Synthesizer

Rethinking Self-Attention in Transformer Models

前言

- 随着Transformer的提出，NLP领域得到了极速的发展，在Transformer中self-attention通过自
- 己与自己的Attention来捕捉token与token之间的关联，而事实上上self-attention真的是我们
- 所理解的那样吗？而本文正是对自注意力的一个反思与探讨.

Self-attention

- 首先我们回顾一下self-attention的机制，self-attention的关键在于query-key-product 的点积注意力，即

$$Attention(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = softmax\left(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{d_k}}\right)\boldsymbol{V}$$

- 在其中各个参数的维度为 $\boldsymbol{Q} \in \mathbb{R}^{n\times d}, \boldsymbol{K} \in \mathbb{R}^{m\times d}, \boldsymbol{V} \in \mathbb{R}^{m\times d}$ 最后通过 $softmax$ 进行归一化计
- 算， 我们把 $\boldsymbol{X} \in \mathbb{R}^{n\times d}$ 通过不同的投影矩阵 $\boldsymbol{W}_q, \boldsymbol{W}_k, \boldsymbol{W}_v \in \mathbb{R}^{d\times d'}$ 得到 $\boldsymbol{Q} = \boldsymbol{X}\boldsymbol{W}_q, \boldsymbol{K} = \boldsymbol{X}\boldsymbol{W}_k, \boldsymbol{V} = \boldsymbol{X}\boldsymbol{W}_v$
- 之后在做Attention

$$\begin{aligned} SelfAttention(\boldsymbol{X}) &= Attention(\boldsymbol{X}\boldsymbol{W}_q, \boldsymbol{X}\boldsymbol{W}_k, \boldsymbol{X}\boldsymbol{W}_v) \\ &= softmax\left(\frac{\boldsymbol{X}\boldsymbol{W}_q\boldsymbol{W}_k^\top\boldsymbol{X}^\top}{\sqrt{d_k}}\right)\boldsymbol{X}\boldsymbol{W}_v \end{aligned}$$

Synthesizer attention实验

- 从公式中可以看出Self-Attention是将一个 $n \times n$ 的矩阵A通过 $d \times d'$ 的矩阵 W_v 将原本的矩阵
- A 变成了 $n \times d'$ 的 B 而作者的第一个实验被称为Dense形式，首先移除了 K, Q, V
- 的形式，而是直接通过了两层的Dense形式 $B = \text{relu}\left(XW_1 + b_1\right)W_2 + b_2$ 得到了矩阵 B
- 相当于将 K 固定为矩阵 $W^\top$ 这其中W跟b用于 Dense 层的计算 最后输出结果为
- $\mathbf{Y} = \text{Softmax}(\mathbf{B})G(\mathbf{X})$ $G(\mathbf{X})$ 可类比于 Transformer中的V 。
- 第二种形式为，注意力权重的初始化不在受到任何输入 token 的影响，而是完全随机初始化。
- 这些初始化的值可以保持不更新，或者随着训练保持更新，此时另 $B = R$ 则 $\mathbf{Y} = \text{Softmax}(\mathbf{R})G(X)$
- 此时不依赖 任何token 对之间的交互或者任何单个 token 的信息。如图

如图

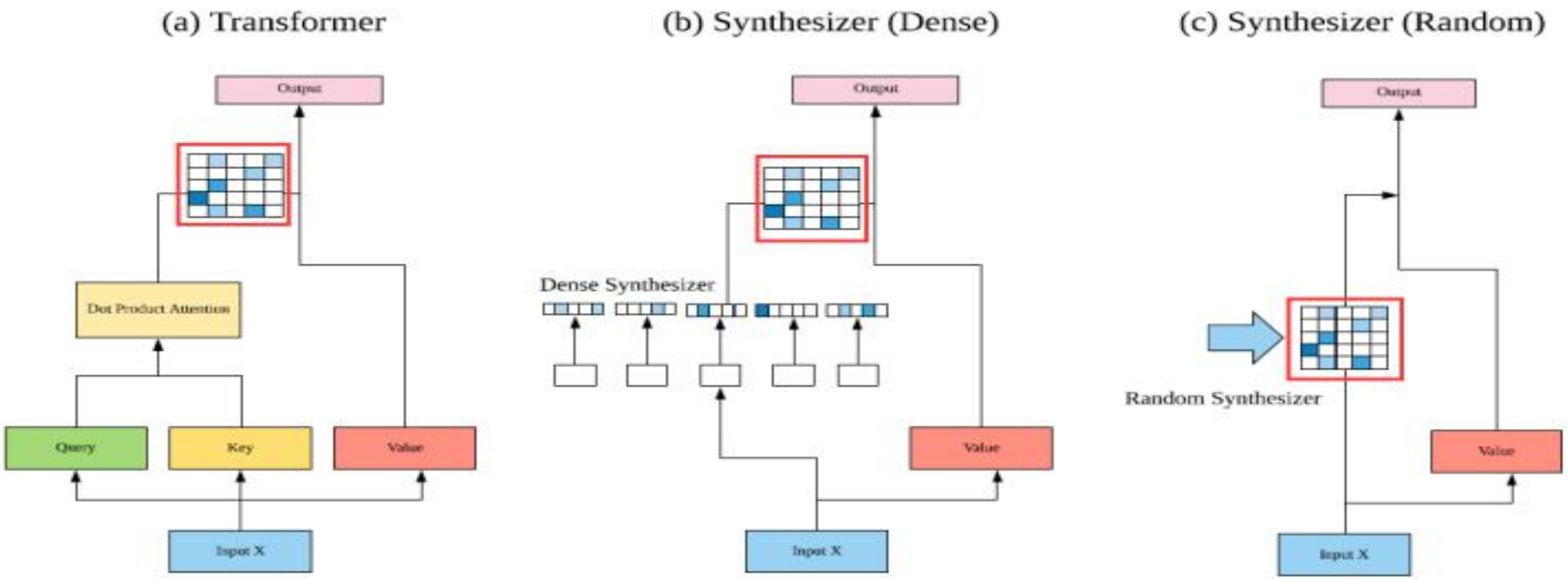

Figure 1: Our proposed SYNTHESIZER model architecture.

模型分解

- 上面的形式，往往带来了过多的参数，为了实践可行，作者提出了两种分解模型的形式，
- 被称为Factorized Dense Synthesizer 和 Factorized Random Synthesizer，
- Factorized Dense Synthesizer：通过Dense的方式，生成两个 $n \times a, n \times b$ 的矩阵 $\boldsymbol{B}_1, \boldsymbol{B}_2$
- 其中 $ab = n$ 令 $\boldsymbol{B}_1$ 重复 b 次，令 $\boldsymbol{B}_2$ 重复 a 次得到 $n \times n$ 的矩阵，然后令两个矩阵做逐位相
- 乘，即可从分解出的两个矩阵恢复到 $\boldsymbol{B}$ 。
- Factorized Random：令 $n \times n$ 的矩阵 $\boldsymbol{R}$ 变成两个 $n \times l$ 的矩阵，$\boldsymbol{R}_1$ $\boldsymbol{R}_2$ ，然后

$$\boldsymbol{B} = \boldsymbol{R}_1 \boldsymbol{R}_2^\top$$

混合模式：

上述所有提出的 Synthetic 注意力变种都可以通过加和形式混合在一起 $\mathbf{Y} = \text{Softmax}(\alpha_1 S_1(\mathbf{X}) + \cdots + \alpha_N S_N(\mathbf{X}))G(\mathbf{X})$ 其中 $S(\cdot)$ 表示一个 Synthesizer 的函数，而 $\sum \alpha = 1$

机器翻译：

- 表格结果显示，除了固定的Random外，所有的自注意力形式表现基本上都差不多，而且就算是固定的Random也有看得过去的效果，
- 这表明以往对自注意力的认知和解释都太过片面，并没有揭示自注意力生效的真正原因。

	NMT (BLEU)			LM (PPL)	
Model	# Params	EnDe	EnFr	# Params	LM1B
Transformer [Vaswani et al., 2017]	68M	27.30	38.10	-	-
Transformer (Our run)	68M	27.67	41.57	70M	38.21
Transformer (Control)	73M	27.97	41.83	-	-
Synthesizer (Fixed Random)	61M	23.89	38.31	53M	50.52
Synthesizer (Random)	67M	27.27	41.12	58M	40.60
Synthesizer (Factorized Random)	61M	27.30	41.12	53M	42.40
Synthesizer (Dense)	62M	27.43	41.39	53M	40.88
Synthesizer (Factorized Dense)	61M	27.32	41.57	53M	41.20
Synthesizer (Random + Dense)	67M	27.68	41.21	58M	42.35
Synthesizer (Dense + Vanilla)	74M	27.57	41.38	70M	**37.27**
Synthesizer (Random + Vanilla)	73M	**28.47**	**41.85**	70M	40.05

Table 2: Experimental Results on WMT'14 English-German, WMT'14 English-French Machine Translation tasks and Language Modeling One Billion (LM1B).

摘要对话

- 在自动摘要这个任务上，标准注意力效果比较好；但是对话生成这个任务上，结果则反过来：标准的自注意力是最差的，Dens
- （D）和Random（R）是最好的，而当Dense和Random混合了标准的自注意力后（即 D+V 和 R+V），效果也变差了。这说明标准注
- 意力并没有什么“独占鳌头”的优势，而几个Synthesizer看起来是标准注意力的“退化”，但事实上它们互不从属，各有优势

	Summarization			Dialogue				
Model	Rouge-1	Rouge-2	Rouge-L	Bleu-1/4	Rouge-L	Meteor	CIDr	Emb
Transformer	38.24	**17.10**	35.77	12.03/3.20	13.38	5.89	18.94	83.43
Synthesizer (R)	35.47	14.92	33.10	14.64/2.25	15.00	6.42	19.57	84.50
Synthesizer (D)	36.05	15.26	33.70	**15.58/4.02**	**15.22**	**6.61**	**20.54**	**84.95**
Synthesizer (D+V)	38.57	16.64	**36.02**	14.24/3.57	14.22	6.32	18.87	84.21
Synthesizer (R+V)	**38.57**	16.24	35.95	14.70/2.28	14.79	6.39	19.09	84.54

Table 3: Experimental results on Abstractive Summarization (CNN/Dailymail) and Dialogue Generation (PersonaChat).

预训练+微调

- 从表中结果可以看出，相比标准自注意力，Dense和Random就显得逊色了，这表明Dense和Random也许会在单一任务上表现得比
- 较好，而迁移能力则比较弱。但是不能否定的是，像Random这样的自注意力，由于直接省去了 $QK^\top$ 这个矩阵运算，因此计
- 算效率会有明显提升。

Model	Glue	CoLA	SST	MRPC	STSB	QQP	MNLI	QNLI	RTE
T5 (Base)	83.5	53.1	**92.2**	**92.0/88.7**	89.1/88.9	88.2/91.2	84.7/**85.0**	91.7	76.9
Syn (R)	75.1	41.2	91.2	85.9/79.4	74.0/74.3	85.5/89.0	77.6/78.1	87.6	59.2
Syn (D)	72.0	18.9	89.9	86.4/79.4	75.3/75.5	85.2/88.3	77.4/78.1	86.9	57.4
Syn (D+V)	82.6	48.6	92.4	91.2/87.7	88.9/89.0	88.6/91.5	84.3/84.8	91.7	75.1
Syn (R+V)	**84.1**	**53.3**	**92.2**	91.2/87.7	**89.3/88.9**	**88.6/91.4**	**85.0**/84.6	**92.3**	**81.2**

Table 4: Experimental results (dev scores) on multi-task language understanding (GLUE benchmark) for *small* model and `en-mix` mixture. Note: This task has been co-trained with SuperGLUE.

Model	SGlue	BoolQ	CB	CoPA	MultiRC	ReCoRD	RTE	WiC	WSC
T5 (Base)	70.3	78.2	72.1/83.9	59.0	73.1/32.1	**71.1/70.3**	77.3	**65.8**	**80.8**
Syn (R)	61.1	69.5	54.6/73.2	60.0	63.0/15.7	58.4/57.4	67.5	64.4	66.3
Syn (D)	58.5	69.5	51.7/71.4	51.0	66.0/15.8	54.1/53.0	67.5	65.2	58.7
Syn (D+V)	69.7	79.3	74.3/85.7	64.0	73.8/33.7	69.9/69.2	78.7	64.3	68.3
Syn (R+V)	**72.2**	**79.3**	**82.7/91.1**	**64.0**	**74.3/34.9**	70.8/69.9	**82.7**	64.6	75.0

Table 5: Experimental results (dev scores) on multi-task language understanding (SuperGLUE benchmark) for *small* model and `en-mix` mixture. Note: This task has been co-trained with GLUE.