similarity score to the user.

TABLE 1. Description, type, and post-processed information of the encoded attributes within the item profiles.

Feature Name	Type	Encoded Type	#Classes	Description
developer_id	categorical	one-hot	2,596	Unique developer ID
province_id	categorical	one-hot	60	Unique province ID
district_id	categorical	one-hot	222	Unique district ID
area_id	categorical	one-hot	36	Unique area ID
subarea_id	categorical	one-hot	285	Unique subarea ID
price_level_id	categorical	one-hot	10	Price level of the project
total_unit	continuous numerical	one-hot	5	Number of total units
functional_space	continuous numerical	one-hot	5	Total size of functional space in sq.m.
n_bedroom	discrete numerical	one-hot	5	Number of bedrooms
n_bathroom	discrete numerical	one-hot	5	Number of bathrooms
n_parking_lot	discrete numerical	one-hot	3	Number of parking lots
created_year	categorical	one-hot	12	Year the project was added to the database
project_status	categorical	binary	1	Status of the project
unit_types	categorical	binary	6	Types of units within the project
facilities	categorical	binary	6	Facilities within the project

V. EXPERIMENTAL SETUP

In this section, we detail our dataset and data preparation. We also explain how the experiment was conducted and evaluated to compare the performance of our approach not only among different applied mechanisms but also against selected baselines.

A. DATASET AND DATA PREPARATION

The dataset used herein comprised one year of website records captured in 2018 obtained from www.home.co.th, a popular real estate search engine website based in Thailand, which included 13,425,274 interactions between 3,005,019 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. Meanwhile, each interaction has its context features.

First, we prepared the item profile from its metadata to provide 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. We removed features with too many unique values because they were difficult to interpret. Likewise, we erased features that have missing values of more than half. Subsequently, we performed data imputation to fill the remaining missing values based on other feature values: the unit type, developer, location, and price level. Thereafter, we differently assigned all values into classes depending on their corresponding feature characteristic. We used each unique value as a class when it was categorical. Likewise, we assigned a value to classes 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. Continuous numerical features were normalized by taking the logarithm into account and categorizing it according to the percentile. After performing classifications, we encoded the assigned class using either one-hot or binary encoding. We used these processes for cases of 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 1.

Second, we processed the interaction records for training and testing the model. We maintained only interactions with items having an item profile because our approach relied on their attributes. These interactions were grouped and sorted by user identifiers and timestamps to provide click sequences. However, many continuously repeated-item interactions occurred owing to consequent clicks on the same item. For example, users pressed the project detail button, then the map button, and then the contact button. This led to a sequence of three interactions with the same item. We mitigated this problem by grouping such consequent clicks into one click and using the number of clicks as a new context feature. From the aforementioned example, we retained only one interaction with these three clicks as context information in the sequence. After accounting for this repetition, some users still showed far greater usage than others. These were categorized as bots by plotting the distribution of the number of interactions, using the last two percentiles, and eliminating their records. Afterward, we split the sequences into a training set and a testing set. The first nine-month interactions within provided the training set, whereas the rest formed the testing set. We also removed sequences with one click, since they were unlearnable for the sequential model. Subsequently, we separated the remaining materials into training and testing sequences using the same method as shown in Fig. 2, obtaining input sequences and ground truths for both the training and the testing sets. However, two sets of ground truths are required. The first comprised the next-clicks of the input sequences whereas the second one included their encoded features. To obtain the second set, we replaced the next clicks in the first set with their item profile. These ground truths were used for the top-n recommendation evaluation and training the profile learner, respectively. Finally, 2,423,585 sequences remained

TABLE 2. Statistics of training and testing set.

	#Users	#Items	#Interactions	#Sequences
training set	665,501	6,105	3,089,086	2,423,585
testing set	221,837 (192,109 new users)	6,213 (250 new items)	1,005,916	784,079

TABLE 3. Description and participation of context features in prefusion and postfusion.

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

in the training set and 784,079 sequences in the testing set. The statistics of these sets are shown in Table 2. They show that new items and users were added to the testing set during the past three months. Although the number of new items is not high, the item cold-start problem does exist. Moreover, the number of new users and interactions emphasize the characteristic of having many cold-start users.

Finally, we prepared two sets of sequences of context features for the training and testing sets. These included context features for the prefusion and postfusion of the LC. Seven context features were used: n_click, requested device, requested province, requested country, user agent, page reference, and delta time. The prefusion considered all such features in its calculation. However, postfusion considered only the last six features because the next-interaction had not yet taken place and the first feature was unobtainable. We describe these features and outline their description and participation in prefusion and postfusion in Table 3.

B. EVALUATION SETUP

We used the top-n recommendation task 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. It thus denotes a proportion of the number of cases containing the actual click among all cases as follows:

$$Recall@K = \frac{n_{hit}}{N}, \tag{8}$$

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:

$$MRR@K = \frac{1}{N}\sum_{c\in C}\frac{1}{rank(c)}, \tag{9}$$

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

Herein, we used Recall@K and MRR@K wherein K in {1, 5, 10, 15, 20} was used as the evaluation metric, which recommend K-items simultaneously. We also evaluated the model performance 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 as cold-start items, whereas the rest were defined as warm-start items. The earliest timestamp of cold-start items defined in the training set was about one month before the splitting point. The numbers of average interactions were 109.84 and 512.59 for cold-start and warm-start items in the training set, respectively. We ended up with 728,066 warm-start item test cases and 56,013 cold-start item test cases.

C. IMPLEMENTATION DETAILS

Our profile learner used the rectifier linear unit (ReLU) as the activation function of the GRU, whereas its local encoder used the hard sigmoid. It also used categorical cross-entropy and binary cross-entropy for the loss function of output layers with multiclass and binary classification, respectively. During the training process, the batch size was fixed at 512 and we used 10% of the training set as the validation set to indicate the best version of the model. We set the number of epochs to 30, used Adam [15] as the optimizer, and applied the model with the lowest loss in the validation set. We adjusted the hyperparameters through grid search. Hyperparameters were as follows: learning rate in {0.0001, 0.001}, categorical feature embedding size in {15, 30, 45}, context feature embedding size in {110, 200, 290}, and number of hidden units in {110, 200, 290}. We treated the embedded item identifier as the categorical feature; hence, its embedding size varied according to the categorical embedding size. The LC restricted the last two hyperparameters to be affected by the categorical feature embedding size owing to element-wise multiplication. These parameters were required to be equal to the sum of the size of the embedded categorical features

TABLE 4. Comparison between the performances of the proposed approach among different applied mechanisms with 728,066 warm-item test cases.

Mechanism										
NE	LC	WCS	Recall@1	Recall@5	MRR@5	Recall@10	MRR@10	Recall@15	MRR@15	Recall@20	MRR@20
-	-	-	8.80%	20.97%	13.06%	28.89%	14.11%	34.42%	14.55%	38.62%	14.78%
-	-	✓	9.02%	21.78%	13.74%	29.84%	14.81%	35.37%	15.25%	39.60%	15.48%
-	✓	-	10.22%	22.70%	14.62%	30.54%	15.66%	36.00%	16.09%	40.14%	16.32%
-	✓	✓	10.86%	23.52%	15.33%	31.45%	16.38%	36.85%	16.80%	41.01%	17.04%
✓	-	-	9.92%	22.38%	14.30%	30.19%	15.33%	35.65%	15.76%	39.82%	16.00%
✓	-	✓	10.27%	22.94%	14.74%	30.91%	15.79%	36.30%	16.22%	40.49%	16.45%
✓	✓	-	11.50%	23.60%	15.76%	31.24%	16.77%	36.56%	17.19%	40.59%	17.42%
✓	✓	✓	**11.97%**	**24.21%**	**16.29%**	**31.90%**	**17.31%**	**37.13%**	**17.72%**	**41.23%**	**17.95%**

TABLE 5. Comparison between the performances of the proposed approach among different applied mechanisms with 56,013 cold-item test cases.

Mechanism										
NE	LC	WCS	Recall@1	Recall@5	MRR@5	Recall@10	MRR@10	Recall@15	MRR@15	Recall@20	MRR@20
-	-	-	1.71%	8.14%	3.76%	13.33%	4.43%	17.37%	4.75%	20.69%	4.94%
-	-	✓	2.02%	9.02%	4.23%	15.74%	5.11%	20.62%	5.50%	24.46%	5.71%
-	✓	-	2.20%	8.44%	4.26%	13.50%	4.92%	17.51%	5.23%	20.86%	5.42%
-	✓	✓	2.43%	9.41%	4.67%	16.24%	5.57%	20.92%	5.94%	24.51%	6.14%
✓	-	-	2.45%	9.35%	4.69%	14.80%	5.40%	19.03%	5.73%	22.18%	5.91%
✓	-	✓	2.56%	10.20%	5.01%	17.08%	5.92%	21.80%	6.29%	25.40%	6.49%
✓	✓	-	3.52%	10.30%	5.77%	15.64%	6.47%	19.85%	6.80%	23.10%	6.98%
✓	✓	✓	**3.60%**	**11.49%**	**6.18%**	**18.08%**	**7.05%**	**22.82%**	**7.43%**	**26.33%**	**7.62%**

and numerical features. As a result, the optimized hyperparameters are as follows: learning rate = 0.0001, categorical feature embedding size = 45, context feature embedding size = 290, and the number of hidden units = 290.

VI. EXPERIMENTAL RESULT AND DISCUSSION

In this section, we report and discuss the evaluation of our approach in two parts. The first part is a comparison among different applied mechanisms. The latter is a comparison against selected baselines.

A. COMPARISON AMONG DIFFERENT MECHANISMS

There are three applied mechanisms within our approach: NARM's encoder (NE), LC, and weighted cosine similarity (WCS). NE and LC were applied to the profile learner, whereas WCS was applied to the filtering component. This evaluation compares the performances among with and without each mechanism. We explained the changes when we do not apply each mechanism as follows:

- without NE: Attention layer of the profile learner was removed.
- without LC: Embedded context features were incorporated by concatenating instead of performing element-wise product.
- without WCS: Cosine similarity was used as the similarity function in the filtering component instead of the weighted version.

We present evaluation results in Table 4 and Table 5. For the profile learner, the results show that applying either NE or LC provides a better recommendation in both warm-start and cold-start item test cases. Moreover, using them together yields the best performance because the profile learner can efficiently incorporate context information and consider it when computing the attention score. For the filtering component, using WCS helps our approach recommend better in both cases, particularly in the cold-starts. This indicates that reducing the influence of the multilabel feature's value improves the performance of similarity score calculation. As a result, the best performances of our approach with all applied mechanisms are used to compare with selected baselines in the following part.

B. COMPARISON AGAINST SELECTED BASELINES

Baselines were categorized by their strengths in terms of two aspects: warm-start and cold-start item recommendations. Warm-start item baselines were as follows:

- Pop: A popularity predictor that recommends items ranked by their overall number of interactions.
- S-Pop: A sequence popularity predictor that recommends items ranked by their number of interactions in the current sequence. The remaining positions of the recommendation list are filled with the nonduplicated items from Pop.
- Item-KNN [24]: A memory-based collaborative filtering recommendation system that recommends items using the item vectors obtained from the rating matrix. It uses the nearest-neighbors approach with cosine similarity between the recent interacted item vector in the current sequence and other item vectors to obtain the most relevant items.
- NARM [17]: An encoder-decoder GRU-based session-based recommendation system with an attention mechanism. It captures the entirety of user behavior and the main purpose of user behavior through global and local encoders, respectively, and then recommends items based on them.

TABLE 6. Comparison between the performances of the proposed approach and selected baselines with 728,066 warm-item test cases.

Method	Recall@1	Recall@5	MRR@5	Recall@10	MRR@10	Recall@15	MRR@15	Recall@20	MRR@20
Pop	0.35%	1.34%	0.68%	2.16%	0.78%	2.82%	0.84%	3.41%	0.87%
S-Pop	6.49%	18.35%	11.07%	21.02%	11.44%	21.88%	11.51%	22.43%	11.54%
Item-KNN	8.95%	25.64%	14.80%	36.56%	16.25%	43.49%	16.80%	48.49%	17.08%
NARM	13.75%	32.79%	20.52%	43.39%	21.93%	49.91%	22.45%	54.59%	22.71%
STAMP	**14.09%**	**33.14%**	**20.88%**	**43.55%**	**22.27%**	**49.98%**	**22.78%**	**54.62%**	**23.04%**
CB (Pop)	0.35%	0.47%	0.38%	0.56%	0.39%	0.56%	0.39%	0.72%	0.40%
CB (S-Pop)	6.49%	13.96%	9.12%	18.83%	9.76%	22.16%	10.02%	24.77%	10.17%
CB (Mean)	6.03%	19.85%	11.14%	26.81%	12.06%	31.45%	12.43%	35.05%	12.63%
Proposed Approach	11.97%	24.21%	16.29%	31.90%	17.31%	37.13%	17.72%	41.23%	17.95%

TABLE 7. Comparison between the performances of the proposed approach and selected baselines with 56,013 cold-item test cases.

Method	Recall@1	Recall@5	MRR@5	Recall@10	MRR@10	Recall@15	MRR@15	Recall@20	MRR@20
Pop	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
S-Pop	2.20%	14.96%	7.02%	18.63%	7.53%	19.61%	7.61%	19.78%	7.62%
Item-KNN	0.73%	2.93%	1.48%	4.50%	1.67%	6.15%	1.80%	7.64%	1.88%
NARM	3.81%	7.32%	5.09%	9.37%	5.36%	10.82%	5.47%	12.00%	5.54%
STAMP	3.59%	6.89%	4.79%	8.80%	5.04%	10.09%	5.15%	11.13%	5.20%
CB (Pop)	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
CB (S-Pop)	2.23%	8.02%	4.31%	12.24%	4.86%	14.76%	5.06%	17.10%	5.19%
CB (Mean)	**4.96%**	**18.67%**	**10.19%**	**24.72%**	**10.99%**	**28.42%**	**11.28%**	**31.39%**	**11.45%**
Proposed Approach	3.60%	11.49%	6.18%	18.08%	7.05%	22.82%	7.43%	26.33%	7.62%

- STAMP [18]: An MLP-based session-based recommendation system with an attention mechanism. It creates a recommendation list based on sequential clicks made by the user and can effectively capture both long-term and short-term user interests from the sequence.

These baselines focus on recommending known items from the training set and not mitigating the item cold-start problem. They are unable to recognize new items and their interactions. Therefore, we removed the new item interactions in the testing set sequences to enable these baselines to perform. Moreover, Recall@K and MRR@K were set to zero when the actual click was the new item. We used only the content-based approach [19] with different profile learners and similar filtering components for the cold-start item baselines. They were as follows:

- CB (Pop): A popularity predictor that predicts the user profile using the item profile of the most interacted item. It uses the nearest-neighbors approach with WCS to obtain the most relevant items.
- 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. It uses the nearest-neighbors approach with WCS to obtain the most relevant items.
- CB (Mean): A model that predicts the user profile from the mean of the item profiles in the current sequence. It uses the nearest-neighbors approach with WCS to obtain the most relevant items.

We present evaluation result of warm-start item recommendation in Table 6. In terms of Recall@K, our 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. Although these results do not match our expectation of outperforming Item-KNN, they are better in terms of ranking and one-item recommendation. Our 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 the items similar to the predicted user profile while users can register their interests in items with different attributes.

We present the evaluation results of cold-start item recommendation in Table 7. In terms of Recall@K, our approach is mostly in second following CB (Mean). The exceptions in terms of Recall@5 and Recall@10 are that our approach is at the third place following S-Pop. In terms of MRR@K, our approach is placed third following S-Pop and CB (Mean). Meanwhile, it performs considerably in both terms compared to Item-KNN, NARM, and STAMP. This evaluation result shows that our approach is not the best method to recommend cold-start items. Using the mean of attributes provides a better user profile than predicting from sequential and context information when recommending these items. Moreover, recommending items based on the popularity of the current sequence performs better than our approach in terms of ranking and 5-item and 10-item recommendations. However, this result does indicate that our approach can solve the item cold-start problem.

When integrating these results, it is difficult for a method to perform well in both recommendations simultaneously. The methods suitable with warm-start items showed meager performance with cold-start items. Likewise, the methods

suitable for cold-start items yielded dissatisfactory performance when recommending warm-start items. In detail, according to Recall@20, the first, second, and third places in warm-start item recommendation are occupied by STAMP, NARM, and Item-KNN, respectively. Despite decent performances with warm-start items, they are placed sixth, fifth, and seventh in the cold-start item recommendation, consecutively. Similarly, CB (Mean), the first place in cold-start item recommendation, is in fifth place when recommending warm-start items. Nevertheless, our approach is in fourth place and second place in warm-start and cold-start item recommendations, respectively. It yields promising results in both recommendations at the same time although neither result is the best. It is particularly important for real estate recommendations to be able to perform well in both recommendations because each real estate listing is unique and users can register concurrent interest in prior and recent projects. Thus, we believe that our method provides a suitable compromise between both recommendations and that it will benefit recommendation tasks focusing on both recommendations.

VII. FURTHER STUDY

Herein, we weighted every feature equally when computing the similarity score; however, this might not match with user's attributes priority. There could be user specific requirements when searching for real estate. For example, users with a car may pay more attention to car parking than users without a car. Hence, making this approach more personalized by incorporating different weights for each feature should improve the recommendation performance.

REFERENCES

[1] G. Adomavicius and A. Tuzhilin, "Context-aware recommender systems," in *Recommender Systems Handbook*. Boston, MA, USA: Springer, 2011, pp. 217–253.

[2] R. Ariyawansa, "An empirical study of consumer behavior in housing market in colombo," *Built-Environ. Sri Lanka*, vol. 8, no. 1, p. 11, May 2010.

[3] T. Badriyah, S. Azvy, W. Yuwono, and I. Syarif, "Recommendation system for property search using content based filtering method," in *Proc. Int. Conf. Inf. Commun. Technol. (ICOIACT)*, Mar. 2018, pp. 25–29.

[4] C. Bakir, "Collaborative filtering with temporal dynamics with using singular value decomposition," *Tehnički Vjesnik*, vol. 25, no. 1, pp. 130–135, 2018.

[5] A. Beutel, P. Covington, S. Jain, C. Xu, J. Li, V. Gatto, and H. Ed Chi, "Latent cross: Making use of context in recurrent recommender systems," in *Proc. 11th ACM Int. Conf. Web Search Data Mining*, 2018, pp. 46–54.

[6] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, Jan. 2003.

[7] J. Chia, A. Harun, A. W. M. Kassim, D. Martin, and N. Kepal, "Understanding factors that influence house purchase intention among consumers in kota kinabalu: An application of buyer behavior model theory," *J. Technol. Manage. Bus.*, vol. 3, no. 2, Dec. 2016.

[8] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," 2014, *arXiv:1409.1259*. [Online]. Available: http://arxiv.org/abs/1409.1259

[9] Y. Deldjoo, M. F. Dacrema, and M. G. Constantin, "Movie genome: Alleviating new item cold start in movie recommendation," *User Model. User-Adapted Interact.*, vol. 29, no. 2, pp. 291–343, Apr. 2019.

[10] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," *CoRR*, abs/1708.05031, pp. 1–5, Oct. 2017.

[11] B. Hidasi, M. Quadrana, A. Karatzoglou, and D. Tikk, "Parallel recurrent neural network architectures for feature-rich session-based recommendations," in *Proc. 10th ACM Conf. Recommender Syst.*, Sep. 2016, pp. 241–248.

[12] B. Hidasi *et al.*, "Session-based recommendations with recurrent neural networks," 2015, *arXiv:1511.06939*. [Online]. Available: https://arxiv.org/abs/1511.06939

[13] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[14] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *Proc. 8th IEEE Int. Conf. Data Mining*, Dec. 2008, pp. 263–272.

[15] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980

[16] J. Knoll, R. Groß, A. Schwanke, B. Rinn, and M. Schreyer, "Applying recommender approaches to the real estate E-commerce market," in *Innovations for Community Services* M. Hodoň, G. Eichler, C. Erfurth, and G. Fahrnberger, Eds. Cham, Switzerland: Springer, 2018, pp. 111–126.

[17] J. Li, P. Ren, Z. Chen, Z. Ren, T. Lian, and J. Ma, "Neural attentive session-based recommendation," in *Proc. ACM Conf. Inf. Knowl. Manage.*, Nov. 2017, p. 1419.

[18] Q. Liu, Y. Zeng, R. Mokhosi, and H. Zhang, "Stamp: Short-term attention/memory priority model for session-based recommendation," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 1831–1839.

[19] P. Lops, M. D. Gemmis, and G. Semeraro, "Content-based recommender systems: State of the art and trends," in *Recommender System Handbook*. Boston, MA, USA: Springer, 2011, pp. 73–105.

[20] T. Luostarinen and O. Kohonen, "Using topic models in content-based news recommender systems," in *Proc. 19th Nordic Conf. Comput. Linguistics*, 2013, pp. 239–251.

[21] J. S. Mang, R. Zainal, and I. S. M. Radzuan, "Influence of location on home buyers' purchase decision," in *AIP Conf. Proc.*, vol. 2016, no. 1, 2018, Art. no. 020078.

[22] F. Pan, S. Li, X. Ao, P. Tang, and Q. He, "Warm up cold-start advertisements: Improving CTR predictions via learning to learn ID embeddings," *CoRR*, vol. abs/1904.11547, pp. 1–7, Jul. 2019.

[23] R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. Lukose, M. Scholz, and Q. Yang, "One-class collaborative filtering," in *Proc. 8th IEEE Int. Conf. Data Mining*, Dec. 2008, pp. 502–511.

[24] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl, "Item-based collaborative filtering recommendation algorithms," in *Proc. 10th Int. Conf. World Wide Web*, 2001, pp. 285–295.

[25] H. Setiadi, R. Saptono, R. Anggrainingsih, and R. Andriani, "Recommendation feature of scientific articles on open journal system using content-based filtering," in *Proc. IEEE 6th Int. Conf. Eng. Technol. Appl. Sci. (ICETAS)*, Dec. 2019, pp. 1–6.

[26] M. Silva and N. Fraser, "Analysis of impact of property attributes on buyer behaviour in luxury condominium apartments market in colombo, Sri Lanka," in *Proc. 2nd Nat. Symp. Real Estate Manage. Valuation*, vol. 2. University of Sri Jayewardenepura, 2016, p. 152.

[27] Y. K. Tan, X. Xu, and Y. Liu, "Improved recurrent neural networks for session-based recommendations," in *Proc. 1st Workshop Deep Learn. Recommender Syst.* New York, NY, USA: Association for Computing Machinery, 2016, pp. 17–22.

[28] M. Vartak, A. Thiagarajan, C. Miranda, J. Bratman, and H. Larochelle, "A meta-learning perspective on cold-start recommendations for items," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6904–6914.

[29] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, and L. Bottou, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, no. 12, pp. 3371–3408, 2010.

[30] J. Wei, J. He, K. Chen, Y. Zhou, and Z. Tang, "Collaborative filtering and deep learning based recommendation system for cold start items," *Expert Syst. Appl.*, vol. 69, pp. 29–39, Mar. 2017.

[31] Y. Yu, C. Wang, L. Zhang, R. Gao, and H. Wang, "Geographical proximity boosted recommendation algorithms for real estate," in *Web Information Systems Engineering*, H. Hacid, W. Cellary, H. Wang, H.-Y. Paik, and R. Zhou, Eds. Cham, Switzerland: Springer, 2018, pp. 51–66.

[32] X. Yuan, J.-H. Lee, S.-J. Kim, and Y.-H. Kim, "Toward a user-oriented recommendation system for real estate websites," *Inf. Syst.*, vol. 38, no. 2, pp. 231–243, Apr. 2013.

[33] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep learning based recommender system: A survey and new perspectives," *ACM Comput. Surv.*, vol. 52, no. 1, pp. 1–38, Feb. 2019.

JIRUT POLOHAKUL received the B.Eng. degree in computer engineering from Chulalongkorn University, Bangkok, Thailand, in 2018, where he is currently pursuing the M.Eng. degree in computer engineering. His research interest includes recommendation systems.

EKAPOL CHUANGSUWANICH received the B.S. and S.M. degrees in electrical and computer engineering from Carnegie Mellon University, in 2008 and 2009, respectively, and the Ph.D. degree from MIT, in 2016. He then joined the MIT Computer Science and Artificial Intelligence Laboratory, Spoken Language Systems Group. He is currently a Faculty Member with the Department of Computer Engineering, Chulalongkorn University. His research interests include speech processing, assistive technology, and health applications.

ATIWONG SUCHATO received the B.Eng. degree in electrical engineering from Chulalongkorn University, in 1998. He was awarded the Ananda Mahidol Foundation scholarship to pursue advanced degrees with the Massachusetts Institute of Technology (MIT), Cambridge, MA, USA. He also received the S.M. and Ph.D. degrees in electrical engineering and computer science from MIT, in 2001 and 2004, respectively. He is currently an Associate Professor with the Department of Computer Engineering, Chulalongkorn University. His research interests include automatic speech recognition and their applications to the disabled. In recent years, his academic interests have covered more extensive topics in the application of computer engineering advancements to education systems as well as digital supports for lifelong learning, including learning management systems and MOOCs.

PROADPRAN PUNYABUKKANA has served as the Assistant Dean of Engineering for Chulalongkorn University, Bangkok, Thailand, from 2004 to 2008, and an Associate Dean, from 2008 to 2013. She is currently an Associate Professor with Chulalongkorn University. She joined University, in 1993, prior to being awarded a Fulbright Scholarship to pursue her Ph.D. degree at Claremont Graduate University, in 2003. Her doctoral thesis discussed programming, specifically object-oriented language design and implementation. When she returned to Thailand, her research interests include speech technology, focusing on speech synthesis and speech recognition in Thai. She and a colleague formed the Spoken Language System Research Group, in 2004. Their applications have been used to assist people with disabilities. While exploring this situation in Thailand, they started the Assistive Technology Research Group, in 2006. To date, the two groups have graduated more than 100 undergraduate and graduate students. She awarded Fulbright Visiting Scholar grants as a Visiting Professor first at MIT and then at Stanford, in 2013 and 2014, respectively.