

Keywords Generation Improves E-Commerce Session-based Recommendation

Yuanxing Liu
Research Center for Social
Computing and Information Retrieval,
Harbin Institute of Technology
yxliu@ir.hit.edu.cn

Zhaochun Ren*
¹ Shandong University
² State Key Laboratory of Cognitive
Intelligence, iFLYTEK, China
zhaochun.ren@sdu.edu.cn

Wei-Nan Zhang*
Research Center for Social
Computing and Information Retrieval,
Harbin Institute of Technology
wnzhang@ir.hit.edu.cn

Wanxiang Che
Research Center for Social
Computing and Information Retrieval,
Harbin Institute of Technology
car@ir.hit.edu.cn

Ting Liu
Research Center for Social
Computing and Information Retrieval,
Harbin Institute of Technology
tliu@ir.hit.edu.cn

Dawei Yin
JD.com
yindawei@acm.org

ABSTRACT

By exploring fine-grained user behaviors, session-based recommendation predicts a user's next action from short-term behavior sessions. Most of previous work learns about a user's implicit behavior by merely taking the last click action as the supervision signal. However, in e-commerce scenarios, large-scale products with elusive click behaviors make such task challenging because of the *low inclusiveness problem*, i.e., many relevant products that satisfy the user's shopping intention are neglected by recommenders. Since similar products with different IDs may share the same intention, we argue that the textual information (e.g., keywords of product titles) from sessions can be used as additional supervision signals to tackle above problem through learning more shared intention within similar products. Therefore, to improve the performance of e-commerce session-based recommendation, we explicitly infer the user's intention by generating keywords entirely from the click sequence in the current session.

In this paper, we propose the *e-commerce session-based recommendation model with keywords generation* (abbreviated as ESRM-KG) to integrate keywords generation into e-commerce session-based recommendation. Specifically, the ESRM-KG model firstly encodes an input action sequence into a high dimensional representation; then it presents a bi-linear decoding scheme to predict the next action in the current session; synchronously, the ESRM-KG model addresses incepts the high dimensional representation of its encoder to generate explainable keywords for the whole session. We carried out extensive experiments in the context of click prediction on a large-scale real-world e-commerce dataset. Our experimental results show that the ESRM-KG model outperforms state-of-the-art baselines with the help of keywords generation.

*Wei-Nan Zhang is the corresponding author. And Zhaochun Ren is the co-corresponding author.

This paper is published under the Creative Commons Attribution 4.0 International (CC-BY 4.0) license. Authors reserve their rights to disseminate the work on their personal and corporate Web sites with the appropriate attribution.
WWW '20, April 20–24, 2020, Taipei, Taiwan
© 2020 IW3C2 (International World Wide Web Conference Committee), published under Creative Commons CC-BY 4.0 License.
ACM ISBN 978-1-4503-7023-3/20/04.
<https://doi.org/10.1145/3366423.3380232>

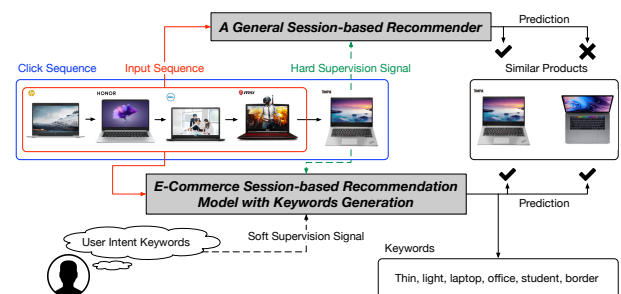


Figure 1: A comparison example between our method and a general session-based recommender. On the top, a general session-based recommender only utilizes the last click of the click sequence to make recommendations. Under such hard supervision shown by the green dotted line, the model learns to recommend the last clicked product, ignoring other similar products that may satisfy the user's needs. On the bottom, our method introduces a soft supervision signal into the recommendation process. Since different but similar products may reflect the same intention, our method is more inclusive for similar products. Moreover, the generated keywords from our method reveal the user's intention.

CCS CONCEPTS

• Information systems → Recommender systems.

KEYWORDS

Session-based recommendation, keywords generation, transformer network

ACM Reference Format:

Yuanxing Liu, Zhaochun Ren, Wei-Nan Zhang, Wanxiang Che, Ting Liu, and Dawei Yin. 2020. Keywords Generation Improves E-Commerce Session-based Recommendation. In *Proceedings of The Web Conference 2020 (WWW '20)*, April 20–24, 2020, Taipei, Taiwan. ACM, New York, NY, USA, 11 pages. <https://doi.org/10.1145/3366423.3380232>

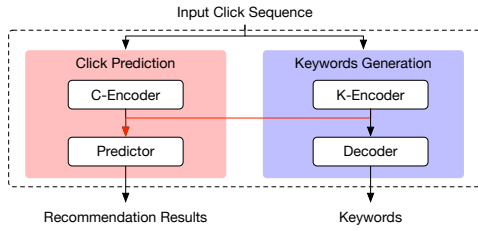


Figure 2: An overview of our method. The red line depicts the way we integrate keywords generation to click prediction.

1 INTRODUCTION

The popularity of e-commerce platforms has successfully enabled explosive changes in people’s online shopping habits¹. Therefore, predicting user intentions is playing an increasingly important role in e-commerce recommender systems [35]. Unfortunately, because of user privacy problem and a large number of anonymous users, e-commerce recommender systems are usually incapable of tracking long-term user intention for a larger fraction of e-commerce users [10]. Moreover, long-term user intention tracking usually generates coarse-grained results, which make the recommendation ambiguous [13, 35]. Consumers usually visit e-commerce sites with a specific short-term intention, which provides clues to track their fine-grained shopping intention [27].

Session-based recommendation predicts a user’s next action based on a short sequence of actions in the ongoing session to track the short-term intention of users. Although many session-based recommendation approaches have been applied for e-commerce recommendation scenarios and show significant progress [9, 13, 16, 18, 26, 32], e-commerce session-based recommendation is still challenging because of the *low inclusiveness problem*. All previous work only uses the target product to supervise the training process. As shown in Figure 1, the model that merely relies on the *hard supervision* signal to recommend the target product, i.e., a ThinkPad laptop, neglects a similar product, i.e., a MacBook Pro, which also satisfies the user’s intention. Therefore, the *hard supervision* signal in the training process hinders the improvement of recommendation performance. Moreover, the user’s click behavior in real-life e-commerce scenarios is complex and elusive, which are easily affected by various recommendations and advertisements in the e-commerce platform. For the lack of supervision of the intention, all previous work suffers from the impact caused by the anomalous click of the implicit click sequence.

To tackle the above problem, we take into account a *soft supervision* signal, i.e., *keywords*, in the current session, and combine them with the hard supervision signal by a multi-task learning mechanism. Specifically, keywords are the top N most common words extracted from the product title set of all products in the current session. Therefore, keywords can explicitly reveal the user’s intention. Then, we utilize keywords as a supervision signal for a sub-task, namely **keywords generation**, which generates keywords entirely based on the click sequence of the current session. Accordingly, by combining e-commerce session-based recommendation with keyword generation in a multi-task learning method, the above

problem in the training process can be relieved. Firstly, keywords are viewed as a soft supervision of products. Specifically, different products (e.g., MacBook Pro and ThinkPad) may provide similar information (i.e., laptop) for keywords generation. Therefore, even suffering from the penalty of the target product, the model may still consider recommending the MacBook Pro to the user with the help of multi-task learning. Secondly, according to the extraction rule of keywords, the information of the accidental click is ignored in the process of keywords generation. Consequently, by a multi-task learning mechanism, the impact of unreliable clicks is also alleviated.

In detail, we propose the e-commerce session-based recommendation model with keywords generation (ESRM-KG). The ESRM-KG model consists of click prediction and keywords generation, as shown with a dotted box in Figure 2. Specifically, the ESRM-KG model firstly encodes an input action sequence into a high dimensional representation; then it presents a bi-linear decoding scheme to predict the next action; synchronously, the ESRM-KG model incepts the high dimensional representation of its encoder to generate explainable keywords for the whole session. The ESRM-KG model is trained by a multi-task learning method with back-propagation in an end-to-end paradigm. Extensive experiments conducted on a large-scale real-world dataset verify the effectiveness of our proposed method. We find our ESRM-KG model significantly outperforms state-of-the-art baselines. Moreover, we also discuss the mechanism of how keywords generation improves the session-based recommender using case studies and error analysis.

To sum up, our main contributions can be summarized as follows: (1) We introduce a soft supervision signal, i.e., keywords generation, into the e-commerce session-based recommendation. (2) We propose a new method, namely ESRM-KG, which provides accurate recommendations while generating keywords. (3) Experimental results show that the ESRM-KG model significantly outperforms state-of-the-art baselines.

2 RELATED WORK

Traditional methods for session-based recommendation can be divided into item-based collaborative filtering and Markov chains models. For item-based methods, Linden et al. [15] propose an item-to-item collaborative filtering method to create a personalized shopping experience for each customer. Sarwar et al. [23] explore item-based techniques and present a new algorithm for collaborative filtering recommender. For sequential models based on Markov-chain adjacent dependency, Shani et al. [24] view the recommendation process as a sequential optimization problem and propose a particular Markov decision processes (MDP) model. Rendle et al. [20] propose factorized personalized Markov chain model which is the combination of a simple Markov chain and the standard matrix factorization. However, a major issue with applying Markov chains in the session-based recommendation is that the state space easily becomes unmanageable[13].

Recently, deep neural networks have been applied to the session-based recommendation [22, 33]. Salakhutdinov et al. [22] first propose to use the restricted Boltzmann machines (RBM) to model extensive tabular data. Zhang et al. [33] use recurrent neural networks (RNN) to model the dependency on the user’s sequential

¹https://www.bigcommerce.com/blog/ecommerce-trends/#cmtoc_anchor_id_0

behaviors. Wang et al. [30] introduce a hierarchical representation model (HRM), which can involve transactions and user representations in prediction to capture both sequential behavior and users' general taste. The gated recurrent unit (GRU) has also been successfully applied to session-based recommendation task. Hidasi et al. [9] propose a GRU-based method with a pairwise ranking loss for the session-based recommendation. Afterwards, Tan et al. [26] present proper data augmentation techniques and accounting for temporal shifts in user behavior to enhance the performance of recurrent models for the session-based recommendation. Quadrana et al. [17] propose a hierarchical RNN model to address the problem of personalizing session-based recommendation. Other similar approaches include a heuristic-based nearest neighbor (kNN) scheme for sessions [10], and a 3D CNN model which combines session clicks and content features [28]. Li et al. [13] apply an attention mechanism to obtain the user's main intentions, and combined with the global representation of the GRU to improve the prediction performance. To emphasize the influence of the last click in the session, Liu et al. [16] propose an attention mechanism to capture both of the long-term and short-term interests of a session. Hidasi and Karatzoglou [8] propose a new class of loss functions combined with modified sampling strategies to improve the performance of session-based recommendations. Wu et al. [32] view session as a graph and use GNN to take complex transitions of items into account. There are also some recent studies from other perspectives but less relevant to our work, such as RepeatNet[18], which focus on repeat consumption and incorporate a repeat-explore mechanism to address the repeat consumption.

Although the above previous work has achieved significant improvement, it only considers the last click of the implicit click, and the low inclusiveness issue still plagues the e-commerce session-based recommendation.

3 METHOD

In this section, we propose our e-commerce session-based recommendation model with keywords generation, abbreviated as ESRM-KG. We start by introducing notions and formulate our tasks at §3.1. We then detail components of click prediction (See §3.2) and keywords generation (See §3.3), respectively. In the end, we illustrate the approach of integrating keywords generation to click prediction for session-based recommendation with keywords generation and introduce the training procedure of the ESRM-KG model at §3.4.

3.1 Problem Formulation

3.1.1 Session-based Recommendation. Before detailing our proposed method, we first formulate the research problem in this paper. For the task of session-based recommendation, we assume there exist N items (i.e., products). Let $[x_1^I, \dots, x_T^I]$ be a sequence click session, where $x_i^I (1 \leq i \leq T)$ is the unique id of one clicked item out of a total number of N items. We thus have the input of the session-based recommendation:

$$\mathbf{x} = [x_1^I, \dots, x_{T-1}^I]. \quad (1)$$

The last item of the click session sequence, i.e. x_T^I , is viewed as the ground truth of recommendation result, i.e. y . The task of *session-based recommendation* learns from input \mathbf{x} , and predicts an output

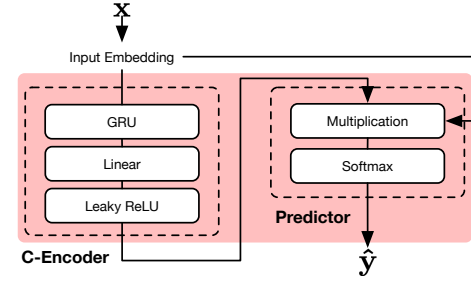


Figure 3: The framework of click prediction, which consists of a C-Encoder and a Predictor. In detail, the C-Encoder aims to learn a representation of sequential behavior. Moreover, the Predictor makes recommendations based on the representation from C-Encoder.

via a model M :

$$\mathbf{y} = M(\mathbf{x}), \quad (2)$$

where $\mathbf{y} = [y_1, \dots, y_N]$ refers to the next click prediction probability of all the items (y_j corresponds to the recommendation score of item j , $1 \leq j \leq N$).

3.1.2 Keywords Generation. Similarly, we can formulate the keywords generation task in the session-based recommendation scenarios. Let $[x_1^E, \dots, x_T^E]$ be a collection of product titles, where $x_i^E (1 \leq i \leq T)$ is the product title of $x_i^I (1 \leq i \leq T)$. By extracting a certain number of words based on the word frequency in $[x_1^E, \dots, x_{T-1}^E]$, we can get keywords, i.e. \mathbf{z} , of the current session, which is viewed as the ground truth of the generation. The task of *keywords generation* learns from input \mathbf{x} , and generates keywords via a model M' :

$$\hat{\mathbf{z}} = M'(\mathbf{x}) \quad (3)$$

where \mathbf{x} is the input sequence and $\hat{\mathbf{z}}$ is the generated keywords.

3.1.3 Session-based Recommendation with Keywords Generation. In this paper, unlike traditional session-based recommendation, we design a model M'' , which not only makes recommendations but also generates keywords, so we combine Equation (2) and Equation (3) into:

$$\mathbf{y}, \hat{\mathbf{z}} = M''(\mathbf{x}), \quad (4)$$

where \mathbf{x} is the input sequence, \mathbf{y} is the recommendation result and $\hat{\mathbf{z}}$ is the generated keywords. Supervised by both \mathbf{z} and the last click x_T^I , M'' outputs the recommendation list \mathbf{y} and the keywords $\hat{\mathbf{z}}$ simultaneously.

3.2 Click Prediction in ESRM-KG

Here we apply a C-Encoder and a Predictor for click prediction, as shown in Figure 3. The C-Encoder learns the representation of the session from the input sequence. Then the Predictor transforms the representation into recommendation results.

3.2.1 C-Encoder. To learn the representation of the session sequence, we use an RNN with Gated Recurrent Units (GRU), which is widely applied in the session-based recommendation for its great performance [8, 9, 13, 18]. Next, the C-Encoder has an additional linear layer after the GRU that defines how to pass the states of

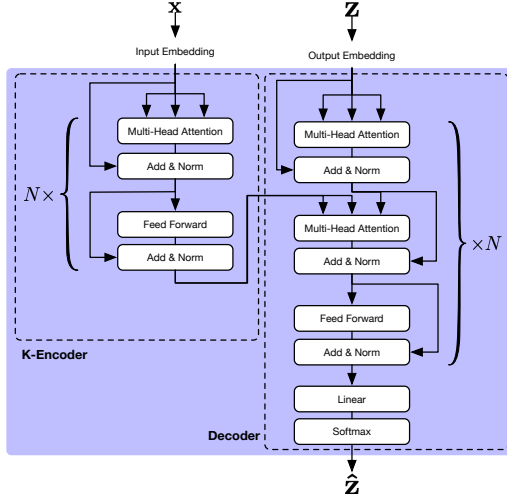


Figure 4: The framework of keywords generation which consists of a K-Encoder and a Decoder. First, the K-Encoder aims to learn long-term dependent information of sequential behavior. Then the Decoder further processes the information and generates keywords.

GRU to the Predictor. Therefore, the C-Encoder formulations are denoted as:

$$\begin{aligned} v_m^g &= \text{GRU}(x) \\ v_{m2}^g &= \text{Linear}(v_m^g) \\ h_t^g &= \text{LeakyReLU}(v_{m2}^g), \end{aligned} \quad (5)$$

where v_m^g and v_{m2}^g are middle variables. The formulations of GRU can be found in [4]. Finally, we view the last layer of h_t^g as the representation c^g of the user's sequential behavior.

3.2.2 Predictor. Same as Li et al. [13], we employ a bi-linear decoding scheme to transform the representation c^g into the items probability matrix. Based on this scheme, the probability of item i can be calculated as:

$$p(\hat{y}_i) = \text{LogSoftmax}(\text{emb}_i^T \mathcal{B} c^g), \quad (6)$$

where \mathcal{B} is a $|\mathcal{E}| \times |\mathcal{D}|$ matrix, $|\mathcal{E}|$ is the dimension of item embedding, $|\mathcal{D}|$ is the hidden state dimension of the C-Encoder, emb_i is the embedding of the item i and LogSoftmax is defined as:

$$\text{LogSoftmax}(x_i) = \log \frac{\exp(x_i)}{\sum_j \exp(x_j)}. \quad (7)$$

Lastly, we train the click prediction with a back-propagation through time (BPTT) method on the negative log-likelihood loss (NLL Loss):

$$\mathcal{L}^e = - \sum_{i=1}^N y_i \log(p(\hat{y}_i)) \quad (8)$$

where N is the number of products, $p(\hat{y}_i)$ is the prediction probability of i -th item and y_i is the ground truth label.

3.3 Keywords Generation in ESRM-KG

For keywords generation, we apply a Transformer, which performs well in text generation tasks [5, 6, 29]. Similar to seq2seq model, the Transformer is composed of a K-Encoder and a Decoder, as shown

in Figure 4. Based on its self-attention mechanism, the K-Encoder can learn long-term dependent information in the current session. Then the Decoder transforms the information into word probability distribution.

3.3.1 K-Encoder. The K-Encoder consists of N identical layers. For the n -th layer, we have:

$$h_n^t, \alpha_n^t = \text{EncoderLayer}(h_{n-1}^t) \quad (9)$$

where h_n^t is the output of the n -th layer and α_n^t is the attention matrix of the n -th layer ($1 \leq n \leq N$). When $n = 1$, the input of the *EncoderLayer* is the embeddings of click sequence x^I . The function *EncoderLayer* is defined as follows:

$$\begin{aligned} v_m^t &= \text{LayerNorm}(h_{n-1}^t) \\ v_c^t, \alpha_n^t &= \text{SelfAttn}(v_m^t, v_m^t, v_m^t) \\ h_n^t &= \text{Dropout}(v_c^t) + h_{n-1}^t \\ h_n^t &= \text{FeedForward}(h_n^t). \end{aligned} \quad (10)$$

where v_m^t and v_c^t are middle variables, *LayerNorm* and *Dropout* refer to [1] and [25] respectively. The formulations of the *SelfAttn* and *FeedForward* can be found in [29]. We also employ a residual connection [7] in the third step of *EncoderLayer* to simplify the optimization.

3.3.2 Decoder. Same to the K-Encoder, the Decoder in keywords generation is also composed of a stack of N identical layers. For the n -th layer, we have:

$$h_n^d, \alpha_n^d = \text{DecoderLayer}(h_{n-1}^d, h_N^t) \quad (11)$$

where h_N^t is the memory bank from the K-Encoder, h_n^d is the output of the n -th layer, and α_n^d is the attention matrix of the n -th layer, $1 \leq n \leq N$. As shown in Figure 4, the Decoder layer is similar to the K-Encoder layer. Comparing with the layer of the K-Encoder, the Decoder layer has one more *SelfAttn* layer, (i.e., masked multi-head attention mechanism). This ensures that the predictions for position i can depend only on the known outputs at positions less than i [29]. Accordingly, the *DecoderLayer* can be defined as follows:

$$\begin{aligned} v_m^d &= \text{LayerNorm}(h_{n-1}^d) \\ v_q^d, \alpha_n^d &= \text{SelfAttn}(v_m^d, v_m^d, v_m^d) \\ v_{q2}^d &= \text{LayerNorm}(\text{Dropout}(v_q^d) + h_{n-1}^d) \\ v_{m2}^d, \alpha_n^d &= \text{SelfAttn}(h_N^t, h_N^t, v_{q2}^d) \\ h_n^d &= \text{FeedForward}(\text{Dropout}(v_{m2}^d) + v_{q2}^d). \end{aligned} \quad (12)$$

We also employ residual connections [7] between every two adjacent sub-layers, followed by layer normalization regulation [1]. Then, we utilize h_N^d from the last layer of the Decoder to generate keywords. Eventually, we employ a fully-connected layer to transform the representation from the Decoder to word probability matrix, so we have:

$$p(\hat{z}) = \text{LogSoftmax}(h_N^d W_{gd}^d + b_{gd}^d), \quad (13)$$

where $p(\hat{z})$ is the probability of keywords, W_{gd}^d is the weight matrix and b_{gd}^d is the bias matrix. At the training stage, we use negative

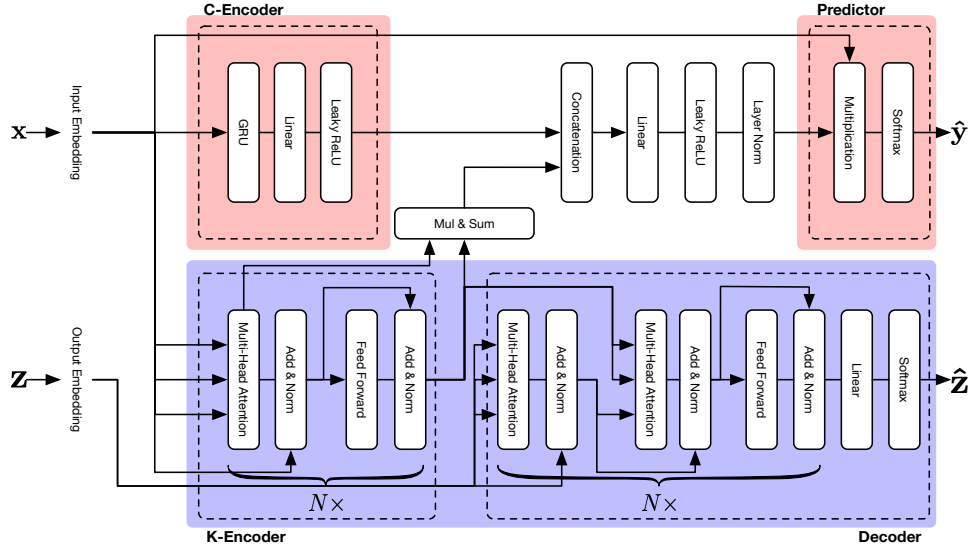


Figure 5: The detailed overview of ESRM-KG framework. The data flow is shown by the arrows.

log-likelihood as the loss function:

$$\mathcal{L}^d = - \sum_{j=1}^{N_z} \sum_{i=1}^{N_V} z_j^i \log(p(\hat{z}_j^i)) \quad (14)$$

where N_z is the length of keywords z and N_V is the word vocabulary size.

3.4 Integration of Keywords Generation and Click Prediction

We have detailed the click prediction and the keywords generation in ESRM-KG, respectively. Here, we illustrate the approach of integrating the keywords generation to the click prediction.

3.4.1 Hybrid Representations. Inspired by Li et al. [13], we explore hybrid representations of the current session to make more precise recommendations. Because the C-Encoder in click prediction can learn the representation of sequential behavior and the K-Encoder in keywords generation does well in capturing the attention information about the current session. Therefore, by concatenating the C-Encoder and the K-Encoder, we can build hybrid representations.

In detail, we use the h_N^t and α_N^t of the K-Encoder to build the representation, where the dimension of h_N^t is $|S|^*|D|$ and the dimension of α_N^t is $|S|^*|S|$. $|S|$ is the length of the input sequence and $|D|$ is the hidden state dimension of the encoder. Thus, we can get a representation by the following equation,

$$c^t = W_S \alpha_N^t h_N^t \quad (15)$$

where the dimension of a transformation matrix W_S is $1 * |S|$ and all values of W_S are $\frac{1}{|S|}$. Then by concatenating the representation c^g of the C-Encoder and the representation c^t , we have hybrid representations c^h :

$$c^h = [c^g; c^t]. \quad (16)$$

Next, we add additional bridge layers to fine-tune the hybrid representations, denoted as:

$$\begin{aligned} v_m^h &= \text{Linear}(c^h) \\ v_{m2}^h &= \text{LeakyReLU}(v_m^h) \\ c^h &= \text{LayerNorm}(v_{m2}^h), \end{aligned} \quad (17)$$

where v_m^h and v_{m2}^h are middle variables. Then, we replace c^g with c^h in the equation 6:

$$P'(item_i) = \text{LogSoftmax}(emb_i^T \mathcal{B} c^h). \quad (18)$$

3.4.2 Multi-task Learning. Multi-task learning is widely applied to leverage useful information contained in multiple related tasks [34]. Since hybrid representations can be viewed as a way of hard parameter sharing in multi-task learning methods [2, 21], thus we use a multi-task learning framework to train the click prediction and the keywords generation components. The hybrid loss function is denoted as:

$$\mathcal{L} = \lambda_e \mathcal{L}^e + (1 - \lambda_e) \mathcal{L}^d \quad (19)$$

where \mathcal{L}^e is the click prediction loss in Equation (8), \mathcal{L}^d is the loss of keywords generation in Equation (14). λ_e is the weight of the click prediction task, where $\lambda_e \in (0, 1)$. Noted that for the ESRM-KG model, we use the prediction probability matrix from Equation (18) to calculate \mathcal{L}^e . Thus, the whole model can be efficiently trained using back-propagation.

The detailed overview of the ESRM-KG framework is shown in Figure 5. At the beginning, the item embeddings are randomly initialized. During the training stage, the C-Encoder and the K-Encoder encode the input embeddings to obtain a global representation of the current session and a transaction representation between two adjacent items separately. Then, these two representations are concatenated as a hybrid representation of current session. Next, the predictor converts the hybrid representation into prediction scores to predict items and calculate the loss. Initialized with the embeddings of ground truth, the Decoder takes the output of

Table 1: Definitions of our collected click data.

Var	Attribute	Example
s_i	Session ID	1-0WklBDRgc2pNcekP-YNQ
s_i^p	Timestamp	1530426235685
x_i^I	Product ID	25050589351
x_i^E	Product Title	Peak basketball shoes, men, 2018 summer new, lightweight, breathable sneakers, fashion, wear-resistant, non-slip sneakers

the K-Encoder as an input to decode the information to generate keywords and computes loss. The losses of the two sub-tasks are summed and then backpropagated to the model to optimize parameters. In the test stage, the process of click prediction is the same as training phase. For keywords generation, since there is no target output embedding, the Decoder is randomly initialized. Besides, the output of the previous step is viewed as the input of the current step to decode the information. Finally, we use a beam search algorithm [12] for decoding and generating the best keywords given a trained model.

4 EXPERIMENTAL SETUP

4.1 Research Questions

We list six research questions that guide the remainder of this paper.

- **RQ1:** Can keywords generation improve the performance of session-based recommendation? What is the overall performance of ESRM-KG in click prediction? (See §5.1).
- **RQ2:** K-Encoder and C-Encoder, which of one plays a vital role in the performance of click prediction? (See §5.2).
- **RQ3:** What is the performance of click prediction under different weights in multi-task learning? Does the performance of click prediction improve as the weight increases? (§5.3).
- **RQ4:** Does the length of the keywords have an impact on the performance of click prediction? Do long keywords achieve better performance? (§5.4).
- **RQ5:** What is the quality of the keywords generated by ESRM-KG? Are the keywords readable? (See §5.5).
- **RQ6:** Does session-based recommendation help keywords generation? (§5.6)

4.2 Dataset

To verify keywords generation improves session-based recommendation and evaluate the performance of the ESRM-KG model in e-commerce session-based recommendation, we collect a large-scale real-world dataset. To start with, we introduce the process of data collection. We collect user click table and product title table on 2018-07-01 from a real-world e-commerce platform. After that, we do an intersection on the two tables to filter clicked items without product title. Next, we use LTP[3], to do the word segmentation for the remaining product titles. Accordingly, we get the processed click data, which includes Session ID, Timestamp, Product ID, and Product Title, as displayed in Table 1.

We then further process the collected click data for experiments. We split all the clicks into sessions by the unique Session ID. Following [13], we also filter out sessions with the session length range

Table 2: Statistics of our collected dataset.

Clicks	Sessions		Items	
	Train	Test	Train	Test
26893361	1217404	37897	108405	59247

of 5 to 20. Then we put sessions in the last two hours as the test set and the rest as the training set. Table 2 provides the detail statistics of our collected dataset for experiments.

4.3 Baseline Methods

4.3.1 Baselines for Click Prediction. To evaluate the performance of click prediction, we compare our model with four widely-applied session-based recommendation methods (i.e., POP, S-POP, Item-KNN, and BPR-MF), four state-of-the-art RNN-based recommendation models (i.e., GRU-Rec, Improved GRU-Rec, NARM, and STAMP) and one GNN-based model (i.e., SR-GNN).

- **POP** recommends items based on overall purchase frequency in the training set.
- **S-POP** recommends the most popular items in the current session.
- **Item-KNN** recommends similar items based on the cosine similarity [23].
- **BPR-MF** applies the stochastic gradient descent method to optimize the pairwise ranking objective function [19].
- **GRURec-TOPK** employs ranking-based loss functions to learn the model to improve the performance of recommendation [8].
- **Improved GRU-Rec** employs data augmentation techniques and accounting for temporal shifts in user behavior to enhance the performance [26].
- **NARM** uses a GRU with attention mechanism to capture the user’s primary intention and a GRU to capture the user’s sequential behavior [13].
- **STAMP** can effectively capture the long-term and short-term interests of a session [16].
- **SR-GNN** considers the complex structure and transitions between items of session sequences and develops a strategy to combine long-term preferences and current interests of sessions to predict next actions of users [32].

4.3.2 Variants of the ESRM-KG Model. To verify the effectiveness of the keywords generation for the session-based recommendation, we compare our model with one variant.

- **ESRM-KG (only Prediction)** means that the ESRM-KG model removes the keywords generation part.

To explore the impact of different encoders on the performance of click prediction, we compare our model with two variants (the *encoder* is either C-Encoder or K-Encoder).

- **ESRM-KG(E=encoder)** means that the Predictor of click prediction only uses the representation from the *encoder* to make recommendations.

To investigate the performance of click prediction under different weights in multi-task learning, we compare our model with five variants (the range of *weight* is 0.1, 0.3, 0.5, 0.7, and 0.9).

Table 3: Recall and MRR values for click prediction. Significant differences are with respect to the NARM.

Methods	Recall@5	MRR@5	Recall@10	MRR@10	Recall@15	MRR@15	Recall@20	MRR@20
POP	0.0103	0.0055	0.0172	0.0065	0.0228	0.0070	0.0279	0.0073
S-POP	0.4431	0.3477	0.4682	0.3520	0.4717	0.3526	0.4742	0.3525
Item-KNN	0.0889	0.0537	0.1389	0.0614	0.1693	0.0640	0.1902	0.0653
BPR-MF	0.0233	0.0194	0.0312	0.0206	0.0374	0.0211	0.0413	0.0214
NARM [13]	0.4432	0.3811	0.4866	0.3869	0.5135	0.3890	0.5321	0.3900
Improved GRU-Rec [26]	0.4101	0.3460	0.4540	0.3519	0.4802	0.3540	0.4989	0.3551
GRURec-TOPK [8]	0.4072	0.2861	0.4562	0.2861	0.4869	0.2861	0.5105	0.2861
STAMP [16]	0.3745	0.3043	0.4149	0.3097	0.4380	0.3115	0.4536	0.3124
SR-GNN [32]	0.4048	0.2965	0.4479	0.2965	0.4724	0.2965	0.4908	0.2965
ESRM-KG (only Prediction)	0.4295	0.3458	0.4754	0.3519	0.5011	0.3539	0.5197	0.3550
ESRM-KG	0.4708[▲]	0.3915[▲]	0.5172[▲]	0.3977[▲]	0.5436[▲]	0.3997[▲]	0.5615[▲]	0.4007[▲]

- **ESRM-KG(W=weight)** means that during the training process of the ESRM-KG model, the loss weight of click prediction is *wegith*.

To explore the impact of keywords length on the performance of click prediction, we compare our model with five variants (the range of *length* is 10, 15, 20, 25, and 30).

- **ESRM-KG(L=length)** means that the ESRM-KG model trained on the top-*length* keywords of the current session.

To study whether the session-based recommendation help keywords generation, we compare our model with one variant.

- **ESRM-KG(only Generation)** means that the ESRM-KG model removes the click prediction part.

For the evaluation of keywords generation, we find that no existing works can generate keywords purely based on the user’s sequential behaviors. Besides, this paper mainly aims to explore the impact of introducing keywords generation to e-commerce session-based recommendations. We show the effectiveness of the generated keywords with a case study and error analysis.

4.4 Evaluation Metrics

We employ two metrics Recall@K and MRR@K to evaluate click prediction. In the experiment, we consider the case of K=5, 10, 15, and 20. **Recall@K** aims to measure the hit of the target item. If the top-k items of the recommendation list contain the target item, the recommendation is viewed as a hit. Then we can get the Recall@K score by the following equation:

$$\text{Recall@K} = \frac{N_h}{N_s} \quad (20)$$

where N_h is the number of hit recommendations, and N_s is the number of all the recommendations. **MRR@K** is the average of reciprocal ranks of the desired items. The reciprocal rank is denoted as:

$$\frac{1}{r} = \begin{cases} \frac{1}{r} & 0 < r \leq K \\ 0 & K < r \end{cases} \quad (21)$$

Thus MRR@K is formulated as follows:

$$\text{MRR@K} = \frac{1}{N_s} \sum_{i=1}^{N_s} \frac{1}{r(i)} \quad (22)$$

To addresses **RQ6**, we apply ROUGE [14], a widely-applied evaluation metric in text summarization, as our evaluation metrics

with standard options. In our experiments, we use ROUGE-1 (R-1), ROUGE-2 (R-2), ROUGE-L (R-L) to evaluate the quality of the generated keywords. It counts the number of overlapping units between the generated keywords \hat{z} and the ground truth keywords z . Assuming that g_n is n-gram, $C(g_n)$ is the number of n-grams in \hat{z} (z or \hat{z}), and $C(g_n)$ is the number of n-grams co-occurring in \hat{z} and z , then the ROUGE-N score for \hat{z} is defined as follows:

$$\text{ROUGE} - N(\hat{z}) = \frac{\sum_{g_n \in z} C_m(g_n)}{\sum_{g_n \in \hat{z}} C_m(g_n)}. \quad (23)$$

When $\hat{z} = z$, we can get ROUGE_{recall} , and when $\hat{z} = \hat{z}$, we get $\text{ROUGE}_{precision}$.

Statistical significance of observed differences between the performance of two runs is tested by using a two-tailed paired t-test and is denoted by using [▲] (or [▼]) for strong significance for $\alpha = 0.01$; or [△] (or [▽]) for weak significance for $\alpha = 0.05$.

4.5 Implementation Details





We implement the ESRM-KG model based on the PyTorch version of OpenNMT [11] with a few modifications². We set both sizes of item embedding and hidden state to 150. In the Transformer and GRU, we use $N = 1$ layer to build our model. We use two heads in the self-attention calculation. The size of the feed-forward network in the transformer is 128. We train our model with 60,000 steps and valid for every 5,000 steps after 30,000 training steps. The batch size of training is 100, and the batch size of validation is 32. We select Adam as optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ for both click prediction and keywords generation. We set the init learning rate is 0.001. After training 30,000 steps, the learning rate will decay half for every 10,000 steps. In the prediction part, we improve the weights of the input clicked items and the last item. The input weight is 10, and the last weight is 15. We set the beam size $\beta = 4$.

5 RESULTS AND DISCUSSIONS

To start with, we analyze the impact of keywords generation on click prediction in §5.1 Next, we discuss the influence of different encoder combinations on click prediction in §5.2. We then explore the effect of different loss weights in multi-task learning on the click prediction (§5.3). Besides, we study the impact of different keywords lengths on click prediction in §5.4. Further, we analyze the quality

²<https://github.com/LeeeeeLiu/ESRM-KG>.

Table 4: An example of input and output of methods. We replace the ID of the product with the image of the product.

Category	Example
Input Click Sequence	
Target	
TOP-20 Prediction of ESRM-KG(only Prediction)	
Rank of Target in the TOP-20 Prediction	7
TOP-20 Prediction of ESRM-KG	
Rank of Target in the TOP-20 Prediction	4
Generated Keywords of ESRM-KG	Games, Computer, DIY, GTX1050Ti-4G, i7, 8700, ASUS, GTX1060-5G, 320G, SSD, Desktop, Monitor,

of the generated keywords in §5.5. Finally, we explore the effect of session-based recommendation on keywords generation in §5.6.

5.1 Influence of Keywords Generation on Click Prediction

5.1.1 Performance of Click Prediction. In order to research RQ1, we evaluate the performance of ESRM-KG on click prediction task to examine if keywords generation could help improve click prediction. Table 3 lists the click prediction performance of all methods in terms of Recall and MRR for a varying range of results. In terms of both Recall and MRR, we find that the ESRM-KG model is always outperforming ESRM-KG (only Prediction) according to various range of results. To evaluate the overall performance of our method, we set NARM as the main baseline because it has the best performance in our experiment. By applying keywords generation to the model, we find our proposed ESRM-KG model outperforms all the baselines in terms of Recall and MRR. Our model achieves a 6.23%, 6.29%, 5.86% and 5.53% increase over NARM in terms of Recall@5, Recall@10, Recall@15, and Recall@20 respectively. In terms of MRR, we find that ESRM-KG also performs quite well. It achieves a 2.73%, 2.79%, 2.75% and 2.74% increase over NARM in terms of MRR@5, MRR@10, MRR@15, and MRR@20 respectively. From above, we can conclude that our model outperforms the state-of-the-art methods significantly. Accordingly, we conclude that keywords generation improves the performance of session-based recommendation.

5.1.2 Embedding Visualization. To further explore the impact of keywords generation, we visualize the product embeddings of the ESRM-KG model and the ESRM-KG (only Prediction). In detail,

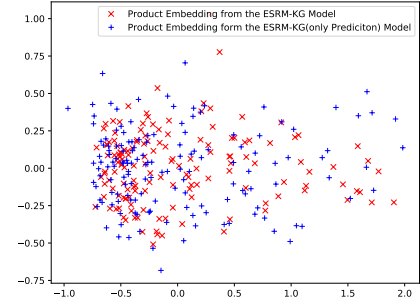


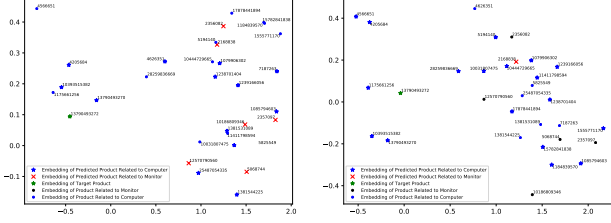
Figure 6: The visualization of embeddings that randomly sampled from the ESRM-KG model and the ESRM-KG (only Prediction) model.

we randomly sample 100 products from the whole products. Then, we use Principal Component Analysis(PCA) [31] to reduce the dimensionality of embedding vectors to 2. From Figure 6, we find that the product embeddings get denser by integrating keywords generation. For two different click sequences that include similar products, e.g., in terms of the product title, the keywords of the two sessions may be similar. Therefore, keyword generation aims to learn the same intention from different inputs, which reduces the gap between products.

5.1.3 Case Study of Input and Output. Here we show an example of the input and output of our methods. In the input click sequence, we notice that most products are related to a class of products(i.e., computers). Therefore, we regard computers as the intention of the current session to drive the following discussion. From Table 4, we can observe that: For lacking additional information, the accidental

Table 5: Recall and MRR values of various encoders in click prediction.

Methods	Recall@5	MRR@5	Recall@10	MRR@10	Recall@15	MRR@15	Recall@20	MRR@20
ESRM-KG(E=C-Encoder)	0.4703	0.3916	0.5147	0.3975	0.5396	0.3995	0.5557	0.4004
ESRM-KG(E=K-Encoder)	0.4652	0.3883	0.5045	0.3936	0.5276	0.3954	0.5434	0.3963
ESRM-KG	0.4708	0.3915	0.5172	0.3977	0.5436	0.3997	0.5615	0.4007



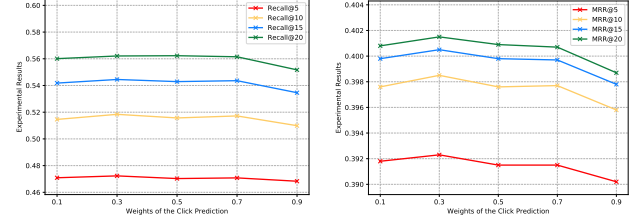
(a) The ESRM-KG (only Prediction) model. (b) The ESRM-KG model.
Figure 7: Embedding visualization of products in the case study for input and output of two models.

click (i.e., the 10th click in the input click sequence) leaves a serious impact on the recommendation. In detail, in the top 20 of the recommendation list provided by the ESRM-KG (only Prediction) model, six products are related to monitors. Under the supervision of keyword generation, the ESRM-KG model learns additional information from the click sequence. Consequently, most of the recommendation results are related to computers. Also, by comparing the rankings of the target product in the recommendation lists of two models, we find that the ESRM-KG model has a higher performance of recommendation. Lastly, from the generated keywords of the ESRM-KG model, we can observe that the ESRM-KG model performs well in capturing the intention of the current session.

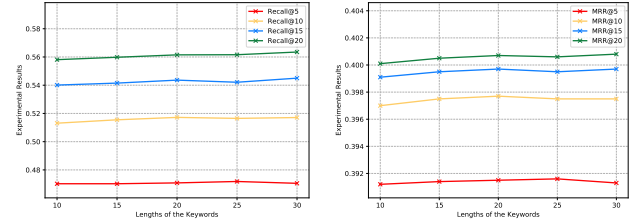
Besides, we show the embedding visualization of products in the above case. As shown in Figure 7(b) and Figure 7(a), we find that the product embeddings produced by the ESRM-KG model are a bit denser than that of the ESRM-KG (only Prediction) model. Based on the above discussions, we conclude that keywords generation helps session-based recommendation.

5.2 Influence of Different Encoders

Next, we turn to RQ2. The structure of the encoder is essential in click prediction. Thus, we compare the performance of different encoder structures. As shown in Table 5, we can see that both C-Encoder and K-Encoder can achieve great results in click prediction. Meanwhile, C-Encoder performs better than K-Encoder. We also find that the ESRM-KG model achieves a 2.53%, 3.03%, and 3.32% increase over the ESRM-KG(E=K-Encoder) model in terms of Recall@10, Recall@15, and Recall@20 respectively. By combining C-Encoder with K-Encoder, both Recall and MRR get improved. We believe that K-Encoder performs well in capturing the attention between clicked items[29]. However, it lacks a global representation of the current session. Therefore, the combination of C-Encoder and K-Encoder can achieve high performance.



(a) Performance on Recall Metric (b) Performance on MRR Metric
Figure 8: Different weights of the click prediction in the training process of the ESRM-KG model.



(a) Performance on Recall Metric (b) Performance on MRR Metric
Figure 9: Different lengths of the keywords as the ground truth in ESRM-KG training process.

5.3 Influence of Different Weights in Multi-task Learning

After that, we consider RQ3. We explore the impact of different weights of click prediction loss on click prediction in multi-task learning (the ESRM-KG model trained on the top-20 keywords of the current session). From Figure 8, we can see that: First, when the weight of click prediction is 0.9, the model gets the worst result (whether on Recall or MRR). We conduct that the low weight of keywords generation leads to the uselessness of supervision provided by the keywords about the current session. Second, different weights (except for 0.9) have a slight impact on click prediction results in terms of Recall. However, in terms of MRR, the model performs the best when the weight is 0.3. We infer that when the click prediction weight is 0.3, the keywords generation can catch the requirements of the session better. Consequently, the ranking of the predicted product is improved.

5.4 Influence of Different Keywords Lengths

To address the research question RQ5, we discuss the effect of different lengths of keywords. TOP-K denotes the ESRM-KG (L=top-k) model. Here the range of K is 10, 15, 20, 25, and 30. As shown in Figure 9, we find TOP-20 outperforms TOP-10 and TOP-30 in terms of both Recall and MRR metrics. However, the results of TOP-10 and TOP-15 are not satisfactory on both Recall and MRR metrics. From this phenomenon, we find that when the length of

Table 6: Four examples of generated keywords.

#	ESRM-KG	Ground Truth
29	Apple, iPhone, 6, 32GB, Gold, Mobile, Unicom, Telecom, 4G, Mobile, 6, 32GB, Gold, 6s, Plus, A1699, 128G, Rose	Apple, iPhone, 6, 32GB, Gold, Mobile, Unicom, Telecom, 4G, Mobile
623	Chandelier, European style, living room, dining room, bedroom, lamp American, country, rural, country, chandelier, luxury, atmosphere, lighting, lamps, packages, restaurant	Chandelier, living room, dining room, bedroom, europe, full, lights, simple, lighting, meet, minus, opp, retro, lighting, modern, countryside, lamps, package, buy, 999
979	Inch, notebook, computer, 8G, 2G, Apple, MacBook, Air, 13.3, silver, 2017, models, Core, i5, processor, 8GB, memory /, 128GB, i5-8250U, 256G, MX150	Inch, notebook, computer, Apple, thin and narrow, border, i5-8250U, 8G, MX150, 2G, IPS, MacBook, Air, 13.3, silver, 2017, models, Core, i5, processor
2486	Beverages, bottles, boxes, containers, Cola, Coca-Cola, soda, carbonates, 330ML, 24, cans, Bulk Pack, the new, old, packing, random, shipping	Qingdao, beer, beverage, TsingTao, classic, degree, 500ml, listening, bottle, 190310, 12, whole, boxed, crafted, mellow, coconut palm juice, positive, scented coconut tree, coconut juice

Table 7: Performance comparison of methods in keywords generation.

Methods	ROUGE-1			ROUGE-2			ROUGE-L		
	P	R	F1	P	R	F1	P	R	F1
ESRM-KG (only Generation)	0.7450	0.6392	0.6777	0.3616	0.3427	0.3465	0.5884	0.5054	0.5191
ESRM-KG	0.7316	0.6352	0.6695	0.3425	0.3327	0.3322	0.5752	0.4995	0.5108

keywords is too small, it may provide insufficient information that will make the improvement quite limited. Also, we can see that TOP-30 achieves the best results on Recall@15, Recall@20, MRR@15, and MRR@20. However, TOP-30 is not as good as TOP-25 on Recall@5 and MRR@5 and is defeated by TOP-20 on Recall@10 and MRR@20. Regarding this phenomenon, we believe that long keywords that may lead to the ESRM-KG model tend to keywords generation. Although the recommendation performance is enhanced in some aspects, e.g., Recall@20, it also faces the problem of the low ranking of the target products in the recommendation list.

5.5 Quality of Generated Keywords

Here we discuss **RQ4**. Here we show four examples of generated keywords in Table 6. Specifically, two of the sentences (i.e., #623 and #979) are generated when the target item is in the top-20 items of the recommendation list, and the remaining two are not. On the one hand, the keywords generated by the ESRM-KG model are of high quality. From sentence #623, we can see the ESRM-KG model performs well in capturing meaningful keywords in the session and the ignoring words that are meaningless, e.g., "buy", "full", and "meet" in the corresponding ground truth. Besides, from #979 we can learn that even the order of the generated keywords is inconsistent with that of keywords in the ground truth, we can still understand the information that keywords want to express. On the other hand, there are also shortcomings in the keywords generation. For example, the ESRM-KG model generates duplicate content, e.g., "Mobile", "6", "32GB" and "Gold" in sentence #29. Moreover, the ESRM-KG model also produces incorrect entities, such as "beer" in the ground truth of sentence #2486 turns into "Coca-Cola". Since the focus of this paper is on exploring the impact of keywords generation on click prediction, we plan to explore how to generate higher quality keywords in future work to address those shortcomings.

5.6 Impact of Recommendation on Keywords Generation

Finally, we address **RQ6**. We compare our ESRM-KG model with ESRM-KG (only Generation). As shown in Table 7, we find that the ESRM-KG (only Generation) model outperforms the ESRM-KG model in terms of all the metrics. From this phenomenon, we believe that session-based recommendation may not be helpful to keywords generation. For the recommendation task, even the predicted product is quite similar to the target product, it will still be considered as a negative example. Therefore, keywords generation may often suffer "noise" from the session-based recommendation, and the performance of keywords generation decreases.

6 CONCLUSION AND FUTURE WORK

In this paper, we have addressed the impact of keywords generation on the session-based recommendation problem in e-commerce. We have proposed a novel hybrid model, namely ESRM-KG, to integrate the keywords generation to the session-based recommendation. Extensive experiments conducted on a large-scale real-world e-commerce dataset evaluate the effectiveness of the ESRM-KG model: First, we have found that keywords generation significantly improves the performance of the next click prediction in e-commerce session-based recommendation. Then, we have compared the performance of different encoder structures. Next, we have explored the impact of different weights of click prediction loss on click prediction in multi-task learning. Also, we have discussed the effect of different lengths of keywords. Moreover, we have shown the advantages and shortcomings of the generated keywords with a case study. Lastly, we have explored the impact of the session-based recommendation on keywords generation.

As to future work, we plan to explore an evaluation for the keywords generation of the ESRM-KG model. Also, we will consider the solution to generate higher quality keywords in future work to address those shortcomings of generated keywords.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their helpful comments. This work is supported by the Natural Science Foundation of China (61902219, 61936010), the Fundamental Research Funds for the Central Universities (30620170037), the Foundation of State Key Laboratory of Cognitive Intelligence, iFLYTEK, P.R.China (COGOSC-20190003), the Tencent AI Lab Rhino-Bird Focused Research Program (JR201932), the Fundamental Research Funds of Shandong University. In addition, we want to acknowledge the Heilongjiang Province Art Planning Project 2019C027 and the Heilongjiang Province Social Science Research Project 18TQB100.

REFERENCES

- [1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450* (2016).
- [2] R Caruna. 1993. Multitask learning: A knowledge-based source of inductive bias. In *ML*. 41–48.
- [3] Wanxiang Che, Zhenghua Li, and Ting Liu. 2010. Ltp: A chinese language technology platform. In *COLING*. 13–16.
- [4] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. In *ACL*.
- [5] Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A Discourse-Aware Attention Model for Abstractive Summarization of Long Documents. In *NAACL*.
- [6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR* abs/1810.04805 (2018). [arXiv:1810.04805](https://arxiv.org/abs/1810.04805) [http://arxiv.org/abs/1810.04805](https://arxiv.org/abs/1810.04805)
- [7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*. 770–778.
- [8] Balázs Hidasi and Alexandros Karatzoglou. 2018. Recurrent neural networks with top-k gains for session-based recommendations. In *CIKM*. ACM, 843–852.
- [9] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based recommendations with recurrent neural networks. In *ICLR*.
- [10] Dietmar Jannach, Malte Ludewig, and Lukas Lerche. 2017. Session-based item recommendation in e-commerce: on short-term intents, reminders, trends and discounts. *User Modeling and User-Adapted Interaction* 27, 3-5 (2017), 351–392.
- [11] Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M. Rush. 2017. OpenNMT: Open-Source Toolkit for Neural Machine Translation. In *ACL*. <https://doi.org/10.18653/v1/P17-4012>
- [12] Philipp Koehn. 2004. Pharaoh: a beam search decoder for phrase-based statistical machine translation models. In *AMTA*.
- [13] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural attentive session-based recommendation. In *CIKM*. ACM, 1419–1428.
- [14] Chin-Yew Lin. 2004. ROUGE: a Package for Automatic Evaluation of Summaries. In *ACL*.
- [15] Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing* 1 (2003), 76–80.
- [16] Qiao Liu, Yifu Zeng, Refuoe Mokhosi, and Haibin Zhang. 2018. STAMP: Short-Term Attention/Memory Priority Model for Session-based Recommendation. In *KDD*.
- [17] Massimo Quadana, Alexandros Karatzoglou, Balázs Hidasi, and Paolo Cremonesi. 2017. Personalizing session-based recommendations with hierarchical recurrent neural networks. In *RecSys*. ACM, 130–137.
- [18] Pengjie Ren, Zhumin Chen, Jing Li, Zhaochun Ren, Jun Ma, and Maarten de Rijke. 2019. RepeatNet: A Repeat Aware Neural Recommendation Machine for Session-based Recommendation. In *AAAI*.
- [19] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI*.
- [20] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing Personalized Markov Chains for Next-basket Recommendation. In *WWW*.
- [21] Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098* (2017).
- [22] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. 2007. Restricted Boltzmann machines for collaborative filtering. In *ML*. ACM, 791–798.
- [23] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *WWW*. ACM, 285–295.
- [24] Guy Shani, David Heckerman, and Ronen I Brafman. 2005. An MDP-based recommender system. *Journal of Machine Learning Research* 6, Sep (2005), 1265–1295.
- [25] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.
- [26] Yong Kiam Tan, Xinling Xu, and Yong Liu. 2016. Improved recurrent neural networks for session-based recommendations. In *DLRS*. ACM, 17–22.
- [27] Maryam Tavakol and Ulf Brefeld. 2014. Factored MDPs for detecting topics of user sessions. In *RecSys*. ACM, 33–40.
- [28] Trinh Xuan Tuan and Tu Minh Phuong. 2017. 3D Convolutional Networks for Session-based Recommendation with Content Features. In *RecSys*. ACM, 138–146.
- [29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *NIPS*. 5998–6008.
- [30] Pengfei Wang, Jiafeng Guo, Yanyan Lan, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2015. Learning hierarchical representation model for nextbasket recommendation. In *SIGIR*. ACM, 403–412.
- [31] Svante Wold, Kim Esbensen, and Paul Geladi. 1987. Principal component analysis. *Chemometrics and intelligent laboratory systems* 2, 1-3 (1987), 37–52.
- [32] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-based Recommendation with Graph Neural Networks. In *AAAI*.
- [33] Yuyu Zhang, Hanjun Dai, Chang Xu, Jun Feng, Taifeng Wang, Jiang Bian, Bin Wang, and Tie-Yan Liu. 2014. Sequential Click Prediction for Sponsored Search with Recurrent Neural Networks.. In *AAAI*, Vol. 14. 1369–1375.
- [34] Yu Zhang and Qiang Yang. 2017. A survey on multi-task learning. *arXiv preprint arXiv:1707.08114* (2017).
- [35] Meizi Zhou, Zhuoye Ding, Jiliang Tang, and Dawei Yin. 2018. Micro behaviors: A new perspective in e-commerce recommender systems. In *WSDM*. ACM, 727–735.