Ensemble Modeling with Contrastive Knowledge Distillation for Sequential Recommendation

Hanwen Du
hwdu@stu.suda.edu.cn
Soochow University
Suzhou, Jiangsu, China

Huanhuan Yuan
hhyuan@stu.suda.edu.cn
Soochow University
Suzhou, Jiangsu, China

Pengpeng Zhao[*]
ppzhao@suda.edu.cn
Soochow University
Suzhou, Jiangsu, China

Fuzhen Zhuang
zhuangfuzhen@buaa.edu.cn
Beihang University
Beijing, China

Guanfeng Liu
guanfeng.liu@mq.edu.au
Macquarie University
Sydney, Australia

Lei Zhao
zhaol@suda.edu.cn
Soochow University
Suzhou, Jiangsu, China

Yanchi Liu
yanchi.liu@rutgers.edu
Rutgers University
New Brunswick, New Jersey, USA

Victor S. Sheng
victor.sheng@ttu.edu
Texas Tech University
Lubbock, Texas, USA

code: https://github.com/hw-du/EMKD.

SIGIR 2023

Reported by Minqin Li

Introduction

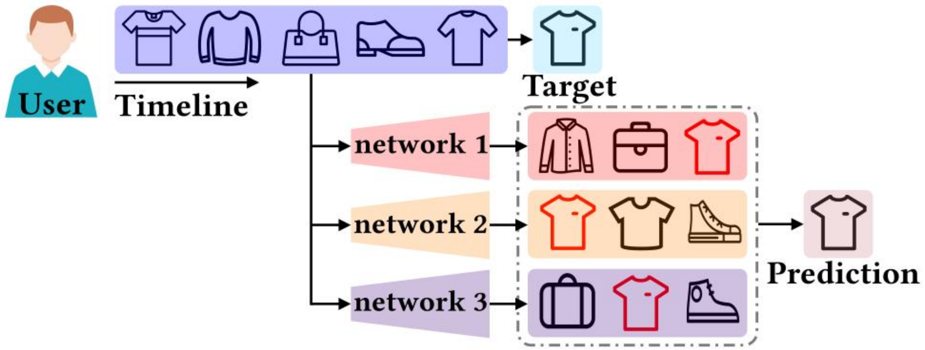


Figure 1: An illustration of ensemble modeling for sequential recommendation. Three parallel networks make different predictions based on users' historical interactions. Although each individual network is unable to make an accurate prediction, combining the predictions of these networks together will get the correct result.

We propose a novel framework called Ensemble Modeling with Contrastive Knowledge Distillation for sequential recommendation (EMKD). To the best of our knowledge, this is the first work to apply the ensemble modeling to sequential recommendation.

We propose a novel contrastive knowledge distillation approach that facilitates knowledge transfer and distills knowledge from both the representation level and the logits level

Method

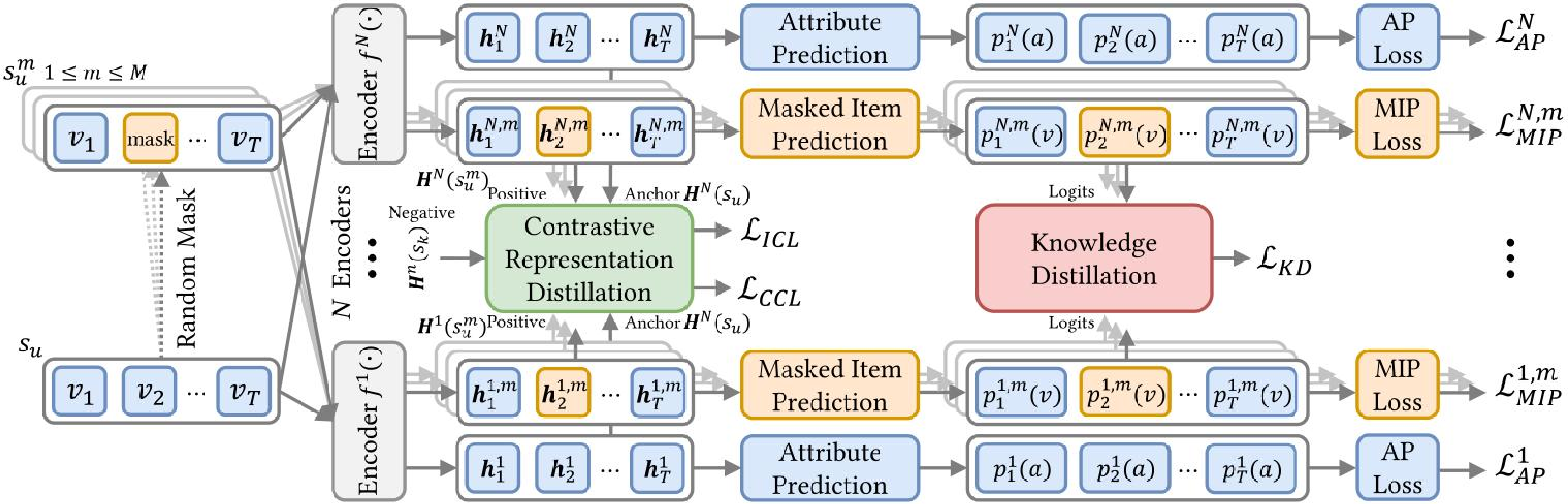
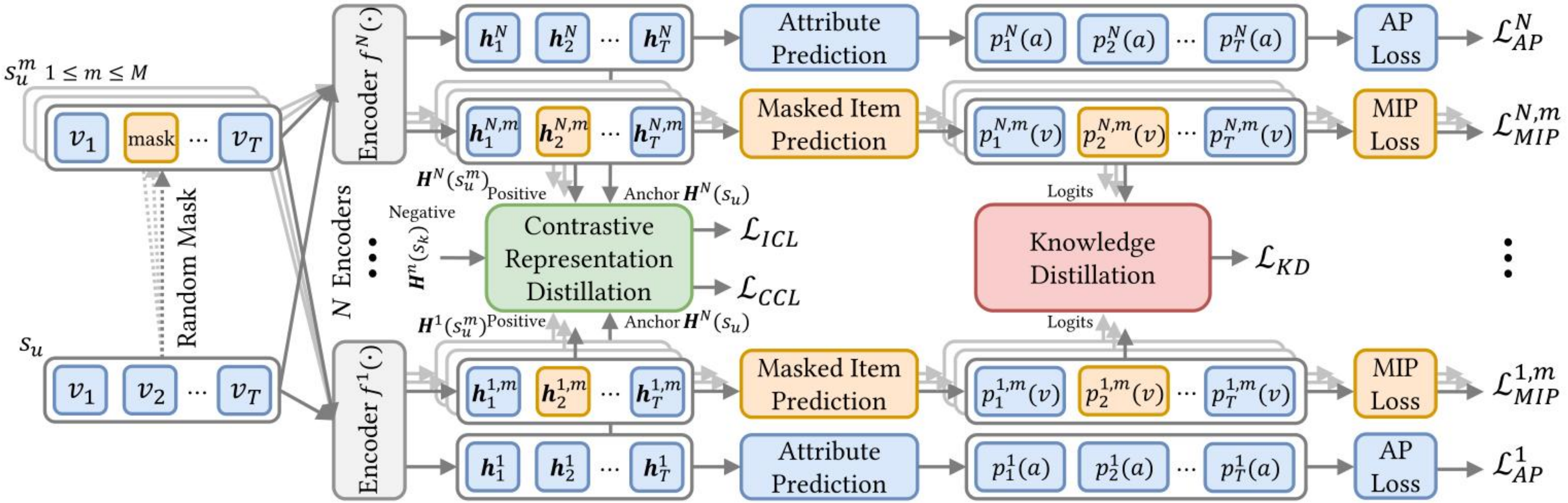


Figure 2: An overview of EMKD with N parallel networks $f^1(\cdot), \cdots, f^N(\cdot)$. For each original sequence s_u, we generate M different masked sequences. The hidden representations of the original sequence $H^1(s_u), \cdots, H^N(s_u)$ serve as the anchor for contrastive representation distillation and are used for the attribute prediction task, while the hidden representations of the masked sequences $H^1(s_u^m), \cdots, H^N(s_u^m)$ serve as positive samples for contrastive representation distillation and are used for the masked item prediction task. Negative samples $H^n(s_k) (1 \le n \le N)$ for contrastive representation distillation are collected from the same batch. We compute the Kullback-Leibler divergence on the logits of the masked item prediction task between different networks for knowledge distillation.

Method



$$E(s_u) = [\boldsymbol{v}_1 + \boldsymbol{p}_1, \boldsymbol{v}_2 + \boldsymbol{p}_2, \cdots, \boldsymbol{v}_T + \boldsymbol{p}_T] \tag{1}$$

$$H(s_u) = [\boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h}_T] = f(s_u) = Trm(E(s_u)) \tag{2}$$

$$s_u^m = \left[\widehat{v}_1, \widehat{v}_2, \cdots, \widehat{v}_T\right], 1 \le m \le M$$

$$\widehat{v}_t = \begin{cases} v_t, & t \notin \mathcal{I}_u^m \\ [\text{mask}], & t \in \mathcal{I}_u^m \end{cases} \tag{3}$$

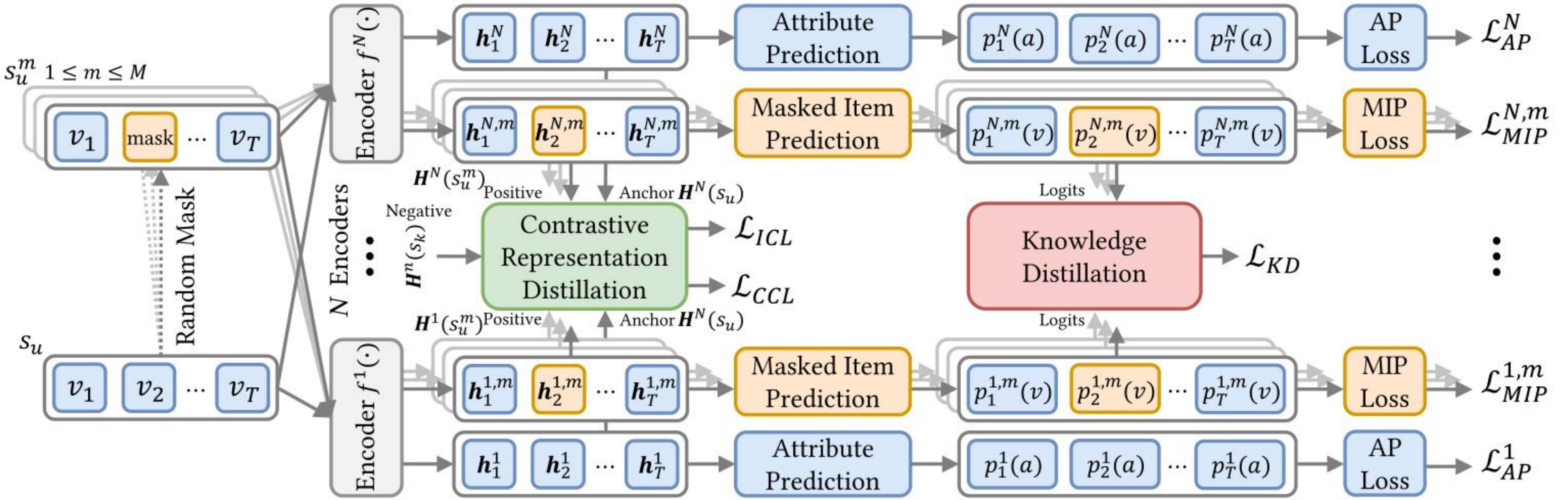
$$H^n(s_u^m) = [\boldsymbol{h}_1^{n,m}, \boldsymbol{h}_2^{n,m}, \cdots, \boldsymbol{h}_T^{n,m}] = f^n(s_u^m), 1 \le n \le N, 1 \le m \le M \tag{4}$$

$$p_t^{n,m}(v) = \boldsymbol{h}_t^{n,m}\mathbf{W} + \boldsymbol{b}, 1 \le t \le T \tag{5}$$

$$q_t^{n,m}(v_i) = \frac{\exp(p_t^{n,m}(v_i))}{\sum_{j=1}^{|\mathcal{V}|} \exp(p_t^{n,m}(v_j))} \tag{6}$$

$$\mathcal{L}_{MIP}^{n,m} = -\sum_{t \in \mathcal{I}_u^m} \sum_{i=1}^{|\mathcal{V}|} y_i \log q_t^{n,m}(v_i)$$

Method



(10)

(10)

$$\mathcal{L}_{MIP} = \sum_{n=1}^{N} \sum_{m=1}^{M} \mathcal{L}_{MIP}^{n,m} \tag{7}$$

$$\boldsymbol{H}^n(s_u) = [\boldsymbol{h}_1^n, \boldsymbol{h}_2^n, \cdots, \boldsymbol{h}_T^n] = f^n(s_u), 1 \le n \le N \tag{8}$$

$$p_t^n(a) = \sigma(\boldsymbol{h}_t^n \boldsymbol{W}' + \boldsymbol{b}'), 1 \le t \le T \tag{9}$$

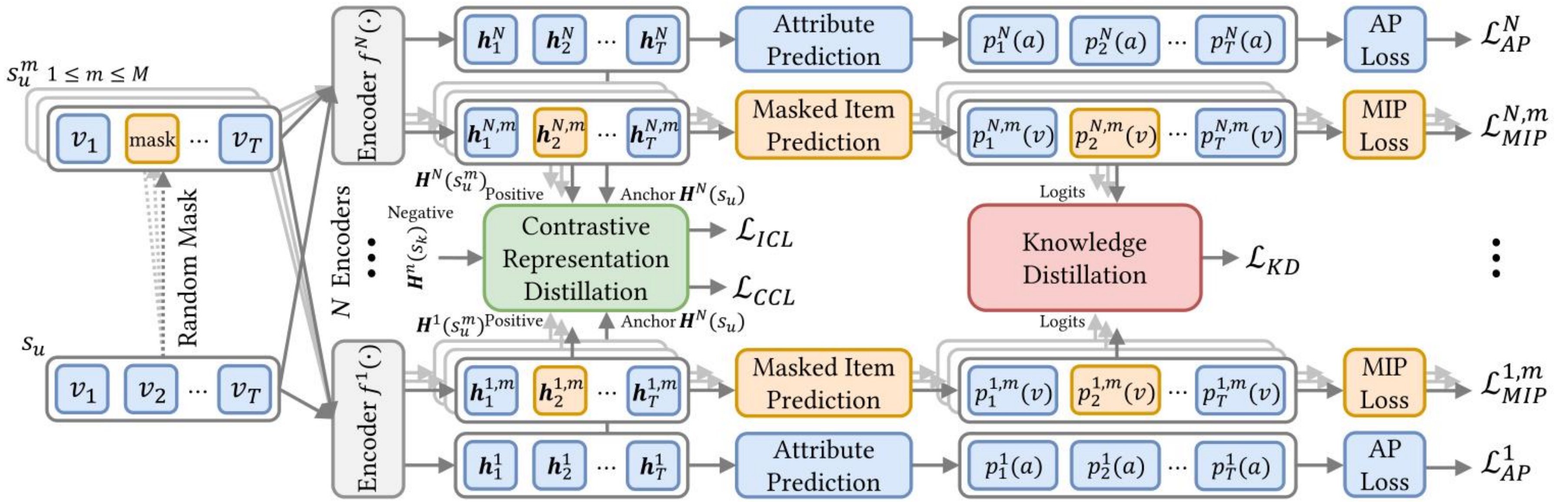
$$\mathcal{L}_{AP}^n = -\sum_{t=1}^{T} \sum_{i=1}^{|\mathcal{A}|} \left[y_i \log p_t^n(a_i) + (1 - y_i) \log(1 - p_t^n(a_i)) \right] \tag{10}$$

$$\mathcal{L}_{AP} = \sum_{n=1}^{N} \mathcal{L}_{AP}^n \tag{11}$$

$$\mathcal{L}_{ICL}^n = -\sum_{m=1}^{M} \log \frac{\exp(g(\boldsymbol{H}^n(s_u), \boldsymbol{H}^n(s_u^m))/\tau)}{\sum_{k=1,k\neq u}^{\mathcal{B}} \exp(g(\boldsymbol{H}^n(s_u), \boldsymbol{H}^n(s_k))/\tau)} \tag{12}$$

$$\mathcal{L}_{ICL} = \sum_{n=1}^{N} \mathcal{L}_{ICL}^n \tag{13}$$

Method



$$\mathcal{L}_{CCL}^{x,y} = -\sum_{m=1}^{M} \log \frac{\exp(g(\boldsymbol{H}^x(s_u), \boldsymbol{H}^y(s_u^m))/\tau)}{\sum_{k=1,k\neq u}^{\mathcal{B}} \exp(g(\boldsymbol{H}^x(s_u), \boldsymbol{H}^y(s_k))/\tau)} \quad (14)$$

$$\mathcal{L}_{CCL} = \sum_{x=1}^{N} \sum_{y=1, y\neq x}^{N} \mathcal{L}_{CCL}^{x,y} \quad (15)$$

$$z_t^{x,m}(v_i) = \frac{\exp(p_t^{x,m}(v_i)/\tau)}{\sum_{j=1}^{|\mathcal{V}|} \exp(p_t^{x,m}(v_j)/\tau)}, z_t^{y,m}(v_i) = \frac{\exp(p_t^{y,m}(v_i)/\tau)}{\sum_{j=1}^{|\mathcal{V}|} \exp(p_t^{y,m}(v_j)/\tau)} \quad (16)$$

$$\mathcal{L}_{KD}^{x,y} = \sum_{m=1}^{M} \sum_{t\in \mathcal{I}_u^m} \sum_{i=1}^{|\mathcal{V}|} z_t^{x,m}(v_i) \log \frac{z_t^{x,m}(v_i)}{z_t^{y,m}(v_i)} \quad (17)$$

$$\mathcal{L}_{KD} = \sum_{x=1}^{N} \sum_{y=1, y\neq x}^{N} \mathcal{L}_{KD}^{x,y} \quad (18)$$

$$\mathcal{L}_{EMKD} = \mathcal{L}_{MIP} + \mathcal{L}_{AP} + \lambda(\mathcal{L}_{ICL} + \mathcal{L}_{CCL}) + \mu \mathcal{L}_{KD} \quad (19)$$

$$\boldsymbol{H}^n(s_{u'}) = [\boldsymbol{h}_2^n, \boldsymbol{h}_3^n, \cdots, \boldsymbol{h}_T^n, \boldsymbol{h}_{\text{mask}}^n] = f^n(s_{u'})$$

$$p(v) = \frac{1}{N} \sum_{n=1}^{N} (\boldsymbol{h}_{\text{mask}}^n \boldsymbol{W} + \boldsymbol{b}) \quad (20)$$

Experiments

Table 1: Performance comparison (NDCG@10) between the original model and the ensemble models. We independently train two parallel networks initialized with different random seeds and compare the result with the original model.

Model	GRU4Rec		Caser		SASRec	
	Original	Ensemble(2×)	Original	Ensemble(2×)	Original	Ensemble(2×)
Beauty	0.0175	0.0199	0.0212	0.0247	0.0284	0.0365
Toys	0.0097	0.0102	0.0168	0.0193	0.0320	0.0378
ML-1M	0.0649	0.0720	0.0734	0.0786	0.0918	0.1032

Experiments

Table 2: Dataset statistics after preprocessing.

Datasets	Beauty	Toys	ML-1M
#users	22,363	19,412	6,040
#items	12,101	11,924	3,953
#actions	198,502	167,597	1,000,209
avg. actions/user	8.9	8.6	163.5
avg. actions/item	16.4	14.1	253.0
sparsity	99.93%	99.93%	95.81%
#attributes	1,221	1,027	18
avg. attributes/item	5.1	4.3	1.7

Experiments

Table 3: Overall performance of different methods for sequential recommendation. The best score and the second-best score in each row are bolded and underlined, respectively. The last column indicates improvements over the best baseline method.

Dataset	Metric	GRU4Rec	Caser	SASRec	BERT4Rec	FDSA	S^3-Rec	MMInfoRec	CL4SRec	DuoRec	EMKD	Improv.
Beauty	HR@5	0.0206	0.0254	0.0371	0.0364	0.0317	0.0382	0.0527	0.0396	0.0559	**0.0702**	25.58%
	HR@10	0.0332	0.0436	0.0592	0.0583	0.0496	0.0634	0.0739	0.0630	0.0867	**0.0995**	14.76%
	NDCG@5	0.0139	0.0154	0.0233	0.0228	0.0184	0.0244	0.0378	0.0232	0.0331	**0.0500**	32.28%
	NDCG@10	0.0175	0.0212	0.0284	0.0307	0.0268	0.0335	0.0445	0.0307	0.0430	**0.0594**	33.48%
Toys	HR@5	0.0121	0.0205	0.0429	0.0371	0.0269	0.0440	0.0579	0.0503	0.0539	**0.0745**	28.67%
	HR@10	0.0184	0.0333	0.0652	0.0524	0.0483	0.0705	0.0818	0.0736	0.0744	**0.1016**	24.21%
	NDCG@5	0.0077	0.0125	0.0248	0.0259	0.0227	0.0286	0.0408	0.0264	0.0340	**0.0534**	30.88%
	NDCG@10	0.0097	0.0168	0.0320	0.0309	0.0281	0.0369	0.0484	0.0339	0.0406	**0.0622**	28.51%
ML-1M	HR@5	0.0806	0.0912	0.1078	0.1308	0.0953	0.1128	0.1454	0.1142	0.1930	**0.2315**	19.95%
	HR@10	0.1344	0.1442	0.1810	0.2219	0.1645	0.1969	0.2248	0.1815	0.2865	**0.3239**	13.05%
	NDCG@5	0.0475	0.0565	0.0681	0.0804	0.0597	0.0668	0.0856	0.0705	0.1327	**0.1616**	21.78%
	NDCG@10	0.0649	0.0734	0.0918	0.1097	0.0864	0.0950	0.1203	0.0920	0.1586	**0.1915**	20.74%

Experiments

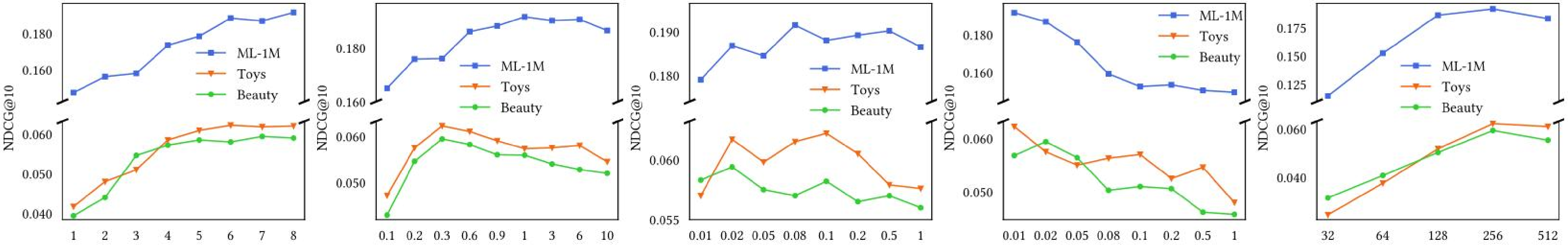
Table 4: Ablation study (NDCG@10) on three datasets. Bold score indicates the performance under the default setting. ↑ indicates the performance better than the default setting.

Architecture	Dataset Beauty	Toys	ML-1M
(1) EMKD(×3)	**0.0594**	**0.0622**	**0.1915**
(2) Remove ICL	0.0529	0.0545	0.1679
(3) Remove CCL	0.0552	0.0560	0.1807
(4) Remove KD	0.0537	0.0571	0.1758
(5) Independent Training	0.0452	0.0484	0.1476
(6) Single Encoder	0.0363	0.0375	0.1183
(7) EMKD(×2)	0.0536	0.0568	0.1792
(8) EMKD(×4)	0.0591	0.0629↑	0.1930↑
(9) Remove AP	0.0578	0.0609	0.1831

Table 5: Performance comparison (NDCG@10) of models with different parameter sizes on three datasets. $*$ indicates the default setting for each model.

	Beauty		Toys		ML-1M	
Architecture	Params.	NDCG@10	Params.	NDCG@10	Params.	NDCG@10
SASRec-2 Layers*	4.69M	0.0284	4.65M	0.0320	2.51M	0.0918
SASRec-4 Layers	6.27M	0.0301	6.23M	0.0313	4.09M	0.0896
SASRec-6 Layers	7.85M	0.0298	7.80M	0.0332	5.67M	0.0857
SASRec-8 Layers	9.43M	0.0279	9.38M	0.0305	7.24M	0.0932
SASRec-10 Layers	11.01M	0.0282	10.96M	0.0310	8.82M	0.0881
BERT4Rec-2 Layers*	7.80M	0.0307	7.71M	0.0309	3.53M	0.1097
BERT4Rec-4 Layers	9.38M	0.0328	9.29M	0.0312	5.11M	0.1113
BERT4Rec-6 Layers	10.96M	0.0332	10.87M	0.0306	6.69M	0.1100
BERT4Rec-8 Layers	12.54M	0.0310	12.45M	0.0298	8.27M	0.1093
BERT4Rec-10 Layers	14.12M	0.0319	14.03M	0.0293	9.85M	0.1099
EMKD(×2)	9.36M	0.0536	9.28M	0.0568	5.08M	0.1792
EMKD(×3)*	14.05M	0.0594	13.91M	0.0622	7.62M	0.1915

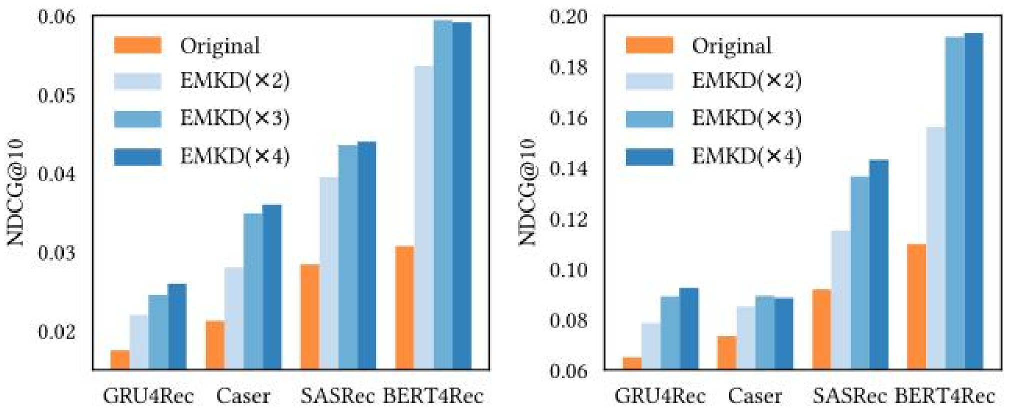
Experiments



(a) Number of Masked Seq. (b) Temperature τ (c) Hyper-parameter λ (d) Hyper-parameter μ (e) Hidden Dimensionality

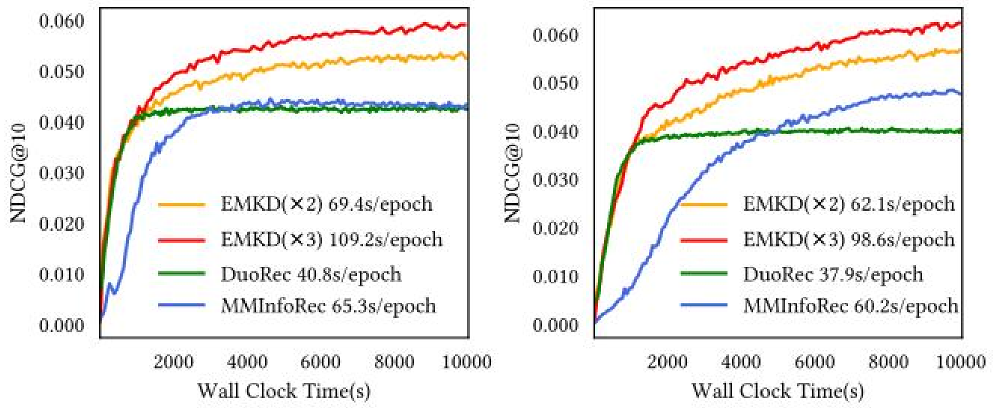
Figure 3: Performance (NDCG@10) comparison w.r.t different hyper-parameters on three datasets.

Experiments



(a) Beauty (b) ML-1M

Figure 4: Performance comparison (NDCG@10) of different models enhanced by EMKD on Beauty and ML-1M datasets. We design three variants for each group of base sequence encoder with 2,3,4 parallel networks respectively.



(a) Beauty (b) Toys

Figure 5: Training efficiency (NDCG@10) on Beauty and Toys datasets. The training speed of EMKD is slightly lower than MMInfoRec, while the convergence speed of EMKD is comparable with DuoRec.

Thanks