IAES International Journal of Artificial Intelligence (IJ-AI)
Vol. 15, No. 3, June 2026, pp. 2761~2773
ISSN: 2252-8938, DOI: 10.11591/ijai.v15.i3.pp2761-2773

Effects of sparse datasets on time interval-aware self-attention sequential recommendation models

Weishan Ooi, Lee Yeng Ong, Meng-Chew Leow
Faculty of Information Science and Technology, Multimedia University, Melaka, Malaysia

Article Info

Article history:

Received Dec 30, 2024
Revised Mar 19, 2026
Accepted Apr 22, 2026

Keywords:

SASRec
Sequential recommendation
Sparse dataset
Taobao
TiSASRec

ABSTRACT

Recommendation models serve as crucial filters in managing information, yet they face a few crucial challenges, such as capturing user-item interaction behaviors in sparse datasets. Data sparsity refers to an issue where there is a lack of interactions or missing values in the recommendation dataset. A sparse dataset with a massive number of missing values and interactions leads to more dynamic user behaviors, which suffers a poor recommendation quality. The self-attention mechanism from Transformer can alleviate the effects of data sparsity in datasets by assigning weights to items of interaction behaviors. This allows the model to capture the user dependencies in complex user behavior, which is beneficial for sparse datasets with patterns that are not immediately apparent. This approach has shown its capability to handle large and sparse datasets, as seen in time interval-aware self-attention sequential recommendation model (TiSASRec). It utilized the self-attention mechanism, considering the timestamp and absolute positions of items to estimate the higher attention weights to show the importance of recent items. Thus, this study aims to investigate the effects of sparse datasets by comparing the performance of TiSASRec model with self-attention based sequential recommendation model (SASRec), which excludes time interval-awareness.

Corresponding Author:

Lee Yeng Ong
Faculty of Information Science and Technology, Multimedia University
Ayer Keroh Lama Road, 75450 Melaka, Malaysia
Email: lyong@mmu.edu.my

1. INTRODUCTION

The product recommendation is considered a well-known technique for bonding customers and products by delivering insightful suggestions [1]. Modern recommendation models stand as one of the most ordinary applications in the big data domain [2] as it could perform predictions on users' preferences by retrieving various volumes of data for purchase history, click through rate (CTR), ratings, or online reviews [3]. Deep learning has been contributing to recommendation models as it has the capability of handling sequential data such as user's browsing or purchase history, which can capture the temporal dynamics and evolution of users' preferences and interests [4].

Although recent recommendation models are able to return remarkable recommendations, there exists a crucial challenge in capturing user-item interaction behaviors in sparse datasets. Data sparsity commonly happens in large-scale data, where many expected values, such as ratings or interactions, are missing or unavailable in the dataset. This is particularly problematic for collaborative filtering algorithms, which rely on these ratings to uncover beneficial relationships among users and items [5] and this is also further mentioned in the recent studies [6] where collaborative filtering algorithms assume all past interactions are equally important and this assumption often fails in real-world scenarios, as the user intent

can shift and short-term behaviors matter more than static long-term ratings. Furthermore, a sparse dataset with a massive number of missing values and interactions leads to more dynamic user behavior. Hence, the recommendation models cannot identify these relationships when datasets are sparse, resulting in lower accuracy and suboptimal recommendation lists [7].

Next, the collaborative recommendation model ought to capture the user behaviors from the user-item interactions to perform recommendations. In user behavior, it can be identified as long-term and short-term user behavior [8]. In long-term user behavior, it refers to a user's consistent interest over an extended period and reflects as the general preferences. For short-term behavior, it reflects the user's immediate or recent interests, which were influenced by temporal needs or specific circumstances at a given time. However, the user behaviors are dynamic, and it tend to change over time [9]. This eventually reduces the capability of collaborative recommendation models to capture the long-term dependencies from user-item interactions for modelling out the user behaviors.

Therefore, the sequential recommendation (SR) model is employed for capturing the long-term dependencies to provide insights for the user behaviors in user-item interactions. This often occurs in e-commerce context where the user behavior in user-item interactions usually happens successively in a sequence, rather than in an isolated manner [10]. SR is commonly referred to sequence-aware commendation where it receives sequentially ordered user interactions as input compared to a traditional user-item rating matrix. It also captures complex user preferences from users' chronological behavior and predicts the next most likely item for recommendation [11]. In SR, the items in the user-item interactions are considered correlated to each other. This is because the choice of the next item depends on the previous items, as it infers the users' selection based on their prior selections. The following item of a sequence is predicted based on the historical elements from the user-item interaction sequence [12]. In order to capture the user behavior from user-item interaction sequences, various SR approaches include recurrent neural networks (RNN), convolutional neural networks (CNN), self-attention, meta transitional learning (MetaTL), temporal graph transformer (TGT), and graph neural networks models have been introduced [12]–[18].

In SR, traditional models like RNNs have an issue of vanishing gradient which leads to lower performance and relatively longer training time because of its nature while capturing long-term dependencies. The employment of self-attention mechanism helps in modeling long-term dependencies by capturing connections between any pairs of items regardless of their distance in the user-item sequence. In addition, it also assigns attention weights to measure the importance of different items to highlight their relevancy and carry out recommendations. From the previous works [12]–[18], recommendations are mostly carried out by encoding users' past item interactions into a vector with a left-to-right sequential model based on this hidden representation. In some works, it employs an attention mechanism to prioritize influential user-item interactions by weighing different items. By referring to each user-item interaction sequence, the attention mechanism is able to capture the user behavior for the training model. Although some approaches include an attention mechanism, they are not time-interval aware. In SR, as the interacted items are arranged in a sequence, the time intervals between interactions are ignored, which is crucial for capturing the user preferences. SR models that lack consideration of time intervals have less insight about user behavior, leading to lower performance of recommendations, particularly in a sparse dataset.

In this study, the implementation of time interval-aware self-attention sequential recommendation model (TiSASRec) aims to capture the temporal dependencies in user-item interactions for datasets with different levels of sparsity. TiSASRec employs time interval-aware by considering the time intervals between each item that are not equally spaced in time. This indicates that the recommendation performance varies based on the values of time gaps. This differs from other self-attention-based SR models, such as SASRec, which excludes time interval-awareness. By utilizing the time intervals between each item, the self-attention mechanism in TiSASRec incorporates the absolute item positions. This allows TiSASRec to assign different attention weights based on items' absolute position and relative time intervals, effectively mapping out the importance of different items in varying timestamps and capturing the temporal user preferences in datasets. In this work, the performance comparison of TiSASRec and SASRec is investigated to showcase the effects of different parameter values of recommendation models on datasets with varying levels of sparsity, which is not explicitly discussed in the existing works. There are two main contributions in this study. Firstly, the application of TiSASRec and SASRec models across datasets with different levels of sparsity, which are MovieLens-1M and Taobao datasets. Secondly, a comparative analysis of the performance of different parameter values in both TiSASRec and SASRec models across two datasets.

The structure of this paper is as follows. Section 2 discusses the development of SR and the applications of contrastive learning, while section 3 explains the methodology used in this study. In section 4, the experimental results from both models are compared across two datasets with different levels of sparsity. Lastly, the effect of sparse datasets on time interval-aware self-attention SR models is concluded in section 5.

2. LITERATURE REVIEW

The issue of data sparsity in recommendation systems occurs when there is insufficient interaction data available to accurately determine user similarities and preferences. High sparsity of the dataset makes it difficult to predict user preferences due to the lack of reliable patterns from dynamic user behavior. Consequently, this results in poor performance of recommendation accuracy and efficiency. For instance, MovieLens-1M and Netflix have sparsity rates of 95.16% and 98.82%, respectively [19], while the e-commerce dataset namely Taobao has a sparsity rate of 99.85%, making it challenging to retrieve sequential patterns like long-term dependencies from the dataset [20]. Although increasing the scale of datasets and their sparsity level enhances data availability and provides more data points for identifying user preferences, sparse datasets also require more computational resources to perform recommendations.

SR performs recommendations based on sequential order of user interactions, have employed various approaches, including RNN, CNN, MetaTL, TGT, and graph neural networks. The GRU4Rec+ model applied with RNN [14] and the Caser model used CNN [15] achieved the NDCG@10 values of 0.5513 and 0.5538, respectively, on MovieLens-1M dataset, indicating their capabilities in ranking recommendation items. Both of them referred to an early attempt in SR studies and used it for comparison in the further study [21]. However, both models struggle to capture the core user preferences from long-term dependencies in sparse dataset, which proven that data sparsity can affect recommendation quality. The challenges of cold-start users with minimal interactions in SR [18] is experimented on three datasets from Amazon Electronics, Movie, and Book. The results show the HR@1 values of 0.224, 0.258, and 0.420 for each dataset, respectively. To improve overcome this challenge, SR model may utilize additional user or item information for cold-start users. Another model named sequential recommendation with graph neural networks (SURGE) [22] integrates different types of user preferences into clusters, forming dense interest graphs. This helps distinguish core interests from noisy behaviors. It achieved NDCG@2 values of 0.3625 and 0.9495 for the Taobao and Kuaishou datasets, respectively. However, its narrow focus on NDCG@2 to show the top two recommendations may not fully reflect its performance in ranking a broader set of relevant items, whereas NDCG@10 would provide a more comprehensive evaluation of the recommendation quality.

On the other hand, the self-attention mechanism in self-attention based sequential recommendation model (SASRec) can capture the entire user sequence without recurrent or convolutional operations and adaptively assign weights to previous items for predictions. The self-attention component takes user-item interaction sequences as input, with each item represented by an embedding. Positional encoding is added to capture the order of interactions. The self-attention component computes the relationship between all pairs of items in the sequences with the query (Q), key (K), and value (V) vectors for each item. These vectors are then used to calculate the attention scores, which are assigned to each item to weigh its contribution in the sequences for next item prediction. The SASRec has shown its capability to capture long-term dependencies with high efficiency and scalability on sparse datasets, achieving an NDCG@10 value of 0.5905 on the MovieLens-1M dataset [16]. The SASRec model is often referred as the baseline model in attention-based SR research. This is because it was one of the pioneer self-attention architectures for SR and it is also used for comparison in recent studies as well [23].

The TGT framework [17] which aims to address the challenge of modelling the time-evolving user preferences, particularly on multi-behavior interactions in SR systems. The TGT model achieved NDCG@10 values of 0.263 and 0.330 on Taobao-data and the IJCAI Contest dataset, respectively, and HR@10 values of 0.452 and 0.519 on the same datasets. The TGT model has demonstrated its ability to dynamically fuse and extract users activated core interests from noisy user behavior sequences by applying the self-attention mechanism. However, most of the SR models only retrieve the orders of user-item interactions, which is insufficient for capturing fully dynamic user behavior. This incomplete understanding of the core user preferences leads to poor recommendation quality.

Compared to SASRec, TiSASRec [23] introduces time interval embeddings to capture the temporal gaps between each user-item interaction in the sequences, which are then used in the computations of attention scores, providing a more nuanced understanding of user behavior. This enhances the distribution of attention scores from the query, key, and value (QKV) components in the self-attention mechanisms. The comparative studies between SASRec and TiSASRec on multiple highly sparse datasets [21], [24]. They are mainly focused on NDCG@10 and HR@10 values without further exploring the effects of other model parameters. Hence, further investigations are needed to find out the effects of model parameters, such as attention block, dropout rates, and time interval across different sparse datasets.

Several prior studies reported experimental configurations and training efficiency under different computational settings, although training time was not consistently documented across all works. Specifically, the research in [14], [18] did not explicitly report training time, despite the studies from [18] included the experimental evaluations involving the SASRec model. In contrast, in the studies from [15] did provided detailed training duration using a 4-core Intel i7 CPU with 32 GB RAM, reporting approximately one hour for the MovieLens dataset, two hours for the Gowalla and Foursquare datasets and one hour for the Tmall dataset.

Next, study from [16] has conducted experiments comparing SR models and reported per-epoch training times, where SASRec required only 1.7 seconds per epoch, compared to 19.1 seconds for Caser and 30.7 seconds for GRU4Rec+. Furthermore, SASRec converged to optimal performance within approximately 350 seconds on the MovieLens-1M dataset. Xia *et al.* [17] conducted experiments using an NVIDIA TITAN RTX GPU and an Xeon W-2133 CPU; while SASRec was evaluated on the Taobao dataset but TiSASRec was not tested, and the proposed model required approximately 32.0 seconds per epoch. Chang *et al.* [22] has reported that the SURGE model converged on the Taobao dataset in 18.74 minutes when trained on an NVIDIA Tesla V100 GPU with 32 GB memory. Similarly, Li *et al.* [25] was trained and evaluated using a single Tesla V100 GPU, although the total training time was not explicitly stated. Ultimately, most of studies are tested with the MovieLens dataset along with the SASRec model in the studies of SR, yet there has not been a study for the training time taken across different parameters of SASRec and TiSASRec under an extremely sparse dataset of the Taobao dataset.

3. METHOD

This study implemented the comparison between SASRec and TiSASRec models with MovieLens-1M dataset and Taobao user-behavior datasets. Compared to SASRec model, the TiSASRec model maps out the absolute positions and relative time intervals of each item index from the user-item interactions to emphasize the importance and correlations between items. This helps the attention blocks calculate attention weights based on these embeddings and compute the users' preference scores for each item. Thus, the TiSASRec model discerns the varying impact (noises) of past interactions on future interactions, regardless of the sparsity of the user-item interaction sequences.

The first experiment is carried out by applying both SASRec and TiSASRec models in MovieLens-1M dataset and Taobao user-behavior dataset, respectively with default model parameters to compare their performances in sparse datasets. Then, further experiments are carried out to perform a comparative analysis on the performance of both models with different parameter values. Both models are trained and run in the environment of NVIDIA GeForce GTX 1080. The descriptions of the datasets are summarized in Table 1 and elaborated as follows:

- MovieLens [26]: a dataset collected by GroupLens which contains 1,000,209 anonymous ratings (1~5) of approximately 3,900 movies made by 6,040 MovieLens users in the year 2000. The user-item interaction records are stored in 'ratings.dat', where it consists of movie ratings with four different labels of 'UserID', 'MovieID' 'Rating', and 'Timestamp'.
- Taobao [27]: a dataset collected and offered from Alibaba for an online shopping platform, which contains 100,150,807 anonymous interactions from 987,994 users for 4,162,024 items with 9,439 categories. The user-item interaction records are stored in 'UserBehavior.csv', where it consists of user behavior data with five different labels of 'User ID', 'Item ID', 'Category ID', 'Behavior type', and 'Timestamp'.

Table 1. Statistics of Taobao user-behavior dataset

Datasets	Numbers of unique user	Numbers of unique item	Total interactions	Data sparsity rate (%)	Data density rate (%)	Average sequence length
Taobao user-behavior	987,994	4,162,024	100,150,807	99.85	0.15	5.50
MovieLens-1M	6,040	3,416	999,611	95.16	4.84	163.5

Dataset sparsity refers to the proportion of missing or unavailable interactions between users and items over time. A sparse dataset has a large number of unobserved interactions relative to the possible total interactions, meaning that users interact with only a small fraction of the available items. Conversely, a dense dataset has more interactions and fewer gaps, indicating a higher coverage of the potential user-item pairs. The relationship between sparsity and density is inverse as the density increases, the sparsity decreases.

The Taobao user-behavior dataset has a higher data sparsity rate compared to the MovieLens-1M dataset despite holding larger numbers of unique users and items. In general, a dataset with higher density holds more user-item interaction history and vice versa. This leads to better training for machine learning models, as it has more data to learn patterns from. Richer content helps the recommendation model capture user preferences more effectively, resulting in improved performance and more accurate recommendations. In contrast, a sparse dataset can lead to underfitting or less reliable recommendations, as the model has limited data to generalize user preferences or infer patterns.

Based on Table 1, both datasets are shown to have different lengths of user-item interaction length for each user. The datasets with longer average sequence length have relatively richer interaction data and

potential to return better values in terms of performance metrics. The $\sum_{u \epsilon user\_train} len(user\_train[u])$ is referred as the total number of interactions across all users while the $|user\_train|$ is the total number of users in the training set. The equation of calculations on average sequence length of a user in a dataset is defined in (1).

$$Average\ Sequence\ Length = \frac{\sum_{u \epsilon user\_train} len(user\_train[u])}{|user\_train|} \quad (1)$$

3.1. Data pre-processing, padding, and train-test-validation split

The data pre-processing steps are conducted based on [16] and [23], which include data cleaning, data filtering with five core user interactions, and arrangements based on timestamps. The process of filtering out at least five core user interactions from each user is to ensure that the training, testing and validation data are sufficient. Upon filtering on 5 core user interactions, each user is pre-processed with padding with number of 200 and 10 respectively for MovieLens-1M and Taobao dataset. Padding is a technique that is widely applied in SR models to handle the sequences of different lengths. For SR, each user has different lengths of user-item interactions, where it requires a fixed length of user-item interaction sets for effective training of the SR model. Padding adds some dummy items at the end of each user's item interaction sequences until it reaches the same length as the longest sequence. For example, when the user-item interaction sequence of $(S_1^u, S_2^u, \dots, S_{15}^u, S_{16}^u)$ has a maximum interaction until S_{16}^u, the padding technique is applied to expand the user-item interaction sequence until S_{50}^u to fulfil the maximum length of SASRec that is fixed to 50 in default. In SASRec, it handles variable-length sequences by filling up dummy data with value '0'. In this way, the model can process the sequences in batches.

In SASRec and TiSASrec, every user-item sequence is split into training, testing and validation sets. For training set, it holds the whole user-item interaction sequence of each user but excluding the two latest items at the end of the sequence input. On the other hand, testing set is retrieved from the latest interacted item of each user, while the second most recent interacted items are treated as validation set. The validation set is used to verify the training performance with 20 epochs. Upon running the last epoch, the model is verified with the testing set. Finally, the performances of both models are evaluated with NDCG and hit rate metrics.

3.2. Model implementation

Both models are implemented to compare their performance on two datasets with the maximum length of sequence 200 and 10 for MovieLens-1M and Taobao dataset respectively. The model parameters and hyperparameters of SASRec and TiSASRec are configured to default values in the first comparison. Then, both models are investigated with different combinations of parameters for both datasets. The key parameters in the SASRec model include: hidden_units', 'num_blocks', 'num_heads', and 'dropout_rate'. The key parameters in TiSASRec are similar with SASRec model with an additional parameter of 'time_span' to indicate different values of time-interval. The default parameter values are stated as: hidden_units =50', 'num_blocks =2', 'num_heads =1', 'dropout_rate =0.2', and 'time_span =256'.

3.3. Evaluation metrics

To evaluate the performance of the models with existing works, this study focuses on implementing normalized discounted cumulative gain (NDCG@10) and hit rate (HR@10) evaluation metrics [10]. The k in (2) is a defined value of a cutoff point while evaluating the ranking quality. For example, the cutoff point in this study for both datasets are 10, which only looks at the top 10 items for recommendation. NDCG is indicated as a position-aware metric, which measures the performance of a recommendation model by considering the rankings of items and assigning scores to the higher ranks of items. Contrary to NDCG, (HR@10) focuses more on the presence of the correct item in the top 10 list, which is deduced in (3). $hit@k$ refers to the number of hits for the user u_i and $\#I(u_i)$ denotes the number of interactions of user u_i. To prevent a severe computation burden on all user-item pairs, the study uses 100 randomly chosen negative samples and mixes with the ground-truth items (positive samples) to rank these items for each user *u*.

$$NDCG@k = \sum_{i=1}^{k} \frac{2^{r_i}-1}{(i+1)} \quad (2)$$

$$HR@k = \frac{1}{|u|} \sum_{i=1}^{|u|} \frac{\#hit@k}{\#I(u_i)} \quad (3)$$

Both $NDCG@k$ and $HR@k$ values range from 0 to 1. The higher value of NDCG and HR indicates better recommendation quality. NDCG@k and HR@k values that equal to 1 in the case of ideal ranking, where all the items are perfectly included in the top-k list and are sorted by relevance. On the other hand,

$NDCG@k$ and $HR@k$ value that equal 0 indicates that there are no relevant items included in the top-k list. Therefore, the higher the values of $NDCG@k$ and $HR@k$, the more relevant items are included in the top-k list and sorted by relevancy.

4. EXPERIMENTAL RESULTS AND ANALYSIS

The experiments are conducted with SASRec and TiSASRec models on two sparse datasets. The experimental results have recorded the scoring in NDCG@10 and HR@10 along with their training time. The experimental results are shown in Tables 2 to 5 to investigate the effects of varying values of model parameters on different sparsity of datasets.

4.1. Effects of model parameters on the lower sparsity dataset

Tables 2 and 3 depict the comparison of SASRec and TiSASRec models with varying parameter values and the training time for processing the MovieLens-1M dataset. Each experiment uses different values for attention blocks and dropout rates, in order to compare with the default parameters (attention blocks =2, dropout rate =0.2). The performance metrics NDCG@10 and R@10 are used to assess the performance of different parameter configurations, with changes compared to the default parameters reflected as percentages.

Table 2. NDCG@10 and training time comparison of SASRec and TiSASRec in MovieLens-1M dataset

Dataset	Model parameters	SASRec			TiSASRec		
		NDCG@10	Changes (%)	Training time	NDCG@10	Changes (%)	Training time
MovieLens-1M	**Attention block =2 Dropout rate =0.2**[1]	0.5871	-	0 h 23 m 35 s	0.6030	-	3 h 57 m 4 s
	Attention block =4 Dropout rate =0.2	0.5977	+1.80	0 h 55 m 2 s	**0.6091**[2]	+1.01	17 h 26 m 56 s
	Attention block =8 Dropout rate =0.2	0.5981	+1.87	1 h 29 m 29s	0.5965	-1.08	23 h 28 m 5 s
	Attention block =2 Dropout rate =0.3	0.5854	-0.29	0 h 37 m 31 s	0.5997	-0.55	4 h 32 m 33 s
	Attention block =2 Dropout rate =0.4	**0.5786**[3]	-1.45	0 h 43 m 48 s	0.5987	-0.71	4 h 32 m 25 s
	Attention block =2 Dropout rate =0.2 Time-span =1,024	-	-	-	0.6028	-0.03	3 h 50 m 39 s
	Attention block =2 Dropout rate =0.2 Time-span =2,048	-	-	-	0.6023	--0.12	3 h 46 m 1 s

*Note:
[1]Default parameter values (additional time-span parameter value is 256 for TiSASRec
[2]Indicates as the highest performance value in the dataset
[3]Indicates as the lowest performance value in the dataset

Table 3. HR@10 and training time comparison of SASRec and TiSASRec in MovieLens-1M dataset

Dataset	Model parameters	SASRec			TiSASRec		
		HR@10	Changes (%)	Training time	HR@10	Changes (%)	Training time
MovieLens-1M	**Attention block =2 Dropout rate =0.2**[1]	0.8214	-	0 h 23 m 35 s	0.8263	-	3 h 57 m 4 s
	Attention block =4 Dropout rate =0.2	0.8257	+0.52	0 h 55 m 2 s	**0.8308**[2]	+0.55	17 h 26 m 56 s
	Attention block =8 Dropout rate =0.2	0.8227	+0.16	1 h 29 m 29 s	0.8224	-0.47	23 h 28 m 5 s
	Attention block =2 Dropout rate =0.3	0.8217	+0.04	0 h 37 m 31 s	0.8233	-0.36	4 h 32 m 33 s
	Attention block =2 Dropout rate =0.4	**0.8166**[3]	-0.59	0 h 43 m 48 s	0.8242	-0.25	4 h 32 m 25 s
	Attention block =2 Dropout rate =0.2 Time-span =1,024	-	-	-	0.8237	-0.31	3 h 50 m 39 s
	Attention block =2 Dropout rate =0.2 Time-span =2,048	-	-	-	0.8263	-	3 h 46 m 1 s

*Note:
[1]Default parameter values (addtional time-span parameter value is 256 for TiSASRec)
[2]Indicates as the highest performance value in the dataset
[3]Indicates as the lowest performance value in the dataset

TiSASRec model achieves the highest values of NDCG@10 and HR@10 in MovieLens-1M dataset with 0.6091 and 0.8308, respectively for the configurations of 4 attention blocks and 0.2 dropout rate. On the other hand, SASRec model exhibits the lowest values of NDCG@10 and HR@10 with 0.5786 and 0.8166 respectively for the configurations of 2 attention blocks and 0.4 dropout rate. Although TiSASRec has a generally longer training time compared to SASRec for MovieLens-1M, it is worth noting that the TiSASRec provides more accurate recommendation ranking with higher NDCG@10 and HR@10 values compared to SASRec. A recommendation model that delivers superior recommendation quality can lead to better customer satisfaction.

Both SASRec and TiSASRec models are tested with different numbers of attention blocks to investigate the effects on their recommendation performance. In the MovieLens-1M dataset, TiSASRec performed best with these modifications. Attention blocks consist of self-attention layers that help preserve sequence order and improve training. Adding more attention blocks increases the models' complexity and training time for deeper representation learning. For SASRec, increasing attention blocks from 2 to 8 improved NDCG@10 from 0.5871 to 0.5981 (a 1.87% improvement). However, HR@10 decreased from 0.52% to 0.16% when increasing from 4 to 8 attention blocks. This means that while more attention blocks improve the ranking of relevant items, they may also result in missing some items in the top 10 list. In denser datasets like MovieLens-1M, increasing attention blocks helps accurately rank relevant items but may narrow the scope of the top 10 recommendations, focusing on a few highly ranked items rather than a broader range. This can make the model less effective at identifying more relevant items.

NDCG@10 of TiSASRec slightly increased from 0.6030 to 0.6091 when the attention blocks were increased to 4, but dropped by 1.08% to 0.5965 when increased to 8. As the model complexity increased with more attention blocks, the training time also increased, but without significant performance gains. This suggests that 8 attention blocks are excessive for this dataset, which has a sparsity of 95.16%, and may lead to poor generalization. Despite using the same number of attention blocks, TiSASRec consistently outperformed SASRec in NDCG@10 and HR@10. This is because TiSASRec incorporates time interval information into its self-attention mechanism, capturing temporal dynamics and evolving user preferences more effectively. This temporal awareness allows TiSASRec to provide better recommendations. However, TiSASRec generally requires more training time due to the added complexity of processing both item sequences and temporal information. Increasing the time-span value for the TiSASRec model in the MovieLens-1M dataset slightly worsened the NDCG@10 performance. The NDCG@10 value dropped from 0.6030 to 0.6028 and 0.6023 for time-span values of 1,024 and 2,048, respectively. For HR@10, the value dropped from 0.8263 to 0.8237 at a time span of 1,024 but remained the same at 2,048. This indicates that increasing the time span in the denser MovieLens-1M dataset does not improve NDCG@10 performance due to added noise in the training data. As the time span increases, earlier interactions that are less relevant to the current recommendation task are included, turning into noise as user preferences change over time. This prevents the model from accurately predicting the next relevant item.

For SASRec, the dropout rate helps prevent overfitting by randomly setting a fraction of the input to zero during training. While HR@10 slightly improves by 0.04% at a dropout rate of 0.3, increasing the dropout rate generally has a minimal positive impact on NDCG@10 and HR@10. At a dropout rate of 0.4, NDCG@10 and HR@10 drop to 0.5786 and 0.8166, respectively, indicating worse recommendation quality and less effective top-10 item inclusion. As the dropout rate increases, SASRec's performance stops improving, leading to underfitting and an inability to learn meaningful patterns from the data. For TiSASRec, increasing the dropout rate does not improve NDCG@10 and HR@10 values. A higher dropout rate causes TiSASRec to lose too much information during training, impairing its ability to capture complex patterns and interactions, and ultimately failing to rank relevant items effectively.

4.2. Effects of model parameters on higher sparsity dataset

Similar to the previous section, Tables 4 and 5 depict the comparison of both models with varying values of model parameters and the training time for processing the Taobao dataset. The default parameters for attention blocks, dropout rate and time span remain the same as in the previous section. SASRec model achieves the highest value of NDCG@10 and HR@10 with 2 attention blocks and a 0.3 dropout rate. In contrast, the TiSASRec model has the lowest NDCG@10 and HR@10 with 8 attention blocks and a 0.2 dropout rate. This shows that SASRec provides more accurate recommendations on the higher sparsity Taobao dataset.

In contrast to MovieLens-1M dataset, SASRec shows better performance with changes in attention blocks instead of TiSASRec. Increasing the attention blocks in SASRec on the Taobao dataset led to significant improvements, especially with 4 attention blocks, showing a 9.05% increment in NDCG@10 and 11.23% in HR@10. With 4 attention blocks and a 0.2 dropout rate, SASRec achieved NDCG@10 of 0.2506 and HR@10 of 0.3823. However, performance dropped to 0.2207 and 0.3442 when the attention blocks increased to 8. Despite the increased complexity and training time, SASRec's performance slightly decreased

after 4 attention blocks due to overfitting on the higher sparsity dataset. The Taobao dataset, being sparser than MovieLens-1M, has a lower density, leading to more missing user-item interactions and inefficient training. Thus, the time interval-aware information used by TiSASRec is less effective in higher sparsity datasets, reducing its advantages over SASRec.

Table 4. NDCG@10 and training time comparison of SASRec and TiSASRec in Taobao dataset

Dataset	Model parameters	SASRec			TiSASRec		
		NDCG @10	Changes (%)	Training time	NDCG@10	Changes (%)	Training time
Taobao	**Attention block =2 Dropout rate =0.2**[1]	0.2298	-	3 h 57 m 4 s	0.2346	-	4 h 28 m 43 s
	Attention block =4 Dropout rate =0.2	0.2506	+9.05	10 h 29 m 42 s	0.2359	+0.55	1 h 39 m 23 s
	Attention block =8 Dropout rate =0.2	0.2207	-3.96	9 h 37 m 6 s	**0.2203**[3]	-6.09	4 h 39 m 35 s
	Attention block =2 Dropout rate =0.3	**0.2646**[2]	+15.14	4 h 12 m 26 s	0.2620	+11.68	1 h 18 m 33 s
	Attention block =2 Dropout rate =0.4	0.2623	+14.14	5 h 13 m 33 s	0.2561	+9.17	3 h 39 m 5 s
	Attention block =2 Dropout rate =0.2 Time-span =1024	-	-	-	0.2527	+7.72%	3 h 13 m 25 s
	Attention block =2 Dropout rate =0.2 Time-span =2048	-	-	-	0.2539	+7.81%	3 h 30 m 35 s

*Note:
[1]Default parameter values (additional time-span parameter value is 256 for TiSASRec)
[2]Indicates as the highest performance value in the dataset
[3]Indicates as the lowest performance value in the dataset

Table 5. HR@10 and training time comparison of SASRec and TiSASRec in Taobao dataset

Dataset	Model parameters	SASRec			TiSASRec		
		HR@10	Changes (%)	Training time	HR@10	Changes (%)	Training time
Taobao	**Attention block =2 Dropout rate =0.2**[1]	0.3437	-	3 h 57 m 4 s	0.3407	-	4 h 28 m 43 s
	Attention block =4 Dropout rate =0.2	0.3823	+11.23	10 h 29 m 42 s	0.3458	+1.50	1 h 39 m 23 s
	Attention block =8 Dropout rate =0.2	0.3442	+0.25	9 h 37 m 6 s	**0.3315**[3]	-2.70	4 h 39 m 35 s
	Attention block =2 Dropout rate =0.3	**0.3947**[2]	+14.85	4 h 12 m 26 s	0.3752	+10.14	1 h 18 m 33 s
	Attention block =2 Dropout rate =0.4	0.3882	+12.95	5 h 13 m 33 s	0.3755	+10.21	3 h 39 m 5 s
	Attention block =2 Dropout rate =0.2 Time-span =1024	-	-	-	0.3565	+4.64	3 h 13 m 25 s
	Attention block =2 Dropout rate =0.2 Time-span =2048	-	-	-	0.3601	+5.70	3 h 30 m 35 s

*Note:
[1]Default parameter values (additional time-span parameter value is 256 for TiSASRec)
[2]Indicates as the highest performance value in the dataset
[3]Indicates as the lowest performance value in the dataset

NDCG@10 of SASRec improved by 15.14% from 0.2298 to 0.2646 when the dropout rate increased from 0.2 to 0.3. Dropout rate helps prevent overfitting and improve generalization, which is why SASRec showed such improvement on the sparse Taobao dataset. Compared to the MovieLens-1M dataset, SASRec had more significant improvements with different dropout rate on Taobao. A high sparsity of 99.85% in Taobao indicates fewer interactions per user, which increases data complexity for recommendation models. This complexity arises from shorter user-item sequences and noise sensitivity, making it hard for models to identify user preferences. Models struggle with pattern recognition due to limited data and increased noise as random interactions may be misinterpreted as significant patterns. As a result, models may learn noise patterns instead of true preferences, reducing their recommendation performance.

SASRec has a simpler architecture without time interval-aware embeddings compared to TiSASRec. Increasing dropout rates in SASRec helps by randomly dropping neurons during training, leading to better generalization, and improved ranking accuracy. For example, with tuned model parameters of 2 attention blocks and 0.3 dropout rates for SASRec, it achieved 0.2646 for the NDCG@10 scoring on Taobao, slightly higher than TiSASRec's best NDCG@10 scoring of 0.2539. This indicates that in an extremely sparse setting like Taobao dataset, incorporating time intervals is yet to yield a large accuracy gain over the simpler SASRec. However, higher dropout rates require more epochs to converge, extending training times. The training time of SASRec increased for approximately 32.26% from 14,224 s at a 0.2 dropout rate to 18, 813 s at a 0.4 dropout rate by referring to Table 6. Conversely, TiSASRec's training time decreased for approximately 70.77% from 16,123 s to 4,713 s when the dropout rate increased to 0.3, hitting a sweet spot for higher sparsity dataset, where it effectively regularized the model, allowing it to converge more quickly by preventing overfitting without severely impacting its capacity to learn from the sparse data. However, further increment of dropout rate to 0.4 led to longer training times due to underfitting and optimization difficulties, which prolonged the training process as the model struggled to capture the limited patterns available in the sparse dataset. Therefore, a notable difference was observed in training efficiency where TiSASRec often converged faster with the existence of the time-span interval. In several configurations, TiSASRec's training time was significantly shorter than SASRec's on the same data. For instance, the SASRec with 4 self-attention blocks over 37,782 s to train on Taobao, whereas TiSASRec finished in about 5,963 s with the same number of blocks plus with the time-span of 256. However, further increment of dropout rate to 0.4 led to longer training times due to underfitting and optimization difficulties, which prolonged the training process as the model struggled to capture the limited patterns available in the sparse dataset.

Table 6. Training time comparison table between SASRec and TiSASRec for Taobao dataset

Dataset	Model parameters	SASRec training time	TiSASRec training time
Taobao	Attention block =2 Dropout rate =0.2 Time-span =256 (Not applicable for SASRec)	3 h 57 m 4 s (14,224 s)	4 h 28 m 43 s (16,123 s)
	Attention block =4 Dropout rate =0.2 Time-span =256 (Not applicable for SASRec)	10 h 29 m 42 s (37,782 s)	1 h 39 m 23 s (5,963 s)
	Attention block =8 Dropout rate =0.2 Time-span =256 (Not applicable for SASRec)	9 h 37 m 6 s (34,626 s)	4 h 39 m 35 s (16,775 s)
	Attention block =2 Dropout rate =0.3 Time-span =256 (Not applicable for SASRec)	4 h 12 m 26 s (15,146 s)	1 h 18 m 33 s (4,713 s)
	Attention block =2 Dropout rate =0.4 Time-span =256 (Not applicable for SASRec)	5 h 13 m 33 s (18,813 s)	3 h 39 m 5 s (13,145 s)

Based on the performance of TiSASRec in Taobao dataset, increasing the time-span value improves NDCG@10 and HR@10. With a time, span of 1,024, NDCG@10 and HR@10 increased from 0.2346 and 0.3407 to 0.2527 and 0.3565. With a time, span of 2,048, they further increased to 0.2539 and 0.3601. The higher time-span value captures long-term dependencies and assigns more accurate attention weights based on temporal distance, which is especially effective for a sparse dataset. However, the training time slightly increased by about 17 minutes when the time-span value is raised from 1,024 to 2,048. This is because the increment of the maximum range of time intervals between user-item interactions requires a broader attention scope in the sparse dataset and more complex calculations for attention weights, as it introduces more distant items into a single user-item interaction vector. Contrary to MovieLens-1M, the lack of training data in higher sparsity dataset like Taobao limits the performance of the recommendation model due to the shorter length of user-item interactions. As the time-span value increases, more data can be included for training TiSASRec, allowing it to learn core patterns that represent user preferences in sparse datasets, which is less likely to be misdirected in denser datasets. Based on the experimental results, TiSASRec has a generally lower training time compared to the results from SASRec model. However, the overall training time for TiSASRec is shorter than SASRec across different parameter configurations, making TiSASRec suitable for higher sparsity dataset with lesser processing time despite its slightly lower accuracy from the result.

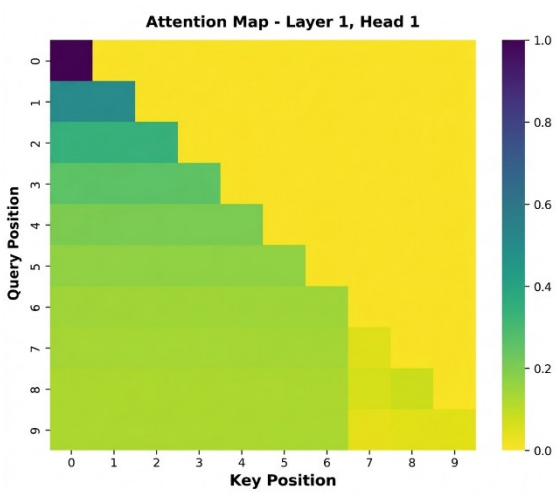
In TiSASRec, it introduced the time-span interval as one of the attempts on improvement of attention efficiency. Similarly in studies of [24], [25], they both implement distinct strategies of sparse cross-attention mechanisms and leverage cross-attention to incorporate contextual attributes, respectively to improve attention efficiency in sparse datasets. Generally, it serves as an enriching approach on the sparse

interaction data with auxiliary signals. Thus, it can achieve faster training and maintain relatively remarkable performance even in highly sparse environments, especially in extremely sparse datasets with a sparsity rate more than 95% and a shorter average sequence length. This suggests better training efficiency by retaining lower value of attention blocks and considering other contextual variables that help in improving attention efficiency when training with an extremely sparse dataset instead of testing with more attention blocks.

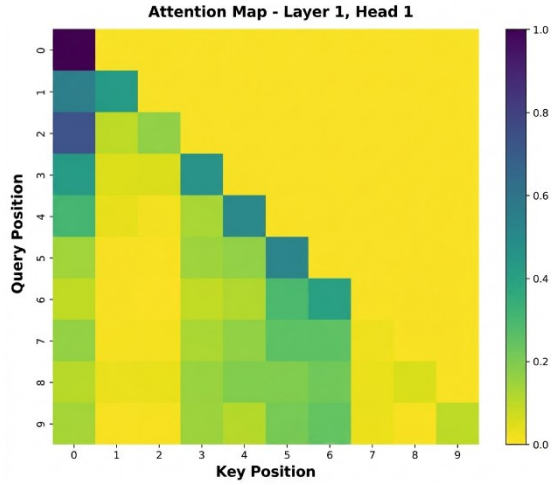
4.3. Attention weight distribution on both models in Taobao dataset

The values of NDCG@10 and HR@10 gradually improve as the value of time-span model parameters increases. Tables 2-5 show that both SASRec and TiSASRec model had significant improvements in performance evaluation metrics for Taobao dataset. However, the values of NDCG@10 and HR@10 in TiSASRec are relatively lower than the values of NDCG@10 and HR@10 in SASRec for Taobao dataset. This is further analysed in Figure 1, which illustrates the attention weight distribution map of the last 10 user-item sequences in both models. In Figures 1(a) and (b), the x-axis represents the position of items in sequence, and each point on the y-axis corresponds to a specific time step in sequence of user-item interactions.

The numbers of started from 0 to 9 which stating from the most recent to the oldest item position. On the other hand, the y-axis represents the time-steps at which the model makes recommendations, where 0 indicates as the first item and 9 refers as the last item to be recommended in a time-step. In Figure 1(a), the parameter settings of the SASRec attention weight distribution map are configured with default parameter values (attention blocks =2, dropout rate =0.2') while Figure 1(b) is configured with 'attention blocks =2', 'dropout rate =0.2' and 'time-span =256'for the TiSASRec model.



(a)



(b)

Figure 1. Attention weight distribution map in last 10 user-item sequence of (a) SASRec attention heatmap on Taobao and (b) TiSASRec attention heatmap on Taobao

The query position of the y-axis in the heatmap refers to the point in the user-item interaction sequence where the SASRec and TiSASRec models make recommendations, while the x-axis represents the index of items in the user-item sequences based on the last 10 interacted items. The attention weight values range from 0.0 to 1.0 is indicated by colors from lighter (yellow) to darker (blue). An attention weight value closer to 1.0 at the diagonal position of the heatmap indicates a more accurate recommendation, showing that the model tracks relevant items and their order in the sequence accurately. Most of the diagonal values of TiSASRec are closer to 1.0 (blue) compared to SASRec. In X-axis, if item number 3 is misplaced at query position 4 or above, it disrupts the self-attention mechanism, leading to incorrect weighting and poorer performance. In sparse datasets, the focus should be on recent items due to limited average sequence length, aligning better with user preferences.

Thus, attention weights in sparser datasets should prioritize recent items over past interactions, which are less informative. In Figure 1(b), the diagonal line from position (0.0) to (9.9) in a diagonal direction shows that attention weights are more concentrated on more recent items with higher attention weights (blue), indicating a better distribution with higher attention weights in a diagonal line in sparser datasets with a time-span value of 1,024. This suggests that TiSASRec has a better understanding of user-item interactions by referring on the time interval differences and prioritizing recent items, while past items receive lower attention weights when the time span is 2,048, indicating difficulty in identifying meaningful patterns in older items. Overall, the TiSASRec model effectively captures user behavior by considering time intervals and self-attention mechanisms, even in sparse datasets.

The use of time intervals in the sparse Taobao dataset provides additional insights beyond item sequences, helping TiSASRec understand the relevancy of each item within the sequence. Unlike SASRec, which only considers the sequence order of interacted items and treats all items as equally relevant, TiSASRec takes into account the time intervals between interactions. This is crucial for the Taobao dataset, which has lower density and shorter user-item sequences compared to MovieLens-1M, resulting in insufficient historical data for training.

If the recommendation model focuses on the most recent interactions in such sparse datasets by considering the latest interactions and their time intervals, TiSASRec can capture the influence of each interaction and understand item relevancy based on time interval differences. The closer time intervals between items contributes to the higher relevancy and attention weights for TiSASRec. This results in darker colors (blue) for recent items and lighter colors (yellow) for past items on the heatmap. Focusing on the latest interactions helps TiSASRec achieve relatively higher NDCG@10 values.

5. CONCLUSION

The self-attention SR model uses a self-attention mechanism to weigh the importance of each item in a user-item sequence by capturing long-range dependencies and relationships. TiSASRec enhances this by considering the time intervals of interactions, outperforming SASRec. This study successfully experimented with both SASRec and TiSASRec models across datasets with different sparsity levels and analyzed key model parameters affecting the recommendation results. The performance of the self-attention model depends on the number of attention blocks and the dropout rate. Increasing attention blocks has a limited positive impact on sparse datasets due to poor data representation, and excessive attention blocks lead to computational overhead and poor performance. Higher dropout rates help prevent overfitting without significantly hindering learning from sparse datasets. For both models, higher dropout rates can improve generalization and prevent overfitting but may also increase training times and cause underfitting if set too high, especially in SASRec when dealing with high sparsity dataset. Conversely, TiSASRec may benefit from moderate increases in dropout rates, achieving faster convergence and reduced training times, but may also experience increased training duration if the dropout rate becomes excessively high. The time-span parameter is crucial in TiSASRec, especially for sparse datasets. It provides context on the recency and relevance of items, helping the model adjust the weight of historical interactions based on temporal patterns. This adjustment improves the handling of irregular interactions as it serves as an improvement in attention efficiency. The increase of attention efficiency in terms of considering other contextual information, like time intervals and item metadata, helps in providing improved training efficiency with remarkable NDCG@10 and HR@10 scoring, especially in extremely sparse datasets that are also short in average sequence length

FUNDING INFORMATION

This work was supported by the Telekom Malaysia Research and Development under Grant RDTC/241125 (MMUE/240066).

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Weishan Ooi	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓			
Lee Yeng Ong		✓			✓	✓	✓	✓		✓	✓	✓	✓	
Meng-Chew Leow					✓		✓			✓				✓

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

DATA AVAILABILITY

Data availability is not applicable to this paper as no new data were created or analyzed in this study.

REFERENCES

[1] A. A. Patoulia, A. Kiourtis, A. Mavrogiorgou, and D. Kyriazis, "A comparative study of collaborative filtering in product recommendation," *Emerging Science Journal*, vol. 7, no. 1, pp. 1–15, Feb. 2023, doi: 10.28991/ESJ-2023-07-01-01.

[2] B. Li, A. Maalla, and M. Liang, "Research on recommendation algorithm based on e-commerce user behavior sequence," in *2021 IEEE 2nd International Conference on Information Technology, Big Data and Artificial Intelligence, ICIBA 2021*, 2021, pp. 914–918, doi: 10.1109/ICIBA52610.2021.9688086.

[3] Y. Peng, "A survey on modern recommendation system based on big data," May 2022. *arXiv:2206.02631v2*.

[4] M. Grover, "The development of sequential recommendation systems using deep learning," in *2023 International Conference on Data Science and Network Security, ICDSNS 2023*, 2023, doi: 10.1109/ICDSNS58469.2023.10245109.

[5] Z. Batmaz, A. Yurekli, A. Bilge, and C. Kaleli, "A review on deep learning for recommender systems: challenges and remedies," *Artificial Intelligence Review*, vol. 52, no. 1, pp. 1–37, Jun. 2019, doi: 10.1007/s10462-018-9654-y.

[6] H. Fang, D. Zhang, Y. Shu, and G. Guo, "Deep learning for sequential recommendation," *ACM Transactions on Information Systems*, vol. 39, no. 1, pp. 1–42, Jan. 2020, doi: 10.1145/3426723.

[7] M. Nasir and C. I. Ezeife, "A survey and taxonomy of sequential recommender systems for e-commerce product recommendation," *SN Computer Science*, vol. 4, no. 6, Nov. 2023, doi: 10.1007/s42979-023-02166-5.

[8] C. Yan, Y. Wang, Y. Zhang, Z. Wang, and P. Wang, "Modeling long- and short-term user behaviors for sequential recommendation with deep neural networks," in *Proceedings of the International Joint Conference on Neural Networks*, Jul. 2021, vol. 2021-July, doi: 10.1109/IJCNN52387.2021.9534103.

[9] D. Roy and M. Dutta, "A systematic review and research perspective on recommender systems," *Journal of Big Data*, vol. 9, no. 1, 2022, doi: 10.1186/s40537-022-00592-5.

[10] S. Wang, L. Hu, Y. Wang, L. Cao, Q. Z. Sheng, and M. Orgun, "Sequential recommender systems: challenges, progress and prospects," *IJCAI International Joint Conference on Artificial Intelligence*, pp. 6332–6338, 2019, doi: 10.24963/ijcai.2019/883.

[11] H. Fang, D. Zhang, Y. Shu, and G. Guo,"Deep learning for sequential recommendation," *ACM Transactions on Information Systems*, vol. 39, no. 1, Nov. 2020, doi: 10.1145/3426723.

[12] S. Wang, Q. Zhang, L. Hu, X. Zhang, Y. Wang, and C. Aggarwal, "Sequential/session-based recommendations: challenges, approaches, applications and opportunities," in *SIGIR 2022 - Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, Jul. 2022, pp. 3425–3428, doi: 10.1145/3477495.3532685.

[13] J. H. Yoon and B. Jang, "Evolution of deep learning-based sequential recommender systems: from current trends to new perspectives," *IEEE Access*, vol. 11, pp. 54265–54279, 2023, doi: 10.1109/ACCESS.2023.3281981.

[14] B. Hidasi and A. Karatzoglou, "Recurrent neural networks with top-K gains for session-based recommendations," in *International Conference on Information and Knowledge Management, Proceedings*, 2018, pp. 843–852, doi: 10.1145/3269206.3271761.

[15] J. Tang and K. Wang, "Personalized top-N sequential recommendation via convolutional sequence embedding," in *WSDM 2018 - Proceedings of the 11th ACM International Conference on Web Search and Data Mining*, Feb. 2018, pp. 565–573, doi: 10.1145/3159652.3159656.

[16] W. C. Kang and J. McAuley, "Self-attentive sequential recommendation," in *IEEE International Conference on Data Mining, ICDM*, Dec. 2018, pp. 197–206, doi: 10.1109/ICDM.2018.00035.

[17] L. Xia, C. Huang, Y. Xu, and J. Pei, "Multi-behavior sequential recommendation with temporal graph transformer," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 6, pp. 6099–6112, Jun. 2023, doi: 10.1109/TKDE.2022.3175094.

[18] J. Wang, K. Ding, and J. Caverlee, "Sequential recommendation for cold-start users with meta transitional learning," in *SIGIR 2021 - Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, Jul. 2021, pp. 1783–1787, doi: 10.1145/3404835.3463089.

[19] N. Idrissi and A. Zellou, "A systematic literature review of sparsity issues in recommender systems," *Social Network Analysis and Mining*, vol. 10, no. 1, Dec. 2020, doi: 10.1007/s13278-020-0626-2.

[20] J. Qin *et al.*, "Learning to retrieve user behaviors for click-through rate estimation," *ACM Transactions on Information Systems*, vol. 41, no. 4, Apr. 2023, doi: 10.1145/3579354.
[21] Z. Luo, Z. Huang, J. Tang, Y. Hou, and Y. Gao, "Time-aware self-attention meets logic reasoning in recommender systems," *2024 International Joint Conference on Neural Networks (IJCNN),* Aug. 2024, doi: 10.1109/IJCNN60899.2024.10651142.
[22] J. Chang *et al.*, "Sequential recommendation with graph neural networks," in *SIGIR 2021 - Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, Jul. 2021, pp. 378–387, doi: 10.1145/3404835.3462968.
[23] J. Li, Y. Wang, and J. McAuley, "Time interval aware self-attention for sequential recommendation," in *WSDM 2020 - Proceedings of the 13th International Conference on Web Search and Data Mining*, Jan. 2020, pp. 322–330, doi: 10.1145/3336191.3371786.
[24] A. Rashed, S. Elsayed, L. S. -Thieme, "CARCA: context and attribute-aware next-item recommendation via cross-attention," Apr. 2022, *arXiv:2204.06519*.
[25] C. Li *et al.*, "STRec: sparse transformer for sequential recommendations," in *Proceedings of the 17th ACM Conference on Recommender Systems, RecSys 2023*, Sep. 2023, pp. 101–111, doi: 10.1145/3604915.3608779.
[26] F. M. Harper and J. A. Konstan, "The MovieLens datasets: history and context," *ACM Transactions on Interactive Intelligent Systems*, vol. 5, no. 4, pp. 1–19, Jan. 2015, doi: 10.1145/2827872.
[27] T. L. Kitten, "User behavior data from Taobao for recommendation," *tianchi.aliyun.com*, 2018, Accessed: Jan. 11, 2024, [Online]. Available: https://tianchi.aliyun.com/dataset/649

BIOGRAPHIES OF AUTHORS

Weishan Ooi studied Bachelor of Computer Science (Hons.) Artificial Intelligence in Multimedia Unversity, Melaka, Malaysia. His area of interest includes machine learning, deep learning, and data sciences. He can be contacted at email: ooiweishan2001@gmail.com.

Lee Yeng Ong received Ph.D. degree in Computer Vision from Multimedia University, Malaysia. She is currently working as an assistant professor with Faculty of Information Science and Technology, Multimedia University, Malaysia. Her research interests include agentic AI, computer vision, data science, and big data analytics. She can be contacted at email: lyong@mmu.edu.my.

Meng-Chew Leow received his Doctor of Philosophy from Multimedia University. His research interest is in game-based learning, specifically in role-playing game-based learning. He is also interested in system science, practical spirituality, and philosophy. He can be contacted at email: mcleow@mmu.edu.my.