Recent Advances in Automated Recommender System

Chen Gao

Ph.D Candidate, Tsinghua University

gc16@mails.tsinghua.edu.cn

KDD2020

Tsinghua University

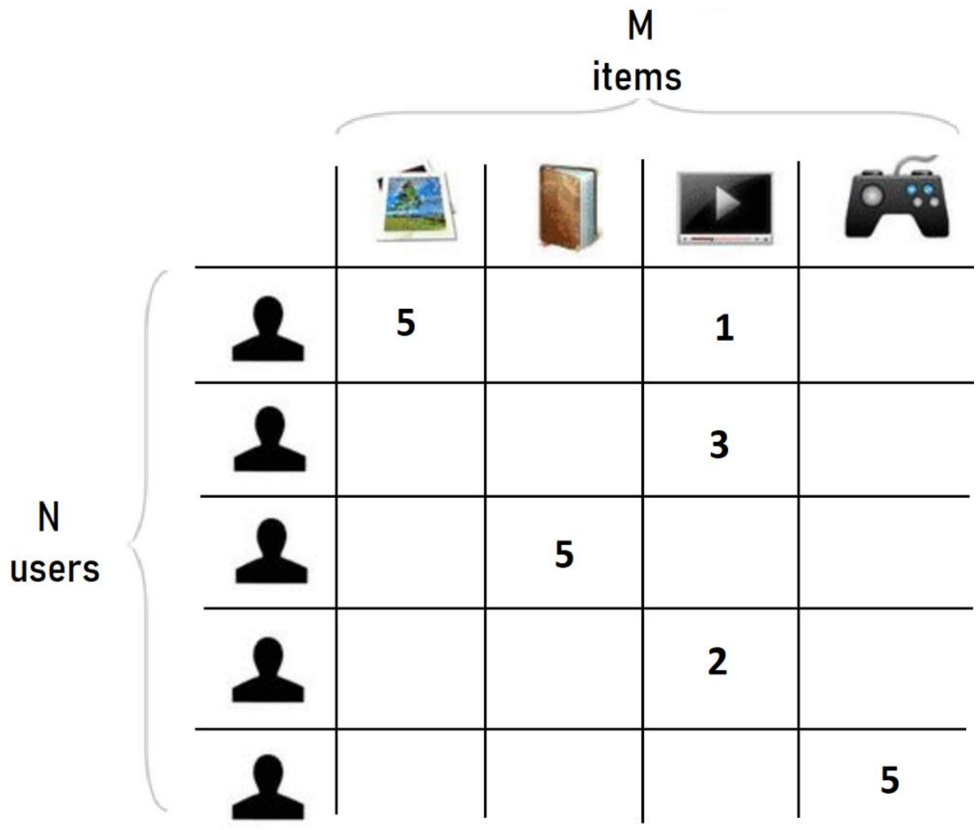
Outline

- AutoML for Collaborative Filtering Task
- AutoML for Click-through Rate Prediction Task
- AutoML for Tuning Hyper-parameters in RecSys

Outline

- AutoML for Collaborative Filtering Task
- AutoML for Click-through Rate Prediction Task
- AutoML for Tuning Hyper-parameters in RecSys

Collaborative Filtering – Problem Setup



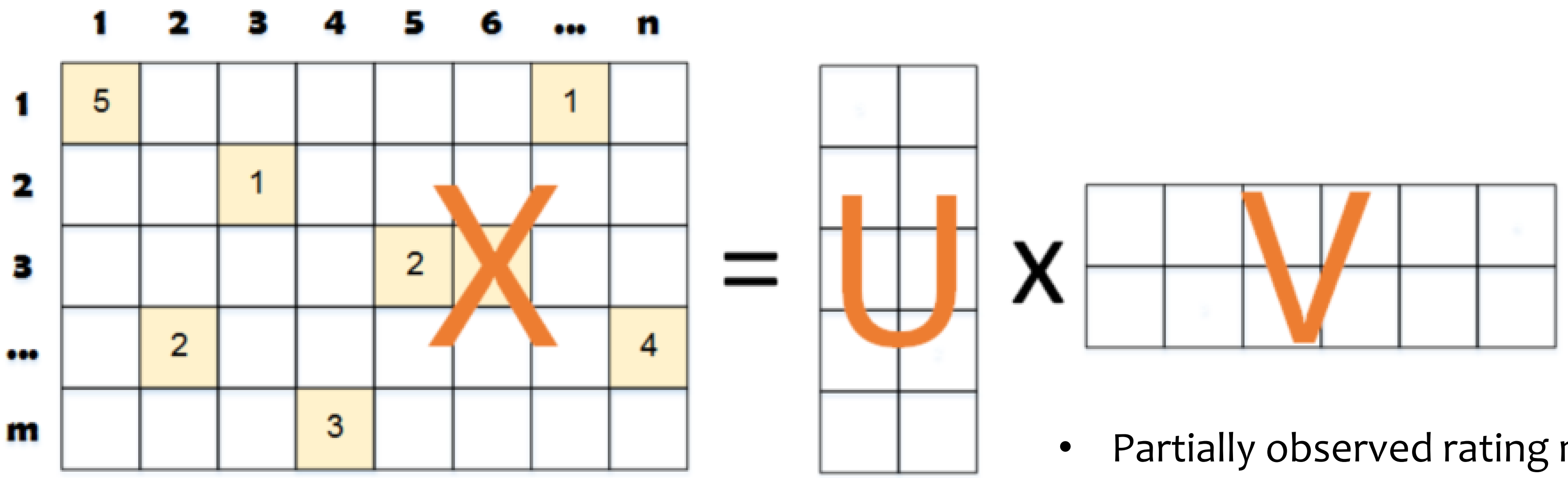
Study the fundamental CF problem [1]

- Data: a rating matrix with many unknown positions
- Task: estimate ratings on unknown positions
- Measurement: RMSE on estimated ratings.

[1]. Exact Matrix Completion via Convex Optimization. Foundations of Computational Mathematics. 2008

Image is from: https://mc.ai/deep-learning-for-collaborative-filtering-using-fastai/

Collaborative Filtering – Low-rank approach [1,2]



$$\min_{U,V} \sum_{\underset{\text{Observed Positions}}{(i,j)\in\Omega}} \ell\left(\overset{\text{Prediction}}{u_i^\top v_j}, \underset{\text{Rating}}{O_{ij}}\right) + \underbrace{\frac{\lambda}{2}\|U\|_F^2 + \frac{\lambda}{2}\|V\|_F^2}_{\text{Regularization}},$$

- Partially observed rating matrix O can be well-approximated by a low-rank matrix X
- Matrix U: user embedding, matrix V: item embedding
- Rating prediction is given by an inner product of user embedding and item embedding

Interaction Function

[1]. Factorization meets the neighborhood: a multifaceted collaborative filtering model. KDD 2008
[2]. Exact Matrix Completion via Convex Optimization. Foundations of Computational Mathematics. 2008

Collaborative Filtering – More Example IFCs

	IFC	operation	space	predict time	recent examples
human-designed	$\langle \boldsymbol{u}_i, \boldsymbol{v}_j \rangle$	inner product	$O((m+n)k)$	$O(k)$	MF [28], FM [37]
	$\boldsymbol{u}_i - \boldsymbol{v}_j$	plus (minus)	$O((m+n)k)$	$O(k)$	CML [19]
	$\max\left(\boldsymbol{u}_i, \boldsymbol{v}_j\right)$	max, min	$O((m+n)k)$	$O(k)$	ConvMF [25]
	$\sigma\left([\boldsymbol{u}_i; \boldsymbol{v}_j]\right)$	concat	$O((m+n)k)$	$O(k)$	Deep&Wide [9]
	$\sigma\left(\boldsymbol{u}_i \odot \boldsymbol{v}_j + \boldsymbol{H}\left[\boldsymbol{u}_i; \boldsymbol{v}_j\right]\right)$	multi, concat	$O((m+n)k)$	$O(k^2)$	NCF [17]
	$\boldsymbol{u}_i * \boldsymbol{v}_j$	conv	$O((m+n)k)$	$O(k \log(k))$	ConvMF [25]
	$\boldsymbol{u}_i \otimes \boldsymbol{v}_j$	outer product	$O((m+n)k)$	$O(k^2)$	ConvNCF [16]

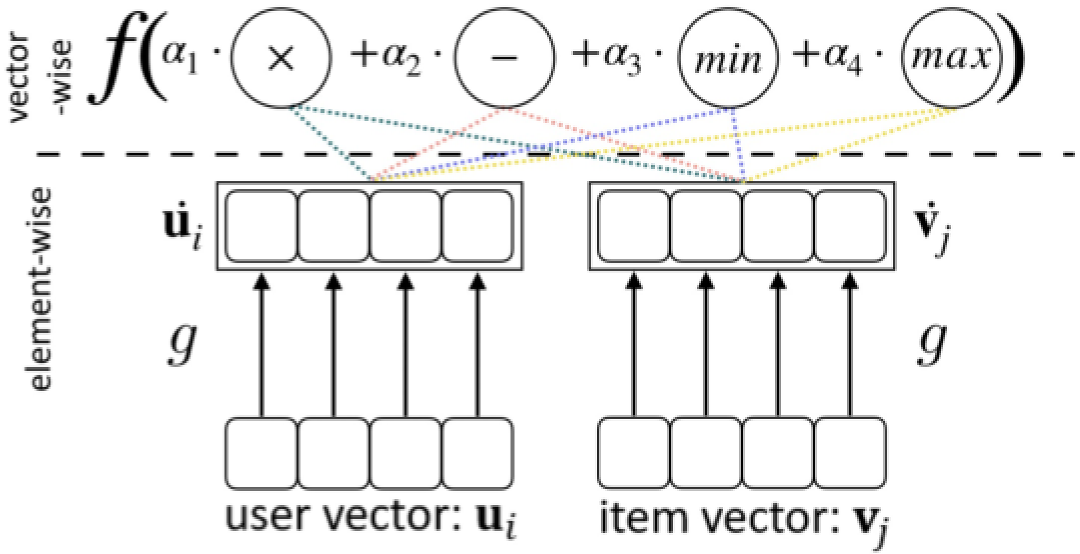
Is there an absolute best IFC? : NO, depends on tasks and datasets [1]

[1]. Are we really making much progress? A worrying analysis of recent neural recommendation approaches. Recsys 2019

SIF (Yao et al, WWW2020)

IFC	operation
$\langle \boldsymbol{u}_i, \boldsymbol{v}_j \rangle$	inner product
$\boldsymbol{u}_i - \boldsymbol{v}_j$	plus (minus)
$\max\left(\boldsymbol{u}_i, \boldsymbol{v}_j\right)$	max, min
$\sigma\left([\boldsymbol{u}_i; \boldsymbol{v}_j]\right)$	concat
$\sigma\left(\boldsymbol{u}_i \odot \boldsymbol{v}_j + \boldsymbol{H}\left[\boldsymbol{u}_i; \boldsymbol{v}_j\right]\right)$	multi, concat
$\boldsymbol{u}_i * \boldsymbol{v}_j$	conv
$\boldsymbol{u}_i \otimes \boldsymbol{v}_j$	outer product

Cut the search space into two blocks



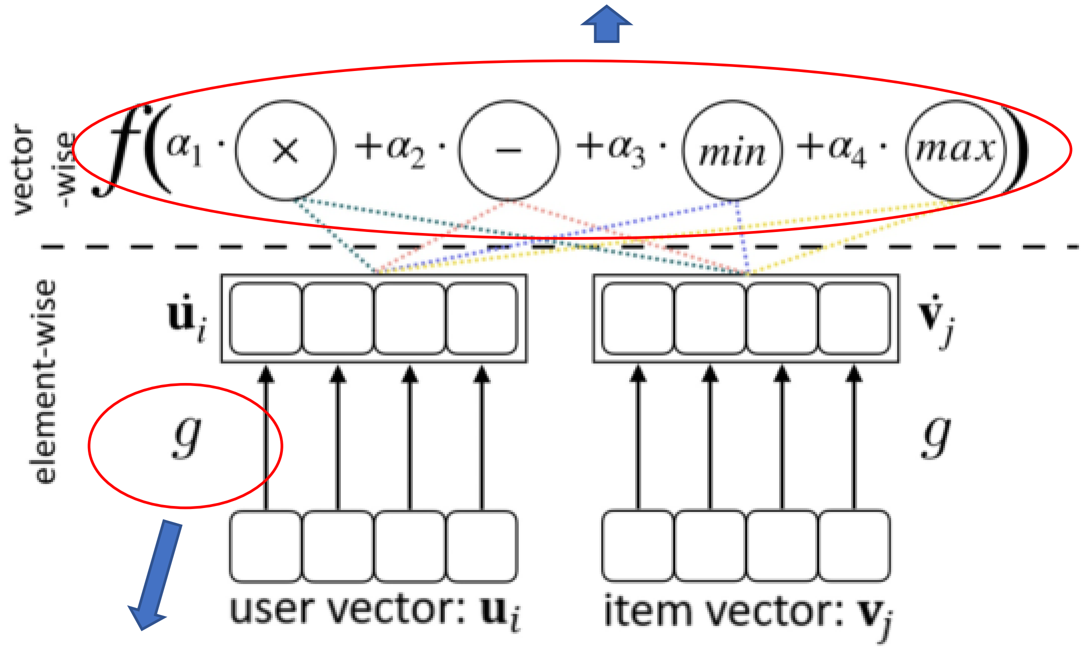
- Vector-level: simple linear algebra operations
- Elementwise: shared nonlinear transformation

Learning from Existing IFCs !

Q. Yao, X. Chen, J. Kwok, Y. Li, C.-J. Hsieh. Efficient Neural Interaction Functions Search for Collaborative Filtering, WWW 2020.

SIF (Yao et al, WWW2020)

Can be seen as choices on operations



Implement using a small MLP

A Supernet Representation

S: architecture hyper-parameters

T: parameters

$$\min_{S} \quad H(S,T) \equiv \sum\nolimits_{(i,j)\in\bar{\Omega}} \mathcal{M}(h_{\alpha}(\boldsymbol{u}_i^*, \boldsymbol{v}_j^*)^\top \boldsymbol{w}_{\alpha}^*, O_{ij}) \qquad (9)$$

$$\text{s.t.} \quad \boldsymbol{\alpha} \in \mathcal{C} \text{ and } T^* \equiv \{U^*, V^*, \{\boldsymbol{w}_m^*\}\} = \arg\min_{T} F_{\alpha}(T; S),$$

High level

where F_α is the training objective:

$$F_{\alpha}(T;S) \equiv \sum\nolimits_{(i,j)\in\Omega} \ell(h_{\alpha}(\boldsymbol{u}_i, \boldsymbol{v}_j), O_{ij}) + \frac{\lambda}{2}\|U\|_F^2 + \frac{\lambda}{2}\|V\|_F^2,$$

$$\text{s.t. } \|\boldsymbol{w}_m\|_2 \le 1 \text{ for } m = 1, \ldots, |O|.$$

Low level

- High level: optimize S
- Low level: optimize T
- Bilevel programming is expensive to solve - T^* needs to be obtained from model training

Q. Yao, X. Chen, J. Kwok, Y. Li, C.-J. Hsieh. Efficient Neural Interaction Functions Search for Collaborative Filtering, WWW 2020.

Algorithm & Evaluation – Reusing NASP

Bilevel objective:

$$\min_S \quad H(S, T) \equiv \sum_{(i,j)\in\bar{\Omega}} \mathcal{M}(h_\alpha(\boldsymbol{u}_i^*, \boldsymbol{v}_j^*)^\top \boldsymbol{w}_\alpha^*, O_{ij}) \tag{9}$$
$$\text{s.t.} \quad \boldsymbol{\alpha} \in \mathcal{C} \text{ and } \boldsymbol{T}^* \equiv \{\boldsymbol{U}^*, \boldsymbol{V}^*, \{\boldsymbol{w}_m^*\}\} = \arg\min_{\boldsymbol{T}} F_\alpha(\boldsymbol{T}; \boldsymbol{S}),$$

where F_α is the training objective:

$$F_\alpha(\boldsymbol{T}; \boldsymbol{S}) \equiv \sum_{(i,j)\in\Omega} \ell(h_\alpha(\boldsymbol{u}_i, \boldsymbol{v}_j), O_{ij}) + \frac{\lambda}{2}\|\boldsymbol{U}\|_F^2 + \frac{\lambda}{2}\|\boldsymbol{V}\|_F^2,$$
$$\text{s.t. } \|\boldsymbol{w}_m\|_2 \leq 1 \text{ for } m = 1, \ldots, |\mathcal{O}|.$$

Reuse NASP for fast optimization

- **Effectiveness:** Maintain discrete architectures for S
- **Efficiency:** Update both S and T in an end-to-end and stochastic manner

Algorithm 2 Searching Interaction Function (SIF) algorithm.

Search space $\mathcal{F}$ represented by a structured MLP (Figure 1);
while epoch $t = 1, \cdots, T$ **do**
  Select one operation $\bar{\boldsymbol{\alpha}} = \text{prox}_{\mathcal{C}_1}(\boldsymbol{\alpha})$;
  sample a mini-batch on validation data set;
  Update continuous $\boldsymbol{\alpha}$ for vector-wise operations

$$\boldsymbol{\alpha} = \text{prox}_{\mathcal{C}_2}(\boldsymbol{\alpha} - \eta \nabla_{\bar{\boldsymbol{\alpha}}} H(\boldsymbol{T}, \boldsymbol{S}));$$

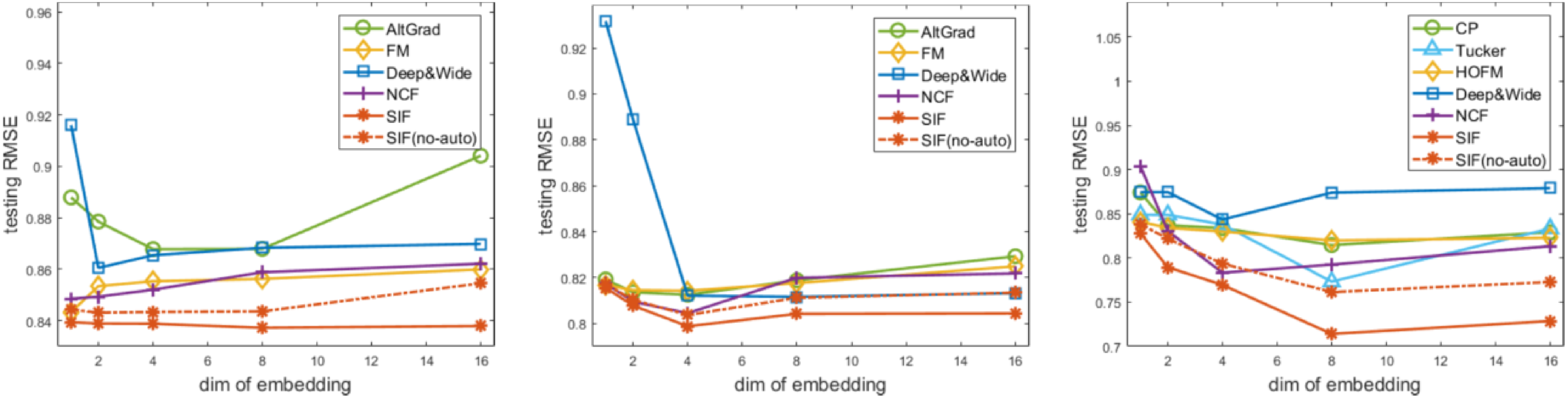
  Update element-wise transformation

$$\boldsymbol{p} = \text{prox}_{\|\cdot\|_2 \leq 1}\left(\boldsymbol{p} - \eta \nabla_{\boldsymbol{p}} H(\boldsymbol{T}, \boldsymbol{S})\right),$$
$$\boldsymbol{q} = \text{prox}_{\|\cdot\|_2 \leq 1}\left(\boldsymbol{q} - \eta \nabla_{\boldsymbol{q}} H(\boldsymbol{T}, \boldsymbol{S})\right);$$

  sample a mini-batch on training data set;
  Get selected operation $\bar{\boldsymbol{\alpha}} = \text{prox}_{\mathcal{C}_1}(\boldsymbol{\alpha})$;
  Update training parameters $\boldsymbol{T}$ with gradients on F_α;
end while
return Searched interaction function (parameterized by $\boldsymbol{\alpha}$, $\boldsymbol{p}$ and $\boldsymbol{q}$, see (7) and (8)).

Comparison with CF Approaches

(i) Alternating gradient descent ("***AltGrad***"); (ii) Factorization machine ("***FM***"); (iii) ***Deep&Wide***; (iv) Neural collaborative filtering ("***NCF***"); (v) ***SIF***; and (iv) ***SIF(no-auto)***, architecture is optimized with training data



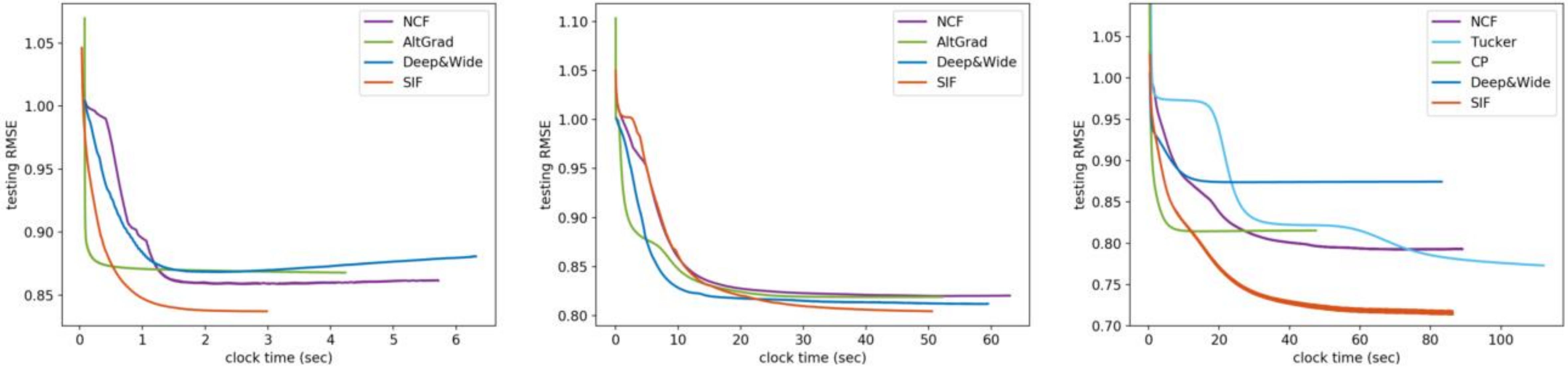
(a) MovieLens-100K. (b) MovieLens-1M. (c) Youtube.

Figure 2: Comparison of testing RMSEs between *SIF* and other CF approaches with different embedding dimension.

SIF is the best, and validation set helps architecture search

Q. Yao, X. Chen, J. Kwok, Y. Li, C.-J. Hsieh. Efficient Neural Interaction Functions Search for Collaborative Filtering, WWW 2020.

Comparison with CF Approaches



(a) MovieLens-100K. (b) MovieLens-1M. (c) Youtube.

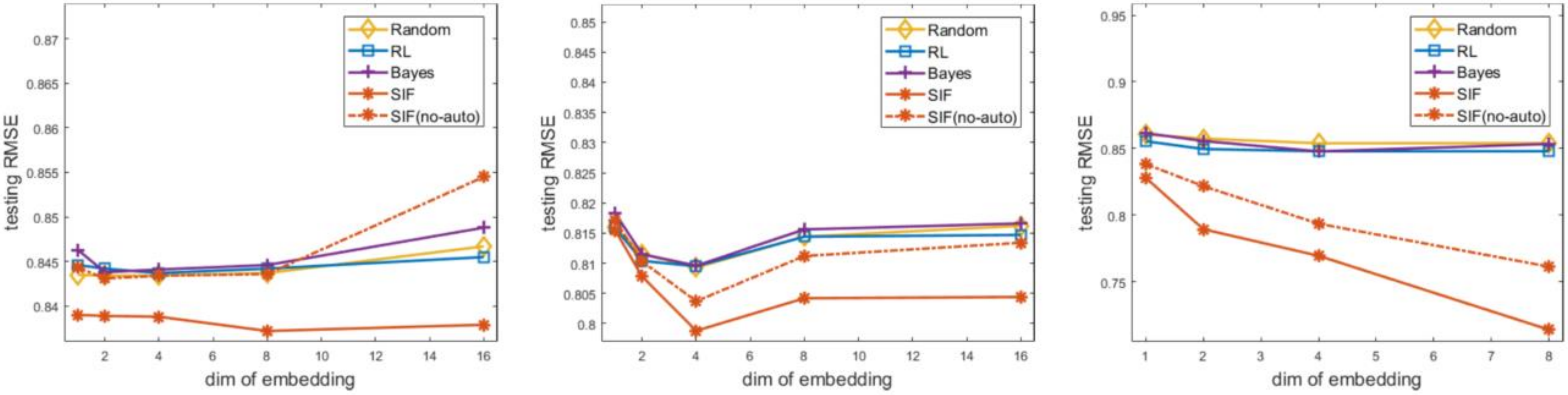
Figure 3: Comparison of the convergence between *SIF* (with searched IFC) and other CF methods when embedded dimension is 8. *FM* and *HOFM* are not shown as their code donot support a callback to record testing performance.

Interaction function obtained from SIF can be trained as fast as state-of-the-art

Q. Yao, X. Chen, J. Kwok, Y. Li, C.-J. Hsieh. Efficient Neural Interaction Functions Search for Collaborative Filtering, WWW 2020.

Comparison with AutoML Approaches

(i) “Random”; (ii) “RL”: reinforcement learning; (iii) “Bayes”: HyperOpt



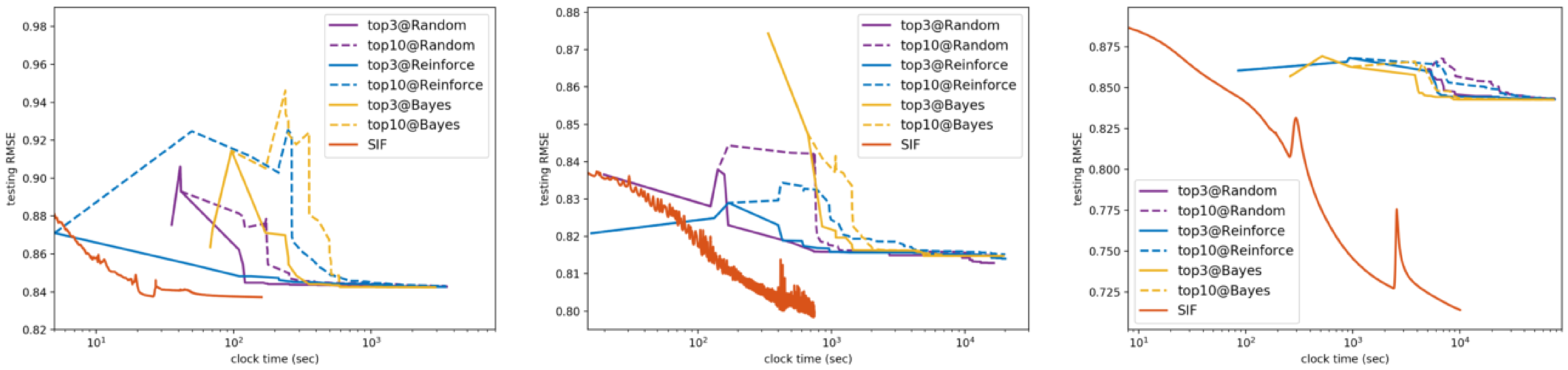
(a) MovieLens-100K. (b) MovieLens-1M. (c) Youtube.

Figure 4: Comparison of testing RMSEs between *SIF* and other AutoML approaches with different embedding dimensions. *Gen-approx* is slow with bad performance, thus is not run on Youtube.

SIF can find better architecture than other AutoML search algorithms

Q. Yao, X. Chen, J. Kwok, Y. Li, C.-J. Hsieh. Efficient Neural Interaction Functions Search for Collaborative Filtering, WWW 2020.

Comparison with AutoML Approaches



(a) MovieLens-100K. (b) MovieLens-1M. (c) Youtube.

Figure 5: Comparison of search efficiency among *SIF* and other AutoML approaches when embedded dimension is 8.

SIF is much faster than other AutoML search algorithms

Q. Yao, X. Chen, J. Kwok, Y. Li, C.-J. Hsieh. Efficient Neural Interaction Functions Search for Collaborative Filtering, WWW 2020.

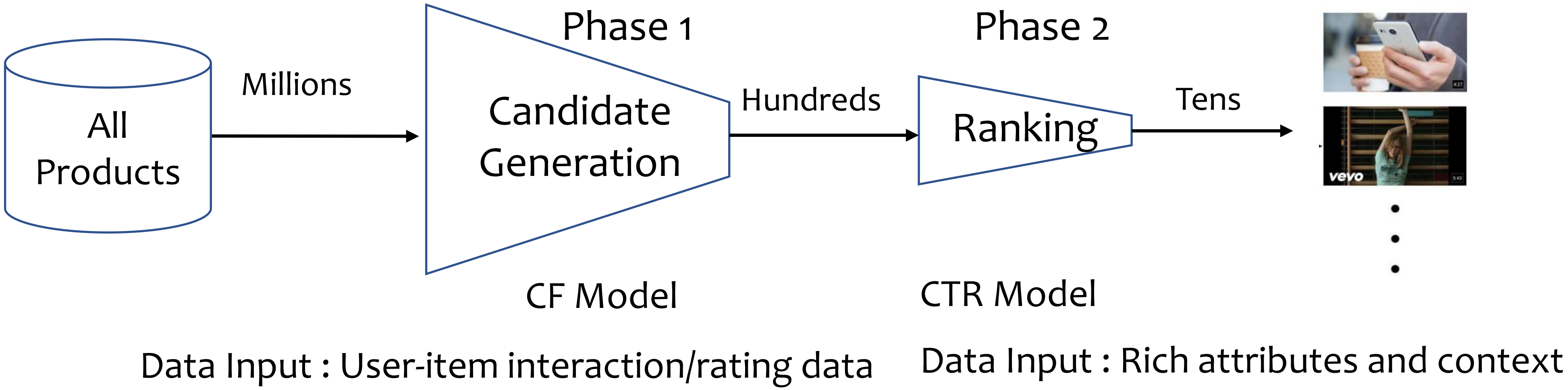
Efficient Neural Interaction Functions Search for Collaborative Filtering

- **Design interaction function automatically**
- **Cover both existing and new interaction functions**
- **One-shot search, update interaction function and embedding jointly**
- **Better performance than experts with slightly higher computation cost**

Outline

- AutoML for Collaborative Filtering Task
- AutoML for Click-through Rate Prediction Task
- AutoML for Tuning Hyper-parameters in RecSys

A pipeline of modern recommendation engine



Click-through rate prediction: tabular data

	age (n)	job (c)	marital (c)	education (c)	balance (n)	housing (c)
0	30	unemployed	married	primary	1787	no
1	33	services	married	secondary	4789	yes
2	35	management	single	tertiary	1350	yes
3	30	management	married	tertiary	1476	yes
4	59	blue-collar	married	secondary	0	yes
5	35	management	single	tertiary	747	no

An example of tabular data (UCI-Bank)

Cross-feature

- What is cross-features?
  - Taking cross-product of sparse features
- Why do we need cross-features?
  - Capture the interaction among categorical features
    - “job $\otimes$ company” can be a strong feature to predict one’s income
  - Achieve great success in real-world business
- Traditional methods
  - Explicit methods: RMI, CMI, etc.
  - Implicit methods: DeepFM[1], xDeepFM[2], etc.

[1] Guo, Huifeng, et al. "DeepFM: a factorization-machine based neural network for CTR prediction." IJCAI 2017
[2] Lian, Jianxun, et al. "xdeepfm: Combining explicit and implicit feature interactions for recommender systems." KDD 2018

Motivation

- Weaknesses of existing methods

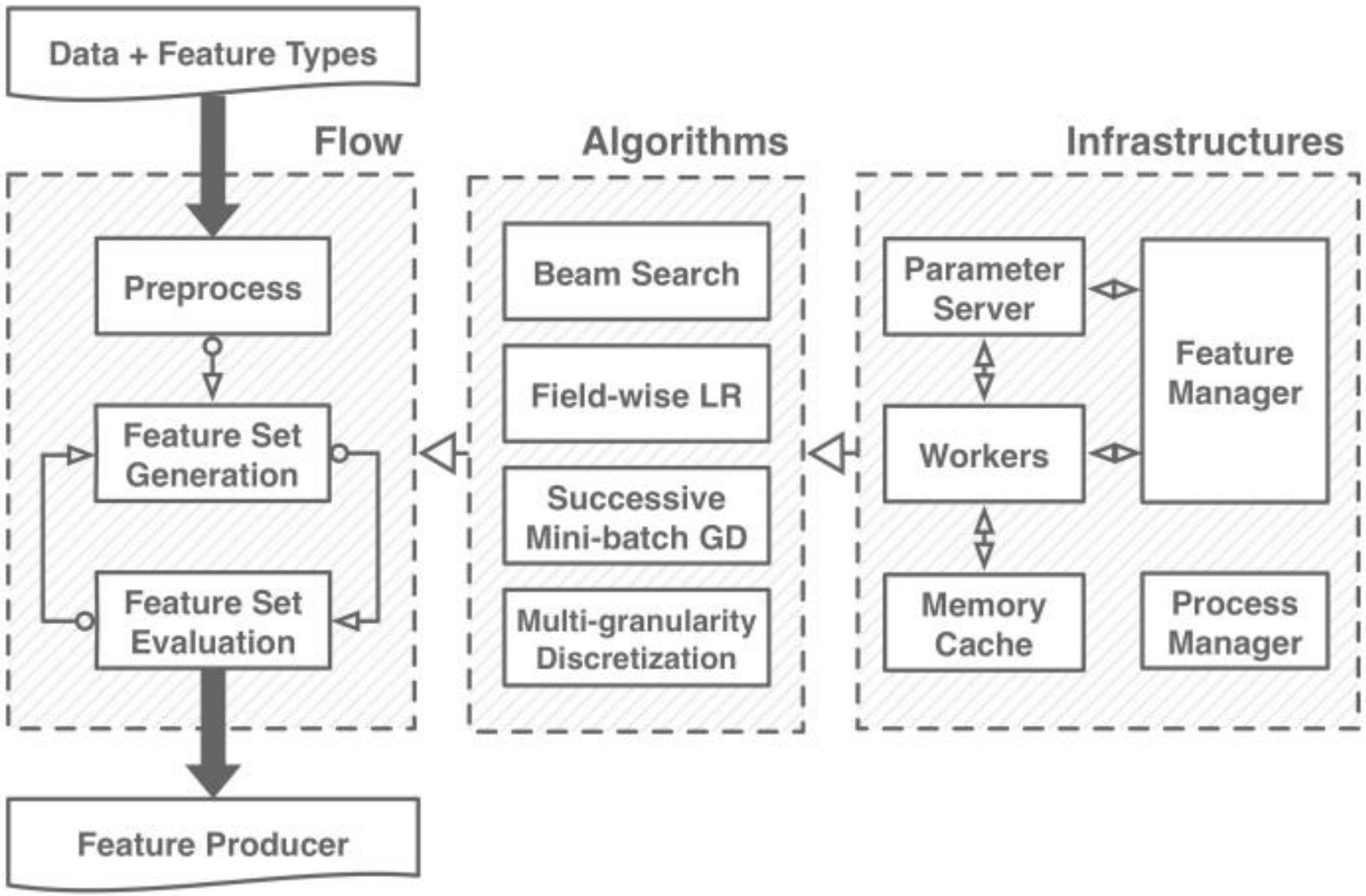
Method	High-order Feature Crossing	Simplicity	Fast Inference	Interpretability
Search-based methods (e.g., [5, 34])	×	medium	√	√
Implicit deep-learning-based methods (e.g., [33, 42])	×	low	×	×
Explicit deep-learning-based methods (e.g., [26, 37])	×	low	×	√
AutoCross	√	high	√	√

Luo, Y et al., Autocross: Automatic feature crossing for tabular data in real-world applications. KDD 2019.

System Framework of AutoCross

- Input
  - Training data
  - Feature type

- Output
  - Feature producer



Luo, Y et al., Autocross: Automatic feature crossing for tabular data in real-world applications. KDD 2019.

AutoCross Loop

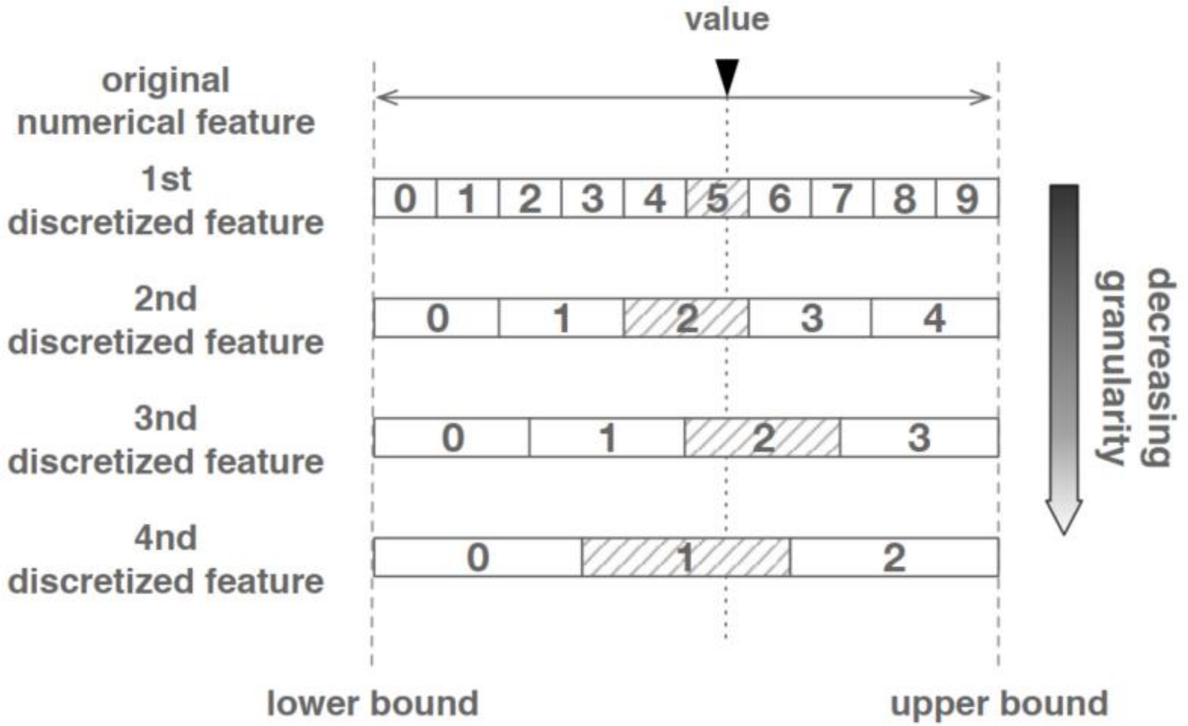
- Feature set generation
  - where candidate feature sets with new cross features are generated

- Feature set evaluation
  - where candidate feature sets are evaluated and the best is selected as a new solution

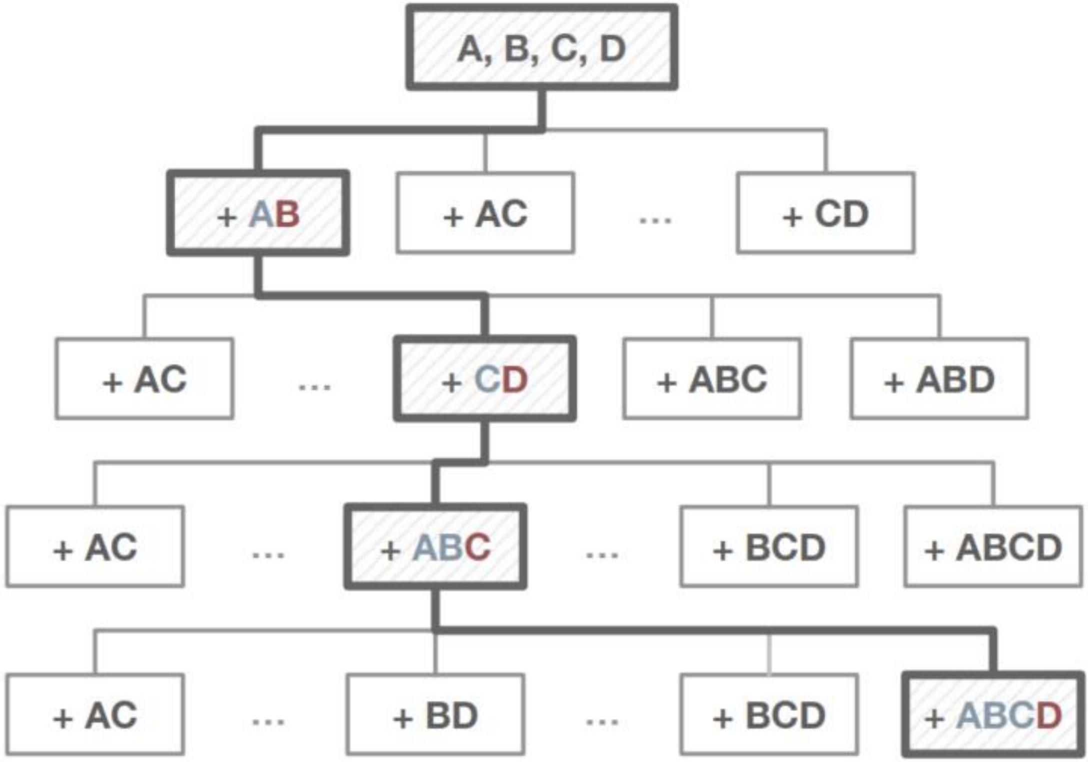
Luo, Y et al., Autocross: Automatic feature crossing for tabular data in real-world applications. KDD 2019.

Method (feature search)

- multi-granularity discretization

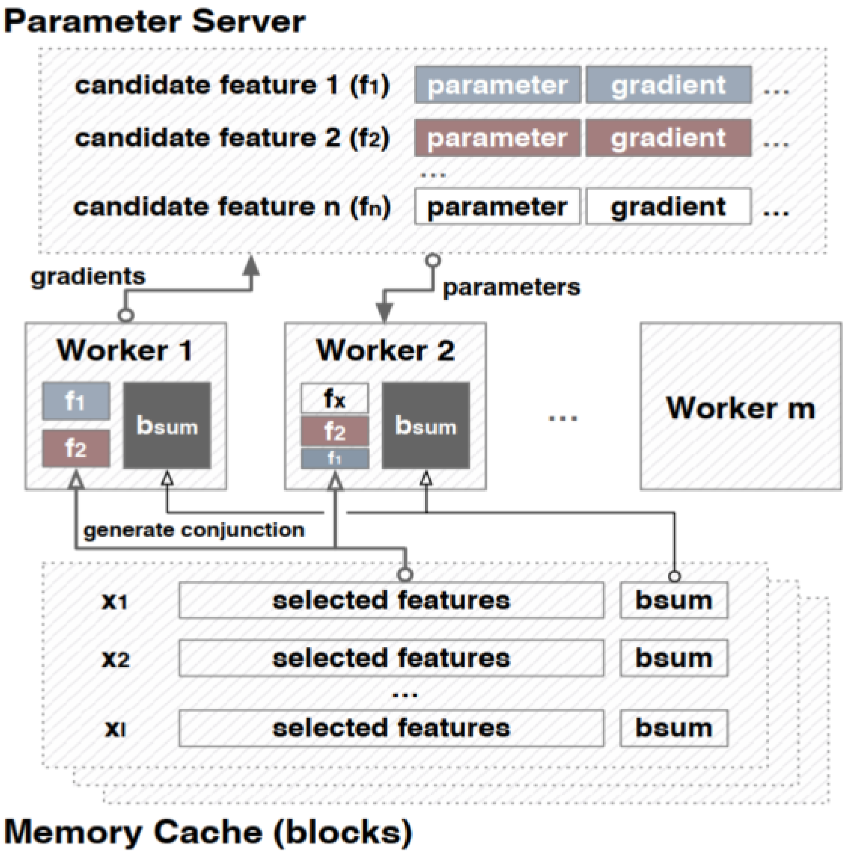


- beam search



Method (feature evaluation)

- field-wise logistic regression



- successive mini-batch gradient descent

Algorithm 2 Successive Mini-batch Gradient Descent (SMBGD).

Require: set of candidate feature sets $\mathbb{S} = \{\mathcal{S}_i\}_{i=1}^{n}$, training data equally divided into $N \geq \sum_{k=0}^{\lceil \log_2 n \rceil - 1} 2^k$ data blocks.

Ensure: best candidate $\mathcal{S}'$.

for $k = 0, 1, \cdots, \lceil \log_2 n \rceil - 1$ **do**

  use additional 2^k data blocks to update the field-wise LR models of all $\mathcal{S} \in \mathbb{S}$, with warm-starting;

  evaluate the models of all $\mathcal{S}$'s with validation AUC;

  keep the top half of candidates in $\mathbb{S}$: $\mathbb{S} \leftarrow$ `top_half`($\mathbb{S}$) (rounding down);

  break if $\mathbb{S}$ contains only one element;

end for

return $\mathcal{S}'$ (the singleton element of $\mathbb{S}$).

Evaluations

- Effectiveness

Benchmark Datasets					
Method	Bank	Adult	Credit	Employee	Criteo
LR (base)	0.9400	0.9169	0.8292	0.8655	0.7855
AC+LR	**0.9455**	**0.9280**	**0.8567**	**0.8942**	0.8034
AC+W&D	0.9420	**0.9260**	**0.8623**	**0.9033**	**0.8068**
CMI+LR	**0.9431**	0.9153	0.8336	0.8901	0.7844
Deep	0.9418	0.9130	0.8369	0.8745	0.7985
xDeepFM	0.9419	0.9131	0.8358	0.8746	**0.8059**
Real-World Business Datasets					
Method	Data1	Data2	Data3	Data4	Data5
LR (base)	0.8368	0.8356	0.6960	0.6117	0.5992
AC+LR	**0.8545**	**0.8536**	**0.7065**	**0.6276**	0.6393
AC+W&D	**0.8531**	**0.8552**	**0.7026**	**0.6260**	**0.6547**
Deep	0.8479	0.8463	0.6936	0.6207	0.6509
xDeepFM	0.8504	0.8515	0.6936	0.6241	**0.6514**

Evaluations

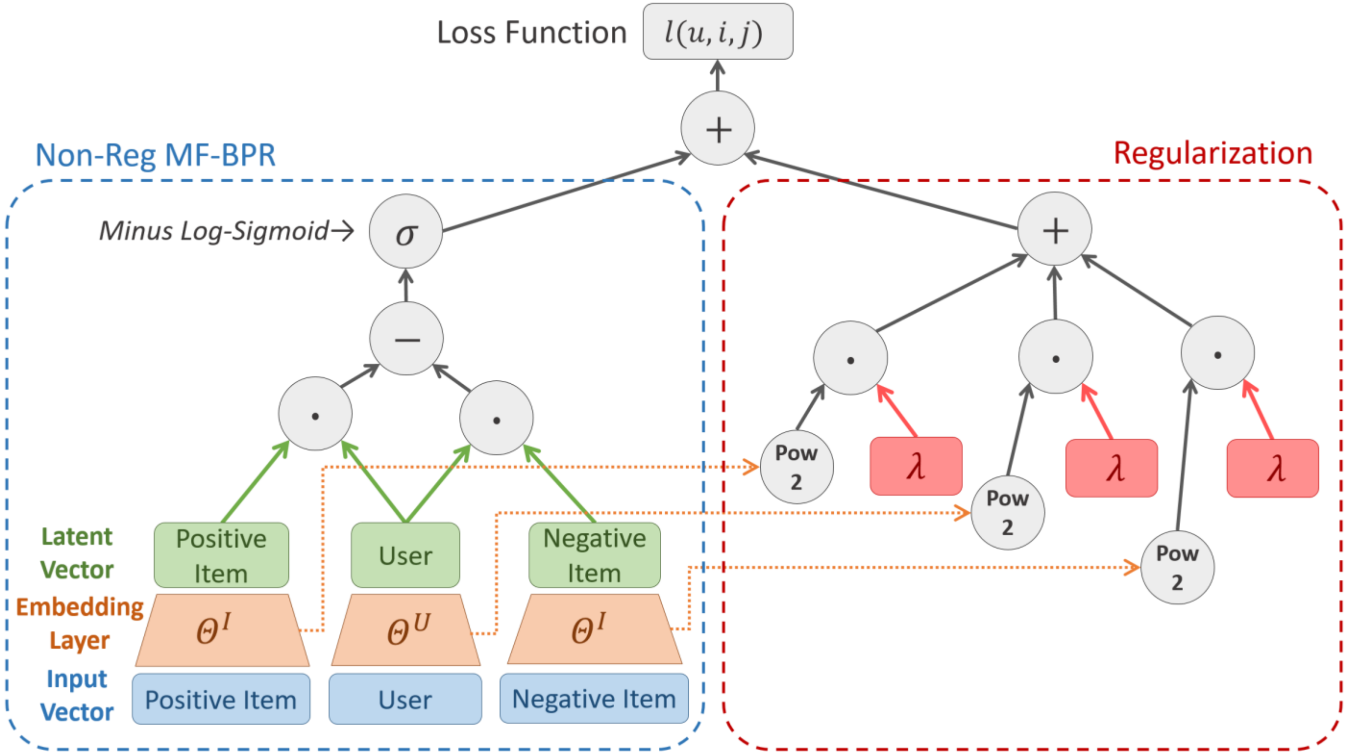
- Efficiency of Inference

Benchmark Datasets					
Method	Bank	Adult	Credit	Employee	Criteo
AC+LR	0.00048	0.00048	0.00062	0.00073	0.00156
AC+W&D	0.01697	0.01493	0.00974	0.02807	0.02698
Deep	0.01413	0.01142	0.00726	0.02166	0.01941
xDeepFM	0.08828	0.05522	0.04466	0.06467	0.18985
Real-World Business Datasets					
Method	Data1	Data2	Data3	Data4	Data5
AC+LR	0.00367	0.00111	0.00185	0.00393	0.00279
AC+W&D	0.03537	0.01706	0.04042	0.02434	0.02582
Deep	0.02616	0.01348	0.03150	0.01414	0.01406
xDeepFM	0.32435	0.11415	0.40746	0.12467	0.13235

Outline

- AutoML for Collaborative Filtering Task
- AutoML for Click-through Rate Prediction Task
- AutoML for Tuning Hyper-parameters in RecSys

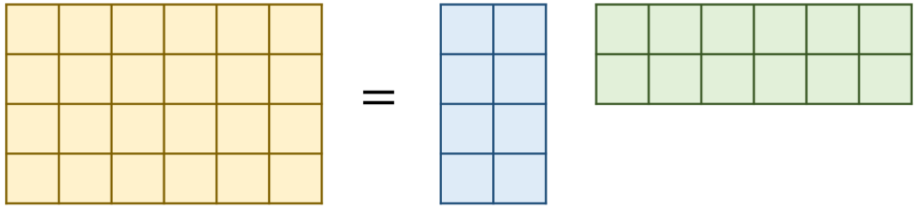
Regularization term in RecSys (take MF-BPR as an example)



$$\begin{aligned} l_{S_T}(\Theta|\lambda) &= \tilde{l}_{S_T}(\Theta) + \Omega(\Theta|\lambda) \\ &= -\sum_{(u,i,j)\in S_T} \ln(\sigma(\hat{y}_{ui}(\Theta) - \hat{y}_{uj}(\Theta))) + \Omega(\Theta|\lambda) \end{aligned}$$

A common concern of RecSys models: Regularization Tuning Headache

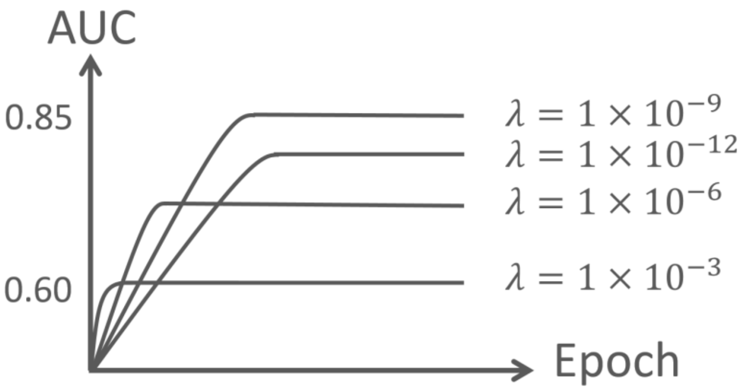
Bob



Model

$$l = \tilde{l}(\theta) + \underbrace{\lambda||\theta||^2}_{\text{Penalty}}$$

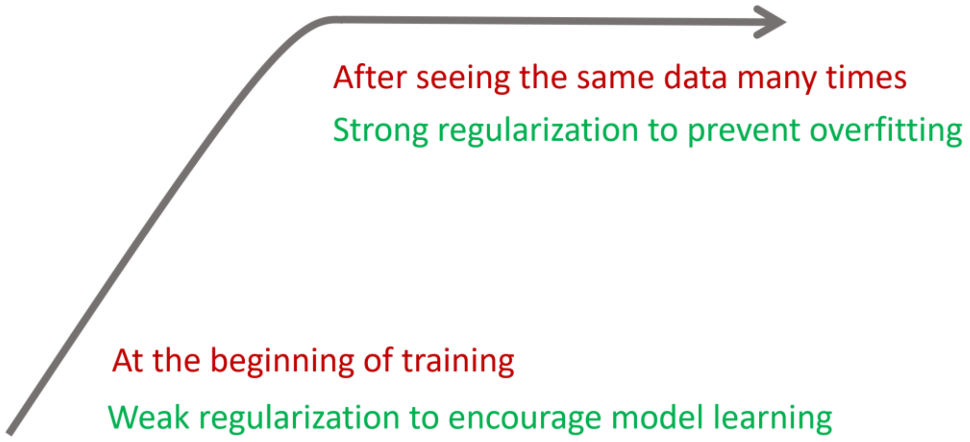
Regularized Loss



What if we can do the regularization automatically?

Why hard to tune?

Hypothesis 1: fixed regularization strength throughout the process



Why hard to tune?

Hypothesis 2: compromise on regularization granularity

What we usually do to determine λ?

- Usually Grid Search or Babysitting $\rightarrow$ global λ

Fine-grained regularization works better

- But unaffordable if we use grid-search!
- Resort to automatic methods!

Dataset Characteristic
Diverse frequencies among users/items

Model Characteristic
Different importance of each latent dimension

Alternating Optimization

$$\min_{\Lambda} \sum_{\{(u',i',j') \in S_V\}} l(u',i',j' \mid \arg\min_{\Theta} \sum_{\{(u,i,j) \in S_T\}} l(u,i,j \mid \Theta, \Lambda))$$

At iteration t

Train the wheel!

- Fix Λ, Optimize Θ

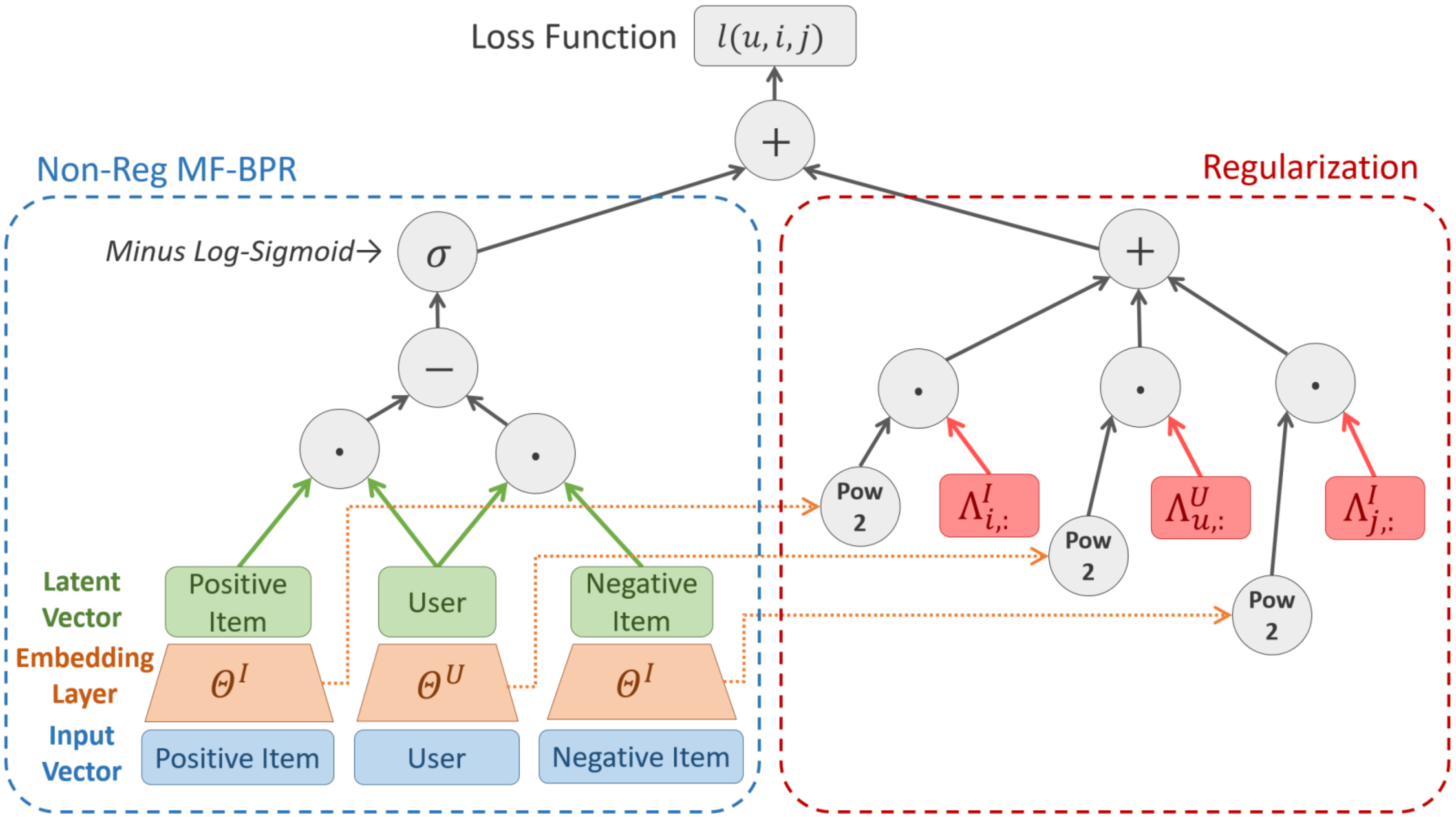
  $\rightarrow$ Conventional MF-BPR except λ is fine-grained now

Train the brake!

- Fix Θ, Optimize Λ

  $\rightarrow$ Find Λ which achieve the smallest validation loss

MF-BPR with fine-grained regularization



Fix Θ, Optimize Λ

Taking a greedy perspective, we look for Λ which can minimize the next-step validation loss

- If we keep using current Λ for next step, we would obtain $\bar{\Theta}_{t+1}$
- Given $\bar{\Theta}_{t+1}$, our aim is $\min_{\Lambda} l_{S_V}(\bar{\Theta}_{t+1})$ with the constraint of non-negative Λ

But how to obtain $\bar{\Theta}_{t+1}$ without influencing the normal Θ update?

- Simulate* the MF update!
  - Obtain the gradients by combining the non-regularized part and penalty part

$$\overline{\frac{\partial l_{S_T}}{\partial \Theta_t}} = \overline{\frac{\partial \tilde{l}_{S_T}}{\partial \Theta_t}} + \frac{\partial \Omega}{\partial \Theta_t}$$

Λ is the only variable here

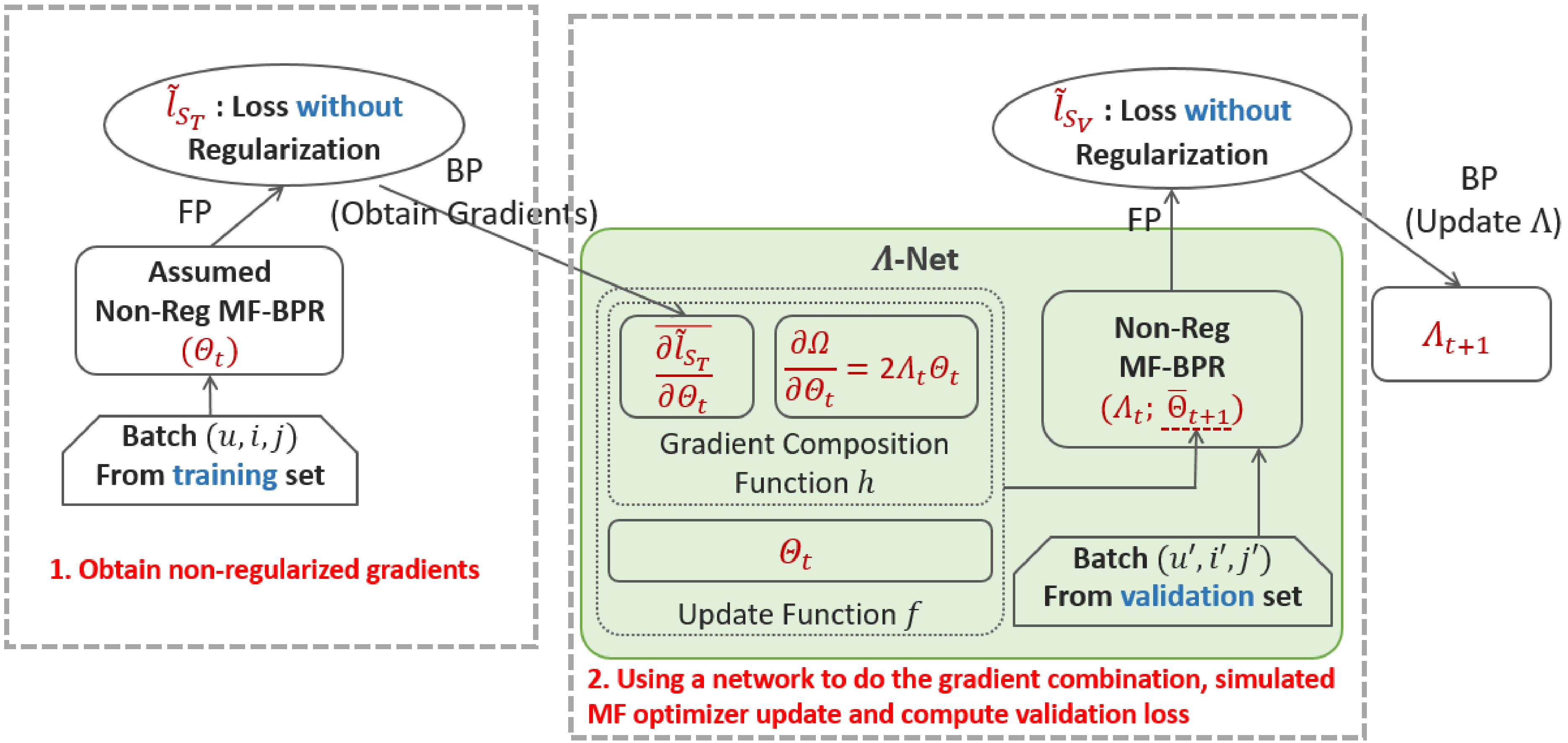
  - Simulate the operations that the MF optimizer would take

$$\bar{\Theta}_{t+1} = f(\Theta_t, \overline{\frac{\partial l_{S_T}}{\partial \Theta_t}})$$

***f* denotes the MF update function**

*: Using – over the letters to distinguish the simulated ones with normal ones

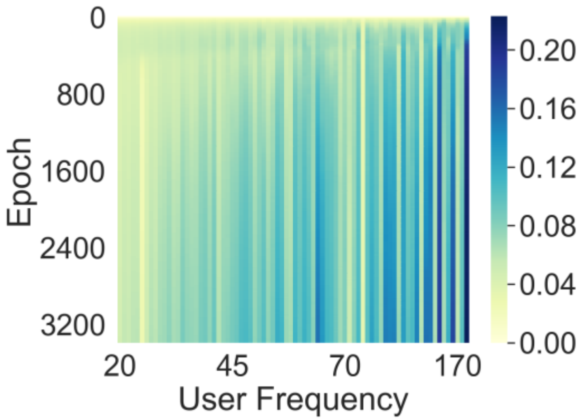
Fix Θ, Optimize Λ in Auto-Differentiation



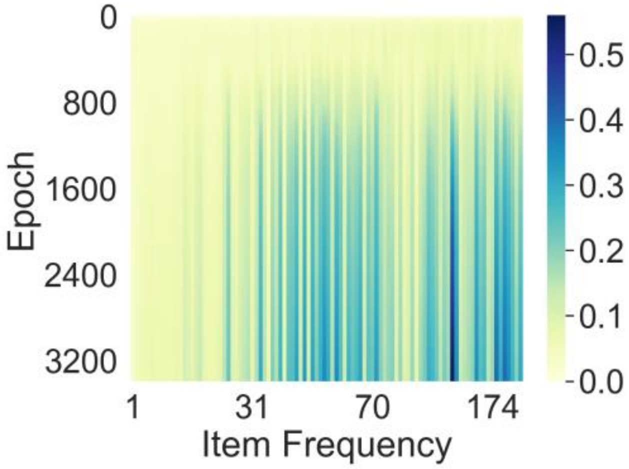
Performance Comparison

Method		Amazon Food Review					MovieLens 10M				
		AUC	HR@50	HR@100	NDCG@50	NDCG@100	AUC	HR@50	HR@100	NDCG@50	NDCG@100
SGDA [26]		0.8130	0.1786	0.3857	0.1002	0.1413	0.9497	0.2401	0.3706	0.0715	0.0934
AMF [15]		0.8197	0.3541	0.4200	0.2646	0.2552	0.9495	0.2625	0.3847	0.0787	0.0985
NeuMF [16]		0.8103	0.3537	0.4127	0.2481	0.2218	0.9435	0.2524	0.3507	0.0760	0.0865
MF-λFix		0.8052	0.3482	0.4163	0.2251	0.2217	0.9497	0.2487	0.3779	0.0727	0.0943
MF-λOpt	-D	0.8109	0.2134	0.3910	0.1292	0.1543	0.9501	0.2365	0.3556	0.0715	0.0909
	-DU	0.8200	0.3694	0.4814	0.2049	0.2570	0.9554	0.2743	0.4109	0.0809	0.1031
	-DI	0.8501	0.2966	0.4476	0.1642	0.2039	0.9516	0.2648	0.3952	0.0804	0.1013
	-DUI	**0.8743**	**0.4470**	**0.5251**	**0.2946**	**0.2920**	**0.9575**	**0.3027**	**0.4367**	**0.0942**	**0.1158**

Analysis of λ-trajectory



(a) For users on Amazon Food Review



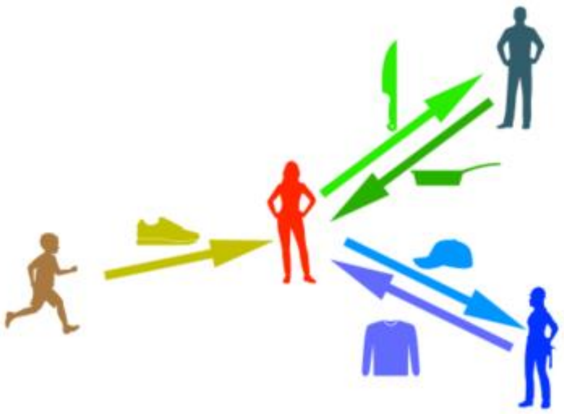
1. Users with higher frequencies are allocated larger λ
2. Items with higher frequencies are allocated larger λ.
3. As training goes on, λs gets larger gradually.

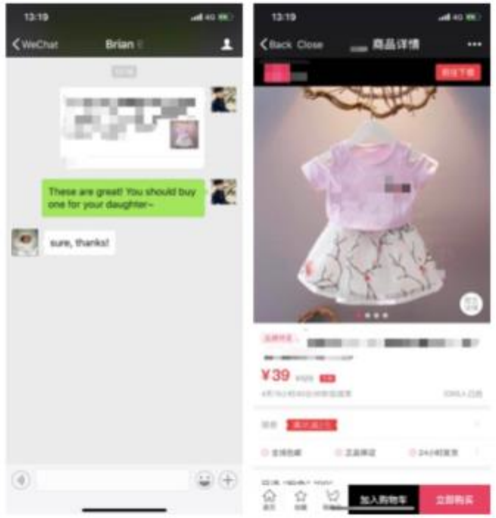
Conclusion

- AutoML can
  - help choose models
  - help select or generate data/feature
  - help tune hyper-parameters.

Discussion

- AutoML can be used in recommendation for more scenarios.
- For example, recommendation in social e-commerce.





In social e-commerce websites, user can purchase products via a link shared by a friend.

Lin, Tzu-Heng, Chen Gao, and Yong Li. "Recommender systems with characterized social regularization." CIKM 2018.
Lin, Tzu-Heng, Chen Gao, and Yong Li. "Cross: Cross-platform recommendation for social e-commerce." SIGIR 2019.

Discussion

- Diverse recommendation tasks in social e-commerce
  - Traditional task: users want to buy some products
  - New task: some users want to share some products
  - New task: some users want to buy some products together with friends
  - New task: …

AutoML can serve as powerful tools when there are diverse recommendation scenarios with different objectives and metrics.

Lin, Tzu-Heng, Chen Gao, and Yong Li. "Recommender systems with characterized social regularization." CIKM 2018.
Lin, Tzu-Heng, Chen Gao, and Yong Li. "Cross: Cross-platform recommendation for social e-commerce." SIGIR 2019.

References

- 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.**

Thank You!

gc16@mails.tsinghua.edu.cn