Self-Supervised Learning in Recommender Systems

Fundamentals and Advances

Presenter: **Junliang Yu, Hongzhi Yin, Tong Chen**

Responsible Big Data Intelligence Lab

The University of Queensland

Tutorial website: https://ssl-recsys.github.io/

Presenters

Junliang Yu

- A third-year PhD student

School of Information Technology and Electrical Engineering, UQ

✉ jl.yu@uq.edu.au

Dr. Hongzhi Yin
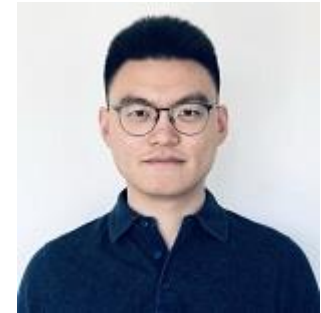
- Associate Professor

School of Information Technology and Electrical Engineering, UQ

✉ h.yin1@uq.edu.au

Dr. Tong Chen

- Lecturer

School of Information Technology and Electrical Engineering, UQ

✉ tong.chen@uq.edu.au

CONTENTS

- **Background and History**

- **Taxonomy of Self-Supervised Recommendation (SSR)**

- **Data Augmentation Techniques**

- **SSR Methods**
  - Contrastive, Generative, Predictive, and Hybrid

- **A Library for SSR**

- **Limitations and Future Research**
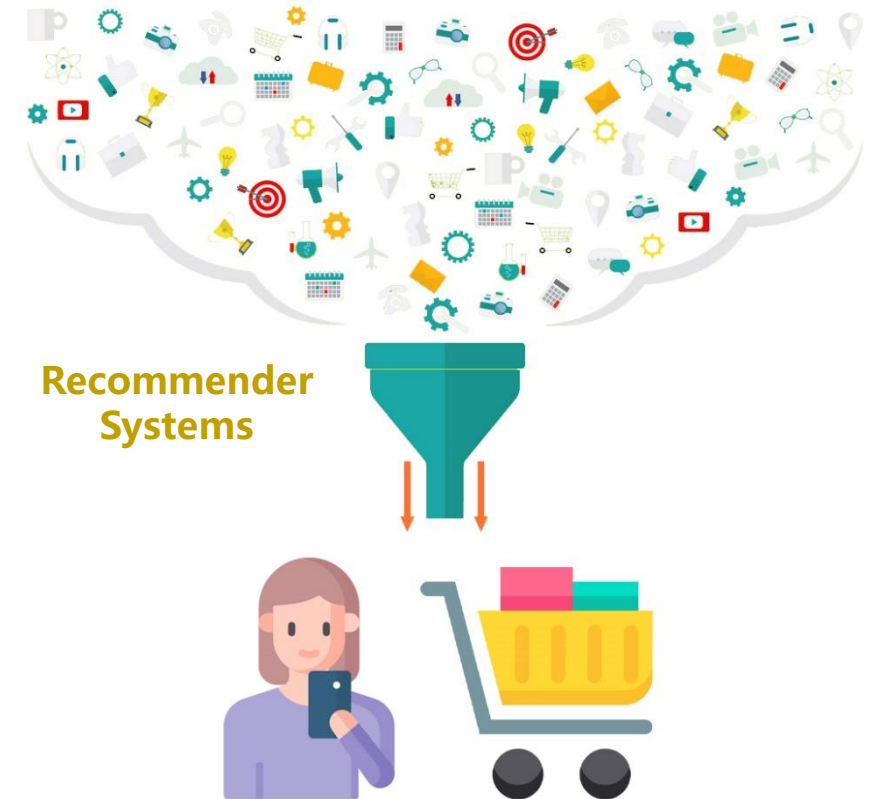
- **Q&A**

Background and History

Recommender Systems

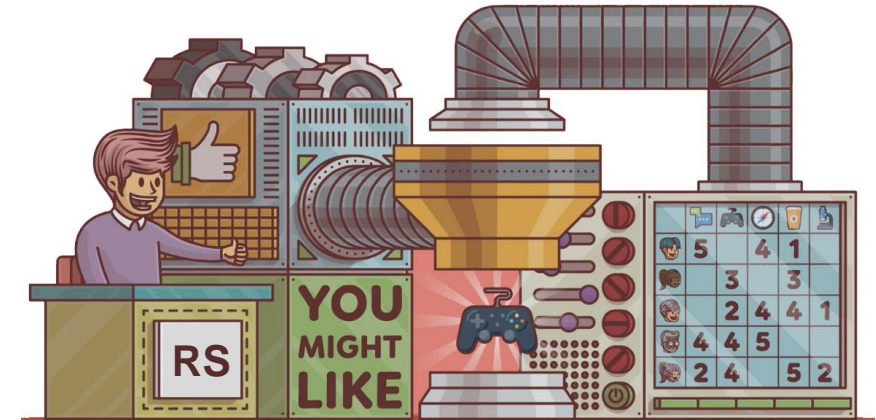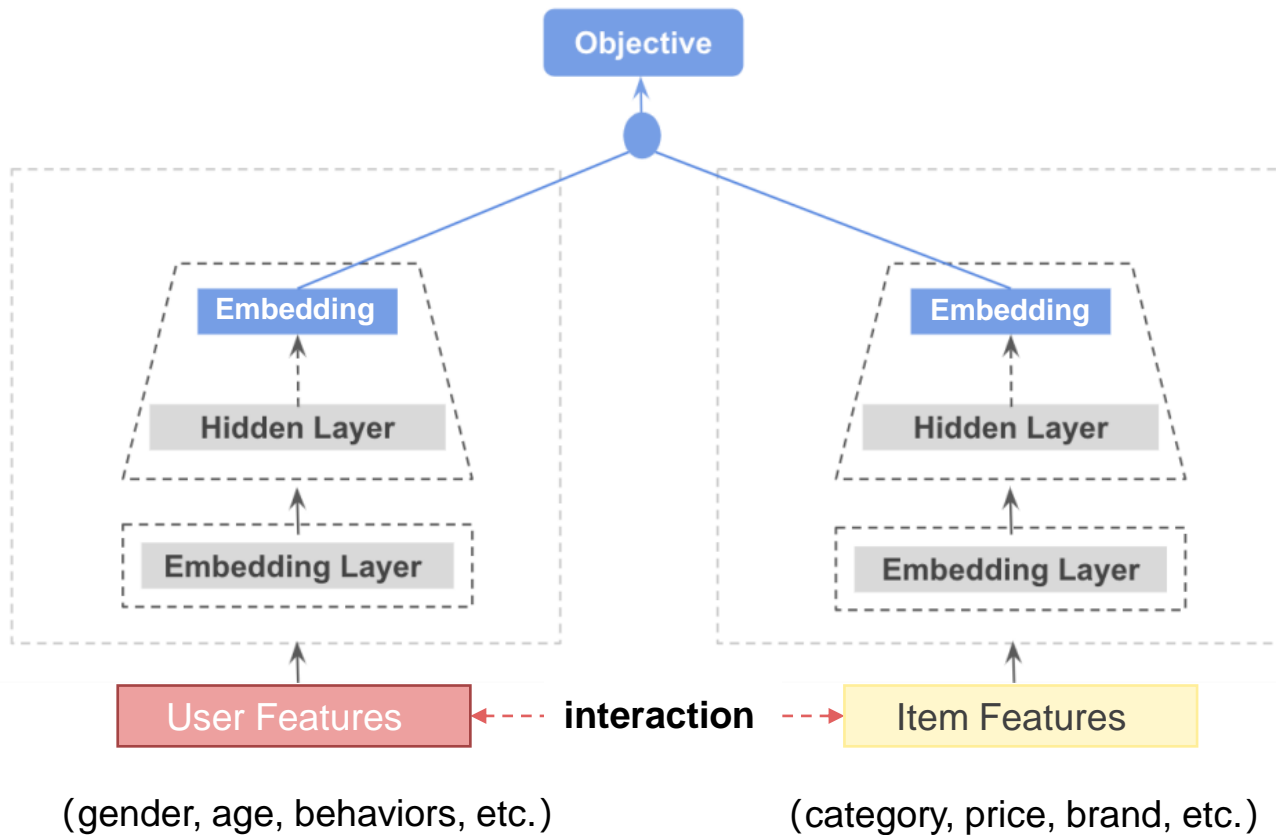We are now living in an age of information explosion.

Recommender Systems

Recommender systems can discover users' interests and ease the way of decision-making.

Recommender Systems

A general neural-architecture for recommendation.

Objective

Embedding | **Embedding**

Hidden Layer | Hidden Layer

Embedding Layer | Embedding Layer

User Features ←--- **interaction** ---→ Item Features

(gender, age, behaviors, etc.) | (category, price, brand, etc.)

Use the historical interaction data to compute item scores and recommend.

Problem of Data Sparsity

Most users generally only consume a tiny fraction of countless items.

How to generate satisfactory recommendations with very sparse interactions?

Self-Supervised Learning (SSL)

What is it?

Self-Supervised Learning (SSL) is a special type of representation learning that enables learning good data representation from unlabeled data.
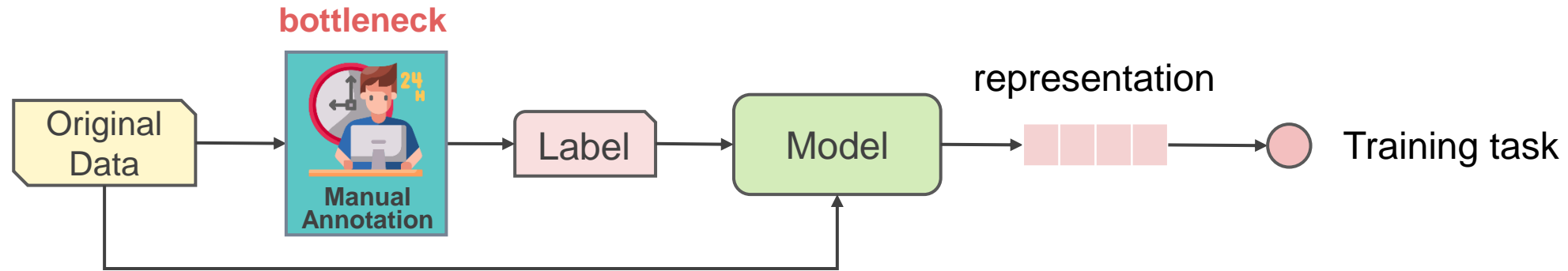
Why do we need it?

Data acquisition in recommender systems is costly since the data can only be generated by users themselves.

Learning good representation makes it easier to transfer useful information hiding in the unlabeled data to recommendation.

Self-Supervised Learning (SSL)

Supervised learning workflow

bottleneck

Original Data → Manual Annotation → Label → Model → representation → Training task

Self-Supervised learning workflow

Original Data → Automated (Data Augmentation) → Pseudo-Label → Model → representation → Transfer to **Downstream task**

Learned by **Self-Supervised task (pretext task)**
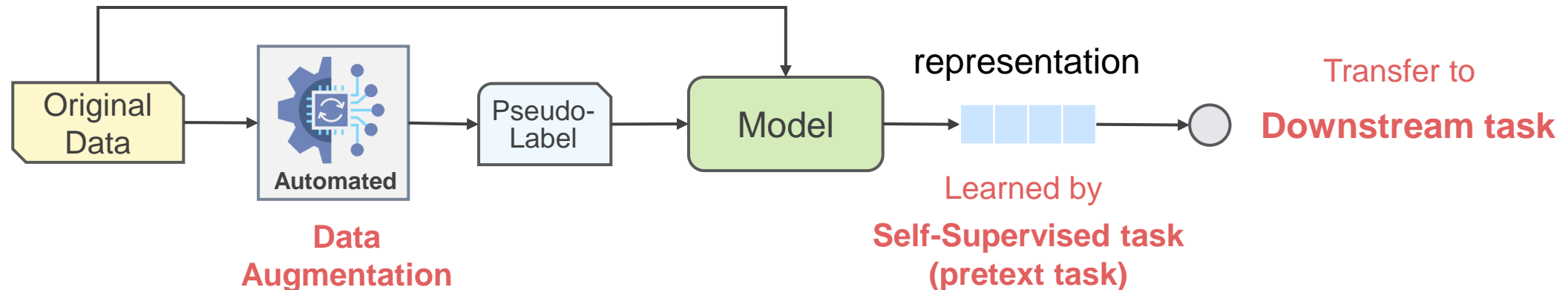
Self-Supervised Recommendation (SSR)

The principle of SSL is well-aligned with recommender systems' needs for more annotated data.
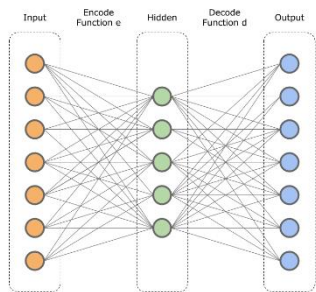
Self-supervised recommendation has following essential features[1]:
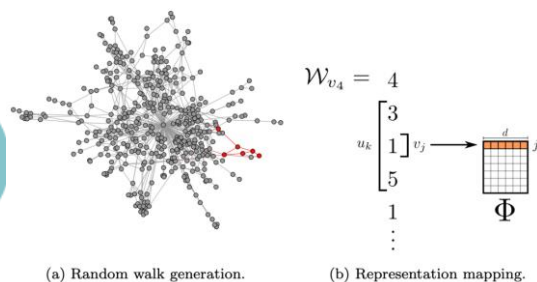
- Obtain more supervision signals by semi-automatically exploiting the raw data itself.
- Have a pretext task(s) to (pre-)train the recommendation model with the augmented data.
- Recommendation task is the unique primary task and the pretext task plays a secondary role to enhance recommendation.

[1]. Yu J, Yin H, Xia X, et al. Self-Supervised Learning for Recommender Systems: A Survey[J]. arXiv preprint arXiv:2203.15876, 2022.

Development of SSR

Autoencoder-Based

Network-Embedding-Based

BERT-Based

GAN-Based

2016　2017　2018　2019　2020

Self-Supervised Learning becomes an independent concept

Becoming Diverse

Early prototypes and implementations
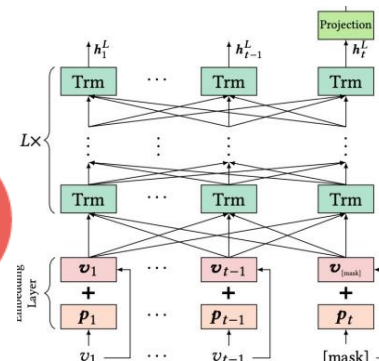
Article number (By 03/2022)

02

Taxonomy of Self-Supervised Recommendation

Architecture of SSR

Unified Architecture: **Encoder** + **Projection Head**

(GNNs, Transformer, etc.)   (nonlinear/linear mapping, identical mapping, etc.)

$$f_{\theta^*}, g_{\phi^*}, \mathbf{H}^* = \underset{f_\theta, g_\phi}{\arg\min} \; \mathcal{L}\left(g_\phi(f_\theta(\mathcal{D}, \tilde{\mathcal{D}}))\right)$$

augmented data

$$\tilde{\mathcal{D}} \sim \mathcal{T}(\mathcal{D})$$

representation   projection-head   encoder   original data   augmentation module

Multi-Task Learning:

$$\mathcal{L} \xrightarrow{\text{split}} \begin{array}{l} \mathcal{L}_{ssl} \\ \\ \mathcal{L}_{rec} \end{array}$$

$\mathcal{L}_{ssl}$ — self-supervised task (a.k.a. pretext task)

$\mathcal{L}_{rec}$ — recommendation task

merged loss
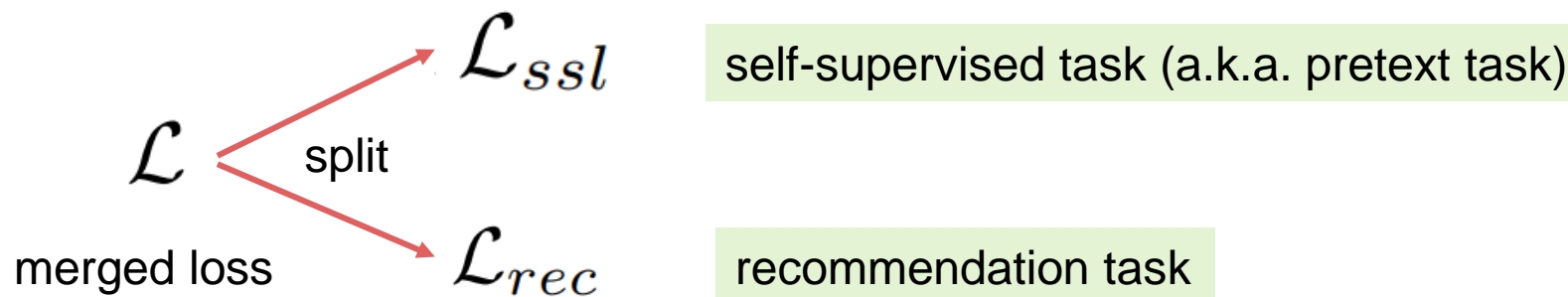
Taxonomy of SSR

According to the characteristics of the pretext tasks, the existing SSR methods can be divided into four categories: **contrastive, predictive, generative, and hybrid**.

Taxonomy of SSR

Contrastive Methods
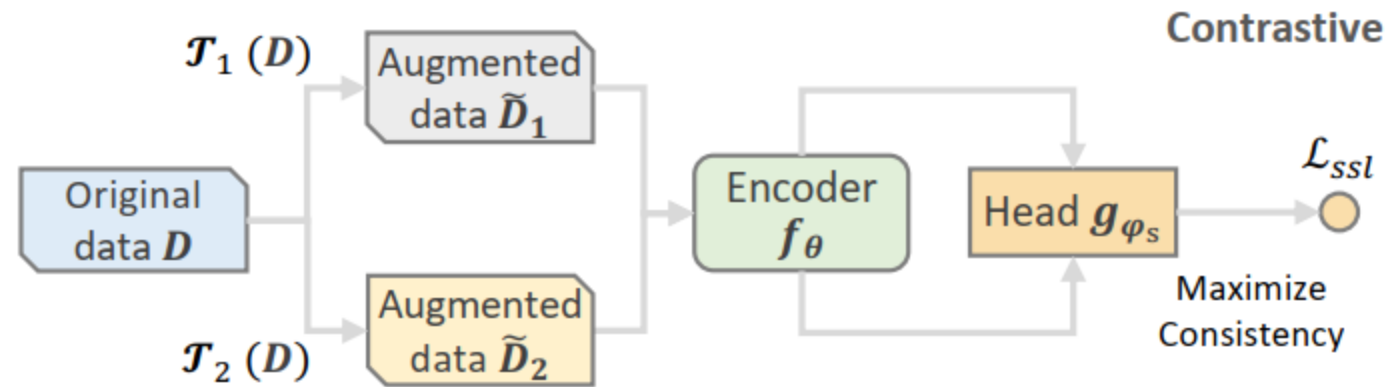
To treat every instance (e.g., user/item/sequence) as a class, and then pull views of the same instance closer in the embedding space, and push views of different instances apart.
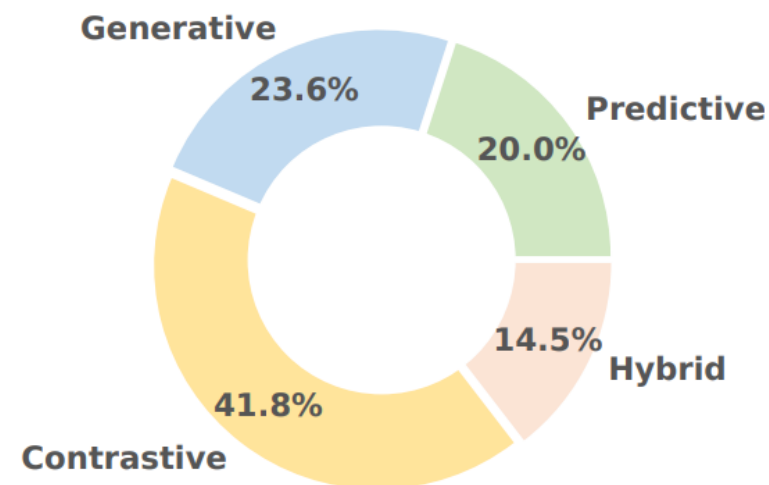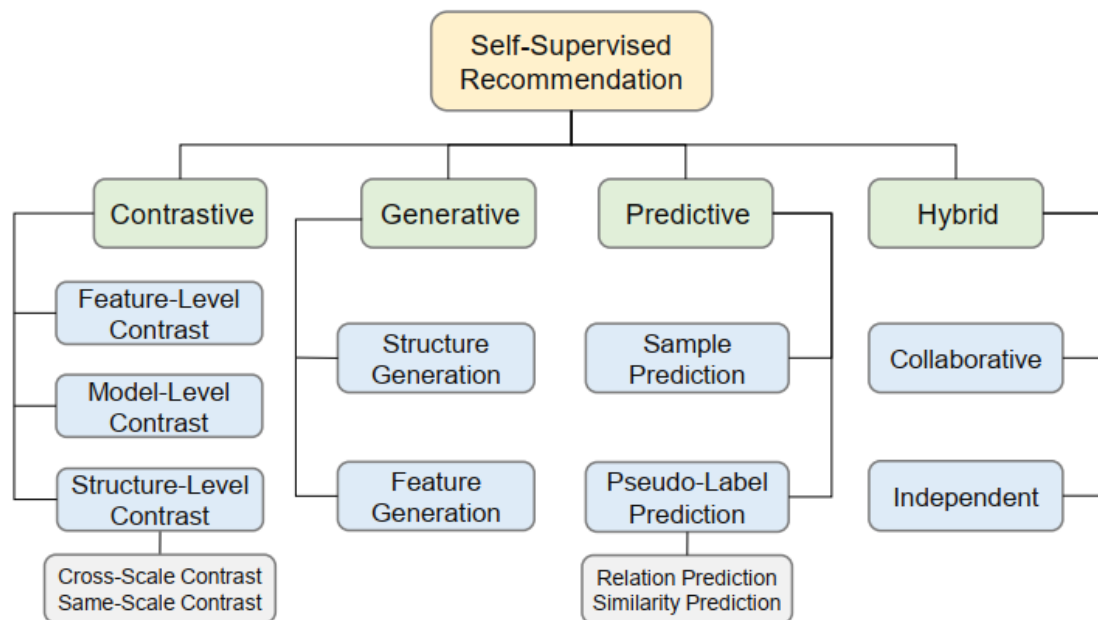
$$f_\theta^* = \arg\min_{f_\theta, g_{\phi_s}} \mathcal{L}_{ssl}\left(g_{\phi_s}\left(f_\theta(\tilde{\mathcal{D}}_1), f_\theta(\tilde{\mathcal{D}}_2)\right)\right)$$

Estimates the mutual information between views.

Views are created by imposing different transformations on the original data.

Taxonomy of SSR

Generative Methods
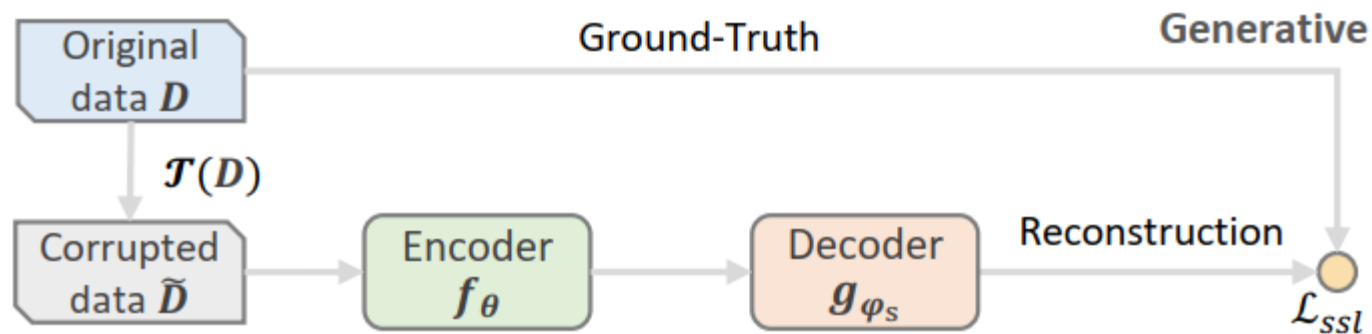
To reconstruct the original user/item profile with its corrupted versions.

$$f_\theta^* = \underset{f_\theta, g_{\phi_s}}{\arg\min} \ \mathcal{L}_{ssl}\Big(g_{\phi_s}\big(f_\theta(\tilde{D})\big), D\Big)$$

Learn to predict a portion of the available data from the rest.

Corrupted data

Example: BERT-based Models

Taxonomy of SSR

Predictive Methods
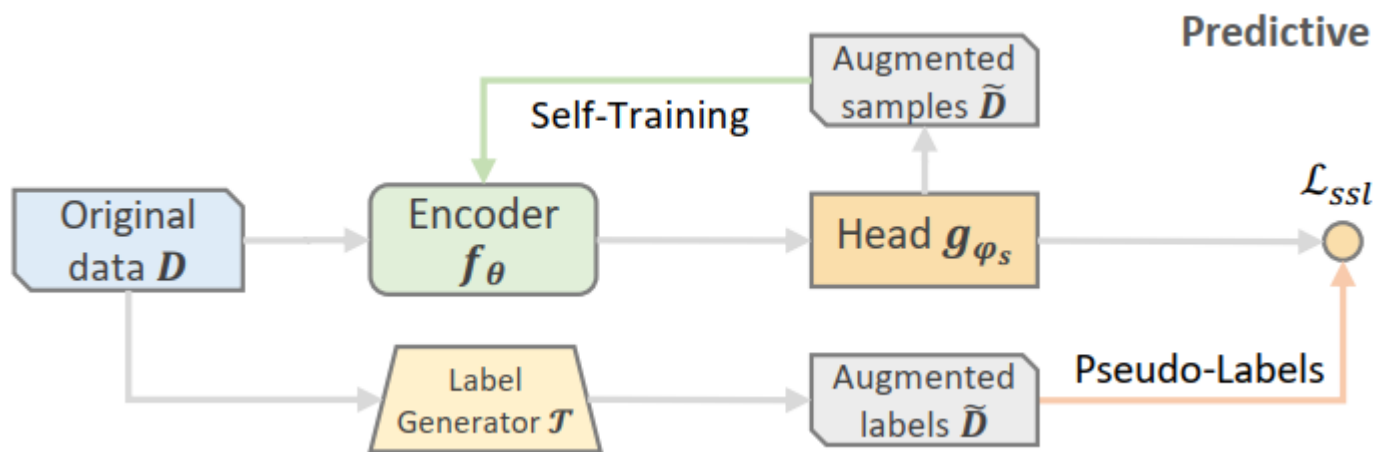
To generate new samples or labels from the original data in order to guide the pretext task.

$$f_\theta^* = \arg\min_{f_\theta, g_{\phi_s}} \mathcal{L}_{ssl}\left(g_{\phi_s}\left(f_\theta(\mathcal{D})\right), \tilde{\mathcal{D}}\right)$$

Predict the augmented pseudo-labels.

Augmented pseudo-labels

Taxonomy of SSR

Hybrid Methods

More than one encoder and projection head may be needed
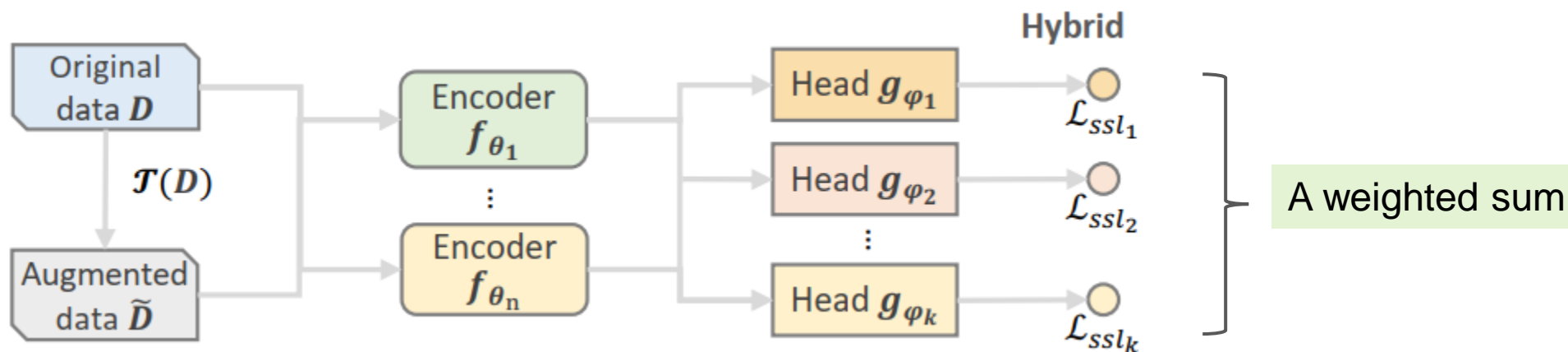
To obtain comprehensive self-supervision by combining different pretext tasks and integrating them into one recommendation model.

Typical Training Schemes

Joint Learning (. mostly used in contrastive methods.
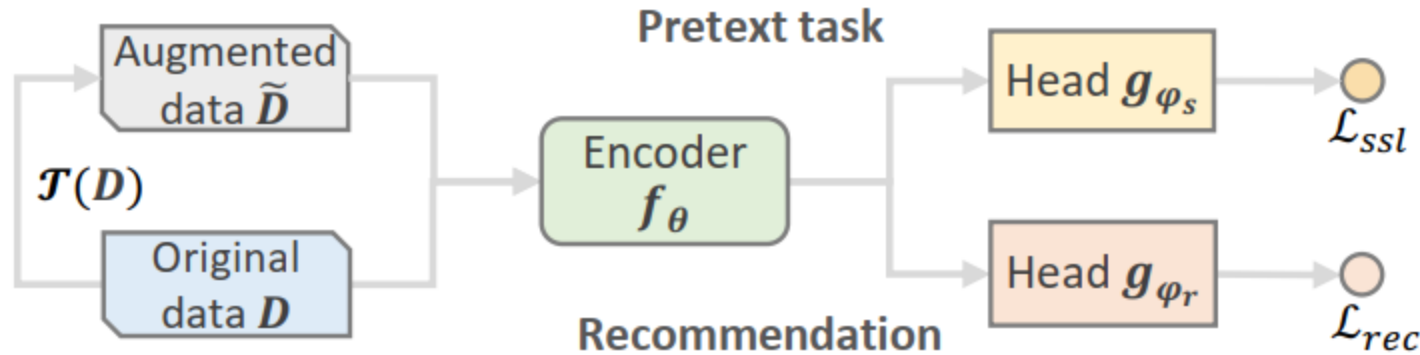
The pretext task and the recommendation task are jointly optimized with a shared encoder.

$$f_{\theta^*}, g_{\phi_r^*}, \mathbf{H}^* = \arg\min_{f_\theta, g_\phi}[\mathcal{L}_{rec}(g_{\phi_r}(f_\theta(\mathcal{D})))$$
$$+ \alpha\mathcal{L}_{ssl}(g_{\phi_s}(f_\theta(\tilde{\mathcal{D}})))]$$

Recommendation: primary task

control the magnitude of self-supervision

SSL: auxiliary task, not concerned

Typical Training Schemes

Pre-training and Fine-tuning (PF) used in most of generative methods and a few contrastive methods.

Step 1: **pre-train**

Augmented data $\tilde{D}$ → Encoder f_θ → Pretext task: Head g_{φ_s} → $\mathcal{L}_{ssl}$

Parameter transfer θ_{init}

$\mathcal{T}(D)$

Step 2: **Fine-tune**

Original data D → Encoder f_θ → Recommendation: Head g_{φ_r} → $\mathcal{L}_{rec}$
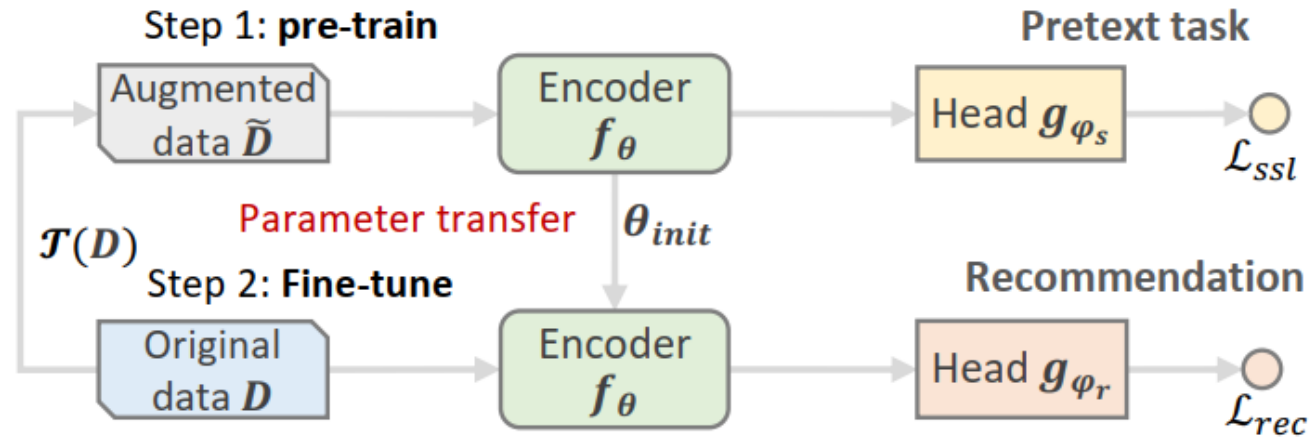
The encoder is firstly pretrained with pretext tasks on the augmented data for a good initialization of the encoder's parameters. It is then is fine-tuned on the original data for recommendation

$$f_{\theta_{init}} = \underset{f_\theta, g_{\phi_s}}{\arg\min} \boxed{\mathcal{L}_{ssl}}(g_{\phi_s}(f_\theta(\tilde{\mathcal{D}}), \mathcal{D}))$$

$$f_{\theta^*}, g_{\phi_r^*}, \mathbf{H}^* = \underset{f_{\theta_{init}}, g_{\phi_r}}{\arg\min} \boxed{\mathcal{L}_{rec}}(g_{\phi_r}(f_\theta(\mathcal{D})))$$

Pre-train the encoder

Fine-tune the encoder for recommendation

Typical Training Schemes

Integrated Learning (IL)

mainly used by the pseudo-labels-based predictive methods and a few contrastive methods.

Integrated Objective
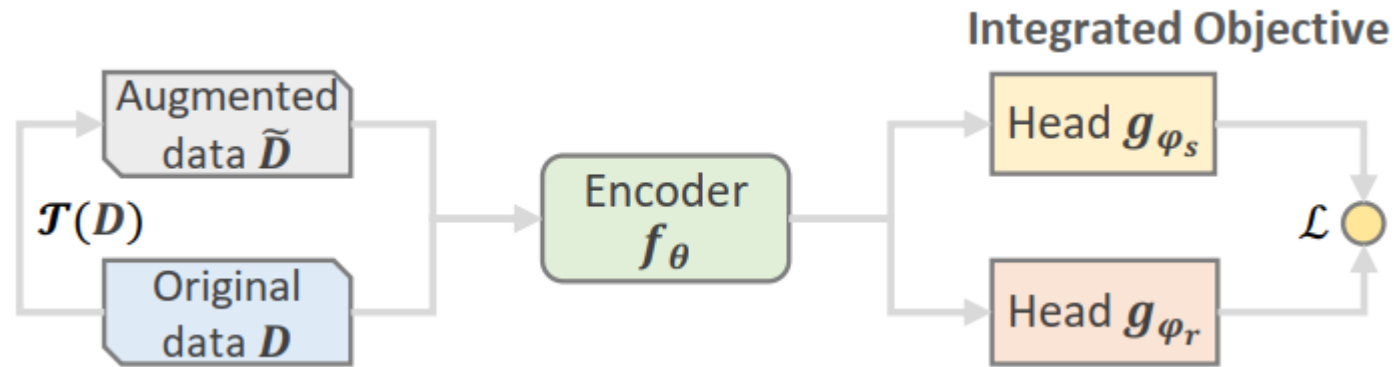
The pretext task and the recommendation task are well-aligned under this setting and they are unified into an integrated objective.

$$f_{\theta^*}, g_{\phi_r^*}, \mathbf{H}^* = \underset{f_\theta, g_\phi}{\arg\min}\; \mathcal{L}\left(g_{\phi_r}(f_\theta(\mathcal{D})), g_{\phi_s}(f_\theta(\tilde{\mathcal{D}}))\right)$$

Integrated loss

Pre-train & Fine-tune 41.8%

10.9% Integrated Learning

47.3% Joint Learning
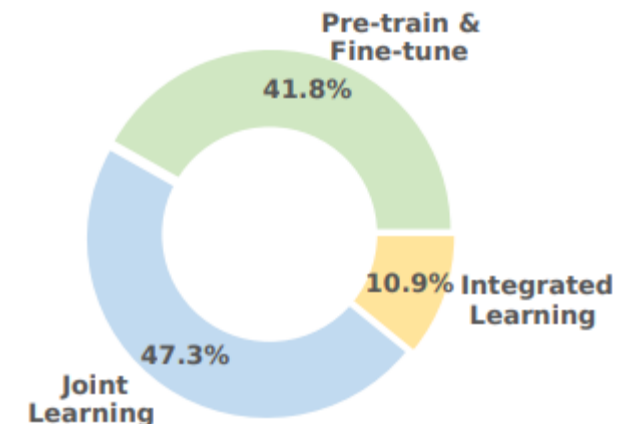
03

Data Augmentation Techniques
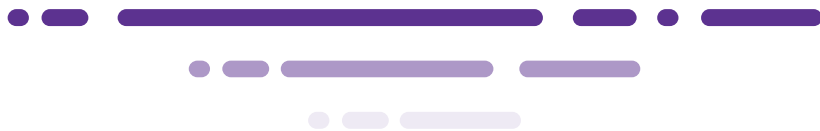
Data Augmentation

- Data augmentation increases the amount of training data by creating plausible variations of existing data without additional ground-truth labels.

- It plays a pivotal role in learning quality and generalizable representations.

- In SSR, the most commonly used data augmentation approaches can be divided into three categories: **graph-based**, **sequence-based**, and **feature-based**.

Original Data → Augment → New Data
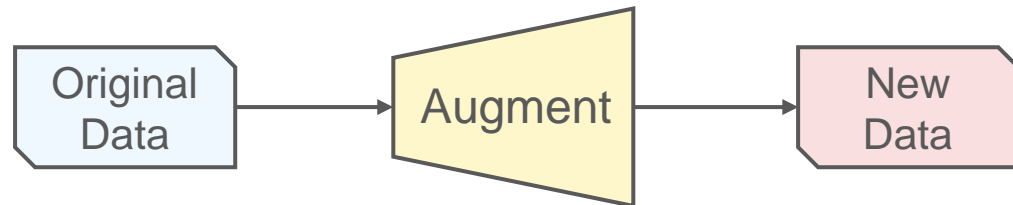
Graph-Based Augmentation
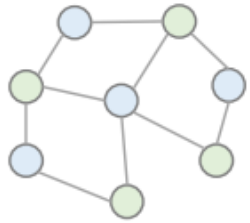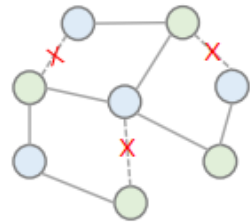
Original Graph   **Edge Dropout**   **Node Dropout**
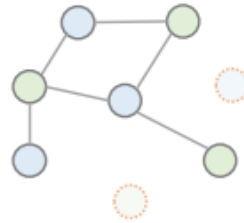
- **Edge/Node Dropout**: With the probability ρ, each edge/node may be removed from the graph.

Idea behind   Only partial connections/nodes contribute to the node representations.

Formulation

$$\tilde{\mathcal{G}}, \tilde{\mathbf{A}} = \mathcal{T}_{\text{E-dropout}}(\mathcal{G}) = (\mathcal{V}, \mathbf{m} \odot \mathcal{E}) \qquad \mathbf{m} \in (0,1)^{|\mathcal{E}|}$$

$$\tilde{\mathcal{G}}, \tilde{\mathbf{A}} = \mathcal{T}_{\text{N-dropout}}(\mathcal{G}) = (\mathcal{V} \odot \mathbf{m}, \mathcal{E} \odot \mathbf{m}') \qquad \mathbf{m} \in (0,1)^{|\mathcal{V}|}$$

Graph-Based Augmentation
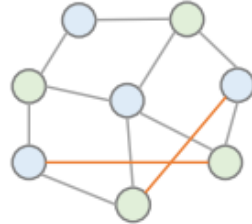
Original Graph   Diffusion

- **Graph Diffusion**: Opposite to the dropout-based method, the diffusion-based augmentation adds edges into the graph to create views.

Idea behind   The missing user behaviors include unknown positive preferences which can be represented with weighted user-item edges.

Formulation

$$\tilde{\mathcal{G}}, \tilde{\mathbf{A}} = \mathcal{T}_{\text{diffusion}}(\mathcal{G}) = (\mathcal{V}, \mathcal{E} + \tilde{\mathcal{E}})$$

Discover the possible edges by calculating the similarities of the user and item representations and retain the edges with top-K similarities.
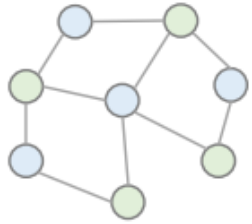
Sequence-Based Augmentation

Original Sequence   Mask

- **Item Masking**: randomly masks a portion of items and replaces them with special tokens [mask].

Idea behind   A user's intention is relatively stable during a period of time. Though part of items are masked, the primary intent is still retained in the rest.

Formulation

$$\tilde{S} = \mathcal{T}_{\text{masking}}(S) = [\tilde{i}_1, \tilde{i}_2, ..., \tilde{i}_k], \tilde{i}_t = \begin{cases} i_t, & t \notin \mathcal{M} \\ [\text{mask}], & t \in \mathcal{M} \end{cases}$$

$\mathcal{M}$ denotes the indices of the masked items.

Sequence-Based Augmentation

Original Sequence   Crop

- **Item Cropping**: a sub-sequence with a certain length is randomly cropped.

Idea behind   The selected sub-sequence is expected to endow the model with the ability to learn generalized representations without the comprehensive user profile.

Formulation

$$\tilde{S} = \mathcal{T}_{\text{cropping}}(S) = [\tilde{i}_c, \tilde{i}_{c+1}, \ldots, \tilde{i}_{c+L_c-1}]$$

where $L_c = \lfloor \eta * |S| \rfloor$ and $\eta \in (0,1)$

Sequence-Based Augmentation

Original Sequence   Reorder

- **Item Cropping**: shuffle a continuous sub-sequence to create sequence augmentations.

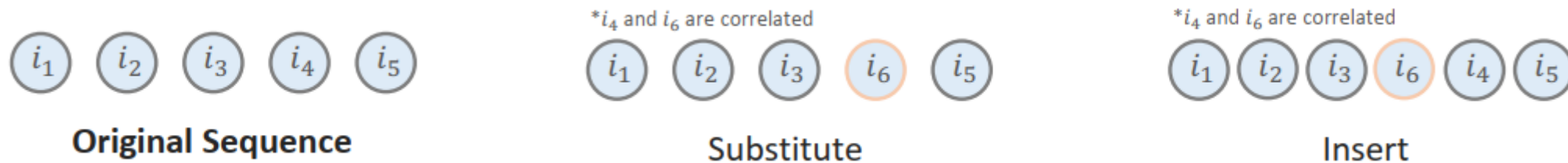| Idea behind	The item order in a sequence is not strict, and different item orders may actually correspond to the same user intent

Formulation

$$\tilde{S} = \mathcal{T}_{\text{reordering}}(S) = [i_1, ..., \tilde{i}_r, \tilde{i}_{r+1}, ..., \tilde{i}_{r+L_r-1}, ..., i_k]$$

Sequence-Based Augmentation

*i_4 and i_6 are correlated

*i_4 and i_6 are correlated

Original Sequence

Substitute

Insert

- **Item Substitution/Insertion**: substitute/insert items in(to) short sequences with highly correlated items

Idea behind

Random item cropping and masking could exaggerate the data sparsity issue in short sequences. Substituting/Inserting highly-correlated items injects less corruptions

Formulation

$$\tilde{S} = \mathcal{T}_{\text{substitution}}(S) = [\tilde{i}_1, \tilde{i}_2, ..., \tilde{i}_k], \quad \tilde{i}_t = \begin{cases} i_t, & t \notin \mathcal{Z} \\ \text{item correlated to } i_t, & t \in \mathcal{Z} \end{cases}$$

$$\tilde{S} = \mathcal{T}_{\text{insert}}(S) = [i_1, ..., \tilde{i}_{id_1}, i_{id_1}, ..., \tilde{i}_{id_l}, i_{id_l} ..., i_k]$$

Feature-Based Augmentation

- **Feature Dropout**: similar to the item masking and edge dropout, which randomly masks/drops a small portion of features.

Idea behind   The whole profile/information can be inferred by a portion of features.

Formulation $$\tilde{\mathbf{X}} = \mathcal{T}_{\text{F-dropout}}(\mathbf{X}) = \mathbf{X} \odot \mathbf{M}$$ The matrix $\mathbf{M}$ is generated by Bernoulli distribution.

- **Feature Shuffling**: switches rows and columns in the feature matrix.

Idea behind   By randomly changing the contextual information, the feature matrix is corrupted to yield augmentations

Formulation $$\tilde{\mathbf{X}} = \mathcal{T}_{\text{shuffling}}(\mathbf{X}) = \mathbf{P}_r \mathbf{X} \mathbf{P}_c$$ $\mathbf{P}_r$ and $\mathbf{P}_r$ are permutation matrices

Feature-Based Augmentation

- **Feature Mixing**: mix the original features with features from other users/items or previous versions to synthesize informative negative/positive examples .

Idea behind	Mixing related features can generate hard examples.

Formulation	$\tilde{\mathbf{x}}_i = \mathcal{T}_{\text{mixing}}(\mathbf{x}_i) = \alpha \mathbf{x}_i + (1 - \alpha)\mathbf{x}'_j, \quad \alpha \in [0, 1]$

- **Feature Clustering**: the augmented prototype representations can be learned via clustering algorithms.

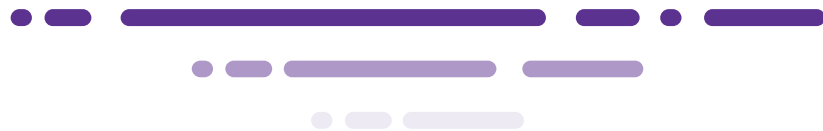 | Idea behind | Assume that there are prototypes in the feature/representation space and user/item representations should be closer to their assigned prototypes. |
  |---|---|

Formulation	$\tilde{\mathbf{C}} = \mathcal{T}_{\text{clustering}}(\mathbf{X}) = \text{EM}(\mathbf{X}, \mathcal{C}),$ where $\mathcal{C}$ is the presupposed clusters (prototypes) and $\tilde{\mathbf{C}}$ is the augmented prototype representations.

04

SSR Methods
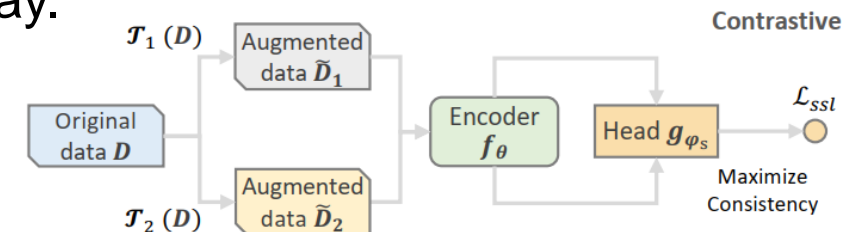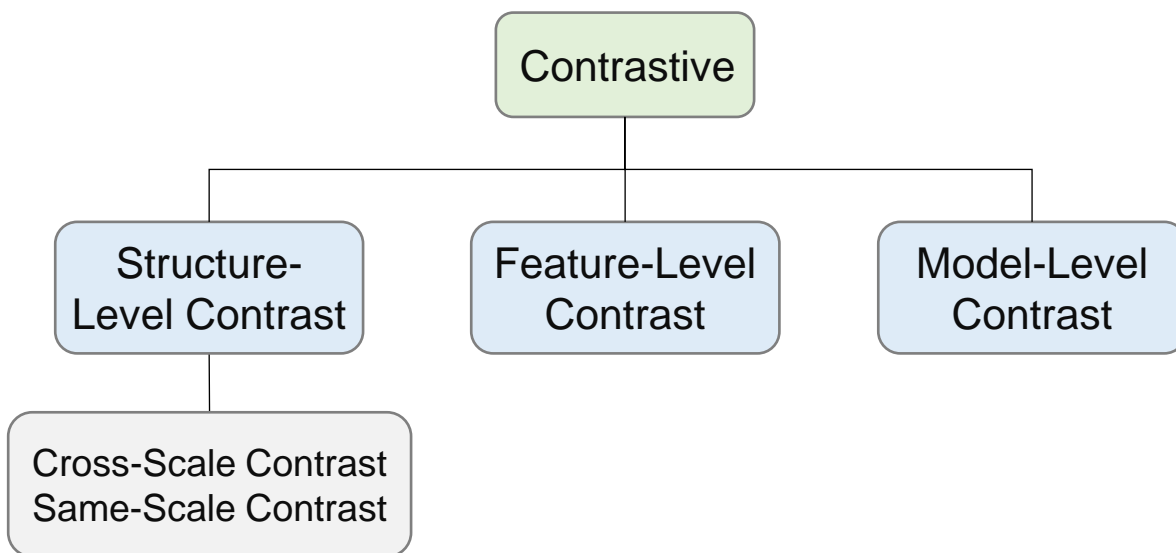
- Contrastive, Generative, Predictive, and Hybrid

Tutorial

Contrastive SSR Methods

Contrastive Methods

Pull similar samples closer and Push dissimilar samples away.

Contrastive

$\mathcal{T}_1(D)$ → Augmented data $\tilde{D}_1$

Original data D

$\mathcal{T}_2(D)$ → Augmented data $\tilde{D}_2$

Encoder f_θ → Head g_{φ_s} → $\mathcal{L}_{ssl}$

Maximize Consistency

$$f_\theta^* = \arg\min_{f_\theta, g_{\phi_s}} \mathcal{L}_{ssl}\left(g_{\phi_s}\left(f_\theta(\tilde{\mathcal{D}}_1), f_\theta(\tilde{\mathcal{D}}_2)\right)\right)$$

Contrastive
- Structure-Level Contrast
  - Cross-Scale Contrast
    Same-Scale Contrast
- Feature-Level Contrast
- Model-Level Contrast

According to where the self-supervision signals come from, we divide them into three categories: **structure-level contrast**, **feature-level contrast**, and **model-level contrast**.

Structure-Level Contrast

Local-Local

Same-Scale

Global-Global

contrast

contrast

$$f_\theta^* = \underset{f_\theta, g_{\phi_s}}{\arg\min} \mathcal{L}_{\mathcal{MI}}\big(g_{\phi_s}(\tilde{\mathbf{h}}_i, \tilde{\mathbf{h}}_j)\big)$$

$\tilde{\mathbf{h}}_i$ and $\tilde{\mathbf{h}}_j$ are node representations

$$f_\theta^* = \underset{f_\theta, g_{\phi_s}}{\arg\min} \mathcal{L}_{\mathcal{MI}}\big(g_{\phi_s}(\mathrm{Agg}(f_\theta(\tilde{S}_i)), \mathrm{Agg}(f_\theta(\tilde{S}_j)))\big)$$

$\tilde{S}_i$ and $\tilde{S}_j$ are two sequence augmentations

Local-Global

Cross-Scale

Local-Context

contrast

contrast

$$f_\theta^* = \underset{f_\theta, g_{\phi_s}}{\arg\min} \mathcal{L}_{\mathcal{MI}}\big(g_{\phi_s}(\tilde{\mathbf{h}}, \mathcal{R}(f_\theta(\tilde{\mathcal{G}}, \tilde{\mathbf{A}})))\big)$$

$\mathcal{R}$ is the readout function that generates global-level graph representation.

$$f_\theta^* = \underset{f_\theta, g_{\phi_s}}{\arg\min} \mathcal{L}_{\mathcal{MI}}\big(g_{\phi_s}(\mathbf{h}_i, \mathcal{R}(f_\theta(\mathcal{C}_j)))\big)$$

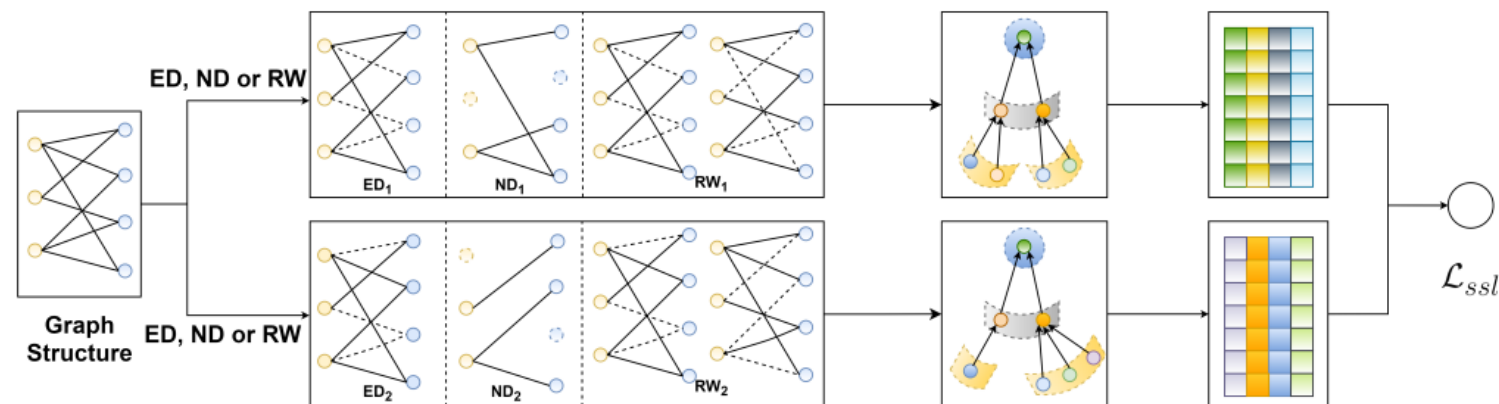$\mathcal{C}_j$ denotes the context of node (sequence) j.

Structure-Level Contrast

Local-Local Contrast: SGL (Wu et al. 2021)

BPR Loss

$$\mathcal{L}_{main} = \sum_{(u,i,j)\in O} -\log \sigma(\hat{y}_{ui} - \hat{y}_{uj})$$
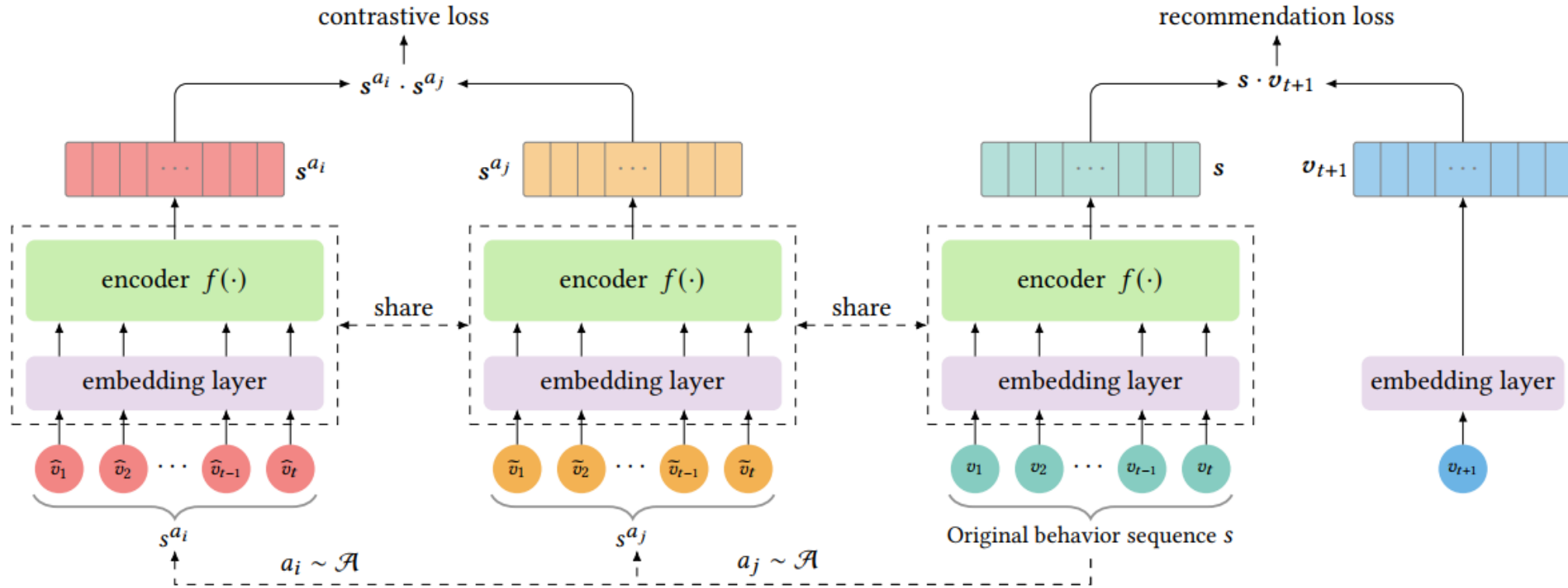
InfoNCE Loss

$$\mathcal{L}_{ssl}^{user} = \sum_{u\in\mathcal{U}} -\log \frac{\exp(s(\mathbf{z}_u', \mathbf{z}_u'')/\tau)}{\sum_{v\in\mathcal{U}} \exp(s(\mathbf{z}_u', \mathbf{z}_v'')/\tau)}$$

$$\mathcal{L} = \mathcal{L}_{main} + \lambda_1 \mathcal{L}_{ssl} + \lambda_2 \|\Theta\|_2^2$$

J. Wu, X Wang et al. Self-supervised Graph Learning for Recommendation. SIGIR'21

Structure-Level Contrast

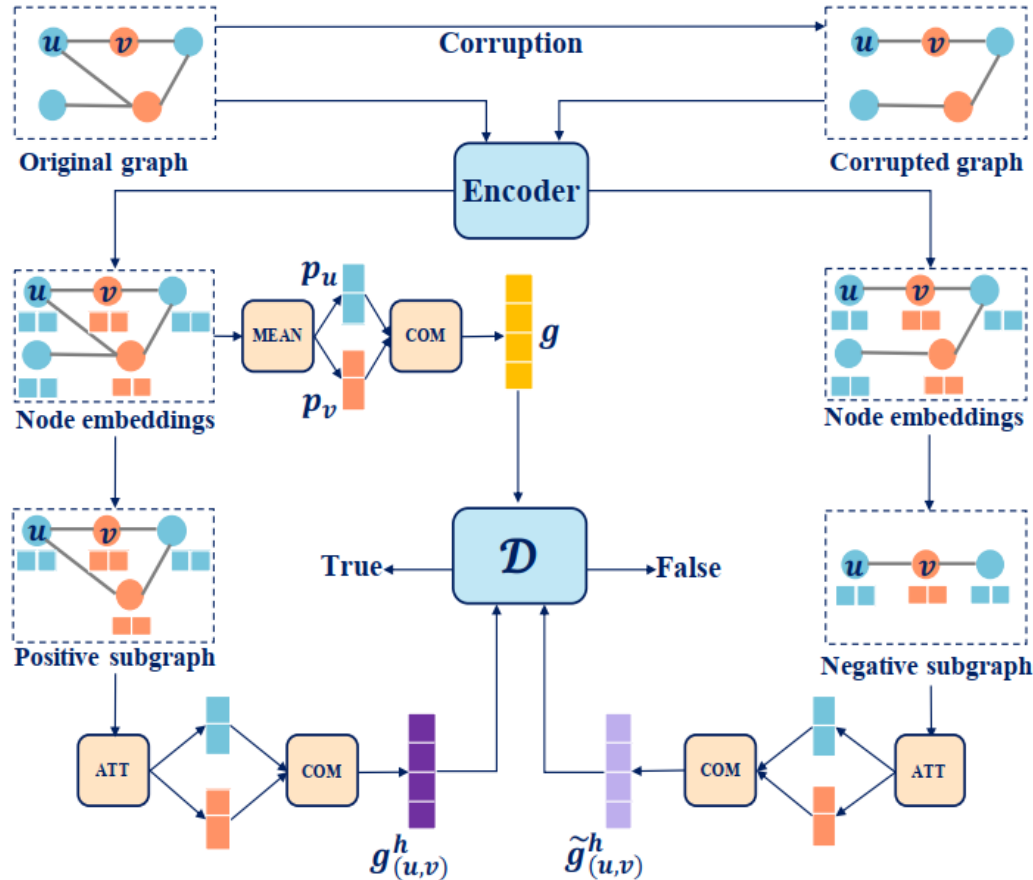**Global-Global Contrast: CL4SRec (Xie et al. 2021)**

$$\mathcal{L}_{\text{cl}}(s_u^{a_i}, s_u^{a_j}) = -\log \frac{\exp\left(\text{sim}(s_u^{a_i}, s_u^{a_j})\right)}{\exp\left(\text{sim}(s_u^{a_i}, s_u^{a_j})\right) + \sum\limits_{s^- \in S^-} \exp\left(\text{sim}(s_u^{a_i}, s^-)\right)}$$

$$\mathcal{L}_{\text{main}}(s_{u,t}) = -\log \frac{\exp(s_{u,t}^\top v_{t+1}^+)}{\exp(s_{u,t}^\top v_{t+1}^+) + \sum\limits_{v_{t+1}^- \in \mathcal{V}^-} \exp(s_{u,t}^\top v_{t+1}^-)}$$

X, Xie et al. Contrastive Learning for Sequential Recommendation. arXiv'21

Structure-Level Contrast

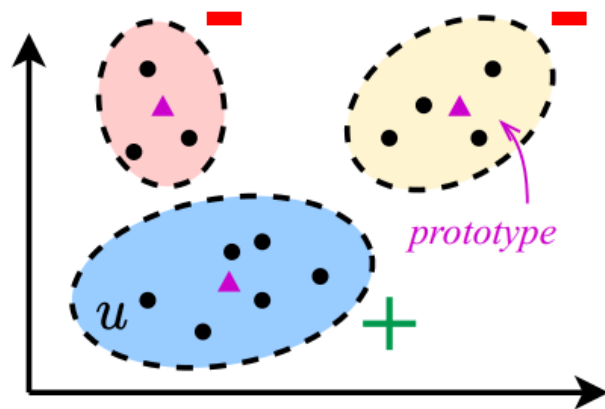**Local-Global Contrast: BiGI (Cao et al. 2021)**

CL Loss

$$\mathcal{L}_m = -\frac{1}{|E| + |\widetilde{E}|}\Big(\sum_{i=1}^{|E|}\mathbb{E}_G\big[\log\mathcal{D}(g^h_{(u,v)_i}, g)\big]+$$

$$\sum_{i=1}^{|\widetilde{E}|}\mathbb{E}_{\widetilde{G}}\big[\log(1 - \mathcal{D}(\widetilde{g}^h_{(u,v)_i}, g))\big]\Big).$$

$$\mathcal{D}(g^h_{(u,v)_i}, g) = \sigma((g^h_{(u,v)_i})^T W_b g).$$

J, Cao et al. Bipartite Graph Embedding via Mutual Information Maximization. WSDM'21

Structure-Level Contrast

Local-Context Contrast: NCL (Lin et al. 2022)

CL Loss

$$\mathcal{L}_P^U = \sum_{u \in \mathcal{U}} -\log \frac{\exp(\mathbf{e}_u \cdot \mathbf{c}_i / \tau)}{\sum_{\mathbf{c}_j \in C} \exp(\mathbf{e}_u \cdot \mathbf{c}_j / \tau)}$$

Similar users/items tend to fall in neighboring embedding space, and the prototypes are the center of clusters that represent a group of semantic neighbors.

Z, Lin et al. Improving Graph Collaborative Filtering with Neighborhood-enriched Contrastive Learning. WWW'22

Feature-Level Contrast

Formulation

$$f_\theta^* = \underset{f_\theta, g_{\phi_s}}{\arg\min} \, \mathcal{L}_{\mathcal{MI}}\big(g_{\phi_s}(f_\theta(\tilde{\mathbf{x}}_i), f_\theta(\tilde{\mathbf{x}}_j))\big)$$

$\tilde{\mathbf{x}}_i$ and $\tilde{\mathbf{x}}_j$ are feature-level augmentations which can be obtained by modifying the input feature
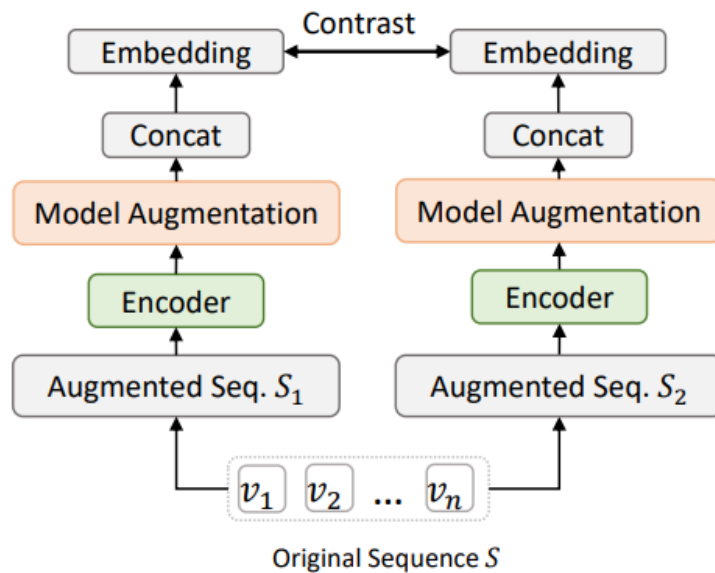
SSL4ItemRec (Yao et al. 2022)

T, Yao et al. Self-supervised Learning for Large-scale Item Recommendations. CIKM'21
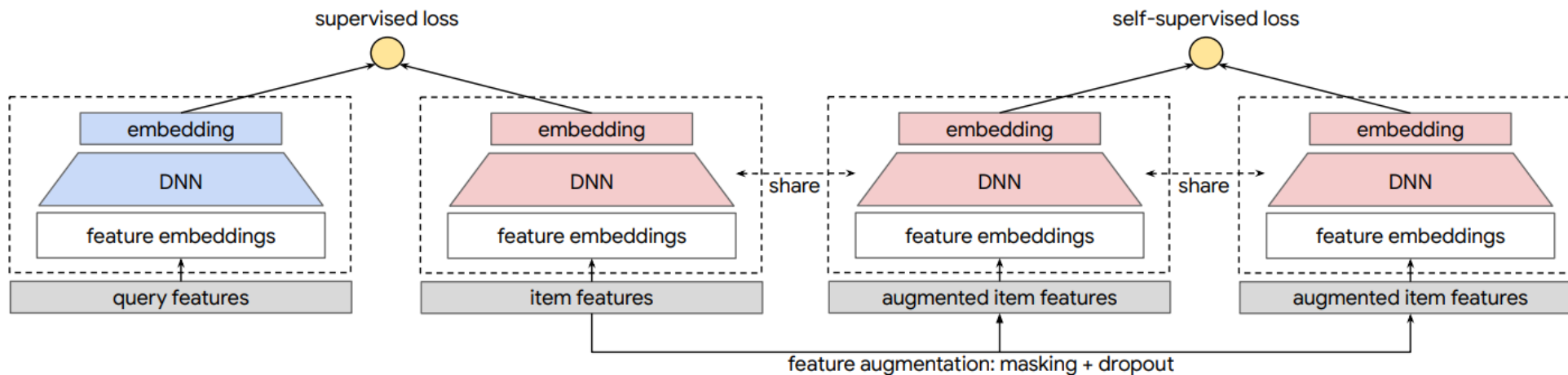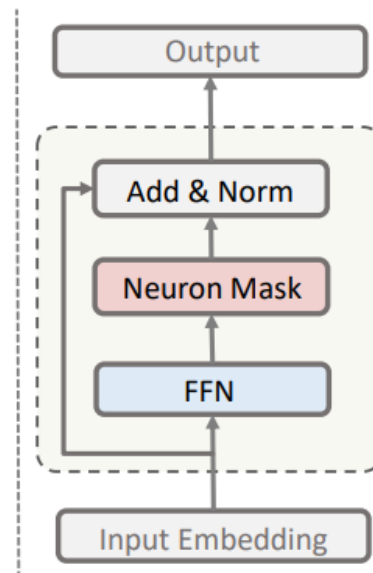
Model-Level Contrast

Formulation

$$f_\theta^* = \arg\min_{f_\theta, g_{\phi_s}} \mathcal{L}_{\mathcal{MI}}\big(g_{\phi_s}(f_{\theta'}(\mathcal{D}), f_{\theta''}(\mathcal{D}))\big)$$

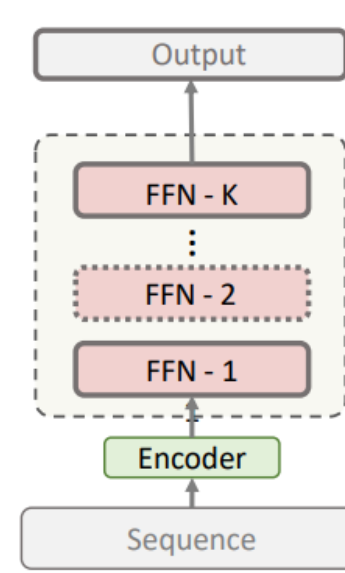$f_{\theta'}$ and $f_{\theta''}$ are perturbed versions of f_θ.
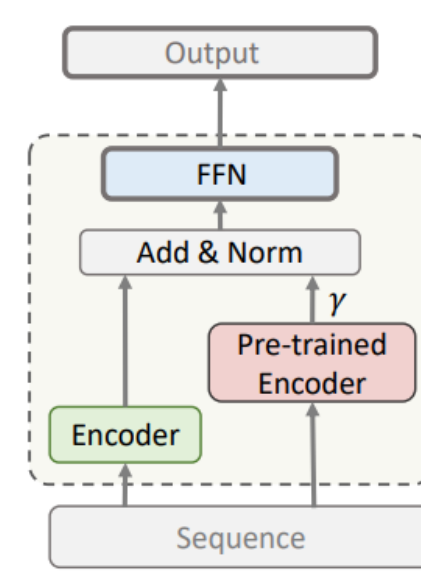
SRMA (Liu et al. 2022)

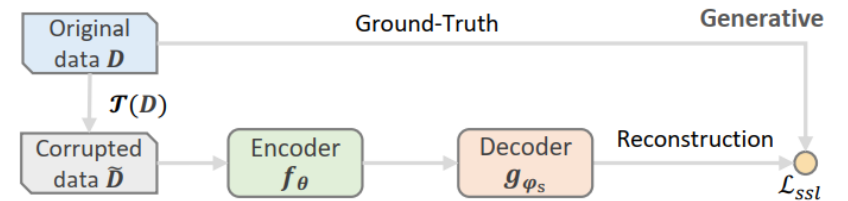**(a) Contrastive Self-supervised Learning**   **(b) Neuron Masking**   **(c) Layer Dropping**   **(d) Encoder Complementing**

Z, Liu et al. Improving Contrastive Learning With Model Augmentation. arXiv'22

Generative SSR Methods

Generative Methods

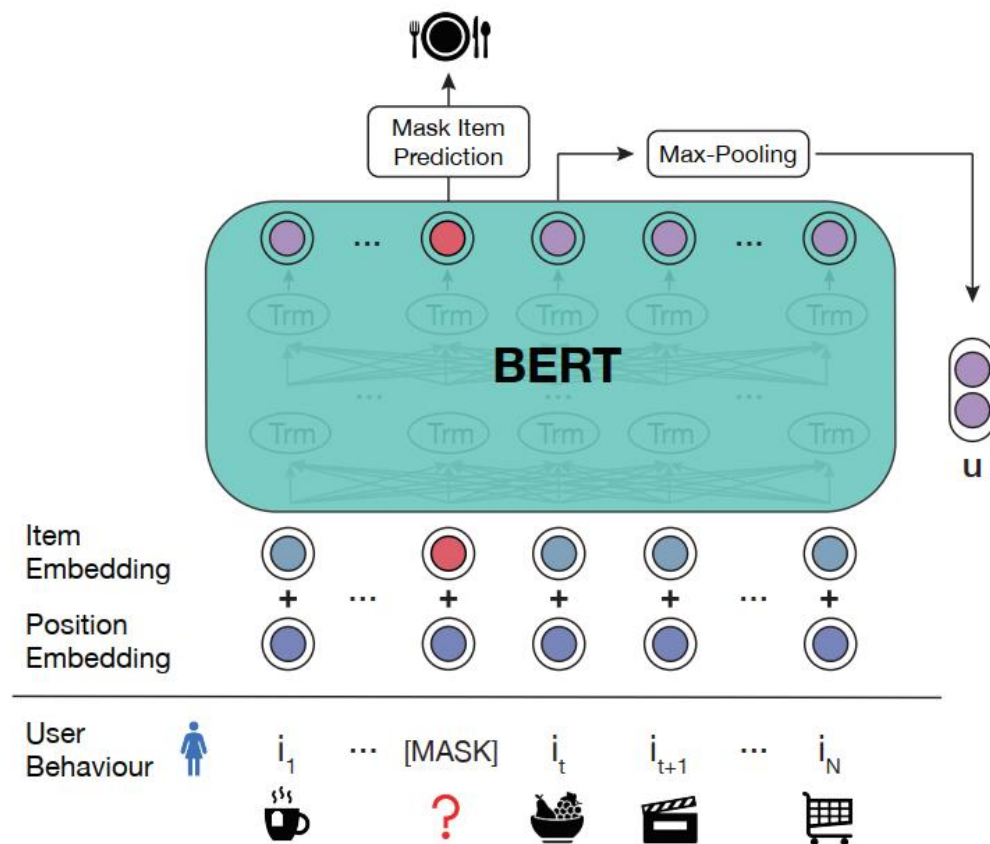Reconstructing the original input with its corrupted version.
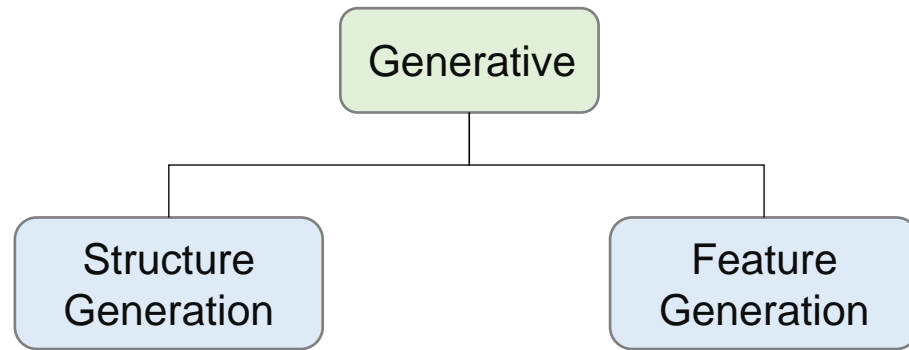
$$f_\theta^* = \arg\min_{f_\theta, g_{\phi_s}} \mathcal{L}_{ssl}\left(g_{\phi_s}\left(f_\theta(\tilde{\mathcal{D}})\right), \mathcal{D}\right)$$

According to the reconstruction objectives, generative SSR methods are divided into two categories: **Structure Generation** and **Feature Generation**.

Sequence Reconstruction

UPRec (Xiao et al. 2022)

Pre-trained with the masked-item-prediction method.

$$f_\theta^* = \underset{f_\theta, g_{\phi_s}}{\arg\min} \mathcal{L}_{ssl}\left(g_{\phi_s}\left(f_\theta(\tilde{\mathcal{S}})\right), S\right)$$
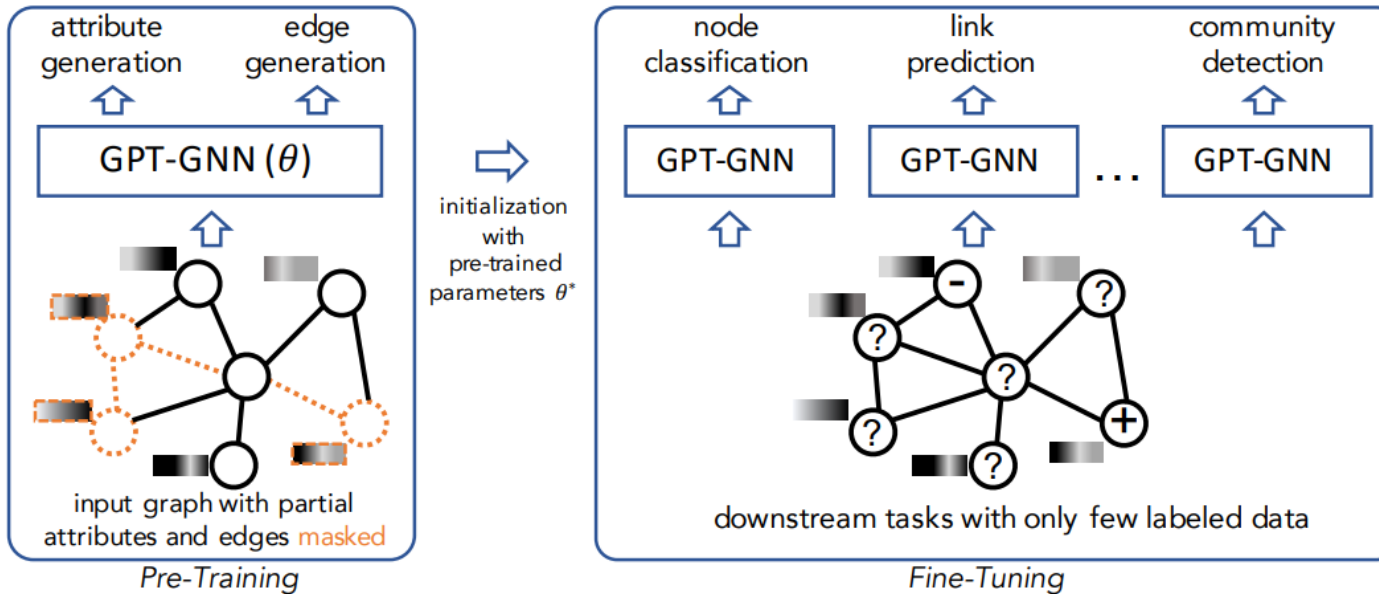
$$\mathcal{L} = \frac{1}{|S_u^m|} \sum_{i_m \in S_u^m} -\log P\left(i_m = i_m^* \mid \tilde{S}_u\right)$$

Fine-tuned by predicting the last item of each sequence.

C, Xiao et al. UPRec: User-Aware Pre-training for Recommender Systems. arXiv'21

Structure Generation

Graph Reconstruction

GPT-GNN (Hu et al. 2020)

A GNN is pre-trained with the self-supervised generative task — structure generations. Second, the pretrained model and its parameters are then used to initialize models for downstream tasks on the input graph or graphs of the same domain such as recommendation.

$$f_\theta^* = \arg\min_{f_\theta, g_{\phi_s}} \mathcal{L}_{ssl}\left(g_{\phi_s}\left(f_\theta(\tilde{\mathcal{G}})\right), \mathbf{A}\right)$$

Z, Hu et al. GPT-GNN: Generative Pre-Training of Graph Neural Networks. KDD'20

Feature Generation

GPT-GNN (Hu et al. 2020)
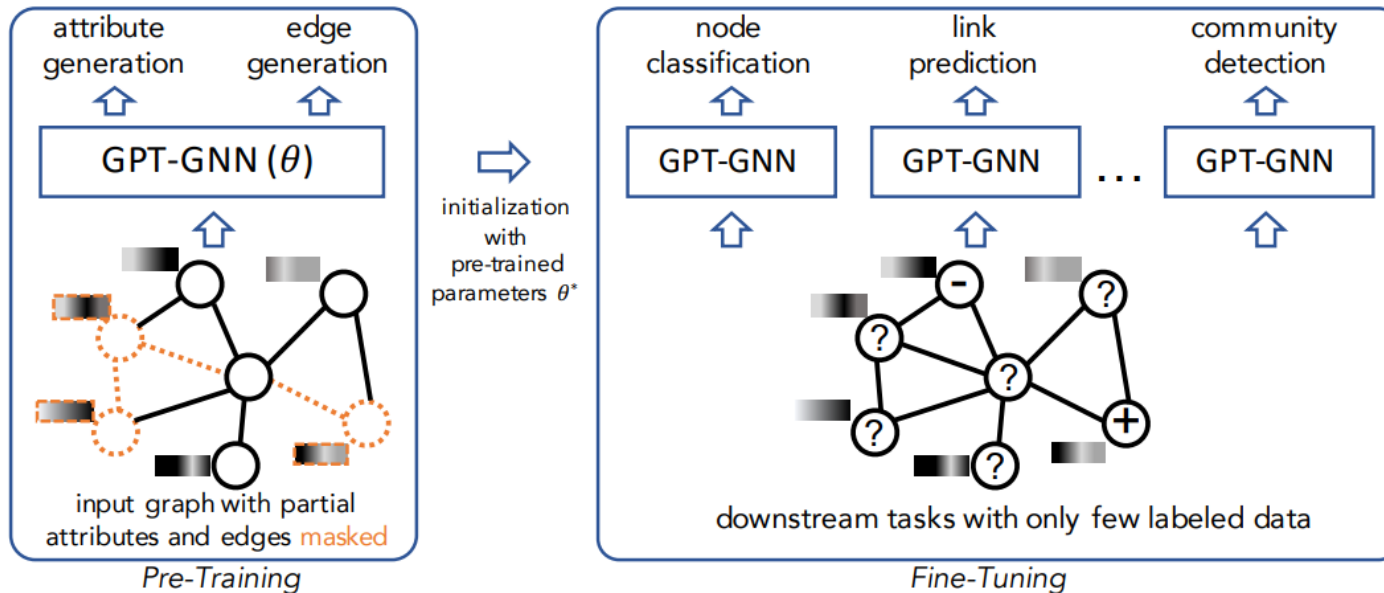

A GNN is pre-trained with the self-supervised generative task — feature generations. Second, the pretrained model and its parameters are then used to initialize models for downstream tasks on the input graph or graphs of the same domain such as recommendation.

$$f_\theta^* = \arg\min_{f_\theta, g_{\phi_s}} \left\| g_{\phi_s}\left(f_\theta(\tilde{\mathcal{D}})\right) - \mathbf{X} \right\|^2$$

Z, Hu et al. GPT-GNN: Generative Pre-Training of Graph Neural Networks. KDD'20

Feature Generation

PT-GNN (Hao et al. 2021)

$$f_\theta^* = \arg\min_{f_\theta, g_{\phi_s}} \left\| g_{\phi_s}\left(f_\theta(\tilde{\mathcal{D}})\right) - \mathbf{X} \right\|^2$$
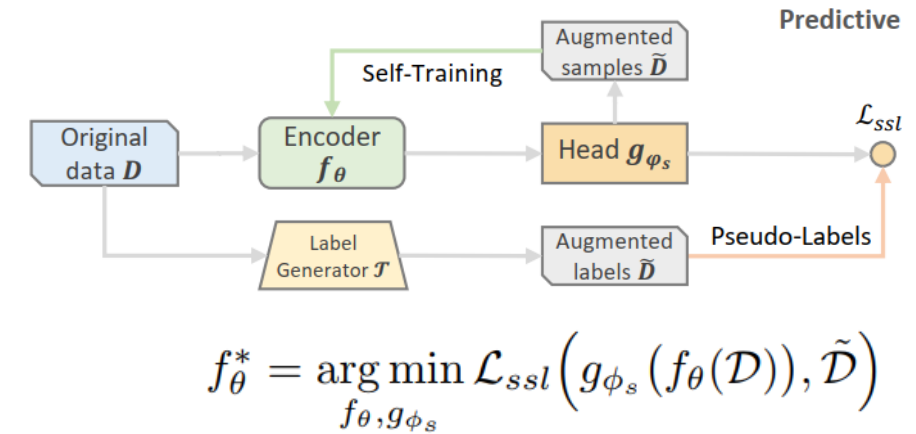
B, Hao et al. GPT-GNN: Generative Pre-Training of Graph Neural Networks. WSDM'21
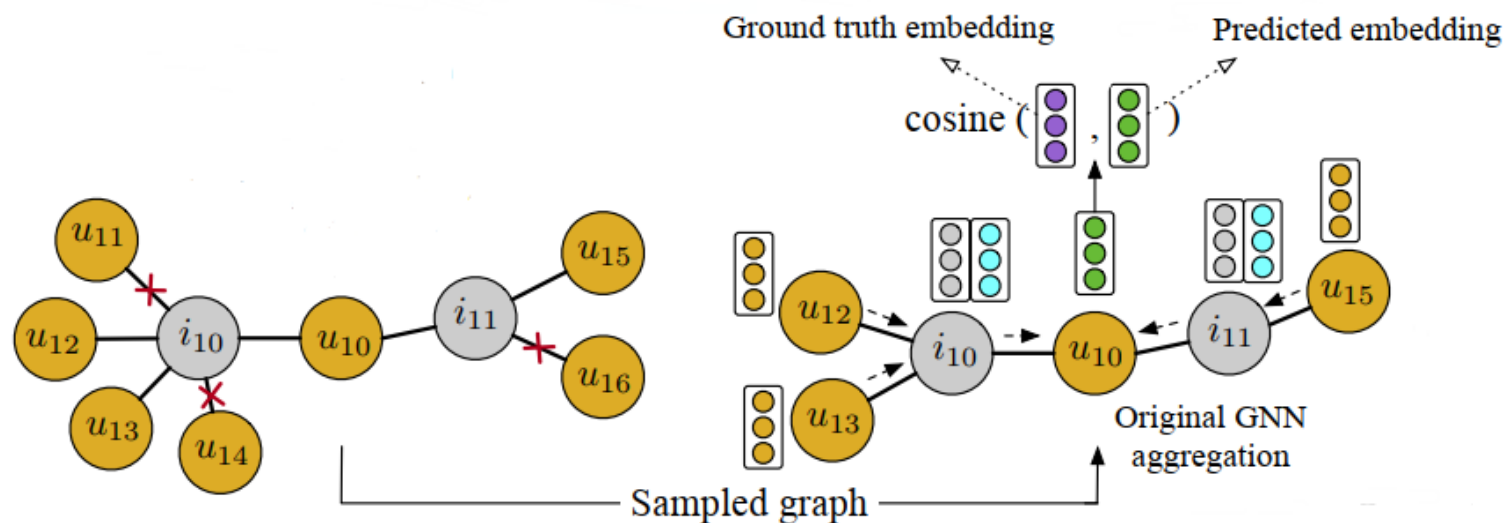
Predictive SSR Methods

Predictive Methods

Deal with the self-generated supervisory signals obtained from the complete original data.

$$f_\theta^* = \arg\min_{f_\theta, g_{\phi_s}} \mathcal{L}_{ssl}\left(g_{\phi_s}\left(f_\theta(\mathcal{D})\right), \tilde{\mathcal{D}}\right)$$
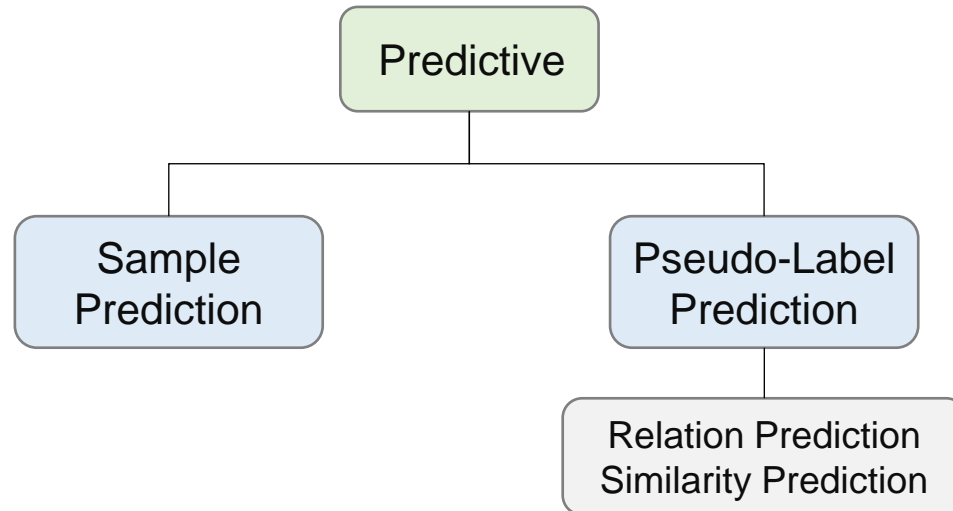
According to what the predictive pretext task predicts, predictive methods can be categorized into two branches: **Sample Prediction** and **Pseudo-Labels Prediction**.

Sample Prediction

ASReP (Liu et al. 2021)
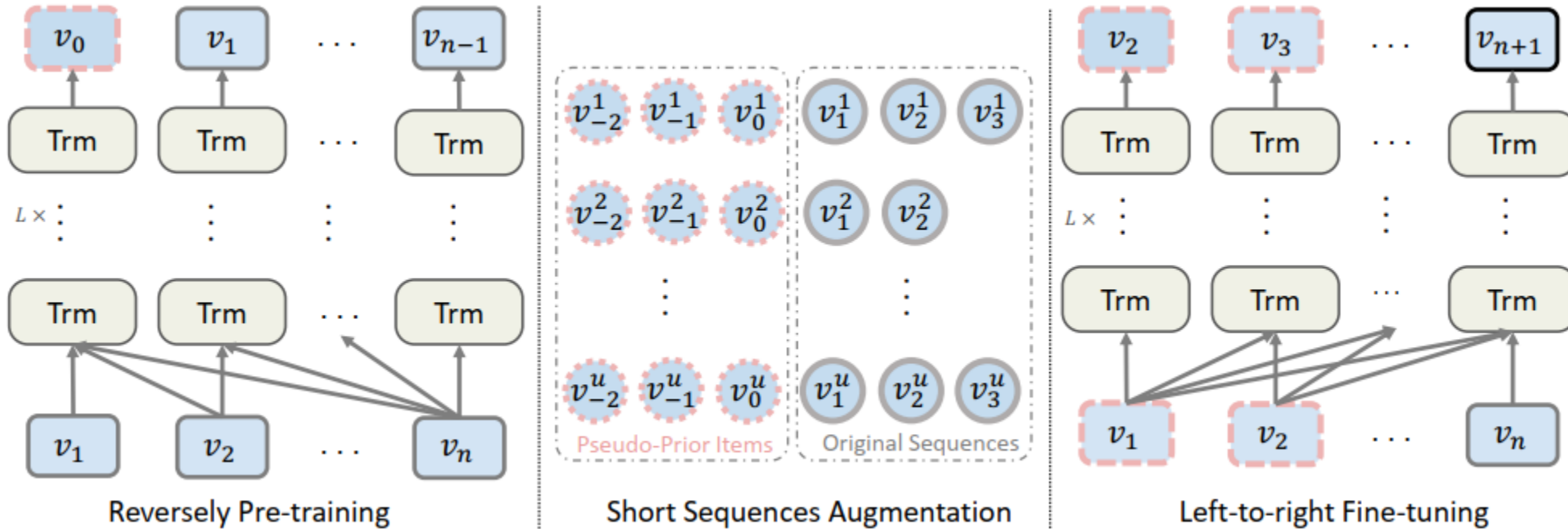
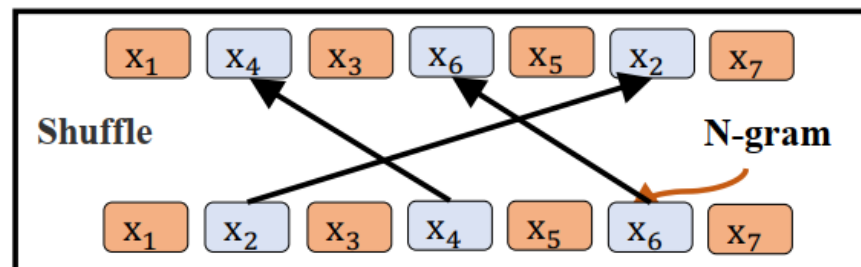Firstly pre-train a transformer in a reverse direction to predict prior items.

Use this transformer to generate fabricated historical items at the beginning of short sequences

Fine-tune the transformer using these augmented sequences from the time order to predict the next item.

Z, Liu et al. Augmenting Sequential Recommendation with Pseudo-Prior Items via Reversely Pre-training Transformer. SIGIR'21

Pseudo-Label Prediction · Relation Prediction

SSI (Song et al. 2021)

(a) Temporal Consistency Task
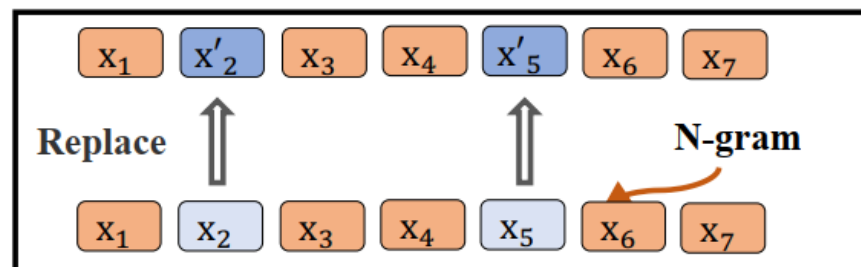
(b) Persona Consistency Task

Shuffle/replace a portion of items in a given sequence, and then predicts if the modified sequence is in the original order/from the same user.

CHEST (Wang et al. 2021)

Path Sampling

Subgraph

UIUI
UIAI
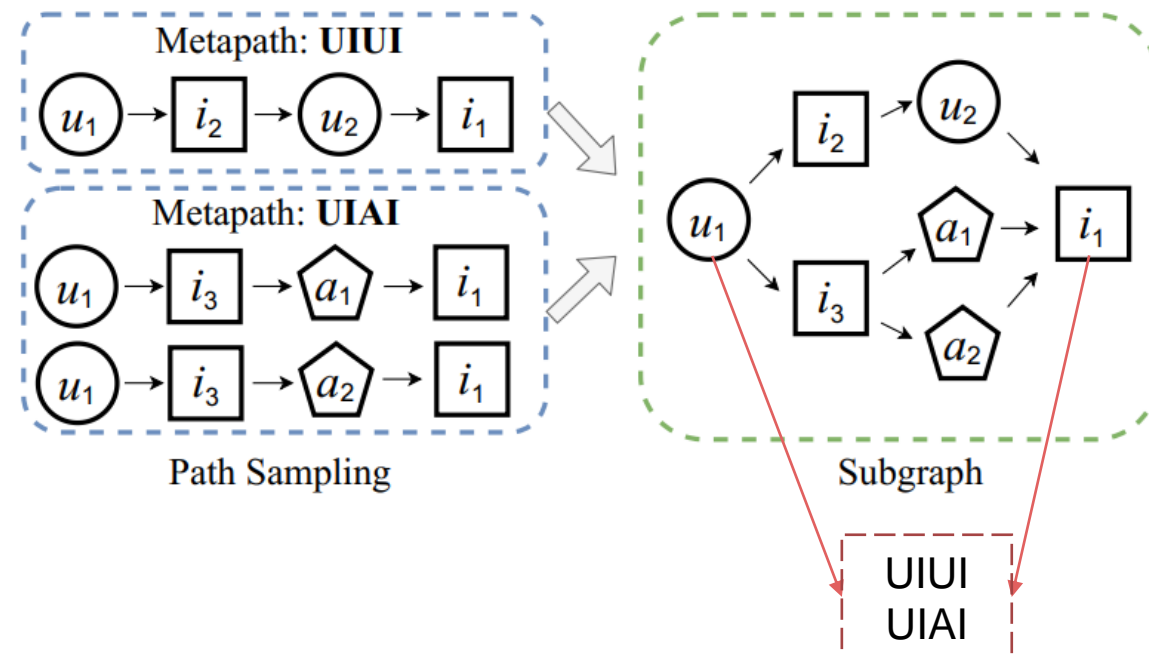
Predict if there exists a path instance of a specific meta-path between a user-item pair.

Y, Song et al. Improving Sequential Recommendation Consistency with Self-Supervised Imitation. IJCAI'21

H, Wang et al. Curriculum Pre-Training Heterogeneous Subgraph Transformer for Top-N Recommendation. arXiv'21

Pseudo-Label Prediction   Similarity Prediction

BUIR (Lee et al. 2021)

$$q_\theta(f_\theta(u)), q_\theta(f_\theta(v))$$

gradient

l_2 Loss

Predictor

stop-gradient

$f_\theta(u), f_\theta(v)$ $f_\xi(v), f_\xi(u)$

momentum

Online Encoder   Target Encoder

u v u v

user u | 0 | 1 | 0 | 0 | 0 | 0 | ⋯ | 0 | 0 | 0 | 1 | 0 | 0 | ⋯ | item v

Input Interaction (u, v)
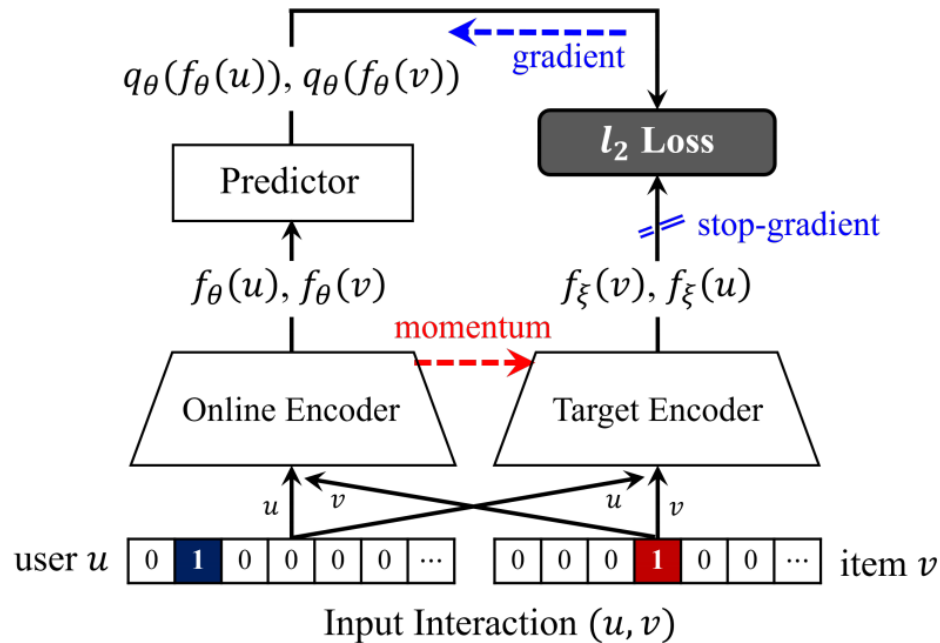
The online network is fed with the user representation and is trained to predict the item representation output by the target network and vice versa.

**CLUE (Cheng et al. 2021)**

D, Lee et al. Bootstrapping User and Item Representations for One-Class Collaborative Filtering. SIGIR'21
M, Cheng et al. Learning Transferable User Representations with Sequential Behaviors via Contrastive Pre-training. ICDM'21
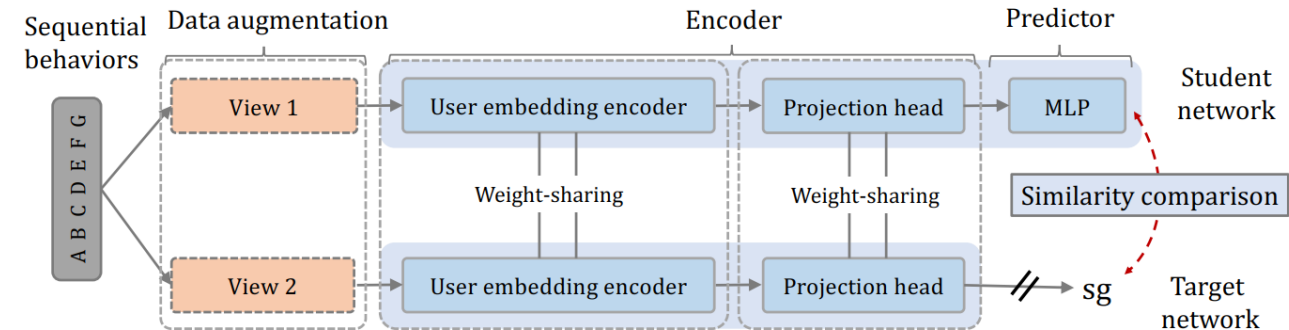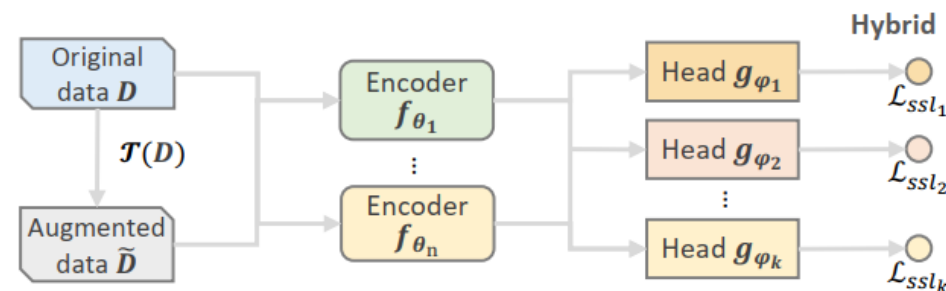
Hybrid SSR Methods

Hybrid Methods

Assemble multiple types of pretext tasks to take advantage of different self-supervision signals.
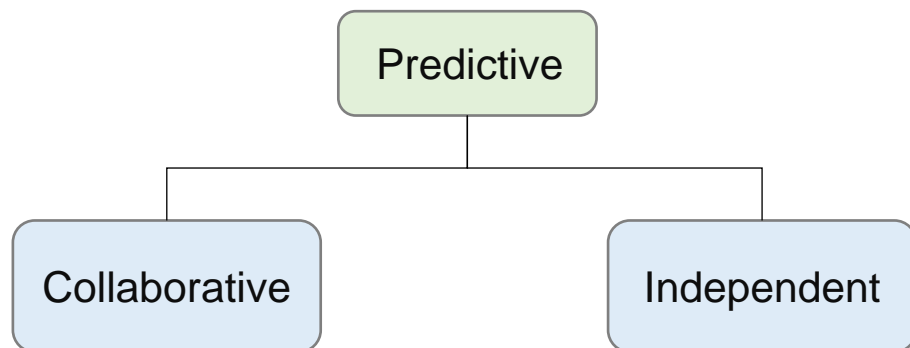
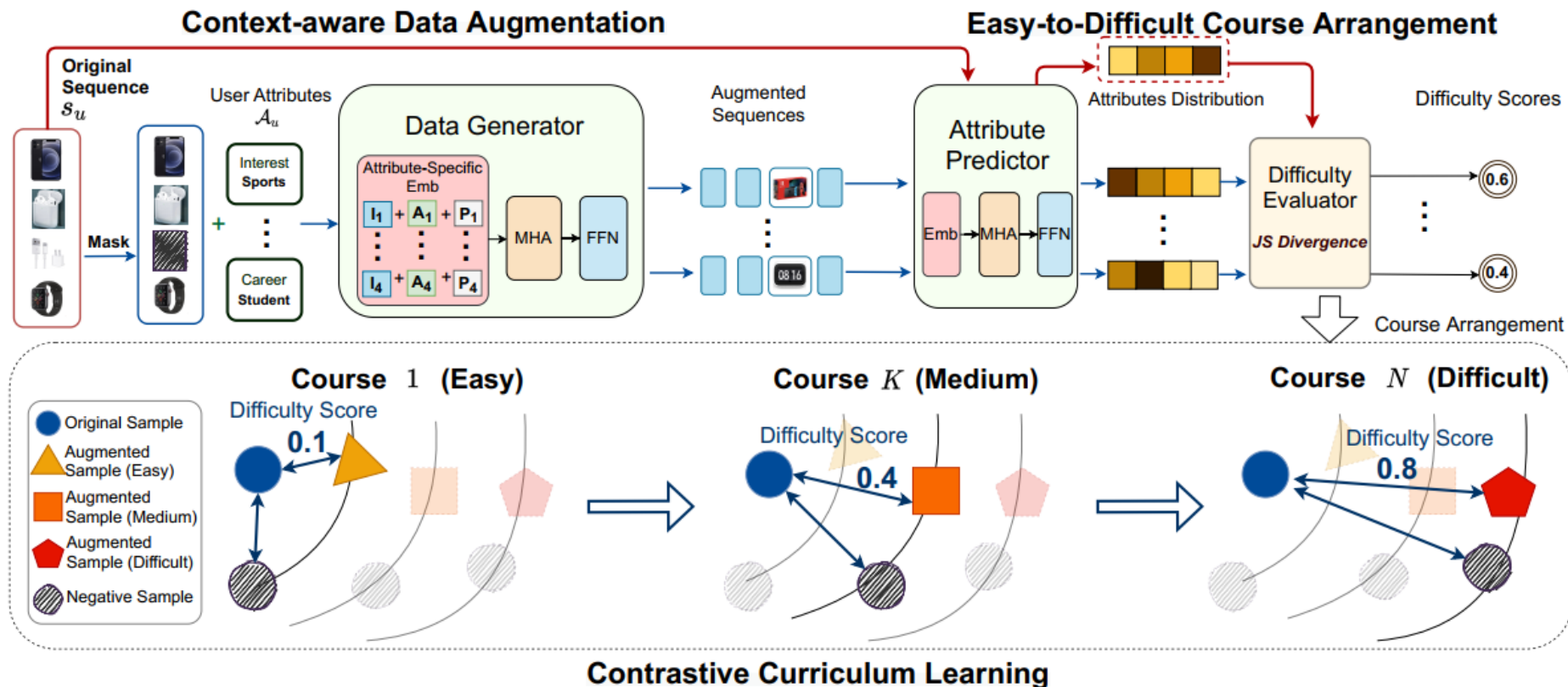**Collaborative**: Different pretext tasks collaborate in a way to serve one primary pretext task.

Independent: There are no correlations between different pretext tasks and they work independently.

Collaborative Methods

CCL (Bian et al. 2021)

The generative task serves the contrastive task.

S, Bian et al. Contrastive Curriculum Learning for Sequential User Behavior Modeling via Data Augmentation. CIKM'21

Independent Methods

UPRec (Xiao et al. 2022)
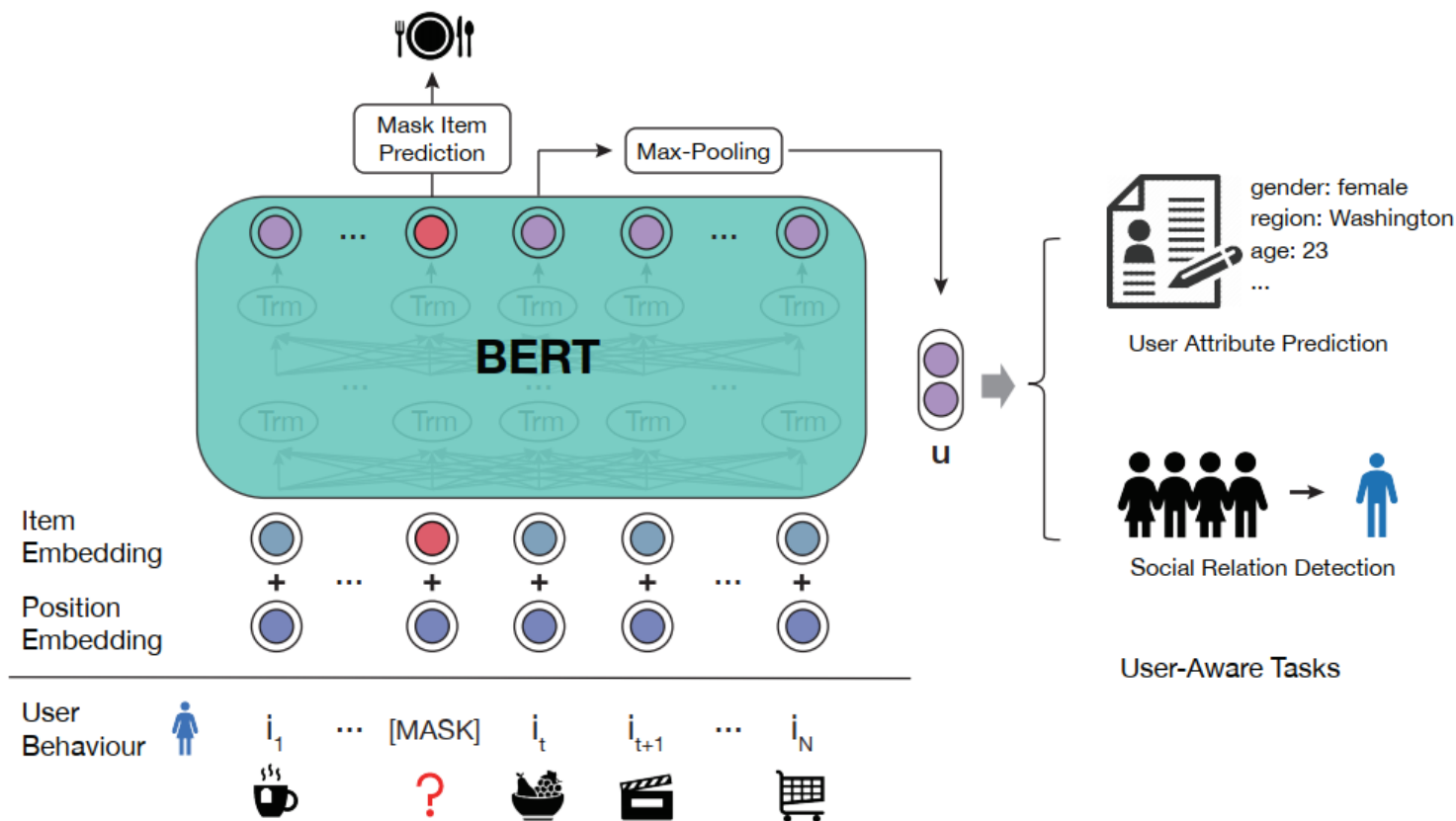
One Generative Task
Two Predictive Tasks

$$\mathcal{L} = \lambda_1 \mathcal{L}_{\mathrm{MIP}} + \lambda_2 \mathcal{L}_{\mathrm{UAP}} + \lambda_3 \mathcal{L}_{\mathrm{SRD}}$$

C, Xiao et al. UPRec: User-Aware Pre-training for Recommender Systems. arXiv'21

Pros and Cons

Contrastive

Pros 😁

Flexible to design augmentations and pretext tasks.

Cons 😟

Often compromised by low-quality augmentations.

Generative

Pros 😁

Can follow the successful experience for training masked language models

Cons 😟

Confronted with heavy computation when building general-purpose models.

Predictive

Pros 😁

Acquire samples and pseudo-labels in more dynamic and flexible ways.

Cons 😟

Collect the pseudo-labels based on heuristics, without assessing how relevant these labels and predictive tasks are to recommendation.

Hybrid

Pros 😁
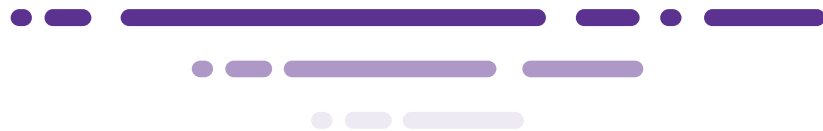
Can get enhanced and comprehensive self-supervision.

Cons 😟

Confronted with the problem of coordinating multiple pretext tasks.
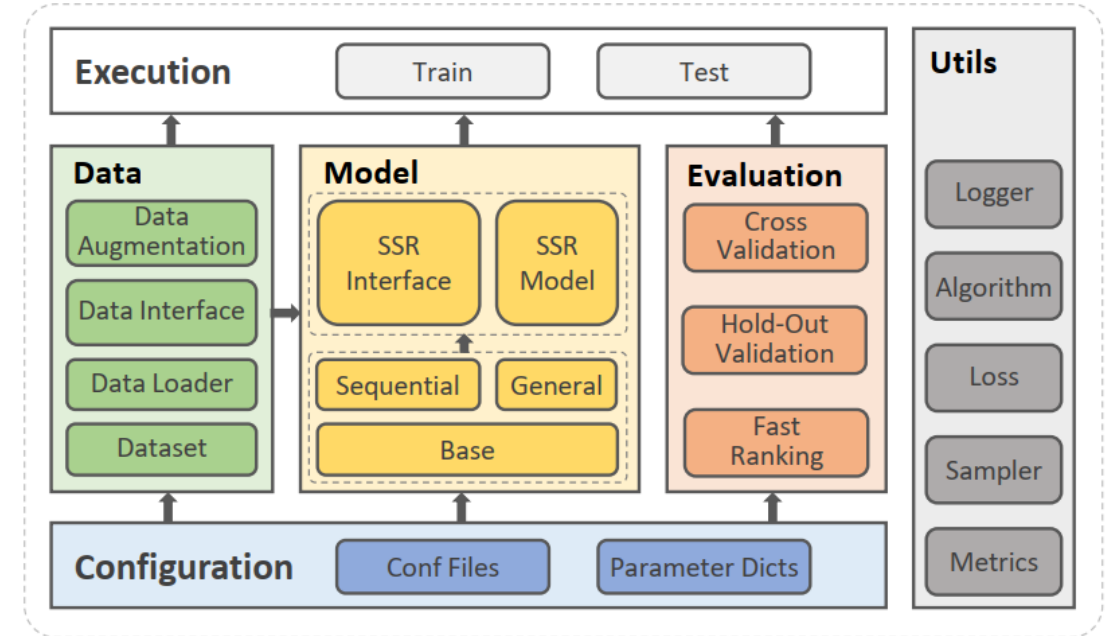
05

SELFRec

- A Library for self-supervised recommendation

Tutorial

SELFRec

- SELFRec incorporates multiple high-quality datasets and metrics.

- There are more than 10+ state-of-the-art SSR methods implemented in SELFRec for fair empirical comparison.

- SELFRec is developed with Python 3.7+, Tensorflow 1.14+ and PyTorch 1.7+.

- SELFRec provides a set of simple and high-level interfaces, by which new SSR models can be easily added in a plug-and-play fashion.

- SELFRec decouples the model design from other procedures.

Execution | Train | Test | **Utils**
Logger
Data | **Model** | **Evaluation**
Data Augmentation | SSR Interface | SSR Model | Cross Validation | Algorithm
Data Interface | | | Hold-Out Validation | Loss
Data Loader | Sequential | General | | Sampler
Dataset | Base | Fast Ranking | Metrics
Configuration | Conf Files | Parameter Dicts

SELFREC
Framework for Self-Supervised Recommendation

https://github.com/Coder-Yu/SELFRec

06

Limitations and Future Research

Limitations and Future Research

- **Theory for Augmentation Selection**
  - Cannot seamlessly transplant augmentation approaches designed for other fields to recommendation
  - Most augmentation approaches are based on heuristics
  - Cumbersome trial-and-error work is needed to search for useful augmentations

  A solid recommendation-specific theoretical foundation for the augmentation selection is urgently needed.

Example

Too much noise   "Sweet spot"   Missing info

What is the criterion for augmentation selection in recommendation?

In vision tasks, views should not share too much information (left) or too little information (right), but should find an optimal mix (the "sweet spot", middle) that maximizes the downstream performance.

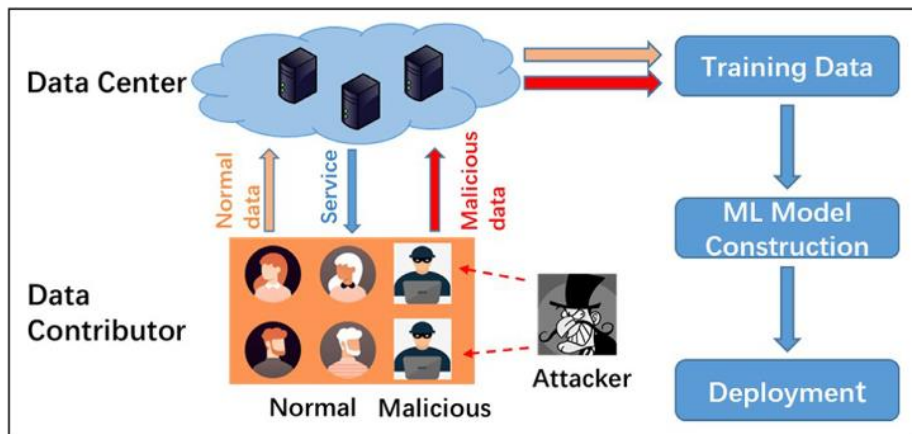Y, Tian et al. Understanding View Selection for Contrastive Learning. Google AI Blog

Limitations and Future Research

- **Attacking and Defending in Pre-trained Recommendation Models**
    - Recommender systems are vulnerable to the data poisoning attack
    - It remains unknown if the recommendation models pretrained in a self-supervised way are robust to such attacks

Developing new-type attacks and defending self-supervised pretrained recommender systems against these attacks will be an interesting future research direction.

Example

Are pre-trained recommendation models robust to data poisoning attacks?

Pic from Poisoning attacks and countermeasures in intelligent networks: Status quo and prospects. Digital Communications and Networks
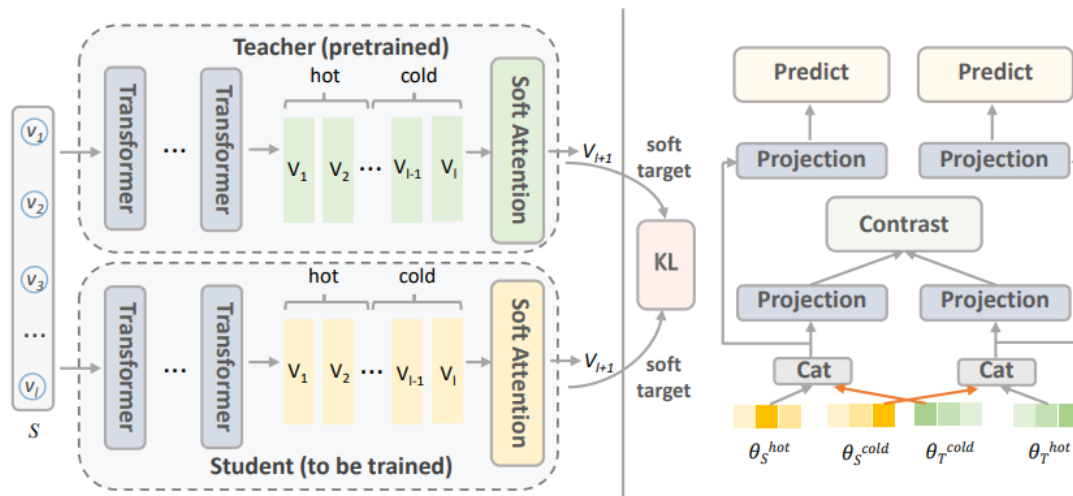
Limitations and Future Research

- **On-Device Self-Supervised Recommendation**
  - On-device recommender systems are compromised by the highly compressed model size and limited labeled data.
  - Combined with the technique of knowledge distillation, SSL may largely compensate for the accuracy degradation of on-device recommendation models.

  On-device self-supervised recommendation is still under-explored, and it deserves further study.

Example

How to empower on-device recommendation with SSL?

Pic from On-Device Next-Item Recommendation with Self-Supervised Knowledge Distillation. SIGIR'22

Limitations and Future Research

- **Towards General-Purpose Pre-Training**
    - Data in recommender systems is multi-modal and scenarios are rather diverse
    - Explore general-purpose recommendation models which are pre-trained with the multi-modal SSL on large-scale data.
    - Enable models to adapt to multiple downstream recommendation tasks with the cheap finetuning.

Example
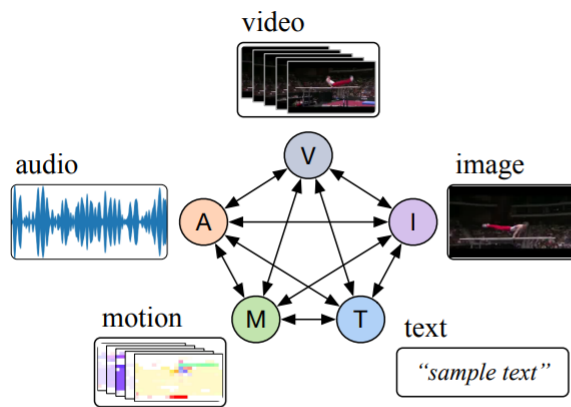
(a) Cross-modal supervision.
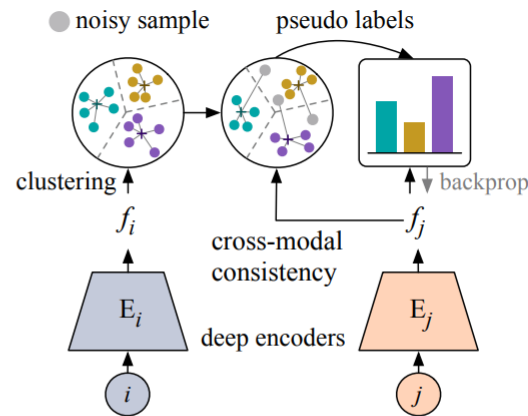
(b) Modality i ⟶ modality j.

How to develop a once and for all pretraining technique for diverse modalities and downstream tasks?

Thanks

Q&A