Practical Lessons from Predicting Clicks on Ads at Facebook

Gelvan Kirill

*Faculty of Computer Science*

*Higher School of Economics*

Moscow, Russia

kpgelvan@edu.hse.ru

Abstract

This document is a review on a paper, devoted to interesting and useful findings with Predicting Clicks on Ads at Facebook. The paper not only introduces a promising hybrid model(over 3% better than uniform ones) which combines decision trees with logistic regression, but also shows what parameters impact the performance the most. Furthermore, there are tricks described to keep memory and latency contained, as well as subsampling ideas.

# Introduction

**Background.** The bid and pay click auctions were introduced in 2007. Today with loads of data, almost a billion Facebook users and millions of active advertisers, predicting clicks on ads is a challenging machine learning task. The goal of the reviewing paper is to share the results from experiments with real world data from Facebook.

**Problem statement.** The efficiency of an ads auction depends on the accuracy of click prediction. The volume of ads that can be showed to a Facebook user is enormous occasionally. In order to choose an ad, firstly, a cascade of classifiers is build. The focus of the paper is the last stage click prediction model that produces predictions for the final set of eligible ads. Additionally, the cost of training should be controlled by reducing the volume of data.

# Main part

**Experimental setup.** The data of one week are fixed and downloaded, then divided into training and test. All experiments in the paper uses those data. Normalized Entropy (NE), which is the ratio of average logarithmic loss per impression to entropy of the average empirical click through rate (CTR) of the entire training data set, is chosen as the evaluation metric. The higher the value is, the worse is the model’s prediction.

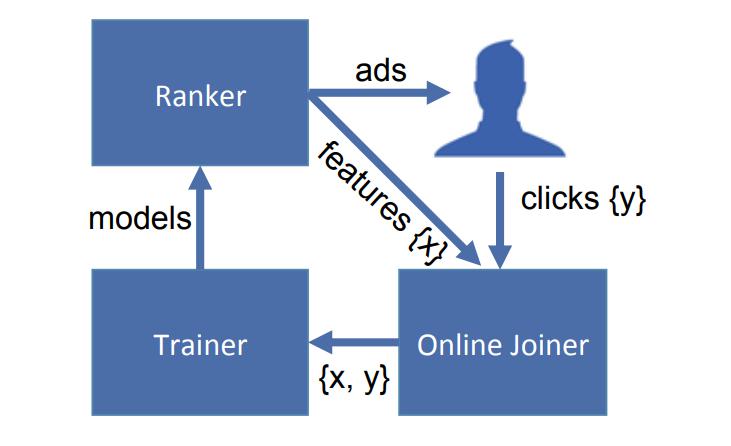
**Prediction model structure.** In this section different parts and parameters of the model are discussed. To start with, all the online learning schemes that are shown in the paper base on the Stochastic Gradient Descent (SGD) algorithm [2] applied to sparse linear classifiers. Mostly described stream learners are: SGD-based Logistic Regression (LR) and Bayesian online learning scheme for probit regression (BOPR) [3].

*Decision tree feature transforms.*Different approaches to improving accuracy of linear classifier by transforming the input features are shown in the reviewing paper. Boosted decision trees turn out to be a powerful way of implementing wanted transformations: each individual tree corresponds to a categorical feature that takes index of the leaf an instance ends up falling in as value. Gradient Boosting Machine (GBM) [4] is used for boosted decision trees: a new tree, modeling the residual of previous trees, is constructed each learning iteration. Authors show that while isolated LR and Tree models have almost the same prediction accuracy in terms of NE, their combination has an accuracy leap. Alike feature transformations largely decrease NE by more then 3.4% relative to the NE of the model without such feature transformations.

*Data freshness.*Authors study how the freshness of data changes performance. The model is trained on one particular day and tested on consecutive days. Results show that for both logistic regression with tree-transformed input features and boosted decision tree model normalized entropy rises by 1%. Thus it is worth retraining every one or two days. However, time needed for this can take more than 24 hours. One of the solutions is to train boosted decision trees daily, but train linear classifier in near real-time by using online learning.

*Online linear classifier.*Firstly, different learning rates for SGD-based LR are compared by NE. Per-coordinate learning rate achieves best accuracy (almost 5% lower than the worst performer) assuming the results of the experiment. Next trial shows that LR trained with SGD with per-coordinate learning rate and BOPR have very similar performance (difference is approximately 0.2%). The advantage of LR over BOPR is half a model size, because there is no need to have both mean and variance, thus faster cache lookup. The major advantage of BOPR over LR is that the first one provides a full predictive distribution over the probability of click, which can be useful.

**Online data joiner.** Online joiner is a system that generates real-time data for linear classifier’s online learning. It joins information about the click to ad impressions. It works as illustrated in Figure 1[1].

**Figure 1: Online Learning Data/Model Flows.[1]**

It is implemented using HashQueue consisting of FIFO queue and hash map. In the setup the trainer learns continuously and makes new models occasionally. There can not be any “no click” button, so there is a waiting window that need to be chosen carefully.

**Containing memory and latency.** In this section three different optimizations are discussed.

*Number of boosting trees.* The time taken to make a prediction can be saved if the number of trees decreases. An experiment varying that theory from 1 to 2000 shows that almost all NE improvements comes from the first 500 trees.

*Boosting feature importance.* Static Boosting Feature Importance is used in the next experiment. It tries to find the cumulative loss reduction for a feature. As expected slight amount of features contributes the majority of explanatory power. The results show that after adding 300 features, all the next ones almost will not change NE.

*Historical features.* There are two types of features in the Boosting model: contextual and historical. By sorting all features by importance and calculating the percentage of those two types of features in the first half, it is shown and proven that historical features provide a lot more explanatory power. From another test we learn that without contextual features model loses only 1% loss in prediction accuracy. The next step is to check our intuition on the dependency on data freshness for both different kinds of features. Turns out that contextual features rely more on data freshness than historical.

**Coping with massive training data.** A common technique used to control the cost of training is presented in this section. It is reducing the volume of data. We learn about 2 different ways of data volume cutback.

*Uniform subsampling.* The idea is to evaluate a set of exponentially increasing rates and train a boosted tree model at each one of them. The most important result is that by using only 10% of the data, NE has only 1% reduction relative to training on the entire data set.

*Negative down sampling.* The idea is simple: To overcome class imbalance authors “downsample” the data set with different rates and compare the result in the NE metrics. Empirically it occurs that the best performance is achieved with negative down sampling rate set to 0.025.

# Conclusion

Practical lessons presented at reviewed paper are important and useful, because they may be used by other researches. I believe that there are even better methods for building a model than described in paper, but those made a significant improvement in accuracy that can be used in practice. Shown practical lessons and tricks play an important role in future researches because it is already cited by more than 200 research articles[5].

# References

[1] Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, Stuart Bowers, Joaquin Quiñonero Candela, “Practical Lessons from Predicting Clicks on Ads at Facebook”, facebook research, Dec, 2016. [Online]. Available:

<https://research.fb.com/wp-content/uploads/2016/11/practical-lessons-from-predicting-clicks-on-ads-at-facebook.pdf>

[2] L. Bottou. Online algorithms and stochastic approximations. In D. Saad, editor, Online Learning and Neural Networks. Cambridge University Press, Cambridge, UK, 1998. revised, oct 2012.

[3] T. Graepel, J. Qui˜nonero 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, pages 13–20, 2010.

[4] J. H. Friedman. Greedy function approximation: A gradient boosting machine. Annals of Statistics, 29:1189–1232, 1999.

[5] Google Scholar, 2019. [Online]. Available:

<https://scholar.google.ru/scholar?hl=en&as_sdt=0%2C5&q=Practical+Lessons+from+Predicting+Clicks+on+Ads+at+Facebook&btnG=>