Fi-GNN Method 정리하기.

Methods.

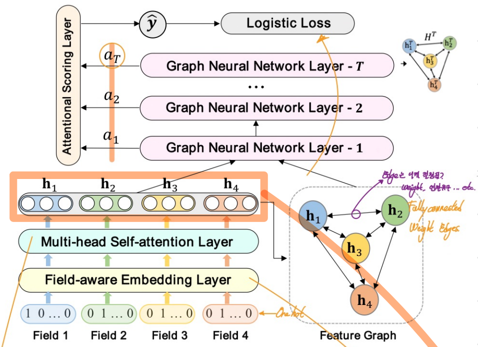

Figure 1: Overview of our proposed method. The input raw multi-field feature vector is first converted to field embedding vectors via an embedding layer and represented as a feature graph, which is then feed into Fi-GNN to model feature interactions. An attention layer is applied on the output of Fi-GNN to predict the click through rate $\hat{y}$. Details of embedding layer and Fi-GNN are illustrated in Figure 2 and Figure 3 respectively.

전체 모델 구성.

1. tabular → Graph

1) Field-aware Embedded Layer

one hot vector → low-dimensional embedded vector.

2) Multi-head Self-attention Layer.

(Head i의 attention)

$$H_i = \mathrm{Softmax}_i\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \quad H_i \in \mathbb{R}^{m \times d_1}$$

- $Q = W_i^{(Q)}E, \quad W_i^{(Q)} \in \mathbb{R}^{d_1 \times d}$
- $K = W_i^{(K)}E, \quad W_i^{(K)} \in \mathbb{R}^{d_1 \times d}$
- $V = W_i^{(V)}E, \quad W_i^{(V)} \in \mathbb{R}^{d_1 \times d}$

We combine the learnt feature representations of each head to preserve the pairwise feature interactions

$$\boxed{H' = \mathrm{ReLU}(H_1 \oplus H_2 \oplus \cdots \oplus H_h)}$$

위의 과정의 결과.

We represent each input multi-field feature as a "feature graph" $G = (N, E)$ where each node $n_i \in N$ corresponds to a feature field i;

it is a weighted fully connected graph while the edge weights reflects importances of different feature interactions.

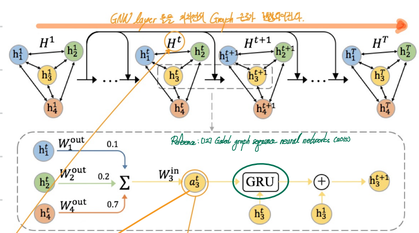

Figure 2: Framework of Fi-GNN. The nodes interact with neighbors and update their states in a recurrent fashion. At each interaction step, each node will first aggregate transformed state information from neighbors and then update its state according to the aggregated information and history via GRU and residual connection.

2. GNN layer 구성.

GNN layer는 크게 2 step으로 구성된다.

1) State aggregation.

at iteration step t, each node will aggregate the state information from neighbors.

$$a_i^t = \sum_{n_j \to n_i \in E} A[n_j, n_i] W_p h_j^{t-1}$$

a. Attentional Edge Weights

일반적인 adjacency matrix는 binary element (0, 1)로 구성되지만,

이러한 표현방식은 node간의 연결성은 반영할 수 있으나

그들의 Relation의 중요도는 반영되지 못한다.

해당 문제에 대한 Solution으로 논문에서는 아래와 같은 adjacency matrix의 사용을 제안한다.

$$A[n_i, n_j] = \begin{cases} w(n_i, n_j), & \text{if } i \neq j \\ 0 & \text{else} \end{cases}$$

$$w(n_i, n_j) = \frac{\exp(\mathrm{LeakyReLU}(W_w[e_i \| e_j]))}{\sum_k \exp(\mathrm{LeakyReLU}(W_w[e_i \| e_k]))}$$

이거 어케했나?

$W_w \in \mathbb{R}^{2d'}$ is a weight matrix, $\|$ is the concatenation operation.

the softmax function is utilized to make weights easily comparable across different nodes.

Edge weights reflects the importances of different interactions.

→ Fi-GNN can provide good explanations on the relation of different feature fields of input instance.

2) Edge-Wise Transformation.

A fixed transformed function on all the edges is unable to model the flexible interactions.

But simply assign a unique transformation weight to each edge will consuming too much parameter space and running time.

$$W_p^{n_i \to n_j} = W_{out}^i W_{in}^j$$

$$a_i^t = \sum_{n_j \to n_i \in E} A[n_j, n_i] \underline{W_{out}^j W_{in}^i} h_j^{t-1} + b_p$$

2. GNN layer 구성

2) State Update.

After aggregating state information, the nodes will update. (GRU, Residual Connection)

a. GRU.

해당논문은 이전 [12] Reference에서 소개한 GGNN에서 사용하는 GRU를 이용한 Update 방식을 그대로 채용한다.

$h_i^t = GRU(h_i^{t-1}, a_i^t)$

해당 Unit의 특성을 이해하기 위해 [12]를 읽어보자. 일단 torch.nn.GRU.

b. Residual Connections.

Residual Connection의 근거 : it's effective to combine the low-order high-order interactions together.

$h_i^t = GRU(h_i^{t-1}, a_i^t) + h_i^1$

내가 볼때에 이 논문의 가장 Weak point임.

3. Attentional Scoring Layer.

after T propagation steps, $H^T = [h_1^T, h_2^T, \cdots, h_m^T]$

the final state of each field node has captured the global information.

(Attentional Node Weights)

Here we predict a score on the final state of each field respectively and sum them with an attention mechanism which measures their influences on the overall prediction.

$\hat{y}_i = MLP_1(h_i^p), \; a_i = MLP_2(h_i^p)$

$\hat{y} = \sum_{i=1}^{m} a_i \hat{y}_i$