Integrating Keywords into BERT4Rec for Sequential Recommendation

Elisabeth Fischer(✉), Daniel Zoller, Alexander Dallmann, and Andreas Hotho

Data Science Chair, Julius-Maximilians University Würzburg, Würzburg, Germany
{elisabeth.fischer,zoller,dallmann,hotho}@informatik.uni-wuerzburg.de

Abstract. A crucial part of recommender systems is to model the user's preference based on her previous interactions. Different neural networks (e.g., Recurrent Neural Networks), that predict the next item solely based on the sequence of interactions have been successfully applied to sequential recommendation. Recently, BERT4Rec has been proposed, which adapts the BERT architecture based on the Transformer model and training methods used in the Neural Language Modeling community to this task. However, BERT4Rec still only relies on item identifiers to model the user preference, ignoring other sources of information. Therefore, as a first step to include additional information, we propose KeBERT4Rec, a modification of BERT4Rec, which utilizes keyword descriptions of items. We compare two variants for adding keywords to the model on two datasets, a Movielens dataset and a dataset of an online fashion store. First results show that both versions of our model improves the sequential recommending task compared to BERT4Rec.

Keywords: Sequential recommendation · Bidirectional Transformer · Item recommendation

1 Introduction

The knowledge of a user's preferences is of great interest for a recommender system. With explicit information about the user's interest often missing, the only clue is the history of previous interactions. To model the preference based on a sequence of historic interactions a number of neural network architectures have been developed, for example, Recurrent Neural Networks (RNNs) [5] or Convolutional Neural Networks (CNNs) [8]. Most of the methods so far model the sequence unidirectional, only taking the previous interactions into account at each step. The recently introduced BERT4Rec method [7] overcomes this limitation by using a bidirectional Transformer [2], allowing it to take context from both sides into account. To build a sequential representation the model relies only on the item identifiers. Other information, like keywords describing items, although available, is not used, but could improve the recommendation of next items in the sequence. For example, if a user has viewed the movie "The Lion King", the information that the item is an "animation", a "musical" and

U. Schmid et al. (Eds.): KI 2020, LNAI 12325, pp. 275–282, 2020.
https://doi.org/10.1007/978-3-030-58285-2_23

(not only) for "children", would be helpful to recommend the next item, because it is more likely that she might be interested in "The Jungle Book" than in "IT". Similar, the information that someone clicked on a page showing some "running shoe", is quite useful for recommending other items of interest.

Previous work has shown that including additional information of items in models like RNNs or CNNs can improve the performance of the recommendation model (e.g., [4,10]). Therefore, as a first step to include additional information into the new state-of-the-art model BERT4Rec, we introduce KeBERT4Rec, a modification, that allows to add keywords describing items (e.g., genres of a movie). To that end, we modify the representation of the sequence items encoded by the Transformer. We evaluate our approach on a Movielens dataset, and a new dataset created from real-world clickstreams of a big online fashion store. The two main contributions of this paper are: 1) We propose two different approaches to include keyword descriptions into the sequential recommendation model BERT4Rec. 2) We compare the two options on two real-world datasets. First results on both datasets show, that our approach of integrating keywords improves the sequential recommendation task.

The remainder of this paper is structured as follows: In Sect. 2 we define the task, followed by a description of our approach in Sect. 3. After reviewing related work in Sect. 4, we describe our datasets, and evaluation setup, and report our results in Sect. 5. Finally, we conclude the paper in Sect. 6.

2 Problem Setting

In this paper we tackle the problem of recommending an item for a user based on her previous sequence of interactions (i.e., previous rated movies or previous clicks in an online shop). Following [2], we denote the set of users with $\mathcal{U} = \{u_1, u_2, \ldots, u_{|\mathcal{U}|}\}$, the set of items with $\mathcal{V} = \{v_1, v_2, \ldots, v_{|\mathcal{V}|}\}$ and the list of interactions of user $u \in \mathcal{U}$ with $S_u = \{v_1^u, v_2^u \ldots, v_{n_u}^u\}$, where user u has interacted with item $v_t^u \in \mathcal{V}$ at the relative time step t. Additionally, we have for every item $v \in \mathcal{V}$ a set of keywords $K_v = \{k_1, k_2, \ldots, k_{|K_v|}\}$ describing each item v. We denote with $\mathcal{K}$ the set of all possible keywords. The recommendation task is now to predict, given the history $\mathcal{S}_u$ with the additional meta information $K_{v_t^u}$ for every $v_t^u \in \mathcal{S}_u$, the next item $v_{n_u+1}^u$ in the sequence of the user's interaction.

3 KeBERT4Rec

Our model builds upon the sequential recommendation model BERT4Rec [7], that transfers the idea of the deep bidirectional self-attention model BERT [2], which is used for language modeling, to the sequential recommendation task. The modified model is shown in Fig. 1, which consists of three different layers, like BERT4Rec: (i) an *embedding layer*, that learns a representation of the inputs (i.e., identifier and keywords), and is fed to (ii) a *Transformer layer*, that consists of L Transformer blocks (see [11] for more details) and (iii) a *projection layer*, that projects the learned hidden representation by the previous layer to the item

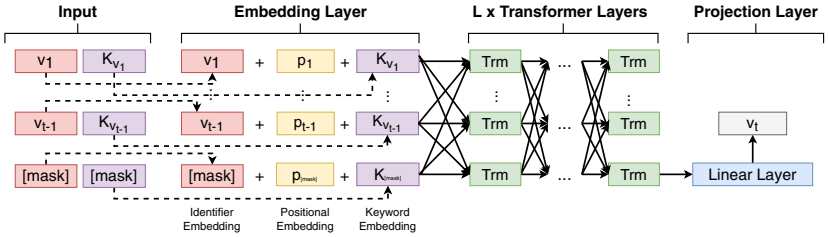


Fig. 1. Model architecture of KeBERT4Rec. In contrast to BERT4Rec, we add an embedding for the keywords of the items and replace the projection layer with a linear layer.

space for prediction using a softmax layer. The Cloze task [9] is used for training, where the model has to predict randomly masked items in the interaction sequence. For evaluation the item to be predicted will be masked. To include keyword descriptions of items as an additional input, we make the following two modifications to the BERT4Rec model:

Embedding Layer: The embedding layer of BERT4Rec, which has a size of d, consists of two different embeddings: (i) an embedding $E_{\mathcal{V}} \in \mathbb{R}^{|\mathcal{V}|\times d}$ of the item identifier and (ii) an auxiliary embedding $E_P \in \mathbb{R}^{N\times d}$ for the position of the items in the sequence, to encode the position for the Transformer blocks, where N is the configurable maximum input sequence length. For every sequence step t, the item embedding $e_t = v_t E_{\mathcal{V}}$ of item v_t and the positional embedding $p_t = tE_P$, the sum $h_t^0 = e_t + p_t$ is used as input for the Transformer layer. Following this idea, we add an additional embedding k_t of the keywords K_{v_t} of item v_t as summand: $h_t^0 = e_t + p_t + k_t$. We propose two different methods to embed multiple keywords into k_t: (i) KE_m merges all keywords of item v_t into a super keyword $K_{v_t}^{u\,*}$ and than embeds this using $E_{\mathcal{K}^*} \in \mathbb{R}^{|\mathcal{K}^*|\times d}$, where $\mathcal{K}^*$ is the set of all possible keyword combinations. (ii) KE_l encodes the categories as a multi-hot vector, which is scaled to the embedding size d using a linear layer. The keyword descriptions are masked accordingly while training and evaluation.

Projection Layer: Given the last hidden state of the L-th Transformer layer h_t^L of the masked item v_t at time step t, BERT4Rec uses a linear layer and the item embedding $E_{\mathcal{V}}$ for projection: $o = \sigma(h_t^L W)E_{\mathcal{V}}^\top$ (bias omitted for readability), where $W \in \mathbb{R}^{|\mathcal{V}|\times d}$ is the weight matrix of the linear layer and σ the GELU activation function [3]. To remove the coupling of the item embedding with the projection layer, we only use a linear layer with parameter matrix $\bar{W} \in \mathbb{R}^{d\times|\mathcal{V}|}$ for projection, $o = h_t^L \bar{W}$, which is also in line with the original BERT model [2].

4 Related Work

Different neural network architectures have been introduced to model the user's interactions for sequential recommendation. These architectures include

Table 1. Statistics of the two preprocessed datasets ML-20m and Fashion.

Dataset	$\lvert\mathcal{U}\rvert$	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{K}\rvert$	#Interactions	Avg.length	Density
ML-20m	138,493	26,744	20	20 m	144,4	0,54%
Fashion	47,158	63,706	301	1.2 m	24.4	0.02%

CNNs [8], RNNs [5], recurrent CNNs [12] and self-attention networks [6]. Recently, [7] introduced BERT4Rec, that adapts the BERT [2] model based on bidirectional Transformers [11], that are currently one of the state-of-the-art architectures for modeling sequences in Natural Language Processing, to the sequential recommendation task. Their method outperforms previous work on four datasets.

Also, modifications to these different neural networks have been proposed to include additional information. For example, [10] adapts CNNs to add textual descriptions of the items using 3D convolutions, or [4] extended the work of [5] by parallel encoding different features (e.g., title, identifier) using different RNNs to improve the recommendation task. A uni-directional Transformer model, that integrates sparse item features, has been presented in [1] to improve the Clickthrough-Rate of an e-commerce online shop.

Instead of using unidirectional models like RNNs, we extend the current state-of-the-art bidirectional model BERT4Rec for sequence recommendation by adding keyword descriptions available for each item into the model. In contrast to [1], we use a bidirectional instead of a unidirectional Transformer and evaluate two different approaches of incorporating item keyword information.

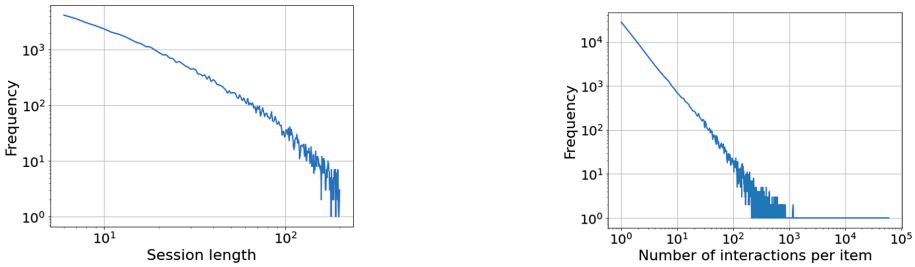
5 Experiments

In this section we introduce the datasets and the setup used in our experiments. At last, we present the results of our evaluation.

5.1 Experimental Setup

Datasets: We evaluate our model on two datasets. As an established dataset for sequential recommendation we use the ML-20m[1] dataset. ML-20m contains movie ratings from an online platform for movie recommendation. We utilize the list of genres of each movie as keyword descriptions. To create interaction sequences, we apply the same preprocess steps as [7]. Our second dataset is from a big online fashion store (Fashion), which consists of user interactions with store pages over the duration of two days. For this dataset we have keywords assigned to each page (e.g., "training pants"). We removed all technical pages (e.g., account pages) and keep only interactions with pages showing one or multiple items. Furthermore, we drop sequences with less than 5 and more than

[1] https://grouplens.org/datasets/movielens/20m/.



(a) Session lengths. (b) Number of item interactions.

Fig. 2. Different frequency distributions for the Fashion dataset.

200 interactions. The resulting frequency distribution of the session length is displayed in Fig. 2a. We observe very few long sessions and an average session length of 24.4 clicks. In Fig. 2b we show the frequency distribution of clicks per items. With most items being rarely visited and only a few frequent items, we only observe a density (avg. number of unique items rated/clicked per user) of 0.02%. In contrast, ML-20m has more ratings per user, but fewer items, so the overall density is a bit higher. Also, a movie can only appear once in a sequence while a page can be visited repeatedly in the Fashion dataset. This happens often, as we treat all paginations of a page as one single page.

Statistics about the two preprocessed datasets are reported in Table 1.

Evaluation Setup: To show that KeBERT4Rec improves the recommendations with the inclusion of keyword descriptions, we compare it with BERT4Rec. For both datasets we used the hyper-parameters reported in [7], and for comparison of the approaches we used the same hyper-parameters for every model.[2] We apply the same evaluation protocol as [7] (i.e., *leave-one-out* evaluation; for more details see [7]) and we use the evaluation metrics *Hit Ratio (HR)* and *Normalized Discounted Cumulative Gain (NDCG)* at various cut-off values k. We apply the Student's t-test to test the statistical significance difference between the results.

Baselines: We also report two baselines in our evaluation: (i) Most-Popular (POP), which recommends items just based on their popularity in the interactions, and (ii) Last-Item (LI), which recommends the previous last item in the sequence. This baseline is only applicable for the Fashion dataset.

5.2 Results

Table 2 shows the recommendation results on our two evaluation datasets.[3] As expected, the performance of POP is far below all other methods on both

[2] We only adapted the batch size to our hardware restrictions and increased the number of epochs for training, because first experiments indicated that our models need more training time. Our code is available at https://dmir.org/KeBERT4Rec.

[3] We train all models on the ML-20m for 200 epochs. Our numbers for BERT4Rec are better than the ones reported in [7], as they train shorter.

Table 2. Results of the two baselines, BERT4Rec and our two versions of KeBERT4Rec on the two evaluation datasets. Both variants of KeBERT4Rec are significantly better than BERT4Rec ($\alpha \leq 0.01$). KE_l marked with * is significant better than KE_m with $\alpha \leq 0.01$ and $^+$ with $\alpha \leq 0.05$.

Dataset	Metric	POP	Bert4Rec	KE_m	KE_l
ML-20m	HR@1	0.022	0.528	0.536	**0.542***
	HR@5	0.081	0.871	0.876	**0.877**$^+$
	HR@10	0.138	0.943	**0.946**	0.945
	NDCG@5	0.051	0.715	0.722	**0.725***
	NDCG@10	0.070	0.739	0.745	**0.747***
Fashion (LI: 0.294)	HR@1	0.029	0.476	0.642	**0.648**$^+$
	HR@5	0.066	0.700	**0.824**	0.823
	HR@10	0.089	0.795	0.871	0.871
	NDCG@5	0.048	0.048	0.741	**0.743***
	NDCG@10	0.056	0.625	0.757	**0.759**$^+$

datasets. The other baseline LI recommends on average about 29% correct on the Fashion dataset. The high HR can be explained by pagination inside the shop. BERT4Rec outperforms the two baselines on Fashion and POP on ML-20m. Both versions of our model KeBERT4Rec achieve better results than BERT4Rec on both datasets, for example, increasing the HR@1 from 0.528 to 0.542 on ML-20m and from 0.476 to 0.648 on Fashion. This proves that including keyword descriptions of items with KeBERT4Rec can improve the sequential recommendation. Moreover, we observe a larger gain on all metrics on the Fashion dataset compared to the ML-20m dataset (on average about 22% vs. 1%). The keywords in the Fashion dataset might be more distinctive, as there are about six times more keywords relative to the number of items. When comparing the variants KE_m and KE_l, we observe, that KE_l outperforms KE_m significantly (only at a level of 0.05 for HR@5) on the ML-20m dataset, except for HR@10, where the difference is not significant. On the Fashion dataset, KE_l is only significantly better than KE_m regarding NDCG@5 (α-level 0.01) and NDCG@5 and HR@1 (α-level 0.05), but regarding the other metrics there is no significant difference.

6 Conclusion

In this paper we introduced KeBERT4Rec, an extension based on BERT4Rec, that includes additional keyword descriptions of items as a first step to integrate additional information about items into BERT4Rec. We evaluated two different approaches to include keywords into the model and compared these with the BERT4Rec model on two datasets. Our evaluation shows that both versions lead to significant improved results in next item recommendation, demonstrating that the inclusion of additional information about the items is a promising way

of improvement. To better understand and improve the model further analysis of the results is needed, especially analyzing the keyword distributions. There are also more options we would like to explore for embedding keywords (e.g., a pre-trained BERT). Data about the items (e.g., title) could also be embedded, requiring an adaption of the proposed model.

References

1. Chen, Q., Zhao, H., Li, W., Huang, P., Ou, W.: Behavior sequence transformer for e-commerce recommendation in Alibaba. In: Proceedings of the 1st International Workshop on Deep Learning Practice for High-Dimensional Sparse Data. ACM, August 2019. https://doi.org/10.1145/3326937.3341261
2. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp. 4171–4186. Association for Computational Linguistics, Minneapolis, Minnesota, June 2019. https://doi.org/10.18653/v1/N19-1423
3. Hendrycks, D., Gimpel, K.: Gaussian error linear units (GELUs) (2016). http://arxiv.org/abs/1606.08415, cite arxiv:1606.08415. Comment: Trimmed version of 2016 draft
4. Hidasi, B., Quadrana, M., Karatzoglou, A., Tikk, D.: Parallel recurrent neural network architectures for feature-rich session-based recommendations. In: Proceedings of the 10th ACM Conference on Recommender Systems. RecSys 2016, pp. 241–248. ACM, New York (2016). https://doi.org/10.1145/2959100.2959167
5. Hidasi, B., Karatzoglou, A., Baltrunas, L., Tikk, D.: Session-based recommendations with recurrent neural networks. In: Bengio, Y., LeCun, Y. (eds.) ICLR (Poster) (2016)
6. Kang, W.C., McAuley, J.: Self-attentive sequential recommendation. In: 2018 IEEE International Conference on Data Mining (ICDM), pp. 197–206. IEEE (2018)
7. Sun, F., Liu, J., Wu, J., Pei, C., Lin, X., Ou, W., Jiang, P.: BERT4Rec: sequential recommendation with bidirectional encoder representations from transformer. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management - CIKM 2019. ACM Press (2019). https://doi.org/10.1145/3357384.3357895
8. Tang, J., Wang, K.: Personalized top-n sequential recommendation via convolutional sequence embedding. In: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining - WSDM 2018. ACM Press (2018). https://doi.org/10.1145/3159652.3159656
9. Taylor, W.L.: "cloze procedure": a new tool for measuring readability. J. Mass Commun. Quart. **30**, 415–433 (1953)
10. Tuan, T.X., Phuong, T.M.: 3D convolutional networks for session-based recommendation with content features. In: Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys 2017, pp. 138–146. ACM, New York (2017). https://doi.org/10.1145/3109859.3109900

11. Vaswani, A., et al.: Attention is all you need. In: Advances in neural information processing systems, pp. 5998–6008 (2017)
12. Xu, C., Zhao, P., Liu, Y., Xu, J., S.Sheng, V.S., Cui, Z., Zhou, X., Xiong, H.: Recurrent convolutional neural network for sequential recommendation. In: The World Wide Web Conference - WWW 2019. ACM Press (2019). https://doi.org/10.1145/3308558.3313408