# Denoising Self-Attentive Sequential Recommendation

Huiyuan Chen, Yu-San Lin, Menghai Pan, Lan Wang, Michael Yeh, Yan Zheng, Xiaoting Li, Fei Wang, Hao Yang

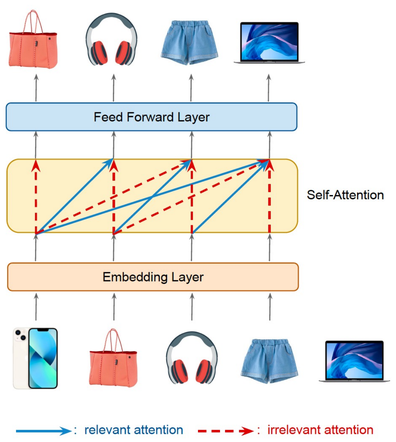
VISA Research

RecSys

## 1. Problem

- **Task: Predict next item, based on a user's historical profile**


- **Challenge: Irrelevant item-item dependencies in the Transformer**




## 2. Proposed Model

- **Differentiable Masks**

$$\mathbf{A}^{(l)} = \text{softmax}\left(\frac{\mathbf{Q}^{(l)}\mathbf{K}^{(l)T}}{\sqrt{d}}\right),$$

$$\mathbf{M}^{(l)} = \mathbf{A}^{(l)} \odot \mathbf{Z}^{(l)}, \quad \mathbf{Z}^{(l)}_{u,v} \sim \text{Bern}(\mathbf{\Pi}^{(l)}_{u,v})$$

$$\text{Attention}(\mathbf{Q}^{(l)}, \mathbf{K}^{(l)}, \mathbf{V}^{(l)}) = \mathbf{M}^{(l)}\mathbf{V}^{(l)},$$

$$\mathcal{R}_M = \sum_{l=1}^{L} \left\|\mathbf{Z}^{(l)}\right\|_0 = \sum_{l=1}^{L}\sum_{u=1}^{n}\sum_{v=1}^{n} \mathbb{I}\left[\mathbf{Z}^{(l)}_{u,v} \neq 0\right].$$

- **Jacobian Regularization**

**The standard dot-product self-attention is not Lipschitz continuous and is vulnerable to the quality of input sequences.**

$$f_i^{(l)}(\mathbf{x}+\boldsymbol{\epsilon}) - f_i^{(l)}(\mathbf{x}) \approx \left[\frac{\partial f_i^{(l)}(\mathbf{x})}{\partial \mathbf{x}}\right]^{\top} \boldsymbol{\epsilon}.$$

Transformer block — Jacobian

$$\mathcal{R}_J = \sum_{l=1}^{L} \|\mathbf{J}^{(l)}\|_F^2.$$

$$\|\mathbf{J}^{(l)}\|_F^2 = \text{Tr}\left(\mathbf{J}^{(l)}\mathbf{J}^{(l)\top}\right) = \mathbb{E}_{\boldsymbol{\eta}\in\mathcal{N}(0,\mathbf{I}_n)}\left[\left\|\boldsymbol{\eta}^{\top}\mathbf{J}^{(l)}\right\|_F^2\right],$$

- **Overall Optimization**

$$\mathcal{L}_{Rec-Denoiser} = \mathcal{L}_{BCE} + \beta \cdot \mathcal{R}_M + \gamma \cdot \mathcal{R}_J,$$

- **Algorithm**

**Algorithm 1:** Rec-denoiser with AR estimator

**Input:** The training sequences $\mathcal{S}$, the number of attention layers $L$, the number of heads $H$, and the regularization coefficients $\alpha$, $\beta$, and $\gamma$.

Initialize model parameters $\mathbf{\Theta}$;
Initialize mask parameters $\mathbf{\Phi}$;
**for** *each mini-batch* **do**
  **for** $l \leftarrow 1$ ***to*** $L$ **do**
    Compute the full attention $\mathbf{A}^{(l)}$ in Eq. (5)
    Draw $\mathbf{U}^{(l)} \sim \text{Uniform}(0, 1)$;
    Compute the mask $\mathbf{Z}^{(l)} = \mathbb{I}[\mathbf{U}^{(l)} < g(\mathbf{\Phi}^{(l)})]$;
    Compute the sparse attention $\mathbf{M}^{(l)} = \mathbf{A}^{(l)} \odot \mathbf{Z}^{(l)}$;
    Feed $\mathbf{M}^{(l)}$ to the rest modules of Transformer;
  **end**
  Compute the loss $\mathcal{L}_{Rec-Denoiser}$ in Eq. (13);
  Update the model parameters $\mathbf{\Theta}$ and mask parameters $\mathbf{\Phi}$ (Eq. (11)) via Stochastic Gradient Descent;
**end**
**Output:** A well-trained Transformer.

- Dataset

Table 1. The statistics of five benchmark datasets.

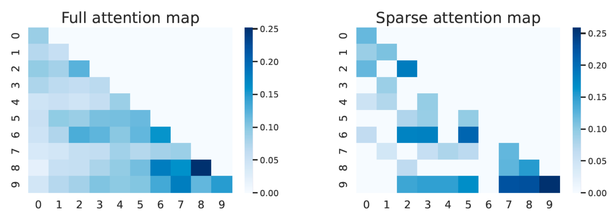
| Dataset | #Users | #Items | Avg actions/user | #Actions |
|---|---|---|---|---|
| MovieLens | 6,040 | 3,416 | 163.5 | 0.987M |
| Beauty | 51,369 | 19,369 | 4.39 | 0.225M |
| Games | 30,935 | 12,111 | 6.46 | 0.2M |
| Movies&TV | 40,928 | 37,564 | 25.55 | 1.05M |
| Steam | 114,796 | 8,648 | 7.58 | 0.87M |

## 3. Experimental Results

- **Overall Performance**

| Dataset | MovieLens | | Beauty | | Games | | Movies&TV | | Steam | |
|---|---|---|---|---|---|---|---|---|---|---|
| Metrics | Hit@10 | NDCG@10 | Hit@10 | NDCG@10 | Hit@10 | NDCG@10 | Hit@10 | NDCG@10 | Hit@10 | NDCG@10 |
| FPMC [36] | 0.7478 | 0.4889 | 0.2810 | 0.1792 | 0.5231 | 0.3410 | 0.4806 | 0.3174 | 0.6012 | 0.4084 |
| GRU4Rec [17] | 0.5582 | 0.3383 | 0.2123 | 0.1205 | 0.2943 | 0.1939 | 0.4210 | 0.2343 | 0.4184 | 0.2687 |
| Caser [39] | 0.7213 | 0.4672 | 0.2670 | 0.1531 | 0.4315 | 0.2652 | 0.4987 | 0.3120 | 0.7137 | 0.4810 |
| SASRec [23] | 0.7434 | 0.5012 | 0.4345 | 0.2765 | 0.6748 | 0.4622 | 0.6521 | 0.4093 | 0.7723 | 0.5514 |
| SASRec+Denoiser | 0.7980 | 0.5610 | 0.4783 | 0.3025 | 0.7391 | 0.5439 | 0.7056 | 0.4718 | 0.8345 | 0.5946 |
| +RI (%) | 7.34% | 11.93% | 10.08% | 9.40% | 9.53% | 17.68% | 8.20% | 15.27% | 5.05% | 7.83% |
| BERT4Rec [38] | 0.7549 | 0.5245 | 0.4528 | 0.3013 | 0.6812 | 0.4815 | 0.6701 | 0.4216 | 0.7901 | 0.5641 |
| BERT4Rec+Denoiser | **0.8045** | **0.5814** | 0.4883 | **0.3348** | **0.7415** | 0.5310 | **0.7212** | **0.4875** | 0.8410 | 0.6223 |
| +RI (%) | 6.57% | 10.85% | 7.84% | 11.12% | 8.85% | 10.28% | 7.63% | 15.63% | 6.45% | 10.32% |
| TiSASRec [27] | 0.7365 | 0.5164 | 0.4532 | 0.2911 | 0.6613 | 0.4517 | 0.6412 | 0.4034 | 0.7704 | 0.5517 |
| TiSASRec+Denoiser | 0.7954 | 0.5582 | **0.4962** | 0.3312 | 0.7331 | 0.4984 | 0.6914 | 0.4671 | **0.8414** | **0.6320** |
| +RI (%) | 7.80% | 8.10% | 9.49% | 13.78% | 10.86% | 10.34% | 7.93% | 15.79% | 9.22% | 14.56% |
| SSE-PT [47] | 0.7413 | 0.5041 | 0.4326 | 0.2731 | 0.6810 | 0.4713 | 0.6378 | 0.4127 | 0.7641 | 0.5703 |
| SSE-PT+Denoiser | 0.8010 | 0.5712 | 0.4952 | 0.3265 | 0.7396 | **0.5152** | 0.6972 | 0.4571 | 0.8310 | 0.6133 |
| +RI (%) | 8.01% | 13.31% | 14.47% | 19.55% | 8.61% | 11.68% | 9.31% | 10.76% | 8.76% | 13.51% |

- **Sparse Attentions**




(a) SASRec (b) SASRec+Denoiser

- **Robustness**

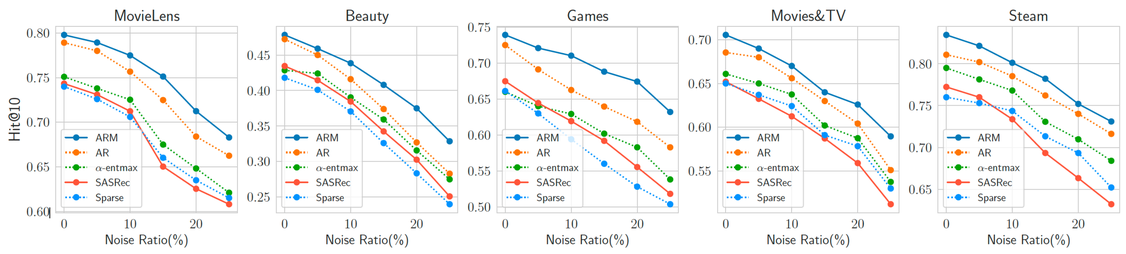



Fig. 3. Overall performance on the training data are corrupted by synthetic noises.

- **Parameter Studies**

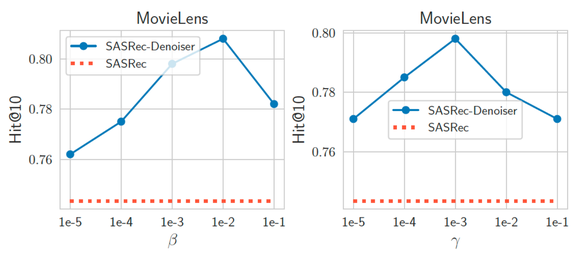



Fig. 5. Effect of regularizers $\beta$ and $\gamma$ on ranking performance (Hit@10).

## 4. Conclusion

- We introduce the idea of denoising item sequences for better of training self-attentive sequential recommenders
- We present a general Rec-Denoiser framework with differentiable masks that can achieve sparse attentions by dynamically pruning irrelevant information .
- We propose an unbiased gradient estimator to optimize the binary masks and apply Jacobian regularization on the gradients of Transformer blocks to further improve its robustness.