Optimizing Feature Set for Click-through Rate Prediction

Fuyuan Lyu[1*], Xing Tang[2*], Dugang Liu[3], Liang Chen[2], Xiuqiang He[2], Xue Liu[1]

[1]School of Computer Science, McGill University, Montreal, Canada
[2]FiT, Tencent, Shenzhen, China
[3]Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ) , Shenzhen, China

Presenter: Fuyuan Lyu

Background

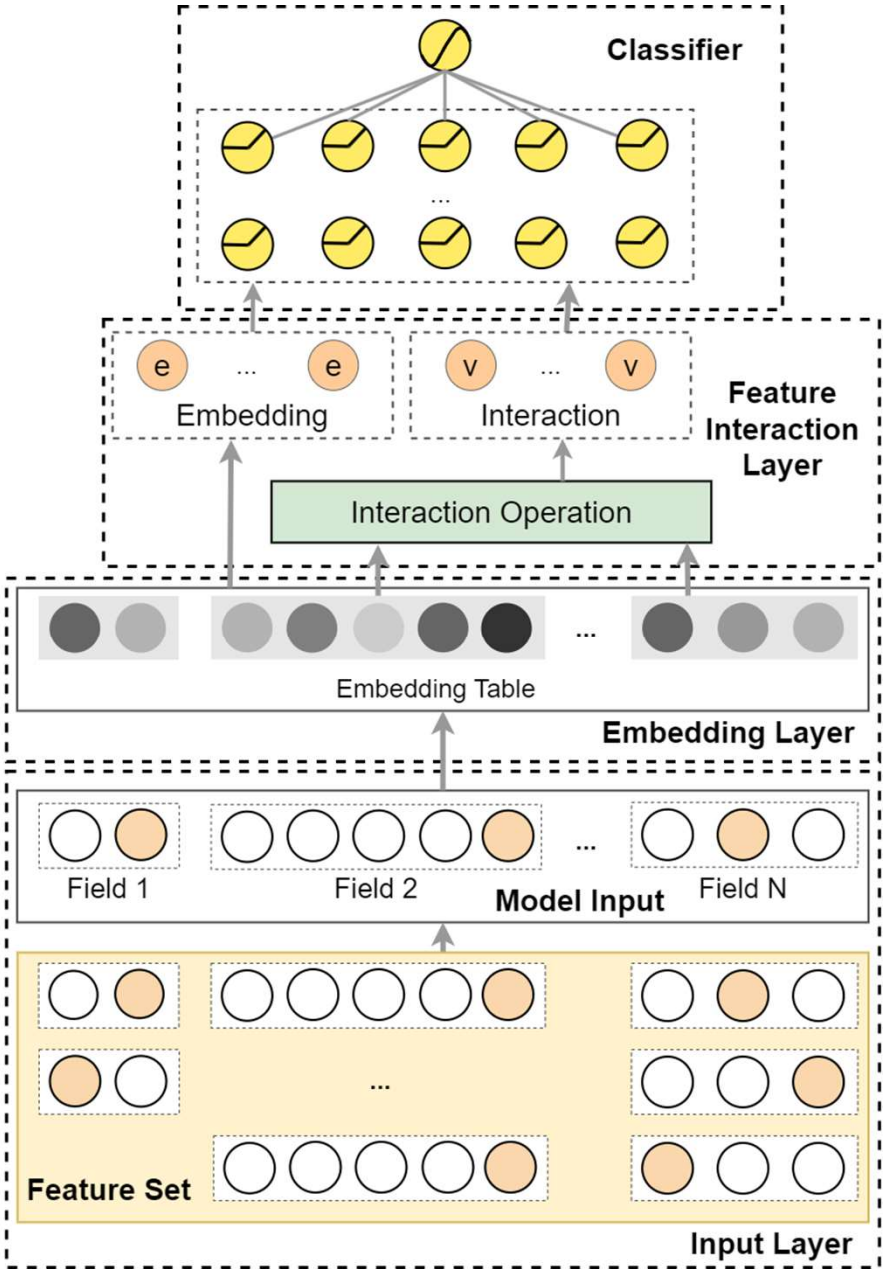
Feature Interaction: *<Male, Montreal>, …*

Enumerate all combinations

Field: *Gender*
Feature: *Male, Female, …*

Field: *City*
Feature: *Montreal, Austin, Shenzhen, …*

Feature Set



Background

Feature Interaction: *<Male, Montreal>, …*

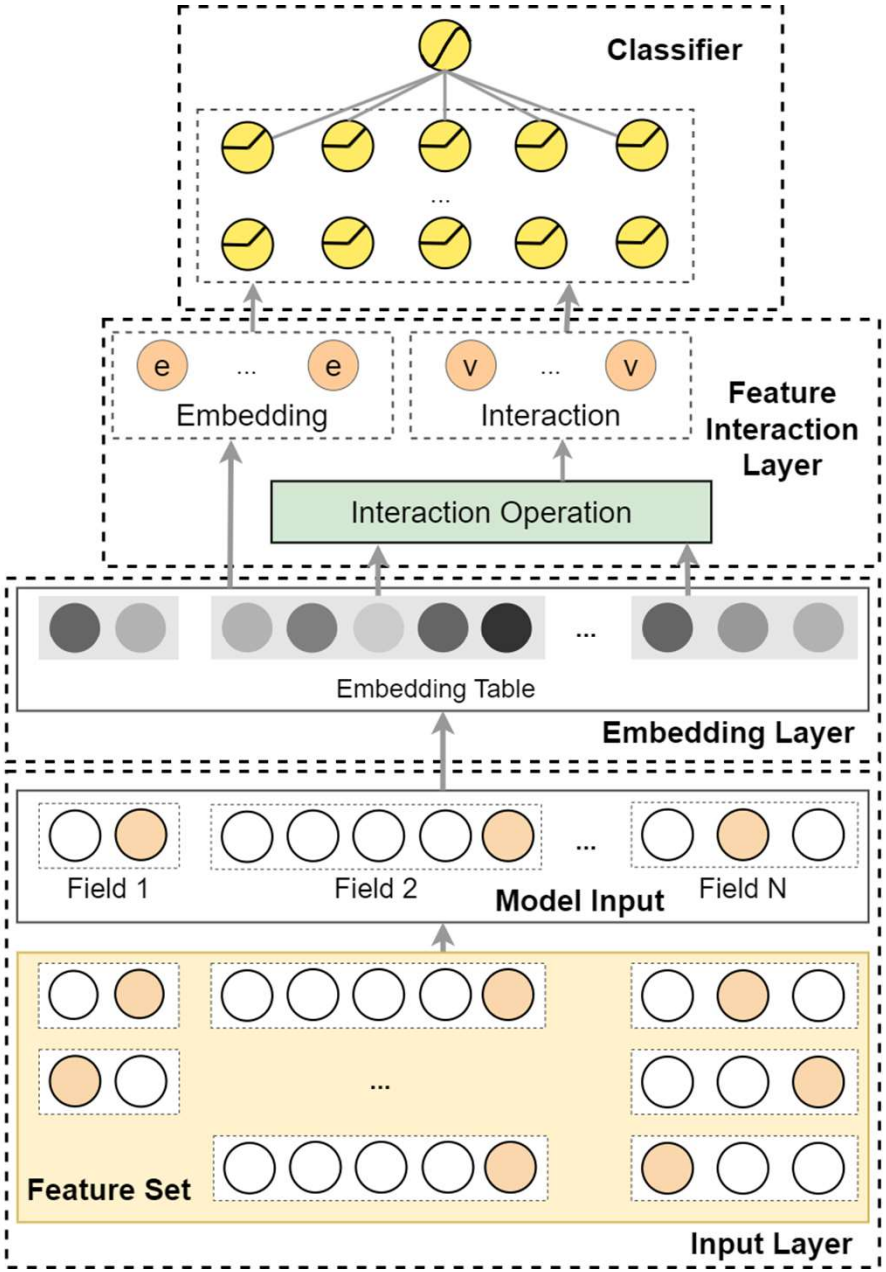
Enumerate all combinations

Field: *Gender*
Feature: *Male, Female, …*

Field: *City*
Feature: *Montreal, Austin, Shenzhen, …*

Feature Set

Sub-optimal



Background

Feature Interaction: *<Male, Montreal>, …*

Enumerate all combinations

Field: *Gender*
Feature: *Male, Female, …*

Field: *City*
Feature: *Montreal, Austin, Shenzhen, …*

Feature Set

Sub-optimal

[4] only focuses on Feature Interaction Selection

- *Redundant Features*

[1,2,3] only focus on Field-level Feature Selection

- *Too Coarse*
- *Redundant Feature Interactions*

[1] Wang, Yejing, et al. "Autofield: Automating feature selection in deep recommender systems." *Proceedings of the ACM Web Conference 2022*. 2022.
[2] Lin, Weilin, et al. "AdaFS: Adaptive Feature Selection in Deep Recommender System." *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2022.
[3] Guo, Yi, et al. "LPFS: Learnable Polarizing Feature Selection for Click-Through Rate Prediction." *arXiv preprint arXiv:2206.00267* (2022).
[4] Liu, Bin, et al. "Autofis: Automatic feature interaction selection in factorization models for click-through rate prediction." *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2020.

Problem Formulation

For all possible features

$$X = \{x_1, x_2, \dots, x_m\}$$

Aim to select

$$X^g \subseteq X$$

subject to

$$\min_{\mathbf{W}} \mathcal{L}(\mathcal{D}|\mathbf{W}),\ \mathcal{D} = \{\mathbf{X}^g, \mathbf{Y}\},$$
$$s.t. \forall \mathbf{x} \in \mathbf{X}^g, \mathcal{L}(\mathbf{X}^g) > \mathcal{L}(\mathbf{X}^g - \{\mathbf{x}\}),$$
$$\forall \mathbf{x} \notin \mathbf{X}^g, \mathcal{L}(\mathbf{X}^g) \geq \mathcal{L}(\mathbf{X}^g + \{\mathbf{x}\}),$$

Jointly consider the influence of feature and its interactions

Select all features that can lower the loss function

Difficulties

Selecting feature and its interactions simultaneously

Traditional pipeline:

1. Select the features
2. Select the feature interactions

Useful interactions cannot help keeping related features

Difficulties

Selecting feature and its interactions simultaneously

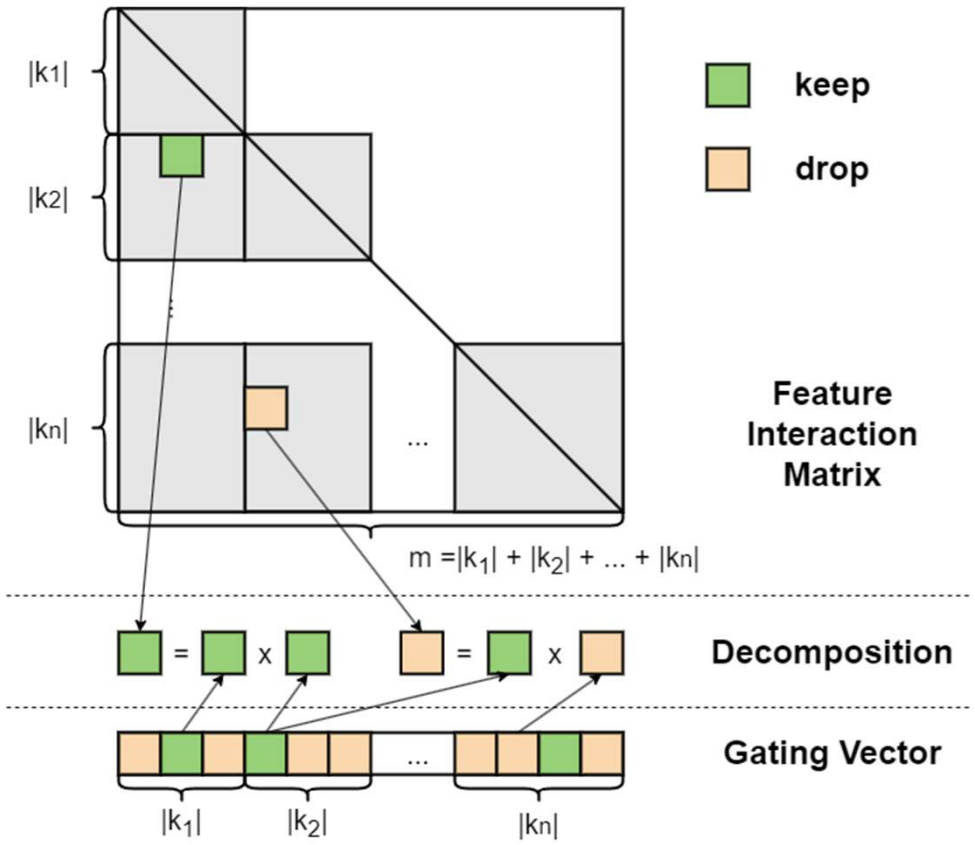
Traditional pipeline:
1. Select the features
2. Select the feature interactions

Useful interactions cannot help keeping related features

$$g_{<i,j>} = g_i \cdot g_j$$

Decomposition

$g_i \in \{0,1\}$: feature selection
$g_{<i,j>} \in \{0,1\}$: feature interaction selection



Difficulties

$g_i \in \{0,1\}$: feature selection
$g_{<i,j>} \in \{0,1\}$: feature interaction selection

Selecting feature and its interactions simultaneously

Traditional pipeline:
1. Select the features
2. Select the feature interactions

Useful interactions cannot help keeping related features

$$g_{<i,j>} = g_i \cdot g_j$$

Decomposition

Granularity from field-level to feature-level

For online advertisement systems

#Field $n \leq 100$

#Feature $m \approx 10^7$

How to efficient explore larger space?

Difficulties

$g_i \in \{0,1\}$: feature selection
$g_{<i,j>} \in \{0,1\}$: feature interaction selection

Selecting feature and its interactions simultaneously

Traditional pipeline:
1. Select the features
2. Select the feature interactions

Useful interactions cannot help keeping related features

$$g_{<i,j>} = g_i \cdot g_j$$

Decomposition

Granularity from field-level to feature-level

For online advertisement systems

#Field $n \leq 100$

#Feature $m \approx 10^7$

How to efficient explore larger space?

$$||g_i||_0 = ||g_i||_1 \approx ||g_i^c||_1$$
$$g_i \in \{0,1\},\ g_i^c \in [0,1]$$

learning-by-continuation training scheme

Experiment

Backbone Models:
FM[1], DeepFM[2], DCN[3], IPNN[4]

Datasets:
Criteo, Avazu, KDD12

Baseline Methods:
- AutoField[5], AdaFS[6], LPFS[7]
- AutoFIS[8]

Evaluation Metrics:
AUC, Logloss and Feature Ratio

$$\text{Ratio} = \#\text{Remaining Features}/m.$$

[1] Rendle, Steffen. "Factorization machines." *2010 IEEE International conference on data mining*. IEEE, 2010.
[2] Guo, Huifeng, et al. "DeepFM: a factorization-machine based neural network for CTR prediction." *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. 2017.
[3] Wang, Ruoxi, et al. "Deep & cross network for ad click predictions." *Proceedings of the ADKDD'17*. 2017. 1-7.
[4] Qu, Yanru, et al. "Product-based neural networks for user response prediction." *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, 2016.
[5] Wang, Yejing, et al. "Autofield: Automating feature selection in deep recommender systems." *Proceedings of the ACM Web Conference 2022*. 2022.
[6] Lin, Weilin, et al. "AdaFS: Adaptive Feature Selection in Deep Recommender System." *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2022.
[7] Guo, Yi, et al. "LPFS: Learnable Polarizing Feature Selection for Click-Through Rate Prediction." *arXiv preprint arXiv:2206.00267* (2022).
[8] Liu, Bin, et al. "Autofis: Automatic feature interaction selection in factorization models for click-through rate prediction." *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2020

Result

Over the Industrial Scenario:

1. Feature Ratio reduced to 0.3136
2. Aver AUC improvement at 0.23% over 3 days

Table 2: Performance Comparison Between OptFS and Feature Selection Methods.

	Method	FM			DeepFM			DCN			IPNN		
		AUC↑	Logloss↓	Ratio↓	AUC↑	Logloss↓	Ratio↓	AUC↑	Logloss↓	Ratio↓	AUC↑	Logloss↓	Ratio↓
Criteo	Backbone	0.8055	0.4457	1.0000	0.8089	0.4426	1.0000	0.8107	0.4410	1.0000	0.8110	0.4407	1.0000
	LPFS	0.7888	0.4604	0.0157	0.7915	0.4579	0.2415	0.7802	0.4743	0.1177	0.7789	0.4705	0.3457
	AutoField	0.7932	0.4567	**0.0008**	0.8072	0.4439	0.3811	**0.8113**	**0.4402**	0.5900	0.8115	**0.4401**	0.9997
	AdaFS	0.7897	0.4597	1.0000	0.8005	0.4501	1.0000	0.8053	0.4472	1.0000	0.8065	0.4448	1.0000
	OptFS	**0.8060**	**0.4454**	0.1387	**0.8100***	**0.4415***	**0.0422**	0.8111	0.4405	**0.0802**	**0.8116**	**0.4401**	**0.0719**
Avazu	Backbone	0.7838	0.3788	1.0000	0.7901	0.3757	1.0000	0.7899	0.3755	1.0000	0.7913	0.3744	1.0000
	LPFS	0.7408	0.4029	0.7735	0.7635	0.3942	0.9975	0.7675	0.3889	0.9967	0.7685	0.3883	0.9967
	AutoField	0.7680	0.3862	**0.0061**	0.7870	0.3773	1.0000	0.7836	0.3782	0.9992	0.7865	0.3770	0.9992
	AdaFS	0.7596	0.3913	1.0000	0.7797	0.3837	1.0000	0.7693	0.3954	1.0000	0.7818	0.3833	1.0000
	OptFS	**0.7839**	**0.3784**	0.8096	**0.7946***	**0.3712***	**0.8686**	**0.7932***	**0.3718***	**0.8665**	**0.7950***	**0.3709***	**0.9118**
KDD12	Backbone	0.7783	0.1566	1.0000	0.7967	0.1531	1.0000	0.7974	0.1531	1.0000	0.7966	0.1532	1.0000
	LPFS	0.7725	0.1578	1.0000	0.7964	0.1532	1.0000	0.7970	**0.1530**	1.0000	0.7967	0.1532	1.0000
	AutoField	0.7411	0.1634	**0.0040**	0.7919	0.1542	0.9962	0.7943	0.1536	**0.8249**	0.7926	0.1541	**0.8761**
	AdaFS	0.7418	0.1644	1.0000	0.7917	0.1543	1.0000	0.7939	0.1538	1.0000	0.7936	0.1539	1.0000
	OptFS	**0.7811***	**0.1560***	0.5773	**0.7988***	**0.1527***	**0.9046**	**0.7987***	**0.1527**	0.8945	**0.7976**	**0.1530**	0.8762

Here $*$ denotes statistically significant improvement (measured by a two-sided t-test with p-value < 0.05) over the best baseline. **Bold** font indicates the best-performed method.

On Criteo, OptFS tends to **reduce feature ratios while keeping the performance**.
On Avazu and KDD12, OptFS tends to **boost model performance**.

Transferability Study

Table 4: Transferability Analysis on Criteo and Avazu.

	Target	Source	Metrics		
			AUC↑	Logloss↓	Ratio↓
Criteo	DeepFM	DeepFM	**0.8100**	**0.4415**	**0.0422**
		DCN	0.8097	0.4419	0.0802
		IPNN	0.8097	0.4418	0.0719
	DCN	DCN	**0.8111**	**0.4405**	0.0802
		DeepFM	0.8106	0.4410	**0.0422**
		IPNN	0.8107	0.4410	0.0719
	IPNN	IPNN	**0.8116**	**0.4401**	0.0719
		DCN	0.8113	0.4404	0.0802
		DeepFM	0.8114	0.4403	**0.0422**
Avazu	DeepFM	DeepFM	**0.7946***	**0.3712***	0.8686
		DCN	0.7873	0.3754	**0.8665**
		IPNN	0.7872	0.3755	0.9118
	DCN	DCN	**0.7932***	**0.3718***	**0.8665**
		DeepFM	0.7879	0.3784	0.8686
		IPNN	0.7860	0.3762	0.9118
	IPNN	IPNN	**0.7950***	**0.3709***	0.9118
		DCN	0.7907	0.3747	**0.8665**
		DeepFM	0.7908	0.3748	0.8686

Here $*$ denotes statistically significant improvement (measured by a two-sided t-test with p-value < 0.05) over the best baseline. **Bold** font indicates the best-performed method.

Aim to prove the necessity of end-to-end training

- Transferability lead to performance drop
- feature interaction operations is correlated with the feature set

Thanks for Listening!

1. A problem called **feature set selection**: jointly considers feature and feature interaction selection.
2. A method named **OptFS**:
   1. decomposition trick
   2. learning-by-continuation training scheme.
3. Superior in efficiency and effectiveness.

Code implementation:
https://github.com/fuyuanlyu/OptFS

Personal webpage:
https://fuyuanlyu.github.io/