Field-weighted Factorization Machines for Click-Through Rate Prediction in Display Advertising

Junwei Pan [1], Jian Xu [2], Alfonso Lobos Ruiz [3], Wenliang Zhao [1], Alan Pan [1], Yu Sun [4], Quan Lu [5]

[1]Yahoo Research [2]TouchPal [3]University of California, Berkeley

[4]LinkedIn [5]Alibaba

April, 2018

Outline

1. Related Works
   - Logistic Regression
   - Degree-2 Polynomial Model
   - Factorization Machines
   - Field-aware Factorization Machines
2. Field-weighted Factorization Machines(FwFMs)
   - Interaction Strength of Field Pairs
   - Field-weighted Factorization Machines(FwFMs)
   - Model Complexity
   - Linear Terms
3. Experiments and Analysis
   - Data Sets
   - Implementation
   - Experiments
   - Hyper-parameter Tuning
   - Analysis

Multi-field Categorical Data

Click-through rate (CTR) prediction is a critical task in online display advertising. The data involved in CTR prediction are typically **multi-field categorical data**.

Here is an example of multi-field categorical data for CTR prediction.

CLICK	User_ID	GENDER	ADVERTISER	PUBLISHER
1	29127394	Male	Nike	news.yahoo.com
-1	89283132	Female	Walmart	techcrunch.com
-1	91213212	Male	Gucci	nba.com
-1	71620391	Female	Uber	tripadviser.com
1	39102740	Male	Adidas	mlb.com

Challenges to Modeling Multi-field Categorical Data

1. Feature interactions are prevalent and need to be specifically modeled.
2. Features from one field often interact differently with features from different other fields.
3. Potentially high model complexity needs to be taken care of in runtime serving.

Outline

1. Related Works
   - Logistic Regression
   - Degree-2 Polynomial Model
   - Factorization Machines
   - Field-aware Factorization Machines
2. Field-weighted Factorization Machines(FwFMs)
   - Interaction Strength of Field Pairs
   - Field-weighted Factorization Machines(FwFMs)
   - Model Complexity
   - Linear Terms
3. Experiments and Analysis
   - Data Sets
   - Implementation
   - Experiments
   - Hyper-parameter Tuning
   - Analysis

Outline

Logistic Regression

LR minimizes the following regularized loss function:

$$\min_{\boldsymbol{w}} \ \lambda \|\boldsymbol{w}\|_2^2 + \sum_{s=1}^{|S|} \log(1 + \exp(-y^{(s)} \Phi_{LR}(\boldsymbol{w}, \boldsymbol{x}^{(s)}))) \tag{1}$$

where λ is the regularization parameter, and

$$\Phi_{LR}(\boldsymbol{w}, \boldsymbol{x}) = w_0 + \sum_{i=1}^{m} x_i w_i \tag{2}$$

is a linear combination of individual features.

Outline

Degree-2 Polynomial Model

However, linear models are not sufficient for tasks such as CTR prediction in which feature interactions are crucial. A general way to address this problem is to add feature conjunctions. It has been shown that Degree-2 Polynomial (Poly2) models can effectively capture the effect of feature interactions.

$$\Phi_{Poly2}(\boldsymbol{w}, \boldsymbol{x}) = w_0 + \sum_{i=1}^{m} x_i w_i + \sum_{i=1}^{m} \sum_{j=i+1}^{m} x_i x_j w_{h(i,j)} \tag{3}$$

where $h(i,j)$ is a function hashing i and j into a natural number in hashing space H to reduce the number of parameters. Otherwise the number of parameters in the model would be in the order of $O(m^2)$. The number of parameters is $m + H$.

Outline

Factorization Machines

FMs implicitly learn a latent vector for each feature. Each latent vector contains k latent factors, where k is a user-specified parameter. Then, the effect of feature interaction is modelled by the inner product of two latent vectors:

$$\Phi_{FMs}((\boldsymbol{w}, \boldsymbol{v}), \boldsymbol{x}) = w_0 + \sum_{i=1}^{m} x_i w_i + \sum_{i=1}^{m} \sum_{j=i+1}^{m} x_i x_j \langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle \tag{4}$$

where $\boldsymbol{v}_i \in \mathbb{R}^K$ is an embedding vector to be learned. The number of parameters is $m + mK$

Outline

FFMs

FMs neglect the fact that a feature might behave differently when it interacts with features from different other fields. Field-aware Factorization Machines (FFMs) model such difference explicitly by learning $n-1$ embedding vectors for each feature, say i, and only using the corresponding one $\boldsymbol{v}_{i,F(j)}$ to interact with another feature j from field $F(j)$:

$$\Phi_{FFMs}((\boldsymbol{w},\boldsymbol{v}),\boldsymbol{x}) = w_0 + \sum_{i=1}^{m} x_i w_i + \sum_{i=1}^{m}\sum_{j=i+1}^{m} x_i x_j \langle \boldsymbol{v}_{i,F(j)}, \boldsymbol{v}_{j,F(i)} \rangle \qquad (5)$$

where $F(i)$ and $F(j)$ are the fields of i and j respectively. Although FFMs have got significant performance improvement over FMs, their number of parameters is in the order of $O(mnK)$. The huge number of parameters of FFMs is undesirable in the real-world runtime systems.

Outline

Outline

Interaction Strengths of Field Pairs

In multi-field categorical data, every feature belongs to one and only one field. We are particularly interested in whether the strength of interactions are different at the field level. In other words, whether the average interaction strength between all feature pairs from a field pair is different from that of other field pairs.

To validate the heterogeneity of field pair interactions, we use **mutual information** between a fiend pair (F_k, F_l) and label variable Y to quantify the interaction strength of the field pair:

$$MI((F_k, F_l), Y) = \sum_{(i,j)\in(F_k,F_l)} \sum_{y\in Y} p((i,j),y) \log \frac{p((i,j),y)}{p(i,j)p(y)} \tag{6}$$

Interaction Strengths of Field Pairs, II

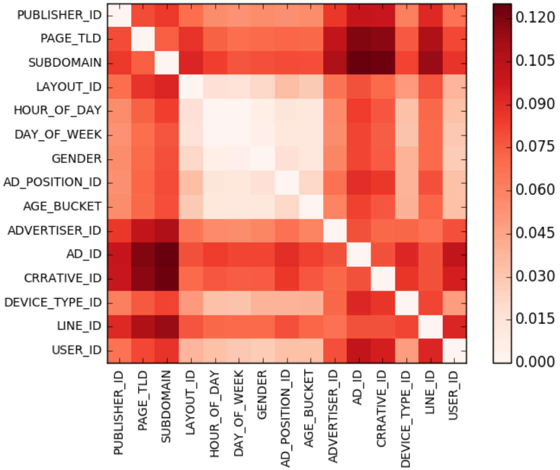


Figure: Heat map of mutual information between each field pair and the label

Outline

Field-weighted Factorization Machines(FwFMs)

We propose to explicitly model the different interaction strengths of different field pairs. More specifically, the interaction of a feature pair i and j in our proposed approach is modeled as

$$x_i x_j \langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle r_{F(i),F(j)}$$

where $\boldsymbol{v}_i, \boldsymbol{v}_j$ are the embedding vectors of i and j, $F(i), F(j)$ are the fields of feature i and j, respectively, and $r_{F(i),F(j)} \in \mathbb{R}$ is a weight to model the interaction strength between field $F(i)$ and $F(j)$. We refer to this model as the Field-weighted Factorization Machines(FwFMs):

$$\Phi_{FwFMs}((\boldsymbol{w}, \boldsymbol{v}, \boldsymbol{r}), \boldsymbol{x}) = w_0 + \sum_{i=1}^{m} x_i w_i + \sum_{i=1}^{m} \sum_{j=i+1}^{m} x_i x_j \langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle r_{F(i),F(j)} \quad (7)$$

Field-weighted Factorization Machines

Here is the list of formulations of all models mentioned above:

$$\Phi_{Linear}(\boldsymbol{w}, \boldsymbol{x}) = w_0 + \sum_{i=1}^{m} w_i x_i$$

$$\Phi Poly_2(\boldsymbol{w}, \boldsymbol{x}) = w_0 + \sum_{i=1}^{m} w_i x_i + \sum_{i=1}^{m} \sum_{j=i+1}^{m} w_{h(i,j)} x_i x_j$$

$$\Phi_{FMs}((\boldsymbol{w}, \boldsymbol{v}), \boldsymbol{x}) = w_0 + \sum_{i=1}^{m} x_i w_i + \sum_{i=1}^{m} \sum_{j=i+1}^{m} x_i x_j \langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle$$

$$\Phi_{FFMs}((\boldsymbol{w}, \boldsymbol{v}), \boldsymbol{x}) = w_0 + \sum_{i=1}^{m} w_i x_i + \sum_{i=1}^{m} \sum_{j=i+1}^{m} \langle \boldsymbol{v}_{i,F(j)}, \boldsymbol{v}_{j,F(i)} \rangle x_i x_j$$

$$\Phi_{FwFMs}((\boldsymbol{w}, \boldsymbol{v}, \boldsymbol{r}), \boldsymbol{x}) = w_0 + \sum_{i=1}^{m} w_i x_i + \sum_{i=1}^{m} \sum_{j=i+1}^{m} \langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle x_i x_j r_{F(i),F(j)}$$

Outline

Model Complexity

The number of parameters in FwFMs is $m + mK + n(n-1)/2$, where m accounts for the weights for each feature in the linear part $\{w_i | i = 1, ..., m\}$ and mK accounts for the embedding vectors for all the features $\{\boldsymbol{v}_i | i = 1, ..., m\}$ and $n(n-1)/2$ accounts for each field pair. Since $n \ll m$, the number of parameters of FwFMs is very close to FMs, and is much less than FFMs.

Model	Number of Parameters
LR	m
Poly2	$m + H$
FMs	$m + mK$
FFMs	$m + m(n-1)K$
FwFMs	$m + mK + \frac{n(n-1)}{2}$

Outline

Linear Terms of FwFMs

In the linear terms of FwFMs formula

$$\phi_{FwFMs}(\boldsymbol{v}, \boldsymbol{x}) = w_0 + \sum_{i=1}^{m} w_i x_i + \sum_{i=1}^{m} \sum_{j=i+1}^{m} \langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle x_i x_j r_{F(i),F(j)}$$

we learn a weight w_i for each feature to model its effect with the label, using binary variable x_i to represent feature i. However, the embedding vectors $\boldsymbol{v}_i$ learned in the interaction terms should capture more information about feature i, therefore we propose to use $x_i \boldsymbol{v}_i$ to represent each feature in the linear terms as well.

We can learn one linear weight vector $\boldsymbol{w}_i$ for each feature and the linear terms become:

$$\sum_{i=1}^{m} x_i \langle \boldsymbol{v}_i, \boldsymbol{w}_i \rangle \tag{8}$$

Linear Terms of FwFMs

There are mK parameters in the feature-wise linear weight vectors, and the total number of parameters is $2mK + \frac{n(n-1)}{2}$, which is almost the double of original FwFMs.

Another approach is to learn one linear weight vector $\boldsymbol{w}_{F(i)}$ for each field and all features from the same field $F(i)$ use the same linear weight vector. Then the linear terms can be formulated as:

$$\sum_{i=1}^{m} x_i \langle \boldsymbol{v}_i, \boldsymbol{w}_{F(i)} \rangle \tag{9}$$

The parameter number of these kind of FwFMs is $nK + mK + \frac{n(n-1)}{2}$, which is almost the same as original FwFMs since both K and n are usually in the order of tens.

Outline

Outline

Data Sets

We use the following two data sets in our experiments.

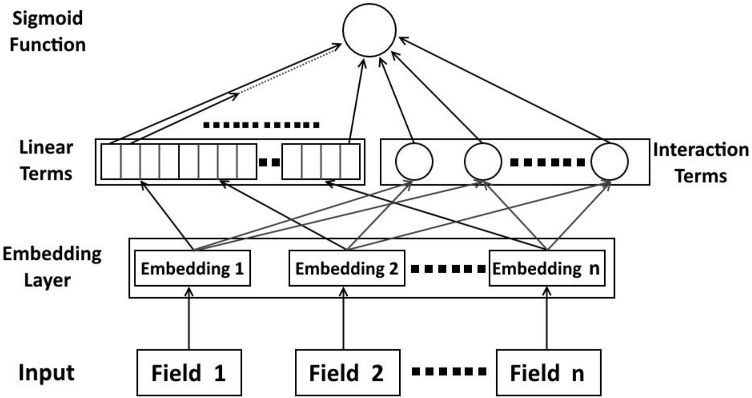
1. Criteo CTR data set: This is the data set used for the Criteo Display Advertising Challenge. We split the data into training, validation and test sets randomly by 60%:20%:20%.
2. Yahoo(Oath) CTR data set: We use two-week display advertising click log from our ad serving system as the training set and the log of next day and the day after next day as validation and test set respectively.

Data set		Samples	Fields	Features
Criteo	Train	27,502,713	26	399,784
	Validation	9,168,820	26	399,654
	Test	9,169,084	26	399,688
Yahoo(Oath)	Train	24,885,731	15	156,401
	Validation	7,990,874	15	101,217
	Test	8,635,361	15	100,515

Outline

Implementation

We implement FwFMs in Tensorflow:



Outline

Performance Comparison

Models	Parameters	AUC		
		Training	Validation	Test
LR	$\eta = 1e-4, \lambda = 1e-7, t = 15$	0.8595	0.8503	0.8503
Poly2	$s = 7, c = 2$	0.8652	0.8542	0.8523
FMs	$\eta = 5e-4, \lambda = 1e-6, k = 10, t = 10$	0.8768	0.8628	0.8583
FFMs	$\eta = 1e-4, \lambda = 1e-7, k = 10, t = 3$	**0.8833**	**0.8660**	**0.8624**
FwFMs	$\eta = 1e-4, \lambda = 1e-5, k = 10, t = 15$	0.8827	0.8659	0.8614

(a) Oath data set

Models	Parameters	AUC		
		Training	Validation	Test
LR	$\eta = 5e-5, \lambda = 1e-6, t = 14$	0.7716	0.7657	0.7654
Poly2	$s = 7, c = 2$	0.7847	0.7718	0.7710
FMs	$\eta = 1e-4, \lambda = 1e-6, k = 10, t = 10$	0.7925	0.7759	0.7761
FFMs	$\eta = 5e-4, \lambda = 1e-7, k = 10, t = 3$	**0.7989**	**0.7781**	**0.7768**
FwFMs	$\eta = 1e-4, \lambda = 1e-6, k = 10, t = 8$	0.7941	0.7772	0.7764

(b) Criteo data set

Figure: Comparison among models on Criteo and Oath CTR data sets.

Comparision with FFMs using the same number of parameters

One critical drawback of using FFMs is that their number of parameters is in the order of $O(mnK)$, which would be too large to fit in memory. There are two solutions to reduce the number of parameters in FFMs: use a smaller K or use hashing tricks on the features with a small hashing space H. We compare FwFMs and FFMs in the same memory size by using the methods mentioned above to reduce the parameters of FFMs:

Model	Oath data set			Criteo data set		
	Training	Validation	Test	Training	Validation	Test
FFMs($K = 2, H = \frac{10}{14 \cdot 2} m$)	0.8743	0.8589	0.8543	0.7817	0.7716	0.7719
FFMs($K = 4, H = \frac{10}{14 \cdot 4} m$)	0.8708	0.8528	0.8418	0.7697	0.7643	0.7641
FwFMs	**0.8827**	**0.8659**	**0.8614**	**0.7941**	**0.7772**	**0.7764**

FwFMs with different Linear Terms

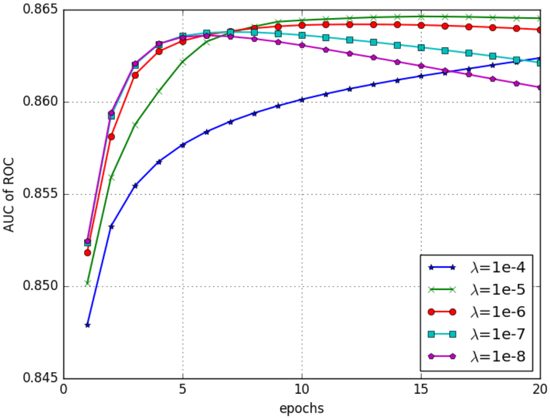
We denote FwFMs with original linear weights as FwFMs_LW, denote FwFMs with feature-wise linear weight vectors as FwFMs_FeLV, denote FwFMs with field-wise linear weight vectors as FwFMs_FiLV.

Model	Oath data set			Criteo data set		
	Training	Validation	Test	Training	Validation	Test
FwFMs_LW	**0.8827**	0.8659	0.8614	0.7941	0.7772	0.7764
FwFMs_FeLV	0.8829	**0.8665**	0.8623	**0.7945**	**0.7774**	0.7763
FwFMs_FiLV	0.8799	0.8643	**0.8635**	0.7917	0.7766	**0.7766**

Outline

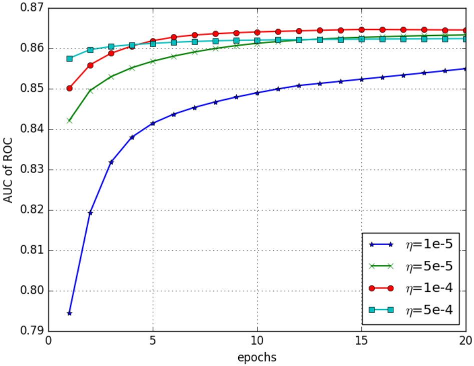
Regularization Coefficient

We add L_2 regularizations of all parameters in FwFMs to the loss function to prevent over-fitting. The following figure shows the AUC on validation set using different λ. We get the best performance on validation set using $\lambda = 1e - 5$.



Learning Rate

We have done experiments to check the impact of learning rate η to the performance of FwFMs. It shows that by using a small η, we can keep improving the performance on validation set slowly in the first 20 epochs, while using a large η will improve the performance quickly and then lead to over-fitting. In all experiments on Oath data set we choose $\eta = 1e - 4$.



Latent Vector Dimension

We conduct experiments to investigate the impact of embedding vector dimension K on the performance of FwFMs. The following table shows that the AUC changes only a little when we use different K. We choose $K = 10$ in FwFMs since it gets best trade-off between performance and training time.

k	Train AUC	Validation AUC	Training time (s)
5	0.8794	0.8621	320
10	0.8827	0.8659	497
15	0.8820	0.8644	544
20	0.8822	0.8640	636
30	0.8818	0.8647	848
50	0.8830	0.8652	1113
100	0.8830	0.8646	1728
200	0.8825	0.8646	3250

Outline

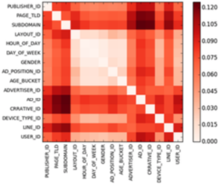
Learned Field Interaction Strengths

To measure the learned field interaction strengths, we define the following metric:

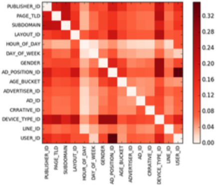
$$\frac{\sum_{(i,j)\in(F_k,F_l)} I(i,j)\cdot\#(i,j)}{\sum_{(i,j)\in(F_k,F_l)}\#(i,j)} \tag{10}$$

where $\#(i,j)$ is the number of times feature pair (i,j) appears in the training data, $I(i,j)$ is the learned strength of interaction between feature i and j. For FMs $I(i,j) = |\langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle|$, for FFMs $I(i,j) = |\langle \boldsymbol{v}_{i,F_l}, \boldsymbol{v}_{j,F_k} \rangle|$, for FwFMs $I(i,j) = |\langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle \cdot r_{F_k,F_l}|$.
Note that we sum up the absolute values of the inner product terms otherwise positive values and negative values would counteract with each others.

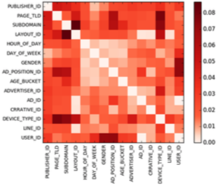
Study of Learned Field Interaction Strengths



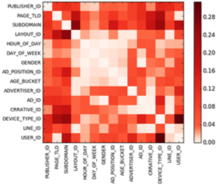
(a) Heat map of mutual informations between field pairs and labels.



(b) Heat map of learned field interaction strengths of FMs. Pearson correlation coefficient with mutual informations: -0.1210.



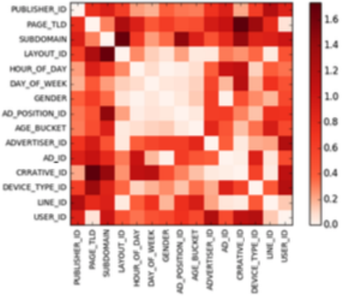
(c) Heat map of learned field interaction strengths of FFMs. Pearson correlation coefficient with mutual informations: 0.1544.



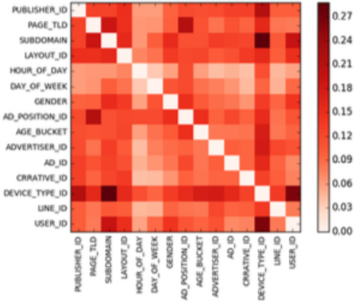
(d) Heat map of learned field interaction strengths of FwFMs. Pearson correlation coefficient with mutual informations: 0.4271.

Figure: Heat maps of mutual information (a) and learned field pair interaction strengths of FMs(b), FFMs(c) and FwFMs(d) on Oath CTR data set

Study of Learned Field Interaction Strengths, II



(a) Heat map of field interaction weight r_{F_k,F_l}. Its Pearson correlation coefficient with mutual information is 0.5554.



(b) Heat map of learned field interaction strengths of FwFMs without r_{F_k,F_l}. Its Pearson correlation coefficient with mutual information is 0.0522.

Figure: Heat map of learned field interaction weight r_{F_k,F_l}, and heat map of learned interaction strengths between field pairs of FwFMs without r_{F_k,F_l}

Summary

In this paper, we propose Field-weighted Factorization Machines (FwFMs) for CTR prediction in online display advertising.

1. We show that FwFMs are competitive to FFMs with significantly less parameters.
2. When using the same number of parameters, FwFMs can achieve consistently better performance than FFMs.
3. We also introduce a novel linear term representation to augment FwFMs so that their performance can be further improved.
4. Finally, comprehensive analysis on real-world data sets also verifies that FwFMs can indeed learn different feature interaction strengths from different field pairs.

Further Reading I

S. Rendle.
Factorization machines Data Mining (ICDM), 2010 IEEE 10th International Conference on

Yuchin Juan, et.al.
Field-aware factorization machines for CTR prediction Proceedings of the 10th ACM Conference on Recommender Systems

Yanru Qu, et.al.
Product-based neural networks for user response prediction Data Mining (ICDM), 2016 IEEE 16th International Conference on

O. Chapelle.
Simple and scalable response prediction for display advertising ACM Transactions on Intelligent Systems and Technology (TIST)

Further Reading II

Weinan Zhang.
Deep learning over multi-field categorical data European conference on information retrieval

Yuchin Yuan.
Field-aware factorization machines in a real-world online advertising system Proceedings of the 26th International Conference on World Wide Web Companion

H. McMahan.
Ad click prediction: a view from the trenches Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining

M. Richardson.
Predicting clicks: estimating the click-through rate for new ads Proceedings of the 16th international conference on World Wide Web