Coarse-to-Fine Sparse Sequential Recommendation

Jiacheng Li[1], Tong Zhao[2], Jin Li[2], Jim Chan[2], Christos Faloutsos[2]
George Karypis[2], Soo-Min Pantel[2], Julian McAuley[2]
j9li@eng.ucsd.edu,{zhaoton,jincli,jamchan,faloutso,gkarypis,pantel,jumcaule}@amazon.com
[1]University of California, San Diego [2]Amazon, United States

ABSTRACT

Sequential recommendation aims to model dynamic user behavior from historical interactions. Self-attentive methods have proven effective at capturing short-term dynamics and long-term preferences. Despite their success, these approaches still struggle to model sparse data, on which they struggle to learn high-quality item representations. We propose to model user dynamics from shopping intents and interacted items simultaneously. The learned intents are coarse-grained and work as prior knowledge for item recommendation. To this end, we present a coarse-to-fine self-attention framework, namely CaFe, which explicitly learns coarse-grained and fine-grained sequential dynamics. Specifically, CaFe first learns intents from coarse-grained sequences which are dense and hence provide high-quality user intent representations. Then, CaFe fuses intent representations into item encoder outputs to obtain improved item representations. Finally, we infer recommended items based on representations of items and corresponding intents. Experiments on sparse datasets show that CaFe outperforms state-of-the-art self-attentive recommenders by 44.03% NDCG@5 on average.

ACM Reference Format:
Jiacheng Li[1], Tong Zhao[2], Jin Li[2], Jim Chan[2], Christos Faloutsos[2], George Karypis[2], Soo-Min Pantel[2], Julian McAuley[2] . 2018. Coarse-to-Fine Sparse Sequential Recommendation. In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

The goal of sequential recommender systems is to predict next items for users by modeling historical interactions as temporally-ordered sequences. Sequential recommenders [9, 10, 16, 17] capture both *long-term* preferences and *short-term* dynamics of users simultaneously in order to improve recommendation accuracy.

Previous works employ Markov Chains (MC) [7, 16], RNN/CNNs [5, 12, 18, 24] and self-attentive models [10, 13, 17, 22] for sequential recommendation. Among these approaches, self-attentive recommenders arguably represent the current state-of-the-art, as the self-attention mechanism [21] is able to efficiently draw context from all past actions and obtain short-term dynamics. Some recent works [14, 23, 25] incorporate item context features into item representations due to the flexibility of self-attention. Despite the effectiveness of existing self-attentive models, in this paper we argue that sequential recommendation on highly-sparse sequences (i.e., containing long-tail items) is still a challenging problem for self-attentive recommenders.

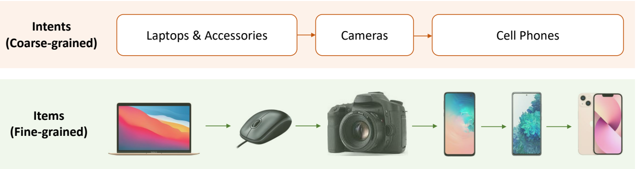


Figure 1: Illustration of a coarse-grained sequence (intents) and a fine-grained sequence (items).

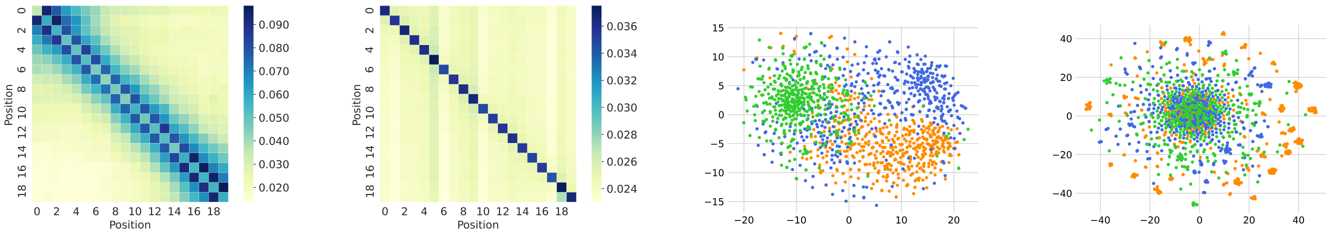
To explore why self-attentive models fail on sparse sequences and validate our motivation, we first conduct two motivating experiments (Section 2.3) with a representative self-attentive recommender (BERT4Rec [17]). Results reveal that main reasons: (1) although self-attentive models directly attend on all interactions, they tend to focus on recent items when trained on (item-)sparse datasets. (2) embeddings of long-tail (infrequent) items are under-trained while models represent frequent items well.

To address the above problems, we can employ another dense sequence (called an *intent* sequence) to provide prior knowledge and well-trained representations for items. As shown in Figure 1, although a user interacts with many items (including infrequent items) in the item sequence, several fall under the same shopping intent. For example, the laptop and the mouse belong to the category `Laptops & Accessories` , and are often purchased together. Hence, if we view categories as intents and explicitly model the intent sequence to predict the next intent, infrequent items can be better understood by their corresponding intent. 'Intents' in our paper could be categories, taxonomies, or sellers which can reveal high-level 'semantics' of items. Critically, intent sequences are relatively dense and make it easy to capture long-term preferences of users. Note that some previous works also modeled user shopping intents by implicitly inferring them from items [3, 11, 19] or feature fusion into item representations [14, 25]. However, we find that these *implicit* intent methods do not improve recommendation performance especially on highly-sparse datasets. In contrast, our method *explicitly* learns intent sequences and item sequences which can improve sequential recommendation on sparse datasets.

In this work, we propose a *Coarse-to-Fine Framework* (CaFe), building on self-attentive networks. CaFe enhances the ability to infrequent item understanding via *explicitly* modeling intent sequences. Specifically, we jointly learn the sequential dynamics of both intents and items with two self-attentive encoders. Compared to previous works that infer the next item via a conditional probability on previous items, CaFe predicts recommended items based on a joint probability distribution of both items and intents. Experiments

Woodstock '18, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06...$15.00
https://doi.org/10.1145/1122445.1122456



(a) *Dense* dataset (b) *Sparse* dataset (c) Frequent items (d) Infrequent items

Figure 2: Motivating experiments. (a) (b) show the average attention map (the first 20 time steps) of BERT4Rec [17]. (c) (d) show item embeddings (projected by t-SNE [20]) of BERT4Rec trained on amazon dataset. Item categories are used as labels for coloring (SPORT GOAL-Green; LIP COLOR-Orange; CRIB-Blue).

show that CaFe significantly outperforms existing self-attentive recommenders by (on average) 44.03% NDCG@5 on sparse datasets.

2 PRELIMINARIES

2.1 Problem Setup

We study sequential recommendation for a user set $\mathcal{U}$, an item set $\mathcal{V}$, an intent set C ($|C| \ll |\mathcal{V}|$) and a set of user interaction sequences $\mathcal{S} = \{S_1, S_2, \ldots, S_{|\mathcal{U}|}\}$. Each item $v \in \mathcal{V}$ has a unique corresponding intent $c \in C$. A user sequence consists of (temporally-ordered) interactions $S_u = (s_1^u, s_2^u, \ldots, s_{|S_u|}^u)$, where $S_u \in \mathcal{S}$, $u \in \mathcal{U}$, $s_i^u = (v_i^u, c_i^u)$. Given the interaction history S_u, we predict the next item $v_{|S_u|+1}^u$.

2.2 Self-Attentive Recommender

Self-attentive recommenders [10, 17, 22, 25] rely on Transformer structure [21] to encode sequential interactions $\mathcal{S}$. In this paper, our backbone model is a directional self-attentive model SASRec [10].

2.2.1 Embedding. For an item set $\mathcal{V}$, an embedding table $\mathbf{E} \in \mathbb{R}^{d\times|\mathcal{V}|}$ is used for all items, whose element $\mathbf{e}_i \in \mathbb{R}^d$ denote the embedding for item v_i and d is the latent dimensionality. To be aware of item positions, SASRec maintains a learnable position embedding $\mathbf{P} \in \mathbb{R}^{d\times n}$, where n is the maximum sequence length. All interaction sequences are padded to n with a special 'padding' item. Hence, given a padded item sequence $S^v = \{v_1, v_2, \ldots, v_n\}$, the input embedding is computed as:

$$\mathbf{M}^v = \text{Embedding}(S^v) = [\mathbf{e}_1 + \mathbf{p}_1, \mathbf{e}_2 + \mathbf{p}_2, \ldots, \mathbf{e}_n + \mathbf{p}_n] \tag{1}$$

2.2.2 Transformer Encoder. The Transformer encoder adopts scaled dot-product attention [21] denoted as f_{att}. Given $\text{H}_i^l \in \mathbb{R}^d$ is an embedding for v_i after the l^{th} self-attention layer and $\text{H}_i^0 = \mathbf{e}_i + \mathbf{p}_i$, the output from multi-head (#head=M) self-attention is calculated as:

$$\mathbf{O}_i = \text{Concat}[\mathbf{O}_i^{(1)}, \ldots, \mathbf{O}_i^{(m)}, \ldots, \mathbf{O}_i^{(M)}]\mathbf{W}_O, \tag{2}$$

$$\mathbf{O}_i^{(m)} = \sum_{j=1}^{n} f_{\text{att}}(\text{H}_i^l \mathbf{W}_Q^{(m)}, \text{H}_j^l \mathbf{W}_K^{(m)}) \cdot \text{H}_j^l \mathbf{W}_V^{(m)}, \tag{3}$$

where $\mathbf{W}_Q^{(m)}, \mathbf{W}_K^{(m)}, \mathbf{W}_V^{(m)} \in \mathbb{R}^{d\times d/M}$ are the m-th learnable projection matrices; $\mathbf{W}_O \in \mathbb{R}^{d\times d}$ is a learnable matrix to get the output $\mathbf{O}_i$ from concatenated heads. Our backbone SASRec model is a directional self-attention model implemented by forbidding attention weights between v_i and v_j ($j > i$).

To prevent overfitting and achieve a stable training process, the next layer H_i^{l+1} is generated from $\mathbf{O}_i$ with Residual Connections [6], Layer Normalization [2] and Pointwise Feed-Forward Networks [21].

2.3 Self-Attentive Models on Sparse Data

To find reasons that self-attentive models fail on sparse data, we conduct two motivating experiments with BERT4Rec. In experiments, we set hidden size $d = 128$ and maximum sequence length of $n = 50$. We adopt the same training method as in [17].

2.3.1 Attention Scope. We investigate the difference between self-attention scope on dense versus sparse data: Amazon [15] (av. 2.81 interactions per item) is used as a sparse dataset; A dense version (av. 11.23 interactions per item) is constructed by setting the minimum item frequency to 5. We visualize the average attention map from the first self-attention layer in Figure 2a/2b which shows that the model attends on more recent items on the sparse dataset, and less recent items for the dense dataset. This indicates that: (1) recent items are important sparse data; (2) self-attentive models combine long and short-term dynamics, but they still struggle to capture long-term preferences on item-sparse datasets.

2.3.2 Trained Embedding. In this experiment, we explore the difference of trained item embeddings between frequent items and infrequent items in the same dataset. Specifically, we first select three intents (i.e., categories in the Amazon dataset), then obtain 500 the most frequent items and 500 the most infrequent items in each intent and visualize their trained embeddings in Figure 2c and 2d respectively. We can see that frequent items with the same intent are usually close to each other (form three colored clusters in Figure 2c) while infrequent item embeddings scatter and are mostly around the origin. These observations indicate that (1) the model represents frequent items well, though infrequent items embeddings are under-trained; (2) intents can provide useful prior knowledge for items because the clusters from well-trained item embeddings are aligned with different intents (see Figure 2c).

3 METHOD

In this section, we propose CaFe to advance sequential recommendation performance on sparse datasets.

3.1 Embedding Layer

In our method, an interaction sequence S_u includes an item sequence S_u^v and an intent sequence S_u^c for user u. Hence, We maintain two embedding tables $\mathbf{E}^v \in \mathbb{R}^{d\times|\mathcal{V}|}$ and $\mathbf{E}^c \in \mathbb{R}^{d\times|C|}$ for items

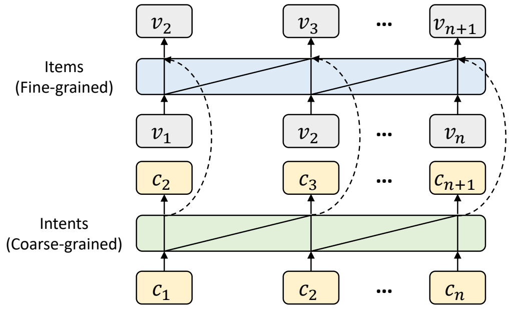


Figure 3: Framework illustration of CaFe.

and intents. Two separate position embeddings $\mathbf{P^v} \in \mathbb{R}^{d\times n}$ and $\mathbf{P^c} \in \mathbb{R}^{d\times n}$ are created. We encode S_u^v and S_u^c to get input embeddings $\mathbf{M}^v$ and $\mathbf{M}^c$ as follows:

$$\mathbf{M}^v = \text{Embedding}^v(S_u^v); \mathbf{M}^c = \text{Embedding}^c(S_u^c), \tag{4}$$

where $\mathbf{M}^v, \mathbf{M}^c \in \mathbb{R}^{d\times n}$ and Embedding$(\cdot)$ is the positional embedding operation in Equation (1).

3.2 Coarse-to-Fine Encoder

Recalling the conclusions in Section 2.3, embeddings of infrequent items are under-trained and the self-attentive model tends to focus on short-term items when the dataset has sparse items. We can also see that intent types are highly aligned with item clusters trained by a self-attentive recommender. Motivated by these observations, we propose to explicitly learn intents in a sequential model and the outputs of the intent model are used as prior knowledge to improve item representations and understand long-term preferences. In this section, we introduce our coarse-to-fine encoder which includes two components, the intent encoder and the item encoder. The overall framework is illustrated in Figure 3.

3.2.1 Intent Encoder. For intent sequences, we aim to capture coarse-grained interest dynamics of users. Intent sequences are usually dense because $|\mathcal{C}|$ is much smaller than $|\mathcal{V}|$. Therefore, we apply a standard SASRec model (in Section 2.2) as the encoder for intent sequences. Given intent embeddings $\mathbf{M}^c$, outputs of the SASRec encoder are used as intent sequence representations $\mathbf{R}^c \in \mathbb{R}^{d\times n}$.

3.2.2 Item Encoder. From our motivating experiments, we see that more recent items are important for next item prediction on sparse datasets. Basically, our item encoder is also a directional Transformer but has enhanced ability to focus on recent items. Inspired by [8], we enhance short-term user dynamics modeling in the item encoder by applying a masking score θ_{ij} on f_{att} in Equation (3). Formally, the re-weighted attention weights are calculated by:

$$f_{\text{att}}(\mathbf{Q}_i, \mathbf{K}_j) = \frac{\exp(w_{ij}) \cdot \theta_{ij}}{\sum_{k=1}^{n} \exp(w_{ik}) \cdot \theta_{ik}}, w_{ij} = \frac{\mathbf{Q}_i \mathbf{K}_j^T}{\sqrt{d}} \tag{5}$$

where $\theta = 1$ for a standard scaled dot-product attention and $\sqrt{d}$ is a scale factor. The masking operation $\exp(w_{ij}) \cdot \theta_{ij}$ can be rewritten as $\exp(w_{ij} + \ln\theta_{ij})$. We learn $\ln\theta_{ij}$ from H_i^l, H_j^l and the distance between items v_i and v_j:

$$\ln\theta_{ij} = (\mathrm{H}_i^l \mathbf{W}_Q^{(m)} + \mathrm{H}_j^l \mathbf{W}_K^{(m)} + \mathbf{d}_{ij})\mathbf{W}_L^{(m)} + \mathbf{b}_L \tag{6}$$

where $\mathbf{W}_L^{(m)} \in \mathbb{R}^{d/M\times 1}$, $\mathbf{b}_L \in \mathbb{R}^1$, distance embedding $\mathbf{d}_{ij} \in \mathbb{R}^{d/M}$ is the $(n+i-j)$-th vector from distance embedding table $\mathbf{D} \in \mathbb{R}^{d\times 2n}$ and $\mathbf{W}_Q^{(m)}, \mathbf{W}_K^{(m)}$ are from Equation (3). We encode $\mathbf{M}^v$ with the item encoder to get the item sequence representations $\mathbf{R}^v \in \mathbb{R}^{d\times n}$

Table 1: Data statistics.

Datasets	#Interaction	#Item	#Intent	#Sequence	Ave. Length	Density
Amazon	5,370,171	1,910,226	1,392	131,248	40.9	2e-5
Tmall	14,460,516	1,788,758	9,999	131,086	110.3	6e-5

Current item sequence outputs $\mathbf{R}^v$ mostly focus on recent items and cannot represent infrequent items well. To add long preferences and obtain prior knowledge from intents for infrequent items, we add $\mathbf{R}^v$ and $\mathbf{R}^c$ together to get final representations $\mathbf{R} \in \mathbb{R}^{d\times n}$:

$$\mathbf{R} = \mathbf{R}^v + \mathbf{R}^c \tag{7}$$

3.2.3 Prediction Layer. In CaFe, we predict the next intent and item simultaneously from $\mathbf{R}^c$ and $\mathbf{R}$. Specifically, we adopt matrix factorization (MF) to compute the relevance at time step t between encoder outputs and embeddings:

$$r_{j,t}^c = \mathbf{R}_t^c \mathbf{E}_j^{cT}, r_{k,t}^v = \mathbf{R}_t \mathbf{E}_k^{vT} \tag{8}$$

where $\mathbf{E}_j^c \in \mathbb{R}^d, \mathbf{E}_k^v \in \mathbb{R}^d$ denotes embeddings of the j-th intent and the k-th item in $\mathbf{E}^c, \mathbf{E}^v$ respectively.

3.3 Network Training

CaFe learns from both item sequences and intent sequences, and we adopt the binary cross entropy loss:

$$\begin{aligned}\mathcal{L} =& \mathcal{L}_c + \mathcal{L}_v \\ =& -\sum_{S_u \in \mathcal{S}} \sum_{1\le t\le n} \left[\log(\sigma(r_{y^c,t}^c)) + \sum_{c_j \notin S_u} \log(1-\sigma(r_{c_j,t}^c))\right] \\ &- \sum_{S_u \in \mathcal{S}} \sum_{1\le t\le n} \left[\log(\sigma(r_{y^v,t}^v)) + \sum_{v_k \notin S_u} \log(1-\sigma(r_{v_k,t}^v))\right]\end{aligned} \tag{9}$$

where y^c, y^v are an expected intent and item; c_j, v_k are a negative intent and item randomly generated for each time step in each sequence. Other training details are the same as in SASRec [10].

3.4 Inference

Previous sequential recommenders infer the next items conditioned on previous items. In contrast, CaFe computes the joint probability distribution of the item and corresponding intent conditioned on previous intents and items. Formally, at inference we find the item v_k and corresponding intent c_j that maximize the probability:

$$\begin{aligned}P(c_j, v_k | S_u^c, S_u^v, \Theta) =& P(c_j | S_u^c, \Theta) P(v_k | c_j, S_u^c, S_u^v, \Theta) \\ =& \sigma(r_{j,t}^c)\sigma(r_{k,t}^v)\end{aligned} \tag{10}$$

where σ is the sigmoid function, Θ denotes the parameter set of CaFe and S_u^c, S_u^v are intent and item sequences for user u.

4 EXPERIMENTS

4.1 Experimental Setting

4.1.1 Data. We consider two sparse datasets (see Table 1): **Amazon** [15] is collected from Amazon.com, and we use item categories

Table 2: Model comparision. The best results are bold and the best baselines are underlined.

		Item-only Methods					Intent-aware Methods					Improvement
Dataset	Metric	PopRec	SASRec	BERT4Rec	SSE-PT	LOCKER	NOVA	FDSA	BERT-F	LOCKER-F	CaFe	
Amazon	NDCG@5	0.0286	0.1418	0.1830	0.2108	0.2170	0.0281	0.0670	0.2199	0.2436	**0.3733**	+53.24%
	HR@5	0.0487	0.1844	0.2240	0.2501	0.2597	0.0475	0.1089	0.2676	0.2947	**0.4813**	+63.32%
	MRR	0.0485	0.1522	0.1956	0.2239	0.2297	0.0477	0.0857	0.2329	0.2529	**0.3656**	+44.56%
Tmall	NDCG@5	0.0360	0.0741	0.2753	0.2106	0.2961	0.0501	0.1083	0.2998	0.3182	**0.4290**	+34.82%
	HR@5	0.0596	0.1205	0.3673	0.2977	0.3872	0.0812	0.1685	0.3917	0.4098	**0.5152**	+25.72%
	MRR	0.0577	0.0948	0.2782	0.2173	0.2979	0.0716	0.1265	0.3014	0.3189	**0.4268**	+33.84%

as coarse-grained sequences; **Tmall** is released in the IJCAI-15 challenge [1]. Sellers of products are used as coarse-grained sequences. We follow [10, 17] to conduct a leave-last-2-out data split.

4.1.2 Baselines. We compare two groups of works as our baselines which include methods with only items and methods using both intents and items. *Item-only Methods*: **PopRec**, a baseline method that recommends items according to item occurrences in the dataset. **SASRec** [10], a directional self-attention method that is used as our backbone model. **BERT4Rec** [17], a bi-directional self-attention method that learns to recommend items via a cloze task similar to BERT [4]. **SSE-PT** [22], extends SASRec by introducing explicit user representations. **LOCKER** [8], enhances short-term user dynamics via local self-attention. *Intent-aware Methods*: **NOVA** [14], uses non-invasive self-attention to leverage side information. We use intents as side information. **FDSA** [25], applies separated item and feature sequences but does not explicitly learn the feature sequences. **BERT-F, LOCKER-F**, our extension of BERT4Rec and LOCKER, which incorporate intents in the same way as FDSA.

4.1.3 Evaluation and Implementation. We choose truncated Hit Ratio (HR@K), Normalized Discounted Cumulative Gain (NDCG) and Mean Reciprocal Rank (MRR) to measure ranking quality (K=5). Following BERT4Rec [17], we randomly sample 100 negative items according to their popularity for each ground truth item. For all baselines on two datasets, the maximum length of sequences n is 50; hidden size d is 128; batch size is 64. We implement all models and tune other hyper-parameters following authors' suggestions.

4.2 Result Analysis

4.2.1 General Performance. Table 2 shows ranking performance on two datasets. We find that: (1) Previous intent-aware methods that fuse intent features into item representations achieve limited improvements. We think the reason is that baselines did not learn intent representations from intent sequences but from item sequences. However, on such sparse datasets, items hardly provide supervision for intent learning hence low-quality intent features cannot provide useful information or even result in performance decay (NOVA and FDSA). (2) Compared to global attention (BERT4Rec), local attention (LOCKER) can consistently improve recommendations on these two datasets which validate our observations in motivating experiments. (3) CaFe outperforms all baselines significantly. Compared to the strongest baseline LOCKER-F, our model gains about 44.03% NDCG@5 and about 44.53% HR@5 improvements on average. See Section 4.2.2 for detailed analysis.

4.2.2 Ablation Study. To validate the effectiveness of our proposed method, we conduct an ablation study on Tmall dataset; Table 3 shows results. Compared to (1) FDSA which fuses intent features into item representations, (2) modeling intents explicitly (i.e., learning intent representations from intent sequences) is critical to make intent representations effective for items. (4) joint probability inference largely improves recommendation performance by providing coarse-grained knowledge during inference. (3) local self-attention can further improve results by focusing on more recent items.

	Backbone (SASRec)	+(1) (FDSA)	+(1)(2)	+(1)(2)(3)	+(1)(2)(4)	+(1)(2)(3)(4) (CaFe)
NDCG@5	0.0741	0.1083	0.3045	0.3159	0.4254	**0.4290**
HR@5	0.1205	0.1685	0.3938	0.4066	0.5117	**0.5152**
MRR	0.0948	0.1265	0.3069	0.3172	0.4239	**0.4268**

Table 3: Ablation study on Tmall dataset. (1) fusing intents into item embeddings; (2) modeling intents explicitly; (3) local self-attention of item encoder; (4) inference with joint probability distribution of items and corresponding intents.

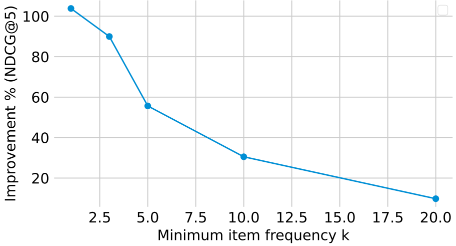


Figure 4: Improvement on Amazon compared to BERT4Rec.

4.2.3 Improvement vs. Sparsity. We investigate CaFe performance compared to BERT4Rec with different dataset sparsity in Figure 4. Models (CaFe and BERT4Rec) are trained and tested on Amazon datasets with different minimum item frequency k. Smaller k means the dataset is sparser. We can see that the improvement is more than 100% on the dataset with $k = 1$ (original) and the improvement is less than 10% on the dataset with $k = 20$ (the most dense). The results show the effectiveness of CaFe on sparse datasets.

5 CONCLUSION

Self-attentive recommenders have shown promising results in sequential recommendation. However, we find that existing methods still struggle to learn high-quality item representations from sparse data. In this paper, we introduce a coarse-to-fine framework (CaFe) that explicitly models intent sequences and enhances infrequent item representations by knowledge from intents. Furthermore, we propose to infer recommended items based on joint probability of intents and items. Experimental results show that CaFe significantly improves recommendation performance on sparse datasets.

REFERENCES

[1] 2015. https://ijcai-15.org/repeat-buyers-prediction-competition/. In *IJCAI*.

[2] Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer Normalization. *ArXiv* abs/1607.06450 (2016).

[3] Wanyu Chen, Pengjie Ren, Fei Cai, Fei Sun, and M. de Rijke. 2020. Improving End-to-End Sequential Recommendations with Intent-aware Diversification. *Proceedings of the 29th ACM International Conference on Information & Knowledge Management* (2020).

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *ArXiv* abs/1810.04805 (2019).

[5] Robin Devooght and Hugues Bersini. 2017. Long and Short-Term Recommendations with Recurrent Neural Networks. *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization* (2017).

[6] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), 770–778.

[7] Ruining He and Julian McAuley. 2016. Fusing Similarity Models with Markov Chains for Sparse Sequential Recommendation. *2016 IEEE 16th International Conference on Data Mining (ICDM)* (2016), 191–200.

[8] Zhankui He, Handong Zhao, Zhe Lin, Zhaowen Wang, Ajinkya Kale, and Julian McAuley. 2021. Locker: Locally Constrained Self-Attentive Sequential Recommendation. *Proceedings of the 30th ACM International Conference on Information & Knowledge Management* (2021).

[9] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based Recommendations with Recurrent Neural Networks. *CoRR* abs/1511.06939 (2016).

[10] Wang-Cheng Kang and Julian McAuley. 2018. Self-Attentive Sequential Recommendation. *2018 IEEE International Conference on Data Mining (ICDM)* (2018), 197–206.

[11] Haoyang Li, Xin Wang, Ziwei Zhang, Jianxin Ma, Peng Cui, and Wenwu Zhu. 2021. Intention-aware Sequential Recommendation with Structured Intent Transition. *IEEE Transactions on Knowledge and Data Engineering* (2021), 1–1.

[12] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural Attentive Session-based Recommendation. *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management* (2017).

[13] Jiacheng Li, Yujie Wang, and Julian McAuley. 2020. Time Interval Aware Self-Attention for Sequential Recommendation. *Proceedings of the 13th International Conference on Web Search and Data Mining* (2020).

[14] Chang Liu, Xiaoguang Li, Guohao Cai, Zhenhua Dong, Hong Zhu, and Lifeng Shang. 2021. Non-invasive Self-attention for Side Information Fusion in Sequential Recommendation. In *AAAI*.

[15] Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying Recommendations using Distantly-Labeled Reviews and Fine-Grained Aspects. In *EMNLP*.

[16] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized Markov chains for next-basket recommendation. In *WWW '10*.

[17] 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. *Proceedings of the 28th ACM International Conference on Information and Knowledge Management* (2019).

[18] Jiaxi Tang and Ke Wang. 2018. Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding. *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining* (2018).

[19] Md. Mehrab Tanjim, Congzhe Su, Ethan Benjamin, Diane J. Hu, Liangjie Hong, and Julian McAuley. 2020. Attentive Sequential Models of Latent Intent for Next Item Recommendation. *Proceedings of The Web Conference 2020* (2020).

[20] Laurens van der Maaten and Geoffrey E. Hinton. 2008. Visualizing Data using t-SNE. *Journal of Machine Learning Research* 9 (2008), 2579–2605.

[21] Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. *ArXiv* abs/1706.03762 (2017).

[22] Liwei Wu, Shuqing Li, Cho-Jui Hsieh, and James Sharpnack. 2020. SSE-PT: Sequential Recommendation Via Personalized Transformer. *Fourteenth ACM Conference on Recommender Systems* (2020).

[23] An Yan, Chaosheng Dong, Yan Gao, Jinmiao Fu, Tong Zhao, Yi Sun, and Julian McAuley. 2022. Personalized complementary product recommendation. In *WWW*.

[24] Fajie Yuan, Alexandros Karatzoglou, Ioannis Arapakis, Joemon M. Jose, and Xiangnan He. 2019. A Simple Convolutional Generative Network for Next Item Recommendation. *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining* (2019).

[25] Tingting Zhang, Pengpeng Zhao, Yanchi Liu, Victor S. Sheng, Jiajie Xu, Deqing Wang, Guanfeng Liu, and Xiaofang Zhou. 2019. Feature-level Deeper Self-Attention Network for Sequential Recommendation. In *IJCAI*.