Contrastive Learning with Frequency-Domain Interest Trends for Sequential Recommendation

Team 5
Maharsh Raval 210108026
Priyanshu Srivastava 210108035
Harsh Diwaker 210108015

INTRODUCTION AND PROBLEM STATEMENT

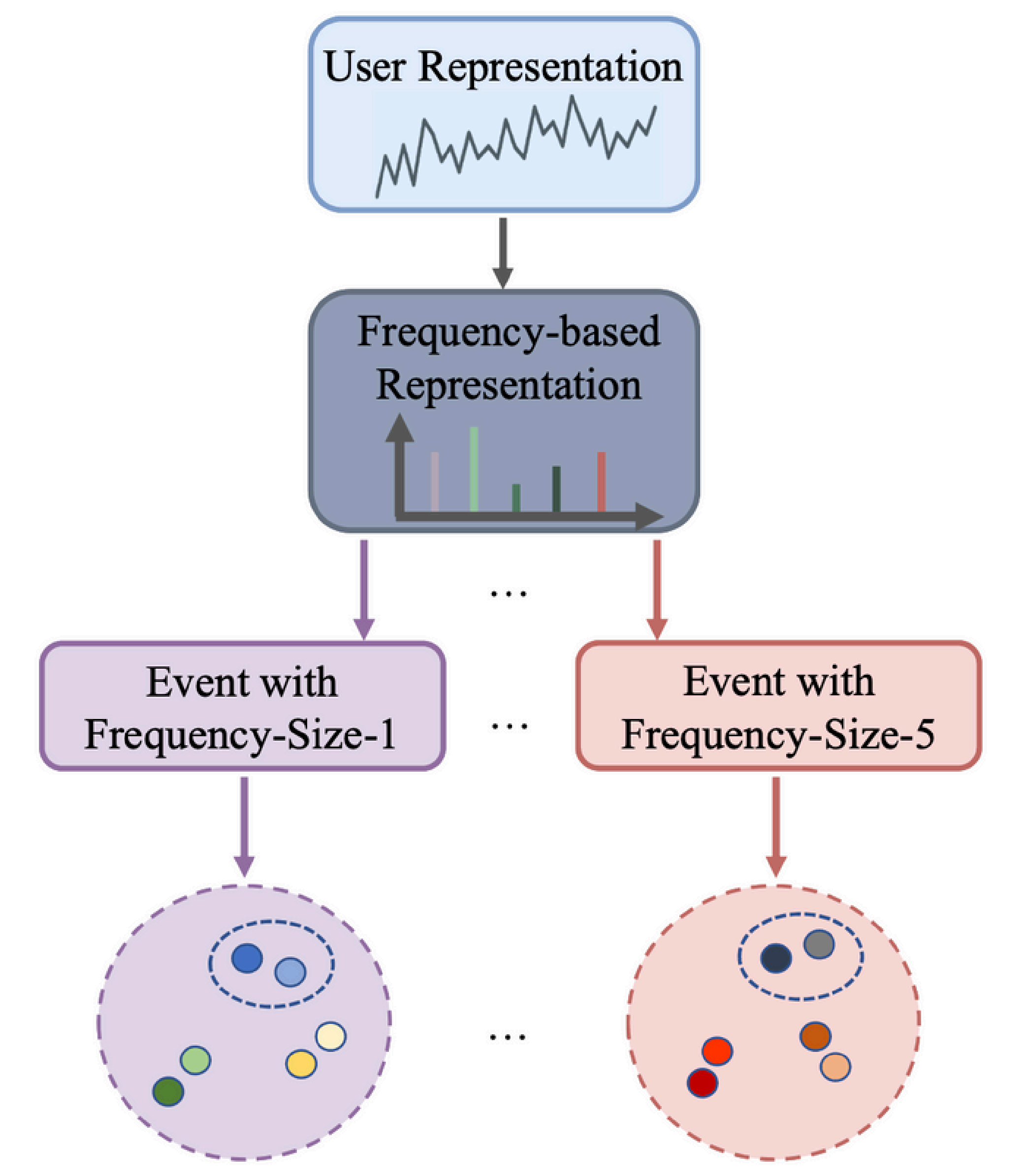
Sequential recommendation systems aim to predict what a user will interact with next, based on their past interactions. Traditional methods such as CNNs, RNNs, and Transformers are commonly used to model these patterns. However, these methods face significant challenges in understanding **fast-changing user preferences** and **noisy interactions.** Most existing models only perform well in low-frequency tasks, meaning they handle gradual changes in user behavior but struggle with sudden, high-frequency interest shifts.

Additionally, the time-domain data these models rely on is **unsuited** for capturing these **abrupt changes.** We address these challenges by introducing a novel approach that constructs augmented samples in the **frequency domain**, using contrastive learning to model both slow and rapid interest changes better. The proposed **CFIT4SRec** model offers a more flexible and effective way to capture complex user behavior trends by focusing on the frequency domain rather than the time domain.

CONTRASTIVE LEARNING

Contrastive learning has been widely adopted in fields like computer vision and natural language processing due to its ability to generate high-quality self-supervised representations. In recommendation systems, contrastive learning enhances models by **maximizing** the similarity between **positive pairs** (sequences with similar meanings) and **minimizing** the similarity between **negative pairs** (dissimilar sequences).

We apply contrastive learning in a novel way by generating self-supervised signals from the frequency domain instead of raw sequential data. The focus is on improving the model's ability to handle both **noisy** and **sparse data**, which are common in real-world recommendation systems.


User Representation
Frequency-based Representation
Event with Frequency-Size-1
Event with Frequency-Size-5
...

(a) Contrastive learning for different frequency events.

FREQUENCY-DOMAIN ANALYSIS USING FOURIER TRANSFORM

A central aspect of this paper is the application of the Fourier Transform to convert user interaction data from the time domain to the frequency domain. By doing so, the model can differentiate between low-frequency components (smooth, gradual trends) and high-frequency components (sudden, rapid changes). This transformation allows for a more detailed analysis of user behaviour patterns.

The paper introduces the concept of treating **user interaction sequences** as **"images"** and applies a **second-order Fourier Transform** to generate self-supervised learning signals. This method captures both spatiotemporal trends and variations between user attributes.

$$F(m, z) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi\left(\frac{mx}{M} + \frac{zy}{N}\right)}$$

AUGMENTATION TECHNIQUES

The proposed CFIT4SRec model builds upon the **Transformer architecture**, a sequence encoder widely known for its effectiveness in recommendation systems. The CFIT4SRec model enhances the Transformer with frequency-domain transformations, introducing three types of augmentations:

- **Low-pass augmentation**: Focuses on smooth trends by filtering out high-frequency noise.
- **High-pass augmentation**: Captures sudden interest changes by emphasizing high-frequency components.
- **Band-stop augmentation**: Randomly removes certain frequency components, forcing the model to generalize better and avoid overfitting to specific patterns.

The architecture integrates these frequency-based augmentations into the learning process through multi-task learning, where the model learns both traditional sequential recommendation tasks and contrastive learning tasks simultaneously.

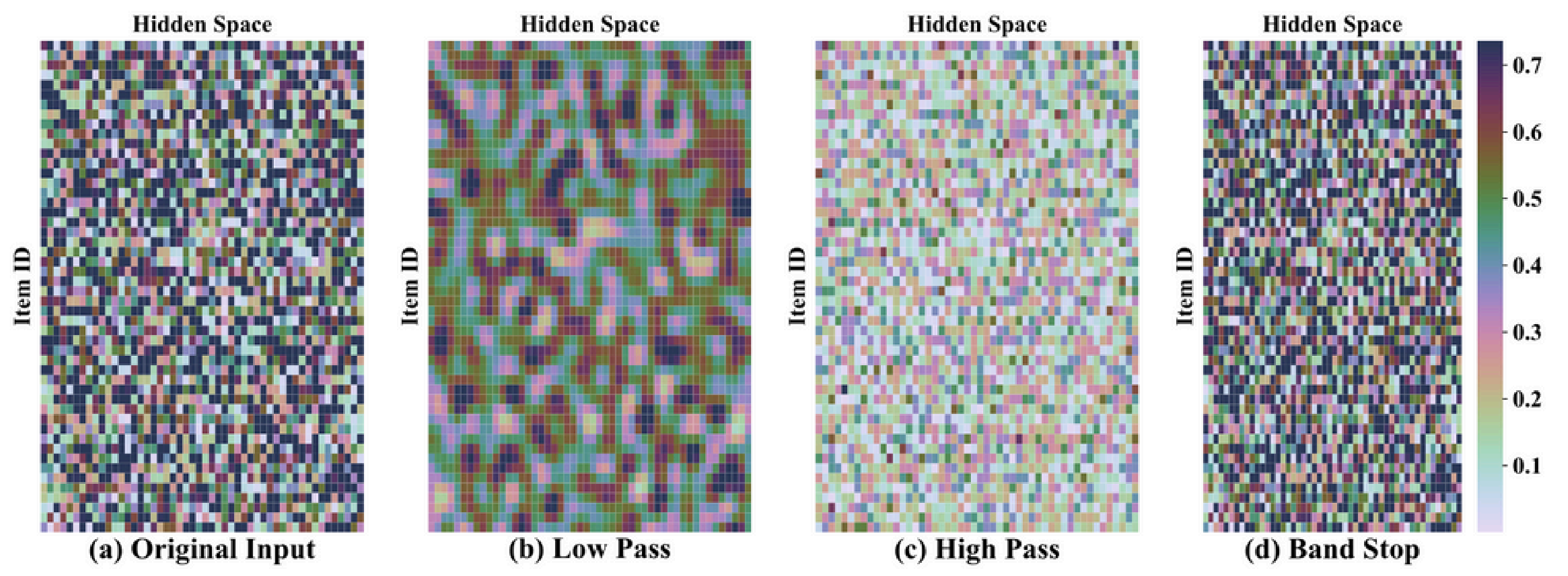


Figure 2: Visualization of three different frequency sizes, including low frequency, high frequency and band stop.

MODEL FLOW

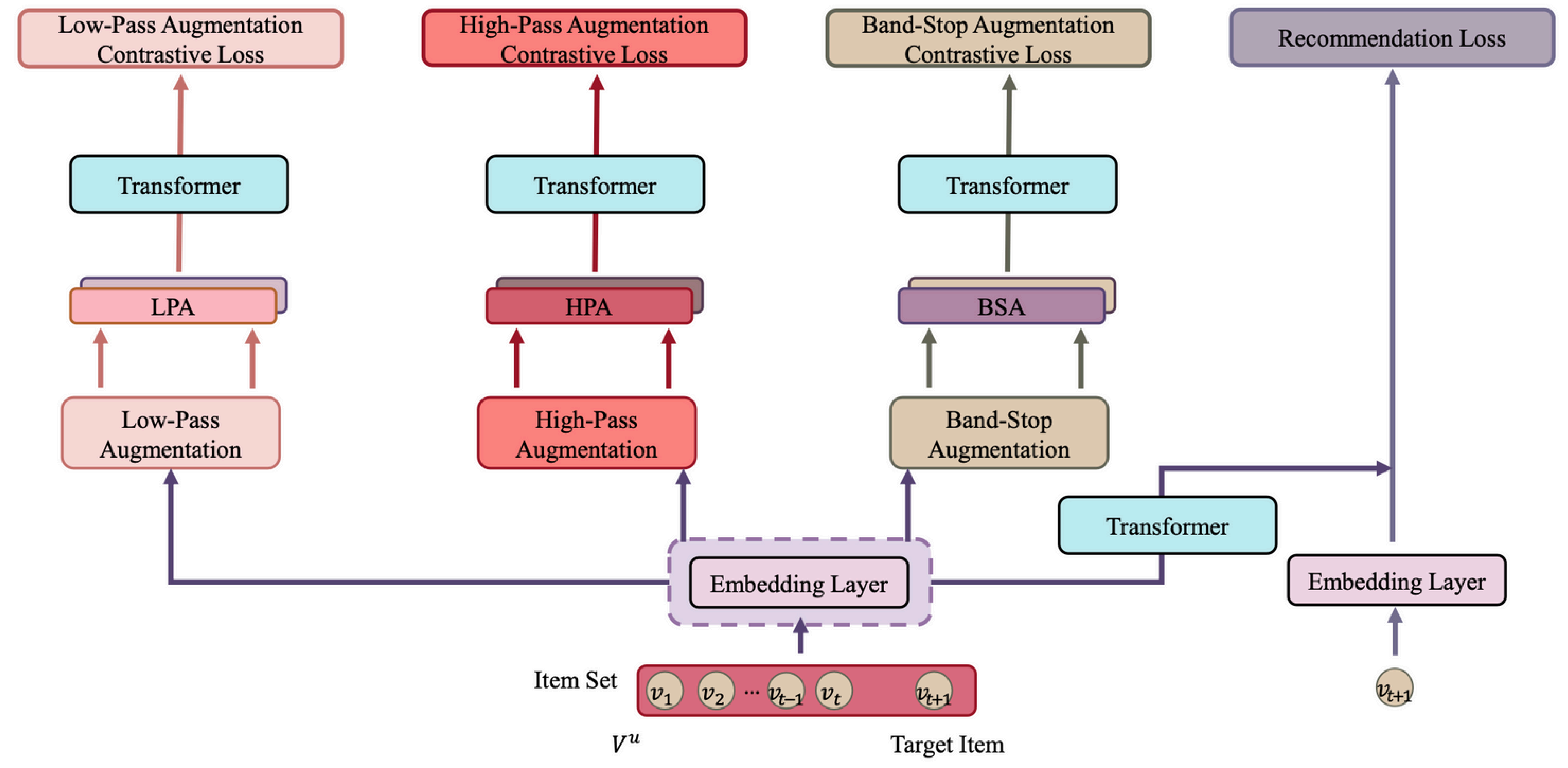


Figure 3: The overall architecture of the proposed CFIT4SRec model.

1. Data Representation: User Interaction Sequences

The model starts by gathering **historical user interaction data**. Each user’s interactions with items (such as products or movies) over time are collected as sequences. These sequences are the core input for the recommendation system, representing the user’s behavior.
For example:

- The user u’s interaction sequence might look like:

$$S^u = \left[v_1^u, v_2^u, \cdots, v_{|S^u|}^u\right]$$

These sequences are treated as time-domain data, which shows how user interests change over time.

2. Embedding Layer: Converting Items to Representations

The model then uses an **embedding layer** to map the raw user-item interaction data (ID) into a **d-dimensional hidden space with an Embedding matrix**. This means that each item is translated into a meaningful vector, representing its attributes in a way the model can understand.
For instance, an item "Item 1" could be converted to a vector like [0.23, -0.45, 0.13, ...].
This embedding layer also adds **position encoding** to preserve the **sequence order**, which ensures that the model knows in which order the items interacted.

3. Transforming Sequences to Frequency Domain

The innovation in the CFIT4SRec model lies in how it processes the interaction sequences. Instead of working directly with time-domain data, the model uses the Fourier Transform to convert these sequences into the frequency domain.

- The **Fourier Transform** separates the sequence into its low-frequency components **(gradual trends)** and high-frequency components **(sudden changes in behavior).**
- By representing the sequence in the frequency domain, the model can better understand both steady, long-term interests and abrupt short-term shifts in user preferences.

4. Data Augmentation: Enhancing the Training Process

Once the sequences are transformed into the frequency domain, the model applies three types of data augmentations. These augmentations generate variations of the data to improve the model’s learning:

- **Low-pass Augmentation (LPA)**: Focuses on retaining the smooth, gradual changes in user behaviour **(low-frequency components)**. Filters out noise and focuses on long-term interest trends.
- **High-pass Augmentation (HPA)**: Focuses on sudden, fast-evolving behaviour **(high-frequency components)**. Captures abrupt changes in interests or spikes in user behaviour.
- **Band-stop Augmentation (BSA)**: Randomly **removes some components** (frequencies) to prevent overfitting. It helps the model generalize better by introducing variation in the trends it learns from.

These augmentations essentially create multiple versions of the user data, each emphasizing different aspects of their behaviour. This variety helps the model learn more robust representations of user interests.

5. Contrastive Learning: Training with Positive and Negative Pairs

The model uses contrastive learning, a technique that involves comparing positive pairs and negative pairs of data:

- Positive pairs are different augmented views of the same user's behavior (e.g., an original sequence and a low-pass augmented version of it).
- Negative pairs are sequences from different users or behaviors that are unrelated.
- The model is trained to maximize the **similarity(Here, dot product)** between positive pairs and minimize the similarity between negative pairs.

For instance:
A positive pair could be two sequences from User A, one in its original form and another augmented.
A negative pair could be one sequence from User A and another from User B, where behaviors differ.

6. Recommendation Learning: Predicting the Next Item

In addition to contrastive learning, the CFIT4SRec model is also trained to perform the traditional recommendation task:
Given a user's past interactions, the model predicts what item they are most likely to interact with next.
This is done by combining the user's representation learned from both contrastive learning and recommendation learning. The model uses a **softmax layer** to calculate the probability of each item being the next one in the sequence and **recommends** the item with the **highest probability**.

7. Multi-task Learning: Combining Both Objectives

CFIT4SRec employs a multi-task learning framework that integrates both contrastive learning and recommendation learning. Contrastive learning enhances the model's ability to distinguish between positive and negative user behavior patterns by comparing augmented views of sequences. Recommendation learning focuses on predicting the next item a user will interact with, based on the learned user representations.

The model's final loss function combines the **contrastive loss**, which strengthens behavior pattern recognition, and the **recommendation loss (Here, Binary Cross Entropy Loss)**, which ensures accurate item predictions. By training both tasks together, the model effectively balances learning user behavior nuances while improving its recommendation accuracy, making it more robust and adaptable to fast-evolving user preferences.

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{rec}} + \lambda \mathcal{L}_{\text{ssl}} + \gamma \|\Theta\|_2^2$$

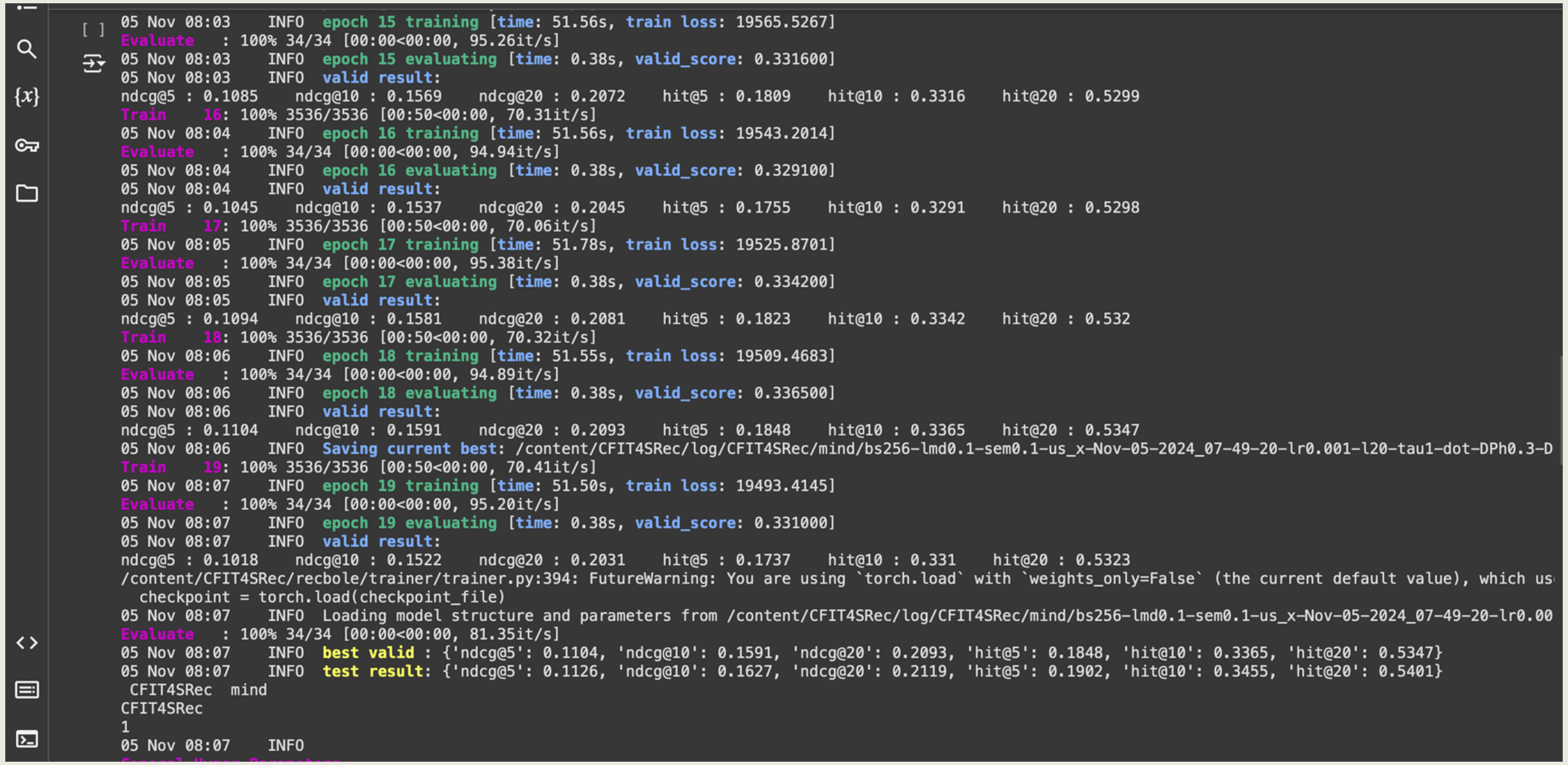
8. Inference: Making Predictions in Real Time

Once the model is trained, it can be used to make real-time recommendations. When a new user interaction sequence is fed into the model, the Transformer-based encoder processes the sequence, applies the learned augmentations, and uses both contrastive learning and recommendation learning to **predict** what the **user will interact with next**.
For example:

- If a user has recently interacted with items [Item 5, Item 7, Item 9], the model will predict the next likely interaction, such as Item 12, based on both the gradual and sudden interest trends.

EXPERIMENTATION

The code was ran on amazon dataset{ beauty, baby }, MIND(Microsoft News), ML-1M



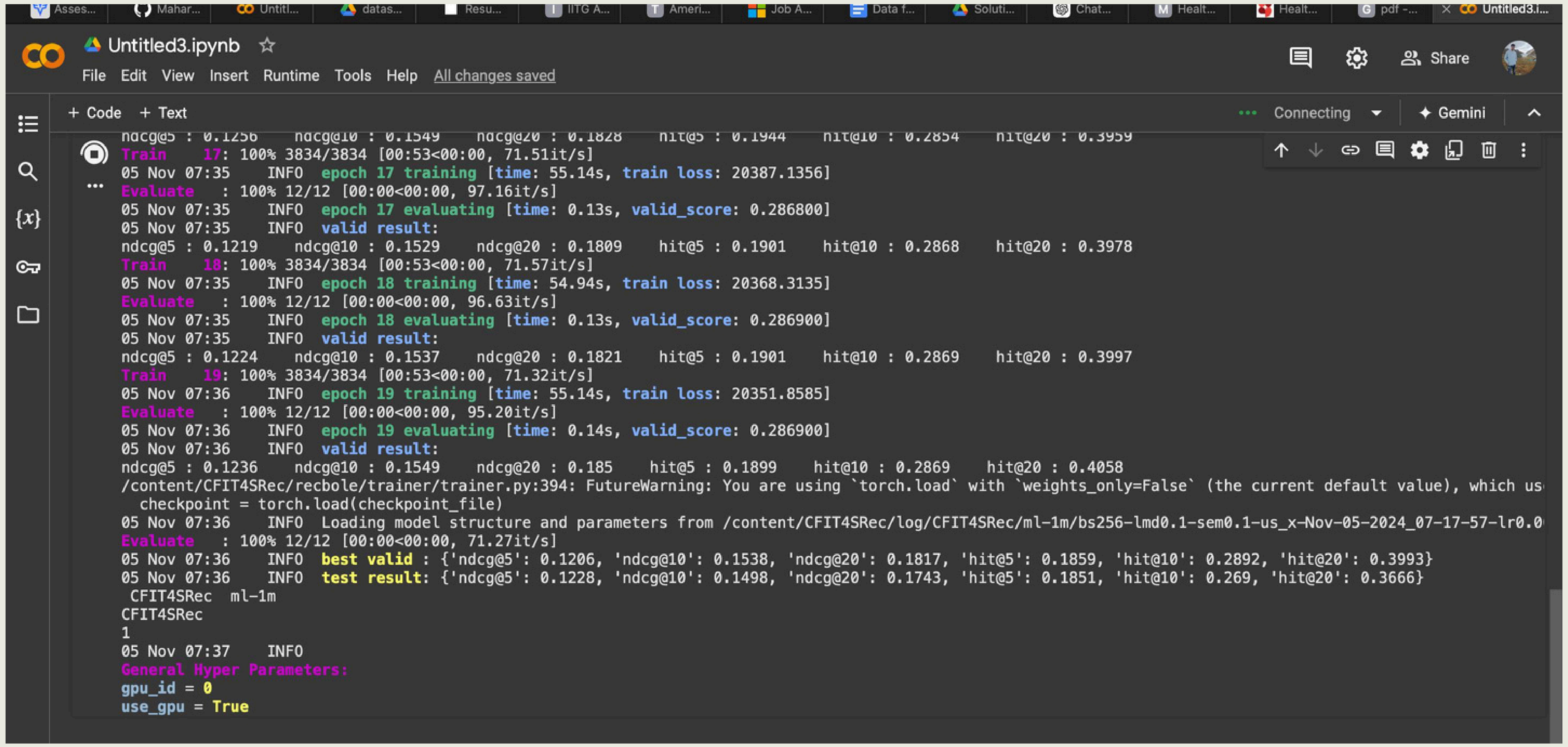
```
05 Nov 08:03    INFO  epoch 15 training [time: 51.56s, train loss: 19565.5267]
Evaluate   : 100% 34/34 [00:00<00:00, 95.26it/s]
05 Nov 08:03    INFO  epoch 15 evaluating [time: 0.38s, valid_score: 0.331600]
05 Nov 08:03    INFO  valid result:
ndcg@5 : 0.1085    ndcg@10 : 0.1569    ndcg@20 : 0.2072    hit@5 : 0.1809    hit@10 : 0.3316    hit@20 : 0.5299
Train    16: 100% 3536/3536 [00:50<00:00, 70.31it/s]
05 Nov 08:04    INFO  epoch 16 training [time: 51.56s, train loss: 19543.2014]
Evaluate   : 100% 34/34 [00:00<00:00, 94.94it/s]
05 Nov 08:04    INFO  epoch 16 evaluating [time: 0.38s, valid_score: 0.329100]
05 Nov 08:04    INFO  valid result:
ndcg@5 : 0.1045    ndcg@10 : 0.1537    ndcg@20 : 0.2045    hit@5 : 0.1755    hit@10 : 0.3291    hit@20 : 0.5298
Train    17: 100% 3536/3536 [00:50<00:00, 70.06it/s]
05 Nov 08:05    INFO  epoch 17 training [time: 51.78s, train loss: 19525.8701]
Evaluate   : 100% 34/34 [00:00<00:00, 95.38it/s]
05 Nov 08:05    INFO  epoch 17 evaluating [time: 0.38s, valid_score: 0.334200]
05 Nov 08:05    INFO  valid result:
ndcg@5 : 0.1094    ndcg@10 : 0.1581    ndcg@20 : 0.2081    hit@5 : 0.1823    hit@10 : 0.3342    hit@20 : 0.532
Train    18: 100% 3536/3536 [00:50<00:00, 70.32it/s]
05 Nov 08:06    INFO  epoch 18 training [time: 51.55s, train loss: 19509.4683]
Evaluate   : 100% 34/34 [00:00<00:00, 94.89it/s]
05 Nov 08:06    INFO  epoch 18 evaluating [time: 0.38s, valid_score: 0.336500]
05 Nov 08:06    INFO  valid result:
ndcg@5 : 0.1104    ndcg@10 : 0.1591    ndcg@20 : 0.2093    hit@5 : 0.1848    hit@10 : 0.3365    hit@20 : 0.5347
05 Nov 08:06    INFO  Saving current best: /content/CFIT4SRec/log/CFIT4SRec/mind/bs256-lmd0.1-sem0.1-us_x-Nov-05-2024_07-49-20-lr0.001-l20-tau1-dot-DPh0.3-D
Train    19: 100% 3536/3536 [00:50<00:00, 70.41it/s]
05 Nov 08:07    INFO  epoch 19 training [time: 51.50s, train loss: 19493.4145]
Evaluate   : 100% 34/34 [00:00<00:00, 95.20it/s]
05 Nov 08:07    INFO  epoch 19 evaluating [time: 0.38s, valid_score: 0.331000]
05 Nov 08:07    INFO  valid result:
ndcg@5 : 0.1018    ndcg@10 : 0.1522    ndcg@20 : 0.2031    hit@5 : 0.1737    hit@10 : 0.331    hit@20 : 0.5323
/content/CFIT4SRec/recbole/trainer/trainer.py:394: FutureWarning: You are using `torch.load` with `weights_only=False` (the current default value), which us
  checkpoint = torch.load(checkpoint_file)
05 Nov 08:07    INFO  Loading model structure and parameters from /content/CFIT4SRec/log/CFIT4SRec/mind/bs256-lmd0.1-sem0.1-us_x-Nov-05-2024_07-49-20-lr0.00
Evaluate   : 100% 34/34 [00:00<00:00, 81.35it/s]
05 Nov 08:07    INFO  best valid : {'ndcg@5': 0.1104, 'ndcg@10': 0.1591, 'ndcg@20': 0.2093, 'hit@5': 0.1848, 'hit@10': 0.3365, 'hit@20': 0.5347}
05 Nov 08:07    INFO  test result: {'ndcg@5': 0.1126, 'ndcg@10': 0.1627, 'ndcg@20': 0.2119, 'hit@5': 0.1902, 'hit@10': 0.3455, 'hit@20': 0.5401}
 CFIT4SRec  mind
CFIT4SRec
1
05 Nov 08:07    INFO
```

MIND(Microsoft News)
Dataset results:

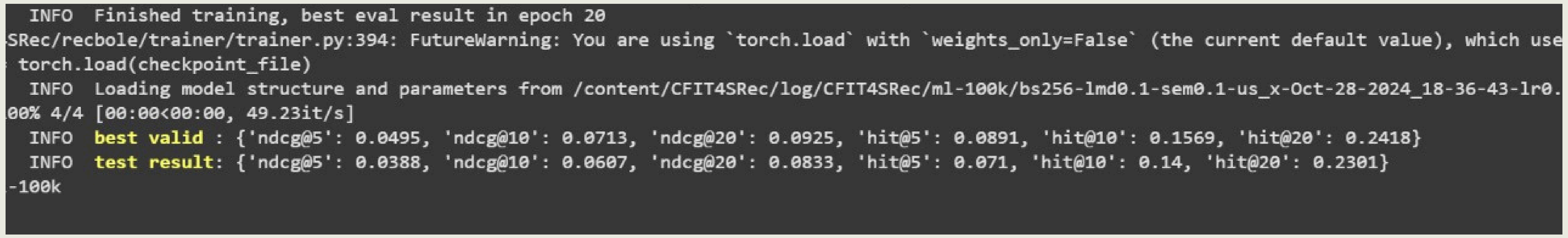
ndcg@5: 0.1126
ndcg@10: 0.1627
hit@5: 0.1902
hit@10: 0.3455

EXPERIMENTATION

The code was ran on amazon dataset{ beauty, baby }, MIND(Microsoft News), ML-1M



```
ndcg@5 : 0.1256    ndcg@10 : 0.1549    ndcg@20 : 0.1828    hit@5 : 0.1944    hit@10 : 0.2854    hit@20 : 0.3959
Train    17: 100% 3834/3834 [00:53<00:00, 71.51it/s]
05 Nov 07:35    INFO  epoch 17 training [time: 55.14s, train loss: 20387.1356]
Evaluate   : 100% 12/12 [00:00<00:00, 97.16it/s]
05 Nov 07:35    INFO  epoch 17 evaluating [time: 0.13s, valid_score: 0.286800]
05 Nov 07:35    INFO  valid result:
ndcg@5 : 0.1219    ndcg@10 : 0.1529    ndcg@20 : 0.1809    hit@5 : 0.1901    hit@10 : 0.2868    hit@20 : 0.3978
Train    18: 100% 3834/3834 [00:53<00:00, 71.57it/s]
05 Nov 07:35    INFO  epoch 18 training [time: 54.94s, train loss: 20368.3135]
Evaluate   : 100% 12/12 [00:00<00:00, 96.63it/s]
05 Nov 07:35    INFO  epoch 18 evaluating [time: 0.13s, valid_score: 0.286900]
05 Nov 07:35    INFO  valid result:
ndcg@5 : 0.1224    ndcg@10 : 0.1537    ndcg@20 : 0.1821    hit@5 : 0.1901    hit@10 : 0.2869    hit@20 : 0.3997
Train    19: 100% 3834/3834 [00:53<00:00, 71.32it/s]
05 Nov 07:36    INFO  epoch 19 training [time: 55.14s, train loss: 20351.8585]
Evaluate   : 100% 12/12 [00:00<00:00, 95.20it/s]
05 Nov 07:36    INFO  epoch 19 evaluating [time: 0.14s, valid_score: 0.286900]
05 Nov 07:36    INFO  valid result:
ndcg@5 : 0.1236    ndcg@10 : 0.1549    ndcg@20 : 0.185    hit@5 : 0.1899    hit@10 : 0.2869    hit@20 : 0.4058
/content/CFIT4SRec/recbole/trainer/trainer.py:394: FutureWarning: You are using `torch.load` with `weights_only=False` (the current default value), which us
  checkpoint = torch.load(checkpoint_file)
05 Nov 07:36    INFO  Loading model structure and parameters from /content/CFIT4SRec/log/CFIT4SRec/ml-1m/bs256-lmd0.1-sem0.1-us_x-Nov-05-2024_07-17-57-lr0.0
Evaluate   : 100% 12/12 [00:00<00:00, 71.27it/s]
05 Nov 07:36    INFO  best valid : {'ndcg@5': 0.1206, 'ndcg@10': 0.1538, 'ndcg@20': 0.1817, 'hit@5': 0.1859, 'hit@10': 0.2892, 'hit@20': 0.3993}
05 Nov 07:36    INFO  test result: {'ndcg@5': 0.1228, 'ndcg@10': 0.1498, 'ndcg@20': 0.1743, 'hit@5': 0.1851, 'hit@10': 0.269, 'hit@20': 0.3666}
 CFIT4SRec  ml-1m
CFIT4SRec
1
05 Nov 07:37    INFO
General Hyper Parameters:
gpu_id = 0
use_gpu = True
```



```
 INFO  Finished training, best eval result in epoch 20
SRec/recbole/trainer/trainer.py:394: FutureWarning: You are using `torch.load` with `weights_only=False` (the current default value), which use
 torch.load(checkpoint_file)
 INFO  Loading model structure and parameters from /content/CFIT4SRec/log/CFIT4SRec/ml-100k/bs256-lmd0.1-sem0.1-us_x-Oct-28-2024_18-36-43-lr0.
00% 4/4 [00:00<00:00, 49.23it/s]
 INFO  best valid : {'ndcg@5': 0.0495, 'ndcg@10': 0.0713, 'ndcg@20': 0.0925, 'hit@5': 0.0891, 'hit@10': 0.1569, 'hit@20': 0.2418}
 INFO  test result: {'ndcg@5': 0.0388, 'ndcg@10': 0.0607, 'ndcg@20': 0.0833, 'hit@5': 0.071, 'hit@10': 0.14, 'hit@20': 0.2301}
-100k
```

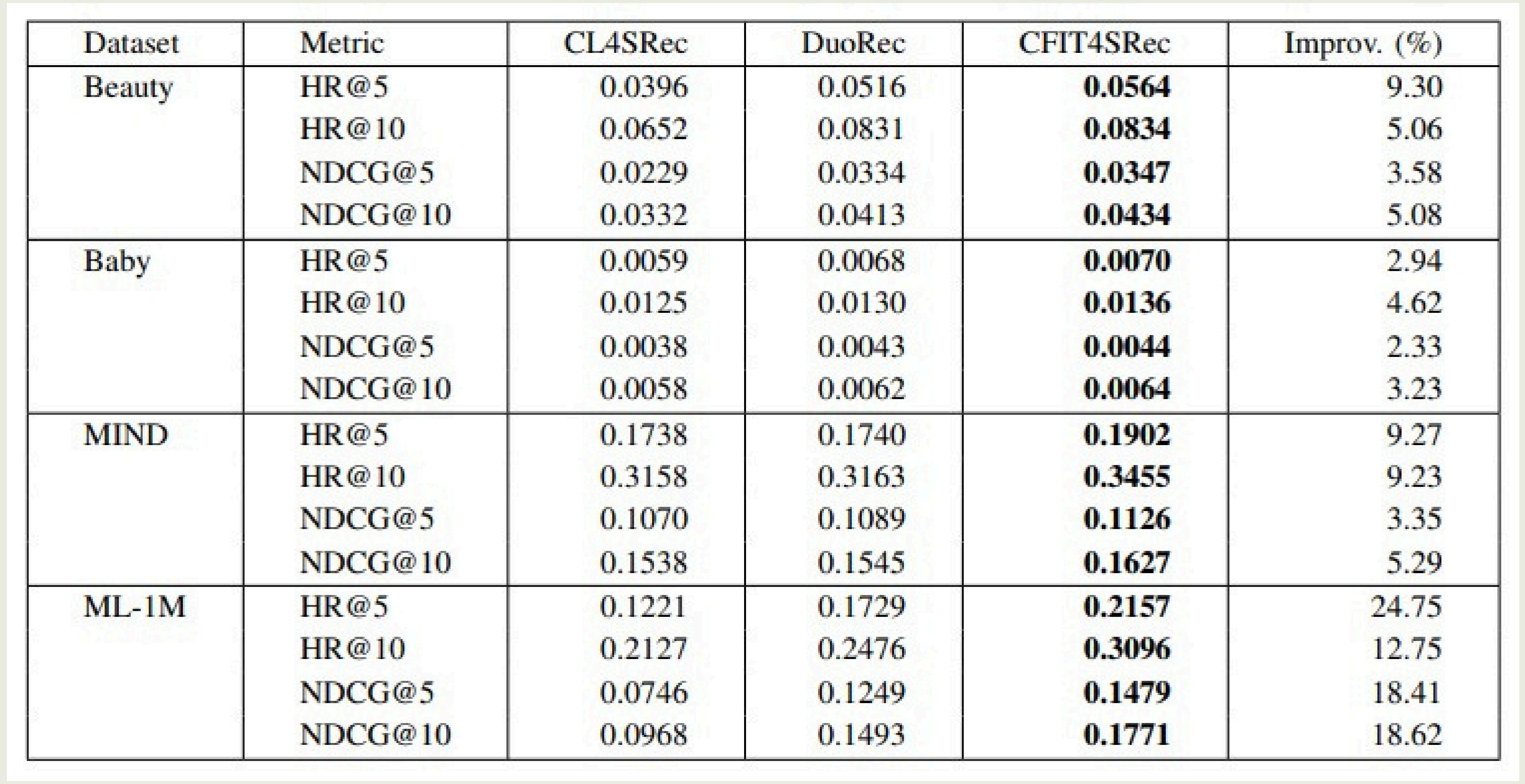
ML-1M
Dataset results:

ndcg@5: 0.1128
ndcg@10: 0.1498
hit@5: 0.1851
hit@10: 0.268

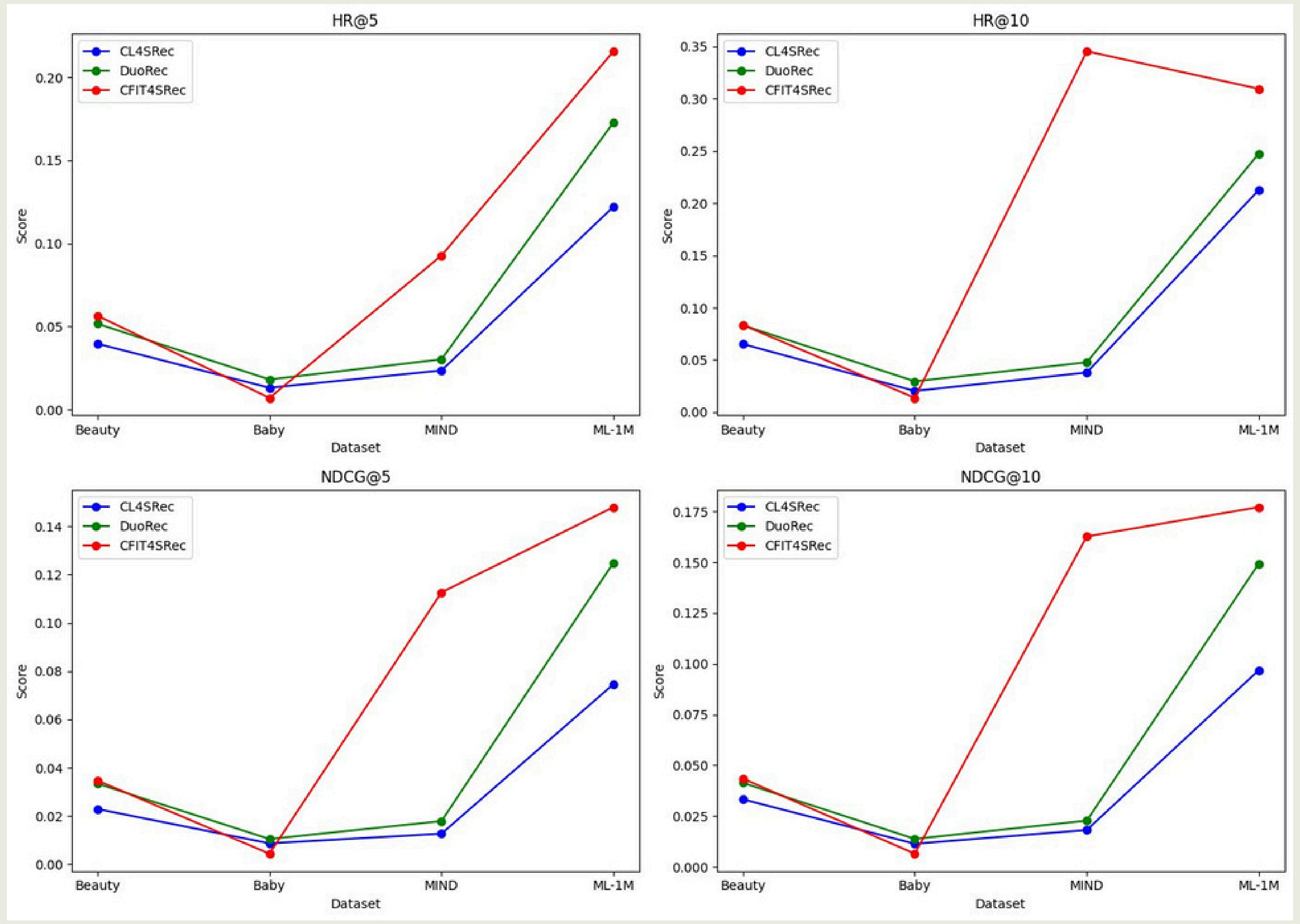
ML-100k
Dataset results:

ndcg@5: 0.038
ndcg@10: 0.0607
hit@5: 0.071
hit@10: 0.14

PERFORMANCE COMPARISON

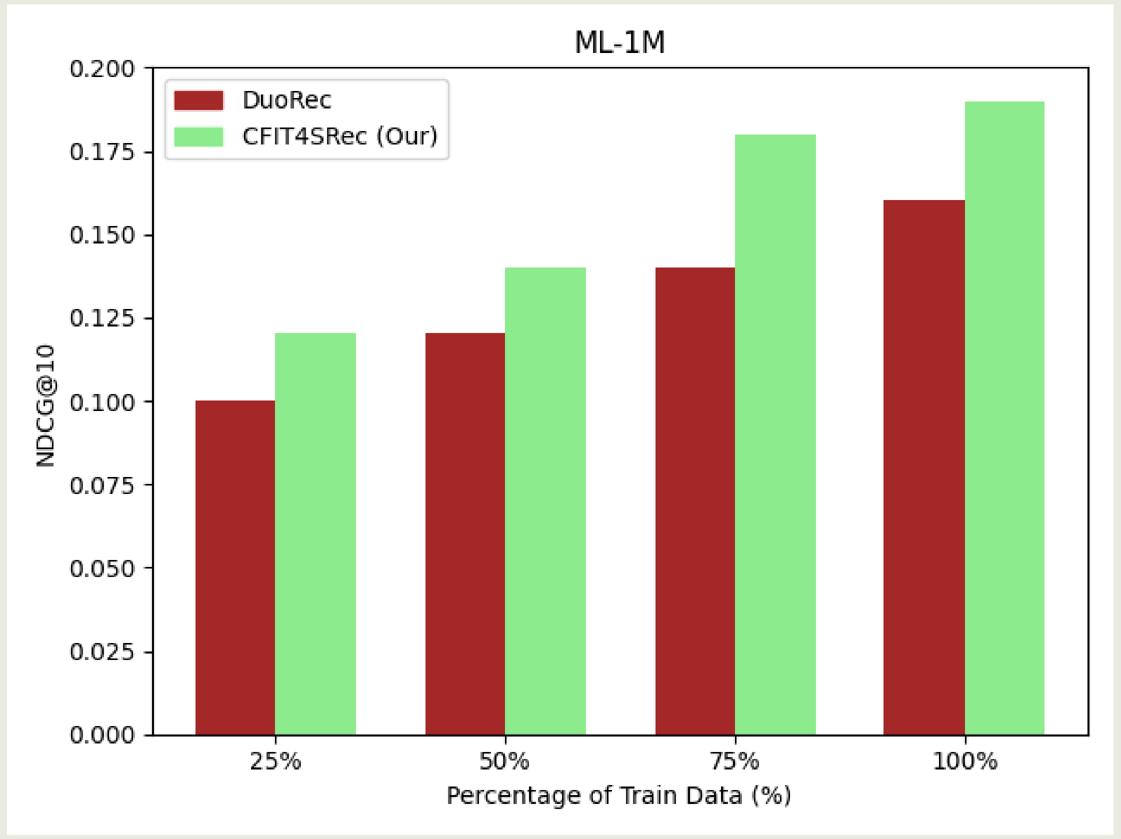


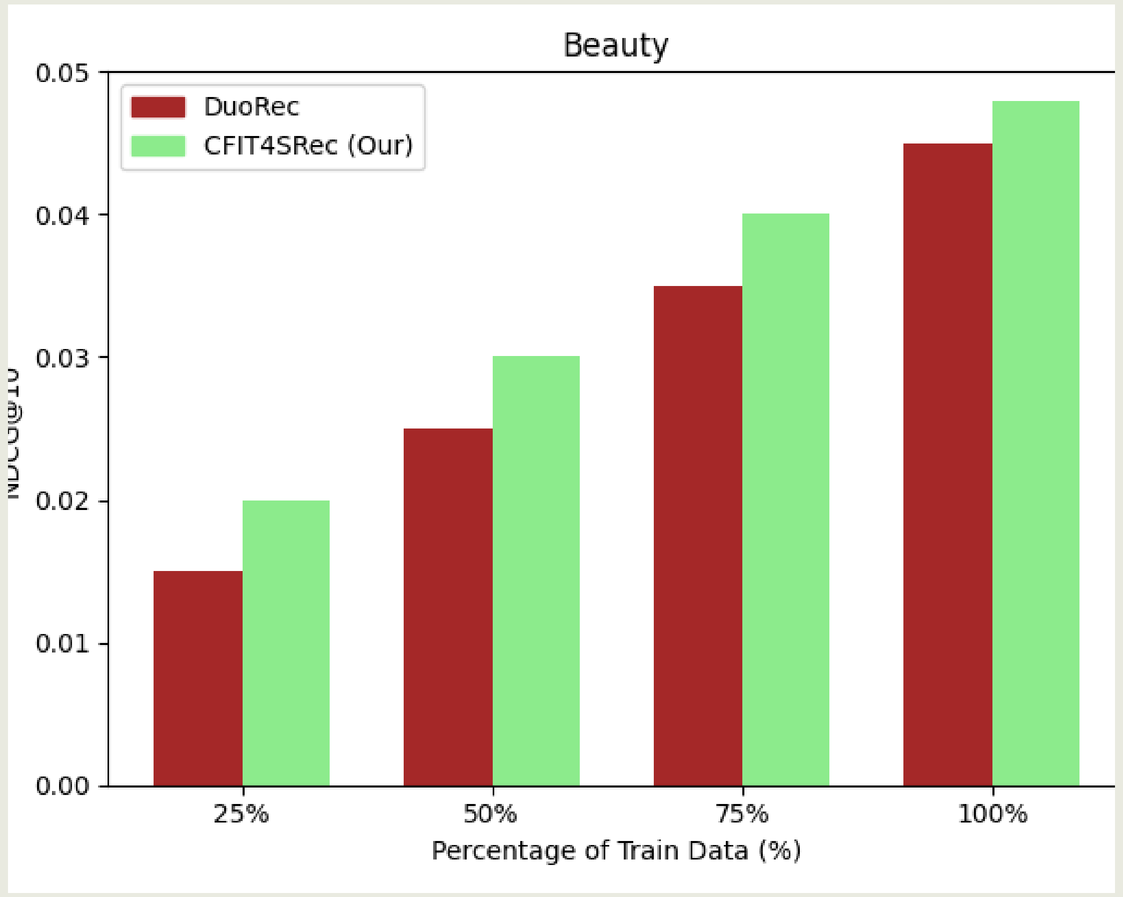
Dataset	Metric	CL4SRec	DuoRec	CFIT4SRec	Improv. (%)
Beauty	HR@5	0.0396	0.0516	**0.0564**	9.30
	HR@10	0.0652	0.0831	**0.0834**	5.06
	NDCG@5	0.0229	0.0334	**0.0347**	3.58
	NDCG@10	0.0332	0.0413	**0.0434**	5.08
Baby	HR@5	0.0059	0.0068	**0.0070**	2.94
	HR@10	0.0125	0.0130	**0.0136**	4.62
	NDCG@5	0.0038	0.0043	**0.0044**	2.33
	NDCG@10	0.0058	0.0062	**0.0064**	3.23
MIND	HR@5	0.1738	0.1740	**0.1902**	9.27
	HR@10	0.3158	0.3163	**0.3455**	9.23
	NDCG@5	0.1070	0.1089	**0.1126**	3.35
	NDCG@10	0.1538	0.1545	**0.1627**	5.29
ML-1M	HR@5	0.1221	0.1729	**0.2157**	24.75
	HR@10	0.2127	0.2476	**0.3096**	12.75
	NDCG@5	0.0746	0.1249	**0.1479**	18.41
	NDCG@10	0.0968	0.1493	**0.1771**	18.62

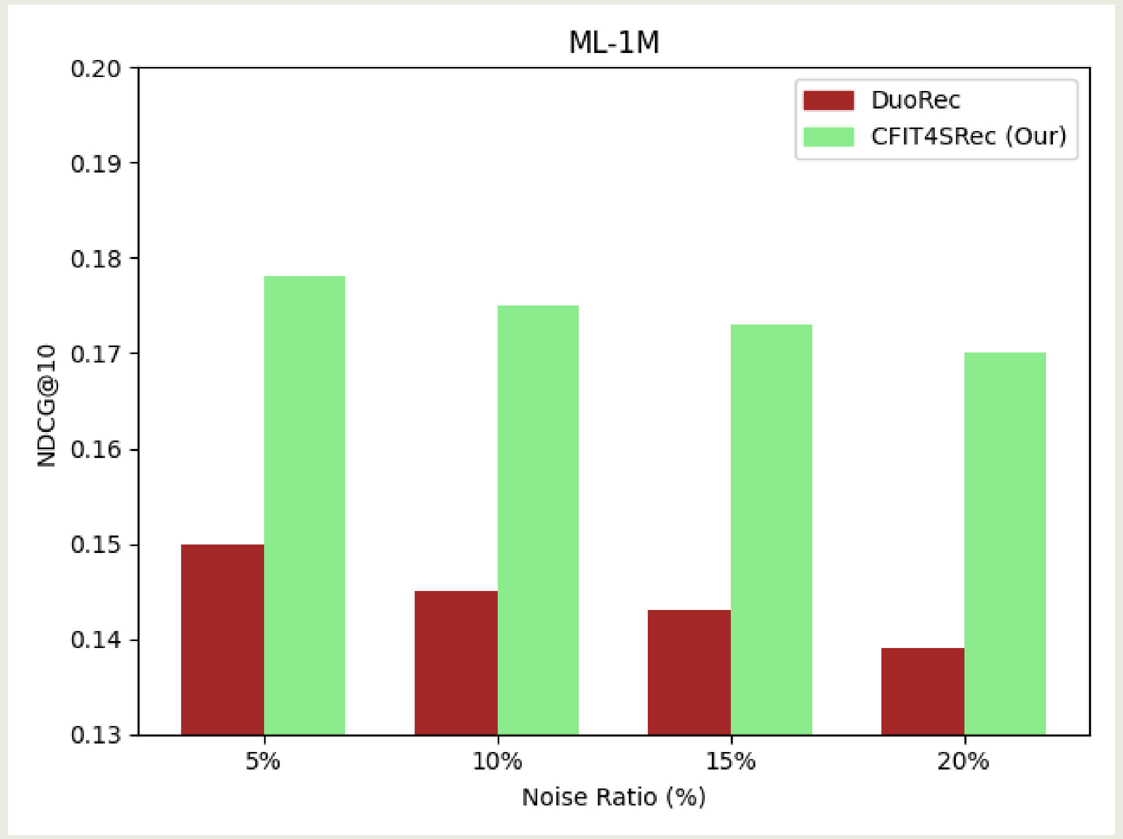


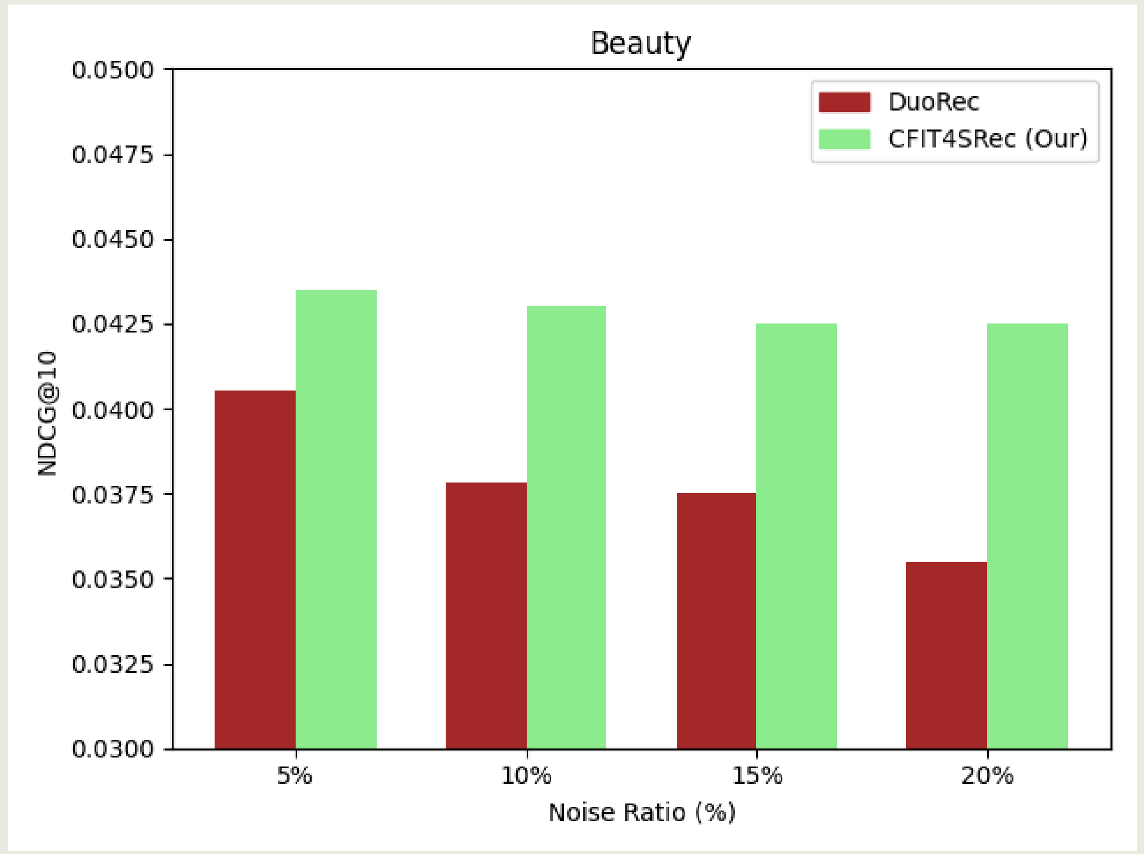
TRAINING RATIO ANALYSIS

NOISE RATIO ANALYSIS









Thank you.