Warm Up Cold-Start Advertisements

Improving CTR predictions via Learning to Learn ID embeddings

Feiyang Pan¹, Shuokai Li¹, Xiang Ao¹, Pingzhong Tang², Qing He¹

¹ Institute of Computing Technology, CAS
² Tsinghua University

Feiyang Pan
24 July 2019
What is CTR prediction?

Binary Classification

Input: \{\text{ad, user, some contexts, \ldots} \}

Label: \{1, 0\} → click or not
What is the cold-start problem?

The model is not familiar with new / small ads (or users).

KDD cup 2012 search ads dataset
5% of the ads accounted for nearly 90% of the samples
pCTR = f(embedding of the ad ID, ad features, contexts)

For new ads:

→ No labeled sample.

→ Unknown ID embedding.

→ Inaccurate CTR prediction.
Warm Up Cold-start Advertisements: Improving CTR Predictions via Learning to Learn ID Embeddings

**Meta-Embedding**

**Look-up Embedding for old IDs**

- Model (e.g. MLP, FM, PNN, …)
- ID embedding
- Look-up table
- Old ID
- Ad features
- Other features
- Prediction

**Meta-Embedding for new IDs**

- Model (e.g. MLP, FM, PNN, …)
- ID embedding
- Meta-Embedding generator
- New ID
- Ad features
- Other features
- Meta-Embedding
- Prediction
Meta-Embedding

Look-up Embedding for old IDs

- ID embedding
- Look-up table
- Old ID
- Ad
- model (e.g. MLP, FM, PNN, …)
- Prediction

Meta-Embedding for new IDs

- ID embedding
- Meta-Embedding generator
- Features
- Other features
- model (e.g. MLP, FM, PNN, …)
- Prediction

Generate the initial embeddings of new IDs to warm up new ads.
How to use it?

(a) Trivial cold-start with random initializations
- Random Initializer (Gaussian or Uniform)
- Ad features $u_{[i]}$

(b) Warm up cold-start with Meta-Embedding
- Meta-Embedding Generator

Initialization (Offline)
- Initialize the embedding for new ad ID

Make predictions & update the embedding (Online)
- Make predictions and update for the first time
- Make predictions and update for the second time
Learning

Two phases & Two goals (for new ads):

(a) **cold-start phase:**

give good predictions for new ads without labeled data.

(b) **warm-up phase:**

learn quickly with a small number of labeled examples.
Learning

\[
\text{loss}_{\text{meta}} = \alpha \text{loss}_a + (1-\alpha) \text{loss}_b
\]

End-to-end training.
Learning

Algorithm 1 Train Meta-Embedding by SGD

Input: $f_{\theta}$: the pre-trained base model.
Input: $\mathcal{I}$: the set of all existing IDs.
Input: $\alpha$: hyper-parameter, the coefficient for meta-loss.
Input: $a, b$: step sizes.

1: Randomly initialize $\mathbf{w}$
2: while not done do
3:     Randomly sample $n$ IDs $\{i_1, \ldots, i_n\}$ from $\mathcal{I}$
4:     for $i \in \{i_1, \ldots, i_n\}$ do
5:         Generate the initial embedding: $\phi_{\theta[i]}^{\text{init}} = h_w(u_{[i]})$
6:         Sample mini-batches $\mathcal{D}^a_{[i]}$ and $\mathcal{D}^b_{[i]}$ each with $K$ samples
7:         Evaluate loss $l_a(\phi_{\theta[i]}^{\text{init}})$ on $\mathcal{D}^a_{[i]}$
8:         Compute adapted embedding: $\phi_{\theta[i]}' = \phi_{\theta[i]}^{\text{init}} - a \frac{\partial l_a(\phi_{\theta[i]}^{\text{init}})}{\partial \phi_{\theta[i]}^{\text{init}}}$
9:         Evaluate loss $l_b(\phi_{\theta[i]}')$ on $\mathcal{D}^b_{[i]}$
10:        Compute loss: $l_{\text{meta},i} = \alpha l_a(\phi_{\theta[i]}^{\text{init}}) + (1 - \alpha) l_b(\phi_{\theta[i]}')$
11:        Update $\mathbf{w} \leftarrow \mathbf{w} - b \sum_{i \in \{i_1, \ldots, i_n\}} \frac{\partial l_{\text{meta},i}}{\partial \mathbf{w}}$

End-to-end training with SGD.

Can be applied in both the offline and online settings.
Details

The basic structure of the embedding generator:

- Output Layer (Fully-connected)
- Simple composition (e.g. Average-pooling)
- Dense embeddings (Reused from the base model) (not trainable here)
- Raw inputs of item features

Parameters $w$
Warm Up Cold-start Advertisements: Improving CTR Predictions via Learning to Learn ID Embeddings

Experiments set-up

For each new ads, we split 3 mini-batches for simulating cold-start, others are held-out for testing.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Minibatch size K</th>
<th># of IDs</th>
<th># of samples</th>
<th># of samples used to train Meta-Embedding</th>
<th># of IDs</th>
<th># of samples</th>
<th># of samples in the hold-out set</th>
</tr>
</thead>
<tbody>
<tr>
<td>MovieLens-1M</td>
<td>20</td>
<td>1127</td>
<td>0.76 M</td>
<td>0.09 M</td>
<td>1058</td>
<td>0.19 M</td>
<td>0.12 M</td>
</tr>
<tr>
<td>Tencent CVR data</td>
<td>200</td>
<td>572</td>
<td>49.33 M</td>
<td>0.45 M</td>
<td>443</td>
<td>5.00 M</td>
<td>4.74 M</td>
</tr>
<tr>
<td>KDD Cup 2012</td>
<td>200</td>
<td>6534</td>
<td>148.55 M</td>
<td>5.22 M</td>
<td>9299</td>
<td>28.71 M</td>
<td>23.13 M</td>
</tr>
</tbody>
</table>

Experiment pipeline:
1. Pre-train the base model with the data of old ads;
2. Train the Meta-Embedding with the training data;
3. Generate initial embeddings of new ad IDs with (random initialization or Meta-Embedding);
4. Update the embeddings with batch-a and compute evaluation metrics on the hold-out set;
5. Update the embeddings with batch-b and compute evaluation metrics on the hold-out set;
6. Update the embeddings with batch-c and compute evaluation metrics on the hold-out set;

Evaluation metrics: Improvements on Log-loss and the AUC score.
Results: Significantly speed up cold-start phase

The experiment results on small dataset MovieLens.

Based on DeepFM, there was an improvement of about 15% against our baseline.
Results: Significantly speed up cold-start phase

The results on the Tencent Social Ads competition 2018 dataset for conversion rate prediction
Results: Significantly speed up cold-start phase

The results on KDD cup 2012 CTR prediction dataset for search ads

On all the tested datasets and base models, Meta-Embedding significantly improves the performance in both the cold-start and the warm-up phase.
Warm Up Cold-Start Advertisements
Improving CTR predictions via Learning to Learn ID embeddings

Feiyang Pan¹, Shuokai Li¹, Xiang Ao¹, Pingzhong Tang², Qing He¹

¹ Institute of Computing Technology, CAS
² Tsinghua University

Thank you!