Scenario Shared Instance Modeling for Click-through Rate Prediction

Dugang Liu[1], Chaohua Yang[1], Yuwen Fu[2], Xing Tang[2], Gongfu Li[2], Fuyuan Lyu[3], Xiuqiang He[4], Zhong Ming[4]

{dugang.ldg, ych981203}@gmail.com, xing.tang@hotmail.com, fuyuan.lyu@mail.mcgill.ca, hexiuqiang@sztu.edu.cn, mingz@szu.edu.cn

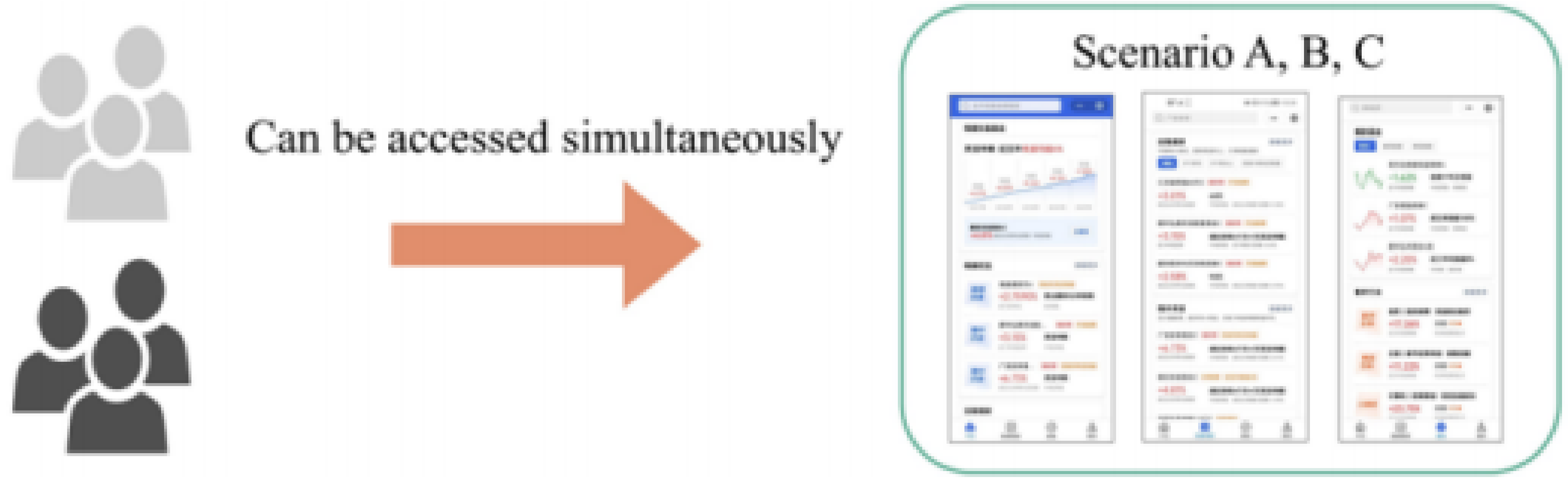
[1]CSSE, Shenzhen University, China

[2]Financial Technology (FiT), Tencent, China

[3]McGill University, Canada

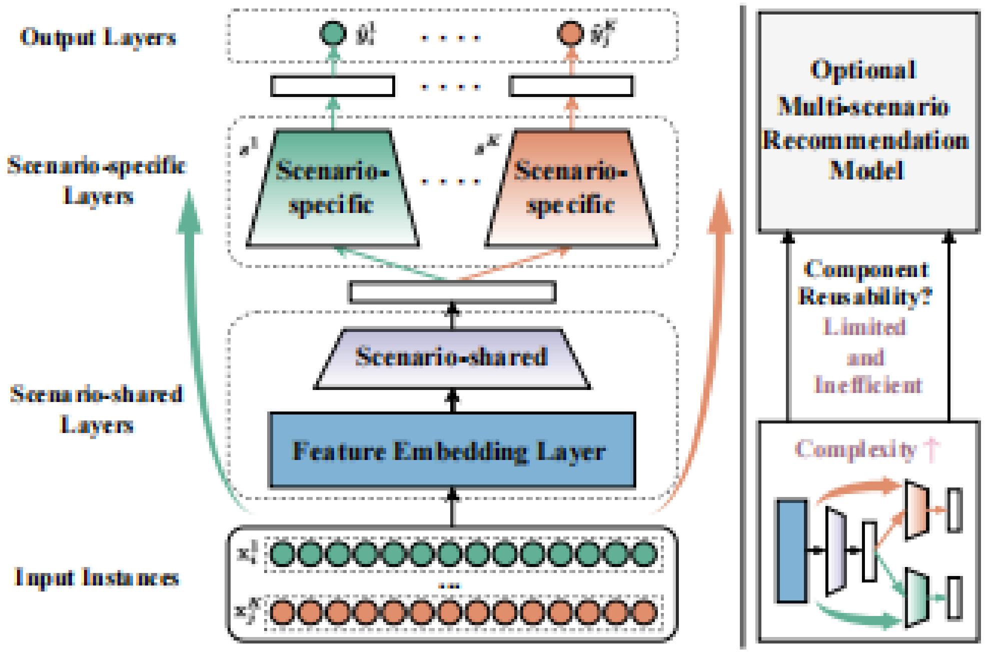
[4]Shenzhen Technology University, China

Multi-scenario Recommendation (MSR)



- Multi-scenario recommendation (MSR) has become a core component of various online platforms due to its excellent advantages in mitigating data sparsity and reducing maintenance costs [Sheng et al., 2021].
- Instead of training a model for each scenario, the MSR aims to effectively utilize user historical feedback in multiple scenarios through a unified model.

MSR Model Architecture



- Most existing MSR models follow a *shared-specific* architecture [Ma et al., 2018].
- The scenario-shared layers serve instances of all scenarios and output a shared representation to capture commonalities between scenarios.
- The scenario-specific layers consider the differences between different scenarios and generate specific predictions from the shared representation in the corresponding scenario.

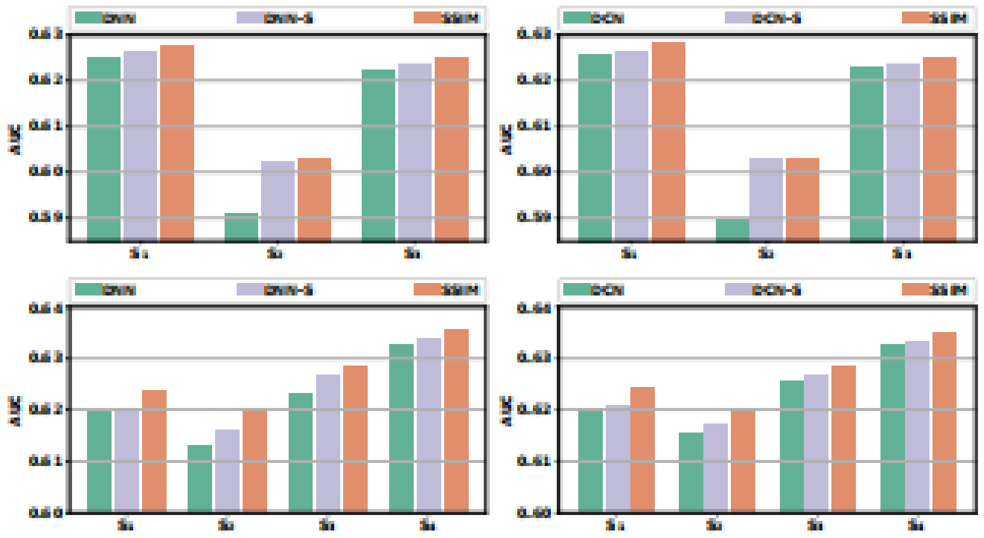
Existing Problems

- Existing MSR models focus more on implicitly extracting and exploiting valuable information across scenarios from the architectural level.
- Problem 1: To improve the effectiveness of this extraction process, they need to successively introduce different customized components in the architecture, but this also continuously increases the complexity of the MSR models and training overhead.
- Problem 2: The effectiveness of each component for extracting implicit information depends heavily on the specific MSR model deployed, limiting their reusability in a broader range of MSR models.

Motivation (1/2)

- To address these problems, we argue that we can switch perspectives from implicit information extraction at the architecture level to explicit extraction at the data level. A potential solution is to directly share valuable instances across scenarios to enhance model training for all scenarios in MSR.
- The idea behind it is that some instances in MSR may carry more required critical information and are suitable as hubs for information propagation, such as instances with similar users or items.
- For Problem 1: Since only a simple reuse operation of instances is required and the operation is model-free, this solution can be integrated with existing MSR models without significantly increasing training overhead.
- For Problem 2: Selected shared instances are easy to save and reuse.

Motivation (2/2)



- We conduct a preliminary experiment to verify the feasibility of this potential solution.
- We implement the MSR architecture using two backbone models, treating user-item interactions across all scenarios as shared instances to obtain model variants.
- As shown in the figure, the variant with full instance sharing (with the suffix '-S' in the legend) outperforms the original backbone based on implicit information extraction.
- This motivates us to develop a training framework to better select shared instances in MSR.

Multi-scenario Learning

- For K scenarios $\mathcal{S} = \{s^1, \cdots, s^K\}$, the multi-scenario training instances can denote $\mathcal{D} = \{(\mathbf{x}_i^k, y_i^k, s^k)\}_{k=1}^K$, where s^k is the k-th scenario, $\mathbf{x}_i^k \in \mathcal{X}$ is the i-th instance's feature vector in k-th scenario, and $y_i^k \in \{0, 1\}$ is corresponding label. Multi-scenario learning aims to train a model using the above training instance set, consisting of shared and specific components,

$$\hat{y}_i^k = f(\mathbf{x}_i^k \mid \theta, \{\theta_{s^k}\}), \tag{1}$$

where $f(\cdot)$ is the mapping function, and θ and $\{\theta_{s^k}\}$ are the scenario-shared and scenario-specific parameters, respectively.
- The cross-entropy function is usually used to optimize the model, where $|s^k|$ denotes the number of instances in scenario s^k and $l(\cdot)$ is the cross-entropy loss.

$$\mathcal{L} = \sum_{k=1}^{K} \sum_{i=1}^{|s^k|} l(y_i^k, \hat{y}_i^k), \tag{2}$$

Shared Instance Selection Problem for MSR (1/3)

- During the training process, training instances $\mathcal{D}$ are typically organized into n mini-batches containing instances from multiple scenarios, $\mathbf{B} = [B_1, B_2, \cdots, B_n]$, where the j-th mini-batch with m instances is denoted as,

$$B_j = \{(\mathbf{x}^{\star}_{j,i}, y^{\star}_{j,i}, s^{\star})\}_{i=1}^{m}, \tag{3}$$

where $\star \in \{1, 2, \cdots, K\}$ represents the scenario corresponding to the i-th instance in this mini-batch. In the traditional MSR model, each instance in B_j can only affect the scenario-shared parameters θ and scenario-specific parameters θ_{s^k} to which it belongs and cannot act on other scenario-specific parameters.

Shared Instance Selection Problem for MSR (2/3)

- Hence, the shared instance selection problem in MSR can be defined as learning the shared instance mask operation **G** to select shared instances beneficial for all scenarios from **B**,

$$\mathbf{B} \odot \mathbf{G} \rightarrow \mathbf{B}^{sh} = [B_1', B_2', \cdots, B_n'], \tag{4}$$

where $\mathbf{G} \in \{0,1\}^{m*n}$, $\odot$ denotes element-wise multiplication, and $\mathbf{B}^{sh}$ is a shared instance set obtained by removing non-shared instances in each batch, and each instance in $\mathbf{B}^{sh}$ can be regarded as a scenario-shared instance. This can be directly done by passing these instances through all scenario-specific networks simultaneously and obtaining a set of losses associated with their labels,

$$\mathcal{L}^{sh} = \sum_{i=1}^{|\mathbf{B}^{sh}|} \sum_{k'=1}^{K} l(y_i^k, \hat{y}_i^{k'}), \tag{5}$$

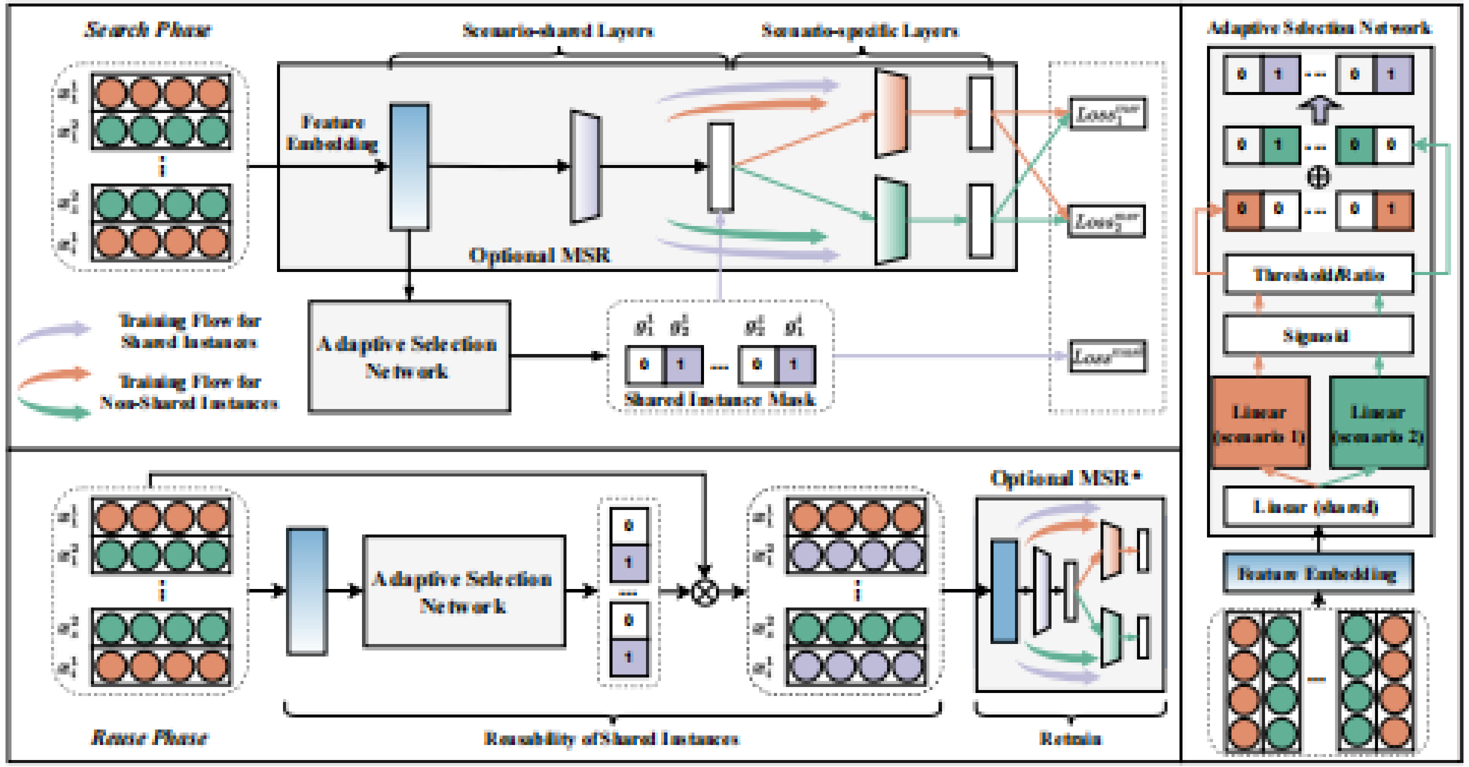
Shared Instance Selection Problem for MSR (3/3)

- Finally, we can formulate our problem as follows,

$$\min_{\Theta,\mathbf{G}} \mathcal{L}(\mathcal{D} \setminus \mathbf{B}^{sh}) + \mathcal{L}^{sh}(\mathbf{B}^{sh}). \tag{6}$$

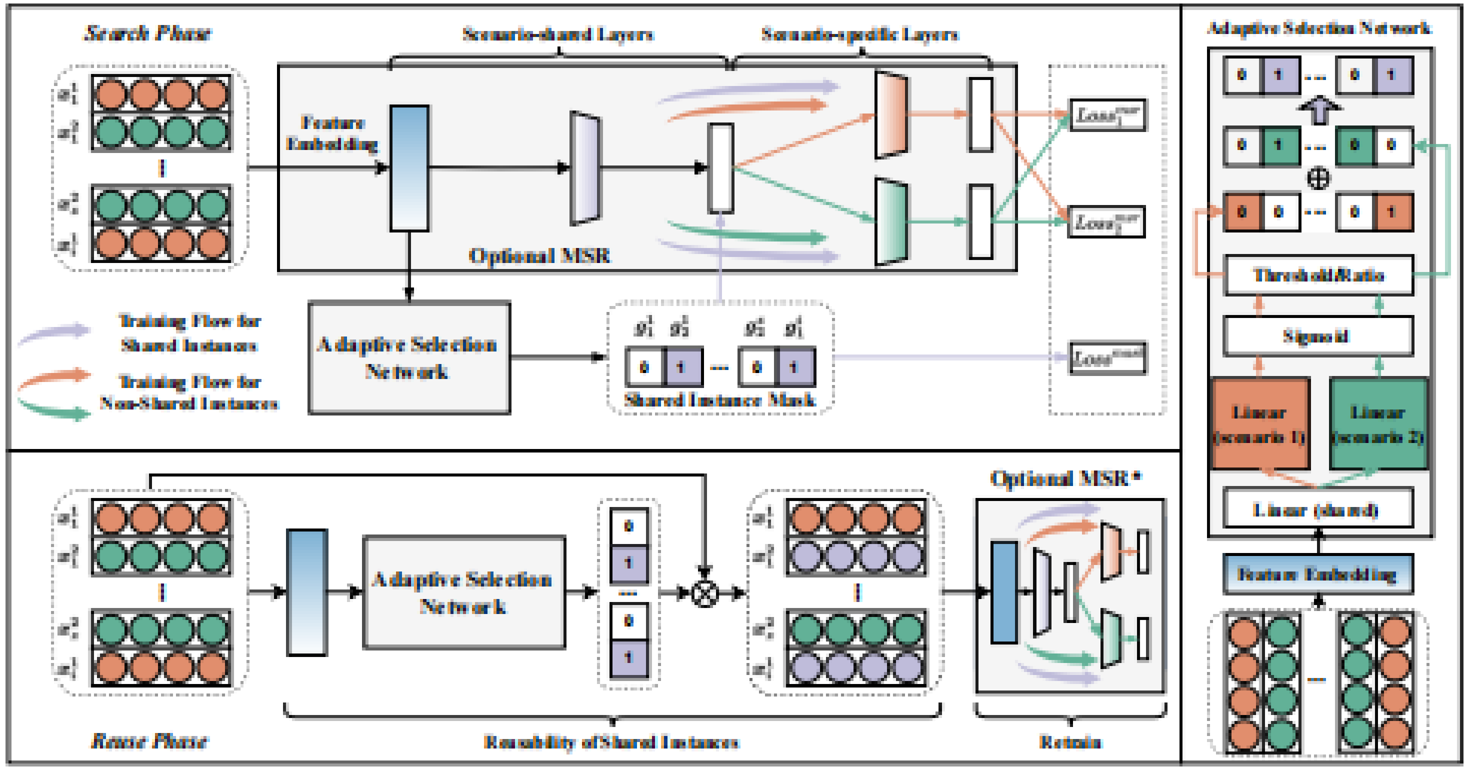
  where $\Theta = \{\theta, \{\theta_{s^k}\}_{k=1}^{K}\}$ denotes the network parameters, and $\mathcal{D} \setminus \mathbf{B}^{sh}$ represents the part of the training instances that do not belong to the multi-scenario shared instances, i.e., they are still scenario-specific instances that follow the traditional MBR training.

Architecture (1/2)



The SSIM framework includes two phases: search and reuse, and two key components: the backbone MSR model and the adaptive selection network.

Architecture (2/2)



- In the search phase, SSIM will use a backbone MSR model as a synergy to train the adaptive selection network to select reasonable shared instances.
- In the reuse phase, SSIM will train a target MSR model based on the selection results of the above well-trained adaptive selection network.

Search Phase (1/7)

- The search phase aims to obtain an adaptive selection network that can efficiently select a set of scenario-shared instances from all scenario instances.
- To guide the adaptive selection network to more accurately identify those shared instances beneficial to MSR training, we need to introduce a backbone MSR model as a practitioner of the selection results of the adaptive selection network.
- The adaptive selection network is based on an end-to-end training form. It can continuously adjust its selection strategy according to the optimization process of the backbone MSR model.

Search Phase (2/7)

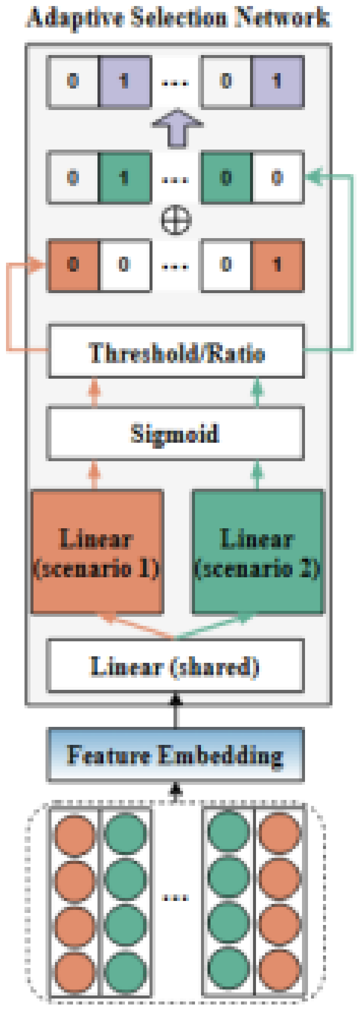
- To better represent and utilize features, given a current mini-batch B_j, we first encode each instance $\mathbf{x}_{j,i}^{k}$ into a low-dimensional and dense real-value vector via an embedding layer with an embedding table,

$$\mathbf{e}_{j,i}^{k} = \mathbf{E} \times \mathbf{x}_{j,i}^{k}, \tag{7}$$

where $\mathbf{E} \in \theta$ denotes the embedding table shared across scenarios.

Search Phase (3/7)

- After obtaining each instance's feature embedding, we expect a selection network to search for instances from B_j that benefit the backbone MSR model in learning in all scenarios.
- To better highlight the characteristics of each scenario, we propose an adaptive selection network with a hyper-network learning structure to make more reasonable selections of instances within the scenario.



Search Phase (4/7)

- Specifically, we first input the feature embedding $\mathbf{e}_{j,i}^{k}$ into a shared feedforward neural network (FNN) for shared feature extraction, where the l-th multi-layer perception (MLP) layer can be expressed as follows,

$$\mathbf{h}_{j,i}^{l} = \sigma\left(\mathbf{W}^{l}\mathbf{h}_{j,i}^{l-1} + \mathbf{b}^{l}\right), \quad l \in [1, L], \tag{8}$$

  where $\mathbf{W}^{l}$ and $\mathbf{b}^{l}$ denote the learnable weight matric and bias vector, $\sigma(\cdot)$ denotes the *ReLu* activation function, and L denotes the number of multilayer perception layers. Note that in the first MLP layer, $\mathbf{h}_{i}^{0} = \mathbf{e}_{i}^{k}$.

Search Phase (5/7)

- Then, to combine the characteristics of each scenario, we introduced a scenario-specific feedforward neural network component for each scenario based on the idea of the hyper-network, where each instance will obtain the final output probability through different components according to its own scenario identity.
- For example, assuming that the current instance $\mathbf{x}_{j,i}^k$ belongs to the k-th scenario, the calculation process is,

$$p_{j,i} = \alpha\left(\mathbf{W}^k \mathbf{h}_{j,i}^L + \mathbf{b}^k\right) \leftarrow (\mathbf{x}_{j,i}^k, s^k), \tag{9}$$

where $p_{j,i}$ denotes the shareability probability of the i-th instance in the j-th batch, $\alpha(\cdot)$ denotes the *Sigmoid* activation function, $\mathbf{W}^k$ and $\mathbf{b}^k$ denote the learnable scenario-specific weight matric and bias vector for k-th scenario.

Search Phase (6/7)

- After obtaining $p_{j,i}$, since we need to explicitly reuse a set of shared instances in all scenarios when training the backbone MSR model later, a discretization operation must be performed to determine each instance's selection.
- We use a straight-through estimator operation $S(\cdot)$ [Courbariaux et al., 2016] to implement this operation to maintain backward differentiability.

$$g_{j,i} = S(\mathbf{relu}(\mathbf{p}_{j,i} - \epsilon)), \tag{10}$$

where ϵ is a learnable threshold, $g_{j,i} \in \{0, 1\}$ is a shared instance mask for each instance $\mathbf{x}_{j,i}^{k}$ in the current batch B_j is to decide whether to share this instance in multi-scenario learning or not, and $g_{j,i} = 1$ and $g_{j,i} = 0$ represent "share" and "unshare", respectively.

Search Phase (7/7)

- Assuming that the selection results of all instances of each scenario after the adaptive selection network are represented as $\{\mathbf{g}_j^1, \cdots, \mathbf{g}_j^K\}$, we can merge them to obtain a final shared instance selection result for current mini-batch B_j.

$$\mathbf{G}_j = \mathbf{g}_j^1 + \mathbf{g}_j^2 + \cdots + \mathbf{g}_j^K. \tag{11}$$

- Based on the selection result of Eq.(11), we can classify the instances in B_j into a scenario-shared subset B_j^{sh} and a scenario-specific subset $B_j \setminus B_j^{sh}$. We can then arbitrarily choose a backbone MSR model as the executor to practice on each batch of selected results.

$$\text{MSR} \xrightarrow{B_j' = \{B_j^{sh}, B_j \setminus B_j^{sh}\}} \text{MSR}'. \tag{12}$$

Optimization (1/2)

- First, to guide the adaptive selection network in maximizing the probability of selecting valuable shareable instances and minimizing the probability of useless instances, we need to force the backbone MSR model after integrating shared instances to optimize towards better multi-scenario recommendation performance.

$$\mathcal{L}^{msr} = \mathcal{L}(\mathcal{D} \setminus \mathbf{B}^{sh}) + \mathcal{L}^{sh}(\mathbf{B}^{sh}). \tag{13}$$

  Note that the first and second terms in Eq. (13) are calculated by Eq. (2) and Eq. (5), respectively.

Optimization (2/2)

- Second, to more flexibly trade off the number of shared instances and MSR performance, we introduce a Kullback-Leibler (KL) divergence constraint between the shared instance mask G and the full selection mask (i.e., a vector with all values 1).

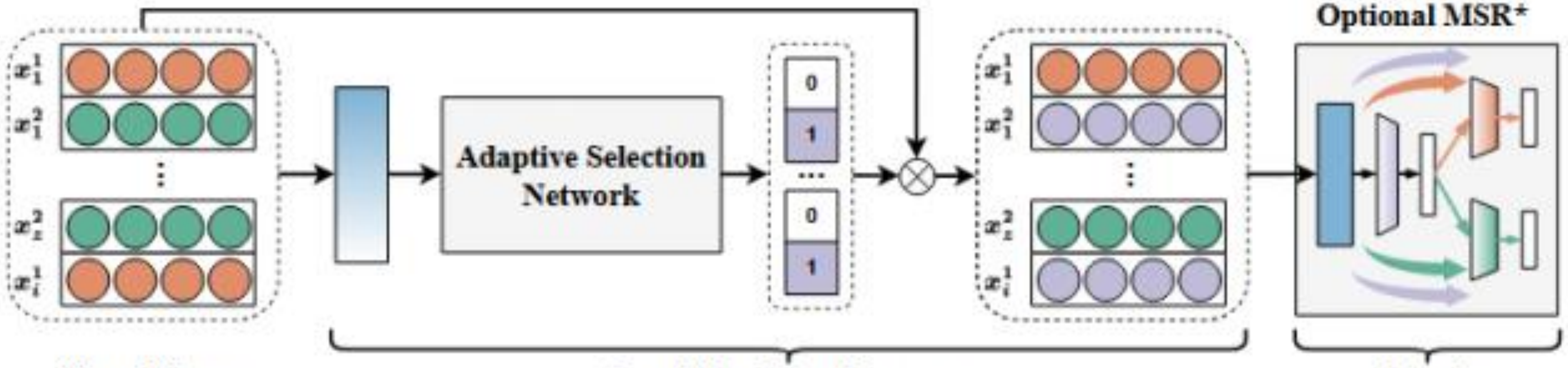
$$\mathcal{L}^{mask} = D_{KL}(\mathbf{1}||\mathbf{G}), \tag{14}$$

- Finally, combining Eq.(13) and Eq.(14), we get the final training objective becomes,

$$\min_{\Theta,\mathbf{G}} \mathcal{L}_{SSIM} = \mathcal{L}^{msr} + \lambda\mathcal{L}^{mask}. \tag{15}$$

where λ is the control weight.

Reuse Phase (1/2)



- After completing the search phase, our SSIM can obtain a well-trained adaptive selection network.
- In the reuse phase, we can construct a "new training set" that includes the shared instances selected by this adaptive selection network and the remaining scenario-specific instances to train various future MSR models.

Reuse Phase (2/2)

- Assume that the backbone MSR model that needs to be newly trained is MSR^*.
- To obtain an optimal ratio to control the proportion of shared instances, we first sort all scenario instances according to the output probability of the adaptive selection network. Then, we use an automatic grid search to obtain the ratio that makes MSR^* achieve the best results on the validation set.
- After determining the optimal selection ratio, assuming that the current shared instance is represented as $\overline{\mathbf{B}}^{sh}$, we constrain MSR^* to achieve an optimal multi-scenario recommendation performance with the help of these shared instances.

$$\min_{\Theta} \mathcal{L}(\mathcal{D} \setminus \overline{\mathbf{B}}^{sh}) + \mathcal{L}^{sh}(\overline{\mathbf{B}}^{sh}). \tag{16}$$

Datasets

Table: Statistics of the processed datasets.

Dataset		AliCCP				AliMama				
		S1	S2	S3	ALL	S1	S2	S3	S4	ALL
Train	#impress	14,296,532	286,913	23,487,225	38,070,670	964,106	711,366	1,659,630	13,742,020	17,077,122
	#click	571,542	12,600	895,607	1,479,749	54,889	41,050	91,422	749,237	936,598
Validation	#impress	1,588,839	31,960	2,608,436	4,229,235	106,702	79,157	183,839	1,528,296	1,897,994
	#click	63,241	1,323	99,943	164,507	6,040	4,598	10,193	82,700	103,531
Test	#impress	16,351,580	321,024	26,344,010	43,016,614	376,175	267,811	616,552	4,956,251	6,216,789
	#click	656,280	14,099	1,003,068	1,673,447	20,614	14,740	32,579	257,994	325,927

Baselines Methods and Backbone Methods

- To assess the generalization ability of all methods, we combined each method with three mainstream backbone models: DNN, DeepFM [Guo et al., 2017], DCN [Wang et al., 2017], STAR [Sheng et al., 2021], and HMoE [Li et al., 2020],
- We compare our SSIM with the MSR baseline, including three MSR models with representative architectures as strong baselines, i.e., STAR [Sheng et al., 2021], HMoE [Li et al., 2020], and DFFM [Guo et al., 2023].
- We compare our SSIM with the instance utilization baseline to verify the effectiveness of leveraging sharable instances in multi-scenario learning: the first is traditional modeling that directly aggregates all scenario instances (ST-Backbone), the second is fine-tuning each scenario using scenario-specific instances based on ST-Backbone (Finetune), the third is modeling based on original shared-specific architecture (MT-Backbone), and the last one is modeling based on full instance sharing (MT-Backbone-S).

Implementation Details (1/2)

- For general hyperparameters, we set the embedding dimension and batch size as 16 and 4096, respectively. For the MLP layer in the backbone models, we use a three-layer fully connected network of size $[1024, 512, 256]$. We use Adam optimizer and Xavier initialization in the experiments. We select the optimal learning ratio from {1e-3, 3e-4, 1e-4, 3e-5, 1e-5} and the l_2 regularization from {3e-5, 1e-5, 3e-6, 1e-6, 3e-7, 1e-7}.
- For the adaptive selection network in SSIM, we use the three-layer fully connected network mentioned above as the shared feedforward neural network (FNN). The scenario-specific FNN uses a single-layer fully connected network, with its output dimension set to 1. Furthermore, we select the optimal control weight λ from {0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1}.

Implementation Details (2/2)

- For other baseline methods, we refer to the open-source repository of DFFM [Guo et al., 2023], and due to the lack of available open-source repositories for the STAR [Sheng et al., 2021] and HMoE [Li et al., 2020], we re-implement them based on the details provided by the original paper.
- Note that all baselines are tuned for optimal parameter settings within the same range of hyperparameters.
- Following the setup of existing works [Wang et al., 2022, Lin et al., 2022], we apply the two primary evaluation metrics for deep recommender systems: area under the ROC curve (AUC) and cross-entropy (log loss).

RQ1: Performance Comparison (1/2)

	Method	AliCCP						AliMama							
		AUC↑			Logloss↓			AUC↑				Logloss↓			
DNN	ST-Backbone	.6246	.6023	.6220	.1661	.1794	.1600	.6184	.6144	.6245	.6323	.2011	.2023	.1957	.1926
	Finetune	.6250	.6025	.6222	.1658	.1794	.1598	.6197	.6155	.6245	.6328	.2013	.2018	.1965	.1931
	MT-Backbone	.6250	.5910	.6221	**.1652**	.1797	.1604	.6195	.6131	.6232	.6323	.2006	.2019	.1960	**.1919**
	MT-Backbone-S	.6261	.6024	.6234	.1656	**.1789**	.1596	.6203	.6161	.6264	.6337	.2008	.2019	.1953	.1924
	SSIM	**.6275***	**.6031***	**.6247***	.1659	.1795	**.1595**	**.6236***	**.6199***	**.6282***	**.6352***	**.1999***	**.2009***	**.1947***	**.1919**
DeepFM	ST-Backbone	.6248	.6024	.6229	.1662	.1800	.1601	.6202	.6158	.6242	.6303	.2010	.2020	.1957	.1933
	Finetune	.6249	.6027	.6230	.1659	.1796	.1597	.6205	.6160	.6245	.6304	.2006	.2017	.1967	.1981
	MT-Backbone	.6252	.5931	.6227	**.1648**	.1837	.1595	.6202	.6152	.6255	.6323	.2001	.2011	.1946	.1928
	MT-Backbone-S	.6266	.6024	.6237	.1653	.1791	.1594	.6204	.6158	.6258	.6330	.2000	.2010	.1944	**.1919**
	SSIM	**.6279***	**.6031**	**.6254***	.1649	**.1785***	**.1591**	**.6234***	**.6199***	**.6282***	**.6352***	**.1996**	**.2007***	**.1943**	.1921
DCN	ST-Backbone	.6251	.6007	.6223	.1651	.1784	.1591	.6207	.6161	.6245	.6306	.2007	.2017	.1955	.1931
	Finetune	.6257	.6008	.6225	.1650	.1787	**.1590**	.6209	.6169	.6259	.6319	.2003	.2015	.1949	.1935
	MT-Backbone	.6254	.5897	.6228	.1651	.1797	.1599	.6202	.6152	.6254	.6324	.2000	.2011	.1946	.1928
	MT-Backbone-S	.6265	.6020	.6236	.1654	.1788	.1592	.6209	.6169	.6264	.6332	.2004	.2016	.1952	.1923
	SSIM	**.6281***	**.6029***	**.6249***	**.1649**	**.1782**	.1593	**.6239***	**.6203***	**.6283***	**.6351***	**.1994***	**.2003***	**.1942**	**.1921**
MSR	STAR	.6240	.5880	.6191	.1691	.1817	.1604	.6152	.6112	.6216	.6283	.2048	.2096	.1985	.1933
	HMoE	.6220	.5952	.6182	.1665	.1809	.1601	.6164	.6099	.6232	.6279	.2006	.2022	.1949	.1923
	DFFM	.6251	.6022	.6220	**.1657**	**.1795**	.1598	.6216	.6157	.6245	.6319	.2003	.2013	.1947	.1924
	STAR+SSIM	**.6277***	.5975	**.6260***	.1736	.3130	.1608	.6223	.6189	.6272	**.6352***	.2141	.2135	.2024	**.1916**
	HMoE+SSIM	.6262	**.6025**	.6232	**.1657**	.1796	**.1596**	**.6244***	**.6205***	**.6281***	.6350	**.1996***	**.2005***	**.1945**	.1919

RQ1: Performance Comparison (2/2)

- Unlike other instance utilization baseline methods, our SSIM consistently achieves significant performance improvements in most cases. This demonstrates that our SSIM can select a more reasonable set of shared instances.
- We can observe that using our SSIM on the basic backbone model achieves better results in multi-scenario recommendation than the baseline methods that focus on architectural design to extract critical information implicitly.

RQ2: Ablation Study (1/2)

Model	Methods	AliCCP						AliMama							
		AUC↑			Logloss↓			AUC↑				Logloss↓			
DNN	SSIM-R	.6252	.6013	.6223	**.1650**	.1785	**.1590**	.6203	.6155	.6232	.6323	.2000	.2021	.1960	**.1919**
	SSIM-P	.6251	.5822	.6210	.1776	.1801	.1674	.6193	.6135	.6225	.6331	.2363	.2345	.2241	.2166
	SSIM-NK	.6250	.6010	.6224	**.1650**	**.1784**	.1591	.6188	.6147	.6239	.6321	.2005	.2023	.1958	.1923
	SSIM-ND	.6264	.6020	.6229	.1658	.1797	.1601	.6200	.6158	.6238	.6240	.2006	.2012	.1951	.1931
	SSIM	**.6275**	**.6031**	**.6247**	.1659	.1795	.1595	**.6236**	**.6199**	**.6282**	**.6352**	**.1999**	**.2009**	**.1947**	**.1919**
DeepFM	SSIM-R	.6251	.6011	.6228	.1658	.1796	.1597	.6200	.6148	.6234	.6325	.2000	.2017	.1953	.1924
	SSIM-P	.6248	.5924	.6226	.1804	.1901	.1705	.6197	.6136	.6238	.6328	.2388	.2383	.2263	.2199
	SSIM-NK	.6250	.6015	.6226	.1650	.1788	**.1588**	.6197	.6145	.6249	0.6325	.2001	.2016	.1949	**.1919**
	SSIM-ND	.6266	.6020	.6244	.1650	.1789	**.1588**	.6146	.6155	.6241	.6321	.2008	.2040	.1967	.1925
	SSIM	**.6279**	**.6031**	**.6254**	**.1649**	**.1785**	.1591	**.6234**	**.6199**	**.6282**	**.6352**	**.1996**	**.2007**	**.1943**	.1921
DCN	SSIM-R	.6252	.6010	.6222	**.1648**	.1785	**.1588**	.6206	.6148	.6246	.6323	.2000	.2010	.1950	.1923
	SSIM-P	.6208	.5759	.6212	.1800	.1867	.1705	.6189	.6123	.6230	.6307	.2296	.2264	.2212	.2164
	SSIM-NK	.6255	.6006	.6225	.1650	.1785	.1592	.6205	.6158	.6253	.6330	.2000	.2010	.1943	.1924
	SSIM-ND	.6263	.6022	.6236	.1651	.1787	.1589	.6204	.6132	.6256	.6337	.2014	.2037	.1959	.1924
	SSIM	**.6281**	**.6029**	**.6249**	.1649	**.1782**	.1593	**.6239**	**.6203**	**.6283**	**.6351**	**.1994**	**.2003**	**.1942**	**.1921**

We conduct an ablation experiment by replacing it with two sampling methods: random selection (‘SSIM-R’) and positive instance selection (‘SSIM-P’). Then, we evaluate two variants that remove the scenario-specific hyper-network structure from the adaptive selection network (‘SSIM-ND’) and remove the KL constraint in optimization (‘SSIM-NK’).

RQ2: Ablation Study (2/2)

- We can observe that the adaptive selection network incorporated into our framework enhances the performance of the MSR model more effectively than the general instance sampling method. This means that the adaptive selection network adopted can carefully evaluate the impact on MSR learning when each instance becomes a shared instance.
- We can also observe that removing the hyper-network structure and the KL divergence constraint negatively impacts the performance of our framework, as expected, which demonstrates that all the critical steps in our SSIM are necessary.

RQ3: Reusability Analysis (1/2)

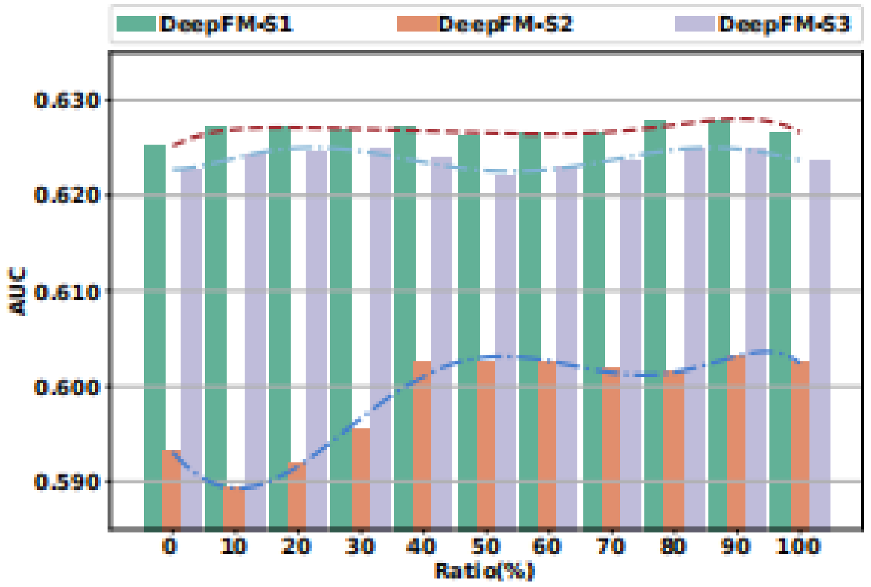
Target	Source	AliCCP						AliMama							
		AUC↑			Logloss↓			AUC↑				Logloss↓			
DNN	DNN	.6275	**.6031**	.6247	.1659	.1795	.1595	**.6236**	**.6199**	**.6282**	**.6352**	**.1999**	**.2009**	**.1947**	**.1919**
	DeepFM	**.6276**	.6029	**.6252**	**.1653**	**.1787**	**.1591**	.6234	.6198	.6281	**.6352**	.1999	.2010	.1947	.1920
	DCN	.6272	.6027	.6243	**.1653**	**.1787**	.1593	.6237	**.6200**	.6282	.6352	**.1999**	**.2009**	.1947	.1920
DeepFM	DeepFM	.6279	**.6031**	.6254	**.1649**	.1785	**.1591**	**.6234**	**.6199**	**.6282**	**.6352**	**.1996**	.2007	**.1943**	**.1921**
	DNN	**.6281**	.6023	.6253	.1650	.1787	.1597	.6229	.6196	.6277	**.6352**	.1997	**.2006**	.1944	.1922
	DCN	.6280	.6028	**.6255**	**.1649**	**.1784**	.1593	**.6234**	.6196	.6280	.6353	**.1996**	.2008	.1943	**.1921**
DCN	DCN	**.6281**	.6029	.6249	**.1649**	**.1782**	.1593	**.6239**	**.6203**	.6283	.6351	**.1994**	**.2003**	**.1942**	1921
	DNN	.6275	.6028	**.6252**	.1651	.1785	.1594	.6239	.6202	.6284	.6352	**.1994**	**.2003**	**.1942**	.1921
	DeepFM	**.6281**	**.6031**	.6251	.1650	.1785	**.1592**	.6236	.6198	**.6284**	**.6353**	.1998	.2008	.1946	**.1919**
STAR	DNN	**.6276**	**.6022**	.6256	**.1705**	**.2963**	.1608	**.6237**	.6195	.6281	**.6353**	.2138	.2127	.2020	.1916
	DeepFM	.6273	.6013	**.6259**	.1722	.3284	**.1601**	.6226	.6195	.6278	.6352	**.2109**	**.2105**	**.2007**	.1916
	DCN	.6270	.6002	**.6259**	.1718	.3193	.1603	.6231	**.6201**	**.6283**	.6349	.2160	.2147	.2034	.1916
HMoE	DNN	**.6262**	**.6032**	**.6235**	.1656	.1790	.1594	**.6244**	.6202	.6279	.6350	**.1996**	.2006	.1946	**.1919**
	DeepFM	.6255	.6025	.6230	**.1654**	.1790	**.1592**	.6241	.6202	.6279	**.6351**	.1997	.2006	.1946	.1920
	DCN	**.6262**	**.6032**	.6234	**.1654**	**.1789**	.1593	**.6244**	**.6204**	.6279	.6349	**.1996**	.2006	**.1945**	**.1919**

To verify the reusable properties of our SSIM, we first use SSIM with different backbone models to obtain shared instance sets, respectively, and then directly use the obtained shared sets and scenario-specific instance sets to train MSR models with different backbones.

RQ3: Reusability Analysis (2/2)

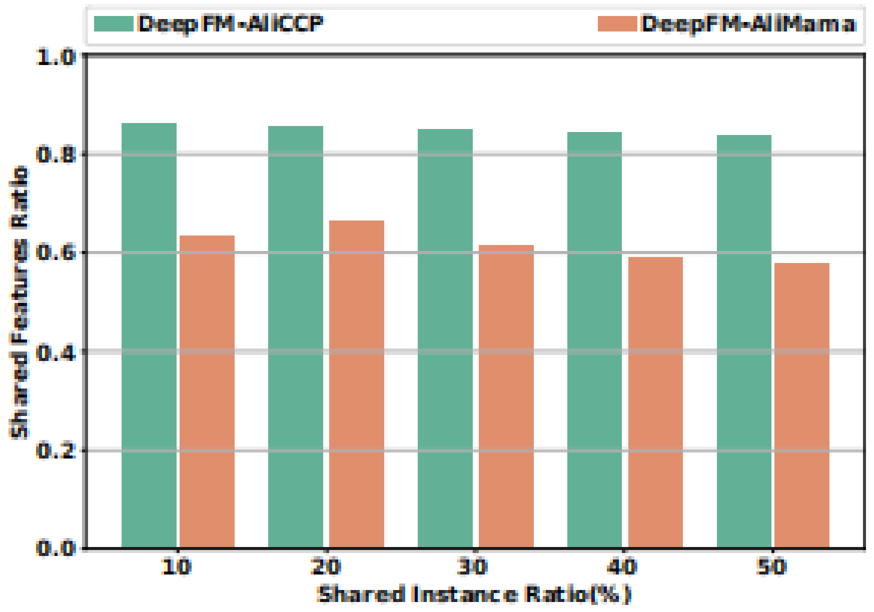
- Good reusability allows a selected set of shared instances to be saved and used directly to train various MSR models, which also prevents these MSR models from repeatedly performing the search phase.
- We can find that no matter the source backbone, the shared subset obtained by our SSIM can maintain good performance in MSR models of different backbones. This verifies the effectiveness of our SSIM in identifying shared instances with high reusability and makes it attractive for deployment in industrial MSR.

RQ4: In-depth Analysis (1/4)



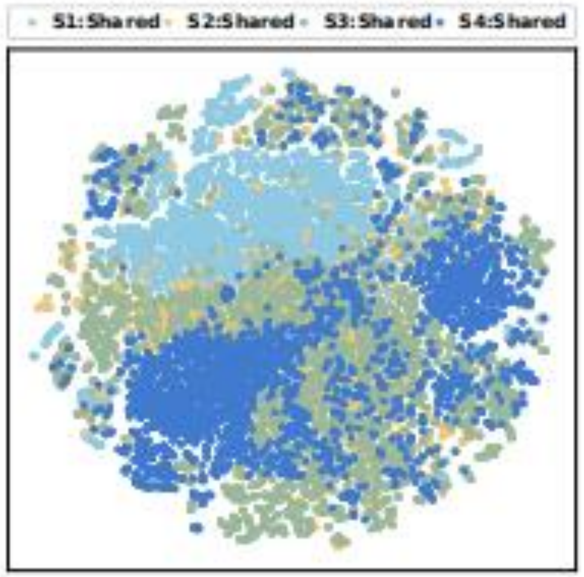
- We conduct an analysis experiment on AliCCP to better understand the impact of introducing shared instances in different scenarios.
- We control the ratio of shared instances obtained by our SSIM and set it to $\{0\%, 10\%, 20\%, \cdots, 100\%\}$.
- We can find that as the ratio of shared instances increases, the performance of niche scenarios (S2) will be significantly improved in the early stage and then maintained at a similar level.

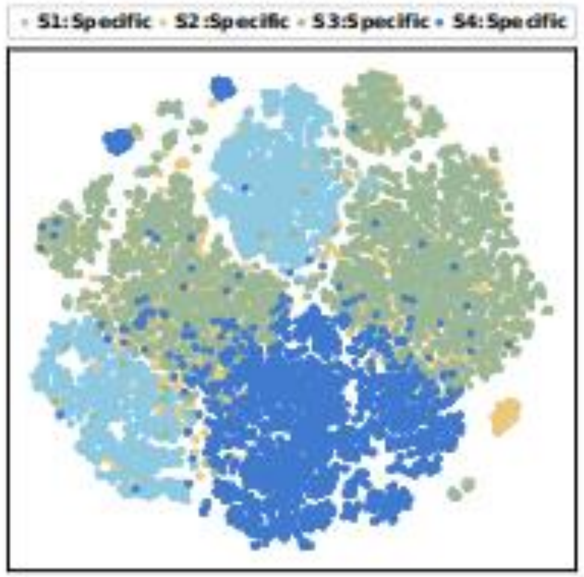
RQ4: In-depth Analysis (2/4)



- Intuitively, instances with the same (or similar) user or the same (or similar) item are more likely to carry some scenario consensus information, i.e., have more shared features. We conducted an analysis experiment to verify this guess.
- We can observe that no matter the value of the shared instance ratio, the shared feature ratio corresponding to the shared instance set is high (e.g., 80% for AliCCP and 60% for AliMama).

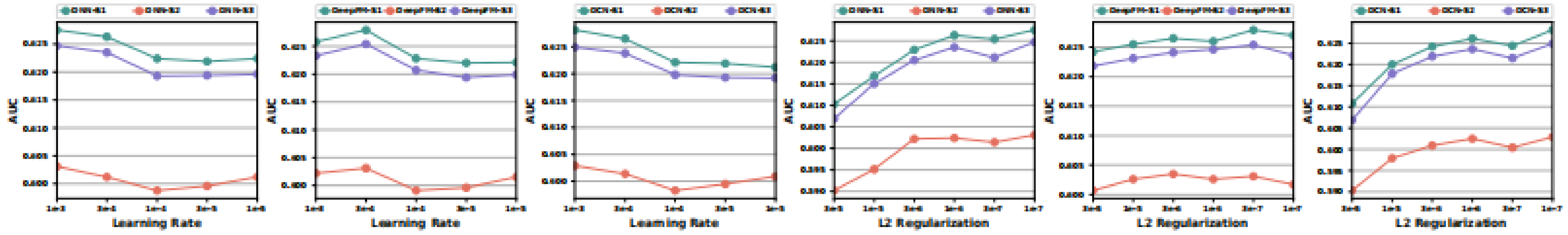
RQ4: In-depth Analysis (3/4)





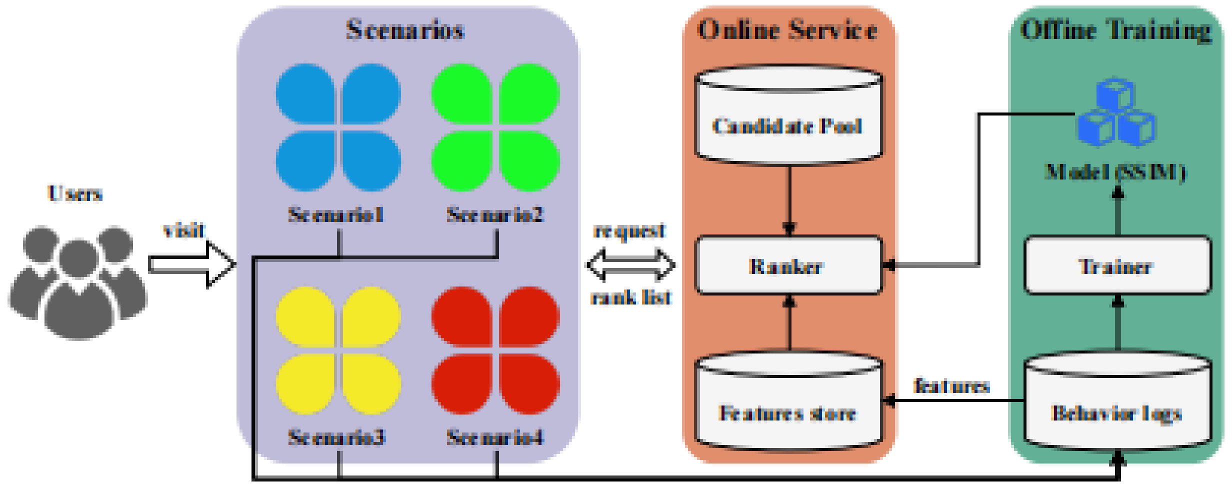
- Taking Alimama as an example, after our SSIM is well-trained, we can get the embeddings of scenario-shared and scenario-specific instances, respectively, and use t-SNE for dimension reduction visualization. Where, with the results of scenario-shared and scenario-specific instances on the left and right.
- As expected, we find that the representations of scenario-specific instances have a more discriminative state than shared instances.

RQ4: In-depth Analysis (4/4)



- We conduct experiments on AliCCP using three backbone models to study the sensitivity of two key hyperparameters: (i) learning rate and (ii) l_2 regularization coefficient.
- Based on the results in the figure, we can find that the performance change trends of our SSIM for the three scenarios are similar under different hyperparameter values.

RQ5: Online Experiments



Metrics	Scenario1	Scenario2	Scenario3	Scenario4
CTR	3.15%	2.08%	0.93%	0.92%
SA	3.07%	1.92%	7.71%	14.81%

- We deploy our SSIM in the MSR scenario of Tencent FiT, one of the large-scale online financial platforms in China, to further evaluate its performance. The specific deployment process is shown in the figure.
- Then, we adopted two core metrics commonly used in the platform (i.e., click-through rate (CTR) and subscription amount (SA)) as evaluation indicators.
- As shown in the table, our SSIM consistently achieves significant gains compared to the deployed baseline model in all scenarios.

Conclusions

- We propose an explicit multi-scenario shared instance modeling (SSIM) framework based on hyper-network learning to improve MSR performance by adaptively selecting useful shareable subsets from training instances.
- Our SSIM optimizes an adaptive selection network with a hyper-network structure to generate sharable subsets from all scenario instances in the search phase. These shareable subsets can be used directly to train future multi-scenario recommendation models in the reuse phase.
- Finally, we evaluate SSIM and demonstrate its effectiveness through experiments on two public multi-scenario benchmarks and an online A/B test.

Thank You!

- We thank the anonymous reviewers for their expert and constructive comments and suggestions.
- We thank the support of the National Natural Science Foundation of China (NSFC) Nos. 62302310 and 62272315.

- Codes and slides are available at `https://github.com/dgliu/KDD25_SSIM`
- If you have any questions, feel free to contact us.

Courbariaux, M., Hubara, I., Soudry, D., El-Yaniv, R., and Bengio, Y. (2016).
Binarized neural networks: Training deep neural networks with weights and activations constrained to +1 or -1.
arXiv preprint arXiv:1602.02830.

Guo, H., Tang, R., Ye, Y., Li, Z., and He, X. (2017).
Deepfm: A factorization-machine based neural network for ctr prediction.
In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 1725–1731.

Guo, W., Zhu, C., Yan, F., Chen, B., Liu, W., Guo, H., Zheng, H., Liu, Y., and Tang, R. (2023).
Dffm: Domain facilitated feature modeling for ctr prediction.
In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 4602–4608.

Li, P., Li, R., Da, Q., Zeng, A.-X., and Zhang, L. (2020).
Improving multi-scenario learning to rank in e-commerce by exploiting task relationships in the label space.
In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*, pages 2605–2612.

Lin, W., Zhao, X., Wang, Y., Xu, T., and Wu, X. (2022).
Adafs: Adaptive feature selection in deep recommender system.
In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3309–3317.

Ma, J., Zhao, Z., Yi, X., Chen, J., Hong, L., and Chi, E. H. (2018).
Modeling task relationships in multi-task learning with multi-gate mixture-of-experts.
In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1930–1939.

Sheng, X.-R., Zhao, L., Zhou, G., Ding, X., Dai, B., Luo, Q., Yang, S., Lv, J., Zhang, C., Deng, H., et al. (2021).
One model to serve all: Star topology adaptive recommender for multi-domain ctr prediction.
In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management*, pages 4104–4113.

Wang, R., Fu, B., Fu, G., and Wang, M. (2017).
Deep & cross network for ad click predictions.
In *Proceedings of the ADKDD 2017*, pages 1–7.

Wang, Y., Zhao, X., Xu, T., and Wu, X. (2022).
Autofield: Automating feature selection in deep recommender systems.
In *Proceedings of the ACM Web Conference 2022*, pages 1977–1986.