IEICE TRANS. INF. & SYST., VOL.E107–D, NO.5 MAY 2024

PAPER Special Section on Data Engineering and Information Management

# A Personalised Session-Based Recommender System with Sequential Updating Based on Aggregation of Item Embeddings

**Yuma NAGI**[†], *Nonmember and* **Kazushi OKAMOTO**[††a)], *Member*

**SUMMARY** The study proposes a personalised session-based recommender system that embeds items by using Word2Vec and sequentially updates the session and user embeddings with the hierarchicalization and aggregation of item embeddings. To process a recommendation request, the system constructs a real-time user embedding that considers users' general preferences and sequential behaviour to handle short-term changes in user preferences with a low computational cost. The system performance was experimentally evaluated in terms of the accuracy, diversity, and novelty of the ranking of recommended items and the training and prediction times of the system for three different datasets. The results of these evaluations were then compared with those of the five baseline systems. According to the evaluation experiment, the proposed system achieved a relatively high recommendation accuracy compared with baseline systems and the diversity and novelty scores of the proposed system did not fall below 90% for any dataset. Furthermore, the training times of the Word2Vec-based systems, including the proposed system, were shorter than those of FPMC and GRU4Rec. The evaluation results suggest that the proposed recommender system succeeds in keeping the computational cost for training low while maintaining high-level recommendation accuracy, diversity, and novelty.

***key words:*** *information recommendation, collaborative filtering, session data, distributed representation, k-nearest neighbor algorithm*

## 1. Introduction

Recommender systems are designed to support and augment decision-making for situations where a person needs to make a choice about a given option without sufficient personal experience [1]. Two basic approaches are used by recommender systems, including content-based filtering and collaborative filtering [2]. Although the effectiveness of these methods has been demonstrated by both industrial and academic communities, Wang et al. discussed some drawbacks in terms of their application to histories of interactions [3]. Session-based recommender systems (SBRSs) have been studied to consider histories of interactions in general recommender systems. Studies on SBRS have increased since 2014 [3], and various types of systems have been proposed, such as factorisation machine-based approaches [4], shallow neural network model-based approaches [5]–[9], and deep neural network model-based approaches [10]–[13].

SBRS covers two scenarios: one where user IDs are attached to interactions or sessions, and one where they are not [14]. The former is also known as personalised SBRS or session-aware recommender system, which use the user's past session information as input [15]. Sequential recommendation, closely related to the personalised SBRSs, is typically defined as the task of predicting the next item a user will browse or buy, based on a list of the user's previous interactions [16]–[21]. Such recommendation focuses and excels at predicting a user's long-term preferences; in contrast, the personalised SBRSs are particularly adept at predicting a user's short-term preferences. In the personalised SBRSs, information of both users and items are generally used in the recommendation model. However, in terms of sequential recommendation, Grbovic et al. found that the recommendation model incorporating user and item information showed a faster decline in prediction (recommendation) accuracy over time than the models based solely on item information [22]. This issue is caused by short-term changes in user preferences. To deal with the issue requires the frequent updating of the recommendation model and consideration of appropriate updating intervals.

Our motivation lies in reducing the computational costs associated with updating the recommendation model in personalised SBRSs, aiming to address the user's short-term preference shift problem through a simplified approach. The key idea is online learning which generates user embeddings for recommendation from item embeddings. In this study, we propose a personalised SBRS that embeds items using Word2Vec and sequentially updates the session and user embeddings with item embeddings for each browsed item. We assume that a session is a set of ordered items and model user interactions as a set of ordered sessions for each user; therefore, there is a hierarchical relationship among the items, sessions, and users embeddings under this assumption. This relationship is defined by aggregation operations. As a similar model, Wang et al. proposed the hierarchical representation model for next basket recommendation [23], but the model requires the reconstruction of user embeddings to handle short-term changes in user preferences. For a recommendation request, the proposed system constructs an embedding of a user, called a real-time user embedding based on the similarity between session and user embeddings, and this embedding is used to search for recommended items. This usage is expected to improve recommendation accuracy with low computational cost.

We performed a recommendation experiment in which we predicted the next item given partial-session data using five baseline systems and three benchmark datasets to eval-

Manuscript received June 24, 2023.
Manuscript revised November 14, 2023.
Manuscript publicized January 9, 2024.

[†]The author is with UNICORN Inc., Tokyo, 160–0023 Japan.
[††]The author is with Department of Informatics, Graduate School of Informatics and Engineering, The University of Electro-Communications, Chofu-shi, 182–8585 Japan.
a) E-mail: kazushi@uec.ac.jp

DOI: 10.1587/transinf.2023DAP0006

uate the effectiveness of the proposed SBRS in terms of accuracy, diversity, novelty, and computational cost. In the experiment, hit ration (HR) and mean reciprocal rank (MRR) were measured as accuracy metrics for the generated ranking of the recommended items. In addition, two measures proposed by Hu et al. [24] and Wang et al. [12], diversity and novelty metrics, respectively, were applied for the ranking. The computational times were measured for model training and prediction of a next-item. Finally, we discuss the baseline and proposed systems based on experimental results.

This paper is organized as follows. Section 2 reviews related work. Section 3 describes the details of the proposed SBRS. Section 4 explains the conditions for the recommendation experiment. Section 5 compares and discusses the experimental results in terms of accuracy, diversity, novelty, and computational cost.

## 2. Literature Review

### 2.1 Tasks of SBRSs

SBRSs (including personalised SBRSs) use session data as input. Wang et al. [3] categorised the session as (a) a set of items collected or consumed in an event or certain period (e.g., a receipt for purchasing items at a store) and (b) a sequence of actions or events in a certain period (e.g., a browsing history when shopping for items on an e-commerce site). In (a), intra-session relations are unordered, whereas inter-session relations are ordered. In (b), both intra- and inter-session relations are order relations. In this study, we consider the task of recommending the next item in a browsing sequence of different items; therefore, a "session" is assumed, as defined in (b). To distinguish this from (b), in this study, we refer to it a "basket" or "transaction" for (a). In SBRS studies, there are three main tasks:

- next interaction (item) recommendation: a task of predicting which item a user will browse next in the current session, given only partial session information.
- next partial-session recommendation: a task of predicting which items a user will browse to complete the current session.
- next session (basket) recommendation: a task of predicting which a user will purchase items in the next session (basket).

There is a difference between sequential recommendations and SBRSs in terms of the length of sequences they handle, with SBRSs typically dealing with shorter sequence lengths. For example, in the context of sequential recommendation scenarios [18], [20], [21], the range of the average sequence lengths in the datasets used was between 6 to 165, whereas in the case of SBRSs [19], it was between 2.5 to 5.5. Therefore, it is difficult to directly compare the experimental results of sequential recommendation with those of SBRSs. In addition, there is another difference: sequential recommendation is interested in a user's long-term preferences, whereas SBRS is interested in a user's short-term preferences.

### 2.2 Approaches of SBRSs

As introduced in Introduction, a user's short-term preference shift leads to a faster decline in recommendation accuracy over time. To address this issue, it is necessary to frequently update the recommendation model. Transformer-based SBRSs, in particular, have been assessed through a process of incremental training and evaluation [19]. In the study by Moreira et al. [19], it has been suggested that while Transformer-based SBRSs demonstrated high levels of recommendation accuracy, there are instances where $k$-nearest-neighbour ($k$-NN) approaches achieved comparable levels of recommendation accuracy depending on the datasets and evaluation metrics. Moreover, in the context of sequential recommendation, similar trends have been suggested in the study by Latifi et al. [21]. Therefore, in SBRSs, it can be considered that $k$-NN is one of the powerful approaches.

Training deep learning models often require substantial computational resources. In contrast, $k$-NN based approaches skip a training process, if they are set up to calculate similarities between sessions and items. This characteristic facilitates ease of implementation and operation. This study focuses the $k$-NN based approaches for personalised SBRSs.

### 2.3 $k$-NN Approaches for SBRSs

In the $k$-NN based approaches, a key challenge is to compute the similarities between sessions and items effectively. Up to now, several methods have been proposed, such as S-KNN [25], V-SKNN, S-SKNN, SF-SKNN [26], STAN [27], and VSTAN [28], but these methods do not treat users differently and are generally non-personalised. In general, personalised recommendations offer a more valuable experience tailored to individual users. One approach for personilised SBRSs based on $k$-NN is usage of embedding techniques.

In natural language processing, Word2Vec [29] and GloVe [30] are one of the typical word-embedding models with additive compositionality that provides a function for obtaining meaningful compound words by adding embedded words. Barkan and Koenigstein proposed Item2Vec [31], which is an application of Word2Vec for item purchase history. Previous studies have attempted to apply Item2Vec to recommender systems and have validated its effectiveness [32]–[34]. Moreover, several studies have focused on word-embedding models with additive compositionality, which have been validated in natural language processing tasks, and have applied them to SBRSs [8], [22], [36], [37]. In particular, in the task of predicting the next item to be purchased given the item a user is currently viewing, Behavior2Vec [8] outperformed Prod2Vec [22] and FPMC [4] in terms of recommendation accuracy.

One advantage of the application of Word2Vec and GloVe to user and item embeddings is their low computational costs to obtain embeddings compared with matrix factorisation and deep neural network approaches. Grbovic et al. [22] proposed user2vec, which simultaneously embeds

users and items, inspired by Paragraph2Vec [35], an applied model of Word2Vec. According to their results in recommendation of $k$-NN items for an input user in the feature space, the user information contributed to an improvement of recommendation accuracy, but the user information deteriorates prediction accuracy faster over time than the item information. Therefore, there is a the challenge that frequent updating of the user embedding are required to maintain prediction accuracy. Conversely, if user embeddings can be generated from item embeddings as online learning, the frequency with which the user embedding is updated may be expected to be reduced.

## 3. Proposed SBRS

In this study, we propose a personalised SBRS with a hierarchical aggregation of the embeddings of items, sessions, and users, based on Word2Vec which has a property of additive compositionality. The property is useful to obtain meaningful compound words by adding embedded words. The proposed system adapts Word2Vec by replacing words with items and, expecting similar effects on item embeddings, uses a linear combination of these embeddings.

Let $U$ and $V$ be sets of $m$ users and $n$ items registered in the system, respectively. The $t$-th observed session of user $u \in U$ in the order of their item browsing is defined as $S_u^{(t)} = \left\{v_1^{(t)}, v_2^{(t)}, \cdots, v_i^{(t)}, \cdots\right\}$, $v_i^{(t)} \in V$. For the $(t+1)$-th session of the user $u$, the observation up to the $i$-th item is given as $S_u^{(t+1)} = \left\{v_1^{(t+1)}, v_2^{(t+1)}, \cdots, v_i^{(t+1)}\right\}$. In this study, we address the prediction task for the next browsing item $v_{i+1}^{(t+1)}$. A recommender system generates a ranking of $k$ recommended items $R_u^{(t+1)} \subset V$, $\left|R_u^{(t+1)}\right| = k$, to predict $v_{i+1}^{(t+1)}$.

### 3.1 Construction and Update of Session, User, and Real-Time User Embeddings

Let the dimensionality of item embeddings be $d$. The proposed system applies Word2Vec as an item-embedding function $f_{\boldsymbol{\theta}} : V \longrightarrow \mathbb{R}^d$, and its parameter $\boldsymbol{\theta}$ is learned by using the observed session data for all users $S = \left\{S_{u_1}, S_{u_2}, \cdots, S_{u_m}\right\}$, $S_u = \left\{S_u^{(1)}, S_u^{(2)}, \cdots, S_u^{(t)}\right\}$. Using learned item embeddings, the proposed system constructs and updates three embeddings, including session, user, and real-time user embeddings. The details are described as follows. In addition, Fig. 1 shows the flow of the construction and update processes of the session and user embeddings.

#### 3.1.1 Session Embedding

Session embedding is a distributed representation of $S_u^{(t)}$ and its role corresponds to sequential behaviour. The $t$-th session embedding $\boldsymbol{s}_u^{(t)} \in \mathbb{R}^d$ of user $u$ is initially set to

$$\boldsymbol{s}_u^{(t)} = f_{\boldsymbol{\theta}}\left(v_1^{(t)}\right) \tag{1}$$

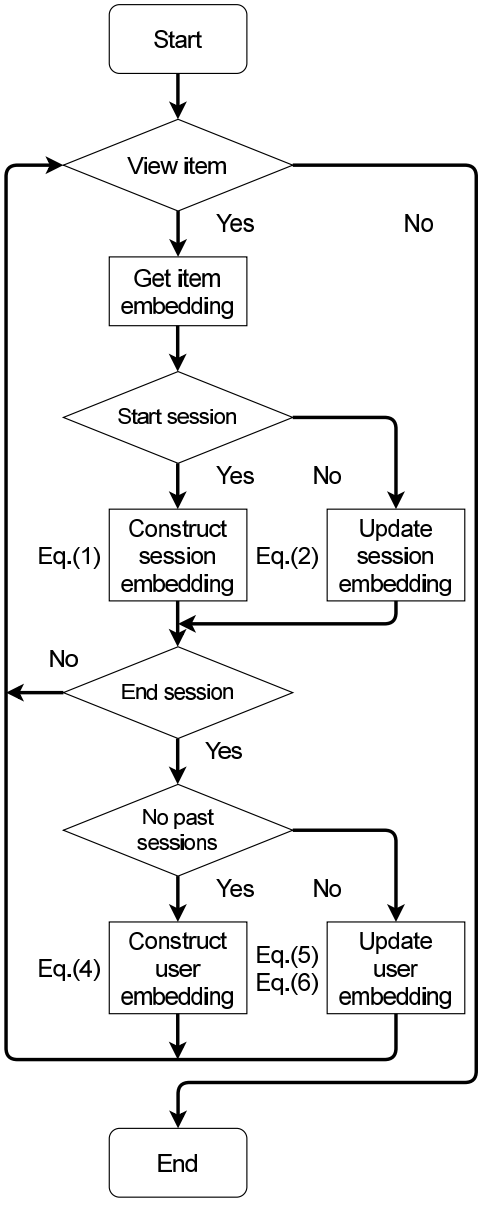



**Fig. 1** Proposed system: construction and update flow of each embedding

and is updated with

$$\boldsymbol{s}_u^{(t)} = \alpha_1 \boldsymbol{s}_u^{(t)} + \alpha_2 f_{\boldsymbol{\theta}}\left(v_i^{(t)}\right) \tag{2}$$

for each observation item during the session, where $\alpha_1, \alpha_2 \in [0, 1]$ is the importance. This update, a linear combination of embeddings of the current session and the observed item, is considered the ordering relationship because the session embedding differs depending on the order in which items are browsed.

Each browsing item is updated; thus, the computational cost of the importance calculation should be as low as possible. In this study, we propose the following two methods for calculating importance.

- **Similarity-based method** considers sequential behaviour with the similarity between an item browsed

and the current session. The method determines $\alpha_1, \alpha_2$, such as $\alpha_1 = \alpha,\ \alpha_2 = 1 - \alpha$, using a weight $\alpha \in [0, 1]$,

$$\alpha = \left(\cos\left(f_{\boldsymbol{\theta}}\left(v_i^{(t)}\right), \boldsymbol{s}_u^{(t)}\right) + 1\right) \div 2, \tag{3}$$

where $\cos(\boldsymbol{x}, \boldsymbol{y})$ is the cosine similarity for two vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$. In the applications of Word2Vec [29], cosine similarity is widely used to calculate similarities, and we also use it for similarity calculation.

- **Order-difference decay (exponential decay) method**, inspired by the method of session-based wide-in-wide-out for session context embedding proposed by Hu et al. [24], considers an approach to reduce the importance of items browsed more in the past. The method determines $\alpha_1, \alpha_2$ as $\alpha_1 = \exp(-\lambda),\ \alpha_2 = 1$ with the decay constant $\lambda$, a hyperparameter of the exponential decay.

#### 3.1.2 User Embedding

User embedding is a distributed representation updated by the context $C_u^{(t)} = \left\{S_u^{(t-c+1)}, \cdots, S_u^{(t-1)}, S_u^{(t)}\right\}$, at the end of the session $S_u^{(t)}$, where $C_u^{(t)}$ is the set of past $c$ sessions including the latest session. The role of user embedding is to consider the general preferences. We assume that general preferences can be estimated from past sessions that are strongly related to current behaviour (the latest session).

The user embedding $\boldsymbol{z}_u \in \mathbb{R}^d$ of user $u$ is initially set to

$$\boldsymbol{z}_u = \boldsymbol{s}_u^{(1)}, \tag{4}$$

and is updated using one of the following two methods

- weighted average

$$\boldsymbol{z}_u = \left(\sum_{i=1}^{c} \beta^{(t-c+i)}\right)^{-1} \sum_{i=1}^{c} \left(\beta^{(t-c+i)} \boldsymbol{s}_u^{(t-c+i)}\right), \tag{5}$$

- weighted sum

$$\boldsymbol{z}_u = \sum_{i=1}^{c} \left(\beta^{(t-c+i)} \boldsymbol{s}_u^{(t-c+i)}\right), \tag{6}$$

at the end of the session, where $\beta^{(t-c+1)}, \cdots, \beta^{(t-1)}, \beta^{(t)} \in [0, 1]$ denotes the importance of the past $c$ sessions. The general preference is estimated by aggregating past sessions based on their importance. Because similar sessions are considered to contain common preferences, the proposed system determines the importance $\beta$ using cosine similarity. The importance values of the past $c$ sessions $\beta^{(t-c+1)}, \cdots, \beta^{(t-1)}, \beta^{(t)}$ are defined as

$$\beta^{(i)} = \left(\cos\left(\boldsymbol{s}_u^{(t)}, \boldsymbol{s}_u^{(i)}\right) + 1\right) \div 2,$$
$$i = t - c + 1, \cdots, t - 1, t, \tag{7}$$

and the importance of similar past sessions is set to be high.

#### 3.1.3 Real-Time User Embedding

A real-time user embedding is a distributed representation that combines the latest session embedding $\boldsymbol{s}_u^{(t)}$ and the user embedding $\boldsymbol{z}_u$, which reflect sequential behaviours and general preferences, respectively. This embedding is calculated when a recommendation request in session $S_u^{(t)}$ has arrived. In this study, we propose a low-computational-cost embedding method to generate rankings of recommended items for many users at any time. The proposed method determines the importance of each factor based on the cosine similarity between the latest session and user embeddings.

Let the weight $\gamma \in [0, 1]$ be $\gamma = \left(\cos\left(\boldsymbol{s}_u^{(t)}, \boldsymbol{z}_u\right) + 1\right) \div 2$. This method uses $\gamma$ to construct a real-time user embedding $\boldsymbol{r}_u \in \mathbb{R}^d$ for user $u$,

$$\boldsymbol{r}_u = \gamma_1 \boldsymbol{s}_u^{(t)} + \gamma_2 \boldsymbol{z}_u, \tag{8}$$

by using a linear combination of the latest session and user embeddings. The weight $\gamma$ is used to determine $\gamma_1,\ \gamma_2$ such that $\gamma_1 = \gamma,\ \gamma_2 = 1 - \gamma$.

### 3.2 Search for Recommended Items

The proposed system uses the $k$-nearest neighbour algorithm to search the vector space around the real-time user embedding $\boldsymbol{r}_u$ and generates a ranking of recommended items $R_u^{(t+1)} \subset V$. The $k$-nearest neighbour algorithm outputs an ordered subset (ranked list) of $V$ as in $NN_V : (\boldsymbol{r}_u, k) \mapsto \{v_1, v_2, \cdots, v_k\} \subset V$, $R_u^{(t+1)} = NN_V(\boldsymbol{r}_u, k)$, for a real-time user embedding $\boldsymbol{r}_u$, a set of items $V$, and number of neighbours $k$. The order is determined by the cosine similarity between $\boldsymbol{r}_u$ and $f_{\boldsymbol{\theta}}(v), \forall v \in \{v_1, v_2, \cdots, v_k\}$.

## 4. Experiment

In this study, we performed a recommendation experiment in which we predicted the next item using partial session data using five baseline systems and three session datasets to evaluate the effectiveness of the proposed SBRS in terms of accuracy, diversity, novelty, and computational cost.

### 4.1 Used Session Datasets

In this experiment, we used the three datasets recorded using real-world services. These datasets differ in terms of the total number of users, items, and item categories considered.

#### 4.1.1 Details of Used Datasets

The three datasets used in this experiment were as follows.

- **Trivago**[†] comprised user sessions over a one-week period on the hotel search site. The dataset contained ten types of behaviours, such as item browsing, rating, and image selection. In this experiment, we only used the logs of behaviours corresponding to item browsing (clicked items and interactions for item information and image).

[†]https://recsys.trivago.cloud/challenge/dataset/

**Table 1** Attributes of the session datasets used in the experiment

| Attribute | Format |
|---|---|
| User ID | A string that uniquely identifies the user |
| Session ID | A string that uniquely identifies the session |
| Item ID | A string that uniquely identifies the item |
| Timestamp | The date and time of logging |

- **REES46**[†] contained behaviour data from a large multi-category online store. This dataset included seven months of log data. However, in this experiment, we only used three days (March 1, 2, and 3, 2020) of logs, about twice as much as the amount of Trivago's logs, since the data size was the limit of our computing resources. In addition, several types of behaviours such as viewing and adding items to the cart are recorded, but only view behaviour was used.
- **DIGINETICA**[††] consisted of user sessions extracted from an e-commerce site search engine logs over six months. The dataset differed from the other datasets because it included anonymous users without user IDs. In this experiment, logs of anonymous users were also used as inputs of the system.

These datasets consisted of logs with the four attributes listed in Table 1 and corresponded to the set of session data (each user would never have the logs for multiple sessions or items at the same time).

#### 4.1.2 Preprocessing for Datasets

Li et al. used DIGINETICA to evaluate SBRSs and preprocess it by removing items with fewer than five appearances and sessions with a length of one [11]. In this experiment, the same preprocessing was applied to all three datasets.

In this study, we considered recommender systems should not predict the same item that the user browsed previously (the previous item). Therefore, in this experiment, the item that appeared consecutively in a session were treated as a single item. Accordingly, each system was adjusted so that the ranking of the recommended items did not include the previous item. Multiple records of non-consecutive views of the same item within one session were retained because repeated recommendations are useful as reminders [15], [28].

Furthermore, if a session consisted of multiple user IDs, the user IDs of all logs within the session were replaced as the first user ID to appear in the session log. As a supplement, there are three possible approaches to treat such session: removing the session; replacing these user IDs to a specific user ID (our case); dividing the session by each user ID and assign new session IDs. Depending on the error process during data collection, we assumed that errors occurred during the assignment of session IDs and we applied the above preprocessing. In this study, such sessions constituted only 0.3% of the total, and we consider that this preprocessing does not affect the experimental results. Anonymous users of DIGINETICA were treated as those who had only one session and assigned dummy user IDs: each anonymous user was assigned a unique ID, resulting in the number of anonymous users being equal to the number of their sessions. Table 2 lists the statistics for each dataset after such preprocessing.

[†]https://www.kaggle.com/mkechinov/ecommerce-behavior-data-from-multi-category-store/

[††]https://competitions.codalab.org/competitions/11161/

**Table 2** Statistics of the session datasets used in the experiment

| Dataset | Log | Item | Session | User |
|---|---|---|---|---|
| Trivago | 1,627,804 | 179,484 | 337,816 | 295,165 |
| REES46 | 3,871,269 | 90,782 | 654,763 | 427,292 |
| DIGINETICA | 915,141 | 44,484 | 194,232 | *190,878 |

*sum of 136,434 anonymous and 54,444 non-anonymous users

**Table 3** Variations of the proposed system

| | Update method for session embedding | Word2Vec architecture |
|---|---|---|
| Proposed-SIM (SG) | Cosine similarity | SG |
| Proposed-SIM (CB) | Cosine similarity | CB |
| Proposed-ODD (SG) | Order difference decay | SG |
| Proposed-ODD (CB) | Order difference decay | CB |

### 4.2 Baseline and Proposed Systems

Five baseline systems were applied in this experiment, including POP, S-POP, FPMC, GRU4Rec, and Word2Vec+kNN (SG: Skip-gram / CB: Continuous Bag-of-words). The proposed system was evaluated for the four types listed in Table 3. For details of the algorithms of FPMC [4] and GRU4Rec [10], please refer to the original papers. Details of POP, S-POP, and Word2Vec+kNN are provided as follows.

#### 4.2.1 POP

POP is a global population-based system that recommends the most frequently observed items in a training set. Let $S' = \{S_1, S_2, \cdots, S_i, \cdots\} \subset S$ be a session set and $V_{\cup}^{S'} = \bigcup_{S_i \in S'} S_i \subset V$ be the union set of items in $S'$. For a session set $S'$ and ranking size $r$, the function $POP$ outputs a subset of $V_{\cup}^{S'}$ as $POP : (S', r) \mapsto \{v_1, v_2, \cdots, v_l\} \subset V_{\cup}^{S'}$ where $l = \min\left(r, \left|V_{\cup}^{S'}\right|\right)$ and $v_i \in \{v_1, v_2, \cdots, v_l\}$ is the $i$-th most frequently observed item in $S'$. POP generates popular item rankings from $S_U = \bigcup_{u \in U} S_u$, the session data of all users. The ranking is denoted by $R_u^{(t+1)} = POP(S_U, k)$. Note that if $POP(S_U, k)$ contains the previous item, then the $(k+1)$-th most frequently observed item is moved up to $R_u^{(t+1)}$.

#### 4.2.2 S-POP

S-POP is a local population-based system that recommends the most frequently observed items in a session $S_u^{(t+1)}$ to which the target item $v_{i+1}^{(t+1)}$ belongs. The system gener-

ates a popular item ranking in $S_u^{(t+1)}$ denoted by $R_u^{(t+1)} = POP\left(\left\{S_u^{(t+1)}\right\}, k\right)$. If the number of appearances of $v_a$ is equal to that of $v_b$ in $S_u^{(t+1)}$ and the number of appearances of $v_a$ is less than that of $v_b$ in $POP(S_U, k)$, then the system ranks $v_a, v_b$ in $POP\left(\left\{S_u^{(t+1)}\right\}, k\right)$ according to the order of $POP(S_U, k)$. That is, the ranking in $R_u^{(t+1)}$ is determined by the priorities of the intra-session relation and the overall ranks. Furthermore, if $\left|R_u^{(t+1)}\right| < k$ is satisfied, the most frequently observed item is obtained from $POP(S_U, k) \setminus POP\left(\left\{S_u^{(t+1)}\right\}, k\right)$, and is added at the end of the ranking until $\left|R_u^{(t+1)}\right| = k$ is satisfied.

#### 4.2.3 Word2Vec+kNN

Item2Vec, as proposed by Barkan et al., applies a skip-gram model with the negative sampling used in Word2Vec [29] to product purchase history [31]. The simplest approach to the SBRS using Item2Vec is to recommend the $k$-neighbour items closest to the last item browsed in the feature space. However, the Item2Vec approach cannot be directly applied to session data. Item2Vec considers the basket data and spatial and temporal information of an input sequence as not included in its objective function, which differs from Word2Vec. Thus, directly applying Item2Vec's objective function which ignores item order is not suitable because the target of this study is session data and a session is a time series. Hence, we used the original objective function of Word2Vec [29],

$$\frac{1}{K}\sum_{i=1}^{K}\sum_{-w\le j\le w, j\neq 0}\log p\left(v_{i+j} \mid v_i\right), \tag{9}$$

where $K$ denotes the session size and $w$ is the window size. In short, we applied Word2Vec+kNN, which consists of Word2Vec and the $k$-nearest neighbour algorithm as a baseline system using only item embeddings inspired by Item2Vec.

In the grid search for the hyperparameters in this experiment, the window size parameter $w$ equal to the maximum session size is a candidate value. This case corresponds to Item2Vec's assumption that any item pair in the same session is similar. In addition to the skip-gram model, the continuous bag-of-words model [38] was also evaluated.

### 4.3 Evaluation Metrics of Accuracy, Diversity, and Novelty

In this study, we used HR@$k$, MRR@$k$, DIV@$k$, and MCAN@$k$, which have been used in previous studies on SBRS [10], [12], [24] as evaluation metrics in terms of recommendation accuracy, diversity, and novelty.

Let $P \subset U \times V$ be a set of pairs $p = \left(u, v_{i+1}^{(t+1)}\right)$ for a target item $v_{i+1}^{(t+1)}$ in the $(t+1)$-th session $S_u^{(t+1)}$ for user $u \in \{u_1, u_2, \cdots\}$ in the testing set. In addition, the number of target items was defined as $N = |P|$. It should be noted that each system could fail to generate the ranking of the recommended items $R_u^{(t+1)}$ owing to the lack of required information. In such a case, the elements of $P$ partially differ among the systems because they adopt the policy of excluding the corresponding pairs $p$ from $P$.

- **HR@$k$**, an accuracy metric, measures the degree to which the target item is observed in $R_u^{(t+1)}$ $\left(\left|R_u^{(t+1)}\right| = k\right)$. HR@$k$ is defined as

$$\mathrm{HR@}k = \frac{1}{N}\sum_{(u,v)\in P}\left|\{v\} \cap R_u^{(t+1)}\right|. \tag{10}$$

- **MRR@$k$**, an accuracy metric, considers the rank of each recommended item. As a rank function, $rank : \left(R_u^{(t+1)}, v\right) \mapsto r \in [1, k]$, which returns the rank $r$ of item $v$ in $R_u^{(t+1)}$, MRR@$k$ is defined as

$$\mathrm{MRR@}k = \frac{1}{N}\sum_{(u,v)\in P}\frac{1}{rank\left(R_u^{(t+1)}, v\right)}. \tag{11}$$

  For $v \notin R_u^{(t+1)}$ with a given $p = (u, v)$, Eq. (11) is computed as $1/rank\left(R_u^{(t+1)}, v\right) = 0$. If $rank\left(R_u^{(t+1)}, v\right) = k$ for the target $p = (u, v)$, its HR@$k$ is 1 but its MRR@$k$ is $1/k$. For $v \in R_u^{(t+1)}$, MRR@$k$ penalizes the prediction with low confidence (e.g., $k$-th) over predictions with high confidence (e.g., 1 st).
- **DIV@$k$**, a diversity metric, is based on the non-overlapping rate of pairs of recommended items rankings for all targets $P$ as proposed by Hu et al. [24]. All recommended items rankings for $P$ are arranged and numbered $R = \{R_1, R_2, \cdots, R_N\}$. DIV@$k$ is then defined as

$$\mathrm{DIV@}k = \frac{2}{N(N-1)}\sum_{i\neq j}\left(1 - \frac{\left|R_i \cap R_j\right|}{\left|R_i \cup R_j\right|}\right). \tag{12}$$

  DIV@$k$ is not an evaluation of each target, but rather an overall evaluation of the diversity of the rankings of recommended items for all target outputs based on calculating the average non-overlapping rate for all of these combinations.
- **MCAN@$k$**, a novelty metric, is based on the non-overlapping rate of pairs between context items and the ranking of recommended items [12]. Let the context items $V_p \subset V$ for the target $p = (u, v)$ be $V_p = \bigcup_{S\in C_u^{(t+1)}} \{v \mid v \in S\}$. MCAN@$k$ is defined as

$$\mathrm{MCAN@}k = \frac{1}{N}\sum_{p\in P}\left(1 - \frac{\left|V_p \cap R_u^{(t+1)}\right|}{k}\right). \tag{13}$$

  MCAN@$k$ has a higher score, where the larger the size of the unknown items $R_u^{(t+1)} \setminus V_p$ for user $u$ of target

**Table 4** Average and maximum session size of the session datasets

| | Average Size | Maximum Size |
|---|---|---|
| Trivago | 4.82 | 930 |
| REES46 | 5.91 | 1,413 |
| DIGINETICA | 4.71 | 69 |

$p = (u, v)$, which indicates that the metric corresponds to a novelty metric. Some studies had approximately determined novelty based on the popularity of recommended items [28]. However, it is not generally true that unpopular items are novel items for each user. In contrast, MCAN@$k$ assumes that items with no contextual relevance are novel. This metric considers whether an item is unknown to the user based on each user's observation. Therefore, we consider the novelty of the item to be highly plausible.

### 4.4 Hyperparameter Tuning

The input of FPMC is basket data, an unordered set, but session data comprising an ordered set were used in this study. Therefore, each session was divided into several consecutive items until they could be regarded as a basket (granularity that allows for negligible ordering relationships) to adapt the data format of this study to that assumed by FPMC. In this experiment, we uniquely defined split size as a hyperparameter to determine the number of consecutive items. For example, if the split size was 2, a session consisting of four items is considered as two baskets.

The candidates for the window size of Word2Vec+kNN vary depending on the dataset used. Based on the average and maximum session sizes for each dataset listed in Table 4, four window sizes were used as candidates, including the minimum window size of two, $[a] \pm 1$, with an average session size $a$, and a maximum session size. The final case is equivalent to using the objective function of Item2Vec [31]. The optimal hyperparameters of Word2Vec+kNN were directly applied to the hyperparameters related to Word2Vec in the proposed system.

The hyperparameters of the baseline and proposed systems are listed in Table 5.

**Table 5** Hyperparameters of the baseline and proposed systems

| Hyperparameters | Candidates |
|---|---|
| FPMC | |
| - Session split size | 1, 2 |
| - Number of factors | 25, 50, 75, 100 |
| - Number of epochs | $1, 2, \cdots, 30$ |
| GRU4Rec | |
| - (Batch size, Number of hidden units) | $(32, 100), (64, 50),$ $(128, 25), (200, 16)$ |
| - Learning rate | 0.01, 0.001 |
| - Number of epochs | $1, 2, \cdots, 30$ |
| Word2Vec+kNN (SG / CB) | |
| - Window size | |
| - Trivago | 2, 4, 6, 930 |
| - REES46 | 2, 5, 7, 1413 |
| - DIGINETICA | 2, 4, 6, 69 |
| - Vector dimensionality | 25, 50, 75, 100 |
| - Subsampling threshold | 0.01, 0.001, 0.0001 |
| - Number of epochs | $1, 2, \cdots, 30$ |
| Proposed system (SG / CB) | |
| - $(\alpha_1, \alpha_2)$ in Eq (2) | |
| - Cosine similarity | $(\alpha, 1-\alpha), (1-\alpha, \alpha),$ $(\alpha, 1), (1, \alpha)$ |
| - Order difference decay | $(\exp(-\lambda), 1),$ $\lambda : 0.6, 0.7, 0.8, 0.9$ |
| - Updating method in Eq (5), (6) | weighted average, weighted sum |
| - Context size | 3, 5 |
| - $(\gamma_1, \gamma_2)$ in Eq (8) | $(\gamma, 1-\gamma), (1-\gamma, \gamma),$ $(\gamma, 1), (1, \gamma)$ |

### 4.5 Experimental Procedure

For each dataset and system combination, the experimental procedure was as follows.

1. We divided the entire dataset into training, validation, and testing sets based on timestamps such that the validation and testing sets contained about 10,000 sessions. In this experiment, we created five different training, validation, and testing sets by shifting the split times.
2. We set the target items as the last items of each session included in the validation and testing sets.
3. We performed hyperparameter tuning of each system by using the training and validation sets. MRR@$k$ was used as a metric to determine the optimal hyperparameters. In this process, hyperparameters are determined independently in each fold, which means different hyperparameters may be selected for each fold. We evaluated the ranking of recommended items generated for each target item in the validation set.
4. In the optimal hyperparameters determined in Step 3, we calculated the four evaluation metrics using the new training sets, which are concatenated from the training and validation sets, and the testing sets. We then evaluated the ranking of recommended items generated for each target item in the testing set.
5. The evaluation in Step 4 was performed for five different training and testing sets; therefore, five evaluation scores were obtained. The final evaluation value was the average of these five scores.

Regarding Step 1, several previous studies have divided a dataset into session series for each user and utilized the last session as the testing set, the second to the last session as the validation set, and the remaining sessions as the training set. However, this splitting method may cause data leakage because future information may be used for predictions in the validation and testing sets unless the time axes of all the users' session series are coincidental.

Note that because POP and S-POP do not have hyperparameters, for these cases, each dataset was split into training and testing sets in Step 1, and the hyperparameter tuning in Step.3 was skipped. The process flow of the experiment is illustrated in Fig. 2.

**Table 6** Recommendation accuracy of the baseline and proposed systems

| | HR@20 [%] | | | MRR@20 [%] | | |
|---|---|---|---|---|---|---|
| | Trivago | REES46 | DIGINETICA | Trivago | REES46 | DIGINETICA |
| POP | 0.625 | 7.90 | 0.871 | 0.139 | 1.87 | 0.180 |
| S-POP | 17.2 | 22.4 | 13.8 | **12.2** | 12.8 | 9.22 |
| FPMC | **38.3** | 3.65 | 16.6 | 11.3 | 1.09 | 5.66 |
| GRU4Rec | 31.9 | 37.2 | **37.2** | 11.9 | **14.7** | **12.7** |
| Word2Vec+kNN (SG) | 23.1 | 34.2 | 28.6 | 6.10 | 11.8 | 7.76 |
| Word2Vec+kNN (CB) | 24.1 | 34.9 | 27.1 | 6.36 | 12.2 | 7.77 |
| Proposed-SIM (SG) | 26.9 | 32.8 | 28.1 | 11.6 | 13.6 | 10.3 |
| Proposed-SIM (CB) | 27.6 | 35.1 | 26.9 | 11.5 | 14.4 | 10.2 |
| Proposed-ODD (SG) | 25.0 | 37.6 | 31.9 | 7.35 | 13.7 | 9.31 |
| Proposed-ODD (CB) | 26.5 | **38.4** | 29.2 | 7.72 | 14.3 | 8.94 |

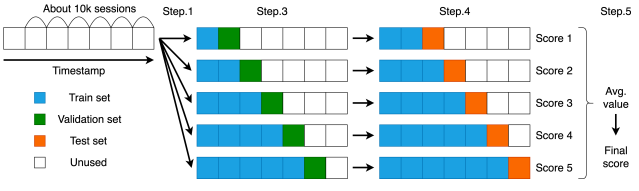



**Fig. 2** Experimental procedure for the evaluation experiment

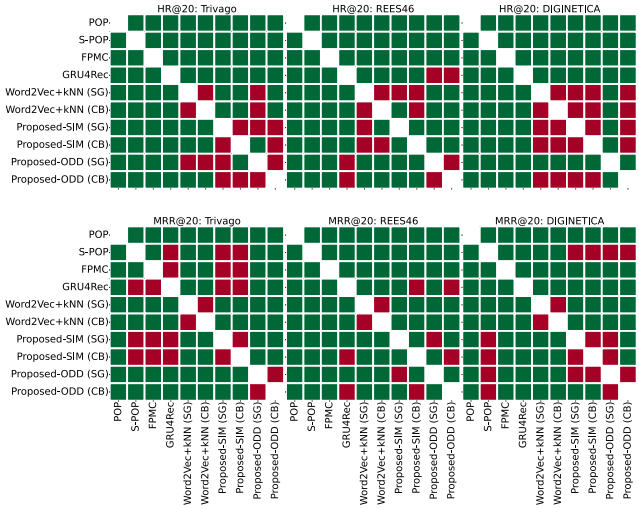



**Fig. 3** Tukey HSD test for HR@20 and MRR@20 (green: significant difference ($\alpha = 0.05$), red: no significant difference)

## 5. Results and Discussion

The evaluation results for each system using the four evaluation metrics corresponding to recommendation accuracy, diversity, and novelty are described and discussed. The results for the system's learning time and the recommendation generation time for one target (prediction time) are also evaluated.

### 5.1 Analysis of Accuracy

Table 6 presents the evaluation results of the recommendation accuracy using HR@20 and MRR@20 for the baseline and proposed systems, respectively. The best scores for each dataset are highlighted in bold and underlined. Furthermore, Fig. 3 presents the results of the Tukey HSD test for HR@20 and MRR@20. From Table 6, it may be observed that GRU4Rec was among the top three for all metrics and datasets. In addition, FPMC outperformed GRU4Rec by more than six points on HR@20 in Trivago but did not outperform on the other metrics and datasets. According to Fig. 3, for Word2Vec architecture, no significant differences were observed between Word2Vec+kNN (SG) and Word2Vec+kNN (CB) for all datasets and metrics.

As shown in Table 6, ignoring the differences in type, the proposed system, an extended approach of item embedding by Word2Vec, outperformed Word2Vec+kNN, especially MRR@20. For example, Proposed-SIM (SG) outperformed Word2Vec+kNN (SG) by 5.5 points for Trivago, 1.8 points for REES46, 3.5 points for DIGINETICA on MRR@20 with significant differences. Other types of the proposed system consistently outperformed Word2Vec+kNN on MRR@20 with significant differences. In contrast, for HR@20, there were some cases of improvements in the scores with the significant differences between the proposed system and Word2Vec+kNN. Proposed-ODD (CB) significantly outperformed Word2Vec+kNN in REES46 and Trivago for HR@20.

When comparing the types of systems proposed, it is often difficult to see significant differences in the HR@20 scores; however, a noticeable trend, such as a difference of over 4 points in the MRR@20 score between Proposed-ODD (SG) and Proposed-SIM (SG) for Trivago, was observed. The experimental results indicate that the selection of the appropriate (SIM/ODD, SG/CB) pairs in variants of the proposed system, in terms of recommendation accuracy, depends on the dataset and metrics used. Hence, such identification should be carried out using methods such as cross-validation.

Furthermore, it may be seen that there were some cases where the proposed system achieved the same level of HR@20 and MRR@20 as GRU4Rec (ex. Proposed-ODD (CB) for REES46) if the appropriate type could be determined. Therefore, we expect that the proposed system would perform close to GRU4Rec by determining the session update method and Word2Vec architecture. Note that, Pereira et al. used the REES46 dataset in their study [19], Transformers4Rec, and measured the performance of GRU4Rec, while the data periods differed. According to the findings,

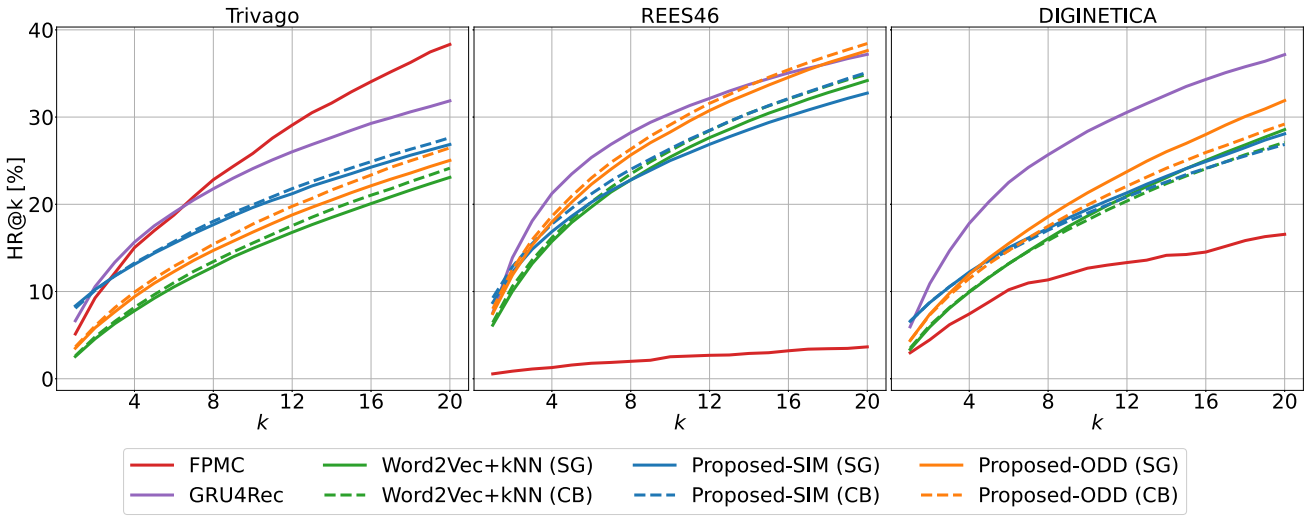



**Fig. 4** Transition of HR@$k$ for $k \in \{1, 2, \cdots, 20\}$

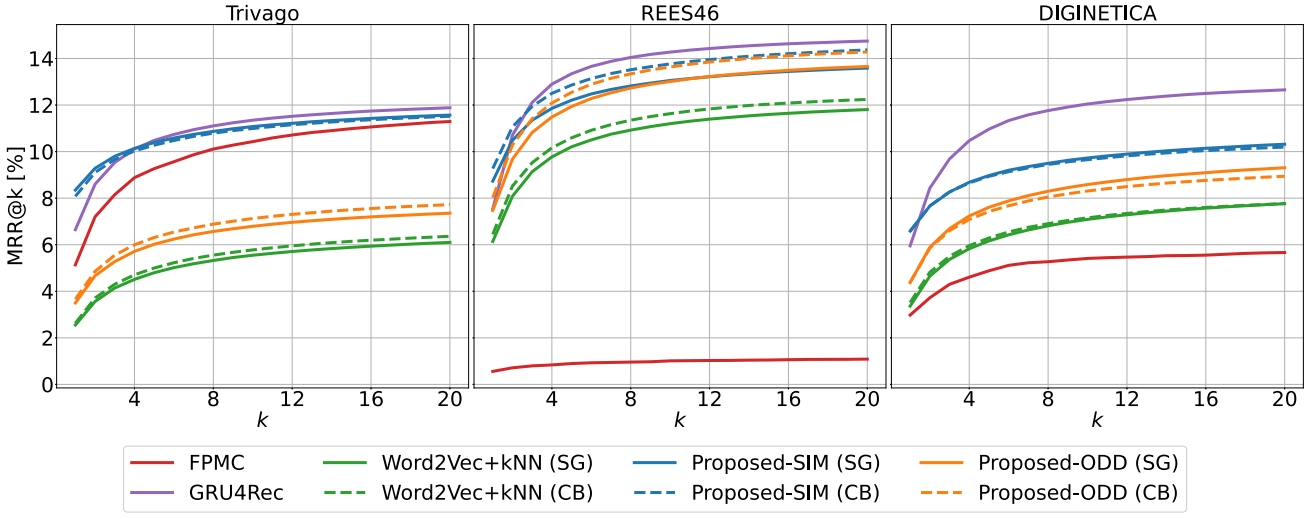



**Fig. 5** Transition of MRR@$k$ for $k \in \{1, 2, \cdots, 20\}$

GRU4Rec reported HR@20 scores ranging from 43.15 to 44.14%, and other $k$NN-based SBRSs used in their study achieved higher HR@20 scores compared to GRU4Rec. This trend is similar to that observed in our study, which implies that our experimental results are reasonable.

Figures 4 and 5 show the score transitions of HR@$k$ and MRR@$k$ for $k \in \{1, 2, \cdots, 20\}$, respectively. The horizontal and vertical axes represent the ranking sizes of the recommended items $k$, and the scores of HR@$k$ and MRR@$k$, respectively. As shown in Fig. 5, similar to the results in Table 6, the proposed system outperformed Word2Vec+kNN for all $k$. According to these results, the proposed aggregation approach of item embeddings contributed to improving the recommendation accuracy of Word2Vec-based SBRSs.

### 5.2 Analysis of Diversity and Novelty

The results for DIV@20 and MCAN@20, which are the metrics of diversity and novelty, are presented in Table 7. The best scores for each dataset are highlighted in bold and underlined. Furthermore, Fig. 6 presents the results of the Tukey HSD test for DIV@20 and MCAN@20. From Table 7, for all datasets, GRU4Rec, Word2Vec+kNN (SG / CB), and Proposed-SIM/ODD (SG / CB) achieved diversity scores greater than 99%. For example, DIV@20 of Proposed-SIM (SG) in Trivago was 99.97%, and for the recommended items ranking, it was equivalent to finding a common item by repeating 167 attempts to find common items in another ranking. Thus, there were very few cases in which the same item was recommended repeatedly. DIV@20 scores of Proposed-SIM/ODD (SG / CB) were not significantly different from those of Word2Vec+kNN (SG / CB) but these scores were high-level; therefore, the proposed system exhibited improved recommendation accuracy while maintaining diversity.

POP consistently achieved almost the best novelty

**Table 7** Diversity and novelty of the baseline and proposed systems

| | DIV@20 [%] | | | MCAN@20 [%] | | |
|---|---|---|---|---|---|---|
| | Trivago | REES46 | DIGINETICA | Trivago | REES46 | DIGINETICA |
| POP | 0.1101 | 1.561 | 0.1949 | **99.92** | 97.68 | **99.87** |
| S-POP | 23.18 | 39.61 | 28.95 | 90.91 | 80.37 | 88.09 |
| FPMC | 99.77 | 61.06 | 99.63 | 84.16 | **97.77** | 96.63 |
| GRU4Rec | 99.83 | 99.20 | 99.81 | 94.62 | 91.71 | 93.57 |
| Word2Vec+kNN (SG) | **99.97** | **99.66** | **99.91** | 97.98 | 94.82 | 97.69 |
| Word2Vec+kNN (CB) | 99.96 | 99.55 | **99.91** | 97.80 | 95.29 | 97.87 |
| Proposed-SIM (SG) | **99.97** | 99.58 | 99.89 | 95.92 | 89.98 | 94.66 |
| Proposed-SIM (CB) | 99.96 | 99.29 | **99.91** | 95.81 | 89.67 | 95.25 |
| Proposed-ODD (SG) | **99.97** | 99.65 | 99.89 | 96.53 | 91.70 | 95.55 |
| Proposed-ODD (CB) | 99.96 | 99.40 | **99.91** | 96.60 | 91.01 | 96.25 |

**Table 8** Runtime of the baseline and proposed systems

| | Training [s] | | | Prediction [ms] | | |
|---|---|---|---|---|---|---|
| | Trivago | REES46 | DIGINETICA | Trivago | REES46 | DIGINETICA |
| POP | 0.472 | 0.971 | 0.214 | 0.0111 | 0.0190 | 0.00954 |
| S-POP | 0.426 | 0.983 | 0.212 | 6.10 | 8.20 | 5.93 |
| FPMC | 9,348 | 3,198 | 1,152 | 127 | 71.3 | 33.9 |
| GRU4Rec (w/ GPU) | 48,319 | 6,889 | 3,959 | 26.1 | 12.9 | 5.98 |
| Word2Vec+kNN (SG) | 188 | 916 | 95.2 | 23.8 | 11.4 | 5.47 |
| Word2Vec+kNN (CB) | 161 | 267 | 56.1 | 68.1 | 17.5 | 13.4 |
| Proposed-SIM (SG) | 239 | 1,045 | 119 | 87.2 | 46.0 | 20.3 |
| Proposed-SIM (CB) | 223 | 399 | 79.5 | 131 | 51.8 | 29.4 |
| Proposed-ODD (SG) | 224 | 995 | 112 | 85.1 | 45.9 | 20.3 |
| Proposed-ODD (CB) | 211 | 364 | 73.5 | 129 | 51.9 | 29.7 |

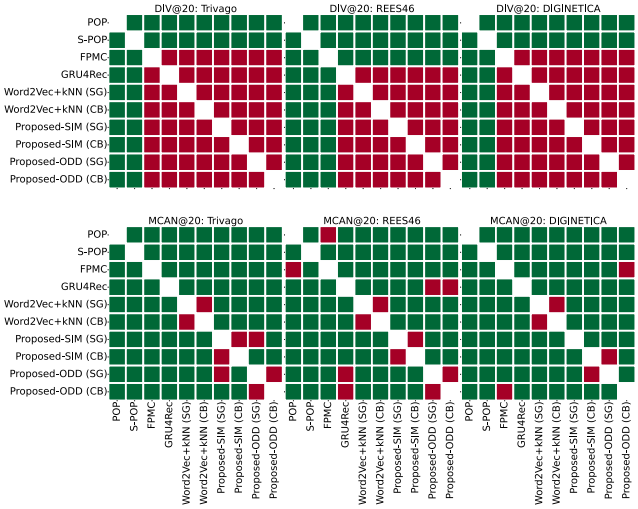



**Fig. 6** Tukey HSD test for DIV@20 and MCAN@20 (green: significant difference ($\alpha = 0.05$), red: no significant difference)

scores for all datasets but the worst accuracy scores. Word2Vec+kNN (SG / CB) achieved high-level scores within 2.9 points for POP, and Proposed-SIM/ODD (SG / CB) were inferior to Word2Vec+kNN (SG / CB) but high-level in the baseline systems on Trivago and DIGINETICA (there were significant differences). Additionally, MCAN@20 scores of Proposed-SIM/ODD (SG / CB) outperformed those of GRU4Rec in many cases with significant differences. According to these results, as discussed for DIV@20, the proposed system succeeded in improving the recommendation accuracy by extending Word2Vec+kNN (SG / CB), while maintaining novelty.

In contrast, among the four types of the proposed systems, the differences in the DIV@20 and MCAN@20 scores were within one and two points, respectively, while significant differences were found in many cases of the MCAN@20 scores. Therefore, the selection of the session update method and Word2Vec architecture did not affect diversity and novelty.

In summary, although the proposed system achieved a relatively higher recommendation accuracy among the baseline systems, its diversity and novelty scores did not fall below 90% for many cases. Therefore, the proposed system was considered to be less affected by the nature of the dataset. We conclude that the proposed system has achieved relatively high-level recommendation accuracy without reducing diversity and novelty.

### 5.3 System Runtime Evaluation

The measured training and prediction times for the baseline and proposed systems are listed in Table 8. The specifications of the machine used included a dual Intel Xeon Processor E5-2650v3 (2.30 GHz), 64 GB RAM, GeForce GTX 1080Ti, and Ubuntu 20.04.2, and the GPU was only used for the measurement of GRU4Rec.

In this experiment, FPMC required a maximum of 2.5 hours for training, and its prediction times were the worst of all systems, expect Proposed-ODD (CB). Word2Vec+kNN (SG / CB) and Proposed-SIM/ODD (SG / CB) required 262 to 1,076 of times longer training times than popularity-based systems. However, these times are 3 to 300 of times shorter than those of FPMC and GRU4Rec. The training

times of the proposed system were affected by the choice of Word2Vec architecture. The training times of Proposed-SIM/ODD (SG) were up to three times longer than those of Proposed-SIM/ODD (CB), depending on the dataset size. The prediction times for the four types of the proposed system, Proposed-SIM/ODD (SG / CB), were up to 5.5 times longer than those for Word2Vec+kNN (SG / CB), owing to the additional processing required to calculate real-time user embedding. In Table 8, GRU4Rec achieved shorter prediction times compared to the proposed system, and the prediction time is a weakness of the proposed system, since it generates real-time user embedding for each recommendation request.

These results indicate that the advantage of the proposed system lies in its shorter training time compared to FPMC and GRU4Rec, while maintaining high-level recommendation accuracy, diversity, and novelty. This shorter training time is advantageous for frequent updates of the recommendation model and embeddings, which is essential to overcome the problem of short-term changes in user preferences. In addition, such updates are necessary when new items or users are registered to the recommender system, and the short training time is also beneficial in this regard.

## 6. Conclusion

In this study, we proposed a personalised SBRS that embeds items using Word2Vec [29] and sequentially updates the session and user embeddings with a hierarchical aggregation of the item embeddings. The proposed system is designed to address short-term changes in user preferences by introducing real-time user embedding. Its advantages are that it can generate user embedding from item embedding without having to compute the user embedding from scratch, and it is a simple process to compute another embedding from the Word2Vec embedding.

According to the evaluation experiment, the proposed system achieved a relatively high recommendation accuracy compared with baseline systems. Furthermore, the diversity and novelty scores of the proposed system did not fall below 90% for many cases. Regarding the runtime of the system, the training times of the Word2Vec-based systems, including the proposed system, were shorter than those of FPMC [4] and GRU4Rec. The evaluation results suggest that the proposed system succeeded in keeping the computational cost for training low while maintaining high-level recommendation accuracy, diversity, and novelty.

Nevertheless, this study involves several limitations. The experimental results of this study were based on an offline evaluation using benchmark session datasets rather than an online evaluation of a real working system. Therefore, the robustness of the proposed system to short-term changes in user preferences should be evaluated in actual operation. In addition, the proposed system computes all session, user, and real-time user embeddings using item embeddings; thus, these embeddings exist in the same vector space. It is possible to measure the distances between instances in different hierarchies. However, in this study, we only evaluated the performance of an item search using real-time user embedding. The use of other embeddings and the validation of their effectiveness remain and application of approximate neighborhood search methods to improve prediction time as topics for future work.

## Acknowledgments

This work was supported by JSPS KAKENHI Grant Numbers JP18K18159, JP21H03553, JP22H03698.

### References

[1] P. Resnick and H.R. Varian, "Recommender systems," Commun. ACM. vol.40, no.3, pp.56–58, March 1997. DOI: 10.1145/245108.245121

[2] M.J. Pazzani, "A framework for collaborative, content-based and demographic filtering," Artif. Intell. Rev., vol.13, no.5–6, pp.393–408, Dec. 1999. DOI: 10.1023/A:1006544522159

[3] S. Wang, L. Cao, Y. Wang, Q.Z. Sheng, M.A. Orgun, and D. Lian, "A survey on session-based recommender systems," ACM Comput. Surv., vol.54, no.7, pp.1–38, July 2021. DOI: 10.1145/3465401

[4] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme, "Factorizing personalized markov chains for next-basket recommendation," Proc. 19th Int. Conf. World Wide Web, pp.811–820, April 2010. DOI: 10.1145/1772690.1772773

[5] S. Wan, Y. Lan, P. Wang, J. Guo, J. Xu, and X. Cheng, "Next basket recommendation with neural networks," Poster Proc. of the 9th ACM Conf. on Recommender Syst., Sept. 2015.

[6] M. Quadrana, A. Karatzoglou, B. Hidasi, and P. Cremonesi, "Personalizing session-based recommendations with hierarchical recurrent neural networks," Proc. 11th ACM Conf. Recommender Syst., pp.130–137, Aug. 2017. DOI: 10.1145/3109859.3109896

[7] Y. Li, W. Chen, and H. Yan, "Learning graph-based embedding for time-aware product recommendation," Proc. 2017 ACM Conf. Inf. and Knowledge Management, pp.2163–2166, Nov. 2017. DOI: 10.1145/3132847.3133060

[8] H.-H. Chen, "Behavior2Vec: Generating distributed representations of user's behaviors on products for recommender systems," ACM Trans. Knowledge Discovery from Data, vol.12, no.4, pp.1–20, April 2018. DOI: 10.1145/3184454

[9] Q. Liu, Y. Zeng, R. Mokhosi, and H. Zhang, "STAMP: Short-term attention/memory priority model for session-based recommendation," Proc. 24th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, pp.1831–1839, July 2018. DOI: 10.1145/3219819.3219950

[10] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based recommendations with recurrent neural networks," arXiv:1511.06939v4, March 2016. DOI: 10.48550/arXiv.1511.06939v4

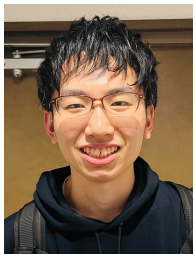
[11] J. Li, P. Ren, Z. Chen, Z. Ren, T. Lian, and J. Ma, "Neural attentive session-based recommendation," Proc. 2017 ACM Conf. Inf. and Knowledge Management, pp.1419–1428, Nov. 2017. DOI: 10.1145/3132847.3132926

[12] S. Wang, L. Hu, L. Cao, X. Huang, D. Lian, and W. Liu, "Attention-based transactional context embedding for next-item recommendation," Proc. 32nd AAAI Conf. Artif. Intell., vol.32, no.1, pp.2532–2539, April 2018. DOI: 10.1609/aaai.v32i1.11851

[13] B. Hidasi and A. Karatzoglou, "Recurrent neural networks with top-k gains for session-based recommendations," Proc. 27th ACM Int. Conf. Inf. and Knowledge Management, pp.843–852, Oct. 2018. DOI: 10.1145/3269206.3271761

[14] D. Jannach, M. Quadrana, and P. Cremonesi, "Session-based recommender systems," Recommender Systems Handbook, eds. F. Ricci, L. Rokach, and B. Shapira, Springer, New York, pp.301–334, 2012. DOI: 10.1007/978-1-0716-2197-4_8

[15] S. Latifi, N. Mauro, and D. Jannach, "Session-aware recommendation: A surprising quest for the state-of-the-art," Inf. Sci., vol.573, pp.291–315, Sept. 2021. DOI: 10.1016/j.ins.2021.05.048
[16] W.-C. Kang and J. McAuley, "Self-attentive sequential recommendation," Proc. 2018 IEEE Int. Conf. Data Mining, pp.197–206, Nov. 2018. DOI: 10.1109/ICDM.2018.00035
[17] H. Fang, D. Zhang, Y. Shu, and G. Guo, "Deep learning for sequential recommendation: Algorithms, influential factors, and evaluations," ACM Trans. Inf. Syst. vol.39, no.1, pp.1–42, Nov. 2020, DOI: 10.1145/3426723
[18] J. Wang, F. Yuan, J. Chen, Q. Wu, M. Yang, Y. Sun, and G. Zhang, "StackRec: Efficient training of very deep sequential recommender models by iterative stacking," Proc. 44th Int. ACM SIGIR Conf. Res. and Dev. in Inf. Retrieval, pp.357–366, July 2021. DOI: 10.1145/3404835.3462890
[19] G. de Souza Pereira Moreira, S. Rabhi, J.M. Lee, R. Ak, and E. Oldridge, "Transformers4Rec: Bridging the gap between NLP and sequential / session-based recommendation," Proc. 15th ACM Conf. Recommender Syst., pp.143–153, Sept. 2021. DOI: 10.1145/3460231.3474255
[20] A. Petrov and C. Macdonald, "Effective and efficient training for sequential recommendation using recency sampling," Proc. 16th ACM Conf. Recommender Syst., pp.81–91, Sept. 2022. DOI: 10.1145/3523227.3546785
[21] S. Latifi, D. Jannach, and A. Ferraro, "Sequential recommendation: A study on transformers, nearest neighbors and sampled metrics," Inf. Sci., vol.609, pp.660–678, Sept. 2022. DOI: 10.1016/j.ins.2022.07.079
[22] M. Grbovic, V. Radosavljevic, N. Djuric, N. Bhamidipati, J. Savla, V. Bhagwan, and D.Sharp, "E-commerce in your inbox: Product recommendations at scale," Proc. 21st ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, pp.1809–1818, Aug. 2015. DOI: 10.1145/2783258.2788627
[23] P. Wang, J. Guo, Y. Lan, J. Xu, S. Wan, and X. Cheng, "Learning hierarchical representation model for nextbasket recommendation," Proc. 38th Int. ACM SIGIR Conf. Res. and Dev. in Inf. Retrieval, pp.403–412, Aug. 2015. DOI: 10.1145/2766462.2767694
[24] L. Hu, L. Cao, S. Wang, G. Xu, J. Cao, and Z. Gu, "Diversifying personalized recommendation with user-session context," Proc. 26th Int. Joint Conf. Artif. Intell., pp.1858–1864, Aug. 2017. DOI: 10.24963/ijcai.2017/258
[25] G. Bonnin and D. Jannach, "Automated generation of music playlists: survey and experiments," ACM Comput. Surv., vol.47, no.2, pp.1–35, Nov. 2014. DOI: 10.1145/2652481
[26] M. Ludewig and D. Jannach, "Evaluation of session-based recommendation algorithms," User Modeling and User-Adapted Interaction, vol.28, no.1, pp.331–390, 2018. DOI: 10.1007/s11257-018-9209-6
[27] D. Garg, P. Gupta, P. Malhotra, L. Vig, and G. Shroff, "Sequence and time aware neighborhood for session-based recommendations: STAN," Proc. 42nd Int. ACM SIGIR Conf. Res. and Dev. in Inf. Retrieval, pp.1069–1072, July 2019. DOI: 10.1145/3331184.3331322
[28] M. Ludewig, N. Mauro, S. Latifi, and D. Jannach, "Empirical analysis of session-based recommendation algorithms," User Modeling and User-Adapted Interaction, vol.31, no.1, pp.149–181, Oct. 2020. DOI: 10.1007/s11257-020-09277-1.
[29] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," Adv. in Neural Inf. Process. Syst., vol.26, 2013.
[30] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," Proc. 2014 Conf. Empirical Methods in Neural Lang. Process., pp.1532–1543, Oct. 2014. DOI: 10.3115/v1/D14-1162
[31] O. Barkan and N. Koenigstein, "Item2Vec: Neural item embedding for collaborative filtering," Proc. 2016 IEEE 26th Int. Workshop on Machine Learning for Signal Process., pp.1–6, Sept. 2016. DOI: 10.1109/MLSP.2016.7738886
[32] V. Kuzmin, "Item2vec-based approach to a recommender system," Bachelor Thesis, University of Tartu, 2017.
[33] R. Goto, H. Fujinami, T. Yang, and M. Goto, "Collaborative filtering based on distributed expression considering differences in evaluation trends," Proc. of the Annual Conf. of the Japan Soc. for Artif. Intell., vol.JSAI2019, pp.2P1J204-2P1J204, June 2019. DOI: 10.11517/pjsai.JSAI2019.0_2P1J204
[34] Z. Yang, J. He, and S. He, "A collaborative filtering method based on forgetting theory and neural item embedding," Proc. 2019 IEEE 8th Joint Int. Inf. Technol. and Artif. Intell. Conf., pp.1606–1610, May 2019. DOI: 10.1109/ITAIC.2019.8785589
[35] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," Proc. of the 31st Int. Conf. on Machine Learning, vol.32, no.2, pp.1188–1196, June 2014.
[36] A. Greenstein-Messica, L. Rokach, and M. Friedman, "Session-based recommendations using item embedding," Proc. 22nd Int. Conf. Intell. User Interfaces, pp.629–633, March 2017. DOI: 10.1145/3025171.3025197
[37] F. Vasile, E. Smirnova, and A. Conneau, "Meta-prod2vec: Product embeddings using side-information for recommendation," Proc. 10th ACM Conf. Recommender Syst., pp.225–232, Sept. 2016. DOI: 10.1145/2959100.2959160
[38] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," arXiv:1301.3781v3, Sept. 2013. DOI: 10.48550/arXiv.1301.3781v3



**Yuma Nagi** received the B.E. and M.E. degree from The University of Electro-Communications, Japan, in 2021 and 2023, respectively. He is currently working at UNICORN Inc. His research interests include recommender system, data mining, and machine learning.

**Kazushi Okamoto** received the B.E. and M.E. degrees from Kochi University of Technology, Japan, in 2006 and 2008, respectively. He received the Dr.Eng. degree from Tokyo Institute of Technology, Japan, in 2011. From 2011 to 2015 and 2015 to 2020, he was an assistant professor in Chiba University, Japan and The University of Electro-Communications, Japan, respectively. He is currently an associate professor in The University of Electro-Communications, Japan. His research interests include machine learning, data mining, and recommender system. He is members of IEEE, IPSJ (Information Processing Society of Japan), JSAI (Japanese Society for Artificial Intelligence), and SOFT (Japan Society for Fuzzy Theory and Intelligent Informatics).