自然语言处理实践任务

2023 年 1 月 10 日

简 表

硬件环境（CPU/GPU）： CPU
操作系统： Windows11
采用的深度学习框架、工具、语言： 框架：Pytorch， 工具：PyCharm， 语言：Python
任务描述/问题定义： 使用 PyTorch 实现 Transformer 模型，能够基本完成英语翻译成捷克语的任务。由于模型训练时间过长，所以设置较小的 batch_size 和 epoch，并且选用了一部分的源数据。
数据集及来源（相关链接）： https://s3.amazonaws.com/web-language-models/paracrawl/release5.1/en-cs.txt.gz

采用的深度学习模型： Transformer	模型提出的年份及发表的会议/期刊： 2017 年 google 的机器翻译团队在 NIPS 上发表了 attention is all you need 的文章

最终设置的超参数（如 Learning rate, Batch size 等）： Learning Rate：0.001, Batch size: 16, Epochs: 15
模型的效果（如准确率等与任务相关的评价指标）： cross entropy 损失值

	成员 1	成员 2	成员 3
姓名	姚明阔		
学号	2201902		
专业	计算机科学与技术		
工作量占比	100%		

1. 采用的深度学习模型的详细描述

Transformer 模型由 N 个 Encoder 模块（下图左边黑框），N 个 Decoder 模块（下图右边黑框）和一个 Linear 层组成。

Encoder 模块：一个 Encoder 有两个子层，一个是 Multi-Head Attention 层，是利用 self-attention 学习源句子内部的关系。另一个是 Feed Forward 层，他就是一个简单的全连接网络，目的是增加非线性，之后产生的输出传给 Decoder。每一个子层后都使用了残差连接和 Layer Normalization 来保证模型不会随着深度的增加而效果变差。其输入是位置编码和源句编码。

Decoder 模块：Decoder 中有三个子层，其中两个 Multi-Head Attention 层。下面的 Attention 层是利用 self-attention 学习目标句内部的关系，之后该层的输出与 encoder 传过来的结果一起输入到上面的 Attention 层，这个 Attention 层并不是 self-attention，而是 Encoder-Decoder attention，用来学习源句与目标句之间的关系。对于第二层 Attention，query 代表 Decoder 中第一层的输出，key 和 value 是来自 Encoder 的输出。第一层 Attention 的 mask 指的是 sequence mask，是为了实现无法从当前了解到以后信息的目的，因为不论是 RNN 还是 LSTM，他们都需要等待上一阶段的输出，而 self-attention 不同，他是并行计算的。

此模型实现不仅需要对 sequence 进行 mask，还需要对 padding 进行 mask。

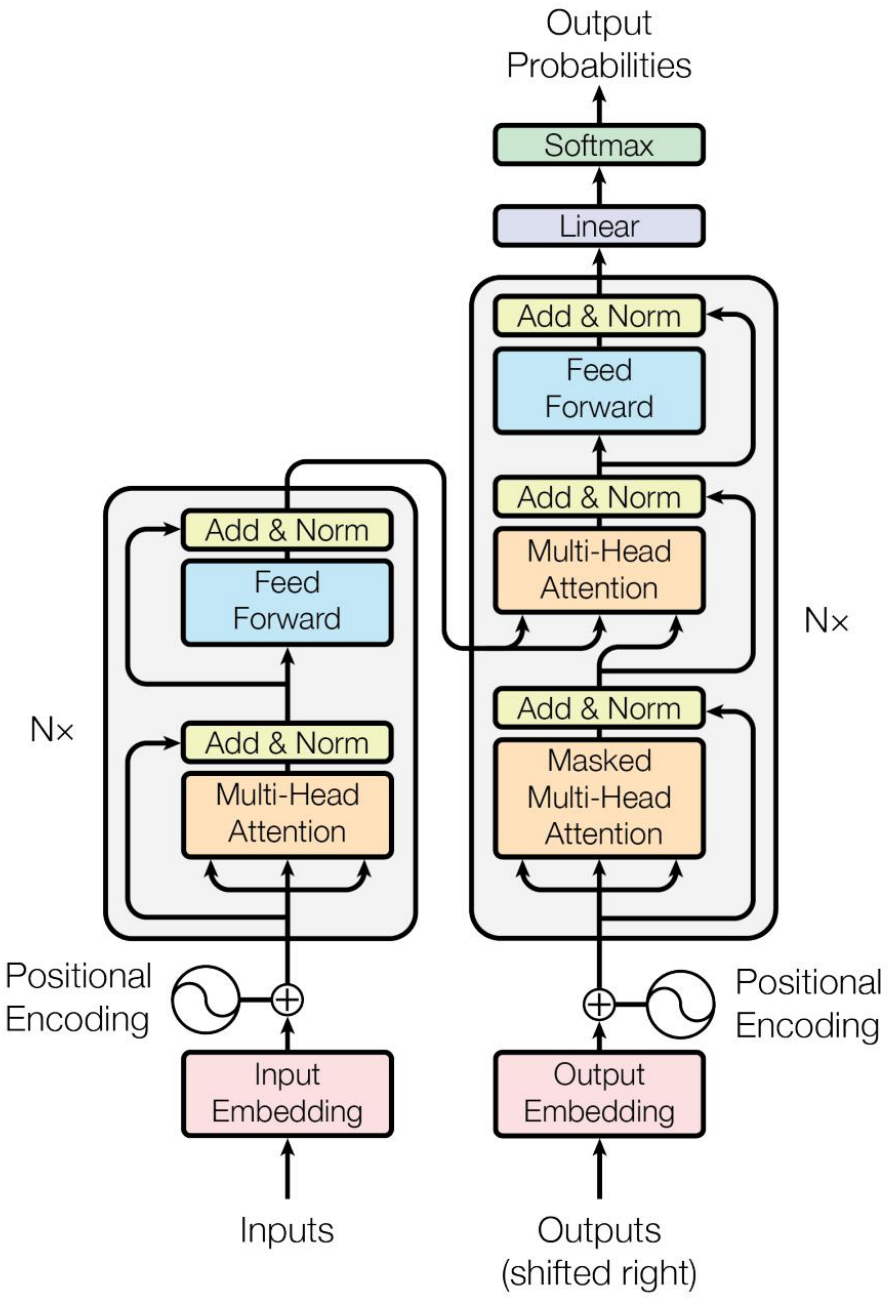

图 1 Transformer 模型框架图

2. 训练曲线（Loss-Epoch）

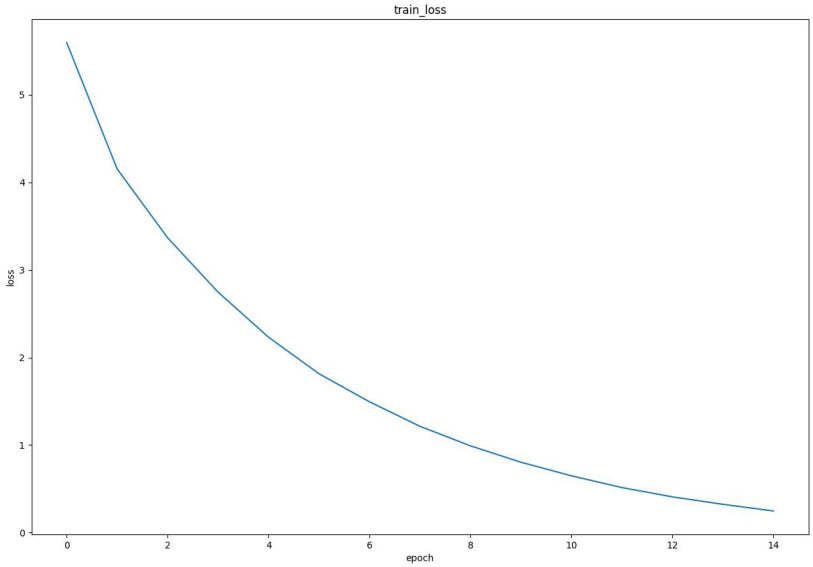

图 2 训练损失值下降曲线

3. 运行或运行结果截图

```
success
5223 Service activities incidental to air transportation --> ['5223', 'činnosti', 'související', 's', 'leteckou', 'dopravou']
<<<<<<< finished evaluate

Process finished with exit code 0
```