Agile and Accurate CTR Prediction Model Training for Massive-Scale Online Advertising Systems

Zhiqiang Xu[1], Dong Li[2], Weijie Zhao[1], Xing Shen[2], Tianbo Huang[2], Xiaoyun Li[1], Ping Li[1]
1 Cognitive Computing Lab, Baidu Research
2 Baidu Search Ads (Phoenix Nest)
No. 10 Xibeiwang East Road, Beijing 100193, China
10900 NE 8th St. Bellevue, Washington 98004, USA
{xuzhiqiang04, lidong06, weijiezhao, shenxing01, huangtianbo, v_lixiaoyun02, liping11}baidu.com

ABSTRACT

Deep neural network has been adopted as the standard model to predict ads click-through rate (CTR) for commercial online advertising systems. Deploying an industrial scale ads system requires to overcome numerous challenges, e.g., hundreds or thousands of billions of input features and also hundreds of billions of training samples, which under the cost budget can cause fundamental issues on storage, communication, or the model training speed. In this work, we present Baidu's industrial-scale practices on how to apply the system and machine learning techniques to address these issues and increase the revenue. In particular, we focus on the strategy for developing GPU-based CTR models combined with quantization techniques to build a compact and agile system which noticeably improves the revenue. With quantization, we are able to effectively increase the model (embedding layer) size without increasing the storage cost. This brings an increase in prediction accuracy and yields a 1% revenue increase and 1.8% higher relative click-through rate in the real sponsored search production environment.

CCS CONCEPTS

• **Information systems → Sponsored search advertising**.

ACM Reference Format:
Zhiqiang Xu, Dong Li, Weijie Zhao, Xing Shen, Tianbo Huang, Xiaoyun Li, and Ping Li. 2021. Agile and Accurate CTR Prediction Model Training for Massive-Scale Online Advertising Systems. In *Proceedings of the 2021 International Conference on Management of Data (SIGMOD '21), June 20–25, 2021, Virtual Event, China.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3448016.3457236

1 INTRODUCTION

Advertising technologies in Baidu Search Ads (a.k.a. "Phoenix Nest") have become fairly matured after about two decades of dedicated development since the advent of www.baidu.com. Perhaps surprisingly, there were very few publications on advertising methodologies from Baidu Search Ads until recently. In MLSys'20, [49] provided a summary of the history of Baidu's efforts in developing new algorithms/frameworks for advertising. As early as 2010, Baidu already adopted the distributed logistic regression (LR) CTR (click-through rate) model and the distributed parameter server. In 2013, Baidu Search Ads deployed, as the standard practice in production ever since, the MPI-based solution for training **massive-scale deep learning CTR** models. Since 2017, there have been two major directions for further improvements. One direction is the use of recent advancements on approximate near neighbor search (ANNS) and maximum inner product search (MIPS) [9, 39, 41, 48, 53] to improve the quality of ads recalls in the early stage of the pipeline of the advertising system. The other direction is to develop GPU-based ads systems [49, 50], internally known as "PaddleBox" (www.paddlepaddle.org.cn). Other examples of engineering efforts in Baidu Search Ads include image advertising [45], video advertising [46], sample optimization [10], reinforcement learning [25], etc.

In this paper, we focus on presenting Baidu's recent development by taking advantage of the GPU-based CTR model compression via quantization to improve the prediction accuracy, which has subsequently lead to a noticeable increase in revenue. Basically, under the constraint of model storage size, our compact and agile system enables engineers to effectively increase the model/embedding dimensionality without incurring additional storage cost.

1.1 Online Advertising and Challenges

Online advertising [7, 42] is the most popular way for advertisers to attract users' attention to the ads, e.g., products for sales promotion, in an increasingly digital world nowadays. How to accurately predict ads CTR [2, 8, 9, 14, 18, 35, 38], i.e., the probability that a user clicks an ad, is a long-standing core problem for online advertising, as the revenue is largely dictated by probabilistic clicks. The drop of merely one-tenth of a percentage point in the CTR prediction accuracy would typically lead to a significant loss in revenue.

Deep neural network, by virtue of its remarkable modeling capability, has been rising as a standard model in the online advertising industry [15, 17, 21, 30, 31, 49, 50, 52]. In this work, we present Baidu's practices on how to apply system and machine learning techniques to address many fundamental issues and boost the prediction accuracy. Our deep CTR model consists of the input layer, embedding layer (holding embedding parameters), five fully connected layers (holding neural network parameters), and output layer. It is non-trivial to deploy the model on an industrial scale, e.g., hundreds to thousands of billions of input features, which under the cost budget can cause fundamental issues on storage, communication, or the model training speed. In addition to the

SIGMOD '21, June 20–25, 2021, Virtual Event, China

ACM ISBN 978-1-4503-8343-1/21/06…$15.00
https://doi.org/10.1145/3448016.3457236

extremely high-dimensional input feature vector, great challenges of the industrial-scale CTR model training lie in training data of petabyte scale (e.g., hundreds of billions of training instances) and model size of over ten terabytes. This entire model size, dominated by the size of the embedding layer, characterizes the extraordinary model capacity of the deep neural network, and is necessary to accommodate the extremely high dimensionality of input feature vectors. This gives rise to unprecedented pressure on the storage consumption in main memories. Also, engineers have to rapidly retrain models in order to capture the feature dynamics.

Until recently, commonly deployed industrial-scale CTR prediction systems were built upon the distributed training system with CPU clusters, for example, MPI (message passing interface). They consume a large amount of communication and computation costs to remain at a high degree of fault-tolerance and synchronization, which in turn means substantial costs for cluster maintenance and energy consumption. Every aspect above could directly affect the model training speed. Particularly, it is unacceptable to trade an even 0.1% decrease of the prediction accuracy for alleviating these issues. This type of revenue-driven practice is indeed very different from typical academic research, for example, the numerous publications on down-sampling or hashing [3, 19, 20, 23, 23, 27–29, 34, 36, 40, 43, 47, 51]. See the hashing experiments (back in 2015) for CTR reported in [49]. In our experience, hashing tricks can be very helpful in certain (important) scenarios. But if the goal is to maximize the machine learning accuracy (and revenue), one typically cannot count on (only) hashing to accomplish the goal.

1.2 Our Approaches

The goal of our system is to overcome the above issues and challenges, with increased prediction accuracy. Adapted from [49, 50], a GPU-based single computing node is used to build a compact and agile deep CTR model training system. This design simply erases our concerns on communication and synchronization costs by MPI clusters and greatly cut the expenses of the computer cluster maintenance and energy consumption. The storage pressure from the embedding table of size exceeding 10TB are addressed by three levels of hardware structure, i.e., SSDs (solid state drives), main memory, and GPU memories. Different from the existing system which distributes the training jobs to both CPUs and GPUs, all the training jobs are distributed only over multiple GPUs, which further reduces the system complexity.

The quantization step is crucial in our system. Despite a recent surge of research interest in quantized deep learning [4, 13, 16, 26, 32, 33, 37, 47], especially research on 1-bit and 2-bit quantization [5, 22, 24], our industrial practice indicates that even 12-bit quantized deep CTR model could lead to an unacceptable drop of both prediction accuracy and revenue. After numerous engineering endeavours, we eventually find that a 16-bit quantization on the embedding layer of double size brings significant increase of AUC without extra technical implementation cost, which accordingly yields 1% revenue increase and 1.8% higher relative click-through rate in the real sponsored search production environment.

There have been a lot of studies on low-precision training for fully connected layers. In our system, the extremely high-dimensional and dense embedding layer is highly sparsely connected to a "small" deep net (e.g., only thousands of internal nodes at each layer). This means that the size of embedding layer dominates the size of the system. Therefore, we choose to quantize the embedding layer. Quantized embedding layer gives us a substantial storage saving, which enables us to trade the saved storage for higher embedding dimensions and thus further improve the prediction accuracy of the CTR model without changing the network structure. Our experimental studies verify the effectiveness of the quantization scheme on the embedding layer of double size in the productions.

2 TRAINING FRAMEWORK FOR ADS SYSTEM

Our framework must be capable of training a massive network (more than 10 TB parameters) in a timely manner. Our GPU computing node is adapted from [6, 49, 50], equipped with 8 GPUs and SSDs.

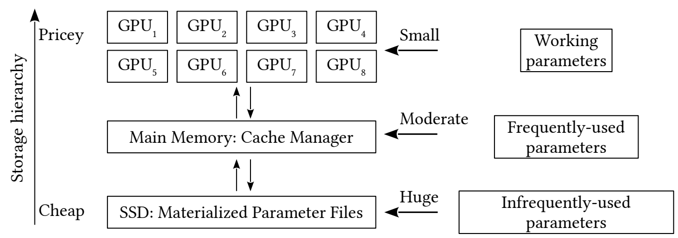

Figure 1: GPU computing node architecture.

Figure 1 depicts the architecture of the GPU computing node of our system. The working parameters, referenced in the current processing batches, are partitioned and stored in the GPU High-Bandwidth Memory (HBM). Collective inter-GPU communications (e.g., AllReduce) are performed to synchronize the updates across the entire cluster. Besides, the main memory of the system acts as a cache manager maintaining the frequently-used parameters. The cache manager also maintains the out-of-the-memory parameters: they are materialized and stored as parameter files in the SSD. Compared to a multi-node system, advantages of our one-node system with multi-GPU include: cheap price, low communication cost, much less synchronization, and low failure rate.

The major performance bottleneck of the training framework is that the excessive inter-GPU communications limit the training time. There are two principal research axes for reducing the communication overhead: (a) reduce number of model parameters; (b) trim the memory footprint of each parameter. We follow the second direction since the model compression techniques are lossy—we cannot tolerate even a small CTR prediction accuracy drop.

3 QUANTIZATION OF DEEP CTR MODELS

Quantization, as a matured field in signal processing [44], has recently attracted lots of attentions in machine learning. The article www.eetimes.com/an-introduction-to-different-rounding-algorithms/ provides a good summary of common quantization methods including "rounding in decimal", "round-toward-nearest", "round-random", etc. In our work, we adopt "round-random" which is referred to as "stochastic quantization (StocQ)". For the purpose of comparison, we also implemented "round-toward-nearest" which is referred to as "fixed quantization (FixQ)". As we will show, FixQ is clearly inferior to StocQ. We should mention that stochastic rounding is a truly ancient idea which dated back to the 1950s [1, 11, 12].

Specifically, we set a quantization range $[-w, w]$ and divide it into 2^b bins of equal length $\triangle = 2w/(2^b - 1)$, where b is the bit number, e.g., $b = 8, 16$ for int8 and int16, respectively. We consider fixed rounding (FixQ) and stochastic rounding (StocQ) for $x \in [-w, w]$. Fixed rounding rounds a value x to its nearest bin border: $Q_f(x) = i^*\triangle$, where $i^* = \lfloor \frac{x}{\Delta} + 0.5 \rfloor$ and $\lfloor a \rfloor$ gives the largest integer that is not greater than a. Stochastic rounding, uses the formula: $Q_s(x) = i^*\Delta$, where $i^* = \lfloor \frac{x}{\Delta} + \text{rand}() \rfloor$ and rand() returns a uniform number between 0 and 1. Second, if $x \notin [-w, w]$, we adopt the standard cut-off strategy to quantize x to its nearest border for both rounding schemes. Enlarging w would force the quantizer to consider lighter tails of parameters but would also enlarge the window size $\triangle$, leading to less accurate approximation to the true signal. In our experiments, we implemented different w to test the impact of quantizing range on generalization performance. Both methods only need to store i^* for each parameter resulting from the output of quantization, which is an (unsigned) int8 or int16. It typically saves the storage by a half or 75%, compared with using float32 data type. Given ultra large dimensionality, e.g., hundreds of billions, of the embedding layer, this storage saving is huge. For example, encoding embedding layers with data type int16 instead of float32 gives us at least a 5TB storage saving in our case.

For the j-th embedding parameter $x_{j,t}$ at iteration t, it is updated as $Q_s(x_{j,t} + \eta g_{j,t})$ with SGD, where η is the learning rate and $g_{j,t}$ represents mini-batch stochastic gradient at the current iteration.

4 EXPERIMENTS

We now show experimental results for evaluating the quantization effect of the CTR prediction model trained on the single computing node system, and also make some comparisons between the current one-node system and the CPU-only multi-node system.

4.1 Performance of Quantized CTR Model

Data. Baidu's user click history data collected are used as the training data of the model, where one data example includes the following information: user query, previous queries of the user, ads title, ads image, ads id, and so on. Note that our scenario is about the training of a super-large network of parameter size over ten terabytes with hundreds to thousands of billions of input features. Most of the public datasets are too small to showcase the true performance of the model in our setting.

Training and testing with online learning. Our experiments collected data for 7.5 days, where the CTR model is trained in an online one-pass fashion with the quantized stochastic gradient descent described in Section 3 on the one-node system that has server-grade CPUs, 8 cutting-edge GPUs, 1 TB of memory, and a RAID 0 with NVMe SSDs. We use the Area Under the Curve (AUC) to evaluate the prediction quality of the trained models online on the test data. Specifically, first, the model was evolving with the training process along the time line of 7.5 days. At each point of this time line, the data of the current day was regarded as the data of the last day relative to the past days. The data of the current day was first used to test the current model so that we got AUC values. After testing, this data was fed to the system to train the current model. Second, we used online learning (i.e., SGD) to train the model. Every batch of data samples was collected in 15 minutes, and they only pass the model once for training. Third, our model's parameters were updated every 15 minutes with online learning to adapt to the current data distribution.

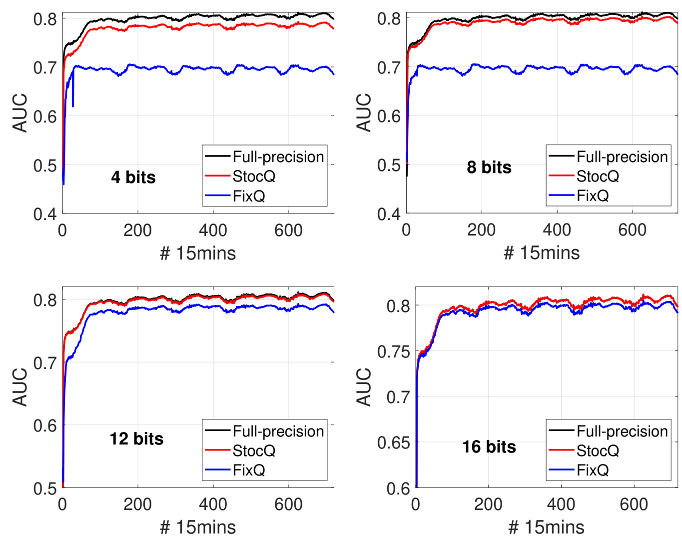

Figure 2: Prediction performance for fixed rounding (FixQ), stochastic rounding (StocQ), full precision (float32).

Figure 2 reports the quantization performance of the CTR model with full precision or fixed/stochastic rounding in 4, 8, 12, 16 bits, in terms of the online test AUC curve over 720×15 minutes on the data for the last day. We observe that for every chosen number of bits, fixed rounding exhibits the quantization bias which finally leads to performance degradation. For the fixed rounding in 16 bits, its AUC is much lower than the other two schemes. For example, there is a $0.6 \sim 0.7$ percent AUC drop. As we can see, the stochastic rounding in 16 bits shown in the lower right figure of Figure 2 has performed equally well compared to the full-precision counterpart. It is worth mentioning that we have used the optimal quantization range $[-1, 1]$, i.e., $w = 1$, in the above experiments.

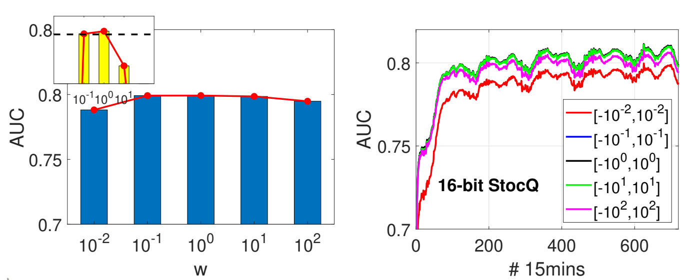

Figure 3: Prediction performance with different quantization ranges, under 16-bit stochastic rounding.

Influence of Quantization Range. The quantization range is another critical factor to improve the prediction accuracy that can be set flexibly in our model. We next show more details on the influence of the quantization range. We demonstrate in Figure 3 the prediction performance of the CTR model with varying quantization ranges, i.e., $w = 10^{-2}, 10^{-1}, 10^0, 10^1, 10^2$. The figure on the left shows that $[-10^{-2}, 10^{-2}]$ is the worst-performing range here, because it can't cover most of the embedding parameters. The performance is increasing with $w \leq 1$, in large part, by the greater coverage, and decreasing when $w \geq 1$ mainly because of the coarse quantization window size Δ. The optimal performance is attained with $[-1, 1]$ (see the zoomed area, which clearly shows the slight AUC difference between two values of the range parameter w, and also the plot on the left in Figure 4), indicating a good balance between the distribution coverage of embedding parameters and the quantization window size $\triangle$ as discussed in Section 3.

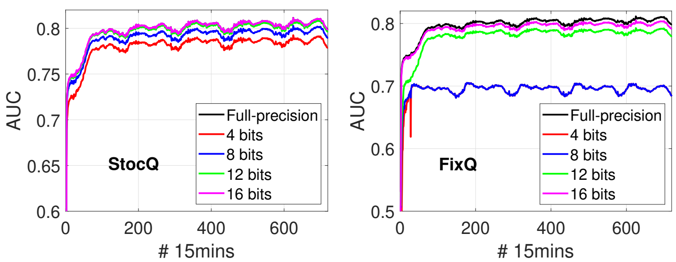

Figure 4: AUC curves with different quantization ranges.

Higher Embedding Dimension. As we show above, the 16-bit quantization scheme already works without sacrificing prediction accuracy. But our ultimate goal is to improve the prediction accuracy. Obviously, the saved half storage by the low-precision quantization over the embedding layer enables us to further improve the CTR prediction performance under the same storage as the baseline, by simply expanding this layer with higher dimension and hence higher model capacity without changing the network structure. Let $\dim_e$ denote the current embedding dimension. Figure 5 shows the test AUC difference curve between the quantized CTR model with embedding layer of double dimension, as well as the AUC difference between the quantized CTR model with $2 \times \dim_e$ and the full-precision CTR model with $\dim_e$ (baseline). The use of double embedding dimension brings $\sim 0.16\%$ AUC improvement in both cases, which in turn means a significant increase of revenue.

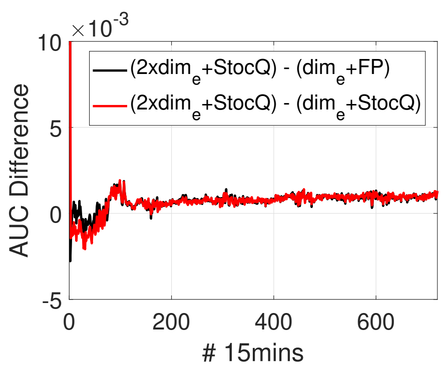

Figure 5: AUC increase after doubling the embedding dimension, with 16-bit quantization. FP stands for full precision.

The results in the above evaluations of the trained CTR model simulate the production environment and are often reflective of the real click-through rate in the production environment[1]. We have deployed the model to the production. Obtained via A/B testing, this CTR model in the real sponsored search production environment yields 1.8% increase of the real online click-through rate, i.e., 1.8% more click ads or show ads, estimated from over one billion valid page views. The revenue in turn rises 1%. This increase is highly substantial from a commercial perspective.

[1]In sponsored search, given a user query and features of an ad, typically CTR $\times$ bid is used to rank the ads for display, where bid is the cost the advertiser is willing to pay.

4.2 Comparison with Multi-Node System

Our legacy CPU-only training system was doing distributed training and then has been upgraded to the current one node GPU training. The distributed training system consisted of 150 CPU-only computing nodes with each having 16-core CPU and 180 GB memory, while the GPU computing node in the current training system employs more pricey hardware: larger memory (1 TB), SSDs and GPUs: the cost of one GPU computing node is around 15 times of the CPU-only node. However, we only need a single one-node GPU computing node to complete the job on the 150 CPU-only nodes cluster: the current one-node system utilizes a much lower expense (one-tenth) comparing with the 150 CPU-only cluster. Besides that, the training time has been greatly reduced: There are 0.6 billion training instances generated in one day. The current GPU training takes only 2.5 hours for this sample size with delay, caused by page view, online update of input features, and online update of the model, not exceeding 3 hours, while the CPU-only multi-node one took 10 hours. The AUCs of both solutions are very similar: the one-node system has a slightly better AUC as it requires fewer synchronizations and uses fewer stale parameters. Comparisons are summarized in Table 1.

Stage	Time	AUC	Cost
one-node	2.5 h.	$1.001 \times auc$	$0.1 \times cost$
multi-node	10 h.	auc	$cost$

Table 1: One-node versus multi-node system.

5 CONCLUSION

As early as in 2013, Baidu Search Ads (a.k.a. "Phoenix Nest") has been successfully using distributed massive-scale deep learning systems for training CTR models. In this paper, we focus on presenting Baidu's recent effort in building a GPU-based single-node system combined with quantization techniques for model compression. Our experiments show that quantization using stochastic rounding (StocQ) noticeably outperforms fixed rounding (FixQ). Our work also reveals that, in contrast to many academic studies on 1-bit or 2-bit quantization-based learning systems, industrial-scale production systems may need substantially more bits, for example, 16 bits in our system. This quantization step enables us to double the dimensionality of the embedding layer without increasing the storage. We have deployed this system in production and observed a substantial increase in the prediction accuracy and the revenue.

ACKNOWLEDGEMENT

We would like to thank Xuewu Jiao, Mingqing Hu, Xuwen Wang, Yawei Li, Hongliang Li, Yuqing Huang, Lian Zhao, Lin Liu, among many colleagues who have contributed to this important project.

REFERENCES

[1] RCM Barnes, EH Cooke-Yarborough, and DGA Thomas. 1951. An electronic digital computor using cold cathode counting tubes for storage. *Electronic Engineering* (1951).

[2] Andrei Broder. 2002. A taxonomy of web search. *SIGIR Forum* 36, 2 (2002), 3–10.

[3] Wenlin Chen, James T. Wilson, Stephen Tyree, Kilian Q. Weinberger, and Yixin Chen. 2015. Compressing Neural Networks with the Hashing Trick. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*. Lille, France, 2285–2294.

[4] Xi Chen, Xiaolin Hu, Hucheng Zhou, and Ningyi Xu. 2017. FxpNet: Training a deep convolutional neural network in fixed-point representation. In *Proceedings of the 2017 International Joint Conference on Neural Networks (IJCNN)*. Anchorage, AK, 2494–2501.

[5] Jungwook Choi, Swagath Venkataramani, Vijayalakshmi Srinivasan, Kailash Gopalakrishnan, Zhuo Wang, and Pierce Chuang. 2019. Accurate and Efficient 2-bit Quantized Neural Networks. In *Proceedings of Machine Learning and Systems (MLSys)*. Stanford, CA.

[6] Henggang Cui, Hao Zhang, Gregory R Ganger, Phillip B Gibbons, and Eric P Xing. 2016. Geeps: Scalable deep learning on distributed gpus with a gpu-specialized parameter server. In *Proceedings of the Eleventh European Conference on Computer Systems (EuroSys)*. London, UK, 1–16.

[7] Benjamin Edelman, Michael Ostrovsky, and Michael Schwarz. 2007. Internet Advertising and the Generalized Second-Price Auction: Selling Billions of Dollars Worth of Keywords. *American Economic Review* 97, 1 (March 2007), 242–259.

[8] Daniel C. Fain and Jan O. Pedersen. 2006. Sponsored search: A brief history. *Bulletin of the American Society for Information Science and Technology* 32, 2 (2006), 12–13.

[9] Miao Fan, Jiacheng Guo, Shuai Zhu, Shuo Miao, Mingming Sun, and Ping Li. 2019. MOBIUS: Towards the Next Generation of Query-Ad Matching in Baidu's Sponsored Search. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, (KDD) 2019*. Anchorage, AK, 2509–2517.

[10] Hongliang Fei, Shulong Tan, Pengju Guo, Wenbo Zhang, Hongfang Zhang, and Ping Li. 2020. Sample Optimization For Display Advertising. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM)*. Virtual Event, Ireland, 2017–2020.

[11] George E Forsythe. 1950. Round-off errors in numerical integration on automatic machinery-preliminary report. In *Bulletin of the American Mathematical Society*, Vol. 56. 61–61.

[12] George E Forsythe. 1959. Reprint of a note on rounding-off errors. *SIAM Rev.* 1, 1 (1959), 66.

[13] Sean Fox, Julian Faraone, David Boland, Kees A. Vissers, and Philip H. W. Leong. 2019. Training Deep Neural Networks in Low-Precision with High Accuracy Using FPGAs. In *Proceedings of the International Conference on Field-Programmable Technology (FPT)*. Tianjin, China, 1–9.

[14] Thore Graepel, Joaquin Quiñonero Candela, Thomas Borchert, and Ralf Herbrich. 2010. Web-Scale Bayesian Click-Through rate Prediction for Sponsored Search Advertising in Microsoft's Bing Search Engine. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*. Haifa, Israel, 13–20.

[15] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI)*. Melbourne, Australia, 1725–1731.

[16] Suyog Gupta, Ankur Agrawal, Kailash Gopalakrishnan, and Pritish Narayanan. 2015. Deep Learning with Limited Numerical Precision. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*. Lille, France, 1737–1746.

[17] Xiangnan He and Tat-Seng Chua. 2017. Neural Factorization Machines for Sparse Predictive Analytics. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. Shinjuku, Tokyo, 355–364.

[18] Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, Stuart Bowers, and Joaquin Quiñonero Candela. 2014. Practical Lessons from Predicting Clicks on Ads at Facebook. In *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising (ADKDD)*. New York City, New York, 5:1–5:9.

[19] Jiawei Jiang, Fangcheng Fu, Tong Yang, and Bin Cui. 2018. SketchML: Accelerating Distributed Machine Learning with Data Sketches. In *Proceedings of the 2018 International Conference on Management of Data (SIGMOD)*. Houston, TX, 1269–1284.

[20] Qing-Yuan Jiang and Wu-Jun Li. 2017. Deep Cross-Modal Hashing. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Honolulu, HI, 3270–3278.

[21] Yuchin Juan, Damien Lefortier, and Olivier Chapelle. 2017. Field-aware Factorization Machines in a Real-world Online Advertising System. In *Proceedings of the 26th International Conference on World Wide Web Companion (WWW)*. Perth, Australia, 680–688.

[22] Shahin Khobahi, Naveed Naimipour, Mojtaba Soltanalian, and Yonina C. Eldar. 2019. Deep Signal Recovery with One-bit Quantization. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Brighton, UK, 2987–2991.

[23] Jason Kuen, Xiangfei Kong, Zhe Lin, Gang Wang, Jianxiong Yin, Simon See, and Yap-Peng Tan. 2018. Stochastic Downsampling for Cost-Adjustable Inference and Improved Regularization in Convolutional Networks. In *Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Salt Lake City, UT, 7929–7938.

[24] Cong Leng, Zesheng Dou, Hao Li, Shenghuo Zhu, and Rong Jin. 2018. Extremely Low Bit Neural Network: Squeeze the Last Bit Out With ADMM. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI)*. New Orleans, LA, 3466–3473.

[25] Dingcheng Li, Xu Li, Jun Wang, and Ping Li. 2020. Video Recommendation with Multi-gate Mixture of Experts Soft Actor Critic. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval (SIGIR)*. Virtual Event, China, 1553–1556.

[26] Hao Li, Soham De, Zheng Xu, Christoph Studer, Hanan Samet, and Tom Goldstein. 2017. Training Quantized Nets: A Deeper Understanding. In *Advances in Neural Information Processing Systems (NIPS)*. Long Beach, CA, 5811–5821.

[27] Ping Li. 2007. Very sparse stable random projections for dimension reduction in l_α ($0 < \alpha \leq 2$) norm. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. San Jose, CA, 440–449.

[28] Ping Li, Xiaoyun Li, and Cun-Hui Zhang. 2019. Re-randomized Densification for One Permutation Hashing and Bin-wise Consistent Weighted Sampling. In *Advances in Neural Information Processing Systems (NeurIPS)*. Vancouver, Canada, 15900–15910.

[29] Xiaoyun Li and Ping Li. 2021. Rejection Sampling for Weighted Jaccard Similarity Revisited. In *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI)*. Virtual Event.

[30] Zeyu Li, Wei Cheng, Yang Chen, Haifeng Chen, and Wei Wang. 2020. Interpretable Click-Through Rate Prediction through Hierarchical Attention. In *Proceedings of the Thirteenth ACM International Conference on Web Search and Data Mining (WSDM)*. Houston, TX, 313–321.

[31] Jianxun Lian, Xiaohuan Zhou, Fuzheng Zhang, Zhongxia Chen, Xing Xie, and Guangzhong Sun. 2018. xDeepFM: Combining Explicit and Implicit Feature Interactions for Recommender Systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*. London, UK, 1754–1763.

[32] Darryl Dexu Lin, Sachin S. Talathi, and V. Sreekanth Annapureddy. 2016. Fixed Point Quantization of Deep Convolutional Networks. In *Proceedings of the 33nd International Conference on Machine Learning (ICML)*, Vol. 48. New York City, NY, 2849–2858.

[33] Shaoshi Ling, Yangqiu Song, and Dan Roth. 2016. Word Embeddings with Limited Memory. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*. Berlin, Germany.

[34] Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory F. Diamos, Erich Elsen, David García, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu. 2018. Mixed Precision Training. In *Proceedings of the 6th International Conference on Learning Representations (ICLR)*. Vancouver, Canada.

[35] Matthew Richardson, Ewa Dominowska, and Robert Ragno. 2007. Predicting Clicks: Estimating the Click-Through Rate for New Ads. In *Proceedings of the 16th International Conference on World Wide Web (WWW)*. Banff, Canada, 521–530.

[36] Christopher De Sa, Matthew Feldman, Christopher Ré, and Kunle Olukotun. 2017. Understanding and Optimizing Asynchronous Low-Precision Stochastic Gradient Descent. In *Proceedings of the 44th Annual International Symposium on Computer Architecture (ISCA)*. Toronto, Canada, 561–574.

[37] Charbel Sakr, Naigang Wang, Chia-Yu Chen, Jungwook Choi, Ankur Agrawal, Naresh R. Shanbhag, and Kailash Gopalakrishnan. 2019. Accumulation Bit-Width Scaling For Ultra-Low Precision Training Of Deep Networks. In *Proceedings of the 7th International Conference on Learning Representations (ICLR)*. New Orleans, LA.

[38] Ying Shan, T. Ryan Hoens, Jian Jiao, Haijing Wang, Dong Yu, and J. C. Mao. 2016. Deep Crossing: Web-Scale Modeling without Manually Crafted Combinatorial Features. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. San Francisco, CA, 255–262.

[39] Anshumali Shrivastava and Ping Li. 2014. Asymmetric LSH (ALSH) for Sublinear Time Maximum Inner Product Search (MIPS). In *Advances in Neural Information Processing Systems (NIPS)*. Montreal, Canada, 2321–2329.

[40] Kai Sheng Tai, Vatsal Sharan, Peter Bailis, and Gregory Valiant. 2018. Sketching Linear Classifiers over Data Streams. In *Proceedings of the 2018 International Conference on Management of Data (SIGMOD)*. Houston, TX, 757–772.

[41] Shulong Tan, Zhixin Zhou, Zhaozhuo Xu, and Ping Li. 2020. Fast Item Ranking under Neural Network based Measures. In *Proceedings of the 13th ACM International Conference on Web Search and Data Mining (WSDM)*. Huston, TX, 591–599.

[42] Hal R. Varian. 2007. Position auctions. *International Journal of Industrial Organization* 25, 6 (2007), 1163 – 1178.

[43] Kilian Q. Weinberger, Anirban Dasgupta, John Langford, Alexander J. Smola, and Josh Attenberg. 2009. Feature hashing for large scale multitask learning.

In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML)*. Montreal, Canada, 1113–1120.

[44] Bernard Widrow and István Kollár. 2008. Quantization noise. *Cambridge University Press* (2008).

[45] Tan Yu, Xuemeng Yang, Yan Jiang, Weijie Zhao, Hongfang Zhang, and Ping Li. 2021. TIRA in Baidu Image Advertising. In *Proceedings of the 36th IEEE International Conference on Data Engineering (ICDE)*. Virtual Event.

[46] Tan Yu, Yi Yang, Yi Li, Xiaodong Chen, Mingming Sun, and Ping Li. 2020. Combo-Attention Network for Baidu Video Advertising. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*. Virtual Event, CA, 2474–2482.

[47] Jian Zhang, Jiyan Yang, and Hector Yuen. 2018. Training with low-precision embedding tables. In *Systems for Machine Learning Workshop at NeurIPS*. Montreal, Canada.

[48] Weijie Zhao, Shulong Tan, and Ping Li. 2020. SONG: Approximate Nearest Neighbor Search on GPU. In *Proceedings of the 35th IEEE International Conference on Data Engineering (ICDE)*. Dallas, TX, 1033–1044.

[49] Weijie Zhao, Deping Xie, Ronglai Jia, Yulei Qian, Ruiquan Ding, Mingming Sun, and Ping Li. 2020. Distributed Hierarchical GPU Parameter Server for Massive Scale Deep Learning Ads Systems. In *Proceedings of Machine Learning and Systems (MLSys)*. Austin, TX.

[50] Weijie Zhao, Jingyuan Zhang, Deping Xie, Yulei Qian, Ronglai Jia, and Ping Li. 2019. AIBox: CTR Prediction Model Training on a Single Node. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM)*. Beijing, China, 319–328.

[51] Ding-Xuan Zhou. 2020. Theory of deep convolutional neural networks: Downsampling. *Neural Networks* 124 (2020), 319–327.

[52] Guorui Zhou, Xiaoqiang Zhu, Chengru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep Interest Network for Click-Through Rate Prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*. London, UK, 1059–1068.

[53] Zhixin Zhou, Shulong Tan, Zhaozhuo Xu, and Ping Li. 2019. Möbius Transformation for Fast Inner Product Search on Graph. In *Advances in Neural Information Processing Systems (NeurIPS)*. Vancouver, Canada, 8216–8227.