تمرین کتبی شماره پنجم

عماد الدین رستمیان

① الف)

* windowsize=2 ⟵ این بار کلمات اول و آخر 4 جفت مثبت

تعداد جفت‌های مثبت = 4×6 + 3×2 + 2×2 = 34

2 جفت برای کلمه اول و آخر یا 3 جفت برای کلمه دوم و یکی مانده به آخر

هر جفت مثبت 5 نمونه منفی تولید می‌کند پس داریم: 34×5 = 170

تعداد کل نمونه‌ها: 34 + 170 = 204

4*

$d_{head} = d_{model} / h \longrightarrow W_Q, W_K, W_V \in \mathbb{R}^{d_{model} \times d_{head}}$ *

کاهش ابعاد از $d_{model} \times d_{model}$ به $d_{model} \times d_{head}$ باعث کاهش سربار محاسباتی هنگام ضرب ماتریس‌ها می‌شود. از طرفی موازی‌سازی باعث نمی‌شود صرف منابع نشده و سربار را افزایش می‌دهد.

ب) *صحیح

* غلط، با افزایش تعداد head‌ها ممکن است ویژگی‌های head‌ها مشابه تولید کنند.

② الف)

$$-\frac{\partial \log p(w_t|w_H)}{\partial v_{w_O}} = -\frac{\partial}{\partial v_{w_O}}\left(u_{w_t}^T v_{w_O} - \log \sum_{k \in V} \exp(u_{w_t}^T v_k)\right)$$

$$= -\left(u_{w_t} - \frac{\sum_k u_{w_t} \exp(u_{w_t}^T v_k)}{\sum_k \exp(u_{w_t}^T v_k)}\right) = \sum_k u_{w_t} p(w_k|w_t) - u_{w_t}$$

ب) برای $c=1$، مدل بیشتر بر روی روابط واژه‌های محلی با فاصله کوتاه متمرکز می‌کند. این باعث می‌شود تا مدل روابط دستوری بین کلمات را یاد بگیرد.

برای $c=100$ بعد بیشتری از کلمات در نظر گرفته می‌شود که باعث می‌شود تا مدل روابط معنایی بین کلمات را بهتر یاد بگیرد

$c=5$ یک حالت میانی و مصالحه‌ای بین دو حالت قبل است

$$\text{Output} = \text{Softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V$$

سوال ۱) الف)

$$\frac{QK^T}{\sqrt{d}} = \frac{1}{\sqrt{d}} X W_Q W_K^T X^T$$

برای اینکه به صورت یک شبکه FCN باشد باید $\frac{QK^T}{\sqrt{d}} = WXX^T$ پس داریم:

$$W \times X^T = \frac{1}{\sqrt{d}} X W_Q W_K^T X^T \xrightarrow{M = \frac{1}{\sqrt{d}} W_Q W_K^T} W \times X^T = XMX^T$$

$$W = f(M) \longrightarrow W_{ij} = \sum_{k,l} M_{kl} \cdot g(X_i, X_j) \longrightarrow \text{output} = \text{softmax}(W \times X^T)V$$

ب)

$$W_Q W_K^T \longrightarrow D \times D \rightarrow D^2$$

$$WXX^T \longrightarrow N \times N \rightarrow N^2$$

$$\longrightarrow D^2 N^2$$

ج)

$$A = \text{softmax}(QK^T)$$

$$A_{ij} = \text{softmax}(Q_i K_j^T)$$

بسیاری از عناصر ماتریس A به دلیل احتمال وزن کوچک مقدار نزدیک به صفر دارد که باعث می‌شود ماتریس Sparse می‌شود. همچنین اگر توکن‌ها وزن های یکسان برای توکن های دیگر داشته باشند درایه های مربوط به آن‌ها نیز صفر خواهد بود

د)

$$H_i = \text{Attention}(Q_i, K_i, V_i)$$

$$Q_i = XW_i^Q, \quad K_i = XW_i^K, \quad V = XW_i^V$$

حذف encoding و $W_i^Q = W_i^K = W_i^V \longrightarrow Q_i = K_i = V_i = X$

$$\text{Softmax}\left(\frac{XX^T}{\sqrt{d}}\right) \longrightarrow \text{redistribution of input}$$

$$\text{Softmax}\left(\frac{XX^T}{\sqrt{d}}\right)X \longrightarrow \text{rearranged input}$$

Input = (32, 2048, 768 : Encoder_1 (E)

V, K, Q = (32, 2048, 768)

Multihead Attention output : (32, 2048, 768)

Feed Forward layer1 : (32, 2048, 384)

Feed Forward Layer2 : (32, 2048, 768)

Input : (32, 2048, 768) : decoder

Self Attention output : (32, 2048, 768)

Cross Attention output : (32, 2048, 768)

Feed forward layer1 : (32, 2048, 384)

Feed forward layer2 : (32, 2048, 768)

Token Embedding : 30000 × 768 = 23040000 : Encoder_Y

Self Attention : 4 × 768 × 768 = 2359296

× 8 → 18874368

Feed Forward : { Layer1 : 768 × 384 = 294912
Layer2 : 768 × 384 = 294912 } += 589824

× 8 → 4718592

Layer Norm. : 2 × 768 × 8 = 1228

Total = 46645248

Decoder:

$12 \times 2359296 = 28311552$: Self Attention

$12 \times 2359296 = 28311552$: Cross Attention

Feed Forward: $12 \times 589824 = 7077888$

Layer Norm : $2 \times 768 \times 12 = 18432$

Total = 63719424

-٣

"He is sleeping" ⟶ $X_{input} = [Token_1, Token_2, Token_3]$ (He is sleeping)

embedding (32, 2048, 768) ⟶ encoder ⟶ Context representation

Decoder input : Y_{input} (ورودی مدل) = [SOS] ⟶ self ⟶ generate tokens

↳ cross ⟶ Best match for generating

⟶ "[SOS]" ⟶ "He" ⟶ "is" ⟶ "sleeping" ⟶ "و توانبره است [SOS]"

[EOS] ⟶ "[SOS] او توانبره است [EOS]"

(ه) آ) یک دلیل بدین کاهش هزینه محاسباتی است. دلیل دیگر Generalization بهتر است زیرا روابط بین جملات معمولا مختص به یک task یا موضوع خاص میشود. دلیل بعدی مشاهده تجربی است که نشان داده مدل هایی مانند RoBERTa که از NSP استفاده نمیکنند از مدل BERT کارا تر اند.

ب) Pooling بردارهای embedding را به یک بردار با طول ثابت مپ میکند. این بردار تضمین میکند که تمام کلمات داخل جمله را نشان دهد. درواقع این تک بردار معنی یک جمله را به صورت فشرده ذخیره میکند اهمیت

که در task هایی مانند Sentiment analysis میتواند بسیار پرکاربرد باشد.