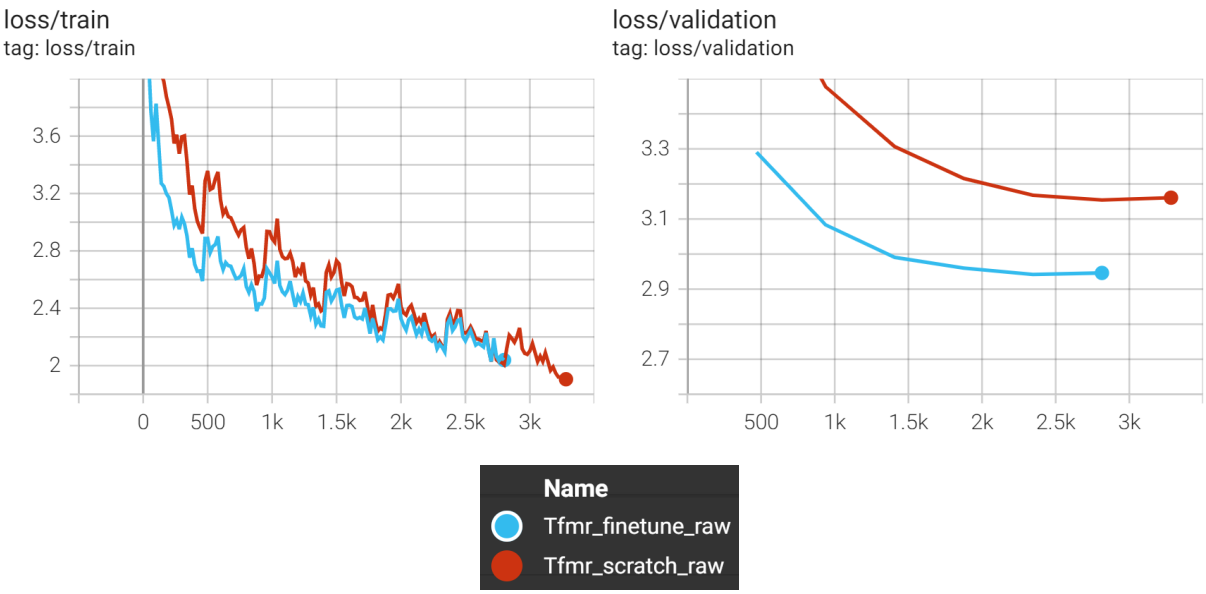
ANN HW3

计02 刘明道 2020011156

1. 随机初始化 v. 微调预训练

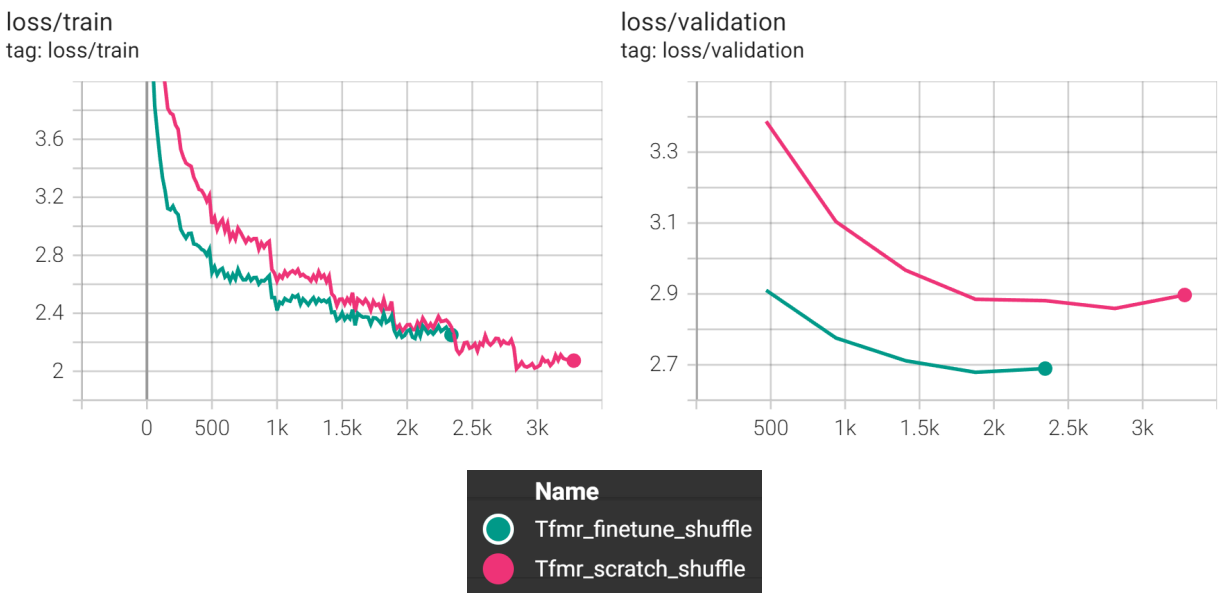
未经 shuffle 的训练



Model	loss/train	loss/validation	perplexity/validatoin
Tfmr-scratch-not-shuffle	**1.90**	3.15	23.75
Tfmr-finetune-not-shuffle	2.03	**2.94**	**19.03**

训练 loss 显示，每个 epoch 开始处的 loss 有显著上升。这提示练数据分布不甚均匀，可能存在某些顺序特征。

经过 shuffle 的训练



Model	loss/train	loss/validation	perplexity/validatoin
Tfmr-scratch-shuffle	**2.07**	2.85	17.44

Model	loss/train	loss/validation	perplexity/validatoin
Tfmr-finetune-shuffle	2.25	**2.67**	**14.57**

为此我们对训练数据在每个 Epoch 进行不同 shuffle。这样做在验证集上得到了显著更低的 Loss 和 Perplexity。

因此在后续实验中，均采用按 epoch shuffle 后的数据进行训练。

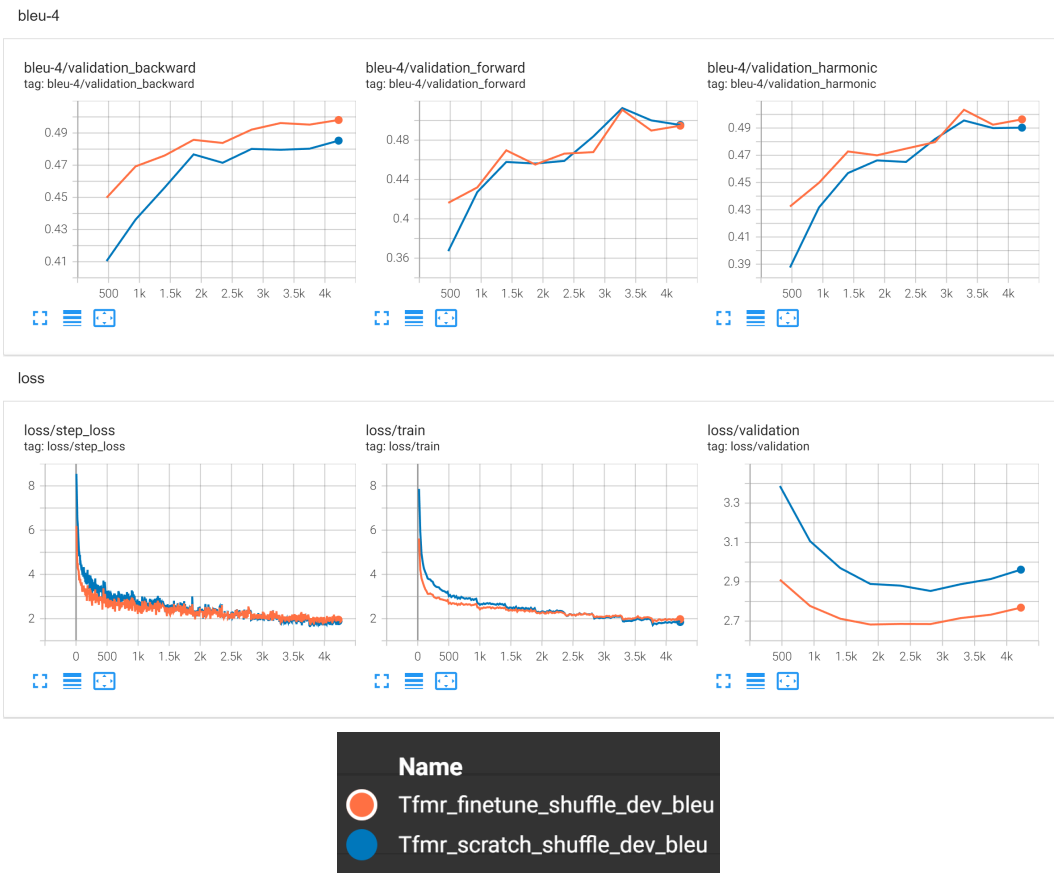
模型表现

选取验证集上表现最好的模型，使用默认参数（即 `decode_strategy=random, temperature=1.0`）进行推理。在测试集上得到的结果为

Model	Perplexity	Forward-BLEU-4	Backword-BLEU-4	Harmonic-BLEU-4
Tfmr-scratch	15.31	**0.567**	**0.516**	**0.540**
Tfmr-fineture	**13.09**	0.529	0.513	0.521

比较与分析

- 相比从随机初始化的参数开始训练，**微调预训练模型收敛速度更快，在验证集和测试集上的 Perplexity 显著更低。**
  - 预训练给模型的参数分布带来了一定的先验的语言知识，有利于模型快速学习当前语料库中的分布。
- 相比从随机初始化的参数开始训练的，微调预训练模型的 BLEU Score 更低
  - 这与直觉相反，我推测可能的原因包括
    - ① **Loss 和 BLEU Score 不一定同时到达最优。**为了验证这一点，以验证集的 BLEU Score 作为停止训练的标准，选取**在验证集上 BLEU Score 最优的模型到进行测试**，结果如下



在模型有 **2** 次 BLEU Score 连续下降的情况下停止训练。这里发现，Validation loss 最低的点并非 BLEU 最高的点。

而从图中可以看出，微调预训练模型在 验证集 的 backward-BLEU 比 随机初始化模型明显更高，而forward-BLEU 则较为接近。

在测试集上的验证结果为

Model	PPL	Forward-BLEU-4	Backword-BLEU-4	Harmonic-BLEU-4
Tfmr-scratch	15.63	0.581	0.510	0.543
Tfmr-fineture	**13.43**	**0.586**	**0.522**	**0.552**

这时 finetune 结果略微优于 scratch。

- ② **train，dev，test 之间的分布差异较大。**为此我们在三个集合间两两计算 BLEU Score，结果如下

	train-dev	train-test	dev-test
Harmonic-BLEU-4	0.550	0.611	0.538

可见，此时即使模型充分地记忆了训练集的统计特征，也不足以保证其在测试集与验证集的 BLEU Score 上也有足够好的表现。而在验证集上 BLEU 最高的模型，到了测试集上也不一定如此。

- ③ **BLEU Score 评价指标可能不能充分反映模型的生成质量。**BLEU Score 更侧重于 n-gram 的重合程度，但无法反映句子内容否合理与否等评价维度。

2. 比较不同的解码策略

鉴于先前的讨论。在这一节我们共测试 4 个不同的模型，分别是

Model	选取方式
scratch-ppl	Tfmr-scratch，取 Validation Perplexity 最低
finetune-ppl	Tfmr-finetune，取 Validation Perplexity 最低
scratch-bleu	Tfmr-scratch，取 Validation Harmonic BLEU 最高
finetune-bleu	Tfmr-finetune，取 Validation Harmonic BLEU 最高

结果如下，其中每组实验的中的三个数字分别为

```
(Forward-BLEU, Backward-BLEU, Harmonic-BLEU)
```

scratch-ppl

	Perplexity/Test = 15.31					
	Temp=0.7			Temp=1.0		
Random	0.800	0.487	**0.605**	0.567	**0.516**	0.540
Top-p	**0.867**	0.421	0.567	0.686	0.514	0.588
Top-k	0.819	0.473	0.599	0.682	0.511	0.584

finetune-ppl

	Perplexity/Test = 13.09					
	Temp=0.7			Temp=1.0		
Random	0.788	0.491	**0.605**	0.529	0.513	0.521
Top-p	**0.858**	0.425	0.569	0.647	**0.519**	0.576
Top-k	0.810	0.478	0.601	0.642	0.514	0.571

scratch-bleu

	Perplexity/Test = 15.63					
	Temp=0.7			Temp=1.0		
Random	0.822	0.474	**0.601**	0.581	**0.510**	0.543
Top-p	**0.889**	0.398	0.550	0.698	0.507	0.588
Top-k	0.837	0.460	0.593	0.690	0.508	0.585

finetune-bleu

	Perplexity/Test = 13.43					
	Temp=0.7			Temp=1.0		
Random	0.809	0.485	**0.607**	0.586	**0.522**	0.552
Top-p	**0.881**	0.410	0.560	0.697	0.518	0.594
Top-k	0.829	0.471	0.601	0.674	0.517	0.585

比较与分析

- **Decoding Strategy.** 对于 Random 解码方式来说，所有的 token 都会以相应的概率被选中；而 top-p 与 top-k 则是在一小部分概率较高的 token（分别是概率和为 p 的最小集合 和 概率最高的 k 个 token 组成的集合）中，按照归一化后的概率进行选择，而不会选取剩余概率较低的那部分 token。
  - 实验中 Random 策略的 Forward-BLEU 较低（最低），而 Backward-BLEU 较高（8组实验中7组为最高）。原因在于，**Random 策略由于可以选择全部的 token，因此内容更加丰富多样，但是流畅性也相对较差。**
  - 对于实验中参数 p 与 k 的取值而言，top-p 总体来说比 top-k 有更高的 Forward-BLEU 和更低的 Backward-BLEU。这说明 $p = 0.9$ 的 top-p 对 token 的选取更为集中（也就是选取 token 数量的期望值少于 $k = 40$ 个），因此生成结果更加流畅，但生成的内容也更为单一。
  - 选取 token 数量的多少的调节，需要权衡多样性与流畅性。
- **Temperature.** 温度参数主要控制概率分布的尖锐程度。**较大的温度参数对应于较为平缓的分布，而较小的温度参则会让放大 logits 之间的差异，使分布更加尖锐。**
  - 从结果来看，tempearture 取 0.7 时的 Forward-BLEU 显著更高。这是生成时更加倾向于选择少量高概率的 token，因此生成内容更加流畅。
  - temperature 取 1.0 时的 Backward-BLEU 显著更高。这是因为较高的温度参数使分布整体趋向平缓，不同 token 之间的概率差异减小，logit 没那么高的 token 被选中的概率被相对放大，因此可以获得更为丰富的采样。
  - temperature 的选择也是对采样的丰富性和生成内容的流畅性之间的权衡。
- **Decoding Strategy & Temperature.** 以 Harmonic-BLEU 作为综合评价指标，**不同的策略可能会适和不同的温度。**
  - 对于 Random 采样策略来说，由于其采样范围较广，本身多样性较强，这时就需要较小的 temperature 来提供较为尖锐的分布，提升流畅度，得到更优的综合结果。
  - 对于 top-p 采样策略来说，就本实验的参数取值 $p = 0.9$，其采样已经相当集中，因此适合较大的 temperature，防止多样性太低。
  - 对于 top-k 采样策略来说，就本实验的参数取值 $k = 40$，其选取的范围还是相对较广的，因此较小的 temperature 得到了更优的结果。

3. 比较不同编码策略的生成内容

这一节我们随机选取的样本来自于 **scratch-bleu** 和 **finetune-bleu** 在每个策略下相同位置（随机状态）的 10 个句子。结果如下

Tfmr-scratch (Test/Perplexity=15.63)

	Temp=0.7	Temp=1.0
Random	A fire hydrant with a couple of giraffes in the background. A fire hydrant is painted red, and street signs. A man on a boat looking up at the water. A black and white photo of a jumbo jet parked on a runway. A man sitting on a green bench with a dog. A set of wooden benches sitting on top of a hill. A man riding a motorcycle with a dog on top of it. A man sitting on a toilet in a small bathroom. The green double decker bus is parked on the side of the road. A white toilet sitting next to a sink in a bathroom. A man riding a motorcycle while riding a bike down the street.	Black and white photograph of man riding at a motorcycle. A fire hydrant is painted red, and street sign. A fireman a woman driving across a mobile soaked street. A chair for guests to pick up water from their bowl. A red car sitting on green wall lined next to a fire hydrant. A set of wooden benches sitting on top of a street night. A man laying in the grass while holding an umbrella. A yellow fire hydrant sitting in front of a forest. The green double decker bus is pulling into the bus. City street light with several cars and cars behind it. The plane is all the entertainment center is shown on the runway.
Top-p	A fire hydrant with a sign on it in the background. A fire hydrant is painted red, and street signs. A man on a motorcycle driving down a street next to a parked cars. A black and white photo of a woman sitting on a bench. A man sitting on a bench with a helmet and his face. A man sitting on a bench with a cat in front of him. A man riding a motorcycle with a dog on top of it. A man sitting on a wooden bench holding a dog. A bathroom with a toilet, sink, and bathtub. A white toilet sitting next to a sink in a bathroom. A man riding a motorcycle while riding a bike down the street.	A fire hydrant with a couple of policemen in the background. A fire hydrant is painted red, and street sign. A fire hydrant sitting on the grass near some grass. A black and white photo of a jumbo jet parked on a runway. A red car sitting on top of a lush green field. A set of wooden benches sitting on top of a street. A man laying in the grass while holding an umbrella. A yellow fire hydrant sitting in front of a forest. The green double decker bus is pulling into the bus. The sheep are grazing together while another person' s sheep. The plane is all the entertainment center is shown on the runway.
Top-k	A fire hydrant with a couple of giraffes in the background. A fire hydrant is painted red, and street signs. A man on a boat looking up at the water. A black and white photo of a jumbo jet parked on a runway. A man sitting on a green bench with a dog. A man sitting on a bench with a cat in front of him. A man riding a motorcycle with a dog on top of it. A man sitting on a toilet in a small bathroom. The bathroom has a toilet, sink, and shower accessories. A white toilet sitting next to a sink in a bathroom. A man riding a motorcycle while riding a bike down the street.	Black and white photograph of man riding at a motorcycle. A fire hydrant is painted red, and street sign. A fireman a woman driving across a street in the city. A black and white photo of a jumbo jet parked on a dock next to a gate. A red car sitting on green grass lined street beside trees. A man sitting on top of a wooden bench near a fence. A man laying in the grass while holding an umbrella. A yellow fire hydrant sitting in front of a forest. The bathroom has a toilet, sink, and bathtub. City street light with several cars and cars behind it. The man is walking the dog up a bike wearing a hat.

Tfmr-finetune (Test/Perplexity=13.43)

	Temp=0.7	Temp=1.0
Random	A fire hydrant with a hose leaking water into the water. A fire hydrant is painted red, white, and blue. A man on a boat traveling a river near a river. A black and white photo of a plane sitting on a runway. A man sitting on a green motorcycle with a car. A set of lights that are on in front of a television. A man riding a motorcycle with a dog on top. A man sitting on a toilet in a bathroom next to a bathtub. The green double decker bus is pulling into the side of a road. A white toilet sitting next to a sink in a bathroom. A man walking down the street lined with a police car.	Black fire hydrant with clear faces drawn on the window. A fire hydrant is painted red, white cabinets and a bar. A vintage roller coaster ride with a weird looking bike parked in front a window. A chair for a platter with pens, computers and a laptop sitting next to each other. A red face drawn on green wall at a car. A set of lights that are on a moped. A man laying in bed next to a fire hydrant. A yellow and gray fighter jet taking off in the sky. The green double decker bus is pulling into the sky. City street light with several cars and cars behind them. The back of a giraffe walking a big triangular turn.
Top-p	A fire hydrant with a hose connected to it. A fire hydrant is painted red, white, and blue. A man on a motorcycle driving a road past a grassy hillside. A black and white photo of a plane sitting on a runway. A man sitting on a bench watching a sail boat through the water. A man sitting on a bench on a beach beach. A man riding a motorcycle with a dog on top. A man sitting on a wooden bench near a building. A bathroom with a toilet, sink, and shower. A white toilet sitting next to a sink in a bathroom. A man riding a motorcycle while riding a bike on the street.	A fire hydrant with a license plate missing its seat. A fire hydrant is painted red, white, and blue is standing on a street. A vintage motorcycle parked on the side of the road. A black and white photo of a jumbo jet parked on a runway. A red car turns a green street at a traffic light. A set of lights that are on a moped. A man laying in bed next to a fire hydrant. A yellow and black fighter jet in an airport. The green double decker bus is pulling into the sky. The sheep are all grazing, besides the man' s bicycle. The back of a giraffe walking a big tree.
Top-k	A fire hydrant with a hose leaking water into the water. A fire hydrant is painted red, white, and blue. A man on a boat traveling a river near a river. A black and white photo of a plane sitting on a runway. A man sitting on a green motorcycle with a car. A man sitting on a bench on a beach beach. A man riding a motorcycle with a dog on top. A man sitting on a toilet in a bathroom next to a bathtub. The bathroom has a toilet, sink, and shower accessories. A white toilet sitting next to a sink in a bathroom. A man walking down the street next to a police car.	Black fire hydrant with a lot of dirt on the side of it. A fire hydrant is painted red, white cabinets and a sink. A man on a boat floating in water beyond the sail boats parked a harbor. A black and white photo of a plane sitting on a runway. A red car turns a green street at a traffic light. A man sitting on top of a wooden bench in a park. A man laying in bed next to a fire hydrant. A yellow and gray fighter jet taking off in the sky. The bathroom has a sink in the shower, bathtub, and sink in it. City street light with several cars and cars behind them. The back of a giraffe walking a big tree.

- 各组生成结果都存在语法错误（这里使用 Grammarly 自动检测，包括：冠词丢失，单复数错误，缺少动词，标点使用不当等，在图中用红线标记）
- 相比来说，**低 temperature / top-p / top-k 的解码方式生成的语句更加连贯流畅，语法错误更少。**这一点与其较高的 Forward BLEU 表现一致。同样地，这些解码结果的丰富性就比较差，这一直观感受与其较低的 Backward-BLEU 也是一致的
    - 例如 finetune 组 temp=0.7 的 top-p 中，有 6 句都以 *A man* 开头，而后面的选词也较为接近，可见其 token 选择是相当局限的。
    - 再如，所有的 temp=0.7 组的句首均为冠词开头。这大概是因为没有任何信息时，模型认为句首出现概率最大的词就是冠词。这当然可以防止缺少冠词的语法错误，但也大大降低了生成内容的多样性。
- 从阅读感觉来说
    - **Perplexity 与 生成结果质量 的关系并不像 Forward/Backward-BLEU 那样明显。**finetune 组显著有更低的 Perplexity，但其对应 解码策略 的 生成流畅程度、多样性、语法错误 等并没有很明显的差异。
    - **BLEU Score 与人类的阅读体验关系则较为明显。**Forward-BLEU 与流畅性，Backward-BLEU 与多样性，Harmonic-BLEU 与整体感觉，都有较为明显的正相关。
- **生成文字质量较高的是 finetune/scratch 的 Random, temp=0.7 输出结果。**其结果较为流畅，同时也因为具有较多内容而不会过于无聊。
- 在某些情形下，**Finetune 比 Scratch 生成的结果更具有语义的相关性 。**例如给定前缀 *A fire hydrant* 时，finetune 组给出了 *hose*, leaking, water 等与水有关的结果，而 scratch 则补全的 *sign*, *giraffe* 的语义相关性则稍微差一些。

4. 最终结果

最终选择的模型为

```
- 取 GPT-2 预训练模型的第 1, 6, 12 层进行微调
- 训练时每个 epoch 的输入进行 shuffle
- 选取在验证集上 Perplexity 最低的模型
- 解码策略选择 Random, temperature=0.7
- 其余超参数保持默认
```

评测指标为

Perplexity/dev	Perplexity/test	Forward-BLEU	Backward-BLEU	Harmonic-BLEU
13.95	12.53	0.797	0.488	0.605

生成结果见 `outputs.txt`

5. 分析 Transformer Decoder

1. 为什么 Multi-head Attention 的效果比 Single-head 效果好?

   Multi-head Attention 相当于在隐状态投影后的多个不同子空间中进行多次 Self-Attention。**多个 attention head 可以 attend 到不同的位置，这有利于在不同子空间中捕捉多种特征。**

2. BPE Tokenizer 相比空格分词有什么优势?

   BPE Tokenizer 可以将合成词分成多个词。这一方面可以使模型能更好地处理出现频率较低的复杂的合成词（因为合成词的各个部分可能单独出现的频率会高一些），也能使模型更好地处理未曾见过的新合成词（因为新词的某个部分可能是模型见过的）。

   例如

```
In [43]: [tokenizer.decode([e]) for e in tokenizer.encode('onehot vector; multihead attention')]
Out[43]: ['one', 'hot', ' vector', ';', ' multi', 'head', ' attention']
```

3. 从时空复杂度，性能，位置编码等角度比较 Transformer 和 RNN。

   设序列长度为 T，隐藏层维数为 d 。

   时间复杂度

   - 单个 Transformer Block 编码整个序列所需要的时间复杂度为 $\mathcal{O}(dT^2 + d^2T)$；而 RNN 编码整个序列所需的时间复杂度为 $\mathcal{O}(d^2T)$ 。由于 Transformer 的时间复杂度中存在平方项，当序列很长时， Transformer Block 的编码时间复杂度将会迅速增加，而 RNN 的时间消耗对序列长度则是线性的。
   - Transformer 可以对整个序列并行编码，而 RNN 则需要对输入逐个编码，这使得 Transformer 在训练时相对 RNN 来说时间效率更高；生成内容时，两者都需要逐个生成内容。

   空间复杂度

   - 对于朴素的 RNN 来说，模型在编码过程中需要存储隐藏状态 h_t ，生成的隐藏状态共占据 $\mathcal{O}(dT)$ 的空间。
   - 对于单一 Transformer Block 来说，其在编码过程中除了生成的隐藏状态占据 $O(dT)$ 的空间，还需要占用 $\mathcal{O}(T^2)$ 大小的 Attention Weights。当序列特别长时，这部分的空间消耗将变得较为明显。

   性能

   - Transformer 通常比 RNN 有更好的表现。原因在于，Transformer 的 Attention 是全局的，同一序列中间隔任意远的两个输入，可以 **直接** attend 到彼此，而无需经过中间状态；而朴素 RNN 中距离较远的两个输入则难以彼此作用，因此输出可能相对而言缺乏全局信息，表现更差。
   - 此外 RNN 在训练过程中，存在矩阵反复作用的情形，因而容易出现梯度消失 / 爆炸等问题，这也可能会让它的训练有一定的困难。

   位置信息

   - RNN 的位置信息无需额外传递。在由输入 x_t 和 上一时间步的隐藏状态 h_{t-1} 获得 h_t 时，x_t 表征了当前时间步的信息，而 h_{t-1} 则传递了之前时间步的信息。由于 RNN 并不能直接与先前的输入直接作用，因此位置信息的传递也仅限于区分 *之前时间步* 和 *当前时间步* 。
   - Transformer 的位置信息则是通过位置编码来表征的，模型需要在训练过程中自行学习从 positional encoding 中提取时序信息，而 position encoding 本身也可以是可学习的。因此，Transformer 可能需要较大的数据量 / 训练量后，模型才能学到时序信息的提取。但模型学会时序信息后，则可以利用该编码直接提取特定位置的信息。

4. 就推理时间复杂度，回答以下问题

   1. 推理时使用 `use_cache=True` 。这一参数有什么作用?

      使用 `use_cache` 返回这一步计算的 K 和 V，便于此后复用本次计算结果，加快推理速度。

      在推理时，设我们希望求得对 l_t 的输出，此时 $l_0, \cdots, l_{t-1}$ 对应的 K_{t-1}, V_{t-1} 已经计算过。因此在推理 l_t 时，只需计算第 t 个时间步的 q_t, k_t, v_t ，也就是

      ```
      query, key, value = self.c_attn(hidden_states).split(self.split_size,
      dim=2)
      ```

      然后令

      $$K_t = \begin{bmatrix} K_{t-1} \\ k_t \end{bmatrix}, \quad V_t = \begin{bmatrix} V_{t-1} \\ v_t \end{bmatrix},$$

      也就是代码中的

```
past_key, past_value = layer_past
key = torch.cat((past_key, key), dim=-2)
value = torch.cat((past_value, value), dim=-2)
```

直接将先前计算的 key 和 value 拼接供本次 Attention 使用。本次 Attension 计算后再将 key 和 value 返回，供下次计算时拼接使用。

2. 设隐藏维度为 d，FeedForward 中间隐藏层维度为 $4d$，Attention Head 有 n 个，Transformer Block 有 B 个，词汇量为 V。设解码的序列为 $L = (l_0 = <|\text{endoftext}|>, l_1, \cdots, l_T)$。求 `inference` 函数循环在第 t 次推理的时间复杂度和解码整个序列 L 的时间复杂度。

   - 我们先来分析产生**单个** l_t 的时间复杂度（这里其实是把 l_1 作为了第一个 token 输入进行分析，但并不会影响渐进复杂度）

     对于单个 Transformer Block

Module	(Asymptotic) Time Complexity
Generate Q, K, V	$3d^2$
Multi-head Self-Attention	dt
Projection	d^2
Feed-Forward	$4d^2$

     所以单个 Transformer Block 的时间复杂度为 $\mathcal{O}(d^2 + dt)$

     对于将隐藏状态转换为 token logits，时间复杂度为 $\mathcal{O}(dV)$，由此可以归纳出

     - 解码单个 l_t 的时间复杂度 $\mathcal{T}(l_t) = \mathcal{O}(Bd^2 + Bdt + dV)$
     - 解码整个序列的时间复杂度 $\mathcal{T}(L) = \sum_{t=1}^{T} \mathcal{T}(l_t) = \mathcal{O}(Bd^2T + dVT + BdT^2)$

3. 当 T 相对较大时，T^2 相比 d^2 占主导，此时 Self-Attention 模块在时间复杂度中占主导

   当 d 相对较大时，d^2 相比 T^2 占主导，此时 Feed-Forward 模块在时间复杂度中占主导。

5. 从生成结果，收敛速度等角度讨论预训练的影响。为什么有这样的效果?

   从实验结果中可以看到，经过预训练的模型收敛速度更快，收敛后 loss 和 Perplexity 更低。这是因为模型在预训练过程中已经习得了一些从文本中提取信息和关系的能力。拥有这些知识的参数可以更快适应下游任务，并且利用在预训练过程中习得的语言能力在下游任务中表现出更好的泛化性。

   在本次实验中，预训练并没有显著提升模型的以 BLEU Score 为度量的生成质量，其可能的原因已经在前面几节讨论过。

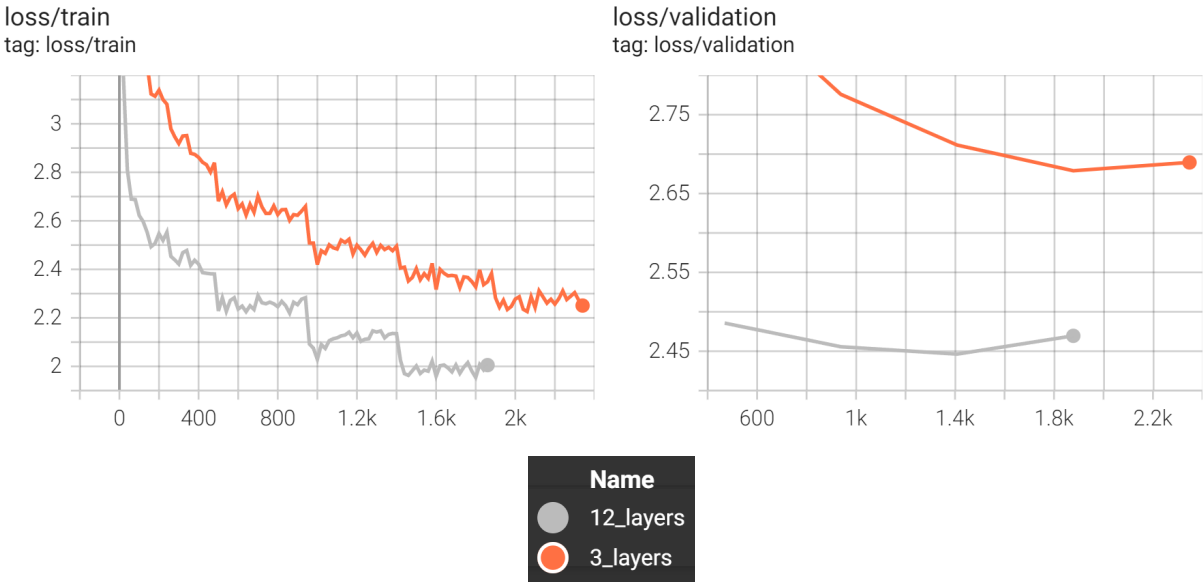
Bonus

这一部分在推理时，选择先前 Harmonic-BLEU 表现最好的解码策略，即

```
temperature=0.7
decode_strategy=random
```

1. 对比 3 层与 12 层 Transformer

使用默认参数分别微调 3 层与 12 层的 GPT-2，实验结果如下

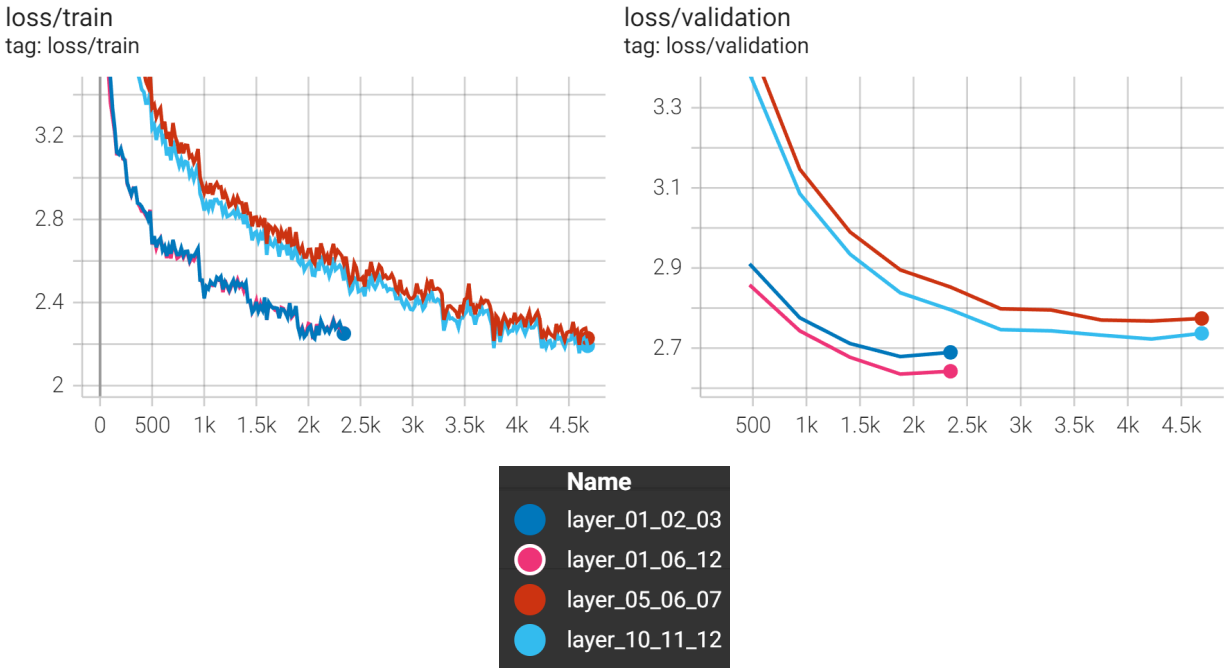


Layer Count	3-layers	12-layers
dev/Perplexity	14.57	**11.55**
test/Perplexity	13.09	**10.56**
Forward-BLEU	**0.788**	0.786
Backward-BLEU	0.491	0.491
Harmonic-BLEU	0.605	0.605

- 从训练过程来，12 层大模型比 3 层收敛更快，Loss 和 Perplexity 更低。这说明更多的预训练参数确实提升了模型的学习能力。
- 从 Perplexity 来说，12 层模型 Perplexity 比 3 层模型显著更低。更大的参数量可以使模型记忆更多的语言信息，生成更为准确的句子。
- 然而，两者的 BLEU Score 差异很小，Harmonic BLEU 差异小于 1×10^{-3} 。
    - 这可能是因为在该语料库中随机生成句子这一任务可能较为简单，并不能充分体现出大模型的优势：该语料库的规模很小，而随机生成时选取的内容大都为高概率内容。而这 3 层模型已经能较好地捕捉该语料库中的这些高概率信息，12 层在这方面额外优势并不明显。
    - 如果在更大规模的语料库进行微调，或者适配更加有挑战性的下游任务，12 层模型或许会有更加突出的优势。

2. 选取不同的 3 层进行微调

在这一节，我们选取 GPT-2 中不同位置的 3 层 Transformer Block 进行堆叠后进行微调。实验结果如下



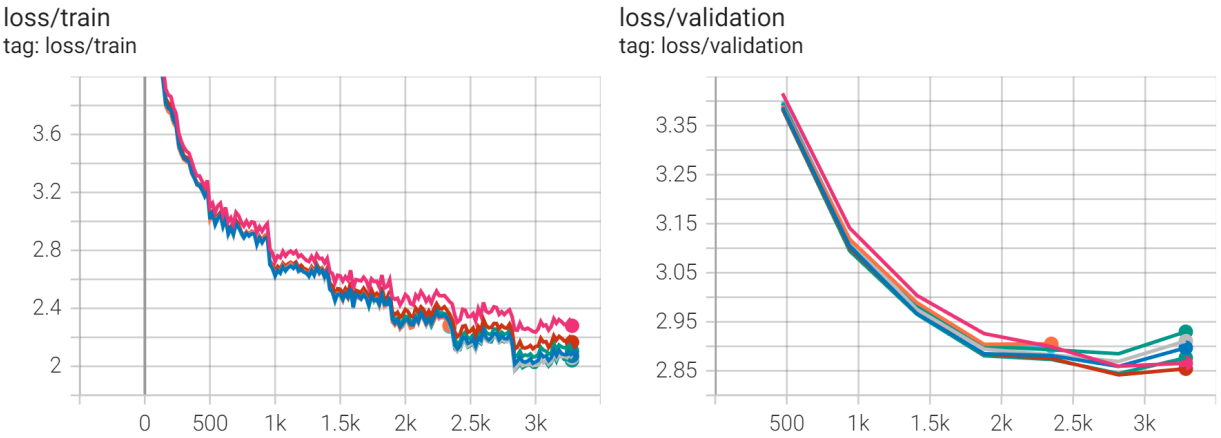
Layer Choice	$[1, 2, 3]$	$[5, 6, 7]$	$[10, 11, 12]$	$[1, 6, 12]$
dev/Perplexity	14.57	15.92	15.22	**13.95**
test/Perplexity	13.09	14.14	13.61	**12.53**
Forward-BLEU	0.788	**0.803**	0.802	0.797
Backward-BLEU	**0.491**	0.484	0.488	0.488
Harmonic-BLEU	0.605	0.604	**0.607**	0.605

其中，层数从 1 开始计数。

- 实验结果表明，跳跃选取第 $[1, 6, 12]$ 层 Transformer Block 收敛最快，Loss 和 Perplexity 最低。而选取中段和后段的模型则 Loss 收敛于较高的位置。就 BLEU-Score 来说，模型之间没有明显差异。
- 而选取 $[5, 6, 7]$，$[10, 11, 12]$ 层进行微调，其结果则和从随机初始化开始训练类似。
- **这说明在预训练过程中，不同层次的 Transformer Block 学到的内容有较为明显的差异。**不同位置的 Block 可能适合于处理不同阶段的 Hidden State。例如开始的 Block 可能适合处理贴近 Embedding 的、较为局部而具体的信息，靠近结尾的 Block 可能适合处理一些较为抽象、同时较为全局化的语义信息。因此跳跃选取不同位置 Transformer 取得了较好的效果。

3. Multi-head Attention

在默认参数的基础上，令 $\text{num\_head} = \{1, 3, 6, 12, 24, 48, 96, 192\}$ 进行训练，结果如下



Number of Head	1	3	6	12	24	48	96	192
dev/Perplexity	17.45	**17.15**	17.19	17.44	17.62	17.90	18.16	18.25
test/Perplexity	15.40	**15.09**	15.13	15.31	15.48	15.67	16.03	16.07
Forward-BLEU	0.795	0.797	0.799	0.800	**0.807**	0.805	0.799	0.801
Backward-BLEU	0.486	0.486	0.486	**0.487**	0.485	0.483	0.478	0.479
Harmonic-BLEU	0.603	0.604	0.605	0.605	**0.606**	0.604	0.598	0.600

- 训练过程中，多个相比单一的 attention head，loss 显著更低。不同 head 有利于捕捉输入序列的不同特征，因此可以加速模型的学习过程，在参数数量一定时，增强模型的特征捕获能力。
- Perplexity 随着 num_head 先减小再增大，而 BLEU-Score 则是先升高再降低。这说明适当增加 num_head 有利于提升模型的表现。
- 然而，如果 num_head 过多，则会导致拆分后的向量维数过低，包含信息过少。此时各个 head 可能更不容易 attend 到正确的向量中，或者不同 head 学到的内容倾向于接近，因此模型的学习效果会有所下降。因此，如果要提升 Attention head 数量，也需要同步提升隐藏层维数，防止拆分后维数太低。
- 在本次实验中，不同 attention head 数量带来的结果差异非常小。这可能是因为语料库较小，多 head 与单一 head 提取到的特征信息类似，模型生成能力（以 BLEU Score 衡量）并未因 head 数量而有显著变化。