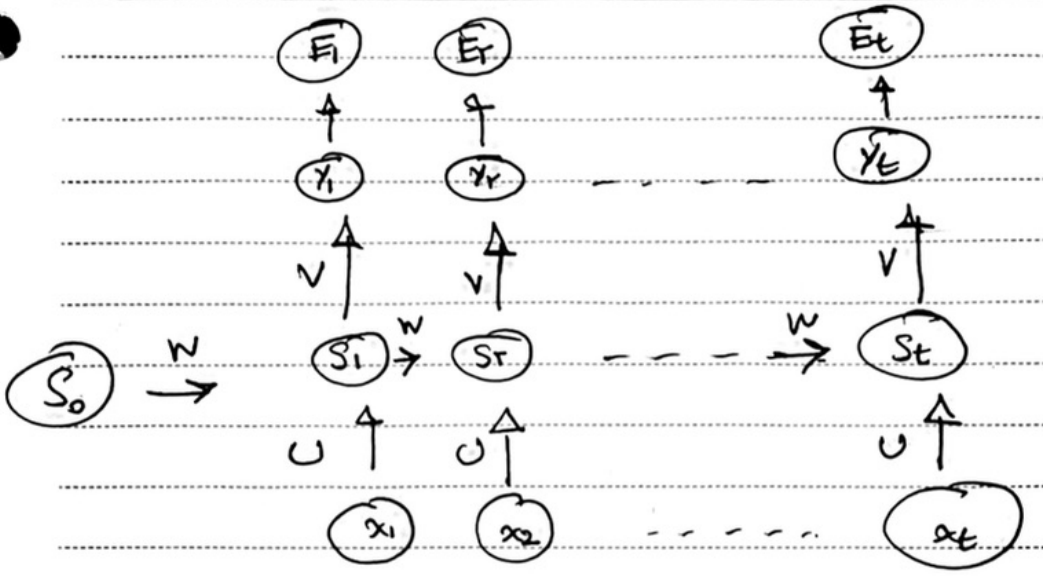
$$E_{total} = -\sum_t y_t \log \hat{y}_t$$

Subject: Date:



$$S_t = \tanh(W S_{t-1} + U x_t + b_S)$$

$$\hat{y}_t = \mathrm{Soft}(V S_t + b_y)$$

$$E_t = -y_t \log \hat{y}_t$$

First V: $\frac{\partial E_t}{\partial V_{ij}} = \frac{\partial E_t}{\partial \hat{y}_{tk}} \times \frac{\partial \hat{y}_{tk}}{\partial o_{tl}} \times \frac{\partial o_{tl}}{\partial V_{ij}}$

$\frac{\partial E_t}{\partial \hat{y}_{tk}} = \frac{y_{tk}}{\hat{y}_{tk}}$

$\frac{\partial \hat{y}_{tk}}{\partial o_{tl}} = \begin{cases} -\hat{y}_{tk}\hat{y}_{tl} & k \neq l \\ \hat{y}_{tk}(1-\hat{y}_{tk}) & k = l \end{cases}$

$$\Rightarrow \frac{\partial E_t}{\partial o_{tl}} = \frac{y_{tl}}{\hat{y}_{tl}} \times \hat{y}_{tl}(1-\hat{y}_{tl}) + \sum_{k \neq l} \left(\frac{y_{tk}}{\hat{y}_{tk}}\right)\left(-\hat{y}_{tk}\hat{y}_{tl}\right)$$

$$= \hat{y}_{tl} - y_{tl} \rightarrow \sum_k y_{tk} = 1 \text{ (one-hot)!}$$

$$\frac{\partial o_{tl}}{\partial V_{ij}} = \frac{\partial}{V_{ij}}\left(V_{lj} S_{tj}\right) = \delta_{il} S_{tj}$$

MICRO

Subject: ضرب خارجی Date:

$$\Rightarrow \frac{\partial E_t}{\partial V} = (\hat{y}_t - y_t) \otimes S_t \rightarrow$$ «K×d» میشه!

$$\Rightarrow \frac{\partial E}{\partial V} = \sum_t (\hat{y}_t - y_t) \otimes S_t$$

MICRO®

Subject: Date:

for W:
$$\frac{\partial E_t}{\partial W_{ij}} = \underbrace{\frac{\partial E_t}{\partial \hat{y}_{t_\ell}} \times \frac{\partial \hat{y}_{t_\ell}}{\partial q_{t_\ell}}} \times \underbrace{\frac{\partial q_{t_\ell}}{\partial S_{t_m}}} \times \frac{\partial S_{t_m}}{\partial W_{ij}}$$

از گسترش

$$= (\hat{y}^t_\ell - y^t_\ell)$$

$$\frac{\partial q_{t_\ell}}{\partial S_{t_m}} = \frac{\partial}{\partial S_{t_m}}\left(V_{\ell j} \cdot S_{t_j}\right) = V_{\ell m}$$

$$\frac{\partial S_{t_m}}{\partial W_{ij}} = \frac{\partial S_{t_m}}{\partial W_{ij}} + \frac{\partial S_{t_m}}{\partial S_{t-1}} \frac{\partial S_{t-1}}{\partial W_{ij}} + \frac{\partial S_{t_m}}{\partial S_{t-1}} \times 0 \times 0 + \dots$$

اولی همه ترم‌ها به جز S_{t-1} بقیه $\vec{0}$ است!

$$\Rightarrow \frac{\partial S_{t_m}}{\partial W_{ij}} = \sum_{r=0}^{t} \frac{\partial S_{t_m}}{\partial S_{r_n}} \times \frac{\partial S_{r_n}}{\partial W_{ij}}$$

$$\Rightarrow \frac{\partial E_t}{\partial W_{ij}} = (\hat{y}_{t_\ell} - y_{t\ell}) \times V_{\ell m} \times \sum_{r=0}^{t} \frac{\partial S_{t_m}}{\partial S_{r_n}} \frac{\partial S_{r_n}}{\partial W_{ij}}$$

$$\Rightarrow \frac{\partial E}{\partial W_{ij}} = \sum_{t} \frac{\partial E_t}{\partial W_{ij}}$$

البته می‌توان صورتی نشان داد جمع بست!

برای این کار باید tanh مشتق گرفته شده که در محاسبه Q تأثیر دارد

مشکل است!

MICRO®

Subject: Date:

$$\frac{\partial E_3}{\partial w_{ij}} = \frac{\partial E_3}{\partial s_3} \times \frac{\partial s_3}{\partial w} = \underbrace{\frac{\partial E_3}{\partial s_3}} \times \frac{\partial s_3}{\partial z_3} \times \frac{\partial z_3}{\partial w}$$

این را داریم!

$$(\hat{y}_{lt} - y_{lt}) \times V_{lm \times 3} \times (1 - S_3^2) \times S_{r\,j}$$

$$\Rightarrow \frac{\partial E_3}{\partial w} = V^T \odot (\hat{y}_3 - y_3) \odot (1 - S_3^2) \odot S_r^T$$

$(d \times d)$

این را چی می‌شود؟

$$\frac{\partial E_3}{w} = W^T \left(V^T \odot (\hat{y}_3 - y_3) \odot (1 - S_3^2) \right) \odot (1 - S_r^2) \odot S_r^T$$

$$\frac{\partial E_3}{\partial w} = W^T \left(W^T \left(V^T \odot (\hat{y}_3 - y_3) \odot (1 - S_3^2) \right) \odot (1 - S_r^2) \right) \odot (1 - S_1^2) \odot S_0^T$$

باسمه تعالی

دانشگاه صنعتی شریف

دانشکده مهندسی کامپوتیر

یادگیری عمیق

مقالات انتهایی سوال دوم

۱ سوال دوم

۱.۱ الف

تابع $attention$ در واقع یک نگاشت از یک $query$ و تعدادی پیر $key - value$ از ورودی به خروجی میباشد، که همگی بصورت بردار هستند.

۲.۱ ب

اگر بعد مقداری key ها زیاد شود، ضرب نقطه ای هزینه محاسباتی و عملیاتی زیادی خواهد داشت و این بعث میشود تابع سافت مکس به گونه ای اشباع شود ، که برای حل این مشکل خروجی را تقسیم بر $\sqrt{d_k}$ میکنیم .

۳.۱ ج

۱. در بخش دیکود کردن : $query$ ها را از لایه دیکودر قبل و پیر های کی ولیو از ایه انکودر بدست آورده میشوند که باعث دسترسی $global$ به همه ورودی ها داشته باشد.

۲. در هر انکودر همه ورودی ها از لایه قبلی انکودر آمده اند. و در هر دیکودر نیز همینطور

۴.۱ د

برای اینکه ترتیب ورودی ها مشابه نگه داشته شود، بجای استفاده از لایه های کنولوشنی یا $recurrent$ از مفهومی به نام $positionalencoding$ استفاده میشود، در این کار با توجه به موقعیت هر کلمه در جمله، به هر کلمه یک بردار با بعد $wordembedding$ نسبت داده میشود که فرمول زیر را ارضا میکند : یعنی هر موقعیت کلمه در جمله تناظر یک به یک با یک تابع سینوسی با فرکانس متفاوت دارد، شبکه ارتباط

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

بین کلمه ها را با این بردار ها یاد خواهد گرفت.
راهکار دیگر نسبت دادن یک بردار به هر موقعیت است که تفاوت آن با قلبلی قابل یادگیری بودن است که نتایج هر دو تقریبا مشابه است همانطور که در سرط E جدول زیر آمده است :

	N	d_{model}	d_{ff}	h	d_k	d_v	P_{drop}	ϵ_{ls}	train steps	PPL (dev)	BLEU (dev)	params $\times 10^6$
base	6	512	2048	8	64	64	0.1	0.1	100K	4.92	25.8	65
(A)				1	512	512				5.29	24.9	
				4	128	128				5.00	25.5	
				16	32	32				4.91	25.8	
				32	16	16				5.01	25.4	
(B)					16					5.16	25.1	58
					32					5.01	25.4	60
(C)	2									6.11	23.7	36
	4									5.19	25.3	50
	8									4.88	25.5	80
		256			32	32				5.75	24.5	28
		1024			128	128				4.66	26.0	168
			1024							5.12	25.4	53
			4096							4.75	26.2	90
(D)							0.0			5.77	24.6	
							0.2			4.95	25.5	
								0.0		4.67	25.3	
								0.2		5.47	25.7	
(E)	positional embedding instead of sinusoids									4.92	25.7	
big	6	1024	4096	16			0.3		300K	**4.33**	**26.4**	213

۵.۱ ه

سه معیار برای این مقایسه بررسی میشود، اول مقدار پیچیدگی محاسباتی در هر لایه، دوم مقدار محاسبه با قابلیت پردازش موازی و سوم ارتباط بین ورودی های با فاصله زیاد از هم. در رابطه با مورد سوم یکی از مشکلات بزرگ شبکه های پردازش زبان این است که با دور شدن استپ های زمانی از هم ارتباط آنها از هم کاهش میابد که منجر به کاهش قدرت ماشین در پردازش جمله های طولانی میشود که در جدول زیر این سه معیار مقایسه شده اند:

Table 1: Maximum path lengths, per-layer complexity and minimum number of sequential operations for different layer types. n is the sequence length, d is the representation dimension, k is the kernel size of convolutions and r the size of the neighborhood in restricted self-attention.

Layer Type	Complexity per Layer	Sequential Operations	Maximum Path Length
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(log_k(n))$
Self-Attention (restricted)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

۶.۱ و

بخش انکودر شامل ۶ لایه مشابه است که هر یک دو زیر لایه دارا هستند، اولی یک $self - attention$ چندسر و دومی یک شبکه فولی کانکتد. در هر زیر لایه از یک $skipconnection$ استفاده شده است که با خروجی هر زیر لایه ورودی را جمع می‌کند، بعد از هر زیر لایه نیز یک $layernormalization$ قرار داده شده. بعد تمام خروجی ها ثابت و برابر با 512 است
در دیکودر نیز ۶ لایه مشابه بالا داریم که زیر لایه های مشابه با بالا نیز دارد و علاوه بر این دو یک لایه $self attention$ چند سر دیگر نیز قرار داده شده است که به خروجی های انکودر متصل است و نهایتا $skip - connection$ ها و $layer - normalization$ های مشابه نیز داریم .
بهینه ساز مورد استفاده مقاله ادام است با پرامتر های $\epsilon = 10^{-9}, \beta_1 = 0.9, \beta_2 = 0.98$ و نرخ یادگیری متغیر با زمان با فرمول به شکل زیر داراست :

$$lrate = d_{\text{model}}^{-0.5} \cdot \min(step\_num^{-0.5}, step\_num \cdot warmup\_steps^{-1.5})$$

از دو $regularizer$ متفاوت استفاده شده، اول دراپ اوت با نرخ 0.1 که یعنی یک دهم وزن ها را در هر ایتریشن دور میندازد، که هم در خروجی زیر لایه ها قرار دارد و هم در هر دو قسمت دیکود و انکود پس از حاصل جمع امبدینگ کلمه ها امدینگ موقعیت ها.
همچنین لیبل اسموسینگ با پرامتر $\epsilon = 0.1$ نیز استفاده شده است..

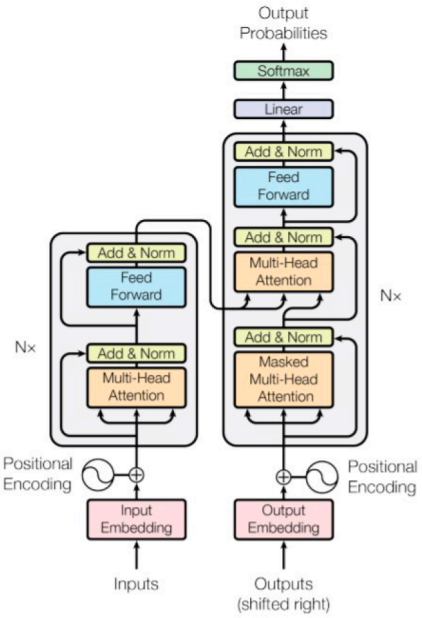
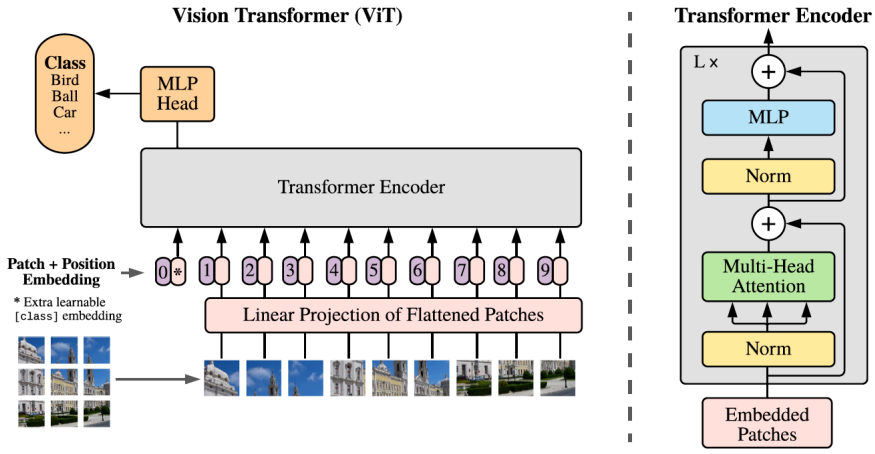


Figure 1: The Transformer - model architecture.

۷.۱ ز

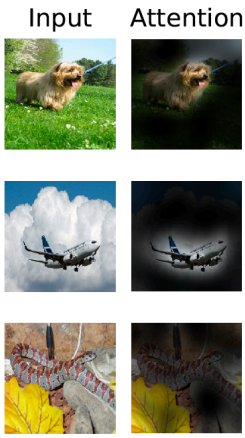
در این مقاله با توجه به کاربرد روزافزون مدل های پردازش زبان های طبیعی و اینکه هر حرف را بعنوان یک توکن در نظر میگیرند ، تلاش شده تا با تقسیم یک عکس بچ های مختلف هر کداک ا ز این تیکه ها را بعنوان یک توکن در نظر گرفته ، سپس یک امبدینگ خطی روی آنها زده، ایندکس هر یک را کنرش گذشاته و وارد یک ساختار ترنسفورمر میکنید که به شکل زیر است ، و در نهایت برای امر کسلیفیکیشن آن را ترین میکند که نیازمند لیبل متنظار بالا سمت چپ تصویر میباشد، :



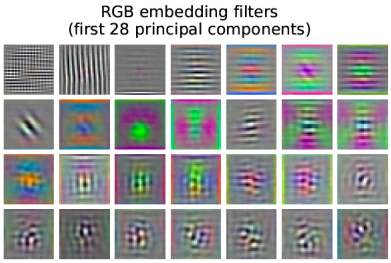
حال اگر فرض شود یک عکس با ابعاد $(H \times W \times C)$ داشته باشیم ، ان را به N بچ با ابعاد $(P \times P)$ تقسیم میکنیم که نهایتا به ابعاد زیر در میاید : $(N \times (P^2C))$
نهایتا روی این بچ ها یک امبدینگ صورت میگیرد تا هر کدام به یک بعد D تصویرد بشوند ، که رابطه آن به شرح زیر است ؛

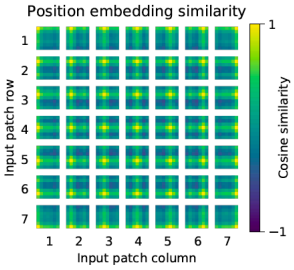
$$\mathbf{z}_0 = [\mathbf{x}_{\text{class}}; \mathbf{x}_p^1\mathbf{E}; \mathbf{x}_p^2\mathbf{E}; \cdots ; \mathbf{x}_p^N\mathbf{E}] + \mathbf{E}_{pos}, \qquad \mathbf{E} \in \mathbb{R}^{(P^2 \cdot C) \times D}, \mathbf{E}_{pos} \in \mathbb{R}^{(N+1) \times D} \qquad (1)$$

که X_{class} بعنوان $classificationHead$ شناخته میشود. .
همچنین برای جلوگیری از مشکل $image - specificinductivebias$ که بر خلاف مدل های کانولوشنی برای مدل ما رخ میدهد سعی میشود تا وردیلایه امبدینگ خطی بجای بچ های تصویر اصلی ، بچ های استخراج شده پس از گذر از یک لایه کانولوشنی باشند .
در ابتدای لایه ترنسفورمر ، بچ ها به یک بعد پایین تر کاهش داده میشوند، و سپس یک امدینگ پوزیشن با آنها اضافه میشود، که قابل ذکر است بچ های به هم نزدیک تر پوزیشن امبدینگ های مشابهی خواهند داشت، و همچنین $self - aattention$ که در لایه ترنسفورمر قرار داده شده است طبق عکس زیر اطلاعات سرتاسر عکس را جمع میکند.



همچنین مفاهیم مثل امبدینگ پوزیشن و امبینگ فیتر ها برای خروجیشان مثال های زیر آمده است :





نهایتا این ساختار با پری ترین شدن روی دیتاست های بزرگ به نتایج قابل توجه و بعضا بهتری از مدل های روز دنیا میرسد.