CORE: Simple and Effective Session-based Recommendation within Consistent Representation Space

Yupeng Hou[†]
houyupeng@ruc.edu.cn
Gaoling School of Artificial Intelligence, Renmin
University of China

Binbin Hu
bin.hbb@antfin.com
Ant Group

Zhiqiang Zhang
lingyao.zzq@antfin.com
Ant Group

Wayne Xin Zhao[✉]
batmanfly@gmail.com
Gaoling School of Artificial Intelligence, Renmin
University of China
Beijing Key Laboratory of Big Data Management and
Analysis Methods
Beijing Academy of Artificial Intelligence

ABSTRACT

Session-based Recommendation (SBR) refers to the task of predicting the next item based on short-term user behaviors within an anonymous session. However, session embedding learned by a non-linear encoder is usually not in the same representation space as item embeddings, resulting in the inconsistent prediction issue while recommending items. To address this issue, we propose a simple and effective framework named **CORE**, which can unify the representation space for both the encoding and decoding processes. Firstly, we design a representation-consistent encoder that takes the linear combination of input item embeddings as session embedding, guaranteeing that sessions and items are in the same representation space. Besides, we propose a robust distance measuring method to prevent overfitting of embeddings in the consistent representation space. Extensive experiments conducted on five public real-world datasets demonstrate the effectiveness and efficiency of the proposed method. The code is available at: https://github.com/RUCAIBox/CORE.

CCS CONCEPTS

• **Information systems → Recommender systems**.

KEYWORDS

session-based recommendation

---

† Work done during internship at Ant Group.
✉ Corresponding author.

---

1 INTRODUCTION

Session-based Recommendation (SBR) aims to capture short-term and dynamic user preferences according to user behaviors (*e.g.,* clicked items) within anonymous sessions, which is an important research topic in recommender systems [23]. Recent SBR methods usually follow the encoder-decoder framework and focus on designing effective neural architecture as encoder, such as Recurrent Neural Network (RNN) [8, 13, 17], Transformer [1, 7, 11, 21, 28, 33, 34] and Graph Neural Network (GNN) [3, 16, 26, 29]. As for the decoder, a widely adopted way is to calculate dot product of session and item embeddings as the interaction probability of the next item.

In line with the focus of the above-mentioned efforts, we notice that session embeddings are not usually in the same representation space as item embeddings. Generally speaking, user behaviors within a short session tend to share a common focus [3]. In session-based recommendation, session embeddings are expected to reflect users' short-term preference and should be similar to embeddings of preferred items. However, as shown in Figure 1(a), when item embeddings are encoded by non-linear neural networks, the resulting session embedding doesn't necessarily fall into the space spanned by bases of item embeddings, and thus it may become unable to concisely represent user's preference. For example, as illustrated in Figure 1(b), for sessions with a common objective (item *a* clicked multiple times for simulation), we observe that their embeddings that encoded by non-linear encoder (*e.g.,* GRU4Rec [8]) fall into different points in embedding space, giving inconsistent prediction when calculating similarities to item embeddings.

Considering the above issues, we strive to unify the representation space for both encoding and decoding processes in session-based recommendation. The basic idea is to represent sessions in the item embedding space, for example, by directly summing up item embeddings within the session. However, in this way, we fail to model the sequential nature, which can be captured by neural
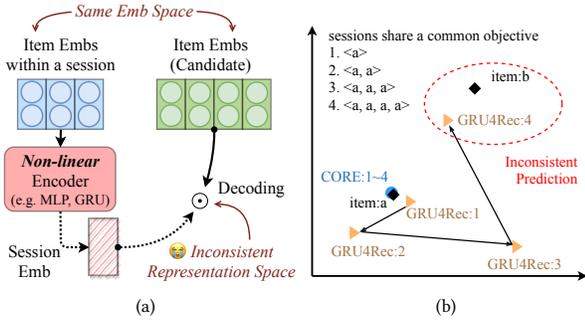
Figure 1: (a) Encoder-decoder framework of most existing session-based recommendation models and (b) Inconsistent prediction issue while measuring the distance between embeddings for recommending.

Figure 2: Overall framework of CORE.

encoders like RNNs. Thus we consider overcoming two key challenges. (1) How to design a more suitable encoder, so that we can take advantage of deep non-linear neural networks' immense capability, given that session and item embeddings share a consistent representation space? (2) Once the representation space is unified and item embeddings are directly involved in score calculation and model optimization, how to measure the distance between embeddings to avoid overfitting of item embeddings?

To this end, we propose a rather simple yet effective framework for session-based recommendation, where session embeddings and item embeddings are in **CO**nsistent **RE**presentation space, namely **CORE**. Firstly, we encode session embeddings as the linear combination of item embeddings within the session, ensuring the encoded embedding has a consistent representation space as item embeddings. The weight of each item in the session is learned via a deep non-linear neural network to incorporate various of inductive biases, such as order and importance of items. Secondly, we revisit the widely adopt dot product distance measuring in the perspective of optimizing tuplet loss and improve the robustness of distance measuring from multiple aspects. Extensive experiments on five public datasets demonstrate the effectiveness and efficiency of the proposed framework. We also show that performances of existing session-based recommendation models can be significantly improved by the proposed techniques.

2 METHODOLOGY

In this section, we introduce the proposed framework CORE, a simple and effective framework for session-based recommendation within consistent representation space. Figure 2 presents the overall architecture of CORE.

Now, we start with a brief description of the typical session-based recommendation, which follows an encoder-decoder framework. Firstly, each item is embedded into an unified embedding space. $h_i = \text{Emb}(v_i) \in \mathbb{R}^d$ denotes the item embedding for item v_i, where $\text{Emb}(\cdot)$ is the item embedding look-up table and d is the dimension of vectors. Then we have $h_s = \text{Encoder}([h_{s,1}, \ldots, h_{s,n}]) \in \mathbb{R}^d$ to encode a session s with n items, where $\text{Encoder}(\cdot)$ is usually a non-linear neural network. Finally, we can predict the probability
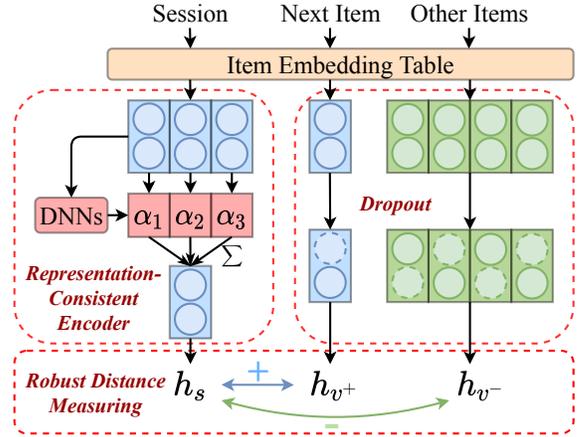
distribution for the next item, *i.e.*, $\hat{y} = \text{Decoder}(h_s) \in \mathbb{R}^m$, where m is the number of all items.

2.1 Representation-Consistent Encoding

Here we aim at encoding sessions into item embedding space to overcome the inconsistent representation space issue. As most existing encoders adopt a non-linear encoder (*e.g.*, RNNs or Transformers) directly stacked over the input item embeddings, the encoded session embeddings are not in the same representation space as items. In order to keep embeddings within the same space, a natural idea is whether we can remove non-linear activation functions over item embeddings and encode sessions as the linear combination of item embeddings.

Along this line, we propose a Representation-Consistent Encoder (RCE), whose output session embedding is the weighted summarization of item embeddings within a session. Linear combination guarantees the session embeddings are always in the same embedding space as items. Although non-linear layers are removed between the encoded session embedding and input item embeddings, they are essential to incorporating inductive biases and learning weights for input item embeddings. Formally, we apply arbitrary deep neural networks to learn weights for each item embedding in a session,

$$\boldsymbol{\alpha} = \text{DNN}([h_{s,1}; h_{s,2}; \ldots; h_{s,n}]), \tag{1}$$

$$h_s = \sum_{i=1}^{n} \alpha_i h_{s,i}, \tag{2}$$

Then we give two detailed implementations of DNN without carefully designed architecture.

2.1.1 Learning Weights via Mean Pooling. This variant adopts a mean pooling layer as DNNs, *i.e.*, $\alpha_i = \frac{1}{n}$. This variant ignores the order of items in a session, as well as the importance of each item.

2.1.2 Learning Weights via Transformer. In this variant, we utilize L-layers self-attention blocks like SASRec [11] as DNNs.

$$F = \text{Transformers}([h_{s,1}; h_{s,2}; \ldots; h_{s,n}]), \tag{3}$$

Table 1: Overall performance comparison on five datasets. "∗" indicates the statistical significance for $p < 0.01$ compared to the best baseline method with paired t-test. Sessions are split into train/validation/test set in a ratio of 8:1:1 for fair evaluation. We indicate performances of FPMC on Yoochoose as "−" due to the OOM issue.

Dataset	Metric	FPMC	GRU4Rec	NARM	SR-GNN	NISER+	LESSR	SGNN-HN	SASRec	GC-SAN	CL4Rec	CORE-ave	CORE-trm	Improv.
Diginetica	R@20	31.83	45.43	47.68	48.76	_51.23_	48.80	50.89	49.86	50.95	50.03	50.21	**52.89***	+3.24%
	M@20	8.79	14.77	15.58	16.93	_18.32_	16.96	17.25	17.19	17.84	17.26	18.07	**18.58***	+1.42%
Nowplaying	R@20	10.18	13.80	14.17	15.28	16.55	17.60	16.75	_20.69_	18.30	20.59	20.31	**21.81***	+5.41%
	M@20	4.51	5.83	6.11	6.10	7.14	7.13	6.13	**8.14**	_8.13_	8.21	6.62	7.35	−
RetailRocket	R@20	46.04	55.32	58.65	58.71	_60.36_	56.22	58.82	59.81	60.18	59.69	59.18	**61.85***	+2.47%
	M@20	21.95	33.18	34.69	36.42	_37.43_	37.11	35.72	36.03	36.85	35.95	_37.52*_	**38.76***	+3.55%
Tmall	R@20	20.30	23.25	31.67	33.65	35.97	32.45	39.14	35.82	35.32	35.59	**44.67***	_44.48*_	+14.13%
	M@20	13.07	15.78	21.83	25.27	27.06	23.96	23.46	25.10	23.48	25.07	**31.85***	_31.72*_	+17.70%
Yoochoose	R@20	−	60.78	61.67	61.84	62.99	62.89	62.49	63.55	63.24	_63.61_	58.83	**64.61***	+1.57%
	M@20	−	27.27	27.82	28.15	_28.98_	28.59	28.24	28.63	**29.00**	28.73	25.05	28.24	−

where $F \in \mathbb{R}^{n \times d'}$ and d' is the output dimension of feed forward network of the last layer of self-attention blocks. Then we can obtain the normalized weights $\boldsymbol{\alpha} \in \mathbb{R}^n$,

$$\boldsymbol{\alpha} = \text{softmax}(\boldsymbol{w} \cdot \boldsymbol{F}^\top), \quad (4)$$

where $\boldsymbol{w} \in \mathbb{R}^{d'}$ are learnable parameters. This variant captures the sequential nature via positional encoding technique in Transformer.

2.2 Robust Distance Measuring for Decoding

As sessions are encoded as linear combination of item embeddings and decoded by measuring the distances to items in embedding space, item embeddings are directly involved in the distance calculation between embeddings, leading to a high risk of overfitting. Thus we seek a robust way to measure the distance in the unified representation space to prevent overfitting. By reviewing the widely adopted dot product distance, we have the following lemma:

LEMMA 1. *Given a session embedding $\boldsymbol{h}_s$ and item embeddings $\{\boldsymbol{h}_v | v \in \mathcal{V}\}$, when dot product is used to measure the embedding distance, optimizing cross entropy loss is approximately proportional to optimize $(N-1)$-tuplet loss [19] with a fixed margin of 2.*

PROOF. The lemma can be proved by rewriting the original loss function as,

$$
\begin{aligned}
\ell_{\text{ori}} &= -\log \frac{\exp(\boldsymbol{h}_s \cdot \boldsymbol{h}_{v^+})}{\sum_{i=1}^m \exp(\boldsymbol{h}_s \cdot \boldsymbol{h}_{v_i})} \\
&= \log \left[1 + (|\mathcal{V}| - 1) \sum_{v^- \in \mathcal{V} \setminus \{v^+\}} \exp(\boldsymbol{h}_s \boldsymbol{h}_{v^-} - \boldsymbol{h}_s \boldsymbol{h}_{v^+}) \right], \\
&\simeq (|\mathcal{V}| - 1) \sum_{v^- \in \mathcal{V} \setminus \{v^+\}} \exp(\boldsymbol{h}_s \boldsymbol{h}_{v^-} - \boldsymbol{h}_s \boldsymbol{h}_{v^+}) \\
&\simeq (|\mathcal{V}| - 1) \sum_{v^- \in \mathcal{V} \setminus \{v^+\}} (\boldsymbol{h}_s \boldsymbol{h}_{v^-} - \boldsymbol{h}_s \boldsymbol{h}_{v^+} + 1), \\
&\propto \sum_{v^- \in \mathcal{V} \setminus \{v^+\}} \left(\|\boldsymbol{h}_s - \boldsymbol{h}_{v^+}\|^2 - \|\boldsymbol{h}_s - \boldsymbol{h}_{v^-}\|^2 + 2 \right).
\end{aligned}
$$

where m denotes the number of items, v^+ denotes the ground-truth next item for session s. □

Table 2: Statistics of the datasets.

Dataset	# Interactions	# Items	# Sessions	Avg. Length
Diginetica	786,582	42,862	204,532	4.12
Nowplaying	1,085,410	59,593	145,612	9.21
RetailRocket	871,637	51,428	321,032	6.40
Tmall	427,797	37,367	66,909	10.62
Yoochoose	1,434,349	19,690	470,477	4.64

Given the above lemma, we consider improving the robustness of distance measuring in three aspects. Firstly, data distributions for different recommendation scenarios may vary a lot, and a fixed margin is inappropriate. Thus, we replace the fixed margin 2 by a controllable hyper-parameter τ to suit different scenarios. Secondly, we utilize Dropout [20], a widely adopted technique for robust training, directly over candidate item embeddings. Thirdly, inspired by recent advances in Contrastive Learning [2, 6, 14], we propose to measure the distance via cosine distance for better alignment and uniformity of item embeddings [24]. Then we design the loss function with the proposed Robust Distance Measuring (RDM) technique as,

$$\ell = -\log \frac{\exp\left(\cos(\boldsymbol{h}_s, \boldsymbol{h}'_{v^+})/\tau\right)}{\sum_{i=1}^m \exp\left(\cos(\boldsymbol{h}_s, \boldsymbol{h}'_{v_i})/\tau\right)}, \quad (5)$$

where $\boldsymbol{h}'$ denotes the item embeddings with dropout.

3 EXPERIMENTS

Datasets and evaluation metrics. We conduct experiments on five public datasets collected from real-world platforms: *Diginetica, Nowplaying, RetailRocket, Tmall* and *Yoochoose* with their statistics shown in Table 2. We filter out sessions of length 1 and items appearing less than 5 times across all datasets [3, 13, 26], and split the sessions in each dataset into train/validation/test set in temporal order in a ratio of 8:1:1 [31]. To evaluate the performance of different methods, we employ two widely-used metrics, top-20 Recall (R@20) and top-20 Mean Reciprocal Rank (M@20) [26, 30].
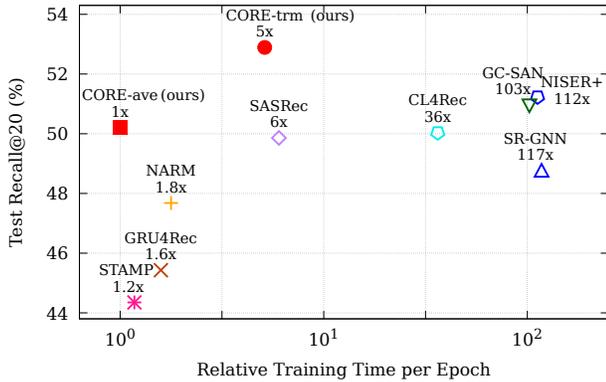
Figure 3: Performances over training time relative to that of CORE-ave on Diginetica.

Baselines. To evaluate the performance of the proposed method, we compare it with the following representative baselines: (1) Matrix factorization based methods: **FPMC** [18]; (2) RNN-based methods: **GRU4Rec** [8] and **NARM** [13]; (3) GNN-based methods: **SR-GNN** [26], **NISER+** [5], **LESSR** [3] and **SGNN-HN** [15]. (4) Transformer-based methods: **SASRec** [11], **GC-SAN** [29] and **CL4Rec** [28]. Note that we don't take methods that introduce additional collaborative filtering information [4, 10, 25, 27] or other side features [9] as baselines. For CORE, we implement two simple variants using mean pooling (Sec. 2.1.1) and Transformers (Sec. 2.1.2) as DNNs in Eqn. (1), namely *CORE-ave* and *CORE-trm*, respectively.

Implementation details The proposed models and all the baselines are implemented based on a popular open-source recommendation library RecBole[1] [32] and its extension RecBole-GNN[2] for easy development and reproduction. The dimension of the latent vectors is fixed to 100, and each session is truncated within a maximum length of 50. We optimized all the compared methods using Adam optimizer [12] with a learning rate of 0.001, and adopted early-stop training if the M@20 performance on the validation set decreased for 5 consecutive epochs. We use a batch size of 2048 for all methods (except TAGNN, for which we use 100 due to the large memory consumption). Other hyper-parameters of baselines are carefully tuned following the suggestions from the original papers and we report each performance under its optimal settings. For CORE, we apply a grid search for controllable margin τ among $\{0.01, 0.05, 0.07, 0.1, 1\}$ and dropout ratio ρ among $\{0, 0.1, 0.2\}$. We finally report metrics on the test set with models that gain the highest performance on the validation set.

Overall comparison. From the experimental results in Table 1, we can observe: CORE outperforms all the baselines significantly over 8 of 10 metrics on the adopted five datasets. Different from these baselines, CORE doesn't apply non-linear layers over item embeddings to encode sessions, but learns weights for each item embedding and adopts a weighted sum for encoding session embeddings and item embeddings in consistent representation space.

[1] https://recbole.io
[2] https://github.com/RUCAIBox/RecBole-GNN

Table 3: Ablation study of CORE's variants on Diginetica and RetailRocket.

Method	Diginetica		RetailRocket	
	R@20	M@20	R@20	M@20
CORE	**52.89**	**18.58**	**61.85**	**38.76**
w/o RCE	49.82	17.41	59.59	36.27
w/o RDM	52.31	18.38	60.93	37.72
SASRec	49.86	17.19	59.81	36.03

Table 4: Performance comparison of different methods and their improved variants on two datasets.

Method	Diginetica		RetailRocket	
	R@20	M@20	R@20	M@20
NARM	47.68	15.58	58.65	34.69
+ RCE	51.86	18.27	60.77	37.01
+ RDM	51.62	17.79	61.33	37.11
+ All	**52.51**	**18.58**	**62.19**	**38.84**
SR-GNN	48.76	16.93	58.71	36.42
+ RCE	49.51	17.53	57.05	35.70
+ RDM	51.36	18.57	61.41	38.27
+ All	**52.38**	**18.95**	**61.43**	**38.38**

Besides, notice that CORE-ave considers neither item order in a session, nor item importances. However, it still outperforms all the baselines on 3 metrics, while gaining comparable results with several strong baselines on other metrics. Without carefully designed encoder architecture, CORE can achieve impressive performance, which further confirms the importance of encoding session embeddings and item embeddings in consistent representation space.

Analysis 1: Efficiency. In Figure 3, we plot the performance of several popular session-based recommendation models as well as CORE-trm over their training time per epoch relative to that of CORE-ave on the Diginetica dataset. We measure the training time on an NVIDIA TITAN V GPU. As we can see, CORE-ave is the fastest while achieving competitive performance. By applying mean pooling over sessions, CORE-ave minimizes memory usage and only learns a single item embedding table during training. CORE-trm has a similar training time as SASRec while achieving the best performance among the compared baselines.

Analysis 2: Ablation study. CORE involves several components (*i.e.,* RCE and RDM) and we now analyze how each part contributes to the performance. As SASRec and CORE-trm (CORE for simplify) shares the same Transformer architecture, we select SASRec as the base model to compare. We mainly consider the following variants: **CORE w/o RCE** means replacing representation-consistent encoder of CORE to SASRec's encoder; **CORE w/o RDM** means replacing the proposed robust distance measuring techniques to the traditional dot product distance; In Table 3, we can observe that the performance order can be summarized as CORE > CORE w/o
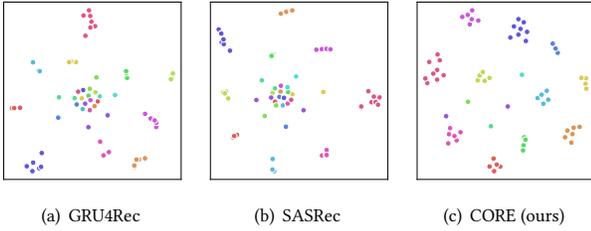
(a) GRU4Rec   (b) SASRec   (c) CORE (ours)

Figure 4: Visualization of learned session embeddings.

RDM > CORE w/o RCE ≃ SASRec. These results indicate that all the parts are useful to improve the final performance.

Analysis 3: Improving existing methods with RCE & RDM. Here we slightly modify several popular existing session-based models, showing how the proposed components RCE and RDM improve the performances of existing methods.

Take *SR-GNN [26] + RCE* as an example, the original encoder can be formulated as $h_s = W[h_{s,n}; h_g]$, where $W \in \mathbb{R}^{d \times 2d}$ are learnable parameters and h_g can be seen as the linear combination of item embeddings within a session. Then we remove W and change the encoder to $h_s = (h_{s,n} + h_g)/2$ and the modified SR-GNN encoder is a variant of the proposed RCE, where the session embedding is linear combination of item embeddings.

Performance comparison between existing works (NARM [13] and SR-GNN [26]) and their improved variants are shown in Table 4. We can see that generally the combination of RCE and RDM can gain dramatic performance improvements compared to the original models. As for only adding one of the proposed techniques, RDM consistently improve the performance, while RCE has a positive effect in most cases.

Analysis 4: Visualization of session embeddings. To show the effectiveness of the proposed model CORE, we visualize the learned session embeddings using t-SNE [22] algorithm in Figure 4. Detailed, sessions with the same next item are viewed as in the same class, and are marked the same color. We randomly sample 15 items as ground truth next items. Then we extract all the corresponding sessions from our test set in Diginetica. We can see that different classes' session embeddings learned by CORE are well separated from each other compared to those learned by GRU4Rec and SASRec.

Analysis 5: Parameter tuning. At last, we exime the impact of several importance parameters towards CORE, *i.e.,* the margin τ and the dropout ratio ρ. In particular, we vary τ in $\{0.01, 0.02, \ldots, 0.1\}$ and present the results in Figure 5 (a) and Figure 5 (b). Our method constantly outperforms the best baseline, and achieves the best performance when $\tau = 0.07$ on Diginetica and $\tau = 0.08$ on Retail-Rocket. Overall, the performance is stable around $0.04 \leq \tau \leq 0.08$. Next, we vary ρ in the range 0 and 0.5 with steps of 0.1. As shown in Figure 5 (c) and Figure 5 (d), when $\rho \leq 0.3$, the item dropout increases the robustness of item embedding learning, while a large ratio of item dropout may hurt the performance, as we can see, the performance decreases sharply when $\rho > 0.3$.
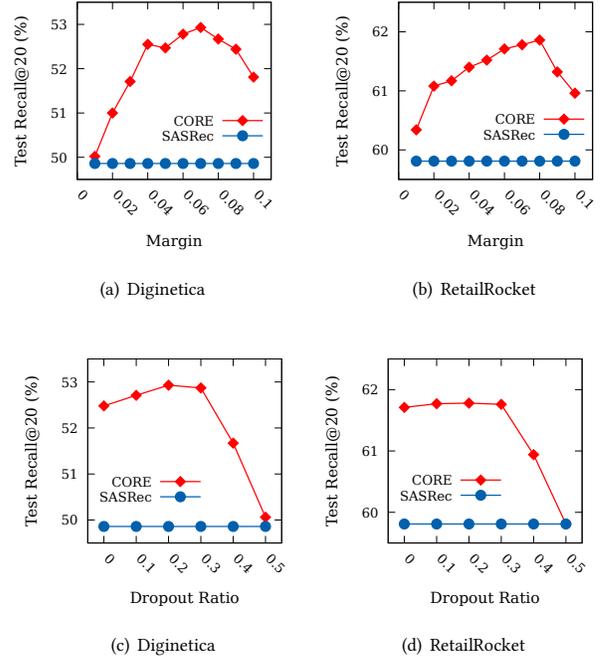

(a) Diginetica   (b) RetailRocket

(c) Diginetica   (d) RetailRocket

Figure 5: Parameter tuning of CORE on Diginetica and Re-tailRocket datasets.

4 CONCLUSION

In this paper, we propose CORE, a simple and effective framework for session-based recommendation within consistent representation space, which unifies the representation space throughout encoding and decoding to overcome the inconsistent prediction issue. Different from stacking multiple non-linear layers over the item embeddings, we propose to just apply weighted sum for item embeddings to encode sessions in the consistent representation space as items. Besdies, we propose robust distance measuring techniques from multiple aspects to prevent overfitting of item embeddings in the proposed framework. Extensive experiments on five public datasets have shown the effectiveness and efficiency of the proposed approach, as well as how the proposed techniques can help existing methods.

For future work, we will consider studying the expressive ability of the proposed representation-consistent encoder both theoretically and empirically. Besides, we will explore how to introduce side features and useful inductive biases to the proposed framework.

REFERENCES

[1] Shuqing Bian, Wayne Xin Zhao, Kun Zhou, Jing Cai, Yancheng He, Cunxiang Yin, and Ji-Rong Wen. 2021. Contrastive Curriculum Learning for Sequential User Behavior Modeling via Data Augmentation. In *CIKM*.

[2] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. 2020. A Simple Framework for Contrastive Learning of Visual Representations. In *ICML*. 1597–1607.

[3] Tianwen Chen and Raymond Chi-Wing Wong. 2020. Handling Information Loss of Graph Neural Networks for Session-based Recommendation. In *KDD*.

[4] Junsu Cho, SeongKu Kang, Dongmin Hyun, and Hwanjo Yu. 2021. Unsupervised Proxy Selection for Session-based Recommender Systems. In *SIGIR*. 327–336.

[5] Priyanka Gupta, Diksha Garg, Pankaj Malhotra, Lovekesh Vig, and Gautam M Shroff. 2019. NISER: Normalized Item and Session Representations with Graph Neural Networks. *arXiv preprint arXiv:1909.04276* (2019).

[6] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. 2020. Momentum Contrast for Unsupervised Visual Representation Learning. In *CVPR*. 9726–9735.

[7] Zhankui He, Handong Zhao, Zhe Lin, Zhaowen Wang, Ajinkya Kale, and Julian McAuley. 2021. Locally constrained self-attentive sequential recommendation. In *CIKM*.

[8] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based Recommendations with Recurrent Neural Networks. In *ICLR*.

[9] Balázs Hidasi, Massimo Quadrana, Alexandros Karatzoglou, and Domonkos Tikk. 2016. Parallel Recurrent Neural Network Architectures for Feature-rich Session-based Recommendations. In *RecSys*. 241–248.

[10] Xunqiang Jiang, Binbin Hu, Yuan Fang, and Chuan Shi. 2020. Multiplex memory network for collaborative filtering. In *SDM*.

[11] Wang-Cheng Kang and Julian J. McAuley. 2018. Self-Attentive Sequential Recommendation. In *ICDM*. 197–206.

[12] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*.

[13] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural Attentive Session-based Recommendation. In *CIKM*. 1419–1428.

[14] Zihan Lin, Changxin Tian, Yupeng Hou, and Wayne Xin Zhao. 2022. Improving Graph Collaborative Filtering with Neighborhood-enriched Contrastive Learning. In *TheWebConf*.

[15] Zhiqiang Pan, Fei Cai, Wanyu Chen, Honghui Chen, and Maarten de Rijke. 2020. Star Graph Neural Networks for Session-based Recommendation. In *CIKM*.

[16] Ruihong Qiu, Jingjing Li, Zi Huang, and Hongzhi Yin. 2019. Rethinking the Item Order in Session-based Recommendation with Graph Neural Networks. In *CIKM*.

[17] 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*. 4806–4813.

[18] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized Markov chains for next-basket recommendation. In *WWW*. 811–820.

[19] Kihyuk Sohn. 2016. Improved Deep Metric Learning with Multi-class N-pair Loss Objective. In *NIPS*. 1849–1857.

[20] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *JMLR* 15, 1 (2014), 1929–1958.

[21] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In *CIKM*. 1441–1450.

[22] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *JMLR* 9, 11 (2008).

[23] Shoujin Wang, Longbing Cao, and Yan Wang. 2019. A Survey on Session-based Recommender Systems. *arXiv preprint arXiv:1902.04864* (2019).

[24] Tongzhou Wang and Phillip Isola. 2020. Understanding Contrastive Representation Learning through Alignment and Uniformity on the Hypersphere. In *ICML*, Vol. 119. 9929–9939.

[25] Ziyang Wang, Wei Wei, Gao Cong, Xiao-Li Li, Xianling Mao, and Minghui Qiu. 2020. Global Context Enhanced Graph Neural Networks for Session-based Recommendation. In *SIGIR*.

[26] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-Based Recommendation with Graph Neural Networks. In *AAAI*. 346–353.

[27] Xin Xia, Hongzhi Yin, Junliang Yu, Qinyong Wang, Lizhen Cui, and Xiangliang Zhang. 2021. Self-Supervised Hypergraph Convolutional Networks for Session-based Recommendation. In *AAAI*.

[28] Xu Xie, Fei Sun, Zhaoyang Liu, Shiwen Wu, Jinyang Gao, Bolin Ding, and Bin Cui. 2020. Contrastive Learning for Sequential Recommendation. *arXiv preprint arXiv:2010.14395* (2020).

[29] Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Victor S. Sheng, Jiajie Xu, Fuzhen Zhuang, Junhua Fang, and Xiaofang Zhou. 2019. Graph Contextualized Self-Attention Network for Session-based Recommendation. In *IJCAI*. 3940–3946.

[30] Feng Yu, Yanqiao Zhu, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2020. TAGNN: Target Attentive Graph Neural Networks for Session-based Recommendation. In *SIGIR*.

[31] Wayne Xin Zhao, Junhua Chen, Pengfei Wang, Qi Gu, and Ji-Rong Wen. 2020. Revisiting Alternative Experimental Settings for Evaluating Top-N Item Recommendation Algorithms. In *CIKM*.

[32] Wayne Xin Zhao, Shanlei Mu, Yupeng Hou, Zihan Lin, Kaiyuan Li, Yushuo Chen, Yujie Lu, Hui Wang, Changxin Tian, Xingyu Pan, Yingqian Min, Zhichao Feng, Xinyan Fan, Xu Chen, Pengfei Wang, Wendi Ji, Yaliang Li, Xiaoling Wang, and Ji-Rong Wen. 2021. RecBole: Towards a Unified, Comprehensive and Efficient Framework for Recommendation Algorithms. In *CIKM*.

[33] Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. 2020. S3-Rec: Self-Supervised Learning for Sequential Recommendation with Mutual Information Maximization. In *CIKM*.

[34] Kun Zhou, Hui Yu, Wayne Xin Zhao, and Ji-Rong Wen. 2022. Filter-enhanced MLP is All You Need for Sequential Recommendation. In *TheWebConf*.