JOURNAL OF APPLIED INTELLIGENT SYSTEMS & INFORMATION SCIENCES
VOL. 2, ISSUE 1, PP. 29-40, JUNE 2021.
Available at: www.journal.research.fanap.com
DOI: https://doi.org/10.22034/jaisis.2021.304225.1033

ATTENTION-BASED AND POSITIONAL-AWARE NEURAL NETWORKS FOR NEXT-ITEM RECOMMENDATION

Nemat Saeidi [1,*], Hossein Shahamat [2]

[1] *Department of Artificial Intelligence Engineering, University of Isfahan, Isfahan, Iran*
[2] *Department of Electrical and Computer Engineering, Tarbiat Moders University, Tehran, Iran*

ABSTRACT

Next-item recommendation intends to predict user interest over sequential items given user historical behaviors. There has been a lot of past works for the next-item recommendation, according to Markov Chains(MCs) and Recurrent Neural Networks(RNNs). MCs perform best in sparse datasets and RNNs perform better in denser datasets. To improve these methods, several recommendation systems have been built on MC and RNN architectures. However, these methods struggle to consider long-range dependencies and uncover complex relationships in next-item recommendation. Due to the limitations of these methods, we apply self-attention using neural networks. The proposed method can consider long-range dependencies using self-attention. Also, the proposed method can uncover complex relationships to capture an efficient features representation using neural networks such as long short-term memory(LSTM), Bi-Directional LSTM and convolutional neural network(CNN) presenting for sequence modeling. In this paper, to choose the best model, several neural network models are evaluated and the best model is selected regarding the performance of self-attention-based neural networks. At each time step, this method tries to identify which items are 'relevant' from a user's action history and apply them to predict the next item. We are able to achieve a high accuracy rate and significantly outperform state-of-the-art next-item methods on sequential datasets.

KEYWORDS: recommendation system, deep learning, sequential data, user behavior, attention model

1. INTRODUCTION

Next-item recommendation is intended to predict the users' future behavior on the basis of their previous action sequences (Li et al., 2020; Rendle et al., 2010). By taking the order of users' previous actions into account sequential models introduce additional behavioral dynamics, unlike traditional personalized recommendation algorithms which seek to capture users' global tastes (QuadranaMassimo et al., 2018). What dynamic recommendation systems try to do is that they merge user behavior models with the concept of 'context' on the users' actions. Receiving the beneficial patterns from sequential dynamics is initially challenging because the dimension of the input space increases significantly with the number of previous actions used as context. Researches on next-item recommendation mainly focus on how to capture these high-order dynamics briefly (Tanjim et al., 2020).

As for the dependency among various interactions in one session as well as the correlation of behavior patterns among different sessions, traditional next-item recommendation systems embrace precisely proper and

* *Corresponding Author, Email:* saeidi.n@eng.ui.ac.ir

RECEIVED: : 11 APRIL, 2021; **ACCEPTED**: 12 JUNE, 2021; **PUBLISHED ONLINE**: 14 JUNE, 2021

valuable machine learning approaches in order to model sequential data, such as Markov Chain (MC) (Rendle et al., 2010; He et al., 2017) and session-based k-nearest neighbors (KNNs) (Yap et al., 2012). Traditional next-item recommendation systems face failure when it comes to modeling users' long-term patterns and they are criticized for their incomplete modeling problem and combining different sessions (Gao et al., 2021). Unlike MC models, Recurrent Neural Network (RNN) models have long 'memory' across users' long-term patterns, but they require a large amount of data before outperforming baselines (Li et al., 2020).

Furthermore, embedding all items relates to mapping items to a sequence of numbers. In a recommendation system, algorithms to train and discover complex relationships use this method of mapping (representing) items with learned vectors. These complex relationships can be long-range dependencies, extracted features, many items and correlations of items in a non-trivial way (Fayyaz et al., 2020; Hernández & Amigó, 2021).

Attention models focus on the part of input or memory modeling the reasoning process and they can be used as a sequential reasoning process. (Hernández & Amigó, 2021). A variant of the attention model is self-attention in that the attention component identifies different positions of a sequence to figure out a sequence representation (Hernández & Amigó, 2021) and self-attention can link distant items of the sequence data more directly than other models. In addition, Because self-attention does not have any models such as recurrent or convolutional, it is not aware of the positions of the previous items (Li et al., 2020). Adding positional encodings to the inputs can be a deterministic function or learnable positional embedding and self-attention can model the relative positions between two input items as pairwise relationships. Therefore, attention models can be applied to next-item recommendation systems to improve methods such as long short-term memory (LSTM), Bi-Directional LSTM and convolutional neural network (CNN).

Neural network models such as RNN, LSTM and CNN were presented for sequence modeling and successfully applied to complex systems and different applications like recommendation systems (Hernández & Amigó, 2021).

Inspired by the points mentioned, we try to apply self-attention mechanisms using state-of-the-art neural network models. Due to the self-attention mechanism and neural network model, the proposed method considers long-range dependencies and can uncover the related complex relationships in user sequential histories and lead to an efficient features representation or embedding by capturing item features and correlations. The proposed model significantly outperforms state-of-the-art next-item recommendation methods and all baselines on sequential datasets.

The remainder of this work is organized as follows. In Section2, the related work is prepared. Section 3 includes our proposed method consisting of the embedding layer, models, preprocessing the input and prediction layer. Experimental results are presented in Section 4. Finally, we provide conclusion in Section 5.

2. Related Work

The main focus of the recommendation systems is on modeling compatibility between users and items on the basis of previous feedback which might be explicit or implicit. Since the interpretation of the 'non-observed' data is ambiguous, modeling the implicit feedback might be difficult. Addressing this problem led to providing two methods i.e. point-wise (Hu et al., 2008) and pairwise (Rendle et al., 2009) methods. Matrix Factorization (MF) methods aim to discover the latent dimensions in order to represent users' preferences and items' properties. Moreover, they estimate interactions through the inner product between the user and item embeddings. Besides, Item Similarity Models (ISM) provides a basis for another line of work which does not explicitly model each user with latent factors. By learning the item-to-item similarity matrix, these models estimate a user's preference toward an item by measuring its similarities with items the user has previously interacted with. A variety of deep learning (DL) techniques have been introduced recently because of their success in the field (Zhang et al., 2019). One line of work seeks to use neural networks to extract item features for content-aware recommendation.

Since the Netflix Prize, temporal recommendation has performed very well on different tasks and this was achieved by explicitly modeling the timestamp of users' activities. By dividing time into several segments and

modelling users and items separately in each, TimeSVD++ yielded good results. These models are necessary to understand datasets that present significant temporal 'drift' (Koren, 2009). Next-item recommendation varies somewhat from this setting, because it only account for the order of actions, and models sequential patterns which are independent of time. Rather than considering temporal patterns per se, sequential models try to model the 'context' of users' actions on the basis of their recent activities (Tanjim et al., 2020).

Next-item recommendation systems mine patterns in user interaction sequences. Some capture item-item transition matrices to predict the next item (Cho et al., 2002; Li et al., 2020). For example, through matrix factorization and a transition matrix, Factorized Personalized Markov Chains (FPMC) models (Rendle et al., 2010) long-term preferences and dynamic transitions of users. The transition matrix is a first-order MC which only considers the relation between the current and the previous item. Fossil uses similarity-based methods and high-order MCs which assumes that the next item is related to several previous items and demonstrates that the high-order MC-based models have strong performance on sparse datasets. CNN based methods have also been proposed to consider several previous items as an 'image' and mine transitions between items via a CNN at a union-level Multi-Order Attentive Rank (MARank) (Yu et al., 2019) unifies both individual- and union-level previous item transitions by combining the residual network and multi-order attention. Sequential Hierarchical Attention Network (SHAN) models (Ying et al., 2018) the several previous items and a long history of a user via a two-layer attention mechanism to obtain the short- and long-term preferences of a user. RNNs based models use RNNs to model entire user sequences. However, RNNs based models generally show poor performance on sparse datasets (Liu et al., 2021).

In recent years, DL techniques, such as RNN and LSTM (HochreiterSepp & SchmidhuberJürgen, 1997), acquired great achievements in natural language processing (NLP), showing their effectiveness in processing sequential data. Hence, they have attracted increasing interest in the next-item recommendation, and many DL-based models have achieved the overall state-of-the-art performance (Zhang et al., 2019; He et al., 2017; Khoali et al., 2021; Fang, 2020).

Also, Attention mechanisms are effective in various tasks. Essentially the idea behind attention mechanisms is that outputs depend on specific parts of an input that are relevant. Attention structure has demonstrated encouraging potential in modeling sequential data, e.g., machine translation and text classification (Hernández & Amigó, 2021). currently, some works apply the attention structure in order to improve superior recommendation performances and interpretability (Sun et al., 2019; Fang, 2020). self-attention structure (for neural machine translation by Google 2017) has also been deployed in the next-item recommendation. Unlike vanilla attention, it does not include RNN structures but carries out much better than RNN-based models in recommendation systems (Fang, 2020).

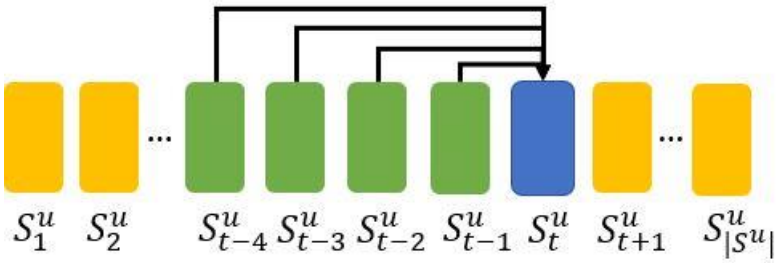

Fig. 1. the recommendation system considers a part of last items and uses the proposed model to focus on relevant items. This action is applied at each time step

3. METHODOLOGY

A next-item recommendation system is a system that captures the user's behavior long-term behavior as input and applies a recommendation algorithm to recommend suitable items or services to the user (Fig. 1). In

the setting of next-item recommendation, we give a user's action sequence $S^u = (S_1^u, S_2^u, \ldots, S_{|S^u|}^u)$, and explore to predict the next item. During the training process, the model predicts the next item depending on the previous t items. As shown in Fig. 1, it is suitable to consider the model's input as $(S_1^u, S_2^u, \ldots, S_{|S^u|-1}^u)$ and its output as a 'shifted' version: $(S_2^u, S_3^u, \ldots, S_{|S^u|}^u)$. In this section, we relate how we build a next-item recommendation model via an embedding layer, several models, and a prediction layer. We also further follow how approach differs from related works.

3.1. Embedding Layer

In the embedding layer, the training sequence $(S_1^u, S_2^u, \ldots, S_{|S^u|-1}^u)$ is adapted into a fixed-length sequence $S = (S_1, S_2, \ldots, S_n)$, where *n* is the behavior maximum length. While the sequence length is less than *n*, this method adds a 'padding' item to the left repeatedly until the length is *n* and while the sequence length is greater than *n*, the proposed method considers the most recent *n* actions. We create an item embedding matrix $M \in R_{|I| \times d}$ where *d* is the latent dimensionality. We also retrieve the input embedding matrix $E \in R_{n \times d}$, where $E_i = M_{si}$. A constant zero vector **0** is applied as the embedding for the padding item. Given that in the self-attention layer isn't any recurrent or convolutional layer, it doesn't understand the positions of past items. So, we inject a learnable position embedding:

$$\hat{\mathbf{E}} = \begin{bmatrix} \mathbf{M}_{s_1} + \mathbf{P}_1 \\ \mathbf{M}_{s_2} + \mathbf{P}_2 \\ \cdots \\ \mathbf{M}_{s_n} + \mathbf{P}_n \end{bmatrix} \tag{1}$$

In the proposed method, models capture the embedding $\hat{E}$ as input, convert it to three matrices through linear projections and finally feed them into layers.

3.2. Models of main Competitors

As mentioned previously, we describe a framework based on attention models to perform a recommendation system. Models are utilized for different problems in NLP as well as vision and are demonstrated to bring the performance of the proposed approach over single models. In the following subsections, we prepare an overview of models.

3.2.1. The CNN model architecture

CNN (Yang et al., 2019) have been very successful for several computer vision as well as NLP tasks in recent years. They are specifically robust in exploiting the local correlation and data pattern learned by their feature maps.

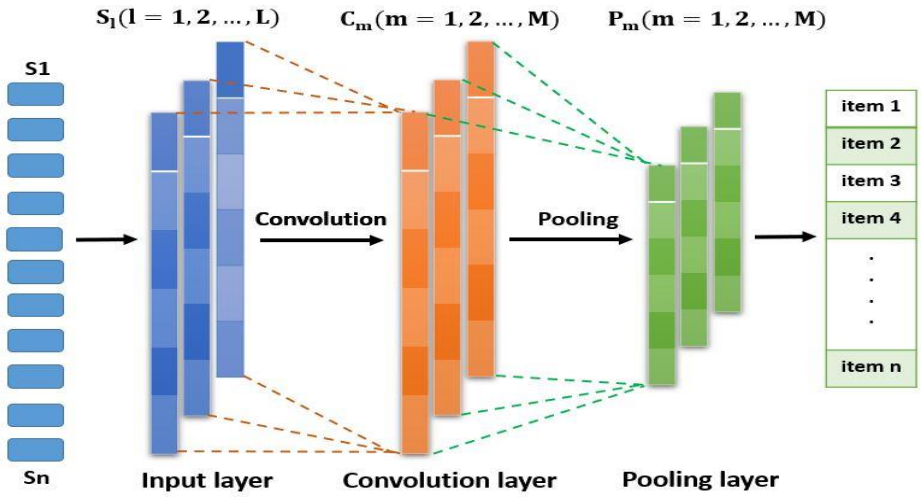

Fig. 2. The architecture of CNN containing of a convolution layer and a pooling layer

In our work, we apply 1D convolution layers. A 1D CNN can be viewed as a variant of a standard neural network. In contrast with standard neural networks that the hidden layer is fully connected, a 1D CNN introduces a network mechanism that alternates between the convolution layer and the pooling layer (Fig. 2).

3.2.2. The LSTM model architecture

In order to model sequential data –intended to be better than vanilla RNN model at capturing long-term dependencies- LSTM is used more often. LSTM network like other kinds of recurrent neural networks at each time-step receives the input from the current time-step and the output from the last time-step and generates an output which is fed to the next time step. After that, the hidden layer from the last time-step and sometimes all hidden layers are applied to classification (Alom et al., 2019).

As mentioned earlier, the vanilla RNN leads to the gradient vanishing or exploding problems and by introducing some internal gates LSTM network attempts to overcome this issue. There are three gates (input gate, output gate, forget gate) and a memory cell the LSTM model. The cell remembers values over arbitrary time intervals and the three gates regulate the flow of information into and out of the cell. Fig. 3 and Fig. 4 show the inner architecture of the single LSTM module and Bi-Directional LSTM.

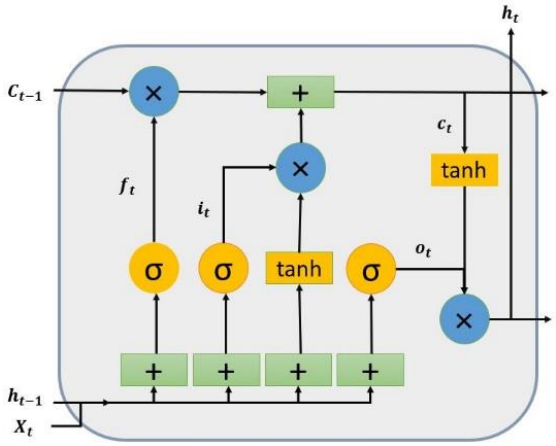

Fig. 3. The architecture of a standard LSTM module (Minaee et al., 2020)

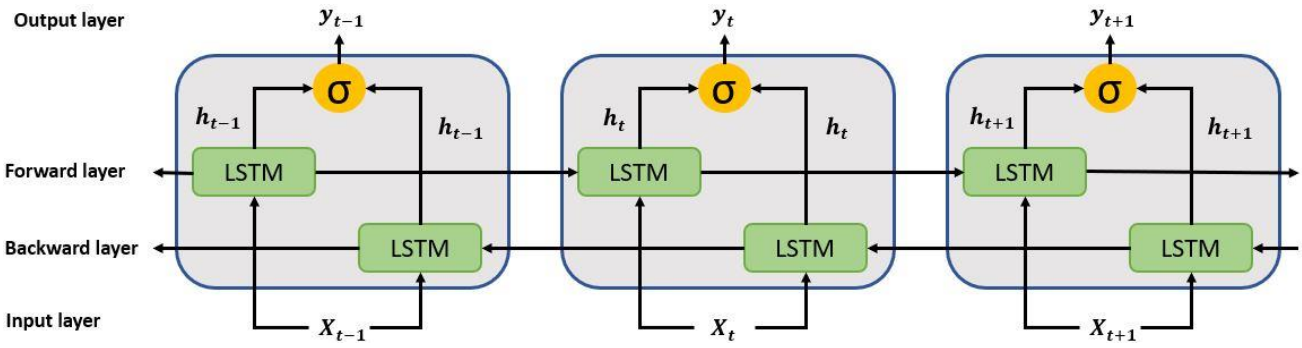

Fig. 4. The architecture of a standard Bi-Directional LSTM module

In Fig. 3 the elements of the architecture are calculated as follows:

$$i_t = \alpha(W_i[h_{t-1}, X_t] + b_i) \quad (2)$$

$$f_t = \alpha\big(W_f[h_{t-1}, X_t] + b_f\big) \quad (3)$$

$$\tilde{c_t} = tanh(W_c[h_{t-1}, X_t] + b_c) \quad (4)$$

$$c_t = f_t * c_{t-1} + i_t * \tilde{c_t} \quad (5)$$

$$o_t = \alpha(W_o[h_{t-1}, X_t] + b_o) \quad (6)$$

$$h_t = o_t * tanh(c_t) \quad (7)$$

Where X_t is the input at time-step t. σ stands for the element-wise sigmoid function to map the values within [0, 1], C_t represents the memory cell planned for lowering the risk of vanishing/exploding gradient, and thus learning the dependencies over a larger period of time is made feasible with traditional recurrent networks. The forget gate, F_t is to reset the memory cell.

We are concerned with the temporal information flow in both directions for many potential applications for which Bi-Directional LSTM, a variant of LSTM network, is introduced. Two hidden layers on the input sequence, the first one being on the input sequence as is and the second one on the reversed copy of the input sequence, are trained by the Bi-Directional LSTM networks. By reviewing all of the past and future information as well as the outcomes of the experiments in faster and better learning, may provide further context for the network (Cui et al., 2018).

3.2.3. The Attention model architecture

The inner relations among each one of the semantic spaces are shown on this layer. The strength of each behavior might be influenced by others in such a way that attention to some behaviors may be deflected while attention to the others might be intensified. This is achieved through the self-attention structure. This layer's output might be considered as the behavior representation sequence which takes into account the impacts of other layers in each latent space (Zhou et al., 2018; Vaswani et al., 2017).

An attention function is defined in Fig. 5. The weight assigned to each value is calculated using a compatibility function of the query with the corresponding key and thus the output is calculated as a weighted sum of the values. The scaled dot-product attention is computed using the following equation (Vaswani et al., 2017):

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}}\right)\mathbf{V}, \quad (8)$$

Where Q denotes the queries, d the feature dimension for each word, K the keys and V the values (each row denotes an item). Intuitively, the attention layer computes a weighted sum of all values, where the weight between query i and value j relate to the interaction between query i and key j. The scale factor $\sqrt{d}$ is to avoid overly large values of the inner product, specifically when the dimensionality is high (Vaswani et al., 2017). In tasks such as machine translation, attention structures are typically set with K = V.

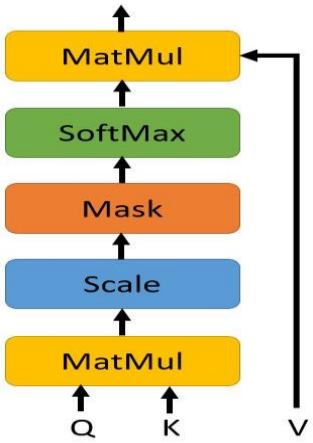

Fig. 5. Scaled Dot-Product Attention, mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors

3.3. *Preprocess input*

We perform the following operation as preprocessing input X:

$$g'(x) = x + \text{Dropout}(g(\text{LayerNorm}(x))) \tag{9}$$

Where $g(x)$ represents a layer. For layer g in each block, we use layer normalization on the input x before feeding into g, use dropout on g's output, and add the input x to the final output using residual connections (Vaswani et al., 2017).

Layer normalization is applied to normalize the inputs across features, which is useful for stabilizing and accelerating neural network training. Assuming the input is a vector x which includes all features of a sample, the operation is defined as:

$$\text{LayerNorm}(\mathbf{x}) = \boldsymbol{\alpha} \odot \frac{\mathbf{x} - \mu}{\sqrt{\sigma^2 + \epsilon}} + \boldsymbol{\beta}, \tag{10}$$

Where $\odot$ is an element-wise product, μ and σ are the mean and variance of $\mathbf{x}$, $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are learned scaling factors and bias terms.

Dropout randomly 'turn off' neurons with probability p during training, and apply all neurons when testing. The further analysis illustrates that dropout can be considered as an ensemble learning form which considers an enormous number of models that share parameters. We also use a dropout layer on the embedding $\hat{E}$ (Warde-Farley et al., 2014).

Residual networks are used to propagate low-layer features to higher layers. Therefore, if low-layer features are useful, the model can readily propagate them to the final layer.

3.4. *Prediction Layer*

After b self-attention blocks extract the information of formerly consumed items, the next item is predicted based on $F_t^{(b)}$. we adopt an MF layer in order to predict the relevance of item i:

$$r_{i,t} = \mathbf{F}_t^{(b)} \mathbf{N}_i^T \tag{11}$$

Where $r_{i,t}$ stands for the relevance of item i being the next item given the first t items and $N \in R_{|I| \times d}$ is an item embedding matrix. Therefore, a high interaction score $r_{i,t}$ means a high relevance, and we can generate recommendations by ranking the scores.

We adapt each user sequence excluding the last action like $S_1^u, S_2^u, \ldots, S_{|S^u|-1}^u$ to a fixed-length sequence $S^u = (S_1, S_2, \ldots, S_n)$ via truncation or padding items.

Our model takes a sequence s as input, the equivalent sequence o as the expected output, and we consider the binary cross-entropy loss as the objective function:

$$J(W) = -\sum_{\mathcal{S}^u \in \mathcal{S}} \sum_{t \in [1,2,\ldots,n]} \left[\log(\sigma(r_{o_t,t})) + \sum_{j \notin \mathcal{S}^u} \log(1 - \sigma(r_{j,t})) \right] \tag{12}$$

The Adam optimizer is applied to optimize the network. Also, the Adam optimizer is a variant of stochastic gradient descent (SGD) with adaptive moment estimation (Kingma & Ba, 2015). According to this optimizer, one negative item j is randomly generated for each time step in each sequence in each epoch.

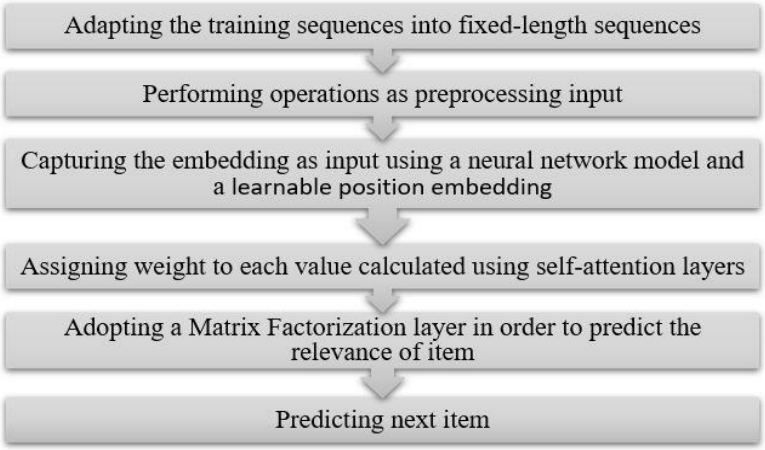

Fig. 6. The overall workflow of next-item recommendation using attention-based neural network model

Finally, the workflow of the presented method in Fig. 6 is prepared.

4. EXPERIMENTS

Before showing the experimental results, let us give an overview of the datasets used in our experiments. We train the applied models for 100 epochs. ADAM optimizer is applied to optimize the loss function, with a learning rate of 0.001.

4.1. Datasets

In this section to evaluate our method, we apply three datasets from three real-world domains. The related datasets vary according to the domains and platforms:

- Amazon (McAuley et al., 2015): This dataset comprises Amazon product reviews crawled from different product categories from Amazon.com from May 1996 to July 2014. Therefore, the amazon dataset is notable for its sparsity as well as variability. We selected to apply the beauty category.
- Steam (Pathak et al., 2017): This dataset was crawled from a large online video game distribution platform. In this regard, the steam dataset contains more than 2,500,000 users, 15,000 games, 7,700,000 English reviews. Other information such as pricing information, users' play hours, category and media score can be applying in the future.
- Digikala: Digikala is the biggest e-commerce startup in Iran. This data contains over 100,000 customer purchases that are anonymized like another digikala data to protect customer privacy.

4.2. Evaluation Metrics

In order to assess the proposed approach, we choose two popular Top-N metrics Hit@ N and the Normalized Discounted Cumulative Gain (nDCG@ N). While Hit metric counts the fraction of times that the ground-truth next item is among the top N items, NDCG metric assigns larger weights on higher positions.

The Discounted Cumulative Gain (DCG) of a retrieval system retrieving N similar images is defined as:

$$DCG = \sum_{n=1}^{\mathcal{K}} \frac{2^{r_n} - 1}{\log(n+1)} \tag{13}$$

Here r_n is the graded relevance of the retrieved image at position-n which is related to the query image. The number of common positive labels which are common on the images makes the graded relevance. the ratio of the DCG over the DCG achieved by a perfect retrieval system given the database defines the NDCG, as follows:

$$nDCG = \frac{DCG}{DCG_{ideal}} \tag{14}$$

100 negative items are sampled arbitrarily and items with the ground-truth item for each user u are ranked in this method in order to prevent high computation on all of the user-item pairs. Eventually using the following metrics, the rankings of all 101 items can be assessed.

4.3. Methods of secondary Competitors

We apply the following representative start-of-the-art approaches as the baselines for the experiments (Wang et al., 2018; Zhou et al., 2018).

- **Translation-based Recommendation (TransRec):**
  An advanced first-order next-item recommendation method that models each user as a translation vector to capture the transition from the current item to the next.
- **Factorized Personalized Markov Chains (FPMC):**
  FPMC uses users' preferences and item-to-item transitions applying a combination of matrix factorization and factorized first-order MCs as its recommendation.
- **Bayesian Personalized Ranking (BPR):**
  BPR is defined as a fundamental recommendation as well as a classic method for learning personalized rankings.
- **Factorized Markov Chains (FMC):**
  A first-order MC method factorizes an item transition matrix applying two item embeddings. After that according to the last item, this method generates recommendations.
- **Multi-Order Attentive Rank (MARank):**
  MARank is an advanced model that uses the most recent items and applies multi-order attention to take individual and union-level historical interactions. Among the baseline methods, MARank has state-of-the-art performance compared with other related methods.
- **Convolutional Sequence Embeddings (Caser):**
  Caser uses convolutional operations on the most recent items and captures high-order MCs. Caser is one of the state-of-the-art next-item recommendation methods.

4.4. Parameter setting

To analyze the importance of various aspects and evaluate many components in our work, we introduce the variants and analyze their effect respectively:

- **Models:**
  To get the best result, several models such as attention model as well as neural network model are used. With attention to all of the models, the Attention-CNN model has the best result among all models (Table 1). Also, in work (Vaswani et al., 2017) have proven attention model outperforms other approaches.
- **Remove Dropout:**
  Dropout evidence that it can regularize the model to get better the performance of the proposed approach. Also, the experimental results signify the overfitting problem is less severe on datasets.
- **Multi-head:**
  it is beneficial to use 'multi-head' attention, which employs attention in h subspaces. However, the prediction performance with two heads is slightly and habitually worse than single-head attention in our case. This might owe in order to the small d in our problem, which is not appropriate for decomposition into smaller subspaces.
- **Number of blocks:**
  The model with one block carries out reasonably well, although using two blocks still boosts the performance of the proposed approach. Specifically, on dense datasets, meaning that the hierarchical self-attention structure is promising to learn more complex item transitions. Using three blocks achieves similar performance to the default model.

Table 1. Performance Comparison of the proposed method according to different models on the steam dataset

Method	NDCG@ 10	Hit@ 10
RNN-Attention-CNN	0.437	0.719
BiLSTM-Attention-LSTM	0.548	0.820
BiLSTM-Attention-CNN	0.550	0.814
Attention-LSTM	0.626	0.862
Attention-CNN	**0.640**	**0.895**

4.5. Recommendation Performance and Evaluation

Table 2 demonstrates the superior performance of all methods on the datasets. By considering the best methods across all datasets, a popular pattern accomplishes better on datasets.

According to Table 2, our proposed model gains the low Hit Rate and HDCG in the digikala dataset against other datasets. The reason is that in the recommendation systems the sequential data is crucial, while the digikala dataset isn't sequential enough. So, the recommendation performance gains the low Hit Rate and HDCG in the digikala dataset. Our method outperforms all baselines on datasets and gains 3.4% and 1.1% Hit Rate and 5.7% and 1.2% NDCG improvements against the strongest baseline which means MARank. One possible reason is that our work can adaptively consider items within various ranges. Also, a novel self-attention-based sequential model using other models can uncover complex relationships leading for a proficient features representation.

Table 2. Performance Comparison of the proposed method according to Hit@10 and NDCG@10

Dataset	Metric	Amazon	Steam	Digikala
FMC	Hit@10 NDCG@10	0.377 0.247	0.773 0.519	-
TransRec (He et al., 2017)	Hit@10 NDCG@10	0.460 0.302	0.762 0.485	-
FPMC (Rendle et al., 2010)	Hit@10 NDCG@10	0.431 0.289	0.771 0.501	-
BPR (Rendle et al., 2009)	Hit@10 NDCG@10	0.377 0.218	0.706 0.443	-
Caser (Tang & Wang, 2018)	Hit@10 NDCG@10	0.426 0.254	0.787 0.538	-
MARank (Yu et al., 2019)	Hit@10 NDCG@10	0.48۷ ۰.32۲	0.861 0.583	-
Our best model	**Hit@10** **NDCG@10**	**0.498** **0.334**	**0.895** **0.640**	**0.408** **0.318**

5. CONCLUSION

This study demonstrates a next-item framework, which is evaluated on recommendation tasks. The framework of the proposed algorithm refers to the entire user sequences and consumed items for predicting over several models from neural networks. Moreover, this research chooses and evaluates different neural network models considering long-range dependencies and complex relationships and selects the best model according to the self-attention-based network performance. Besides, the proposed method applies relevant items from a

user's action history to predict the next item. Extensive empirical results on datasets illustrate that our model outperforms state-of-the-art baselines and is an order of magnitude faster than other approaches.

In the future, we plan to expand the model by incorporating rich context information such as images and videos and research approaches to deal with very long sequences. Another plan can be using ensemble models to achieve a better prediction performance for the proposed approach.

REFERENCES

Alom, M.Z., Taha, T.M., Yakopcic, C., Westberg, S., Sidike, P., Nasrin, M.S., Hasan, M., Van Essen, B.C., Awwal, A.A.S., & Asari, V.K. (2019). A State-of-the-Art Survey on Deep Learning Theory and Architectures. *Electronics*, 8(3), 292.

Cho, Y.H., Kim, J.K., & Kim, S.H. (2002). A Personalized Recommender System Based on Web Usage Mining and Decision Tree Induction. *Expert Systems with Applications*, 23(3), 329-342.

Cui, Z., Ke, R., Pu, Z., & Wang, Y. (2018). *Deep Bidirectional and Unidirectional LSTM Recurrent Neural Network for Network-Wide Traffic Speed Prediction*, Retrieved from https://arxiv.org/abs/1801.02143.

Fang, H. (2020). Deep Learning for Sequential Recommendation: Algorithms, Influential Factors, and Evaluations. *ACM Transactions on Information Systems*, 39(1), 1-41.

Fayyaz, Z., Ebrahimian, M., Nawara, D., Ibrahim, A., & Kashef, R. (2020). Recommendation Systems: Algorithms, Challenges, Metrics, and Business Opportunities. *Applied Sciences*, 10 (21), 1–20.

Gao, C., Lei, W., He, X., Rijke, M.D., & Chua, T. (2021). Advances and Challenges in Conversational Recommender Systems: A Survey. *AI Open*, 2(1), 100–126.

He, R., Kang, W.C., & McAuley, J. (2017). Translation-Based Recommendation. *RecSys 2017 - Proceedings of the 11th ACM Conference on Recommender Systems*, Italy.

Hernández, A., & Amigó, J.M. (2021). Attention Mechanisms and Their Applications to Complex Systems. *Entropy*, 23(3), 1-283.

Sepp, H., & Jürgen, S. (1997). Long Short-Term Memory. *Neural Computation*, 9(8), 1735–80.

Hu, Y., Volinsky, C., & Koren, Y. (2008). Collaborative Filtering for Implicit Feedback Datasets. *Proceedings - IEEE International Conference on Data Mining*, USA.

Khoali, M., Tali, A., & Laaziz, Y. (2021). A Survey of Artificial Intelligence-Based e-Commerce Recommendation System. *Advances in Science, Technology and Innovation* (1nd Edition). USA:Springer.

Kingma, D.P., & Ba, J.L. (2015). Adam: A Method for Stochastic Optimization. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, USA.

Koren, Y. (2009). Collaborative Filtering with Temporal Dynamics. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, France.

Li, J., Wang, Y., & McAuley, J. (2020). Time Interval Aware Self-Attention for Sequential Recommendation. *WSDM 2020 - Proceedings of the 13th International Conference on Web Search and Data Mining*, USA.

Liu, S., Ni'mah, I., Menkovski, V., Mocanu, D.C., & Pechenizkiy, M. (2021). Efficient and Effective Training of Sparse Recurrent Neural Networks. *Neural Computing and Applications*, 33(15), 25–36.

McAuley, J., Targett, C., Shi, Q., & Hengel, A.V.D. (2015). Image-Based Recommendations on Styles and Substitutes. *SIGIR 2015 - Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, Chile.

Minaee, S., Boykov, Y., Porikli, F., Plaza, A., Kehtarnavaz, N., & Terzopoulos, D. (2020). *Image Segmentation Using Deep Learning: A Survey*, Retrieved from https://arxiv.org/abs/2001.05566.

Pathak, A., Gupta, K., & Mcauley, J. (2017). Generating and Personalizing Bundle Recommendations on Steam, *40th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Japan.

Quadranamassimo, C., & Dietmar, J. (2018). Sequence-Aware Recommender Systems. *ACM Computing Surveys*, 51(4), 1-36.

Rendle, S., Freudenthaler, C., Gantner, Z., & Schmidt-Thieme, L. (2009). BPR: Bayesian Personalized Ranking from Implicit Feedback. *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, Canada.

Rendle, S., Freudenthaler, C., & Schmidt-Thieme, L. (2010). Factorizing Personalized Markov Chains for Next-Basket Recommendation. *Proceedings of the 19th International Conference on World Wide Web*, USA.

Sun, F., Liu, J., Wu, J., Pei, C., Lin, X., Ou, W., & Jiang, P. (2019). Bert4rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. *International Conference on Information and Knowledge Management, Proceedings*, Ireland.

Tang, J., & Wang, K. (2018). Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding. *Proceedings of the 11th ACM International Conference on Web Search and Data Mining*, USA.

Tanjim, M.M., Su, C., Benjamin, E., Hu, D., Hong, L., & McAuley, J. (2020). Attentive Sequential Models of Latent Intent for Next Item Recommendation. *The Web Conference 2020 - Proceedings of the World Wide Web Conference*, USA.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., & Polosukhin, I. (2017). Attention Is All You Need. *Advances in Neural Information Processing Systems*, USA.

Wang, S., Hu, L., Cao, L., Huang, X., Lian, D., & Liu, W. (2018). Attention-Based Transactional Context Embedding for next-Item Recommendation. *32nd AAAI Conference on Artificial Intelligence*, USA.

Warde-Farley, D., Goodfellow, I.J., Courville, A., & Bengio, Y. (2014). An Empirical Analysis of Dropout in Piecewise Linear Networks. *2nd International Conference on Learning Representations*, Canada.

Yang, K., Huang, Z., Wang, X., & Li, X. (2019). A Blind Spectrum Sensing Method Based on Deep Learning. *Sensors, 19*(10), 22-70.

Yap, G., Li, X., & Yu, P.S. (2012). Effective Next-Items Recommendation via Personalized Sequential Pattern Mining. *International Conference on Database Systems for Advanced Applications*, South Korea.

Ying, H., Zhuang, F., Zhang, F., Liu, Y., Xu, G., Xie, X., Xiong, H., & Wu, J. (2018). Sequential Recommender System Based on Hierarchical Attention Network. *IJCAI International Joint Conference on Artificial Intelligence*, Sweden.

Yu, L., Zhang, C., Liang, S., & Zhang, X. (2019). Multi-Order Attentive Ranking Model for Sequential Recommendation. *Proceedings*

of the AAAI Conference on Artificial Intelligence, 33(1), 09–16.

Zhang, S., Yao, L., Sun, A., & Tay, Y. (2019). Deep Learning Based Recommender System: A Survey and New Perspectives. *ACM Computing Surveys*, 25(1), 1-38.

Zhou, C., Bai, J., Song, J., Liu, X., Zhao, Z., Chen, X., & Gao, Z. (2018). ATRank: An Attention-Based User Behavior Modeling Framework for Recommendation. *32nd AAAI Conference on Artificial Intelligence*, USA.