告别自注意力，谷歌为Transformer打造新内核Synthesizer

原创 舒意恒 夕小瑶的卖萌屋 6月9日

来自专辑
卖萌屋@自然语言处理

一只小狐狸带你解锁 **炼丹术&NLP** 秘籍

作者：舒意恒（南京大学硕士生，知识图谱方向）

今天给大家介绍一篇来自Google的最新论文《**SYNTHESIZER: Rethinking Self-Attention in Transformer Models**》[4]，该论文重新探索了Transformer中注意力机制的必要性，并引入了新的attention计算方法Synthesizer。实验显示，即使不进行token之间的attention交互计算，synthesizer在翻译、语言模型、GLUE等任务上也可以达到很好的效果。

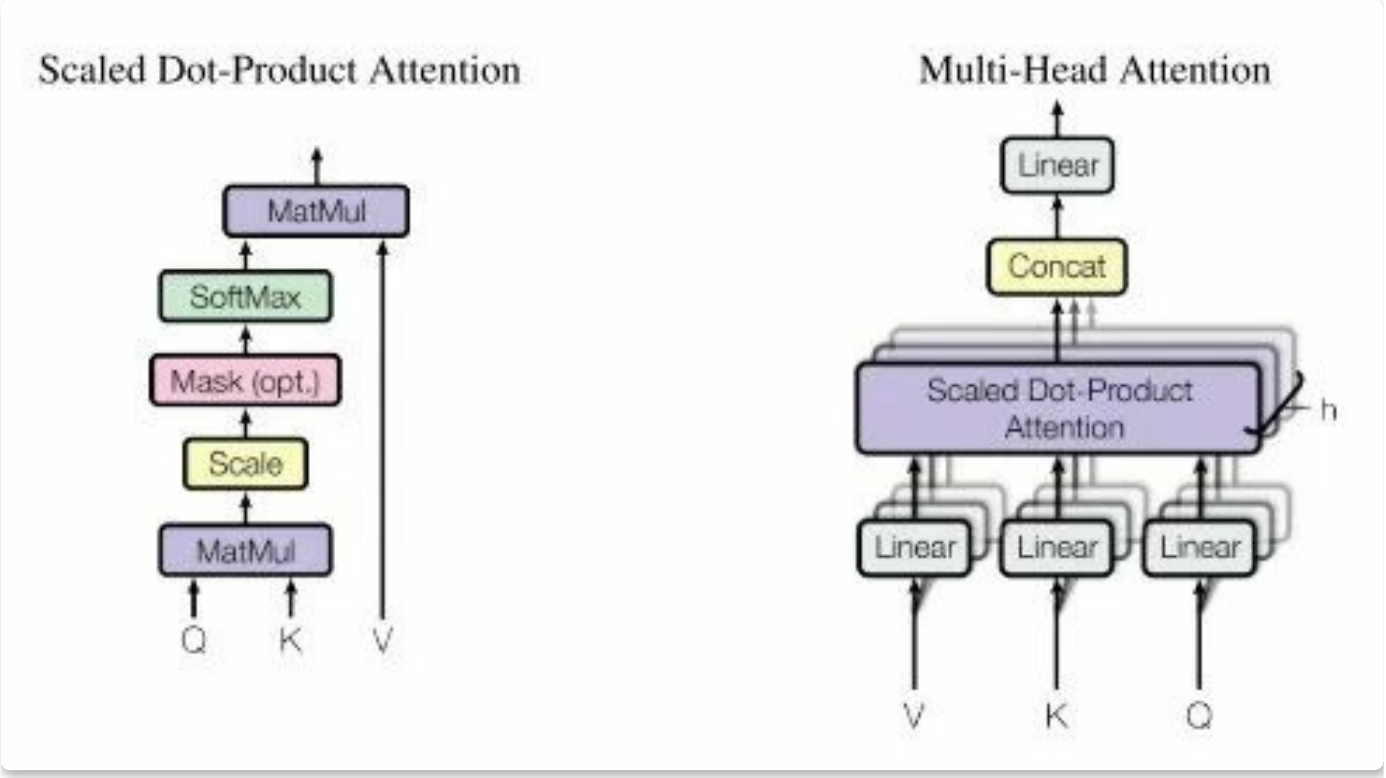
前言

什么是自注意力?

2017 年，Vaswani 等人 [1] 提出了 Transformer 模型，在众多领域取得了成功的应用，证明了它相比于自卷积模型和循环模型的优势。

Transformer 的关键在于 query-key-product 的点积注意力，token 被完全连接，能够对远距离的依赖关系进行建模。

Transformer 存在的问题



点积自注意力提供了强大的建模能力，但同时作者对点积自注意力提出了质疑，它的计算可能是不必要的。

点积的基本作用是确定单个 token 相对于序列中所有其他 token 的相对重要性。key、query、value 暗示自注意力模拟一个 **基于内容的检索过程**，过程的核心是 pairwise 的交互。该文对整个过程进行了反思。

技术简介

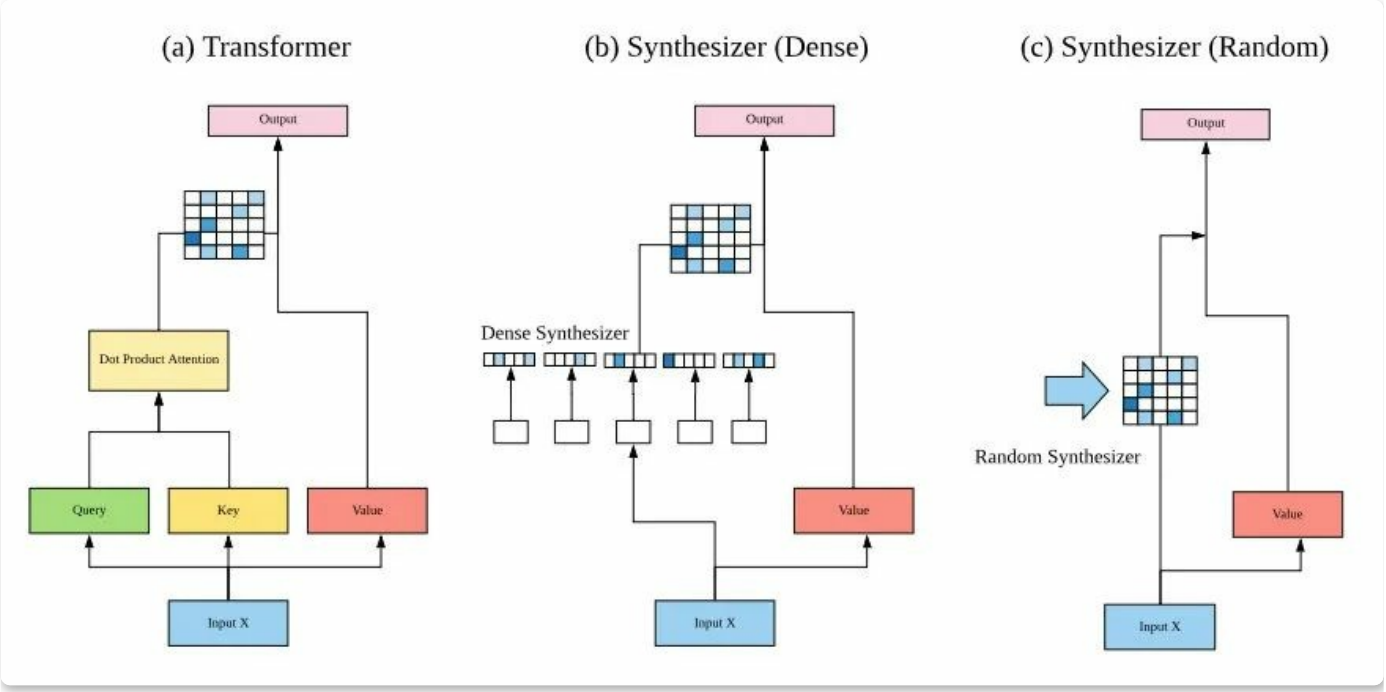
Synthesizer 的关键思想

Synthesizer 的核心思想是用低复杂度的attention计算代替dot product式的注意力机制。传统 Transformer 的注意力机制需要进行 token 之间的两两交互，虽然可以获得更多的交互信息，但attention score会很依赖实例，难以保证模型学到更多的泛化特征。

因此，synthesizer提出了新的attention matrix学习方法，只通过简单的前馈神经网络即可得到注意力分数，完全省去了token之间的点积交互。

实现细节

Synthesizer 大体上可以理解为一个 Transformer，其中自注意力模块被 Synthetic Attention 模块替换。



上图表示了 Transformer、Dense Synthesizer 和 Random Synthesizer 的关键思想。

Synthesizer 移除了 query-key-value 的概念，而直接合成一个对齐矩阵。具体而言，即去除了 K、Q、V，而使用一个行和列大小等同于序列长度 l的矩阵 B来表示任意 token 之间的关系。作者提出了两类 synthesizer，分别是 Dense Synthesizer 和 Random Synthesizer。

Dense Synthesizer

给定模型的输入 X，表示了 l个 token，每个 token 的维度为 d. 该方法做如下的变换：

$$B = W(\mathrm{ReLU}(W(X) + b)) + b$$

这可以理解为两个 Dense 层，W和 b用于 Dense 层的计算。而最后模型的输出 Y，由表示 token 间关系的矩阵 B得到。

$$Y = \mathrm{Softmax}(B)G(X)$$

其中，$G(X)$可类比为标准 Transformer 的 V.

该思路并不复杂，但是，作者进一步描述了 Random Synthesizer。

Random Synthesizer

Dense Synthesizer 方法实际上是给定每个 token，然后映射到 l维，而不是如同原生的 Transformer 对 token 间交互进行建模。Random Synthesizer 的方法中，注意力权重的初始化不是受任何输入 token 的影响，而是完全随机初始化。这些随机初始化的值可以被训练，或者保持固定。

以 R表示一个随机初始化矩阵，则 Random Synthesizer 被定义为：

$$Y = \mathrm{Softmax}(R)G(X)$$

即 B初始化的值是 R. 该方法不依赖 token 对之间的交互或者任何单个 token 的信息，而是学习一个能跨实例有效的任务特定的对齐。作者表示这是最近固定自注意力方法 [2]的直接产物。

换句话说，作者认为，学习一个跨实例有效的模型意味着在初始化时不直接依靠任何 token 信息。

分解模型

Dense Synthesizer 为网络添加了大小为 $d \times l$的参数，用于映射；而 Random Synthesizer 添加了大小为 $l \times l$的参数。如果序列很长，将导致很大的参数量。因此，为了实践中更加可行，作者提出了分解模型，分别针对 Dense Synthesizer 和 Random Synthesizer 称为 Factorized Dense Synthesizer 和 Factorized Random Synthesizer。该方法的作用是减少参数量，并防止过拟合。

1. Factorized Dense Synthesizer

针对可能过大的序列长度 l，取两个整数 a和 b使得 $ab = l$，分别按 Dense Synthesizer 算得两个矩阵记为 B_1和 B_2，两矩阵大小分别是 $l \times a$和 $l \times b$. 然后将 B_1中表示 token 的每个向量重复 b次，将 B_2中表示 token 的每个向量重复 a次，再做元素积，即可从分解出的两个矩阵恢复到 B. 参数量减小，同时模型的表征能力可能也受到了影响。

2. Factorized Random Synthesizer

类似地，随机矩阵也可被分解为两个低秩矩阵。

混合模型

上述所有提出的 synthetic 注意力变种都可以通过加和形式混合到一起。

$$Y = \mathrm{Softmax}(\alpha_1 S_1(X) + \cdots + \alpha_N S_N(X))G(X)$$

其中，$S(\cdot)$表示一个 synthesizer 的函数，并且 $\sum \alpha = 1$，α是可学习的。

另外，类似于原生 Transformer 的 multi-head attention，Synthesizer 同样支持多个 head.

效果

作者在机器翻译、语言模型、文本生成、多任务自然语言处理等任务上进行了实验。

机器翻译与语言建模

作者采用常见的 WMT'14 英德(EnDe)和英法(EnFr)翻译测试。

Model	NMT (BLEU) # Params	EnDe	EnFr	LM (PPL) # 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

关于机器翻译任务，可以看到相同参数量的 Synthesizer(Random + Vanilla) 与其他模型拉开了一定差距，也比相同参数量的 Transformer (Control) 表现更好。值得注意的是，两个分解方法取得的提升并不如混合模型取得的提升更多，但在一定程度上减少了参数量。

关于语言建模任务，使用的数据集是 LM1B，取得最好效果的是 Synthesizer (Dense + Vanilla)，它仍然是一个混合模型，同样是 Synthesizer 的各种设置中唯一超过 Transformer 的模型。

文本生成

评测使用 CNN/Dailymail 数据集的抽象摘要任务和使用 PersonaChat 数据集的对话生成任务。其中，Synthesizer 的各个模型表现不一。

Model	Summarization Rouge-1	Rouge-2	Rouge-L	Dialogue 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

多任务 NLP

在多任务 NLP 上，作者遵循 T5 [3] 所使用的使用 GLUE 和 SuperGLUE 评测方法，并在多项指标超过了 T5(base)。在众多测试中，仍然是加上 Vanilla 的 Synthesizer 取得较好效果。

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.

总结

该文提出了 Synthesizer，一个新的 Transformer 模型，它采用了合成注意力（Synthetic Attention）。作者试图更好地理解和评估全局对齐和局部、实例对齐（独立 token 和 token 到 token 的对齐）在自注意力中的效用。

在机器翻译、语言建模和对话生成等多个任务上，合成注意力与原有自注意力相比，表现出有竞争力的性能，点积自注意力的作用与效率值得怀疑与进一步研究。此外，在对话生成任务上，token 之间的交互信息实际会损害性能。

实际上，Synthesizer 的不同设置没有绝对的优劣，也和具体的任务相关。个人认为为何在各种任务中 Synthesizer 的表现存在明显差异、它与点积注意力分别适合于哪些任务，以及背后的成因是值得深究的。

参考文献

[1] Attention Is All You Need. arXiv preprint arXiv:1706.03762, 2017.

[2] Fixed encoder self-attention patterns in transformer-based machine translation. arXiv preprint arXiv:2002.10260, 2020.

[3] xploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. arXiv preprint arXiv:1910.10683, 2019.

[4] Tay Y, Bahri D, Metzler D, et al. Synthesizer: Rethinking Self-Attention in Transformer Models[J]. arXiv preprint arXiv:2005.00743, 2020.

- NLP中的少样本困境问题探究
- ACL20 | 让笨重的BERT问答匹配模型变快！
- 7款优秀Vim插件帮你打造完美IDE
- 卖萌屋原创专辑首发，算法镇魂三部曲！
- GPT-3诞生，Finetune也不再必要了！NLP领域又一核弹！

夕小瑶的卖萌屋

关注&星标小夕，带你解锁AI秘籍

订阅号主页下方「撩一下」有惊喜哦