Poly Encoder Mathematical Foundation

Sam Grouchnikov

September 16, 2025

1 Context Encoding

- Input: context tokens $x = [x_1, x_2, ..., x_n]$
- Encode with a transformer
- Output hidden states:

$$H^c = [h_1^c, h_2^c, ..., h_n^c] \in R^{n \times d}$$

  Where d is the embedding dimension

2 Candidate Encoding

- Each candidate y is tokenized and encoded seperately
- Only pooled embedding is used:

$$h^y \in R^d$$

- With m candidates:

$$H^y \in R^{m \times d}$$

3 Poly Code Vectors

- Choose a fixed number M (64 from original paper)
- Initialize M learned codes:

$$P = [p_1, p_2, ..., p_M] \in R^{M \times d}$$

4 Poly Codes Attend to Context

- For each poly code p_i:

$$h_i^c = Attention(p_i, H_C, H_C) \in R^{M \times d}$$

- where

$$Attention(q, K, V) = softmax(\frac{qK^T}{\sqrt{d}})V$$

- Which yields:

$$H_c' = [h_1^c, ..., h_M^c] \in R^{M \times d}$$

5 Candidates Attend to Poly-Context

Candidates' embeddings query the compressed poly-context

$$h_{ctx} = Attention(y, H'_c, H'_c) \in R^d$$

- Result: single final context vector

$$h_{ctx}$$

6 Scoring

Compute the dot product between the candidate and the context

$$score = H_{ctx}^T \times y$$

7 Loss Function (Pairwise)

Binary cross-entropy loss:

- Compute predict scores from model s_A and s_B
- Compute difference score
$$s_\Delta = s_A - s_B$$
- Compute soft target from actual scores
$$y_\Delta = \frac{y_A}{y_A + y_B}$$
- Apply BCE
$$\mathcal{L} = -[y_\Delta \times log(\sigma(s_\Delta) + (1 - y_\Delta) \times log(1 - \sigma(s_\Delta)]$$

8 Backpropogation update

- Reset gradients
- Compute gradients
- Update all trianable parameters