异戊烯基转移酶关于Transformer超参数的研究

异戊烯基转移酶
中国科学院大学

1. 任务定义

机器翻译，是指使用计算机将一种自然语言转换为另一种自然语言的过程。这里，自然语言是指日常使用的人类语言（如中文、英语），区别于人工为特定目的创造的语言（如计算机编程语言）。

这是一个颇有历史的自然语言处理任务。从二十世纪三十年代G. B. Arsouni提出的机器脑（mechanical brain）开始，使用机器来进行语言翻译的方法就在不断演进。

2017年，谷歌在论文《Attention Is All You Need》(Vaswani et al. 2017)中提出了Transformer模型。这种基于自注意力机制的模型能够很好地解决序列模型的问题，比如机器翻译。Transformer应用在机器翻译任务中，不仅提高了翻译的效果，由于其高度并行化的设计，还大幅提高了训练的效率。我们小组的实验内容就是以Transformer为核心实现一个机器翻译模型，并且讨论其中各种超参数对最终模型的影响。

2. 小组分工

- 李润宽
    1. 数据预处理；
    2. LSTM，CNN，Transformer模型的实现；
    3. Transformer超参数评估实验的设计；
    4. 实验报告的撰写。
- 缪涵
    1. 实验环境的配置、模型训练和超参数评估实验的实施；
    2. 前端界面的编写和web服务的搭建。

3. 模型描述

我们小组在算法选取上走了一些弯路。最开始希望对几个较旧的神经网络机器翻译模型进行对比，例如基于LSTM或GRU建立encoder和decoder，利用这个seq2seq模

型实现机器翻译。但是这些模型即使使用了attention机制，在英文到中文的翻译任务上表现得也不好。于是我们又在encoder和decoder部分尝试采用了CNN，也就是Facebook于2017年的论文《Convolutional Sequence to Sequence Learning》(Gehring et al. 2017)中的方法，以及Transformer，也就是Google于2017年晚些时候发表的著名论文《Attention Is All You Need》(Vaswani et al. 2017)中的方法。最终的结果表明，Transformer确实是有显著优势的。至此我们才确定了我们最终的实验内容：对Transformer模型的超参数进行比较细致的调试，希望获得Transformer超参数的一个较为合理的搭配。

接下来是我们使用Transformer进行机器翻译的方法描述。

3.1 Transformer中的attention机制

Attention是Transformer的核心机制，本小节对其简单介绍。

Attetion可以理解为一个映射或者函数，有三个输入值，向量query、矩阵key和矩阵value。其中query与key进行一定运算得到一个注意力向量（attention vector），该向量一般是经过正规化（normalization）处理，诸元素为正实数且和为1。这使其具有权重（或概率）的性质，可以与value中的每个列向量进行加权，得到一个新的矩阵作为attention这个映射的输出。在Transformer中的attention机制可以表达为式1，此式是将多个query组成矩阵同时计算attention：

$$attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{1}$$

Transformer中的attention机制使用的正规化方法是softmax，这也是一种在神经网络中常见的处理输出的函数。$\sqrt{d_k}$用于缩小QK^T值的规模。这种按比例缩小可以使得softmax的结果更加平滑，也就是各个元素之间的差值更小，不会出现一个值很接近1而其他所有值都几乎为0的情况。实际上这是softmax中的temperature参数。因为其中的点乘和缩放机制，所以这种attention被称为scale dot-product attention。

另外Transformer的attention机制是一个多头（multi-head）attention机制。它会将Q、K和V分成等大小的几份，分别计算scale dot-product attention，最后再将结果拼接起来。这个过程可以表达为式2：

$$\begin{aligned} head_i &= attention(QW_i^Q, KW_i^K, VW_i^V) \\ multihead(Q, K, V) &= concat(head_1, \cdots, head_h)W^O \end{aligned} \tag{2}$$

式2中，h表示了Q、K和V分成的份数，也就是多头注意力机制中的“头”数，$concat$表示对矩阵的拼接（concatenate）操作。$W_\cdot^\cdot$表示全连接层的权重矩阵，其中是可训练参数。

3.2 Transformer的encoder部分

Transformer的encoder部分中，输入序列inputs在经过embedding层之后与positional embedding的结果进行求和运算（它们的向量维度一致），然后作为真正的输入进入encoder部分。整个encoder部分会重复n次，每次会将上一次的输出作为输入，n是一个超参数。Encoder内部由两个残差（residual）网络组成，第一个的残差部分是多头attention，第二个的残差部分是一个前馈神经网络（feed forward）。多头attention部分的Q、K和V都是复制自encoder的输入，完全相同的三份，并且这个输入还有一个副本用于残差网络的直接映射。其中每个残差网络，包括整个encoder输出的向量维度都是相同的，且是作为超参数进行预设的。

3.3 Transformer的decoder部分

Transfomer的decoder部分相较encoder复杂一些，是由三个残差网络组成。与encoder相同decoder也会将其输出作为输入重复若干次。Transfomer的decoder部分面向的是有监督学习，所以存在相应的目标序列output。对output做与input相同的处理之后输入encoder部分。同样是复制四份输入，三份输入进多头attention输出自注意力（decoder self-attention），一份直接映射计算残差网络输出。这个输出会成为decoder中第二个多头attention的Q，而这第二个多头attention的K和V则都是来自于encoder的输出，复制两份。这个多头attention被称为decoder-encoder attention。最终再经过一个前馈神经网络，输出decoder。

3.4 Transformer中的mask机制

Mask机制在RNN、attention、Transformer以及BERT中都有使用，这里只简介一下Transformer中的mask。因为输入序列并不是定长的，所以往往需要补齐（padding）,而在计算attention时Transformer不应该受到这些padding项的干扰，所以这时就需要mask去屏蔽这些padding项。具体做法就是在进入softmax之前，对于需要屏蔽的部分设定其值为负无穷，这样softmax的对应项的结果就会为0，即不分配attention在这些项上。这是Transformer中encoder部分的mask。

Decoder部分的mask则是为了防止模型提前“看到答案”，也就是所谓的标签泄漏。在模型处理t时刻的输入序列时不应该看到t时候之后的序列信息，为了达到这个目的，需要mask把t时候之后项屏蔽。

4. 可变参数设置

我们小组对Transformer中的若干参数做了实验，旨在挖掘Transformer内部各个结构对整体模型的影响，以及确认它们之间是否存在某种“协作关系”。目前已经有

很多研究关注过Transformer内部的超参数，《Attention Is All You Need》(Vaswani et al. 2017)这篇论文内就有一定规模的调参实验，但即使如此，我们的实验依然是有意义的。其一，我们的训练数据规模较小，训练轮次较少，在这种情况下什么样的参数设置可以使Transformer尽快收敛，得到较好的结果，这是一个值得实验的问题。其二，我们大胆地修改了Transformer里的一些结构，例如去掉前馈网络，修改部分网络中的激活函数，不仅限于对模型中各种向量维度、神经元个数的调整。这使得我们可以获得一个新的视角来审视Transformer的内部结构。

如下是我们详细的可变参数设置。

4.1 HID_DIM

HID_DIM是《Attention Is All You Need》(Vaswani et al. 2017)中的d_{model}。它出现的位置包括且不限于：input和output序列中每个token经过word embedding之后的向量的维度，positional embedding之后的向量的维度，encoder和decoder输出的向量的维度，Q、K和V中每个向量的维度，前馈神经网络中输入输出向量的维度。另外，HID_DIM也对应式1中的$\sqrt{d_k}$，调整HID_DIM的同时会改变$\sqrt{d_k}$，因为它们虽然不是同一个值，但是$d_k = HID_DIM/N_HEADS$。

4.2 N_HEADS

N_HEADS就是计算d_k时的分母。它是指多头attention机制中Q、K和V切分的份数，每一份都要输入一个独立的attention结构，最后再将输出向量拼接起来。事实上在多头attention的输入中，每个输入都会先经过一个全连接层，如式2中的W_i，变换成为Q、K和V，所以即使是切分得到的QKV也可以部分地获得整句话的信息，产生的attention矩阵也是可以全面的关注整个句子的。

《Attention Is All You Need》(Vaswani et al. 2017)中关于N_HEADS这个参数有两个论述。一是N_HEADS的大小不会影响训练时间，因为头的数量多了，每个头内向量的维度也会成比例减小，所以和单头（single-head）在计算复杂度上（computational cost）是相似的。二是N_HEADS的数量越多，模型就可以学习到越多不同层面（different representation subspaces）的语义。我们的实验结果对这两个论述分别给出了答案。

4.3 ENC_LAYERS/DEC_LAYERS

ENC_LAYERS/DEC_LAYERS表示的是encoder和decoder重复的次数。这个值过大过小都对模型训练不利，但是这个值究竟设计多少为好，这可能是一个需要调试的问题。我们为了减少总模型数量，所以限定encoder和decoder的重复次数相同。

4.4 PF

PF的意思是position-wise feed-forward network。这个结构在encoder和decoder部分中都出现了，如果把encoder和decoder部分的$\times N$展开，可以看到PF层总是出现在两个attention层之间。也就是说， Transformer通过对输入的文本不断进行这样的注意力机制层和普通的非线性层交叠来得到最终的文本表达。

PF内部的结构是非常简单的，主要由两个线性变换以及一个激活函数ReLU组成，如式3所示：

$$FFN(x) = max(0, xW_1 + b_1)W_2 + b_2 \tag{3}$$

这样一个用于增强模型表达能力的非线性层会对模型产生什么样的具体影响，我们也做了针对性的实验。我们构建了包含position-wise feed-forward network和不包含position-wise feed-forward network的模型，通过对比它们的效果判断其影响。

4.5 PF_ACT

PF层选用的激活函数是ReLU，但是《Attention Is All You Need》(Vaswani et al. 2017)这篇论文对此并没有太多解释。相反，BERT模型则是在此处选择了GELU。GELU在激活中引入了随机正则的思想，是一种对神经元输入的概率描述，直观上更符合自然的认识，同时实验效果要比Relu与ELU都要好。关于GELU的更多描述，可以参考论文《Gaussian Error Linear Units》(Hendrycks and Gimpel 2016)。为了验证两种激活函数在Transformer中的表现，我们设置了激活函数选项，可以调整所有PF层的激活函数。

4.6 ENC_PF_DIM/DEC_PF_DIM

式3中的W_1和W_2是两个线性变换，W_1把向量由HID_DIM维映射到PF_DIM维，W_2把向量由PF_DIM维重新映射回HID_DIM维。这其中的中转维度PF_DIM根据《Attention Is All You Need》(Vaswani et al. 2017)中的建议，应该远大于（a lot large than）HID_DIM。我们根据这个建议分别做了两倍和四倍的实验。

4.7 TE_TYPE

在《Attention Is All You Need》(Vaswani et al. 2017)介绍embedding层的段落的结尾，出现了一句比较奇怪的话：In the embedding layers, we multiply those weights by $\sqrt{d_{model}}$。全文其他地方并没有对这句话的解释。为了确认这个参数的

作用，我们设置了专门的选项用于选择是否在embedding层乘这个参数。d_{model}就是HID_DIM。

4.8 ENC_DROPOUT/DEC_DROPOUT

最后一个可变参数是dropout率，用于防止过拟合的常见的参数。事实上，因为我们的实验规模是比较小的，存在一定风险出现过拟合或者欠拟合，所以对所有模型都分别进行了dropout率的变化，有三个可选值：0，0.1，0.2。

4.9 可变参数总表

表1列出了所有可变参数，以及其可选值。

表 1: 可变参数总表

HID_DIM	LAYERS	HEADS	PF	PF_ACT	PF_DIM	TE_ACT	DROPOUT
256/512	3/6	4/8	True/False	ReLU/GELU/none	1024/2048	0/1	0/0.1/0.2

5. 其他模型细节

虽然我们设置了很多可变参数来对模型进行全面的调试，但是模型中更多的是不变的部分。本节以无序列表的形式对其进行介绍。

- Positional embedding：位置编码分为了绝对位置编码（Learned Positional Embedding）和相对位置编码（Relative Positional Embedding）。尽管相对位置编码的方法很多，看起来也非常的有道理，但是我们还是在实验中使用了目前用的更多的绝对位置编码。
- 预训练模型：我们没有在word embedding部分采用预训练模型，其中的参数与模型中其他的参数一起进行初始化。
- Attention机制：我们没有对attention机制内部做任何调整，但是我们小组组员李润宽的综述是关于Transformer里的attention的各种变体的介绍。
- 优化器（optimizer）：我们使用了学习率为0.0005的Adam优化器，没有warm up和cool down阶段。
- 参数初始化：我们用Xavier uniform进行了参数初始化，每个参数以均匀分布随机赋值。
- 硬件与训练规模：一块Tesla-K80，batch size为256，epoch为30。

6. 数据集介绍

我们的数据集是一个小规模数据集，其中包含了英文的原语句，中文的目标语句，长短不一，并且都是单句。在数据清洗的过程中我们删掉了部分过长的语句，对英文缩写进行了展开。但是有些无法准确展开的缩写，例如“Tom’s”无法区分是Tom has还是Tom is还是就是Tom’s，不得已进行了删除。最大程度保留了训练样本，剔除了不合要求的样本。

中文分词使用的是jieba，不过实际上是因为spacy对中文的分词用的标准方法就是jieba。预处理部分使用的是spacy+torchtext，自动生成了词汇表（并手动保存了词汇表）。因为要处理的任务是机器翻译，所以没有去除停用词。

以下是数据集的一些基本信息：

- Tokens数量：
    1. 英文token 3391
    2. 中文token 4166。
- 样本数量：
    1. 训练集17587；
    2. 验证集2197；
    3. 测试集2189。

7. 实验结果与分析

根据第4节的参数设置，我们总共训练了384个参数不同的Transformer模型，分别记录了它们的训练时间、参数数量、loss、PPL和BLEU值作为评价指标。其中BLEU的参数设定与论文《BLEU: a Method for Automatic Evaluation of Machine Translation》(Papineni et al. 2002)中的baseline设定一致，也是评估机器翻译模型的常用指标。

以下是逐个参数的结果分析。

7.1 模型规模相关参数

在本实验中，会影响模型规模的参数有三个：HID_DIM、ENC_LAYERS/DEC_LAYERS、ENC_PF_DIM/DEC_PF_DIM。N_HEADS并不影响模型规模，具体原因可见4.2节中对N_HEADS参数的说明。

因为我们的数据集很小（训练集17587个样本），训练轮次也非常有限（30轮，共2161steps），所以模型规模不能太大，否则会无法收敛。体现在实验中，HID_DIM=512且ENC_LAYERS/DEC_LAYERS=6的情况下训练的模型均未收敛，

本节故不讨论这种参数情况。除了模型规模相关参数，其他参数在本节实验被同一设定，分别为：HEADS=8，PF_ACT=relu，TE_TYPE=0，DROPOUT=0.1。

表 2: HEADS=8，PF_ACT=relu，TE_TYPE=0，DROPOUT=0.1

HID_DIM	LAYERS	PF	PF_DIM	训练时间	参数数量 10^6	loss	BLEU
256	3	True	1024	7m55s	8.51	2.48	17.42
			2048	9m26s	11.74	2.62	16.72
		False		5m32s		2.51	**18.28**
	6	True	1024	14m48s	14.12	7.46	-
			2048	16m52s	20.42	7.23	-
		False		10m10s		2.96	1.84
512	3	True	1024	13m51s	21.89	2.98	6.11
			2048	16m52s	28.18	2.86	8.73
		Fasle		10m11s		2.8	15.98

表2列出了模型规模相关参数的实验结果，其中PF决定了模型是否含有前馈神经网络层，BLEU项的“-”表示该模型未收敛。可以看到模型总体随着参数的减少获得了更优秀的性能，无论是训练时间还是BLEU的测试值。值得注意的是，在本实验的小样本集上，不含前馈神经网络的Transformer模型（即PF=False）表现更好，训练时间和翻译准确率都得到了较大提升。这一提升幅度很大，显然不是由于模型初始化的随机性所导致的。同时，去除前馈神经网络层，还使得模型对训练集的过拟合得到了缓解。这点从HID_DIM=512且LAYERS=3的这组模型就可以看出，它们的loss相近，但是PF=FALSE的BLEU值很高，说明在测试集上表现很好，而其他含有前馈神经网络的模型，在相近loss情况下BLEU相对低，说明有一定的过拟合情况出现。

7.2 多头注意力机制的head数量

表3是一组对比实验。在表3中，未标注的超参数取值为HEADS=4，PF_ACT=relu，TE_TYPE=0，DROPOUT=0.1，只有HEADS与表2减少了一倍，与其对比即可得知多头注意力机制的头数对模型的影响。其中括号内为变化量。

首先是训练时间随着头数减少而统一减少，虽然幅度各有不同，但是趋势完全一致。可以确定，其实多头注意力机制的数量对训练时间还是有一定影响的，但是影响幅度总体上是不大的。再来看loss与BLEU的情况。这两个指标的变化幅度同样

表 3: HEADS=4，PF_ACT=relu，TE_TYPE=0，DROPOUT=0.1

HID_DIM	LAYERS	PF	PF_DIM	训练时间	参数数量 10^6	loss	BLEU
256	3	True	1024	7m33s(-22s)	8.51	2.51(+0.03)	15.95(-1.47)
			2048	9m1s(-25s)	11.74	2.65(+0.03)	17.38(+0.66)
		False		5m11s(-21s)		2.47(-0.04)	**18.11(-0.17)**
	6	True	1024	14m0s(-48s)	14.12	6.55(-0.91)	-
			2048	16m48s(-4s)	20.42	6.96(-0.27)	-
		False		9m19s(-51s)		2.88(-0.08)	3.1(+1.26)
512	3	True	1024	13m43s(-8s)	21.89	2.83(-0.15)	7.68(+1.57)
			2048	16m42s(-10s)	28.18	2.8(-0.06)	6.25(-2.48)
		Fasle		10m9s(-2s)		2.76(-0.04)	13.09(-2.89)

不大，尤其是在loss这一项上，只要模型已经收敛，头数几乎不会影响最终loss。这种结果可能是因为训练集句子长度普遍较短，不需要很多多头注意力机制来构造不同语义空间上的注意力矩阵。

7.3 前馈神经网络中的激活函数

我们的实验一共测试了三种前馈神经网络的激活函数：ReLU，GELU和无激活函数（none）。实验结果如表4。

在表4中，我们限定了模型必须包含前馈神经网络（否则自然没有激活函数），且PF_DIM=1024，TE_TYPE=0，DROPOUT=0.1。因为表4中只显示了收敛的模型，所以LAYERS=6的模型都未列出，具体情况可以参考表2。在评价指标部分，激活函数显然不会影响模型参数数量，所以这一列也未列出。实验结果显示，GELU和ReLU在反向传播过程中都确实没有增加模型运算量，所以其训练时间与无激活函数表现相同。这一点是由激活函数本身的导数的性质决定的。第二，在各种情况下GELU都显示出了较ReLU更好的性能，证明了其确实适合作为前馈神经网络的激活函数。这种优越性来自于GELU的设计，是一种全面的优势，而不是限定在某种模型维度上。但是不得不说，GELU缺少了一些sigmoid式的数学上的优雅，恐怕并不是激活函数发展的最终方向。

7.4 Embedding层中的HID_DIM

在4.7节中提到，Transformer的原模型在embedding层乘了一个参数HID_DIM，但是具体的原因没有说明，于是我们也对这一点进行了实验。本节的实验结果在各个

表 4: PF_DIM=1024，TE_TYPE=0，DROPOUT=0.1

HID_DIM	LAYERS	HEADS	PF_ACT	训练时间	loss	BLEU
256	3	8	ReLU	7m55s	2.48	17.42
		4		7m33s	2.51	15.95
		8	GELU	7m59s	2.58	17.07
		4		7m31s	2.5	19.29
		8	none	7m54s	2.74	14.41
		4		7m22s	2.7	14.68
512	3	8	ReLU	13m51s	2.98	6.11
		4		13m43s	2.83	7.68
		8	GELU	13m53s	2.84	9.7
		4		13m48s	2.74	12.31
		8	none	13m45s	3.36	3.69
		4		13m33s	3.26	1.93

参数下都非常一致，故本节只展示一小部分收敛模型的实验结果，可以说明清楚即可。

表5中未标注的超参数取值为：PF=False，DROPOUT=0.1。可以看到，这个只作用于embedding层的超参数很大程度上影响了模型的效果。将embedding之后的向量与HID_DIM相乘可以大幅度提升模型效果，这一提升的幅度很大，几乎和模型复杂度的提升带来的效果相近。

7.5 最终的模型参数

最终模型参数如表6所示。两个模型虽然在BLEU上有差距，但是训练时间相差幅度也很大。包括以上所有参数组合的模型，可以在我们的web服务器上尝试它们的组合，并且实际执行翻译测试直观效果。

表 5: PF=False，DROPOUT=0.1

HID_DIM	LAYERS	HEADS	TE_TYPE	训练时间	loss	BLEU
256	3	8	0	5m32s	2.51	18.28
		4		5m11s	2.47	18.11
		8	1	5m37s	2.45	16.23
		4		5m9s	2.37	15.84
512	3	8	0	10m15s	2.8	15.57
		4		10m9s	2.76	13.09
		8	1	10m18s	2.82	13.43
		4		10m9s	2.77	13.33

表 6: 最终模型参数与效果

HID_DIM	LAYERS	HEADS	PF	PF_ACT	PF_DIM	TE_ACT	DROPOUT	训练时间	loss	BLEU
256	3	4	False			0	0.1	5m6s	2.49	18.7
256	3	4	True	GELU	1024	0	0.1	7m31s	2.5	19.29

8. 部署与使用

8.1 环境要求

本项目的环境要求为：

- 模型训练环境：
  1. 操作系统： Ubuntu 18.04.3 LTS (GNU/Linux 3.10.0-957.el7.x86_64 x86_64)
  2. 模型训练环境:两块Tesla K80，CUDA Version为11.0，256 for each batch, 30 epochs.
  3. 代码环境：Pytorch1.6.0，python3.6及以上
- Web服务运行环境：
  1. 操作系统：Ubuntu 18.04.4 LTS (GNU/Linux 4.15.0-118-generic x86_64)
  2. 部署平台：腾讯云服务器,公网ip：62.234.19.125
  3. 代码环境：Flask轻量级web框架，nginx，pytorch1.7.0，python 3.7.9

8.2 项目访问方式

浏览器输入ip地址62.234.19.125:5000即可访问我们的项目（注意一定要输入端口5000），推荐使用Chrome、Firefox、Edge浏览器。

8.3 使用介绍

- 功能简介
  1. 实验简介：本项目主页介绍了模型训练使用的数据集情况，模型训练中的不变的假设和所测试的Transformer超参数的解释。
  2. 翻译功能：本项目可以实现简单的英文到中文的翻译。
  3. Transformer超参数评估：通过下拉菜单对384个参数组合进行选择，并返回各项评估指标和attention图来评估参数组合的对模型效果的影响。
- 使用方法
  1. 在文本框输入英文，选择参数组合（注意，当PF选择为False时，PF_ACT只能选择None，若选择其他值也将视作None），提交后可以通过翻译结果、评估指标、attention图判断参数的效果。
- 效果演示

1. 如图1所示，在输入文本框里输入句子，例如:I like sleeping.然后，选择超参数，例如将PF_ACT调整为gelu，将TE_TYPE调整为1，其他值保持不变，提交。

图 1: 输入句子，选择参数

2. 返回的对应超参数下的模型效果评价指标如图2所示：

选择的参数如下:

HID_DIM	ENC_LAYERS/DEC_LAYERS	N_HEADS	PF
256	3	8	gelu
PF_ACT	**ENC_PF_DIM/DEC_PF_DIM**	**TE_TYPE**	**ENC_DROPOUT/DEC_DROPOUT**
gelu	1024	1	0.1

模型效果的评价指标如下:

Running Time	Parameters Count	Val_loss	Val_PPL	BLEU
7m58s	8586054	2.45	11.65	14.21

图 2: 对应的评价指标结果

3. 该组超参数生成的模型对这句话翻译效果的Attention图如图3所示：

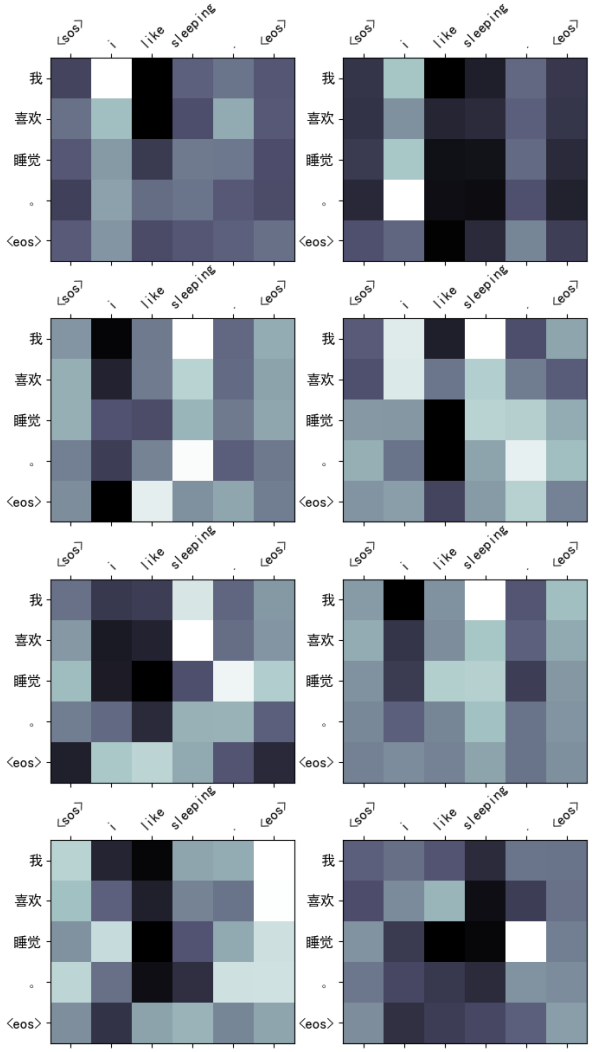

图 3: Attention图

4. 同理，可以修改参数组合重复上述操作来观察对应的模型效果。

9. 总结

本实验重点关注了Transformer中包含结构信息的超参数，并对它们做了详细的调试。有些证实了确实是与原文中建议一致，而有些在我们的小样本实验环境下的结果存在出入。我们又在此基础上按照最优参数训练了翻译模型，取得了一定的效果。

References

Gehring, Jonas, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning. *CoRR*, abs/1705.03122.

Hendrycks, Dan and Kevin Gimpel. 2016. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. *CoRR*, abs/1606.08415.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, page 311–318, Association for Computational Linguistics, USA.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, undefinedukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Curran Associates Inc., Red Hook, NY, USA.