a4

g.

1. **影响：**

   对于每个批次，注意力得分中对应零填充标记（zero-padded embeddings）的部分会被设置为 $-\infty$

   这样，在计算注意力分布 a_t 时，概率只会基于非填充标记的得分进行计算，而对应填充单词的得分会被忽略（近似为零）。

   最后，在计算注意力输出 A_t 时，对于每个批次，只有那些没有被接近零的概率乘以的隐藏状态才会起作用。

2. **必要性：**

   以这种方式使用掩码是确定真实注意力分布的高效方法，只涉及非填充部分。

   如果包含填充部分，将会导致错误的注意力表示。

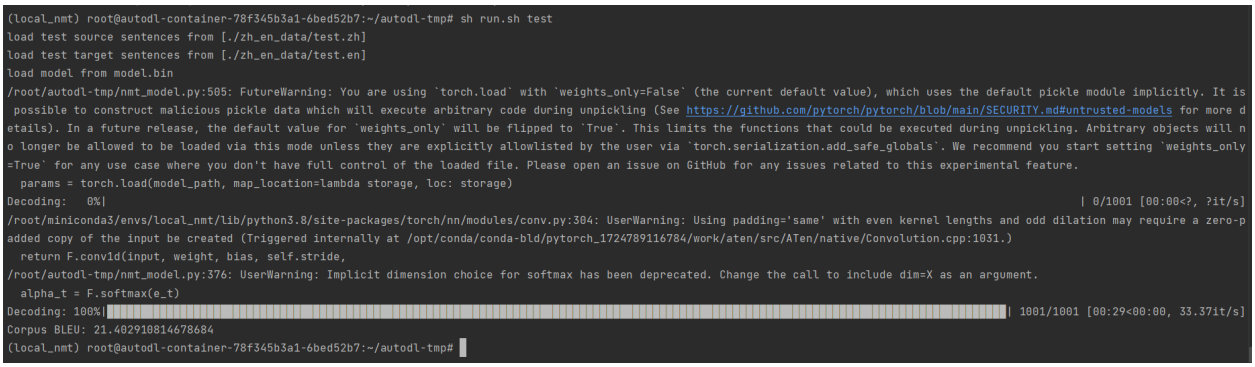
h.BLEU分数为21.40

训练过程

```
epoch 14, iter 81780, avg. loss 30.19, avg. ppl 3.29 cum. examples 56960, speed 11876.52 words/sec, time elapsed 5261.96 sec
epoch 14, iter 81790, avg. loss 32.88, avg. ppl 3.50 cum. examples 57280, speed 13556.74 words/sec, time elapsed 5262.58 sec
epoch 14, iter 81800, avg. loss 33.17, avg. ppl 3.59 cum. examples 57600, speed 13463.14 words/sec, time elapsed 5263.19 sec
epoch 14, iter 81810, avg. loss 32.79, avg. ppl 3.42 cum. examples 57920, speed 12229.55 words/sec, time elapsed 5263.89 sec
epoch 14, iter 81820, avg. loss 32.63, avg. ppl 3.50 cum. examples 58240, speed 12883.89 words/sec, time elapsed 5264.54 sec
epoch 14, iter 81830, avg. loss 31.20, avg. ppl 3.29 cum. examples 58560, speed 13098.44 words/sec, time elapsed 5265.18 sec
epoch 14, iter 81840, avg. loss 32.64, avg. ppl 3.45 cum. examples 58880, speed 14029.01 words/sec, time elapsed 5265.78 sec
epoch 14, iter 81850, avg. loss 32.49, avg. ppl 3.51 cum. examples 59200, speed 12742.77 words/sec, time elapsed 5266.43 sec
epoch 14, iter 81860, avg. loss 30.45, avg. ppl 3.20 cum. examples 59520, speed 12846.53 words/sec, time elapsed 5267.08 sec
epoch 14, iter 81870, avg. loss 33.55, avg. ppl 3.65 cum. examples 59840, speed 13481.56 words/sec, time elapsed 5267.70 sec
epoch 14, iter 81880, avg. loss 31.30, avg. ppl 3.30 cum. examples 60160, speed 13291.64 words/sec, time elapsed 5268.33 sec
epoch 14, iter 81890, avg. loss 33.00, avg. ppl 3.55 cum. examples 60480, speed 13625.86 words/sec, time elapsed 5268.94 sec
epoch 14, iter 81900, avg. loss 31.91, avg. ppl 3.39 cum. examples 60800, speed 13926.31 words/sec, time elapsed 5269.54 sec
epoch 14, iter 81910, avg. loss 34.47, avg. ppl 3.70 cum. examples 61120, speed 12239.12 words/sec, time elapsed 5270.23 sec
epoch 14, iter 81920, avg. loss 31.56, avg. ppl 3.36 cum. examples 61440, speed 12408.19 words/sec, time elapsed 5270.90 sec
epoch 14, iter 81930, avg. loss 30.64, avg. ppl 3.29 cum. examples 61760, speed 13581.44 words/sec, time elapsed 5271.51 sec
epoch 14, iter 81940, avg. loss 29.76, avg. ppl 3.14 cum. examples 62080, speed 14157.02 words/sec, time elapsed 5272.10 sec
epoch 14, iter 81950, avg. loss 30.56, avg. ppl 3.24 cum. examples 62400, speed 13154.70 words/sec, time elapsed 5272.73 sec
epoch 14, iter 81960, avg. loss 32.90, avg. ppl 3.54 cum. examples 62720, speed 13077.99 words/sec, time elapsed 5273.37 sec
epoch 14, iter 81970, avg. loss 31.96, avg. ppl 3.40 cum. examples 63040, speed 13694.98 words/sec, time elapsed 5273.98 sec
epoch 14, iter 81980, avg. loss 29.94, avg. ppl 3.22 cum. examples 63360, speed 12883.69 words/sec, time elapsed 5274.61 sec
epoch 14, iter 81990, avg. loss 32.56, avg. ppl 3.61 cum. examples 63680, speed 13772.44 words/sec, time elapsed 5275.20 sec
epoch 14, iter 82000, avg. loss 30.13, avg. ppl 3.11 cum. examples 64000, speed 14009.23 words/sec, time elapsed 5275.81 sec
epoch 14, iter 82000, cum. loss 32.95, cum. ppl 3.56 cum. examples 64000
begin validation ...
validation: iter 82000, dev. ppl 11.696337
hit patience 5
hit #5 trial
early stop!
(local_nmt) root@autodl-container-78f345b3a1-6bed52b7:~/autodl-tmp#
```

训练过程可视化

测试结果



i.i

优点：点积注意力

- **计算效率高**：
  点积注意力的公式是
  $e_{t,i} = s_t^T h_i$ ，它直接进行向量的点积运算，没有额外的矩阵乘法（如乘性注意力中的 W ），因此计算量较小，计算效率更高。

缺点：点积注意力

- **无法调整不同维度的特征**：
  点积注意力直接计算两个向量的点积，当向量维度较高时，点积的数值会变得很大，可能导致数值不稳定；另外，它无法通过参数 W 调整权重分布，灵活性较低。

i.ii

优点：加性注意力

- **对向量维度更鲁棒**：
  加性注意力通过非线性变换（如 tanh ）和可学习的权重矩阵
  W_1 和 W_2 ，对输入向量的不同特征进行加权，有更好的表达能力，并且不容易受到输入向量维度大小的影响。

缺点：加性注意力

- **计算效率较低**：
  加性注意力涉及额外的矩阵变换（

$W_1 h_i + W_2 s_t$ ）和非线性操作（tanh），相比乘性注意力的简单矩阵乘法 Wh_i ，计算复杂度更高，尤其在处理长序列时可能成为性能瓶颈。

2.a

1. **捕捉局部上下文信息**：

   一维卷积层可以学习到字符或词片段之间的局部模式。例如，中文中字符“电”和“脑”独立时含义不同，但组合成“电脑”后有全新的含义。一维卷积能够捕捉到这些相邻字符之间的关联关系，从而更好地建模子词到单词的语义。

2. **处理多字符词汇的复杂性**：

   中文中，许多单词由多个字符组合而成，例如“计算机”是由三个字符组成的复杂词汇。一维卷积可以帮助整合这些字符级别的信息，生成更有意义的词级表示，从而更准确地捕捉语义。

3. **增强模型的特征表达能力**：

   通过卷积层提取的特征，嵌入层的单个字符表示可以转换为更高层次的语义表示。这使得后续的双向编码器可以处理更丰富的上下文信息，提高翻译的整体效果。

4. **减少对句子分割的依赖**：

   一维卷积层能够缓解分词错误的影响。即使分词不准确，卷积层仍然可以有效捕捉片段间的关系，提高模型的鲁棒性。

2.b

i.

- **识别 NMT 翻译中的错误**：

  NMT 翻译中“the culprit”将复数的“culprits”翻译成了单数形式。

- **模型可能出错的原因**：

  模型可能没有很好地捕捉复数形式的上下文，因为中文中没有复数形式标记。

- **可能的修改方式**：

  引入更多带有复数形式的训练数据，以帮助模型更好地理解和生成复数形式。

ii.

1. **识别 NMT 翻译中的错误**：

   下划线部分的翻译是“resources have been exhausted”，在句子中重复出现。这表明模型生成的翻译中存在重复现象。

2. **模型可能出错的原因**：

   - **语言生成过程中的重复问题**：模型可能在生成目标句时，注意力机制或解码器未能有效捕捉上下文，导致重复生成相同的片段。
   - **缺乏多样性约束**：训练数据或解码策略可能缺乏对重复问题的约束，例如使用贪婪搜索而不是引入多样性搜索策略（如采样或束搜索）。
   - **缺乏训练数据多样性**：模型可能对训练数据中的模式过拟合，导致在句子结构不明确时倾向于重复高频模式。

3. **可能的修改方式**：

   - **引入重复惩罚机制**：在生成过程中，可以对重复出现的单词或短语增加惩罚分数。例如，在束搜索（Beam Search）中，对已经生成的词进行概率调整，降低重复词的选择概率。
   - **改进解码策略**：采用更复杂的解码算法，例如多样性束搜索（Diverse Beam Search）或随机采样（Stochastic Sampling），以减少单一模式的生成。
   - **优化注意力机制**：通过改进注意力机制，让模型更好地理解句子的整体结构和词汇的分布，避免重复关注相同部分。
   - **增强数据质量**：使用去重后的高质量训练数据，或通过数据增强策略生成更多多样化的翻译实例，以提高模型对重复现象的鲁棒性。

iii.

1. **识别 NMT 翻译中的错误**：

   下划线部分 "today's day" 是一个语义错误，翻译并未准确表达源句中 "国殇日" 的含义，而是生成了不符合上下文的词组 "today's day"。

2. **模型可能出错的原因**：

   - **词汇级别语义理解不足**：模型可能未能准确捕捉到 "国殇日" 的含义，而是逐字翻译为 "today's day"，缺乏对固定表达（如 "national mourning day"）的识别

能力。

- **上下文语境建模不足**：模型未能从上下文中理解 "国殇日" 是一种特定的国家事件，而将其简单处理为普通日期表达。
- **训练数据不足**：训练数据中可能缺乏类似 "国殇日" 这种固定短语的翻译示例，导致模型对这一表达的翻译能力较弱。

3. **可能的修改方式**：
   - **改进词汇嵌入或预训练模型**：通过使用更大的预训练模型（如 BERT 或 GPT）生成更丰富的词汇表示，帮助模型更好地理解固定表达的语义。
   - **增强训练数据**：在训练数据中加入更多类似 "国殇日" 的固定短语和相关翻译，以提高模型对这些短语的翻译能力。
   - **引入术语表或固定短语翻译机制**：在模型中加入术语表（glossary）功能，对特定短语如 "国殇日" 提供直接映射的翻译选项，避免模型生成不准确的翻译。
   - **改进注意力机制**：优化注意力机制，帮助模型更好地结合上下文信息，识别特殊词组的确切含义。

iv.

1. **识别 NMT 翻译中的错误**：

   下划线部分 “it’s not wrong” 是错误的翻译。原句是引用了“俗语”中的固定表达“勿做勿错”（act not, err not），而模型生成了不同含义的翻译，失去了原句的哲学含义和措辞结构。

2. **模型可能出错的原因**：
   - **固定表达识别问题**：模型未能识别“勿做勿错”是一种固定的、语义丰富的表达，而是逐字生成翻译。
   - **语境缺乏建模**：模型未能理解整句上下文是引用俗语的表达，而是将其翻译为普通陈述句。
   - **训练数据不足**：训练数据中可能缺乏对类似俗语或固定短语的翻译示例，导致模型在生成类似句子时无法准确传达其意义。
3. **可能的修改方式**：

- **固定短语翻译增强**：在训练数据中加入更多类似“勿做勿错”的俗语和固定表达，以及它们的翻译示例，以帮助模型学习如何正确翻译这些表达。
- **引入术语表或固定翻译规则**：为模型引入术语表，明确固定短语的正确翻译方式。例如，“勿做勿错”应始终翻译为“act not, err not”。
- **增强注意力机制**：优化注意力机制，使模型能够更好地理解上下文并聚焦于整个短语，而不是逐字翻译。
- **使用预训练语言模型**：引入更强大的预训练语言模型（如 GPT 或 BERT）以增强模型对固定表达的语义理解能力。

c.i

(c) i. $$p_n = \frac{\sum_{ngram \in c} \min\left(\max_{i=1,\dots,k} Count_{r_i}(ngram), Count_c(ngram)\right)}{\sum_{ngram \in c} Count_c(ngram)}$$

c_1: (c_1, r_1, r_2)

1-gram there $(1,0,0)$ is $(1,0,0)$ a $(1,0,0)$
need $(1,0,0)$ for $(1,0,0)$ adequate $(1,0,1)$
and $(1,1,1)$ predictable $(1,1,1)$ resources $(1,1,1)$

2-gram there is $(1,0,0)$ is a $(1,0,0)$ a need $(1,0,0)$
need for $(1,0,0)$ for adequate $(1,0,0)$
adequate and $(1,0,1)$ and predictable $(1,0,1)$
predictable resources $(1,0,1)$

c_2:

1-gram resources $(1,1,1)$ be $(1,2,0)$ sufficient $(1,1,0)$
and $(1,1,1)$ predictable $(1,1,1)$ to $(1,2,0)$

2-gram resources be $(1,0,0)$ be sufficient $(1,1,0)$
sufficient and $(1,1,0)$ and predictable $(1,0,1)$
predictable to $(1,0,0)$

$len(c_1) = 9$ $len(c_2) = 6$ $len(r) = \min(len(r_1), len(r_2))$
$= \min(11, 6)$
$= 6$

c_1:

$$p_1 = \frac{0+0+0+0+0+1+1+1+1}{1+1+1+1+1+1+1+1+1} = \frac{4}{9}$$

$$p_2 = \frac{0+0+0+0+0+1+1+1}{1+1+1+1+1+1+1+1} = \frac{3}{8}$$

$BP = 1$

$BLEU = 1 \times \exp(0.5 \cdot \log\frac{4}{9} + 0.5 \cdot \log\frac{3}{8})$

≈ 0.678

C_2:

$p_1 = \frac{1+1+1+1+1+1}{1+1+1+1+1+1} = 1$

$p_2 = \frac{0+1+1+1+0}{1+1+1+1+1} = \frac{3}{5}$

$BP = 1$

$BLEU = 1 \times \exp(0.5 \cdot \log 1 + 0.5 \times \log\frac{3}{5})$

≈ 0.895

因 $BLEU_{C_1} < BLEU_{C_2}$ 故根据 BLEU 分数, C_2 更好. 但我不同意该结果, 因为 C_1 更完整自然, 而 C_2 语法不正确且不完整

c.ii

ii.

c_1: (c_1, r_2)

1-gram there (1, 0) is (1, 0) a (1, 0)
need (1, 0) for (1, 0) adequate (1, 1)
and (1, 1) predictable (1, 1) resources (1, 1)

2-gram there is (1, 0) is a (1, 0) a need (1, 0)
need for (1, 0) for adequate (1, 0)
adequate and (1, 1) and predictable (1, 1)
predictable resources (1, 1)

c_2:

1-gram resources (1, 1) be (1, 0) sufficient (1, 0)
and (1, 1) predictable (1, 1) to (1, 0)

2-gram resources be (1, 0) be sufficient (1, 0)
sufficient and (1, 0) and predictable (1, 1)
predictable to (1, 0)

$len(c_1) = 9$ $len(c_2) = 6$ $len(r_2) = 6$

c_1:

$$p_1 = \frac{0+0+0+0+0+1+1+1+1}{1+1+1+1+1+1+1+1+1} = \frac{4}{9}$$

$$p_2 = \frac{0+0+0+0+0+1+1+1}{1+1+1+1+1+1+1+1} = \frac{3}{8}$$

$BP = 1$

$$BLEU = 1 \times \exp(0.5 \cdot \log\frac{4}{9} + 0.5 \times \log\frac{3}{8})$$

$$\approx 0.678$$

c_2:

$$p_1 = \frac{1+0+0+1+1+0}{1+1+1+1+1+1} = \frac{1}{2}$$

$$p_2 = \frac{0+0+0+1+0}{1+1+1+1+1} = \frac{1}{5}$$

$$BP = 1$$

$$BLEU = 1 \times \exp(0.5 \cdot \log\frac{1}{2} + 0.5 \cdot \log\frac{1}{5})$$

$$\approx 0.607$$

因 $BLEU_{c_1} > BLEU_{c_2}$ 故根据 BLEU 分数，c_1 更好。我同意该结果，因为 c_1 更完整自然，而 c_2 语法不正确且不完整

c.iii

1. **单一参考翻译的局限性**：语言的多样性导致同一句话可以有多种正确的翻译。如果只用一个参考翻译作为评估标准，模型生成的其他可能正确但与参考翻译不同的句子可能会被错误地认为是质量较差的翻译，从而导致 BLEU 分数较低。
2. **多参考翻译的优势**：当存在多个参考翻译时，BLEU 分数可以更全面地反映翻译质量，因为它能够更好地涵盖目标语言中的多样性。这有助于避免因语言表达方式不同而对模型进行不公平的评估。
3. **影响评估的客观性**：单一参考翻译会限制评估的客观性，使评估结果过于依赖于参考翻译的特定表达，而非实际的翻译语义质量。

c.iv

优点：

1. **自动化评估**：

   BLEU 可以快速计算，不需要人工参与评估，大大节省了时间和人力成本，特别是在大规模数据集上的评估中具有优势。

2. **一致性和客观性**：

   BLEU 使用固定的算法评估模型输出，避免了人工评估中可能存在的主观性和评估者之间的不一致性。

缺点：

1. **忽略语义正确性**：

   BLEU 分数主要基于 n-gram 匹配，不能很好地评估翻译的语义是否正确。如果候选翻译与参考翻译的表达方式不同，但语义正确，BLEU 可能会给出低分。

2. **对单一参考翻译的依赖**：

   当只有一个参考翻译时，BLEU 可能无法准确衡量翻译质量，因为语言表达具有多样性。模型生成的合理翻译可能因为与参考翻译不匹配而被误判为错误翻译。