Recommender System: Basic and Why AutoML is Needed?

Yong Li
(Tsinghua University)

Outline

1. **What is recommender system**

2. **Recent advances in recommender system**
   a) **Deep Learning**
   b) **Graph Neural Networks**

3. **Problem of human-crafted recommender system and why AutoML is needed**

Outline

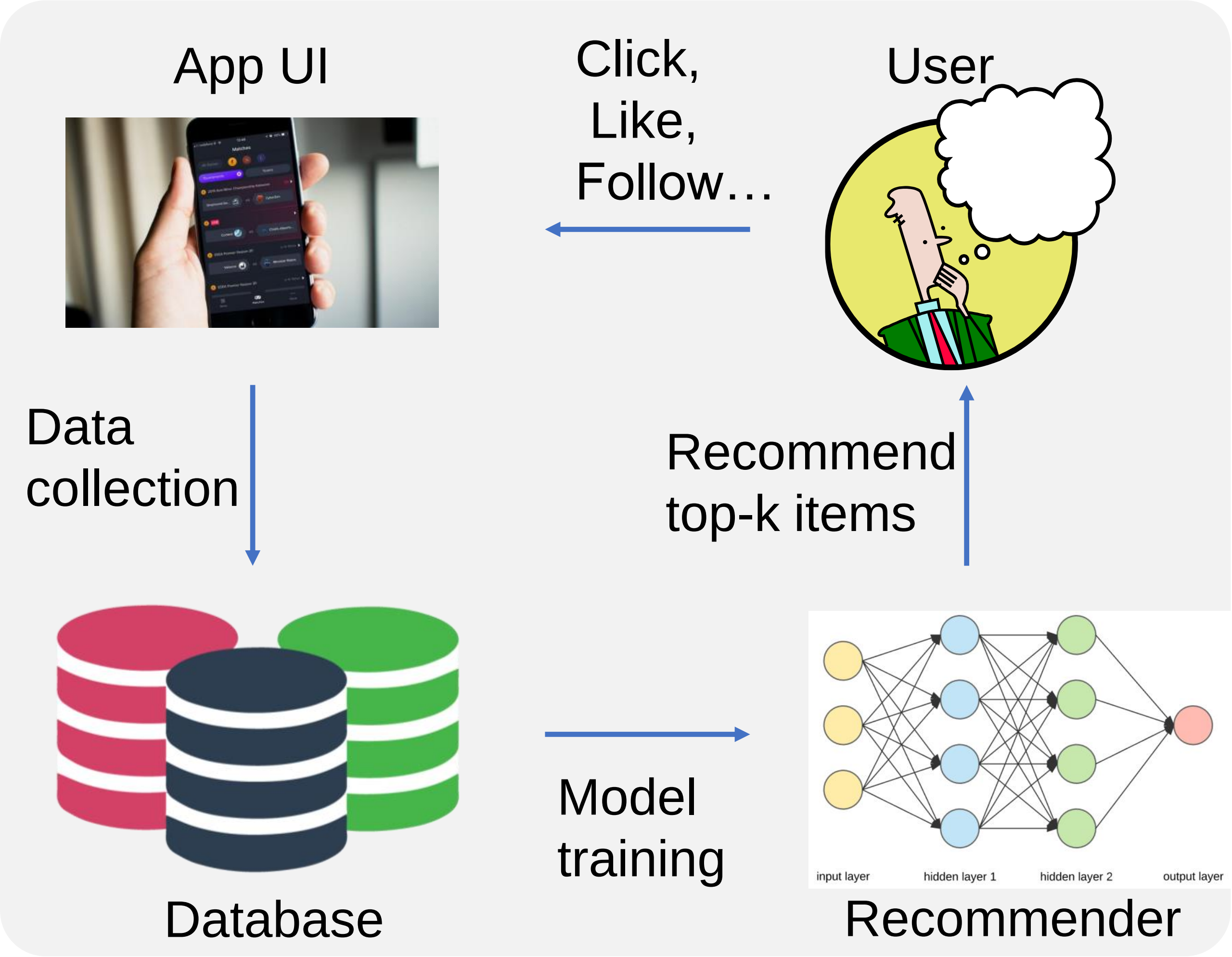
1. **What is recommender system**

2. **Recent advances in recommender system**
   a) **Deep Learning**
   b) **Graph Neural Networks**

3. **Problem of human-crafted recommender system and why AutoML is needed**

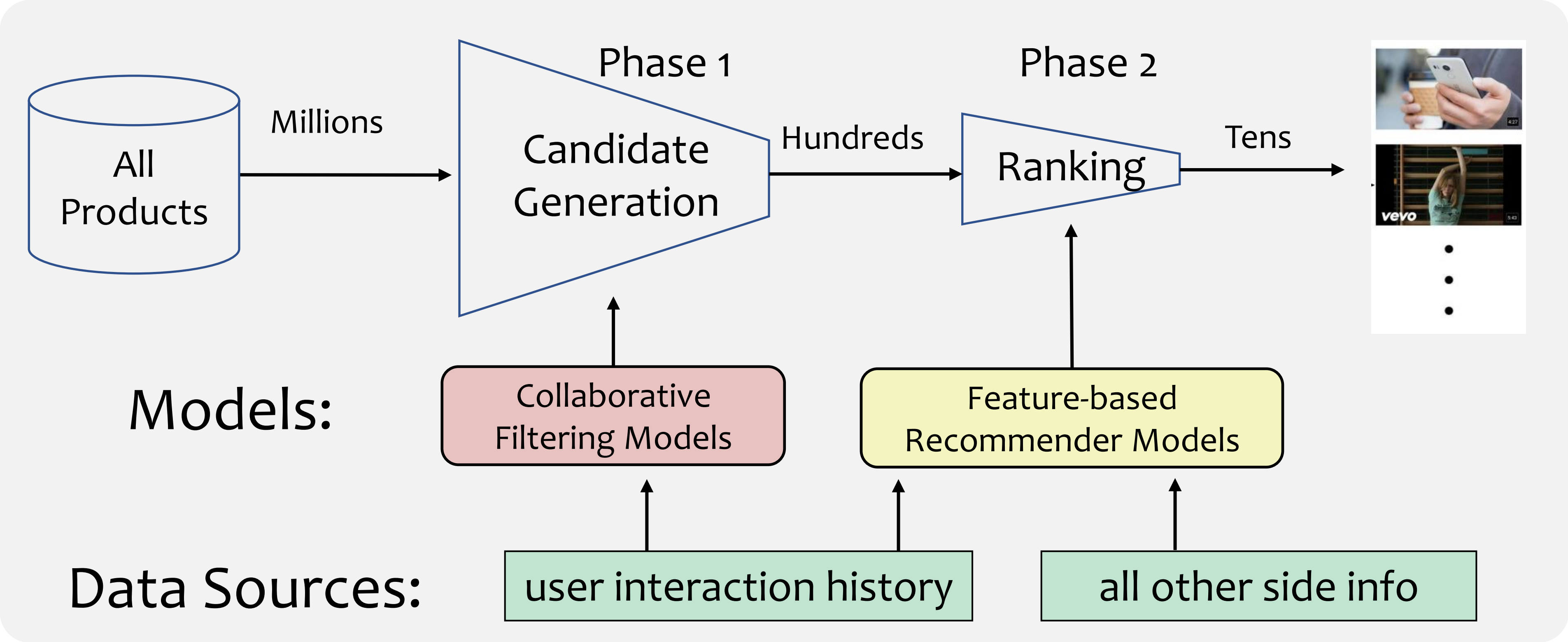
Recommender System

Modelling users' preference towards items based on historical behaviours, such as click, like, follow, etc.



Problem Formulation

- **Input:** historical user-item interactions or additional side information (e.g., user profile, item profile)
- **Output**: given a target Item (e.g., movie, song, product), how likely a user would interact with it (e.g., click, view, or purchase)



Outline

1. **What is recommender system**

2. **Recent advances in recommender system**
   a) **Deep Learning**
   b) **Graph Neural Networks**

3. **Problem of human-crafted recommender system and why AutoML is needed**

Deep Learning for Recommendation

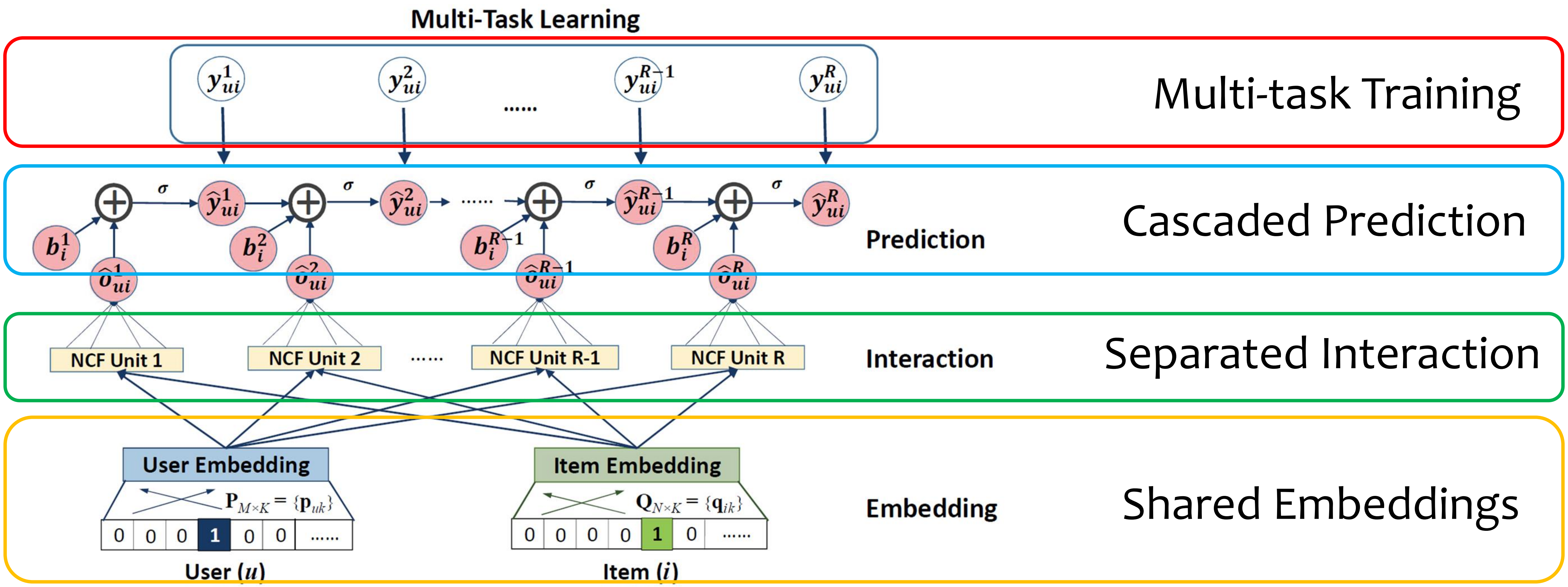
- Deep Learning is utilized to substitute nearly all components in recommender system.
  - Feature extraction
  - Representation learning
  - Matching function learning

- Deep Learning shows great power in modeling high-order similarity in recommender system, e.g. multi-behavior and social relationship in recommendation…

Neural Multi-Task Recommendation from Multi-Behavior Data (Gao et al, ICDE'19)

Target behavior **+** ***Auxiliary behaviors***

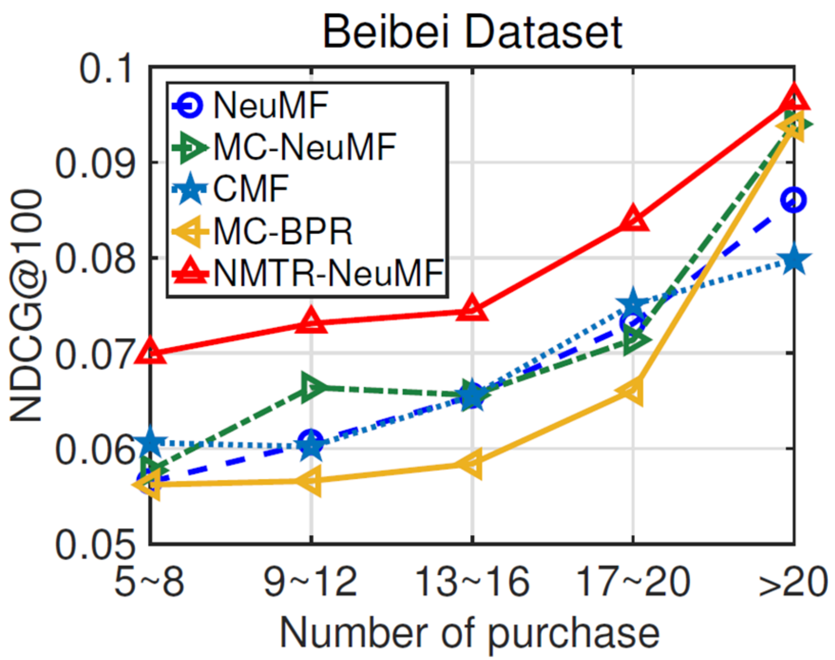
***E-commerce*:**



Neural Multi-Task Recommendation from Multi-Behavior Data (Gao et al, ICDE'19)

		Beibei Dataset			
Group	**Method**	**HR@50**	**NDCG@50**	**HR@100**	**NDCG@100**
Our NMTR Model	NMTR-GMF	0.2050	0.0590	0.3119	0.0741
	NMTR-MLP	0.1928	0.0560	0.2690	**0.0762**
	NMTR-NeuMF	**0.2079**	**0.0609**	**0.3193**	0.0760
Multi-behavior	CMF	0.1596	0.0481	0.2829	0.0663
	MC-BPR	0.1743	0.0503	0.2659	0.0647
	MC-GMF	0.1822	0.0508	0.2975	0.0690
	MC-MLP	0.1810	0.0534	0.2810	0.0684
	MC-NeuMF	0.2014	0.0577	0.3010	0.0719
Single-behavior	BPR	0.1199	0.0348	0.2002	0.0463
	GMF	0.1792	0.0475	0.2920	0.0665
	MLP	0.1711	0.0483	0.2679	0.0617
	NeuMF	0.1828	0.0573	0.2929	0.0714

NMTR achieves the best overall performance.



NMTR achieves the best performance under different sparsity.

Outline

1. **What is recommender system**

2. **Recent advances in recommender system**
   a) **Deep Learning**
   b) **Graph Neural Networks**

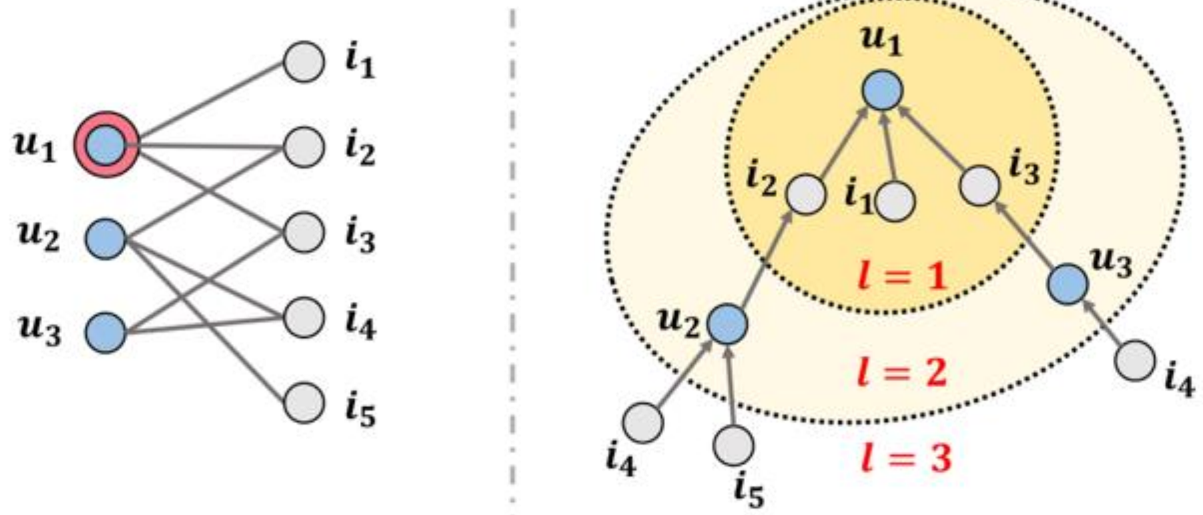
3. **Problem of human-crafted recommender system and why AutoML is needed**

Collaborative Filtering (CF)

- Revisit CF via **high-order connectivity**
  - The paths that reach u_1 from any node with the path length l larger than 1
  - A natural way to encode collaborative signal is to encode it by the **interaction graph structure**

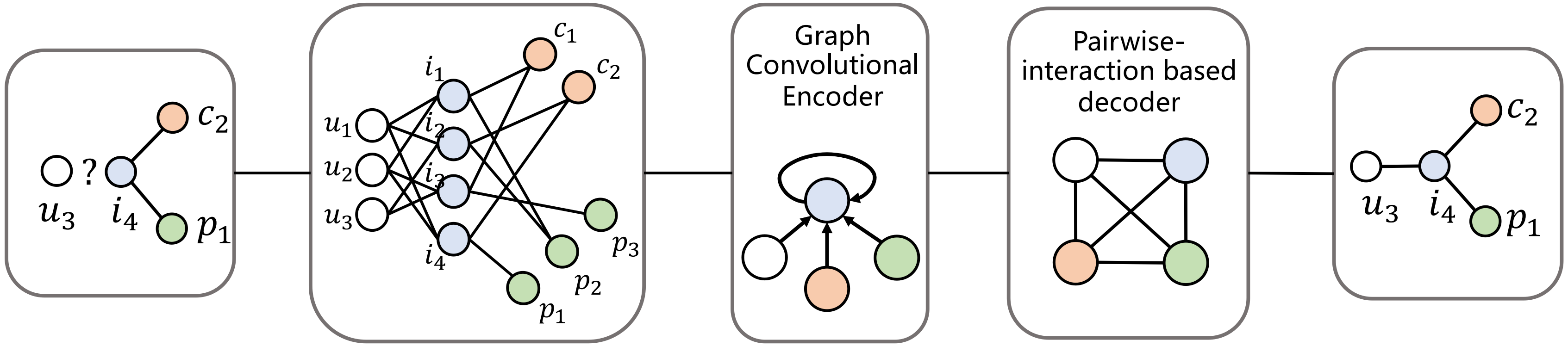
Why u_1 may like i_4

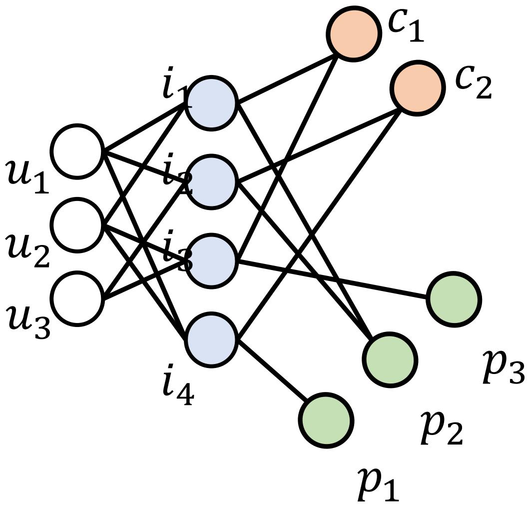
- $u_1 \leftarrow i_2 \leftarrow u_2 \leftarrow i_4$
- $u_1 \leftarrow i_3 \leftarrow u_3 \leftarrow i_4$

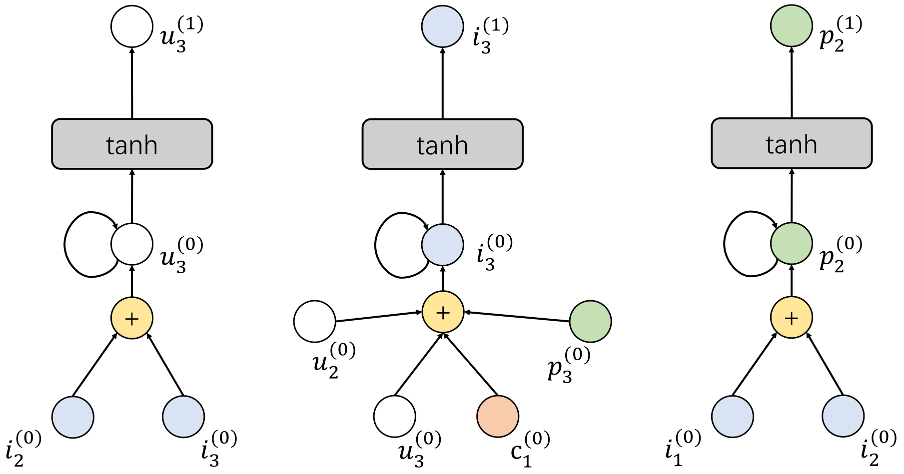


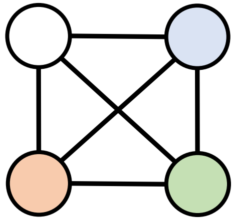
User-Item Interaction Graph | High-order Connectivity for u_1

Price-aware Recommendation with Graph Convolutional Networks (Zheng et al, ICDE2020)









$$s_g = e_{ug}^T e_{ig} + e_{ug}^T e_{pg} + e_{ig}^T e_{pg}$$

$$s_c = e_{uc}^T e_{pc} + e_{uc}^T e_{cc} + e_{cc}^T e_{pc}$$

$$s = s_g + \alpha s_c$$

Unified graph of user, item, price, and category

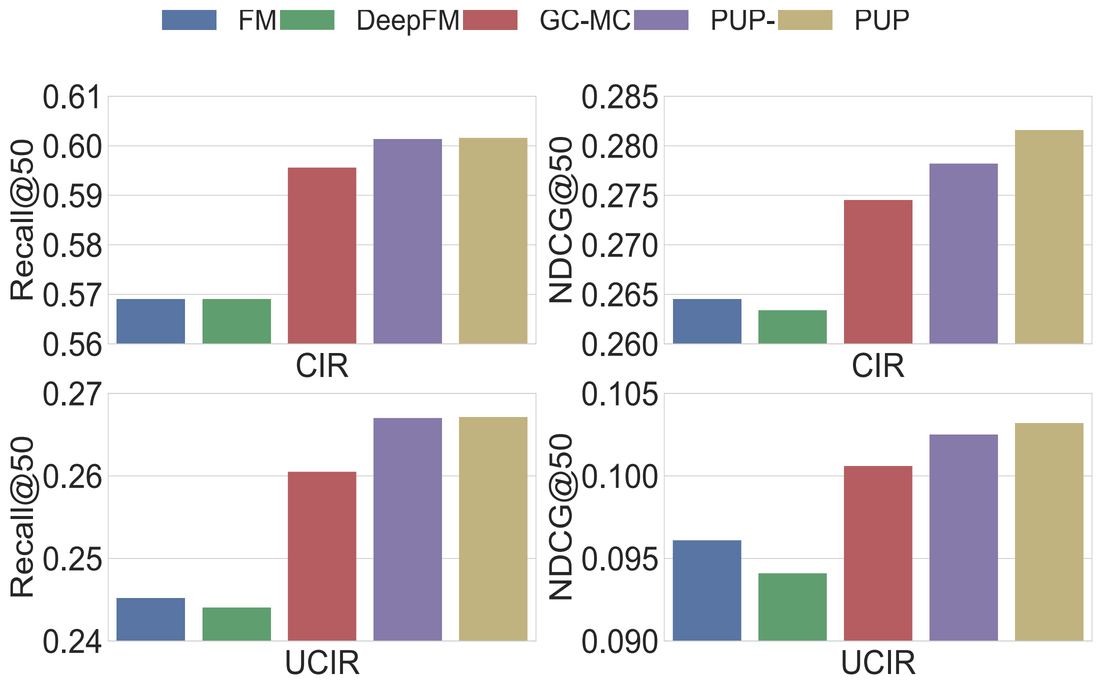
Graph convolutional encoder to learn robust representations for different entities

Pair-wise decoder to learn both global and local price awareness.

Price-aware Recommendation with Graph Convolutional Networks (Zheng et al, ICDE2020)

TABLE II
TOP-K RECOMMENDATION PERFORMANCE COMPARISON ON THE YELP AND BEIBEI DATASETS (K IS SET TO 50 AND 100)

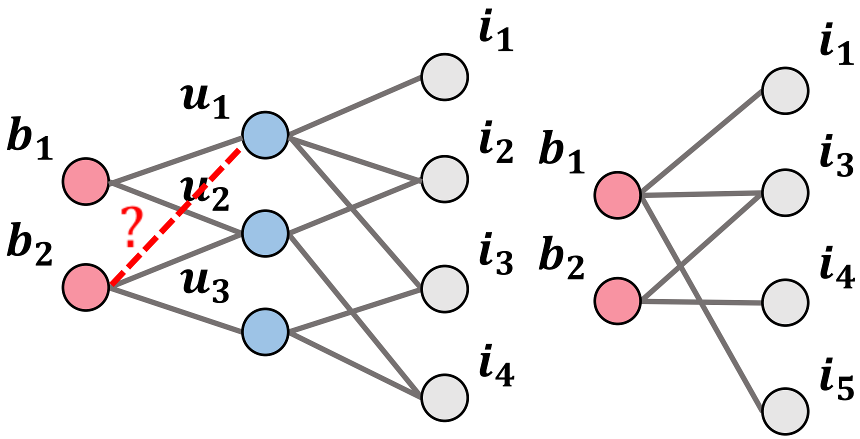
	Yelp dataset				Beibei dataset			
method	Recall@50	NDCG@50	Recall@100	NDCG@100	Recall@50	NDCG@50	Recall@100	NDCG@100
ItemPop	0.0401	0.0182	0.0660	0.0247	0.0087	0.0027	0.0175	0.0046
BPR-MF	0.1621	0.0767	0.2538	0.1000	0.0256	0.0103	0.0379	0.0129
PaDQ	0.1241	0.0572	0.2000	0.0767	0.0131	0.0056	0.0186	0.0068
FM	0.1635	**0.0771**	0.2538	0.1001	**0.0259**	0.0104	0.0384	0.0130
DeepFM	0.1644	0.0769	0.2545	0.0998	0.0255	0.0090	**0.0400**	0.0122
GC-MC	0.1670	0.0770	**0.2621**	**0.1011**	0.0231	0.0100	0.0343	0.0124
NGCF	**0.1679**	0.0769	0.2619	0.1008	0.0256	**0.0107**	0.0383	**0.0134**
PUP	**0.1765**	**0.0816**	**0.2715**	**0.1058**	**0.0266**	**0.0113**	**0.0403**	**0.0142**
impr.%	5.12%	5.84%	3.59%	4.65%	2.70%	5.61%	0.75%	5.97%

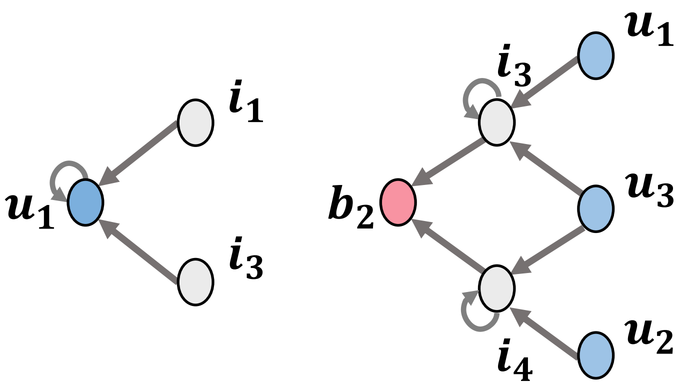


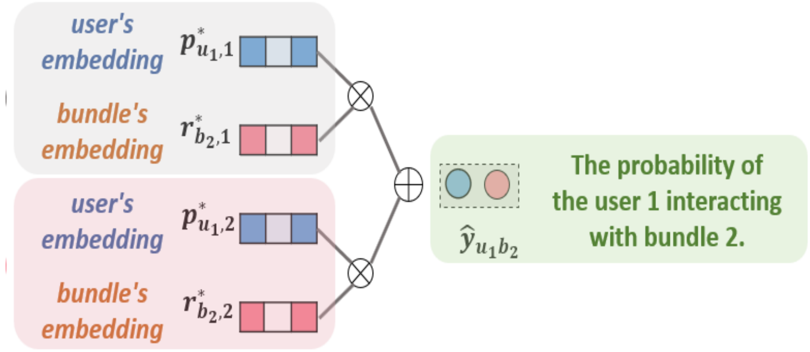
PUP successfully captures users' price awareness and achieves best performance compared to strong baselines.

PUP tackles cold-start problem with the help of price awareness modeling.

Bundle Recommendation with Graph Convolutional Networks (Chang et al, SIGIR2020)







Construct heterogeneous graph of user, item and bundle

Item-level and bundle-level propagation

Item-level and bundle-level prediciton

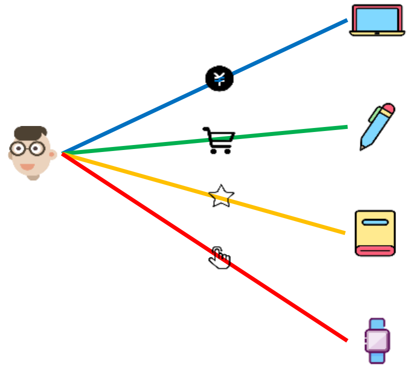
Bundle Recommendation with Graph Convolutional Networks (Chang et al, SIGIR2020)

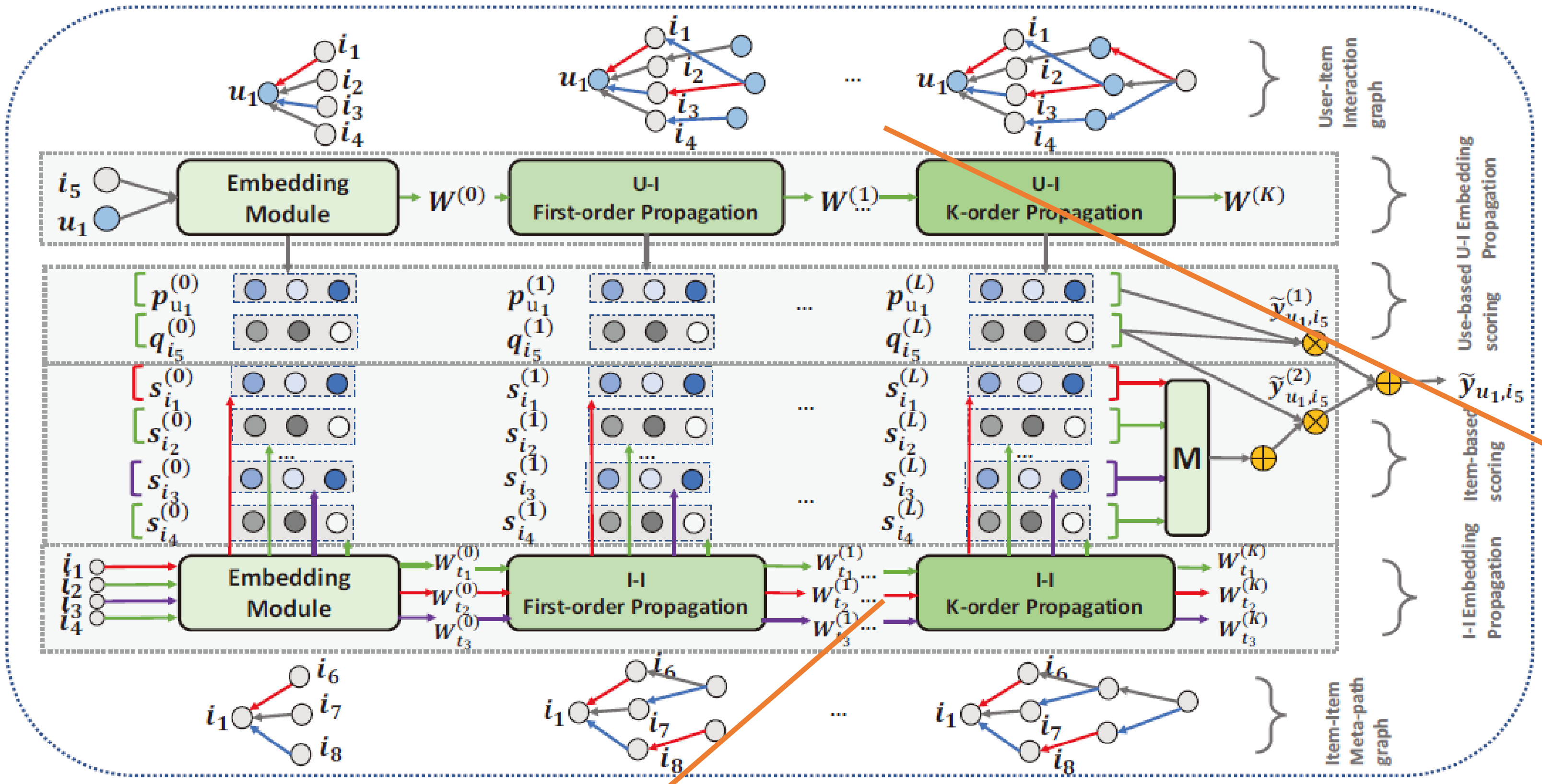
Table 2: Performance comparisons on two real-world datasets with six baselines

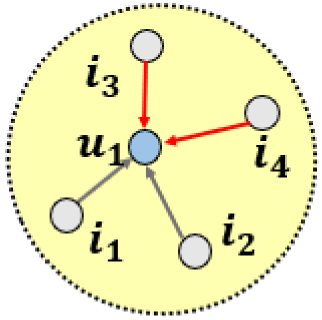
Method	Netease						Youshu					
	Recall@20	**NDCG@20**	**Recall@40**	**NDCG@40**	**Recall@80**	**NDCG@80**	**Recall@20**	**NDCG@20**	**Recall@40**	**NDCG@40**	**Recall@80**	**NDCG@80**
MF-BPR	0.0355	0.0181	0.0600	0.0246	0.0948	0.0323	0.1959	0.1117	0.2735	0.1320	0.3710	0.1543
GCN-BG	0.0370	0.0189	0.0617	0.0255	0.1000	0.0342	0.1982	0.1141	0.2661	0.1322	0.3633	0.1541
GCN-TG	0.0402	0.0204	0.0657	0.0272	0.1051	0.0362	0.2032	0.1175	0.2770	0.1371	0.3804	0.1605
NGCF-BG	0.0395	0.0207	0.0646	0.0274	0.1021	0.0359	0.1985	0.1143	0.2658	0.1324	0.3542	0.1524
NGCF-TG	0.0384	0.0198	0.0636	0.0266	0.1015	0.0350	0.2119	0.1165	0.2761	0.1343	0.3743	0.1561
DAM	0.0411	0.0210	0.0690	0.0281	0.1090	0.0372	0.2082	0.1198	0.2890	0.1418	0.3915	0.1658
BGCN	**0.0491**	**0.0258**	**0.0829**	**0.0346**	**0.1304**	**0.0453**	**0.2347**	**0.1345**	**0.3248**	**0.1593**	**0.4355**	**0.1851**
% Improv.	19.67%	22.89%	20.17%	23.18%	19.65%	21.76%	10.77%	12.22%	12.36%	12.33%	11.23%	11.62%

BGCN achieves the best performance.

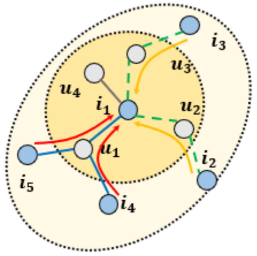
Multi-behavior Recommendation with Graph Convolutional Networks (Jin&Gao et al, SIGIR2020)







$$\alpha_{ut} = \frac{w_t \cdot n_{ut}}{\sum_{m \in N_r} w_m \cdot n_{um}}$$



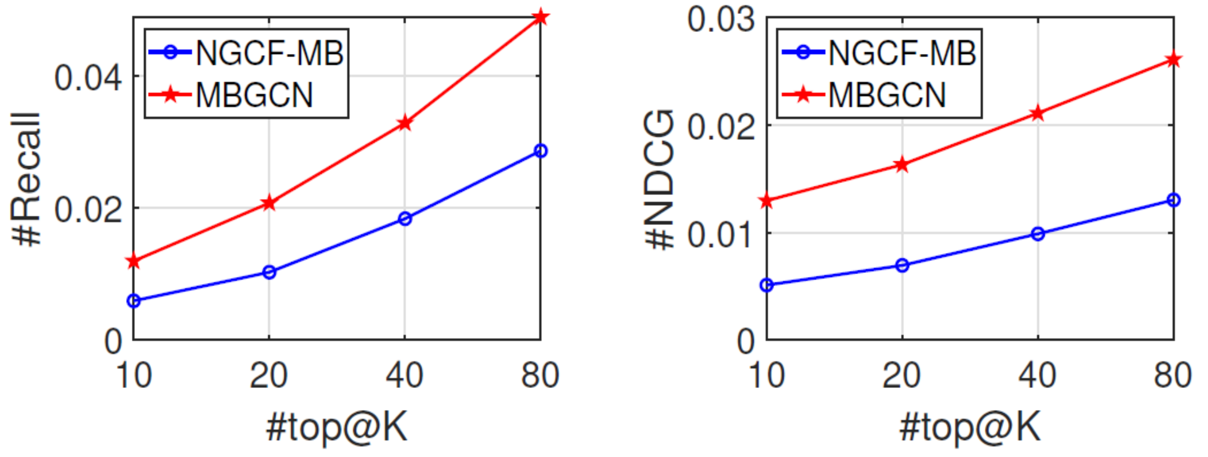
Item-item propagation to capture different behavior semantics.

User-item propagation to capture different behavior strength.

Multi-behavior Recommendation with Graph Convolutional Networks (Jin&Gao et al, SIGIR2020)

Table 2: Comparisons on Tmall and improvement comparing with the best baseline.

	Method	Recall@10	NDCG@10	Recall@20	NDCG@20	Recall@40	NDCG@40	Recall@80	NDCG@80
One-behavior	MF-BPR	0.02331	0.01306	0.03161	0.01521	0.04239	0.01744	0.05977	0.02049
	NCF	0.02507	0.01472	0.03319	0.01683	0.04502	0.01931	0.06352	0.02252
	GraphSAGE-OB	0.01993	0.01157	0.02521	0.01296	0.03368	0.01474	0.04617	0.01693
	NGCF-OB	0.02608	0.01549	0.03409	0.01757	0.04612	0.02010	0.06415	0.02324
Multi-behavior	MCBPR	0.02299	0.01344	0.03178	0.01558	0.04360	0.01813	0.06190	0.02132
	NMTR	0.02732	0.01445	0.04130	0.01831	0.06391	0.02279	0.09920	0.02891
	GraphSAGE-MB	0.02094	0.01223	0.02805	0.01406	0.03804	0.01616	0.05351	0.01887
	NGCF-MB	*0.03076*	*0.01754*	*0.04196*	*0.02042*	0.05857	*0.02389*	0.08408	0.02833
	RGCN	0.01814	0.00955	0.02627	0.01165	0.03877	0.01426	0.05749	0.01750
	MBGCN	**0.04006**	**0.02088**	**0.05797**	**0.02548**	**0.08348**	**0.03079**	**0.12091**	**0.03730**
	Improvement	30.23%	19.04%	37.04%	24.78%	24.91%	28.88%	8.90%	26.40%

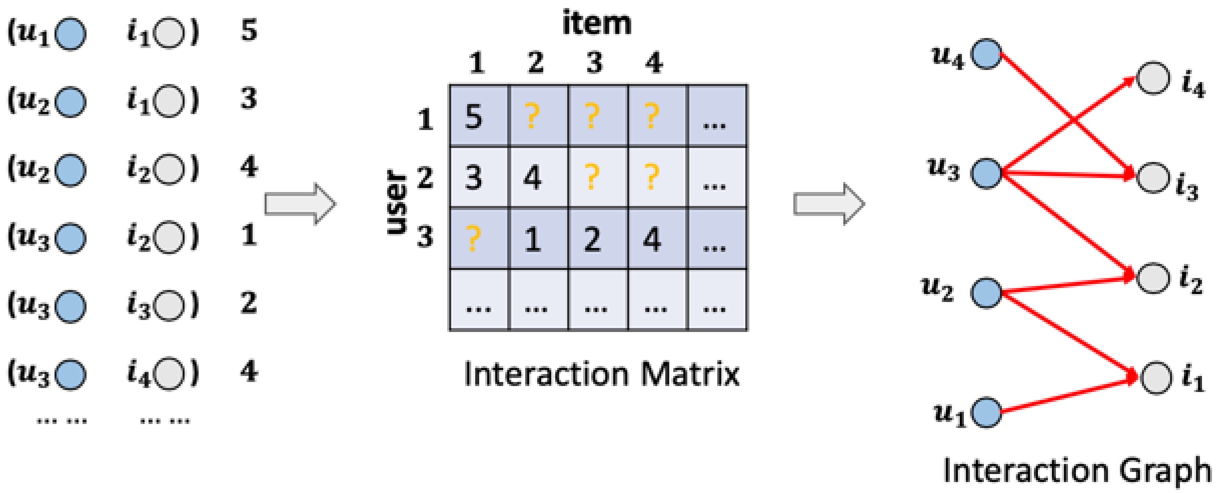


(a) Recall (b) NDCG

MBGCN performs the best against state-of-the-art algorithms.

Summary: GNN for CF

- Encode high-order connectivity via GNN → collaborative signals
- Enhance representation learning of users and items → combine user-item bipartite with other features (such as price, bundle, behavioral type, etc.)



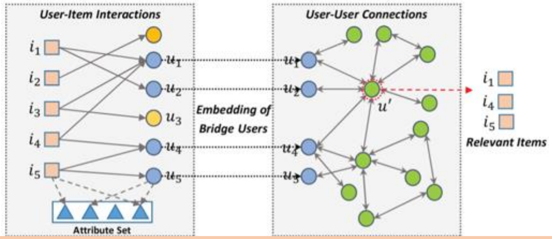
Outline

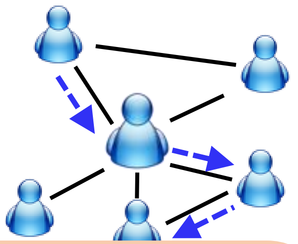
1. **What is recommender system**

2. **Recent advances in recommender system**
   a) **Deep Learning**
   b) **Graph Neural Networks**

3. **Problem of human-crafted recommender system and why AutoML is needed**

Evolution of recommendation tasks





Social recommendation

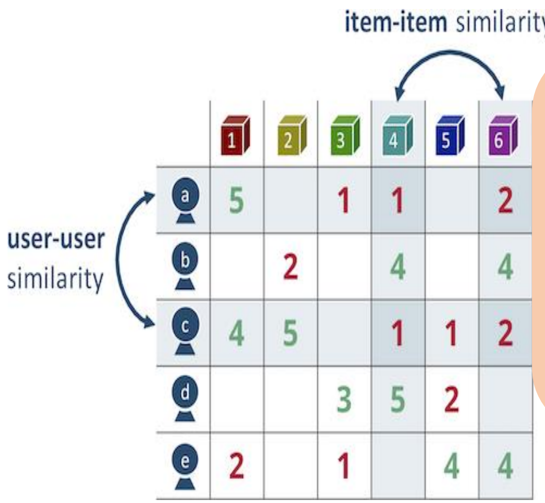
Recommendation tasks are getting more diverse!

Sequential recommendation

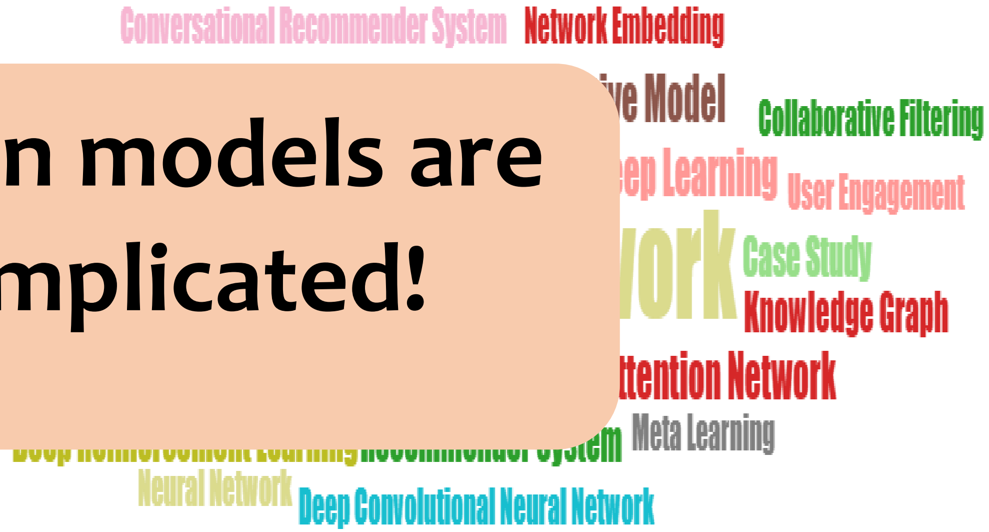
Product rating prediction

Bundle recommendation

Evolution of recommendation models



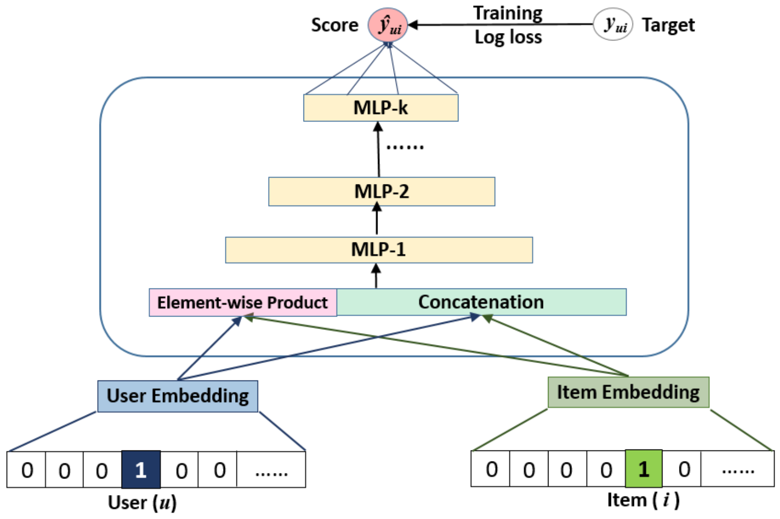
Recommendation models are getting more complicated!



Simple KNN

DNN, GNN, KG, Attention,…

Human-crafted recommender system



Too many decisions to be made!

1. **Input Features**

   feature selection, feature crossing, …

2. **Model Architecture**

   #layers, #blocks, activation functions,…

3. **Interaction Function**

   inner product, minus/plus, min/max, MLP, …

4. **Hyper-parameters**

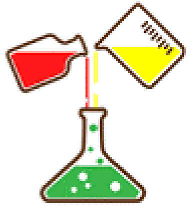
   embedding size, regularization, learning rate,…

Human-crafted recommender system

How to make decisions ?

1. **Designed manually by experts**

2. **Enumerated by experiments**



Both are suboptimal and inefficient! Why?

Human-crafted recommender system

1. **Designed manually by experts**
   1. **Introduce noise?**

      E.g. unnecessary features
   2. **Miss counter-intuitive design?**

      E.g. novel model architecture
   3. **High labor cost**

2. **Enumerated by experiments**
   1. **Large memory and computation cost**

Human-crafted recommender system

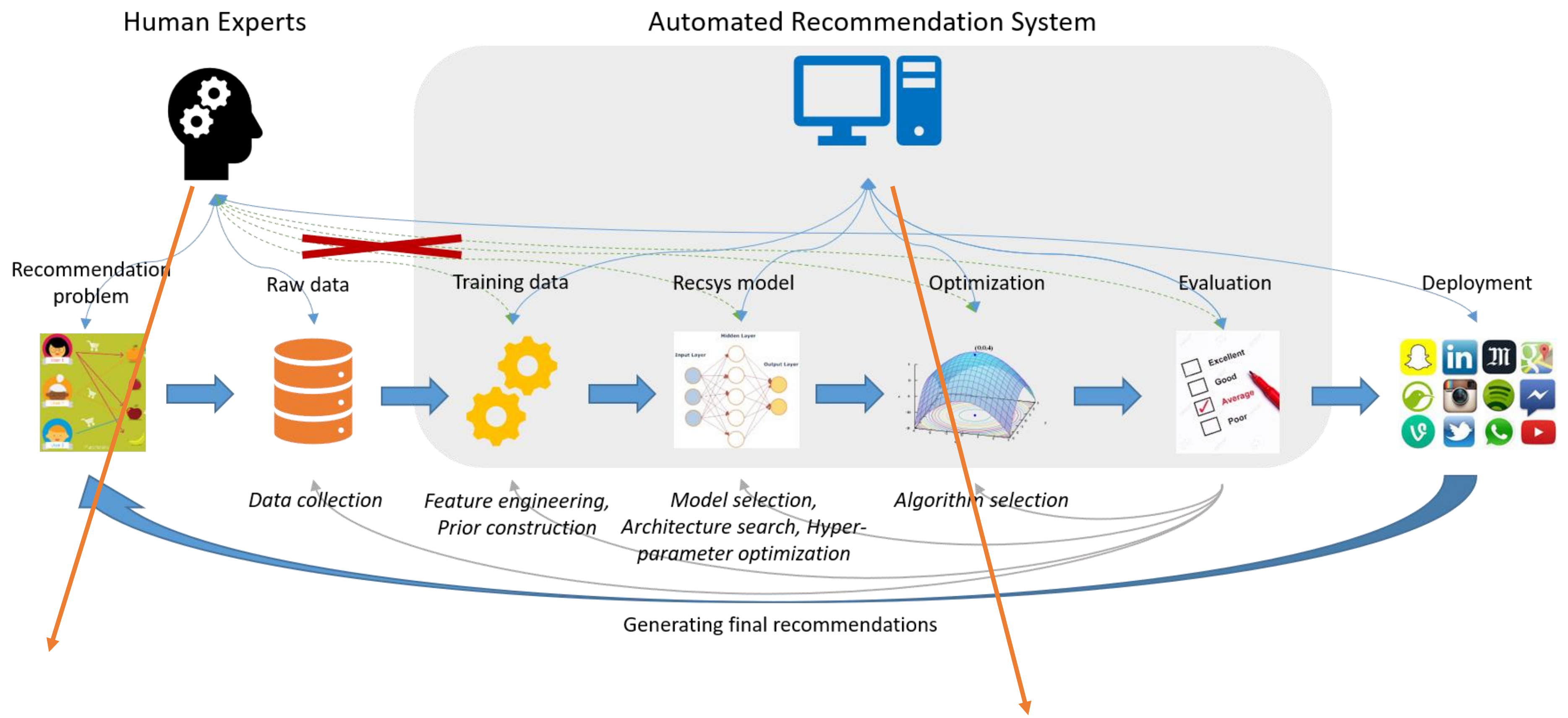
- **Most importantly, there is no silver bullet that is universally optimal.**

- **Performance of different choices on feature/model architecture/interaction function/hyper-parameter depends on datasets and tasks.**

How to always make good decisions on different datasets and tasks?

AutoML!

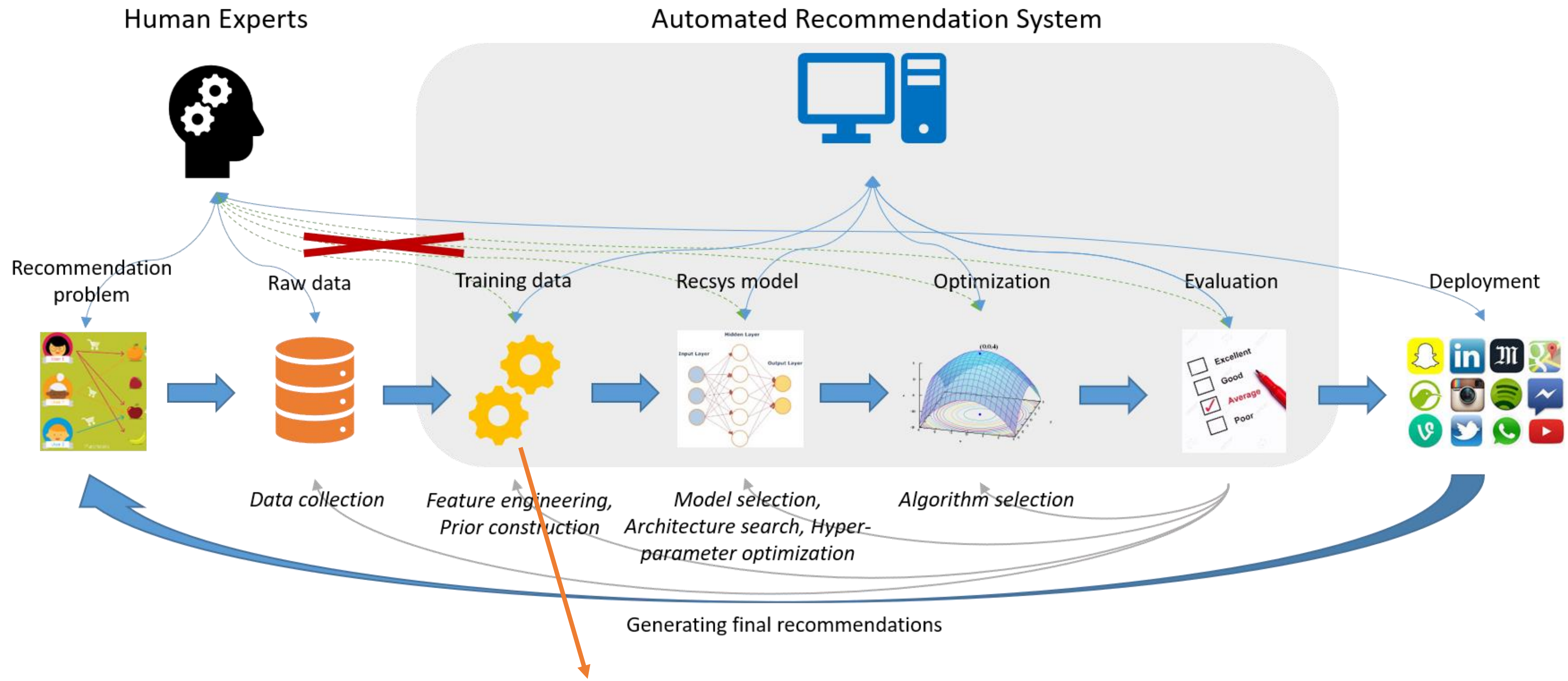
AutoML for Recommendation



Massive human efforts

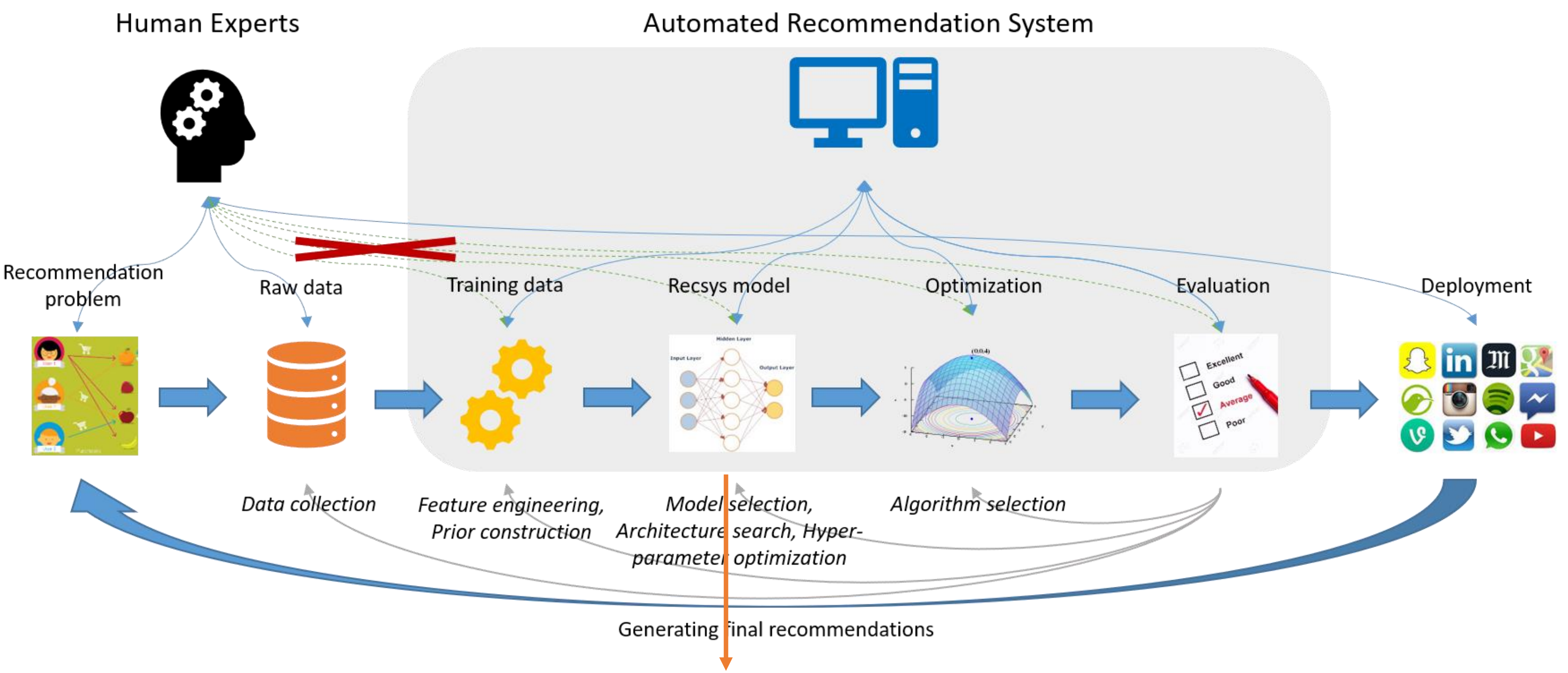
Automatically build recommendation models

What to be automated ?



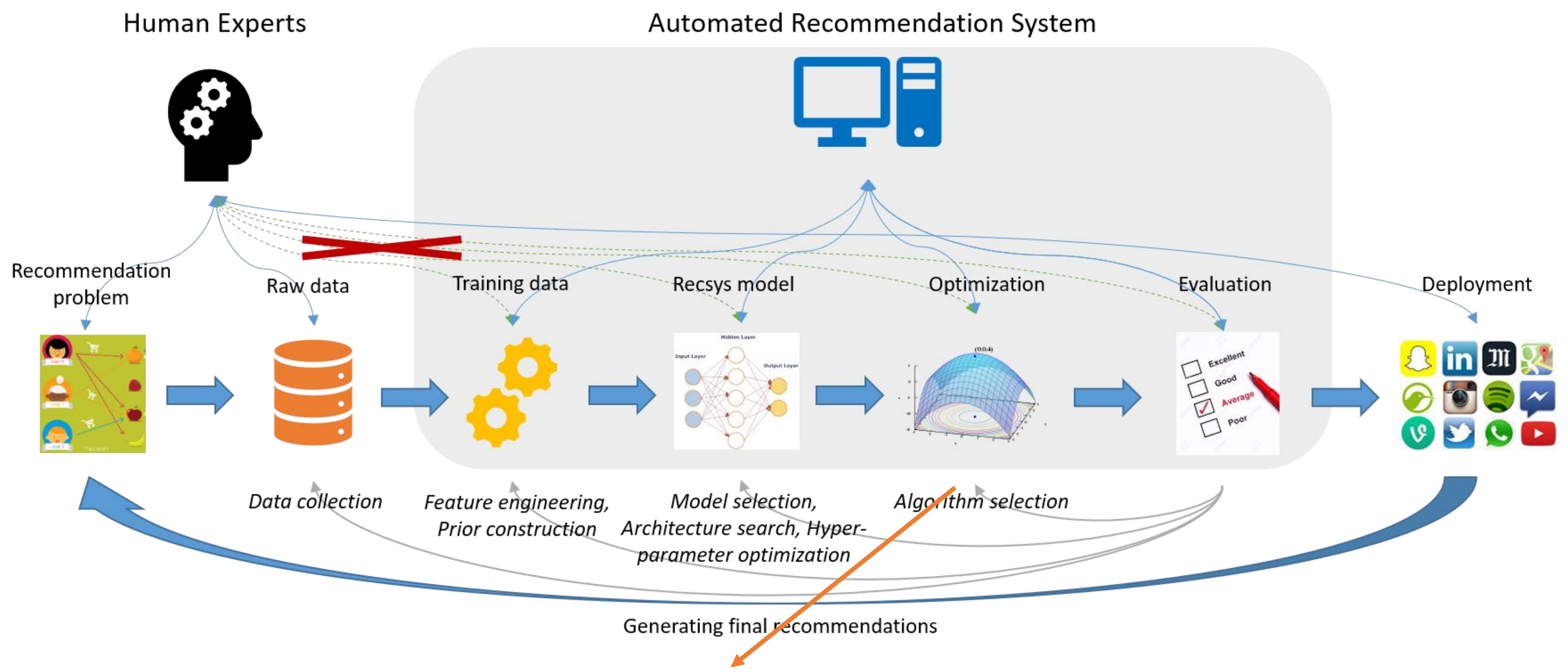
Feature Engineering is a tedious and task-specific work. AutoML helps to generate informative and discriminative features

What to be automated ?



Model Design/Hyperparameter Tuning requires heavy human and computation cost. AutoML helps to identify better architectures than handcrafted ones.

What to be automated ?



Algorithm Selection. Optimization algorithms greatly influence model performance. AutoML helps to set the right configurations for the optimization algorithm.

Recent Advances in Automated RecSys

- Yao et al., Efficient Neural Interaction Functions Search for Collaborative Filtering. **WWW 2020.**
- Chen et al., lambdaOpt: Learn to Regularize Recommender Models in Finer Levels. **KDD 2019.**
- Luo et al., AutoCross: Automatic Feature Crossing for Tabular Data in Real-World Applications. **KDD 2019.**

Will be introduced in detail by next tutor.

Summary

- **Advanced techniques are incorporated into recommender systems, such as deep learning, graph neural networks and knowledge graph. Better performance is achieved.**

- **Human-crafted recommender system requires heavy manual designs or computation cost on multiple components, including feature engineering, model architecture design and algorithm selection.**

- **AutoML help automatically make reasonable decisions on different datasets and tasks.**

Thank You!

liyong07@Tsinghua.edu.cn