بسم الله الرحمن الرحیم

User Conversion Prediction in Display Advertisements

Mohammadreza Rezaei

Supervised by
Hamid R. Rabiee

Data Science & Machine Learning Lab (DML)
Sharif University of Technology

Mar 2021

Outline

1. Introduction
2. Related Works
3. Proposed Method
4. Experiments
5. Future Work

Introduction

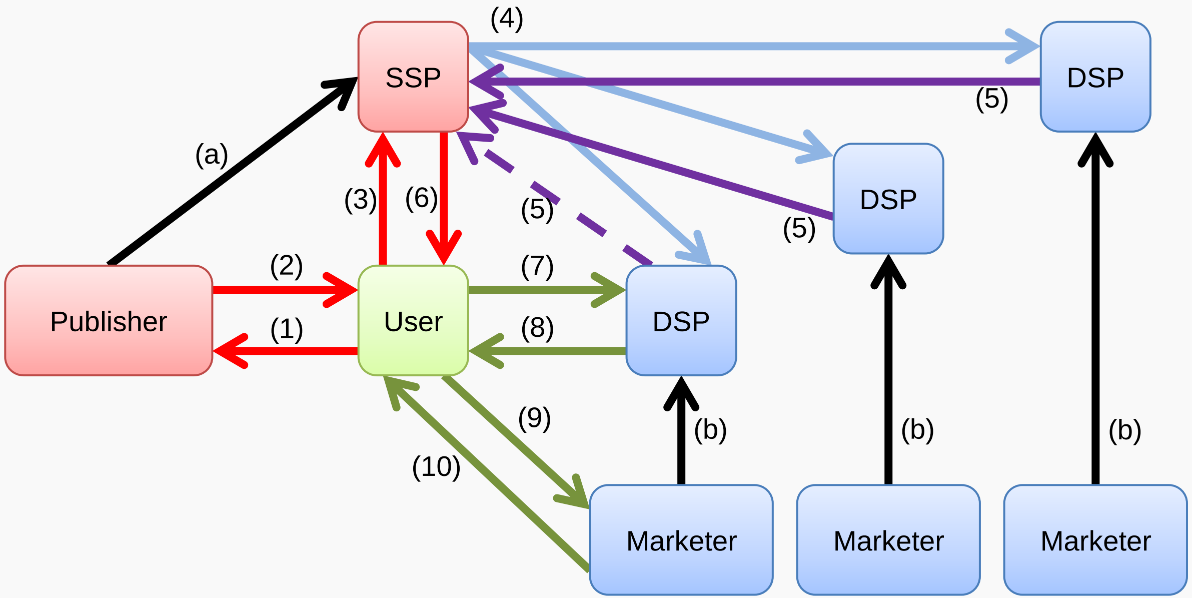
Background

Display Advertising

- An easy and effective type of advertisement for businesses
- Creates significant revenues for big tech companies

Real Time Bidding (RTB):

- A process that decides which ad will be shown to the user



Problem Definition

Click Through Rate (CTR) Prediction

- Predict probability of a click
  - On a certain **ad**
  - By a certain **user**
  - In a certain **context**
- Binary classification

Main challenges

Challenges:

- Imbalanced data
- High dimensionality and sparsity
- Cold Start
    - Ad Cold Start
    - User Cold Start
- Training speed
- Testing speed

Contribution

How to estimate click probability despite the mentioned challenges?

- Study the previous work on CTR prediction
- Explain the effect of these challenges on their performances
- Introduce a novel model for CTR prediction

Related Works

Related Work

- Classical approaches
- Factorization Machines
- Deep Methods

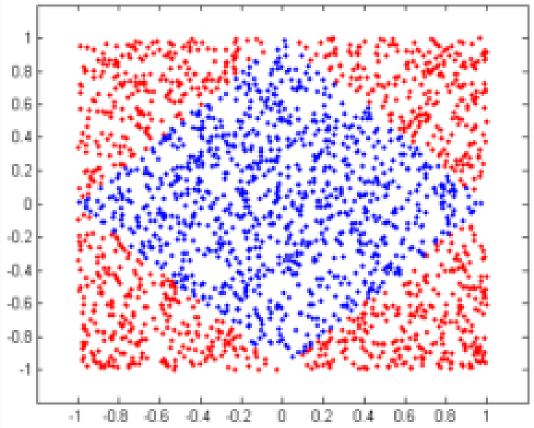
Classical approaches:

- SVM

$$\hat{y}(x) = w_0 + \sum_{i=1}^{n} w_i x_i, \qquad w_0 \in \mathbb{R}, \quad w \in \mathbb{R}^n$$

- Piece-wise Linear

$$y = g\left(\sum_{j=1}^{m} \sigma(u_j^T x)\eta(w_j^T x)\right)$$



Related Work (Factorization Machines)

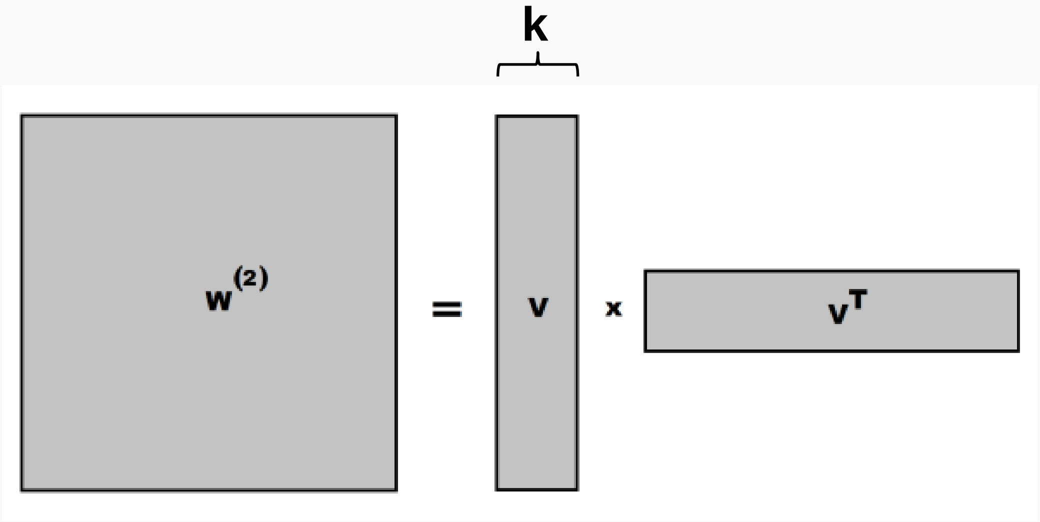
Factorization Machines

- Giant multi-hot vector
- Each categorical feature is called a **Field** → (n ≈ 20)
- Each category of a field is called a **Feature** → (f > 100k)

Field 1						Field 2				Field 3			Field 4							
0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0	0

Factorization Machines

$$\hat{y}(x) = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \underbrace{w'_{i,j}}\underbrace{x_i x_j}, \quad w'_{i,j} = \sum_{l=1}^{k} v_{i,l} v_{j,l},\ v_i \in \mathbb{R}^k$$



Cross features (interactions)

Related Work (Upgrades on Factorization Machines)

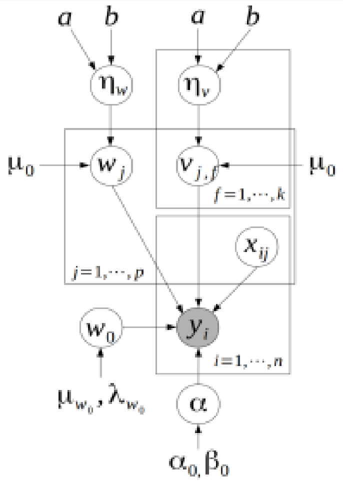
- **Field-Aware FM**

$$\hat{y}_{FFM}(x) = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} x_i x_j < v_{i,F_j}, v_{j,F_i} >$$

- **Field-Weighted FM (FFM)**

$$\hat{y}_{FwFM}(x) = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} x_i x_j < v_i, v_j > r_{F_i,F_j}$$

- **Bayesian FM**
  - Sparse FM



- **Attentional FM**

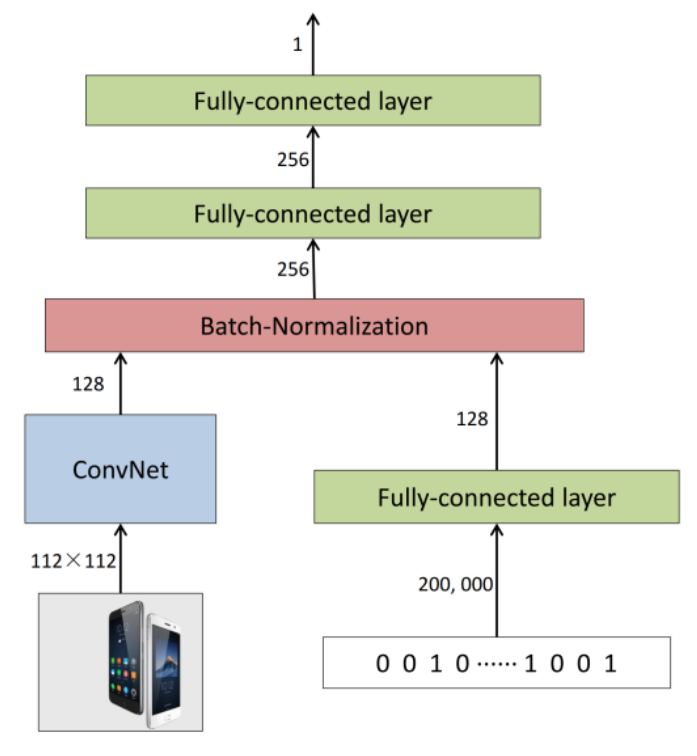
$$a_{i,j} = Softmax_{i,j}\{\mathbf{h}^T ReLU(\mathbf{W}\mathcal{E}_{i,j} + \mathbf{b})\}$$

$$\mathcal{E}_{i,j} = (v_i \odot v_j) x_i x_j$$

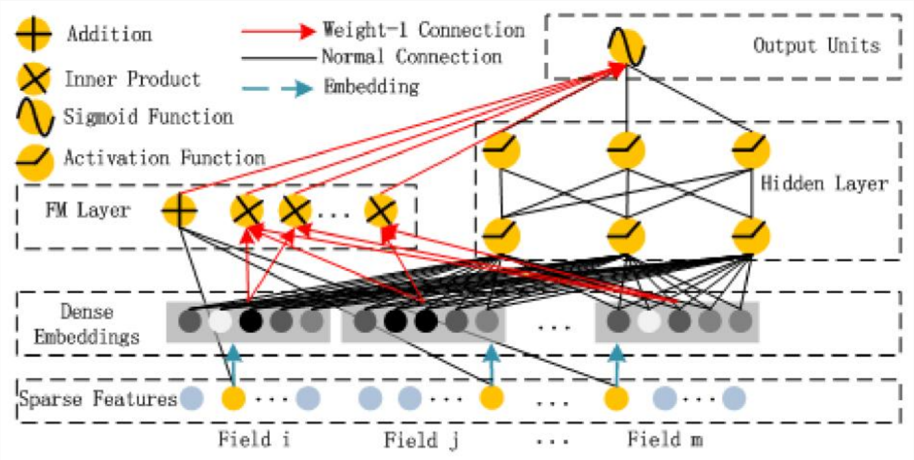
$$\hat{y}_{AFM}(x) = w_0 + \sum_{i=1}^{n} w_i x_i + \mathbf{P}^T \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} a_{i,j}\mathcal{E}_{i,j}$$

Related Work (Deep approaches)

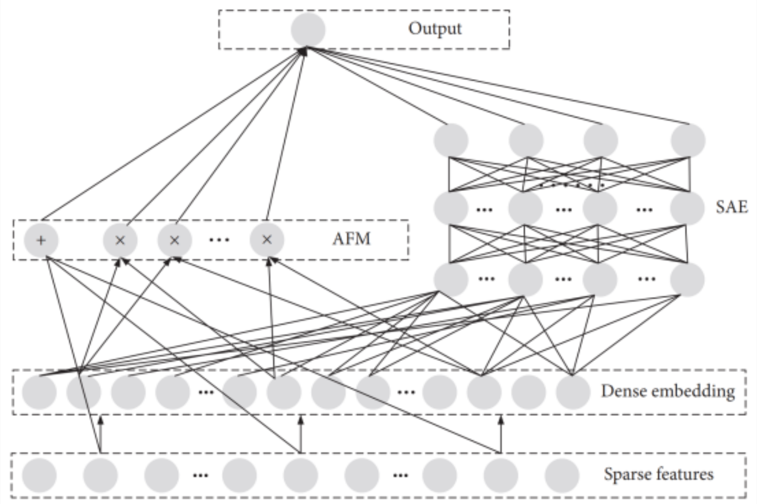
Deep CTR Prediction:



- **Deep FM (Wide & Deep)**



- **ASAE (AFM + SAE)**



Conclusion

Model	Weaknesses	Strengths
SVM (with polynomial kernel)	• Too many parameters	• Fast training
Piece-wise linear model	• Too many parameters • Slow training • Difficult to tune hyper-parameters	• High flexibility • Sparse Parameters • Good interpretability
Factorization Machine	• Low complexity • Low interpretability	• Works well with sparse data
Field-Aware FM	• Too many parameters • Prone to overfitting	• Modeling differences between fields
Field-Weighted FM	• Low complexity	• Modeling differences between fields • Less parameters
Bayesian FM	• Intractable inference • Wrong gaussian assumption	• Balance of exploration and exploitation
Sparse FM	• Intractable inference • Approximate Laplace distribution	• More interpretability • More sparsity

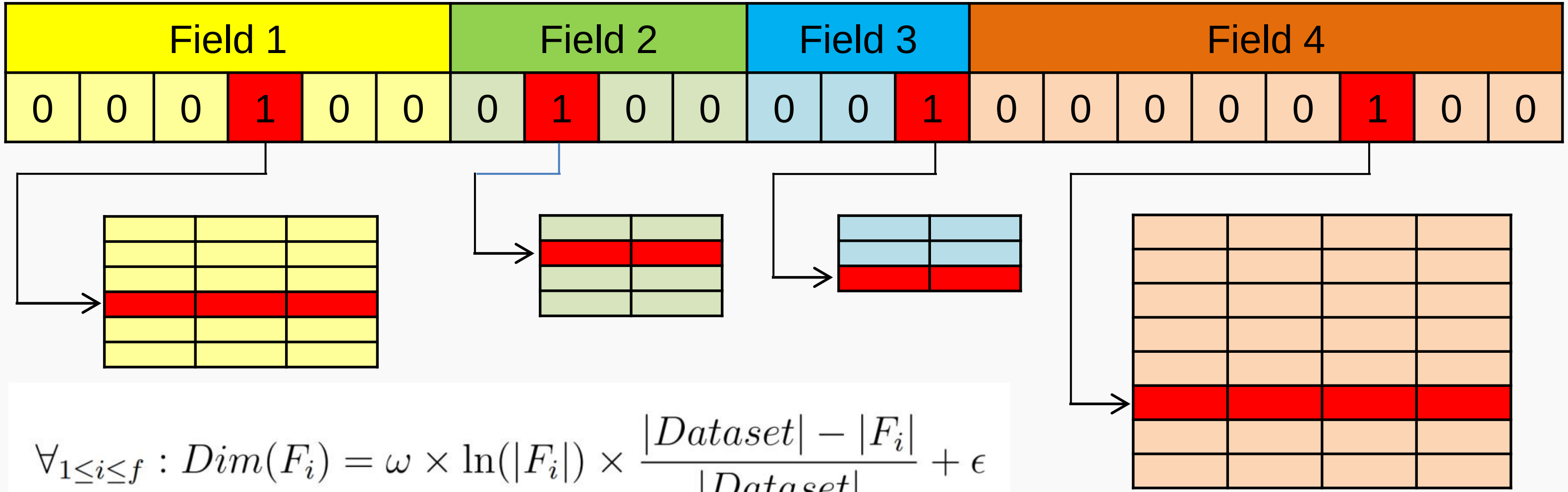
Conclusion

Model	Weaknesses	Strengths
Attentional FM	• Possible overfitting • Need to regularize	• More complexity • More interpretability
Deep CTR prediction model	• Need for Ad image • Possible overfitting	• Models higher degree interactions • More generalization when provided enough data
Deep FM	• Too many hyper-parameters • Low interpretability	• Models higher degree interactions • No bias in high or low degree interactions
Wide and Deep	• Needs feature engineering • Too many parameters	• Fast implementation • More complexity
ASAE	• Too many parameters • Possible overfitting	• More complexity • More parameter sharing

Proposed Method

Proposed Method

Different embedding sizes for different fields



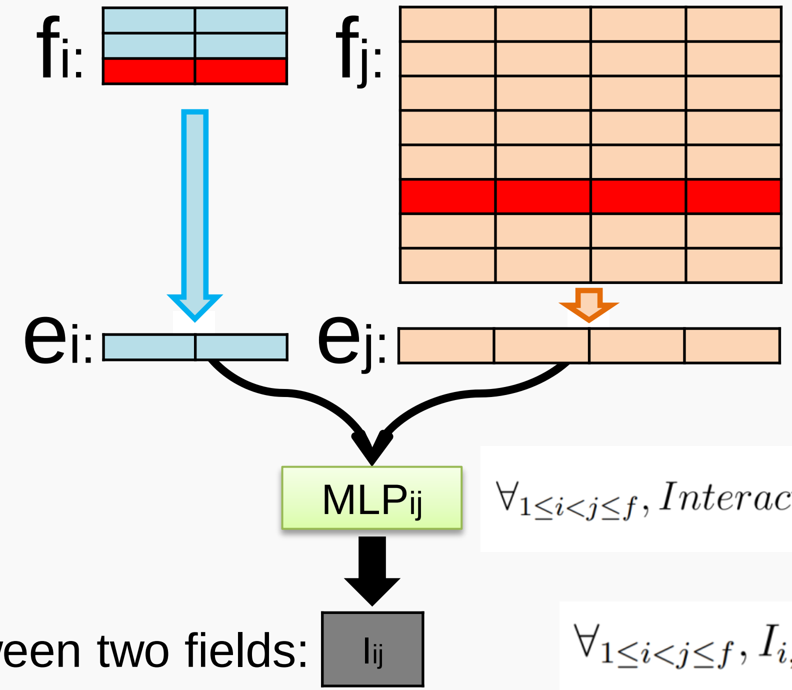
$$\forall_{1 \leq i \leq f} : Dim(F_i) = \omega \times \ln(|F_i|) \times \frac{|Dataset| - |F_i|}{|Dataset|} + \epsilon$$

$$\forall_{1 \leq i \leq f} : \mathbf{E}_i \in \mathbb{R}^{|F_i| \times Dim(|F_i|)}$$

$$\forall_{1 \leq i \leq f} : e_i = \mathbf{E}_i^{x_i} \in \mathbb{R}^{Dim(|F_i|)}$$

Proposed Method

Compute interactions via neural networks

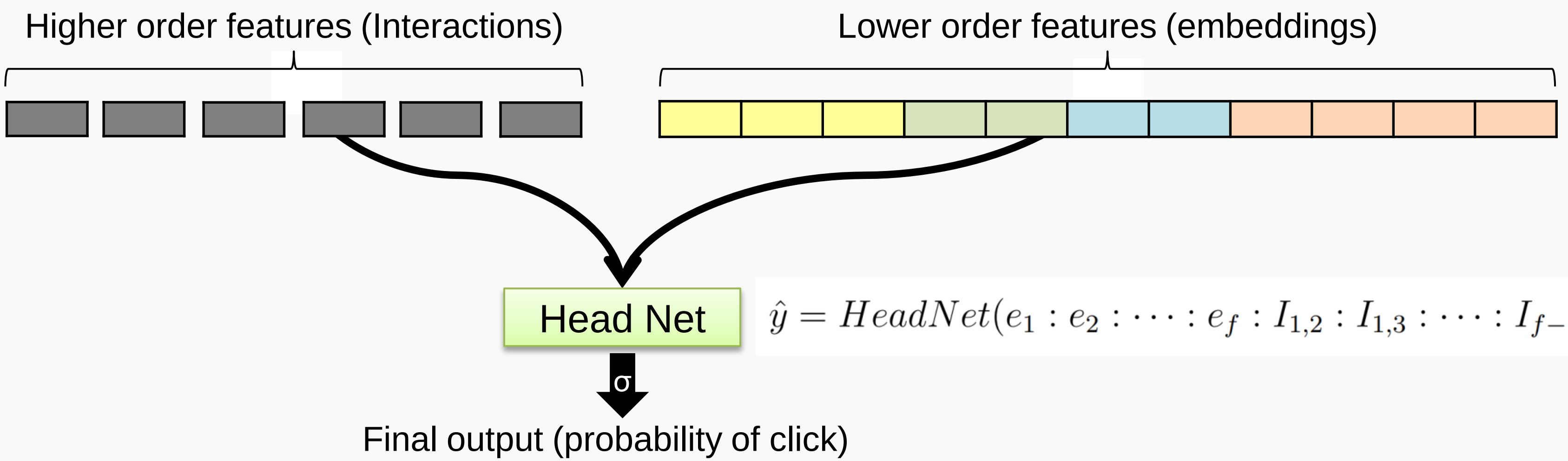


$$\forall_{1 \leq i < j \leq f},\, InteractionNet_{i,j} : \mathbb{R}^{Dim(|F_i|)+Dim(|F_j|)} \rightarrow \mathbb{R}^{Dim_{Int}}$$

Interaction between two fields:

$$\forall_{1 \leq i < j \leq f},\, I_{i,j} = InteractionNet_{i,j}(e_i : e_j)$$

Combine interactions and first order features using another neural network

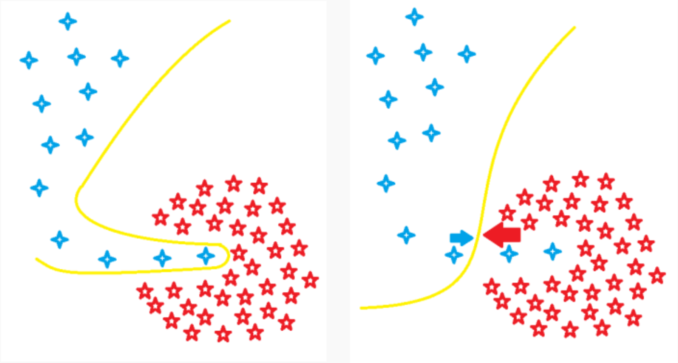


$$\hat{y} = HeadNet(e_1 : e_2 : \cdots : e_f : I_{1,2} : I_{1,3} : \cdots : I_{f-1,f})$$

Loss function:

- **Binary crossentropy**
- **Weight each class by reversed ratio of their samples**

$$LogLoss(y,\hat{y}) = -y\log(\hat{y})W_{Click} - (1-y)\log(1-\hat{y})(1-W_{Click})$$



Proposed Method

	Better data modeling	Class imbalance	High dimensionality	Cold start	Training speed	Testing speed
Variant-dimensional embeddings			Reduce unnecessary parameters – Avoid overfitting	Effective embeddings can help reduce the cold start problem	Fewer parameters easier the learning process	Less computation, faster prediction
Use MLP to compute interactions	Extract more complex interactions			Take advantage of dense embedding spaces		Fast implementations for MLPs exist
Multi-dimensional interactions	Interactions can have more valuable information					
Combine interactions and embeddings	Both lower order and higher order features can help the classification			Without higher order features, lower order features can help	Gradient from different paths speed up learning process of the embeddings	
Use class weighting		Consistent decision boundary and deny majority class bias				

Experiments

Datasets

Outbrain dataset

- +87M records
- Unbalanced (19% clicked)
- +40 fields (+100M features)

Outbrain preprocessed

- 87M records
- Unbalanced (19% clicked)
- 12 fields (~32K features)

Datasets

Criteo dataset

- +45M records
- Unbalanced (26% clicked)
- 26 categorical fields (+33M features)
- 12 integer features

Criteo-22

- +45M records
- Unbalanced (26% clicked)
- 22 categorrical fields (~2.7M features)

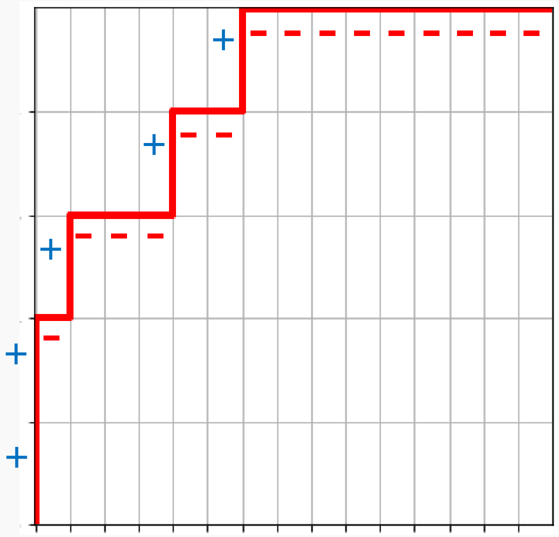
Criteo-21

- +45M records
- Unbalanced (26% clicked)
- 21 categorrical fields (~569K features)

Criteo-20

- +45M records
- Unbalanced (26% clicked)
- 20 categorrical fields (~283K features)

- Precision, Recall, F1
- Area Under ROC Curve (AUC)



Ranking!

Regularization techniques

Dropout

- Embedding parameters
- Interaction network parameters
- Head network parameters

L2-Regularization

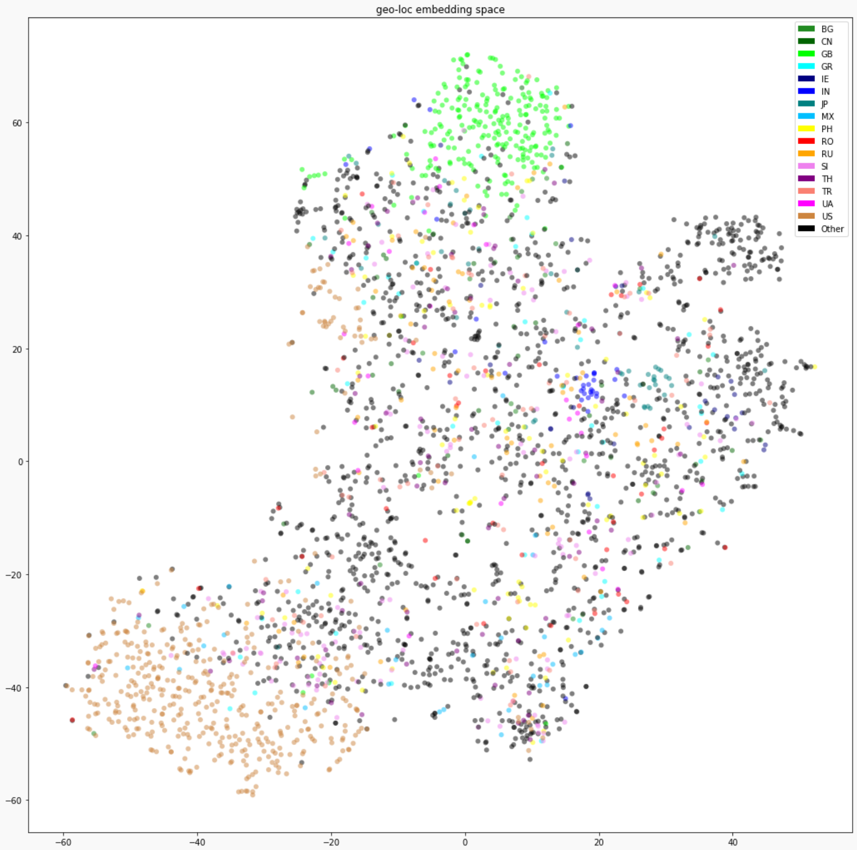
- Embedding parameters
- Interaction network parameters
- Head network parameters

Interpretation of embedding spaces

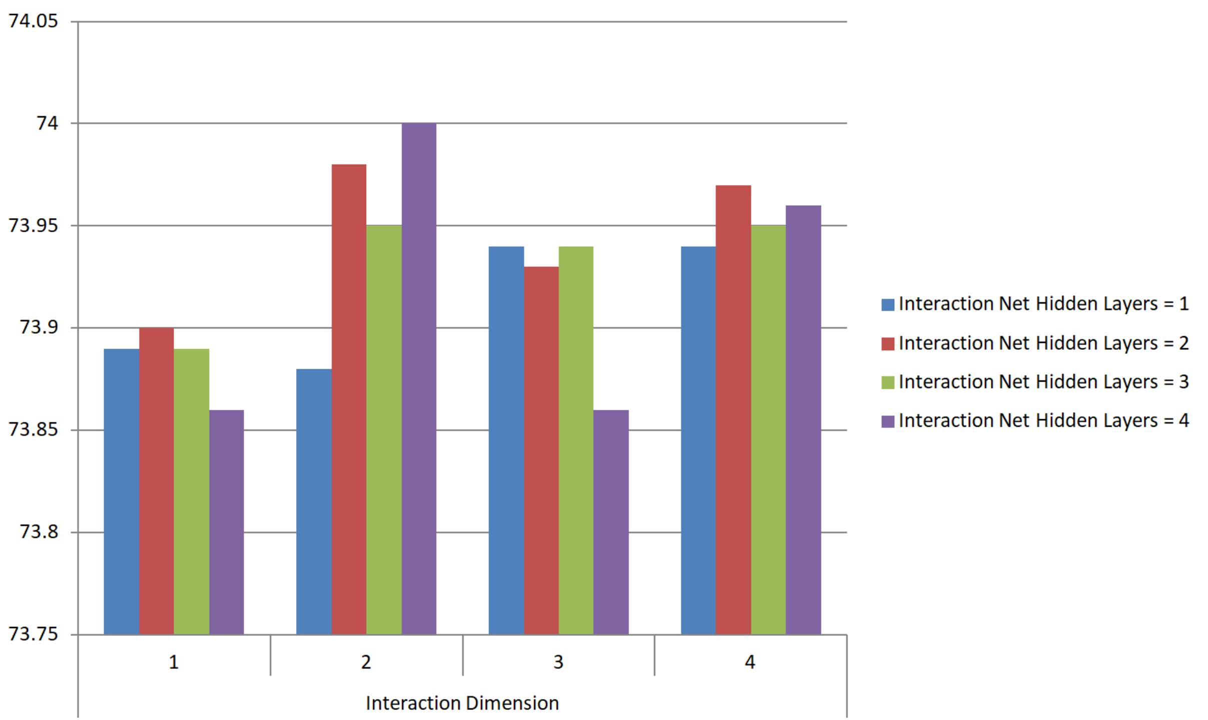
Outbrain dataset

(A close look at the embedding spaces)

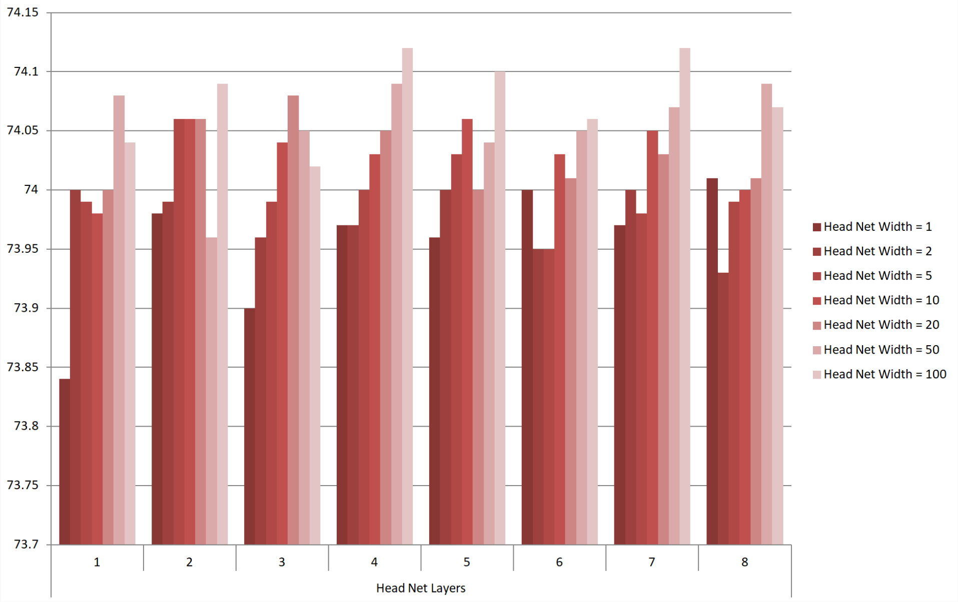
(Used T-SNE algorithm for visualization)



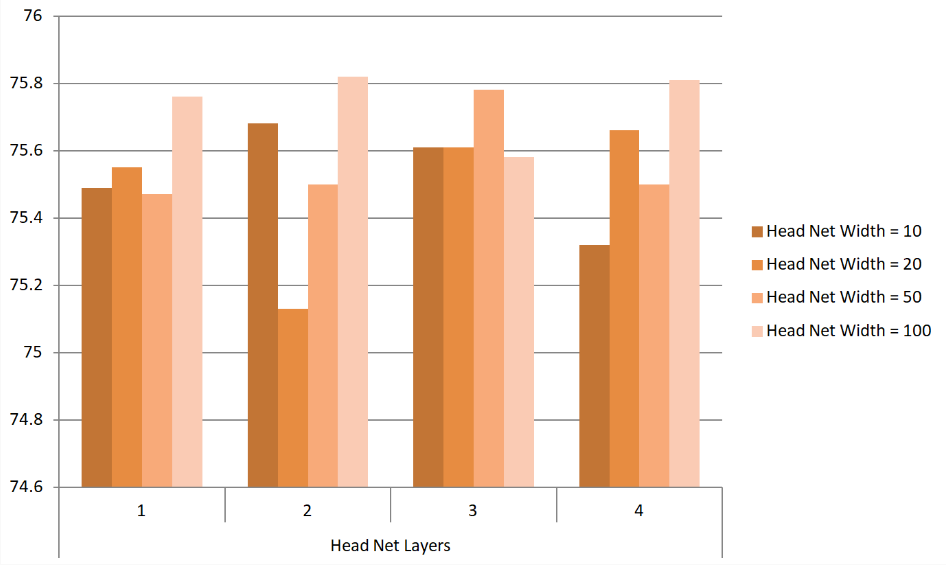
Interaction dimension



Head Network layers and # of neurons per layer (Width)

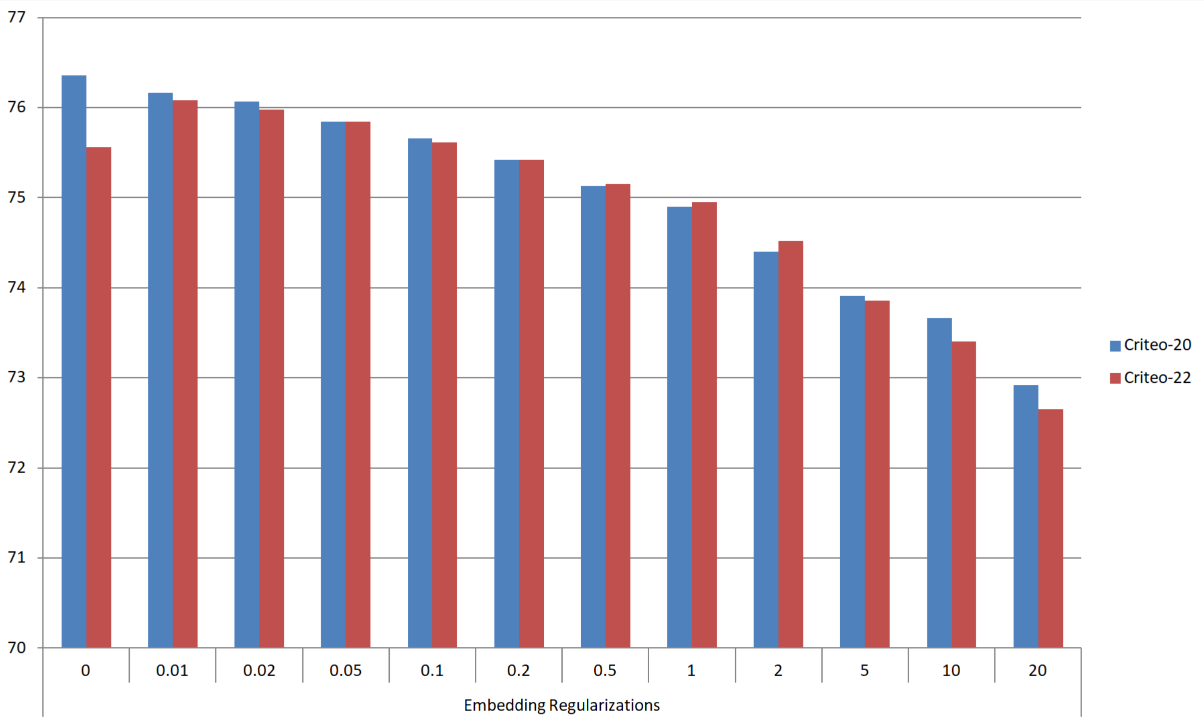


Outbrain



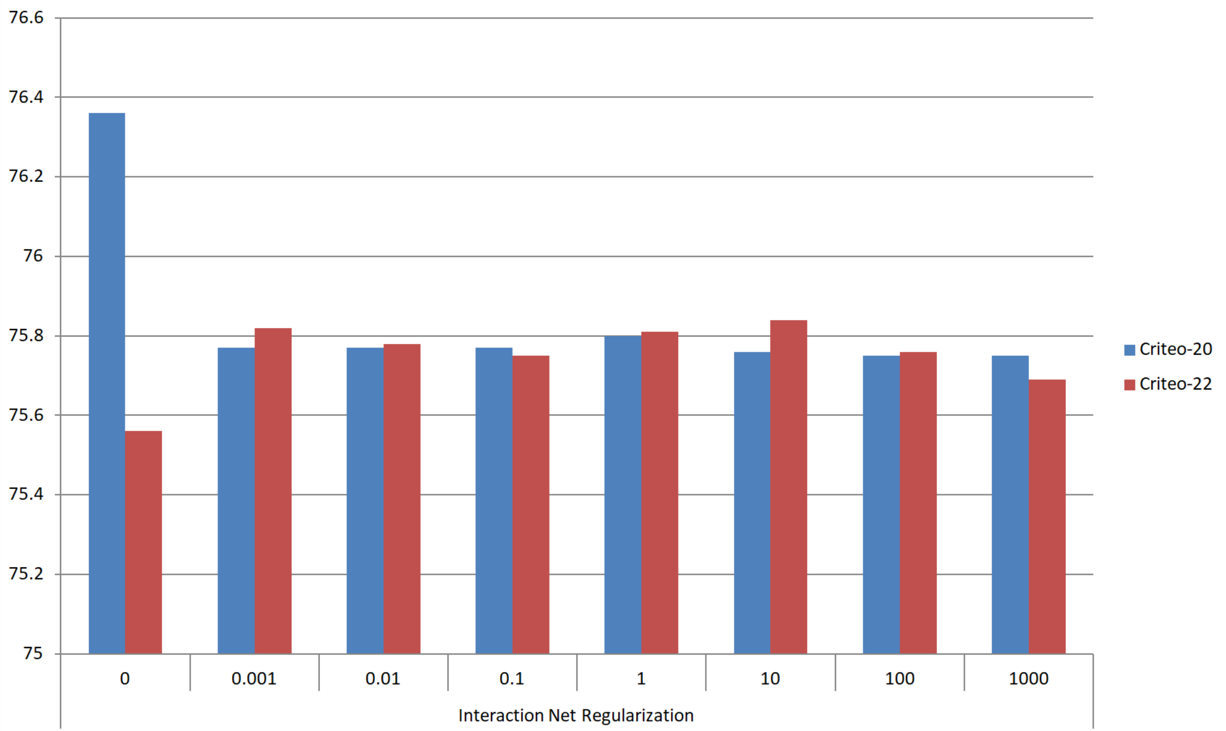
Criteo

L2-Reg on embedding parameters



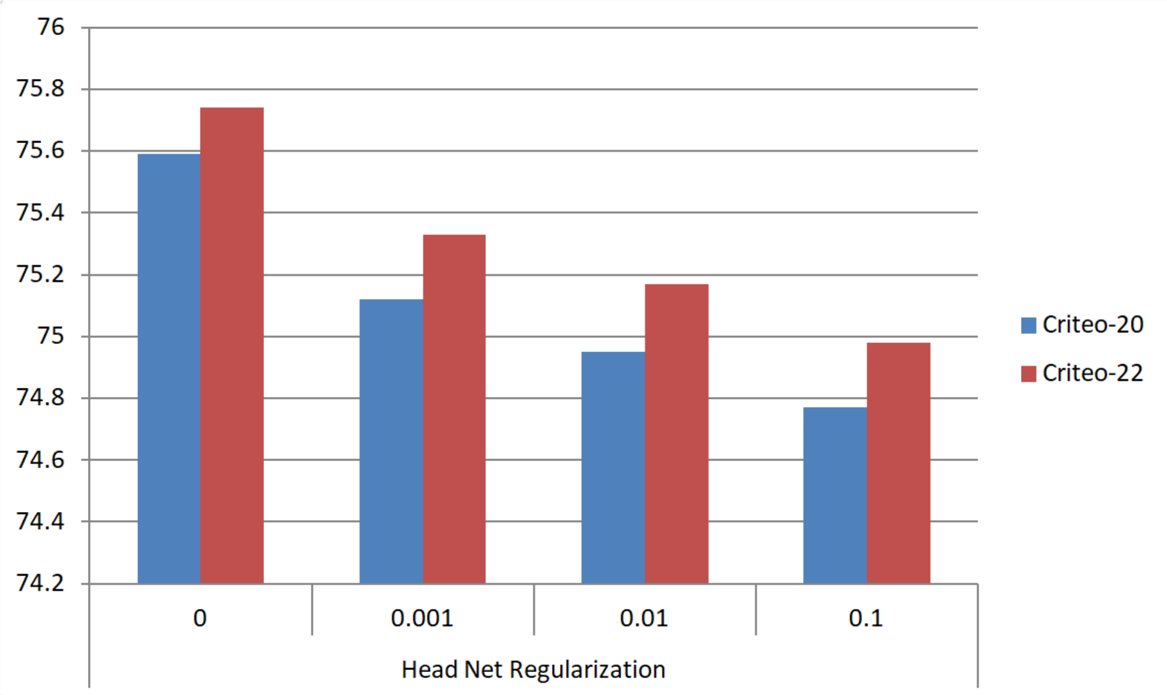
Hyper-parameter research

L2-Reg on interaction network parameters

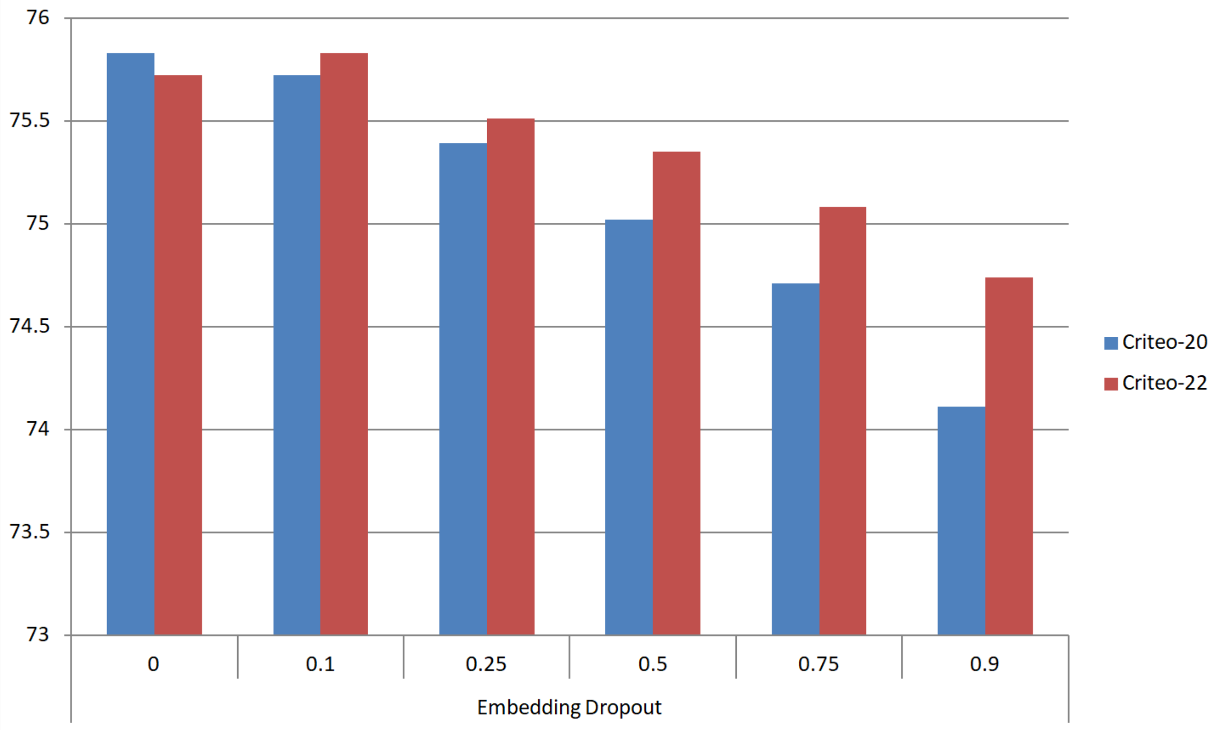


Hyper-parameter research

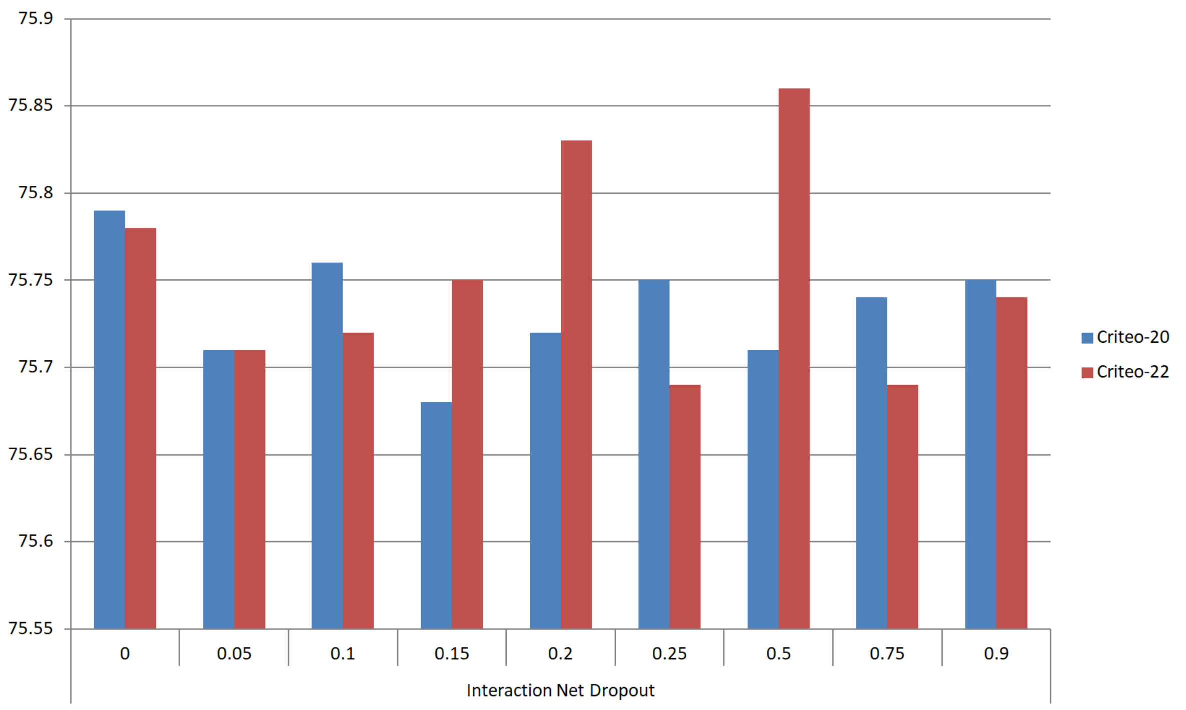
L2-Reg on head network parameters



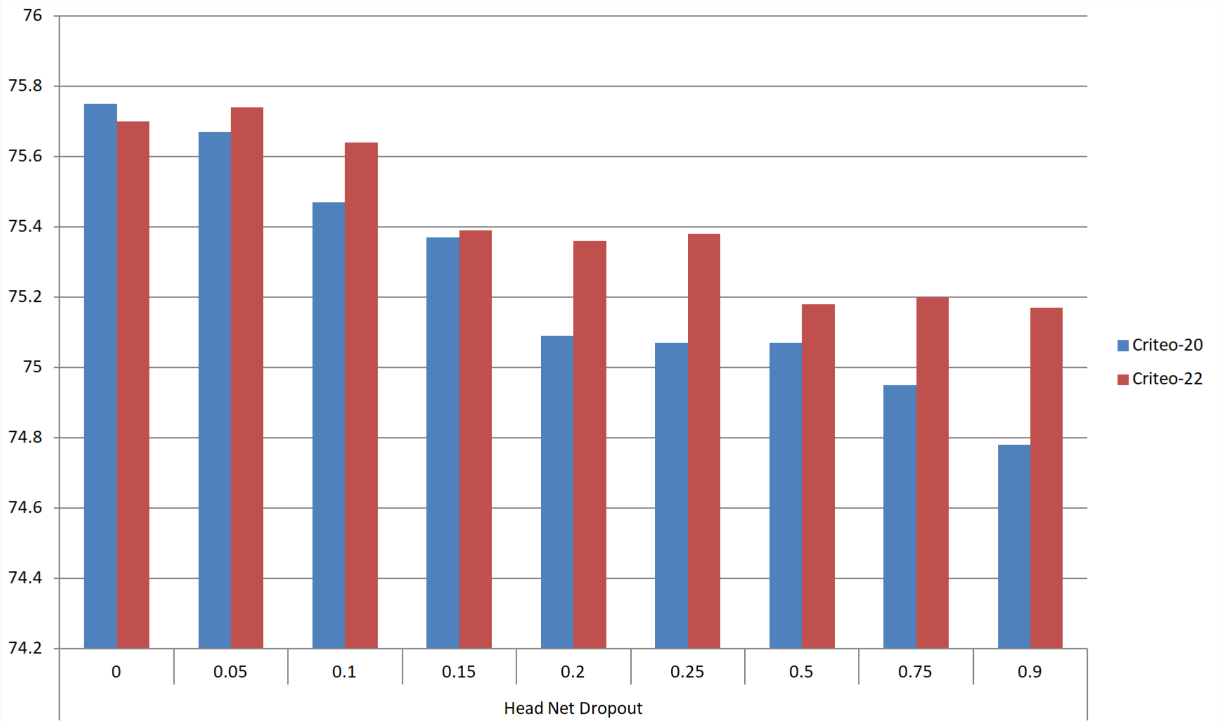
Dropout on embedding parameters



Dropout on interaction network parameters



Dropout on head network parameters



Results

Outbrain preprocessed dataset

Method	AUC (%)
FM (k=9)	**74.22**
DeepFM (k = 10, Deep layers = [20, 20, 20])	72.27
DeepFM (k = 10, Deep layers = [100, 100, 100])	73.00
DeepFM (k = 10, Deep layers = [400, 400, 400])	73.44
Proposed	**74.13**

Criteo-22 dataset

Method	AUC (%)	Precision	Recall	F1
FM (k=5)	75.41	0.5655	0.3458	0.4292
FM (k=10)	74.75	0.5489	0.3542	0.4306
FM (k=40)	72.38	0.5012	0.3720	0.4270
FM (k=100)	70.30	0.4692	0.3832	0.4219
Proposed	**76.08**	0.4307	0.7039	**0.5344**

Criteo-21 dataset

Method	AUC (%)	Precision	Recall	F1
FM (k=5)	75.83	0.5877	0.3173	0.4121
FM (k=10)	75.49	0.5775	0.3249	0.4159
FM (k=40)	73.68	0.5360	0.3440	0.4191
FM (k=100)	71.71	0.5014	0.3508	0.4128
DeepFM (k = 10, Deep layers = [20, 20, 20])	74.85	0.3271	0.9181	0.4823
DeepFM (k = 10, Deep layers = [100, 100, 100])	76.01	0.3816	0.8251	0.5218
DeepFM (k = 10, Deep layers = [400, 400, 400])	76.24	0.4221	0.7334	0.5358
Proposed	**76.70**	0.4370	0.6994	**0.5379**

Results

Criteo-20 dataset

Method	AUC (%)	Precision	Recall	F1
FM (k=5)	75.57	0.5920	0.3035	0.4012
FM (k=10)	75.30	0.5822	0.3113	0.4056
FM (k=40)	73.62	0.5424	0.3293	0.4098
FM (k=100)	71.75	0.5062	0.3432	0.4090
DeepFM (k = 10, Deep layers = [20, 20, 20])	74.70	0.4285	0.6645	0.5210
DeepFM (k = 10, Deep layers = [100, 100, 100])	75.44	0.5594	0.3206	0.4076
DeepFM (k = 10, Deep layers = [400, 400, 400])	75.45	0.3364	0.9063	0.4907
Proposed	**76.37**	0.4276	0.6861	**0.5344**

Future Work

Future Work

Further work:

- Find a way for the model to explore at uncertain conditions and exploit at confident conditions (exploration / exploitation tradeoff)
- Deploy and evaluate proposed method in online settings
- Provide stable and fast implementation

Thank You!

Any Questions?

Refs

[1] B. E. Boser, I. M. Guyon, and V. N. Vapnik, “A training algorithm for optimal margin classifers,” in *Proceedings of the 5th Annual Workshop on Computational Learning Theory (COLT’92)*, D. Haussler, Ed. Pittsburgh, PA, USA: ACM Press, July 1992, pp. 144–152.
[2] K. Gai, X. Zhu, H. Li, K. Liu, and Z. Wang, “Learning piece-wise linear models from large scale data for ad click prediction.” *CoRR*, vol. abs/1704.05194, 2017.
[3] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *roceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
[4] Y. Xiao, Z. Wei, and Z. Wang, “A limited memory bfgs-type method for large-scale unconstrained optimization.” *Comput. Math. Appl.*, vol. 56, no. 4, pp. 1001–1009, 2008.
[5] T. Graepel, J. Q. Candela, T. Borchert, and R. Herbrich, “Web-scale bayesian click-through rate prediction for sponsored search advertising in microsoft’s bing search engine.” in *ICML*, J. Fürnkranz and T. Joachims, Eds. Omnipress, 2010, pp. 13–20.
[6] S. Rendle, “Factorization Machines,” in *Proceedings of the 2010 IEEE International Conference on Data Mining*, ser. ICDM ’10. IEEE, Dec. 2010, pp. 995–1000.
[7] Y.-C. Juan, Y. Zhuang, W.-S. Chin, and C.-J. Lin, “Field-aware factorization machines for ctr prediction.” in *RecSys*, S. Sen, W. Geyer, J. Freyne, and P. Castells, Eds. ACM, 2016, pp. 43–50.
[8] Y. Juan, D. Lefortier, and O. Chapelle, “Field-aware factorization machines in a real-world online advertising system.” *CoRR*, vol. abs/1701.04099, 2017.
[9] J. Pan, J. Xu, A. L. Ruiz, W. Zhao, S. Pan, Y. Sun, and Q. Lu, “Field-weighted factorization machines for click-through rate prediction in display advertising.” *CoRR*, vol. abs/1806.03514, 2018.
[10] S.-T. L. Freudenthaler, C. and S. Rendle, “Bayesian factorization machines,” in *In Proceedings of the NIPS Workshop on Sparse Representation and Lowrank Approximation*, 2011.

Refs

[11] Z. Pan, E. Chen, Q. Liu, T. Xu, H. Ma, and H. Lin, “Sparse factorization machines for click-through rate prediction.” in *ICDM*, F. Bonchi, J. Domingo-Ferrer, R. Baeza-Yates, Z.-H. Zhou, and X. Wu, Eds. IEEE Computer Society, 2016, pp. 400–409.
[12] J. Xiao, H. Ye, X. He, H. Zhang, F. Wu, and T.-S. Chua, “Attentional factorization machines: Learning the weight of feature interactions via attention networks.” *CoRR*, vol. abs/1708.04617, 2017.
[13] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A Simple Way to Prevent Neural Networks from Overftting,” *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014.
[14] A. N. Tikhonov, “On the stability of inverse problems,” in *Dokl. Akad. Nauk SSSR*, vol. 39, 1943, pp. 195–198
[15] J. Chen, B. Sun, H. Li, H. Lu, and X.-S. Hua, “Deep ctr prediction in display advertising.” in *ACM Multimedia*, A. Hanjalic, C. Snoek, M. Worring, D. C. A. Bulterman, B. Huet, A. Kelliher, Y. Kompatsiaris, and J. Li, Eds. ACM, 2016, pp. 811–820.
[16] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” 2015, cite arxiv:1512.03385Comment: Tech report.
[17] V. Nair and G. E. Hinton, “Rectifed linear units improve restricted boltzmann machines.” in *ICML*, J. Fürnkranz and T. Joachims, Eds. Omnipress, 2010, pp. 807–814.
[18] C. Guo and F. Berkhahn, “Entity embeddings of categorical variables.” *CoRR*, vol. abs/1604.06737, 2016.
[19] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” 2015, cite arxiv:1502.03167.
[20] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, “Deepfm: A factorization-machine based neural network for ctr prediction.” in *IJCAI*, C. Sierra, Ed. ijcai.org, 2017, pp. 1725–1731.
[21] H. Guo, R. Tang, Y. Ye, Z. Li, X. He, and Z. Dong, “Deepfm: An end-to-end wide and deep learning framework for ctr prediction.” *CoRR*, vol. abs/1804.04950, 2018.

Refs

[22] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, R. Anil, Z. Haque, L. Hong, V. Jain, X. Liu, and H. Shah, “Wide and deep learning for recommender systems.” in *DLRS@RecSys*, A. Karatzoglou, B. Hidasi, D. Tikk, O. S. Shalom, H. Roitman, B. Shapira, and L. Rokach, Eds. ACM, 2016, pp. 7–10.

[23] Q. Wang, F. Liu, S. Xing, and X. Zhao, “A new approach for advertising ctr prediction based on deep neural network via attention mechanism.” *Comput. Math. Methods Medicine*, vol. 2018, pp. 8 056 541:1–8 056 541:11, 2018.

[24] D. H. Ballard, “Modular learning in neural networks.” in *AAAI*, K. D. Forbus and H. E. Shrobe, Eds. Morgan Kaufmann, 1987, pp. 279–284.

[25] C. E. Shannon and W. Weaver, *The Mathematical Theory of Communication*. Urbana and Chicago: University of Illinois Press, 1949.

[26] M. Naumov, “On the dimensionality of embeddings for sparse features and data.” *CoRR*, vol. abs/1901.02103, 2019.

[27] A. Ginart, M. Naumov, D. Mudigere, J. Yang, and J. Zou, “Mixed dimension embeddings with application to memory-efcient recommendation systems.” *CoRR*, vol. abs/1909.11810, 2019.

[28] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, “Neural collaborative fltering,” in *Proceedings of the 26th International Conference on World Wide Web*, ser. WWW ’17. Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee, 2017, p. 173–182.

[29] A. L. Maas, A. Y. Hannun, and A. Y. Ng, “Rectifer nonlinearities improve neural network acoustic models,” in *Proc. icml*, vol. 30, no. 1. Citeseer, 2013, p. 3.

[30] L. van der Maaten and G. Hinton, “Visualizing data using t-SNE,” *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.