# A Unified Framework for Campaign Performance Forecasting in Online Display Advertising

Anonymous Author(s)

**ACM Reference Format:**
Anonymous Author(s). 2021. A Unified Framework for Campaign Performance Forecasting in Online Display Advertising. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

## 1 IMPLEMENTATION DETAILS

In order to improve the reproducibility of our framework, we present essential details for implementation in this part, including the construction of base log, unified replay algorithm, calibration model, and online latency observation.

### 1.1 Log Construction

In this section, we introduce how we organize and collect base logs for campaign performance forecasting, and the detailed schema of these logs. For better display of the log construction process, the overall log stream is presented in Figure 1, and the detailed log schema is described in Figure 2.

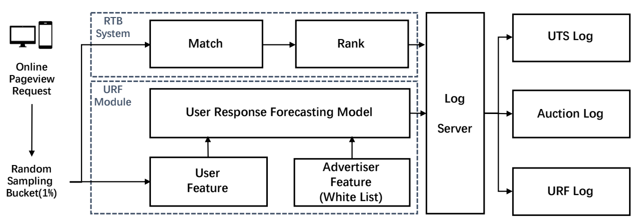


**Figure 1: The overall log stream.**

| UTS Log Schema | Auction Log Schema | URF Log Schema |
|---|---|---|
| • pageview_id | • pageview_id | • pageview_id |
| • user_id | • user_id | • user_id |
| • tag_type1 | • hour | • advertiser_id |
| • tag_type2 | • area | • pctr |
| ... | • adzone | • pcvr |
| | • winner | ... |
| | • b1 | |
| | • b2 | |
| | • click | |
| | • conversion | |

**Figure 2: Detailed log schema.**


*Conference'17, July 2017, Washington, DC, USA*

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM...$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

- **Down Sampling.** We build a $\frac{1}{100}$ down sampling bucket with a special marker on the online traffic platform, online page view requests arrive continuously while users log in and browse on Taobao.com. The probability of entering the bucket is equal for every request, those who pass through the bucket will be always marked in the following log stream. It should be noted that the bucket marker on requests plays an important role in computation complexity and latency optimization in our framework.
- **UTS Log and Auction Log.** All the ad requests will go through the real-time bidding(RTB) system for auction, in which match module retrieves all targeted bidders, and rank module calculates the bid price for all bidders and rank to decide the final winner. In match phase, the targeting types and values from bidders which retrieves a request will be recorded into User-Tag-Service(UTS) Log. The context, winner, rank score and corresponding clicks/conversions of a request will be recorded as Auction Log either. In final deployment, the UTS and Auction Logs without the bucket marker are filtered out to optimize computation complexity.
- **URF Log.** When ad requests arrives at User Response Forecasting(URF) module, the module will retrieve essential features for marked requests and white-list advertisers, and then feed into trained URF model to predict *pctr* and *pcvr* values. All the outputs will be written into URF Log. Notably, the deployed URF model here is trained on Action Log, and the white list on advertisers is designed for online A/B test.

As most advertising platforms have similar bidding system to Taobao.com, the log construction in our framework could be well generalized to others.

### 1.2 Unified Replay Algorithm

In this section, we show more additional details about the unified replay algorithm that are essential in supporting our reproducibility. With the base logs as described in Appendix A, we focus on the elaboration of 'Match Phase' and 'Rank Phase' in replay and the latency optimization for online deployment.

- **Match Phase.** Match phase aims to retrieve complete auction information from historical logs after advertisers specify their campaign criteria. The detailed logic is shown in Figure 3.
- **Rank Phase.** Based on the auction information retrieved in Match Phase, Rank phase determines the final auctions in which the campaign would win, and then calculate the performance for campaigns. The detailed logic is shown in Figure 4.
- **Online Latency.** In the proposed framework, most of the computation is concentrated in the replay algorithm, we adopt several speed-up actions to optimize the latency for online service. (1) As described in Appendix A, a down sampling rate $\frac{1}{100}$ is applied in log stream to reduce computation complexity. (2) Only essential Match Phase and Rank Phase in auction process are considered in replay, those strategies which are hard to reproduce

are neglected, leaving the replay deviation to be calibrated in performance calibration module. (3) Engineering optimization on MaxCompute Hologres[6]. We present the online service latency monitor of 27/12/2020 in Figure 5, it's clear that most requests are responded in 2 seconds.

**Unified Replay: Match**

```
SELECT pageview_id, user_id, advertiser_id, pctr, value, cost
FROM (
   SELECT pageview_id, user_id, IF($advertiser_id=winner, b2, b1) AS cost
   FROM AUCTION_LOG
   WHERE hour IN ($hours)
   AND area IN ($areas)
   AND adzone IN ($adzones)
) A
JOIN (
   SELECT pageview_id, user_id
   FROM UTS_LOG
   WHERE tag_$target_type1 IN ($target_value1)
   OR tag_$target_type2 IN ($target_value2)
) B
ON A.pageview_id = B.pageview_id
AND A.user_id = B.user_id
JOIN (
   SELECT pageview_id, user_id, advertiser_id, pctr,
   CASE WHEN $objective=IMPRESSION THEN 1
   WHEN $objective=CLICK THEN pctr
   WHEN $objective=CONVERSION THEN pctr*pcvr
   ELSE pctr END AS value
   FROM RTP_LOG
   WHERE advertiser_id = $advertiser_id
) C
ON A.pageview_id = C.pageview_id
AND A.user_id = C.user_id
;
```

**Figure 3: The SQL logic of Match Phase, variables in orange are inputs from given campaign criteria.**

## 1.3 Calibration Model

In this section, we mainly elaborate the construction of calibration model. We first present our consideration on feature selection, then the implementation details of offline training and online deployment respectively.

- **Feature Selection.** The calibration features consist of two parts, campaign criteria and replay outputs. For campaign criteria, targeting option, objective and bidding type are usually tied to delivery strategies, thus we choose these criteria to capture calibration patterns. For replay outputs, statistical features in match phase are calculated to represent the targeting crowd quality for a campaign, and the final results in rank phase are applied as base campaign performance. As detailed campaign criteria is described in Section 3.1, and replay outputs of Match Phase and Rank Phase are clearly illustrated in Algorithm 1, we list the overall input for calibration model in Table 1.
- **Offline Training.** The replay algorithm are accomplished on the high-speed distributed cloud computing frame MaxCompute Hologres[6]. We collect campaigns samples as described in Section 4.1. Campaign criteria, replay outputs and the true

**Unified Replay: Rank**

**Manual Bidding**

```
SELECT
$campaign_id AS campaign_id,
SUM(IF($bidprice > standard, 1, 0))/$sampling_rate AS impression,
SUM(IF($bidprice > standard, pctr, 0))/$sampling_rate AS click,
SUM(IF($bidprice > standard, cost, 0))/$sampling_rate AS cost,
SUM(1)/$sampling_rate AS audience_size,
MEAN(pctr) AS pctr_mean,
MEAN(value) AS value_mean,
MEAN(cost) AS cost_mean,
MEDIAN(pctr) AS pctr_median,
MEDIAN(value) AS value_median,
MEDIAN(cost) AS cost_median
FROM (
  SELECT pctr, value, cost, CASE WHEN $objective=IMPRESSION THEN cost
  ELSE cost/(pctr*1000) END AS standard
  FROM (
    ......    --MATCH SQL
  ) A
) A
;
```

**Automatic Bidding**

```
SELECT
$campaign_id AS campaign_id,
MAX(IF(s_cost<=$budget*$sampling_rate AND IF($constraint IS NOT NULL,
s_cost/s_value<=$constraint, True), s_impression, 0))/$sampling_rate AS impression,
MAX(IF(s_cost<=$budget*$sampling_rate AND IF($constraint IS NOT NULL,
s_cost/s_value<=$constraint, True), s_click, 0))/$sampling_rate AS click,
MAX(IF(s_cost<=$budget*$sampling_rate AND IF($constraint IS NOT NULL,
s_cost/s_value<=$constraint, True), s_cost, 0))/$sampling_rate AS cost,
SUM(1)/$sampling_rate AS audience_size,
MEAN(pctr) AS pctr_mean,
MEAN(value) AS value_mean,
MEAN(cost) AS cost_mean,
MEDIAN(pctr) AS pctr_median,
MEDIAN(value) AS value_median,
MEDIAN(cost) AS cost_median
FROM (
  SELECT
  SUM(1) OVER(PARTITION BY 1 ORDER BY standard) AS s_impression ,
  SUM(pctr) OVER(PARTITION BY 1 ORDER BY standard) AS s_click,
  SUM(value) OVER(PARTITION BY 1 ORDER BY standard) AS s_value,
  SUM(cost) OVER(PARTITION BY 1 ORDER BY standard) AS s_cost,
  pctr, value, cost
  FROM (
    SELECT pctr, value, cost/1000 AS cost, cost/value AS standard
    FROM (
      ......    --MATCH SQL
    ) A
  ) A
) A
;
```

**Figure 4: The SQL logic of Rank Phase, including both manual bidding and automatic bidding. Variables in orange are inputs from given campaign criteria.**

performance are adopted as features and labels respectively for offline training, and the discrete values in campaign criteria are processed to 1-hot vectors. It should be noticed that most settings for model training follow the description in [2, 5]. The number of tasks and experts are set to 3 and 6, hidden units of expert layer and tower layer are set to 64, 32 respectively. Batch-Norm layers[3] are adopted in the models, and Adam[4] is adopted for optimization with a initial learning rate=$10^{-3}$. Our code is implemented with TensorFlow[1] in python.

- **Online Service.** The trained model is deployed on the Real-Time Prediction(RTP) center in Taobao advertising system for online

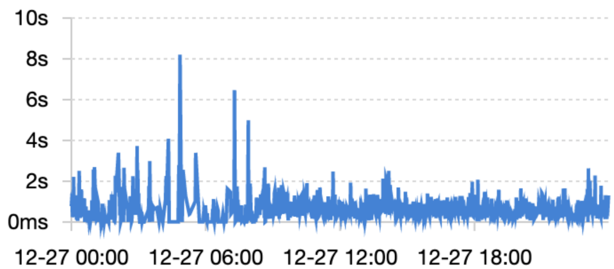


**Figure 5: The latency of online service.**

service. The unified replay algorithm receives campaign criteria from user interface and calculate the results in real-time on MaxCompute[6], then the campaign criteria and replay outputs are combined as input to call the calibration model by an HTTP request. Finally, calibrated performance is fed to advertisers for campaign optimization.

**Table 1: Input of Calibration Model.**

| Domain | Element |
|---|---|
| Campaign criteria | targeting_option, objective, bidding_type. |
| Replay outputs | pctr_mean, pctr_median, cost_mean, cost_median, cost, value_mean, value_median, audience_size, click, impression. |

# REFERENCES

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. 2016. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467* (2016).

[2] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. *arXiv preprint arXiv:1703.04247* (2017).

[3] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167* (2015).

[4] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[5] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H Chi. 2018. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1930–1939.

[6] Xu Yang and Sheng Zhang. 2016. A High Performance Distributed Cloud Computing Frame for Geophysical Applications. In *2016 Workshop: Workshop High Performance Computing, Beijing, China, 14-16 November 2016*. Society of Exploration Geophysicists, 7–8.