Code Available at https://github.com/imcsq/KDD23-G3-RecformerMini

COMP 5331 Project Presentation

Learning Language Representations for Sequential Recommendation

Group No.: 3

Group Members: CHEN Xiao, LI Tsz On, XU Congying, CHEN Songqiang, LU Weiqi, XU Mingshi

(upgrade approved by Raymond)

Project Type: Research ~~Implementation~~ (with much better model efficiency!)

Work to Implement: Li et al., Text Is All You Need: Learning Language Representations for Sequential Recommendation, KDD' 23.

Overview

- **Introduction:** LU, Weiqi
- **Methodology:**
  - Model Architecture: CHEN, Xiao
  - Model Modification: CHEN, Songqiang
  - Learning Framework: XU, Mingshi
- **Evaluation:**
  - Datasets: LU, Weiqi
  - Setup & Overall Perf: XU, Congying
  - Ablation Study: LI, Tsz On

Introduction – Sequential Recommendation

- **Goal:** Model user behavior based on historical interactions.
- **An example:** A user bought MacBook and Mouse is likely to buy a new iPhone in the future.

Item sequence

Introduction – Related Work

- **ID-based methods:**
  - Idea: Learnable embedding tables for item ID encoding.
  - Limitations:
    - Cold start problem of new items.
    - Not transferable to new datasets.

- **Text-based methods:**
  - Idea: Pre-trained language models for item representation based on texts.
  - Limitations:
    - Item representation is sub-optimal for recommendation task.
    - Lack of importance weighting of item attributes.

Introduction – Key Idea & Problem Definition

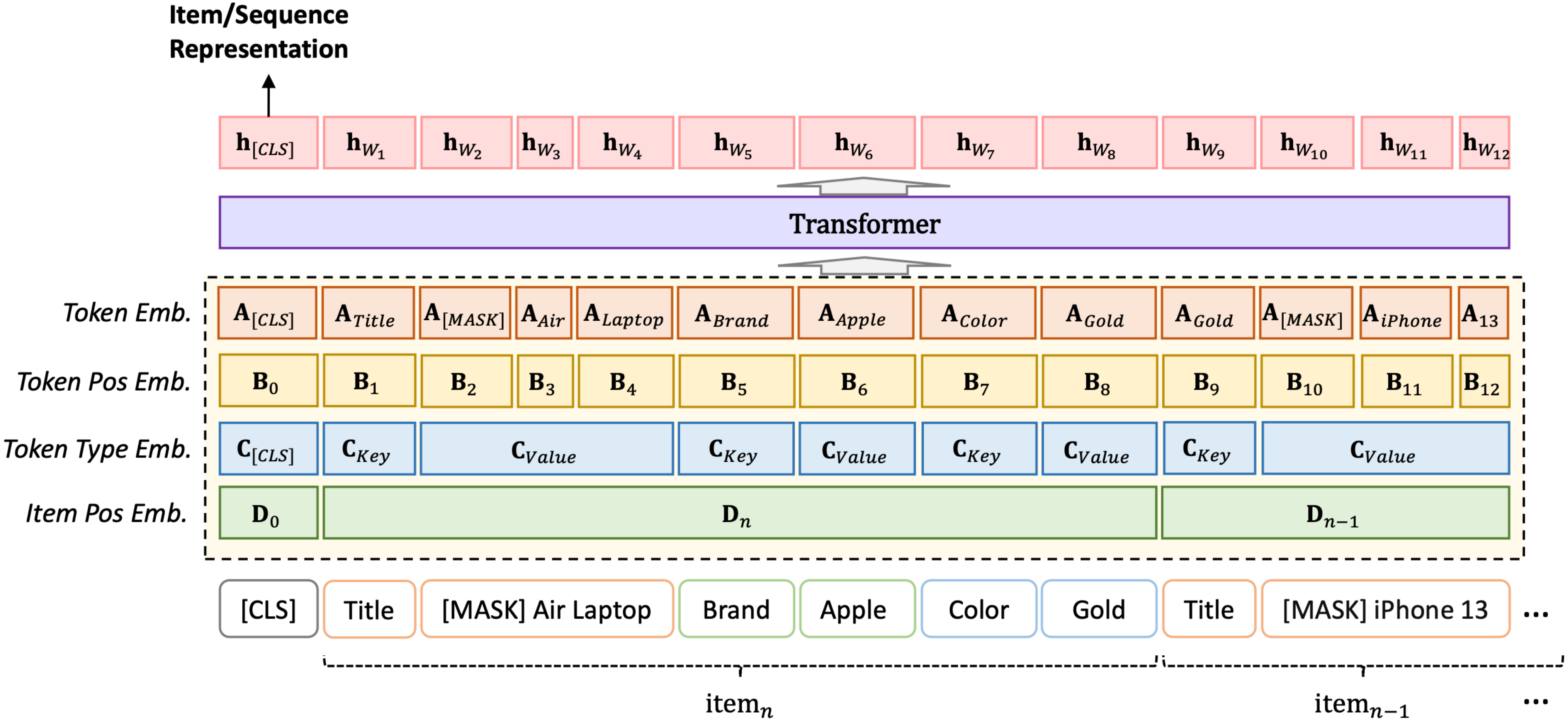
- **Recformer**
- **Main idea:** Leverage the generality of pre-trained language models through joint training of:
  - language understanding
  - sequential recommendations,

  to build a transferable recommendation model.

Introduction – Key Idea & Problem Definition

- Given an item set I and a user's chronological interaction sequence $s = \{i_1, i_2, \dots, i_n\}$, predict the next item based on the sequence s.
  - Each item i_k is described by a dictionary D_k with attribute pairs $\{(k_1, v_1), (k_2, v_2), \dots, (k_m, v_m)\}$.

Methodology – Base Model Architecture & General Workflow



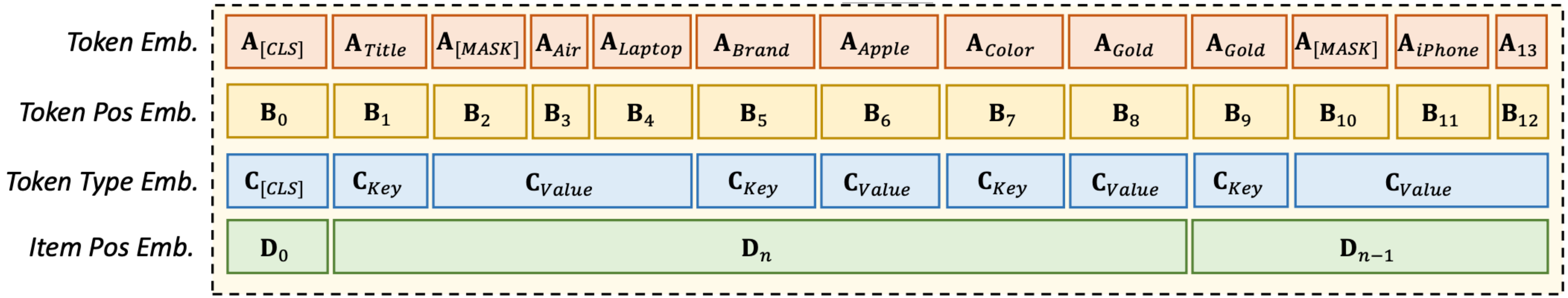
Methodology – Model Inputs

Input Processing Steps:

- **Flatten** the (key, value) pair into sentence for each item
- Generate user's interaction sequence
  - **Concatenate** each item sentence in order
  - **Reverse** the item sentence sequence
- Add a **special token** [CLS] at the beginning

$$X = \{[\mathtt{CLS}], T_n, T_{n-1}, \ldots, T_1\}$$

Methodology – Four Embedding Layers

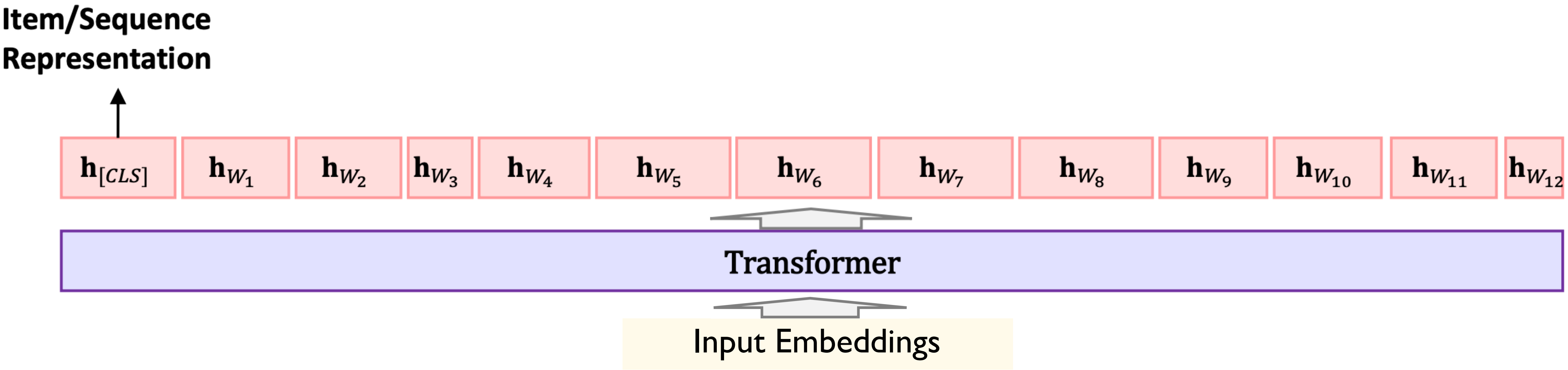


Word Embedding: $\mathbf{E}_w = \mathrm{LayerNorm}(\mathbf{A}_w + \mathbf{B}_w + \mathbf{C}_w + \mathbf{D}_w)$

Model Input Embedding: $\mathbf{E}_X = [\mathbf{E}_{[\mathtt{CLS}]}, \mathbf{E}_{w_1}, \ldots, \mathbf{E}_{w_l}]$

```
(embeddings): RecformerEmbeddings(
  (token_embeddings): Embedding(30522, 256, padding_idx=0)
  (token_position_embeddings): Embedding(1026, 256, padding_idx=0)
  (token_type_embeddings): Embedding(4, 256)
  (item_position_embeddings): Embedding(51, 256)
  (LayerNorm): LayerNorm((256,), eps=1e-05, elementwise_affine=True)
  (dropout): Dropout(p=0.1, inplace=False)
)
```

Methodology – Item/Sequence Representation & Prediction



Item/Sequence Representation:

- $[\mathbf{h}_{\texttt{[CLS]}}, \mathbf{h}_{w_1}, \ldots, \mathbf{h}_{w_l}] = \text{Longformer}([\mathbf{E}_{\texttt{[CLS]}}, \mathbf{E}_{w_1}, \ldots, \mathbf{E}_{w_l}])$
- The first token $h_{[CLS]}$ is used as the sequence representation.

Prediction:

- Given a user's interaction sequence s, and a next item i.
- Calculate score: cosine similarity $r_{i,s} = \frac{\mathbf{h}_i^\top \mathbf{h}_s}{\|\mathbf{h}_i\| \cdot \|\mathbf{h}_s\|}$
- Predict next item: highest score $i_s = \text{argmax}_{i \in \mathcal{I}}(r_{i,s})$

Methodology – Efficiency Enhancement Beyond Reimplementation

- **Efficiency Problem of the Original Model Architecture & Setup:**

  > The model is huge and takes too much GPU memory and time to run.

	Longformer Size	Max Input Length	Batch Size	Pre-Training		Fine-Tuning	
				GPU Mem	Time/Epoch	GPU Mem	Time/Epoch
Original:	Base	1024	16	33,660 MB	49.5 hours	25,656 MB	1800+ sec

* 256 epochs
= 128h = 5.3 days
(*6 datasets)

- **Reasons to Enhance the Model Efficiency:**

  > We **need to run** the model!

  > We need to do **many** experiments in the limited time!

  > We will contribute a more practical implementation to the proposed methodology. [WE DID IT!]

Methodology – Efficiency Enhancement Beyond Reimplementation

- **Modification 1: Reduce the Maximum Input Length**
  > Truncate the input by 512 (original: 1024) tokens.
  > The fewer tokens input to the model, the **fewer hidden states & calculations** are needed.

Longformer Size	Max Input Length	Batch Size	Pre-Training		Fine-Tuning	
			GPU Mem	Time/Epoch	GPU Mem	Time/Epoch
Base	1024	16	33,660 MB	49.5 hours	25,656 MB	1800+ sec
Base	**512**	16	19,958 MB	41 hours	19,966 MB	1600+ sec

Runnable now.

* 256 epochs = 128h = 4.7 days
Time Cost is Still Intolerant.

Methodology – Efficiency Enhancement Beyond Reimplementation

- **Modification 2: Substitute the Model Architecture**

  # The <u>LM Longformer-Base is huge</u>.

  > Longformer-Base v.s. -Mini:

  > *12 v.s. 6 Transformer layers;*

  > *768d v.s. 256d hidden state;*

  > *GPT (bigger vocab) v.s. BERT (smaller vocab) tokenizers.*

  > 12x768d v.s. 6x256d hidden states per sample.

  > 6 v.s. 1 storage & calculation.

```
Recformer-Base(
  (embeddings): RecformerEmbeddings( # param: 41,794,560
    (word_embeddings): Embedding(50265, 768)
    (position_embeddings): Embedding(4098, 768)
    (token_type_embeddings): Embedding(4, 768)
    (item_position_embeddings): Embedding(51, 768)
    (LayerNorm): LayerNorm((768,))
  )
  (encoder): LongformerEncoder( # LongFormer-Base
    (layer): ModuleList(
      (0): LongformerLayer( # param: 8,859,648 x 12 ls = 106,315,776
        (attention): LongformerAttention(
          (self): LongformerSelfAttention(
            (query): Linear(in_features=768, out_features=768)
            (key): Linear(in_features=768, out_features=768)
            (value): Linear(in_features=768, out_features=768)
            (query_global): Linear(in_features=768, out_features=768)
            (key_global): Linear(in_features=768, out_features=768)
            (value_global): Linear(in_features=768, out_features=768)
          )
          (output): LongformerSelfOutput(
            (dense): Linear(in_features=768, out_features=768)
            (LayerNorm): LayerNorm((768,))
          )
        )
        (intermediate): LongformerIntermediate(
          (dense): Linear(in_features=768, out_features=3072)
        )
        (output): LongformerOutput(
          (dense): Linear(in_features=3072, out_features=768)
          (LayerNorm): LayerNorm((768,))
        )
      )
      (1): LongformerLayer(...)
      ...
      (11): LongformerLayer(...)
    )
  )
  (pooler): RecformerPooler()
)
```

```
Recformer-Mini(
  (embeddings): RecformerEmbeddings( # param: 8,090,880
    (word_embeddings): Embedding(30522, 256)
    (position_embeddings): Embedding(1026, 256)
    (token_type_embeddings): Embedding(4, 256)
    (item_position_embeddings): Embedding(51, 256)
    (LayerNorm): LayerNorm((256,))
  )
  (encoder): LongformerEncoder( # LongFormer-Mini
    (layer): ModuleList(
      (0): LongformerLayer( # param: 987,136 x 6 layers = 5,922,816
        (attention): LongformerAttention(
          (self): LongformerSelfAttention(
            (query): Linear(in_features=256, out_features=256)
            (key): Linear(in_features=256, out_features=256)
            (value): Linear(in_features=256, out_features=256)
            (query_global): Linear(in_features=256, out_features=256)
            (key_global): Linear(in_features=256, out_features=256)
            (value_global): Linear(in_features=256, out_features=256)
          )
          (output): LongformerSelfOutput(
            (dense): Linear(in_features=256, out_features=256)
            (LayerNorm): LayerNorm((256,))
          )
        )
        (intermediate): LongformerIntermediate(
          (dense): Linear(in_features=256, out_features=1024)
        )
        (output): LongformerOutput(
          (dense): Linear(in_features=1024, out_features=256)
          (LayerNorm): LayerNorm((256,))
        )
      )
      (1): LongformerLayer(...)
      ...
      (5): LongformerLayer(...)
    )
  )
  (pooler): RecformerPooler()
)
```

Longformer Size	Max Input Length	Batch Size	Pre-Training GPU Mem	Pre-Training Time/Epoch	Fine-Tuning GPU Mem	Fine-Tuning Time/Epoch
Base	1024	16	33,660 MB	49.5 hours	25,656 MB	1800+ sec
Base	512	16	19,958 MB	41 hours	19,966 MB	1600+ sec
Mini	512	16	5,932 MB ~1/6	14.5 hours ~1/3	6,060 MB <1/4	502 sec <1/3

Methodology – Efficiency Enhancement Beyond Reimplementation

- **Last Modification: Increase the Batch Size**
  > To fully utilize the GPU memory and computation resource left.
  > Increase the batch size for higher speedup & more stable convergence guidance.

Longformer Size	Max Input Length	Batch Size	Pre-Training		Fine-Tuning	
			GPU Mem	Time/Epoch	GPU Mem	Time/Epoch
Base	1024	16	33,660 MB	49.5 hours	25,656 MB	1800+ sec
Base	512	16	19,958 MB	41 hours	19,966 MB	1600+ sec
Mini	512	16	5,932 MB	14.5 hours	6,060 MB	502 sec
Mini	512	80	20,248 MB	5 hours ~1/10	17,552 MB	312 sec ~1/6

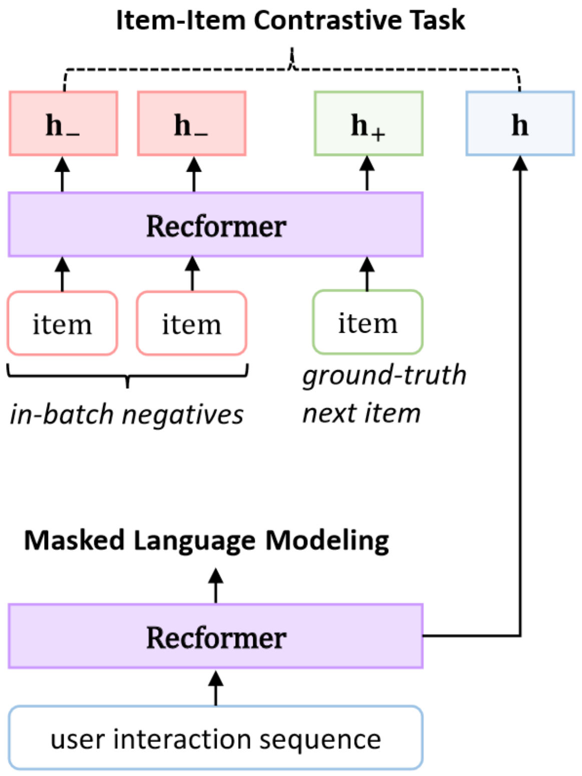
* 256 epochs
= 128h = 5.3 days
(*6 datasets)

* 256 epochs
= 128h = 0.9 days
(*6 datasets)

SO OBVIOUS!

Performance? No hurry.

Methodology – Pretraining (Masked Language Modeling Task)



(b) Pretraining

Goal of Pretraining:

- obtain a parameter initialization for downstream tasks

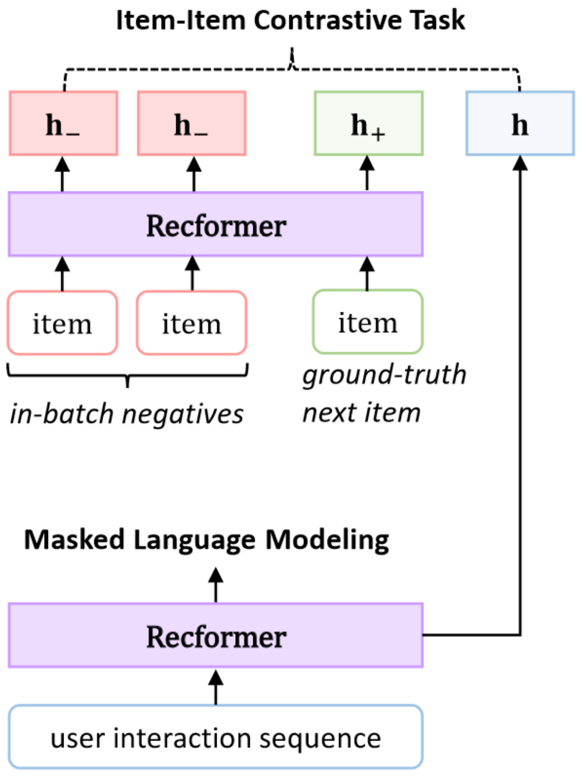
Masked Language Modeling (following BERT)**:**

- the training data generator chooses 15% of the token positions at random for prediction.
  (1) the [MASK] with probability 80%;
  (2) a random token with probability 10%;
  (3) the unchanged token with probability 10%.
- **Loss Function of MLM:**

$$\mathcal{L}_{\mathrm{MLM}} = -\sum_{i=0}^{|\mathcal{V}|} y_i \log(p_i)$$

- Prevent language models from forgetting the word semantics
- Eliminate the language domain gap between a general language corpus and item texts.

Methodology – Pretraining (Item-Item Contrastive Task)



(b) Pretraining

We adopt in-batch next items as negative instances instead of negative sampling to accelerate the Pre-training process.

- **Similarity Function:**

$$r_{i,s} = \frac{\mathbf{h}_i^\top \mathbf{h}_s}{\|\mathbf{h}_i\| \cdot \|\mathbf{h}_s\|}$$

- **Loss Function of IIC:**

$$\mathcal{L}_{\mathrm{IIC}} = -\log \frac{e^{\mathrm{sim}(\mathbf{h}_s,\mathbf{h}_i^+)/\tau}}{\sum_{i\in\mathcal{B}} e^{\mathrm{sim}(\mathbf{h}_s,\mathbf{h}_i)/\tau}}$$

Loss Function for Pre-training:

$$\mathcal{L}_{\mathrm{PT}} = \mathcal{L}_{\mathrm{IIC}} + \lambda \cdot \mathcal{L}_{\mathrm{MLM}}$$

Methodology – Finetuning

Algorithm 1: Two-Stage Finetuning

1 **Input**: $D_{\text{train}}, D_{\text{valid}}, \mathcal{I}, M$
2 **Hyper-parameters**: n_{epoch}
3 **Output**: $M', \mathbf{I}'$

1: $M \leftarrow$ initialized with pre-trained parameters
2: $p \leftarrow$ metrics are initialized with 0
Stage 1
3: **for** n in n_{epoch} **do**
4: $\quad \mathbf{I} \leftarrow \text{Encode}(M, \mathcal{I})$
5: $\quad M \leftarrow \text{Train}(M, \mathbf{I}, D_{\text{train}})$
6: $\quad p' \leftarrow \text{Evaluate}(M, \mathbf{I}, D_{\text{valid}})$
7: $\quad$ **if** $p' > p$ **then**
8: $\qquad M', \mathbf{I}' \leftarrow M, \mathbf{I}$
9: $\qquad p \leftarrow p'$
10: $\quad$ **end if**
11: **end for**
Stage 2
12: $M \leftarrow M'$
13: **for** n in n_{epoch} **do**
14: $\quad M \leftarrow \text{Train}(M, \mathbf{I}', D_{\text{train}})$
15: $\quad p' \leftarrow \text{Evaluate}(M, \mathbf{I}', D_{\text{valid}})$
16: $\quad$ **if** $p' > p$ **then**
17: $\qquad M' \leftarrow M$
18: $\qquad p \leftarrow p'$
19: $\quad$ **end if**
20: **end for**
21: **return** $M', \mathbf{I}'$

Item Feature Matrix: $\mathbf{I} \in \mathbb{R}^{|\mathcal{I}| \times d}$

Item Feature Matrix $\mathbf{I}$ is obtained by encoding all items with Recformer

- **Stage 1:** Updating $\mathbf{I}$ by encoding all items with Recformer (line4) per epoch.
  - The reason is although we have pre-trained the model, the representation of the item can still be improved by further training on the downstream dataset.
  - To accelerate the training, we update the $\mathbf{I}$ **every epoch.**
- **Stage 2:** Freeze $\mathbf{I}$ and update only parameters in model M.

Loss Function for Finetuning:

$$\mathcal{L}_{\text{FT}} = -\log \frac{e^{\text{sim}(\mathbf{h}_s, \mathbf{I}_i^+)/\tau}}{\sum_{i \in \mathcal{I}} e^{\text{sim}(\mathbf{h}_s, \mathbf{I}_i)/\tau}}$$

Evaluation – Datasets

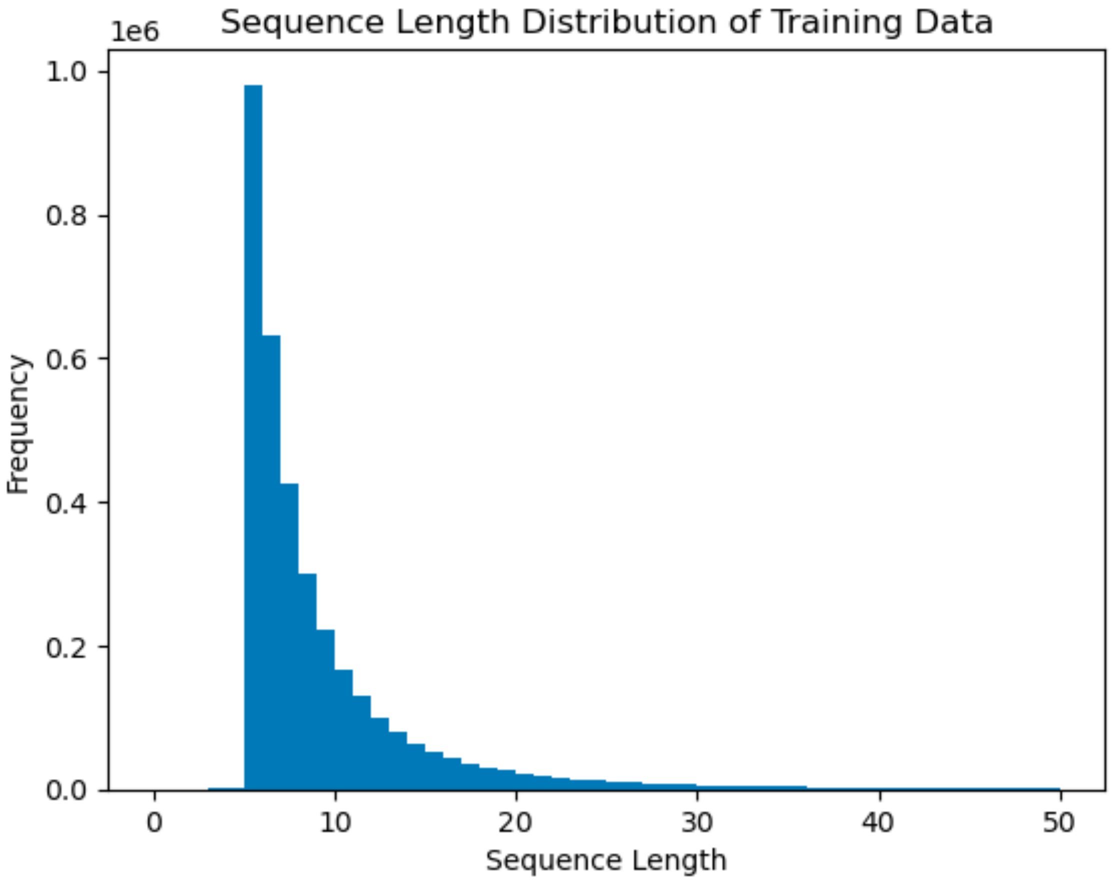
- **The Amazon Review Dataset (2018)**
  - Scope: Features 233.1 million product reviews spanning from May 1996 to October 2018.
  - Rich Metadata: Includes product descriptions, brands, categories, and image features.
  - Significance: Provides a comprehensive view of user preferences over time.


```
7138258879: {'title': 'Elite Mailers 9&quot;x2&quot; i-VTEC SOHC Vinyl Decal Sticker -
White - 2 pieces', 'brand': 'Elite Mailers', 'category': 'Automotive Exterior
Accessories Bumper Stickers, Decals & Magnets'}
```


The meta data of an item with detailed description (title), brand, and fine-grained category

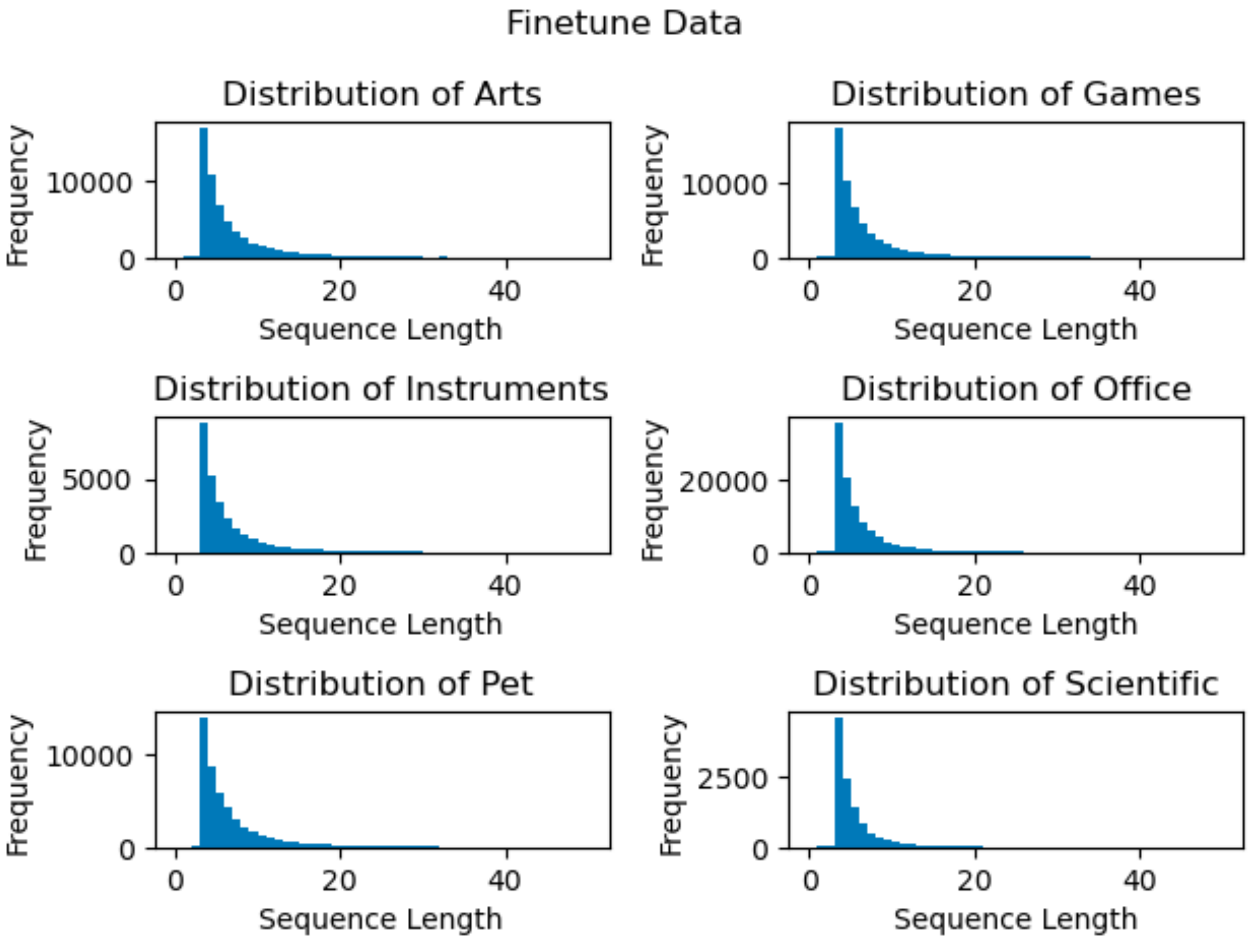
Evaluation – Datasets

- **Dataset for Pre-training**
- Sampled from seven categories:
  - "Automotive"
  - "Cell Phones and Accessories"
  - "Clothing Shoes and Jewelry"
  - "Electronics"
  - "Grocery and Gourmet Food"
  - "Home and Kitchen"
  - "Movies and TV"
- Size: 3.5 millions of interaction sequences.



Evaluation – Datasets

- **Six Datasets (Size) for Fine-tuning**
  - "Arts, Crafts and Sewing" (56210)
  - "Industrial and Scientific" (11041)
  - "Musical Instruments" (27530)
  - "Office Products" (101501)
  - "Pet Supplies" (47569)
  - "Video Games" (55223)



Evaluation – Setup

- **Metrics**

  > Recall: measures the proportion of relevant items

  > MRR: measures the position of the first relevant item

  > NDCG: measures both the relevance and the position of the items

- **Baselines**

  > ID-only methods: GRU4Rec (2015), SASRec (2018), BERT4Rec (2019), and RecGRU (2021)

  > Text-only methods: ZESRec (2021) and UniSRec (2022).

  > ID-Text methods: FDSA (2019) and S3-Rec (2020)

Evaluation – Overall performance

- **Motivation: larger model = better model ?**

NO

Dataset	Metric	GRU4Rec	SASRec	BERT4Rec	RecGRU	FDSA	S3-Rec	ZESRec	UniSRec	Recformer	Recformer -mini
Scientific	NDCG@10	0.0826	0.0797	0.0790	0.0575	0.0716	0.0451	0.0843	0.0862	0.1027	0.1040
	Recall@10	0.1055	0.1305	0.1061	0.0781	0.0967	0.0804	0.1260	0.1255	0.1448	0.1451
	MRR	0.0702	0.0696	0.0759	0.0566	0.0692	0.0392	0.0745	0.0786	0.0951	0.0967
Instruments	NDCG@10	0.0633	0.0634	0.0707	0.0468	0.0731	0.0797	0.0694	0.0785	0.0830	0.0805
	Recall@10	0.0969	0.0995	0.0972	0.0617	0.1006	0.1110	0.1078	0.1119	0.1052	0.1034
	MRR	0.0707	0.0577	0.0677	0.0460	0.0748	0.0755	0.0633	0.0740	0.0807	0.0780
Arts	NDCG@10	0.1075	0.0848	0.0942	0.0525	0.0994	0.1026	0.0970	0.0894	0.1252	0.1179
	Recall@10	0.1317	0.1342	0.1236	0.0742	0.1209	0.1399	0.1349	0.1333	0.1614	0.1539
	MRR	0.1041	0.0742	0.0899	0.0488	0.0941	0.1057	0.0870	0.0798	0.1189	0.1113
Office	NDCG@10	0.0761	0.0832	0.0972	0.0500	0.0922	0.0911	0.0865	0.0919	0.1141	0.1114
	Recall@10	0.1053	0.1196	0.1205	0.0647	0.1285	0.1186	0.1199	0.1262	0.1403	0.1405
	MRR	0.0731	0.0751	0.0932	0.0483	0.0972	0.0957	0.0797	0.0848	0.1089	0.1055
Games	NDCG@10	0.0586	0.0547	0.0628	0.0386	0.0600	0.0532	0.0530	0.0580	0.0684	0.0637
	Recall@10	0.0988	0.0953	0.1029	0.0479	0.0931	0.0879	0.0844	0.0923	0.1039	0.0989
	MRR	0.0539	0.0505	0.0585	0.0396	0.0546	0.0500	0.0505	0.0552	0.0650	0.0601
Pet	NDCG@10	0.0648	0.0569	0.0602	0.0366	0.0673	0.0742	0.0754	0.0702	0.0972	0.0958
	Recall@10	0.0781	0.0881	0.0765	0.0415	0.0949	0.1039	0.1018	0.0933	0.1162	0.1161
	MRR	0.0632	0.0507	0.0585	0.0371	0.0650	0.0710	0.0706	0.0650	0.0940	0.0922

Light green:
outperforming all baselines

Dark green:
further outperforming Recformer

> Recformer-mini almost outperforms all baselines across datasets and metrics.

> Recformer-mini achieves comparable results to Recformer.

Evaluation – Zero-shot performance

- **Motivation: evaluate the knowledge transferability in recommendation scenarios**

  > Measure the contribution of pre-training on downstream tasks.

Dataset	Metric	pretrain only	pretrain & fine-tune	Con.	Dataset	Metric	pretrain only	pretrain & fine-tune	Con.
Scientific	NDCG@10	0.0823	0.1040	79%	Office	NDCG@10	0.0476	0.1114	43%
	Recall@10	0.1259	0.1451	87%		Recall@10	0.0767	0.1405	55%
	MRR	0.0734	0.0967	76%		MRR	0.0417	0.1055	40%
Instruments	NDCG@10	0.0436	0.0805	54%	Games	NDCG@10	0.0426	0.0637	67%
	Recall@10	0.0700	0.1034	68%		Recall@10	0.0685	0.0989	69%
	MRR	0.0395	0.0780	51%		MRR	0.0386	0.0601	64%
Arts	NDCG@10	0.0692	0.1179	59%	Pet	NDCG@10	0.0523	0.0958	55%
	Recall@10	0.1153	0.1539	75%		Recall@10	0.0771	0.1161	66%
	MRR	0.0591	0.1113	53%		MRR	0.0468	0.0922	51%

Dark green: significant contribution

> Recformer-mini generally contributes 40%+ ~ 70%+ across datasets and evaluation metrics.
> On the "Scientific", the contribution is up to 87%.
> Finding: Recformer-mini can transfer learned knowledge in pre-training to new domains or tasks.

Evaluation – Motivation of Alation Study

- To evaluate the effectiveness of Recformers' components
- Conduct an ablation study with 4 extra model setups (variants)

Evaluation – Ablative Model Variants

Variants	Has training stage	Has fine-tuning stage	Item Embedding for training	Item Embedding for FT
Original	✓	✓	Fix	Trainable
Fix	✓	✓	Fix	Fix
Variable	✓	✓	Trainable	Fix
Training-only	✓	✗	Fix	--
Fine-tuning–only	✗	✓	--	Trainable

Evaluation – Ablation Experiment Setup

- **Subjects**: "Industrial and Scientific", "Musical Instruments", "Arts, Crafts and Sewing", "Office Products", "Video Games", and "Pet".

- **Environment**: NVIDIA GeForce RTX 3090 GPU cards.
- **Training configuration**: 5 epochs for pre-training, and 20 epochs for fine-tuning.

Evaluation – Ablation Experiment Result

Table 4: Ablation Study with Different Variants of Recformer
(Best performance is in green)

Dataset	Metric	Original	Fix	Variable	Training-only	FT–only
Scientific	NDCG@10	0.0986	0.0984	0.0986	0.0867	0.0294
	Recall@10	0.1400	0.1394	0.1407	0.1321	0.0413
	MRR	0.0906	0.0903	0.0903	0.0767	0.0392
Instruments	NDCG@10	0.0660	0.0625	0.0666	0.0470	0.0162
	Recall@10	0.0902	0.0870	0.0911	0.0787	0.0314
	MRR	0.0624	0.0588	0.0627	0.0410	0.0143
Arts	NDCG@10	0.1003	0.0789	0.0966	0.0783	0.0536
	Recall@10	0.1462	0.1238	0.1464	0.1230	0.0880
	MRR	0.0900	0.0684	0.0855	0.0689	0.0470
Office	NDCG@10	0.0892	0.0530	0.0892	0.0538	0.0887
	Recall@10	0.1260	0.0831	0.1248	0.0843	0.1275
	MRR	0.0806	0.0473	0.0810	0.0467	0.0797
Games	NDCG@10	0.0556	0.0831	0.0562	0.0538	0.0195
	Recall@10	0.0863	0.0633	0.0865	0.0843	0.0375
	MRR	0.0524	0.0397	0.0533	0.0400	0.0191
Pet	NDCG@10	0.0886	0.0447	0.0878	0.0443	0.0363
	Recall@10	0.1109	0.0734	0.1097	0.0727	0.0463
	MRR	0.0841	0.0575	0.0834	0.0572	0.0352

Finding 1: Original or Variable always outperform other variants.

Evaluation – Ablation Experiment Result

Table 4: Ablation Study with Different Variants of Recformer
(Best performance is in green)

Dataset	Metric	Original	Fix	Variable	Training-only	FT–only
Scientific	NDCG@10	0.0986	0.0984	0.0986	0.0867	0.0294
	Recall@10	0.1400	0.1394	0.1407	0.1321	0.0413
	MRR	0.0906	0.0903	0.0903	0.0767	0.0392
Instruments	NDCG@10	0.0660	0.0625	0.0666	0.0470	0.0162
	Recall@10	0.0902	0.0870	0.0911	0.0787	0.0314
	MRR	0.0624	0.0588	0.0627	0.0410	0.0143
Arts	NDCG@10	0.1003	0.0789	0.0966	0.0783	0.0536
	Recall@10	0.1462	0.1238	0.1464	0.1230	0.0880
	MRR	0.0900	0.0684	0.0855	0.0689	0.0470
Office	NDCG@10	0.0892	0.0530	0.0892	0.0538	0.0887
	Recall@10	0.1260	0.0831	0.1248	0.0843	0.1275
	MRR	0.0806	0.0473	0.0810	0.0467	0.0797
Games	NDCG@10	0.0556	0.0831	0.0562	0.0538	0.0195
	Recall@10	0.0863	0.0633	0.0865	0.0843	0.0375
	MRR	0.0524	0.0397	0.0533	0.0400	0.0191
Pet	NDCG@10	0.0886	0.0447	0.0878	0.0443	0.0363
	Recall@10	0.1109	0.0734	0.1097	0.0727	0.0463
	MRR	0.0841	0.0575	0.0834	0.0572	0.0352

Finding 2: Original/Fix/Variable always outperform Training-only or FT-only

Evaluation – Ablation Experiment Result

Table 4: Ablation Study with Different Variants of Recformer
(Best performance is in green)

Dataset	Metric	Original	Fix	Variable	Training-only	FT–only
Scientific	NDCG@10	0.0986	0.0984	0.0986	0.0867	0.0294
	Recall@10	0.1400	0.1394	0.1407	0.1321	0.0413
	MRR	0.0906	0.0903	0.0903	0.0767	0.0392
Instruments	NDCG@10	0.0660	0.0625	0.0666	0.0470	0.0162
	Recall@10	0.0902	0.0870	0.0911	0.0787	0.0314
	MRR	0.0624	0.0588	0.0627	0.0410	0.0143
Arts	NDCG@10	0.1003	0.0789	0.0966	0.0783	0.0536
	Recall@10	0.1462	0.1238	0.1464	0.1230	0.0880
	MRR	0.0900	0.0684	0.0855	0.0689	0.0470
Office	NDCG@10	0.0892	0.0530	0.0892	0.0538	0.0887
	Recall@10	0.1260	0.0831	0.1248	0.0843	0.1275
	MRR	0.0806	0.0473	0.0810	0.0467	0.0797
Games	NDCG@10	0.0556	0.0831	0.0562	0.0538	0.0195
	Recall@10	0.0863	0.0633	0.0865	0.0843	0.0375
	MRR	0.0524	0.0397	0.0533	0.0400	0.0191
Pet	NDCG@10	0.0886	0.0447	0.0878	0.0443	0.0363
	Recall@10	0.1109	0.0734	0.1097	0.0727	0.0463
	MRR	0.0841	0.0575	0.0834	0.0572	0.0352

Finding 3: Finding 1 or Finding 2 are applicable to Arts, Office, Games or Pet.

Thanks for Listening!
Q&A?

Group No.: 3

Group Members: CHEN Xiao, LI Tsz On, XU Congying, CHEN Songqiang, LU Weiqi, XU Mingshi

Project Type: Implementation-Oriented (with some modification)

Work to Implement: Li et al., Text Is All You Need: Learning Language Representations for Sequential Recommendation, KDD' 23.

(BACKUP – PARAMETER TUNING)

Dataset	Metric	GRU4Rec	SASRec	BERT4Rec	RecGRU	FDSA	S3-Rec	ZESRec	UniSRec	Recformer	l512ptlr1e-5 ftlr1e-4	l512ptlr1e-5 ftlr5e-5	l512ptlr5e-5 ftlr5e-5	l1024ptlr5e-5 ftlr5e-5	l512ptlr5e-5 ftlr1e-4
Scientific	NDCG@10	0.0826	0.0797	0.0790	0.0575	0.0716	0.0451	0.0843	0.0862	0.1027	0.1040	0.1034	0.1013	0.1016	0.1024
	Recall@10	0.1055	0.1305	0.1061	0.0781	0.0967	0.0804	0.1260	0.1255	0.1448	0.1451	0.1420	0.1414	0.1451	0.1459
	MRR	0.0702	0.0696	0.0759	0.0566	0.0692	0.0392	0.0745	0.0786	0.0951	0.0967	0.0968	0.0942	0.0932	0.0943
Instruments	NDCG@10	0.0633	0.0634	0.0707	0.0468	0.0731	0.0797	0.0694	0.0785	0.0830	0.0805	0.0785	0.0788	0.0765	0.0801
	Recall@10	0.0969	0.0995	0.0972	0.0617	0.1006	0.1110	0.1078	0.1119	0.1052	0.1034	0.1031	0.1029	0.0995	0.1030
	MRR	0.0707	0.0577	0.0677	0.0460	0.0748	0.0755	0.0633	0.0740	0.0807	0.0780	0.0754	0.0758	0.0741	0.0776
Arts	NDCG@10	0.1075	0.0848	0.0942	0.0525	0.0994	0.1026	0.0970	0.0894	0.1252	0.1179	0.1129	0.1076	0.1155	0.1147
	Recall@10	0.1317	0.1342	0.1236	0.0742	0.1209	0.1399	0.1349	0.1333	0.1614	0.1539	0.1519	0.1449	0.1532	0.1530
	MRR	0.1041	0.0742	0.0899	0.0488	0.0941	0.1057	0.0870	0.0798	0.1189	0.1113	0.1054	0.1001	0.1084	0.1073
Office	NDCG@10	0.0761	0.0832	0.0972	0.0500	0.0922	0.0911	0.0865	0.0919	0.1141	0.1114	0.0974	0.1008	0.0986	0.1089
	Recall@10	0.1053	0.1196	0.1205	0.0647	0.1285	0.1186	0.1199	0.1262	0.1403	0.1405	0.1354	0.1368	0.1348	0.1394
	MRR	0.0731	0.0751	0.0932	0.0483	0.0972	0.0957	0.0797	0.0848	0.1089	0.1055	0.0884	0.0925	0.0902	0.1025
Games	NDCG@10	0.0586	0.0547	0.0628	0.0386	0.0600	0.0532	0.0530	0.0580	0.0684	0.0637	0.0635	0.0628	0.0628	0.0639
	Recall@10	0.0988	0.0953	0.1029	0.0479	0.0931	0.0879	0.0844	0.0923	0.1039	0.0989	0.0976	0.0971	0.0972	0.0982
	MRR	0.0539	0.0505	0.0585	0.0396	0.0546	0.0500	0.0505	0.0552	0.0650	0.0601	0.0602	0.0594	0.0594	0.0603
Pet	NDCG@10	0.0648	0.0569	0.0602	0.0366	0.0673	0.0742	0.0754	0.0702	0.0972	0.0958	0.0938	0.0940	0.0947	0.0966
	Recall@10	0.0781	0.0881	0.0765	0.0415	0.0949	0.1039	0.1018	0.0933	0.1162	0.1161	0.1160	0.1156	0.1166	0.1169
	MRR	0.0632	0.0507	0.0585	0.0371	0.0650	0.0710	0.0706	0.0650	0.0940	0.0922	0.0896	0.0900	0.0905	0.0929

Longformer Size	Max Input Length	Batch Size	Pre-Training		Fine-Tuning	
			GPU Mem	Time/Epoch	GPU Mem	Time/Epoch
Base	1024	16	33,660 MB	49.5 hours	25,656 MB	1800+ sec
Base	512	8	18,634 MB	**66+ hours**	13,640 MB	**2100 sec**