CTR prediction models

Choi.Y

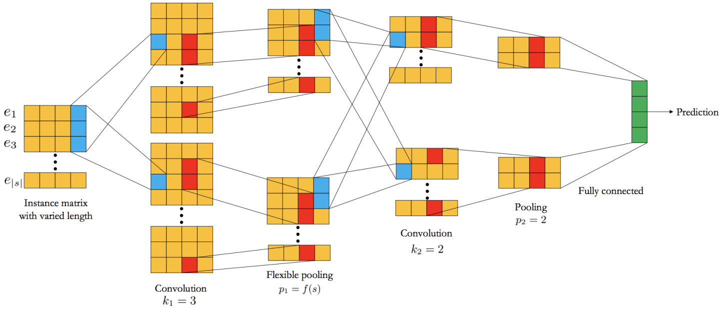
2018.06.28

A Convolutional Click Prediction Model

Click prediction

- Assume that there are historical click sequences.
- RNN is a good model for sequential data.
- Historical click sequence of a certain user is divided by different time intervals.
- RNN may have its limitation for these.
- CNN architecture can fully extract local-global key feature.

Structure



e_1
e_2
e_3
$e_{|s|}$
Instance matrix
with varied length
Convolution
$k_1 = 3$
Flexible pooling
$p_1 = f(s)$
Convolution
$k_2 = 2$
Pooling
$p_2 = 2$
Fully connected
Prediction

Notation

- $e_i \in R^d$ where e_i is an embedding vector
- $w \in R^d$ is a weight matrix
- s $= [e_1, \ldots, e_n] \in R^{d \times n}$ is a instance matrix.

Convolution layer

- One-dimensional Convolution step.
- Given w_i, s_i, convolution output $r_i = w_i^T s_{i,j-\omega+1:j}$ for $j = 1, \ldots, n + \omega - w$
- The optimized weight in the filter w detects feature and recognizes specific ranges of neighborhood in input instance.

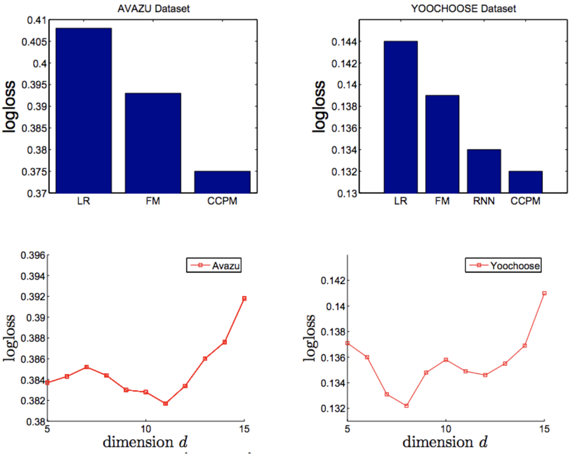
Flexible p-max pooling layer

- Given a convolutional output vector $r_i \in R^n$
- p-max pooling selects a sub-vector $s_i \in R^p$ which contains the p biggest values in r_i
- This strategy is due to the difference of each user's instance dimension.
- $p_i = \begin{cases} (1 - (i/l)^{l-i})n & i = 1, \ldots, l-1 \\ 3 & i = l \end{cases}$
- where l is the total number of convolutional layers, n is the length of the input instance.

Feature Maps

- Use $tanh(x) = \frac{exp(x) - exp(-x)}{exp(x) + exp(-x)}$
- F^i : i-th order feature map (convolution + pooling + activation)
- $F^i_j = \sum_{k=1}^{m_i} w^i_{j,k} * F^{i-1}_k$
- softmax + negative log likelihood

Result


AVAZU Dataset
logloss
0.41
0.405
0.4
0.395
0.39
0.385
0.38
0.375
0.37
LR
FM
CCPM
YOOCHOOSE Dataset
logloss
0.144
0.142
0.14
0.138
0.136
0.134
0.132
0.13
LR
FM
RNN
CCPM
Avazu
logloss
0.396
0.394
0.392
0.39
0.388
0.386
0.384
0.382
0.38
5
10
15
dimension d
Yoochoose
logloss
0.142
0.14
0.138
0.136
0.134
0.132
5
10
15
dimension d

Position-Normalized Click prediction in Search Avertising

Biased CTR estimates

- The observed click-through data has been confounded by positional bias.
- Since users tend to click more on ads shown in higher position.
- Want to find position-denoised CTR

Notation

- i : a quary-ad pair
- j : ad position
- v : the number of ad impressions.
- The observed CTR $= p(click|i,j)$

Assumption

1. Clicking an ad is independent of its position, given that it is physically examined.
2. Examining an ad is independent of its content or relevance, given its position.

- $p(click|i,j) = p(click|exam,i)p(exam|j)$
- $p(click|exam,i)$, denoted as p_i, is a position-normalized CTR.
- $p(exam|j)$, denoted as q_j, reflects the positional bias.

The binomial model

- model : $c_{ij} \sim B(v_{ij}, p_i q_j), \forall i, j$
- Training dataset D $= \{(c_{ij}, v_{ij})\}$
- parameter $\theta = (p, q)$
- $\ell(\theta) = \sum_{i,j} log \begin{pmatrix} v_{ij} \\ c_{ij} \end{pmatrix} c_{ij} \log(p_i q_j) + (v_{ij} - c_{ij}) \log(1 - p_i q_j)$

The binomial model

- regarding one of model parameters as latent variable(e.g. q)
- To estimate the MLE of both $\theta = (p, q)$, Use E-M Algorithm
- E-step: $q_j^{'} \leftarrow \frac{\sum_i c_{ij}}{\sum_i (v_{ij} - c_{ij}) p_i / (1 - p_i q_j}$
- M-step : $p_i^{'} \leftarrow \frac{\sum_j c_{ij}}{\sum_j (v_{ij} - c_{ij}) q_j / (1 - p_i q_j)}$

The Poisson Model

- If n is sufficiently large and p is sufficiently small ($np \to \lambda$)
- We can approximate Binomial dist as Poisson dist
- Model : $c_{ij} \sim Poisson(v_{ij} p_i q_j), \forall i, j.$
- E-step: $q_j^{'} \leftarrow \frac{\sum_i c_{ij}}{\sum_i (v_{ij} p_i}$
- M-step : $p_i^{'} \leftarrow \frac{\sum_j c_{ij}}{\sum_j v_{ij} q_j}$

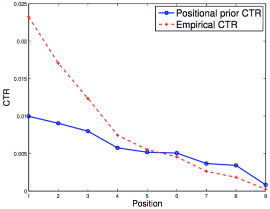
The Gamma-Poisson Model

- Impose a gamma prior on q
- $q_j \sim Gamma(\alpha, \beta), \forall j.$
- $p(D, q|p, \alpha, \beta) = \prod_{i,j} \frac{(v_{ij}p_iq_j)^{c_{ij}} exp(-v_{ij}p_iq_j)}{c_{ij}!} \times \prod_j \frac{q_j^{\alpha-1} exp(-q_j/\beta)}{\beta^\alpha \Gamma(\alpha)}$
- E-step: $q_j^{'} \leftarrow \frac{\sum_i c_{ij} + (\alpha - 1)}{\sum_i (v_{ij}p_i + 1/\beta}$
- M-step : $p_i^{'} \leftarrow \frac{\sum_j c_{ij}}{\sum_j v_{ij}q_j}$

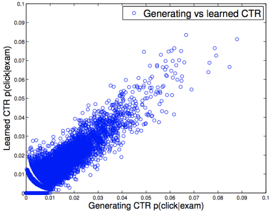
Simulation

1. $\forall$ position $j \in [1, \ldots, m]$, generate a $q_j \sim \text{Gamma}(\alpha, \beta)$, sort q in descending order, and scale q by $1/q_1$;
2. $\forall$ query-ad pair $i \in [1, \ldots, n]$, generate a $p_i \sim \text{Beta}(\gamma, \delta)$;
3. $\forall i$, generate a number of impressions $s_i \sim \text{Poisson}(\lambda)$;
4. $\forall i$, construct a multinomial distribution over positions $\phi_i \propto 1/\left(p_i/\mu(p_i)\right)^{j-1}$, to push good ads higher up;
5. $\forall i$, generate an impression allocation vector over positions $v_i \sim \text{Multinomial}(s_i, \phi_i)$, to form an $n \times m$ matrix of impression V;
6. Derive an $n \times m$ matrix of CTRs $Z = pq^\top$;
7. Derive an $n \times m$ matrix of Poisson means $Y = V.Z$, where '.' is element-wise multiplication;
8. Generate an $n \times m$ matrix of clicks $C \sim \text{Poisson}(Y)$, and $C \leftarrow \min(C, V)$, element-wise.

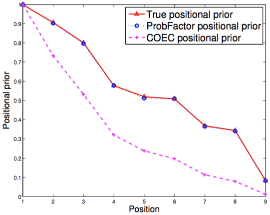
Simulation



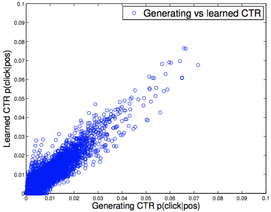
(a) Positional prior and empirical CTR



(a) True vs. learned $p_i = p(\text{click}|\text{exam})$



(b) True positional, Factor and COEC learned



(b) True vs. learned $p_{ij} = p(\text{click}|\text{pos})$