自然语言处理中期报告

姓名：饶文均

时间：2022.11.26

当前进度

我查看SQuAD 数据集官网https://rajpurkar.github.io/SQuAD-explorer/后，发现很多团队使用了Google BERT框架，都得到了很好的效果。在项目中期阶段，另一名组员负责尝试用TF-IDF算法处理数据集，而我则尝试微调Google BERT模型，通过神经网络的方式处理数据集。

BERT模型理解

模型简介

BERT是Bidirectional Encoder Representations from Transformers的缩写，是一种双向编码器，旨在通过在**左右上下文中共有的条件**计算来预先训练来自无标号文本的深度双向表示。因此，经预先训练的BERT模型只需一个额外的输出层就可以进行微调，从而为各种自然语言处理任务服务，生成最新的模型。

模型优点

- 由于BERT 模型的预训练是包含整个Wiki 百科的大语料库和图书语料库中进行的，所以这个模型的训练是非常充分的，模型在大量的训练中获得对语言知识的更深入理解，对其后续处理各种自然语言任务都是很有利的。而本次处理的SQuAD 数据集中所选择的文章恰好都来自Wiki 百科，说明使用BERT 模型来处理SQuAD 数据集是非常合适的。
- BERT 模型是一个“深度双向”的模型。“双向”意味着BERT 模型在训练阶段从所选文本的左右上下文中汲取信息。BERT 模型的双向性对于理解语言的真正意义很重要。在本次处理的SQuAD 数据集中，同样要求机器通过分析文章的上下文得到问题的答案，所以BERT 模型适合处理本次的SQuAD数据集。
- 此外，BERT 模型还可以通过添加几个额外的输出层来进行微调，大大提高了模型对任务的适配度和精度，拓展性较高。

环境配置

- 使用Anaconda 创建pytorch实验环境。
- 在实验环境中安装python 3.8/pytorch/sklearn/pandas等常用包。

- 在pycharm项目中修改环境变量。

```
>>> conda create -n pytorch
>>> conda activate pytorch
>>> conda install python == 3.8
>>> conda install pytorch torchvision torchaudio
-c pytorch
```

模型获取

- 从https://github.com/google-research/bert 仓库中pull下**Google BERT** 官方的代码。
- 再从 https://storage.googleapis.com/bert_models/2019_05_30/wwm_uncased_L-24_H-1024_A-16.zip 下载预训练模型**BERT-Large, Uncased (Whole Word Masking)**。
- 从https://github.com/huggingface/transformers pull 下transformer 工具，将上一步下载的模型转换为pytorch 可以调用的模型。再导入项目中。

进行实验

pytorch环境已部署完毕，模型可以试运行，在我的电脑中预计运行时间为36小时。在本次试运行中，我们将观察模型运行状态和数值变化。

实验环境

硬件：MacBook Pro (13-inch, 2020, Four Thunderbolt 3 ports).

软件：macOS Monterey 12.6 Version.

参数设置

batch size

论文**BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding**中将batch size设为**16**，本次试运行中受设备内存限制，将batch size微调至**12**。我们预测这样的改变会使结果不那么收敛，有较大**发散**。

epoch

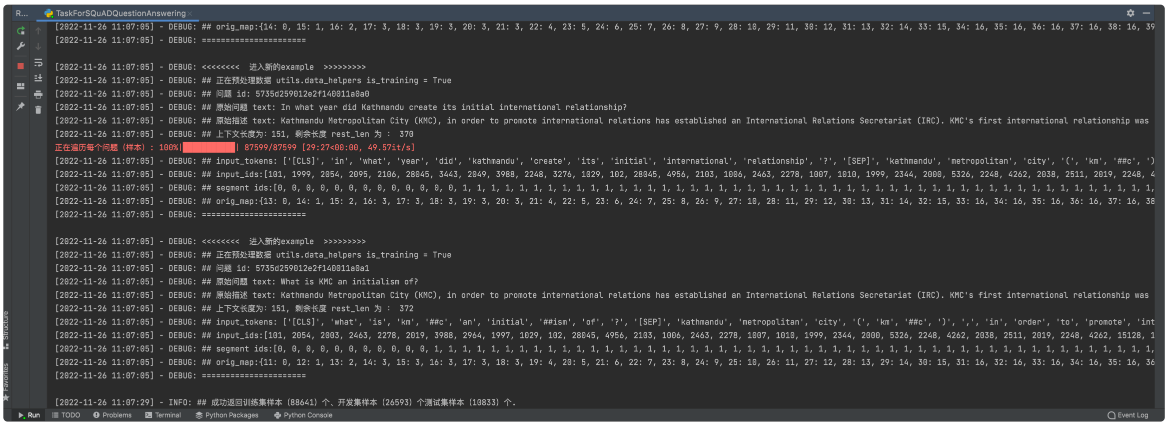
受电脑性能影响，暂设为**1**。

Learning Rate

遵循论文**BERT: Pre-training of Deep Bidirectional Transformers forLanguage Understanding**中将learning rate值设为**5e-5**。

模型运行状况分析

- 模型运行后，对数据进行预处理，将数据集分为训练集样本（88641）个、开发集样本（26593）个测试集样本（10833）个，再对每条训练样本数据进行分析学习。



- 根据设置的batch size，模型将数据分为7387个batch进行迭代，模型每迭代处理10个batch 就输出当前的Train loss和Train accuracy值供我们参考。可以看到初期Train loss较高，模型准确率Train accuracy非常低。

```
[2022-11-26 11:11:14] - INFO: Epoch: 0, Batch[10/7387], Train loss :5.129, Train acc: 0.125
[2022-11-26 11:14:13] - INFO: Epoch: 0, Batch[20/7387], Train loss :4.498, Train acc: 0.042
[2022-11-26 11:17:55] - INFO: Epoch: 0, Batch[30/7387], Train loss :4.212, Train acc: 0.000
[2022-11-26 11:21:12] - INFO: Epoch: 0, Batch[40/7387], Train loss :3.468, Train acc: 0.208
[2022-11-26 11:23:49] - INFO: Epoch: 0, Batch[50/7387], Train loss :3.076, Train acc: 0.250
[2022-11-26 11:26:38] - INFO: Epoch: 0, Batch[60/7387], Train loss :3.447, Train acc: 0.333
[2022-11-26 11:29:14] - INFO: Epoch: 0, Batch[70/7387], Train loss :2.927, Train acc: 0.292
[2022-11-26 11:32:10] - INFO: Epoch: 0, Batch[80/7387], Train loss :2.803, Train acc: 0.333
[2022-11-26 11:34:51] - INFO: Epoch: 0, Batch[90/7387], Train loss :2.516, Train acc: 0.375
[2022-11-26 11:37:28] - INFO: Epoch: 0, Batch[100/7387], Train loss :2.798, Train acc: 0.375
```

- 每进行100次iteration，就输出一次问答过程，供我们观察模型对问题答案的预测情况。可以看到目前模型对问题答案的预测效果并不理想。

```
[2022-11-26 11:37:28] - INFO: ### Question: [CLS] what led to a reduction in the power of the peers ?
[2022-11-26 11:37:28] - INFO:   ## Predicted answer: 1910
[2022-11-26 11:37:28] - INFO:   ## True answer: conflict between the two houses of parliament over the people ' s budget
[2022-11-26 11:37:28] - INFO:   ## True answer idx: (tensor(130), tensor(142))
[2022-11-26 11:37:28] - INFO: ### Question: [CLS] who is the most recent member to join the ibm board of directors ?
[2022-11-26 11:37:28] - INFO:   ## Predicted answer: 14 member board of directors is responsible for overall corporate management . as of cathie black
[2022-11-26 11:37:28] - INFO:   ## True answer: andrew n . liveris
[2022-11-26 11:37:28] - INFO:   ## True answer idx: (tensor(122), tensor(126))
[2022-11-26 11:37:28] - INFO: ### Question: [CLS] what was youtube ' s revenue as estimated in 2008 ?
[2022-11-26 11:37:28] - INFO:   ## Predicted answer: 200 million
[2022-11-26 11:37:28] - INFO:   ## True answer: $ 200 million
[2022-11-26 11:37:28] - INFO:   ## True answer idx: (tensor(58), tensor(60))
[2022-11-26 11:37:28] - INFO: ### Question: [CLS] when was the ifab ' s decision on the fixed size of the pitch become implemented ?
[2022-11-26 11:37:28] - INFO:   ## Predicted answer: 2008
[2022-11-26 11:37:28] - INFO:   ## True answer: never
[2022-11-26 11:37:28] - INFO:   ## True answer idx: (tensor(152), tensor(152))
[2022-11-26 11:37:28] - INFO: ### Question: [CLS] who could naturalize a polish noble ?
[2022-11-26 11:37:28] - INFO:   ## Predicted answer:
[2022-11-26 11:37:28] - INFO:   ## True answer: polish king
[2022-11-26 11:37:28] - INFO:   ## True answer idx: (tensor(71), tensor(72))
```

- 进行4000次iteration后，一个epoch 已运行过半，我们可以观察到Train loss值总体上大幅降低，准确率Train accuracy大幅提高到接近80%，但也可以观察到这两个值的变化较不稳定，有跳变的情况出现。通过机器的Predicted answer可以看出，模型对问题答案的预测情况变好，对于一些问题甚至可以一字不差地预测出准确答案。

```
[2022-11-27 04:53:50] - INFO: Epoch: 0, Batch[4080/7387], Train loss :0.989, Train acc: 0.667
[2022-11-27 04:56:02] - INFO: Epoch: 0, Batch[4090/7387], Train loss :1.321, Train acc: 0.625
[2022-11-27 04:58:15] - INFO: Epoch: 0, Batch[4100/7387], Train loss :1.557, Train acc: 0.542
[2022-11-27 04:58:15] - INFO: ### Question: [CLS] which species is commonly found more in spruce - fir forests ?
[2022-11-27 04:58:15] - INFO:   ## Predicted answer: appalachian northern flying squirrel
[2022-11-27 04:58:15] - INFO:   ## True answer: appalachian northern flying squirrel
[2022-11-27 04:58:15] - INFO:   ## True answer idx: (tensor(102), tensor(105))
[2022-11-27 04:58:15] - INFO: ### Question: [CLS] which look of madonna became a fashion trend ?
[2022-11-27 04:58:15] - INFO:   ## Predicted answer: spanish
[2022-11-27 04:58:15] - INFO:   ## True answer: spanish look
[2022-11-27 04:58:15] - INFO:   ## True answer idx: (tensor(120), tensor(121))
[2022-11-27 04:58:15] - INFO: ### Question: [CLS] as a result , what became the religion of galician society ?
[2022-11-27 04:58:15] - INFO:   ## Predicted answer: romance language
[2022-11-27 04:58:15] - INFO:   ## True answer: christian
[2022-11-27 04:58:15] - INFO:   ## True answer idx: (tensor(88), tensor(88))
[2022-11-27 04:58:15] - INFO: ### Question: [CLS] in what geographic region did most of the invaders settle in gaul ?
[2022-11-27 04:58:15] - INFO:   ## Predicted answer: north - east
[2022-11-27 04:58:15] - INFO:   ## True answer: north - east
[2022-11-27 04:58:15] - INFO:   ## True answer idx: (tensor(50), tensor(52))
[2022-11-27 04:58:15] - INFO: ### Question: [CLS] what does dna consist of ?
[2022-11-27 04:58:15] - INFO:   ## Predicted answer: a chain made from four types of nucleotide subunits
[2022-11-27 04:58:15] - INFO:   ## True answer: a chain made from four types of nucleotide subunits
[2022-11-27 04:58:15] - INFO:   ## True answer idx: (tensor(38), tensor(49))
[2022-11-27 05:00:27] - INFO: Epoch: 0, Batch[4110/7387], Train loss :1.473, Train acc: 0.458
[2022-11-27 05:02:39] - INFO: Epoch: 0, Batch[4120/7387], Train loss :1.576, Train acc: 0.625
[2022-11-27 05:04:55] - INFO: Epoch: 0, Batch[4130/7387], Train loss :0.580, Train acc: 0.750
[2022-11-27 05:07:07] - INFO: Epoch: 0, Batch[4140/7387], Train loss :0.934, Train acc: 0.750
[2022-11-27 05:09:18] - INFO: Epoch: 0, Batch[4150/7387], Train loss :1.252, Train acc: 0.625
[2022-11-27 05:11:32] - INFO: Epoch: 0, Batch[4160/7387], Train loss :1.770, Train acc: 0.542
TODO  Problems  Terminal  Python Packages  Python Console
```

模型运行中Train loss 值和Train accuracy 值都会出现跳变情况，振幅较大，我们认为这是减小了butch size 的原因。

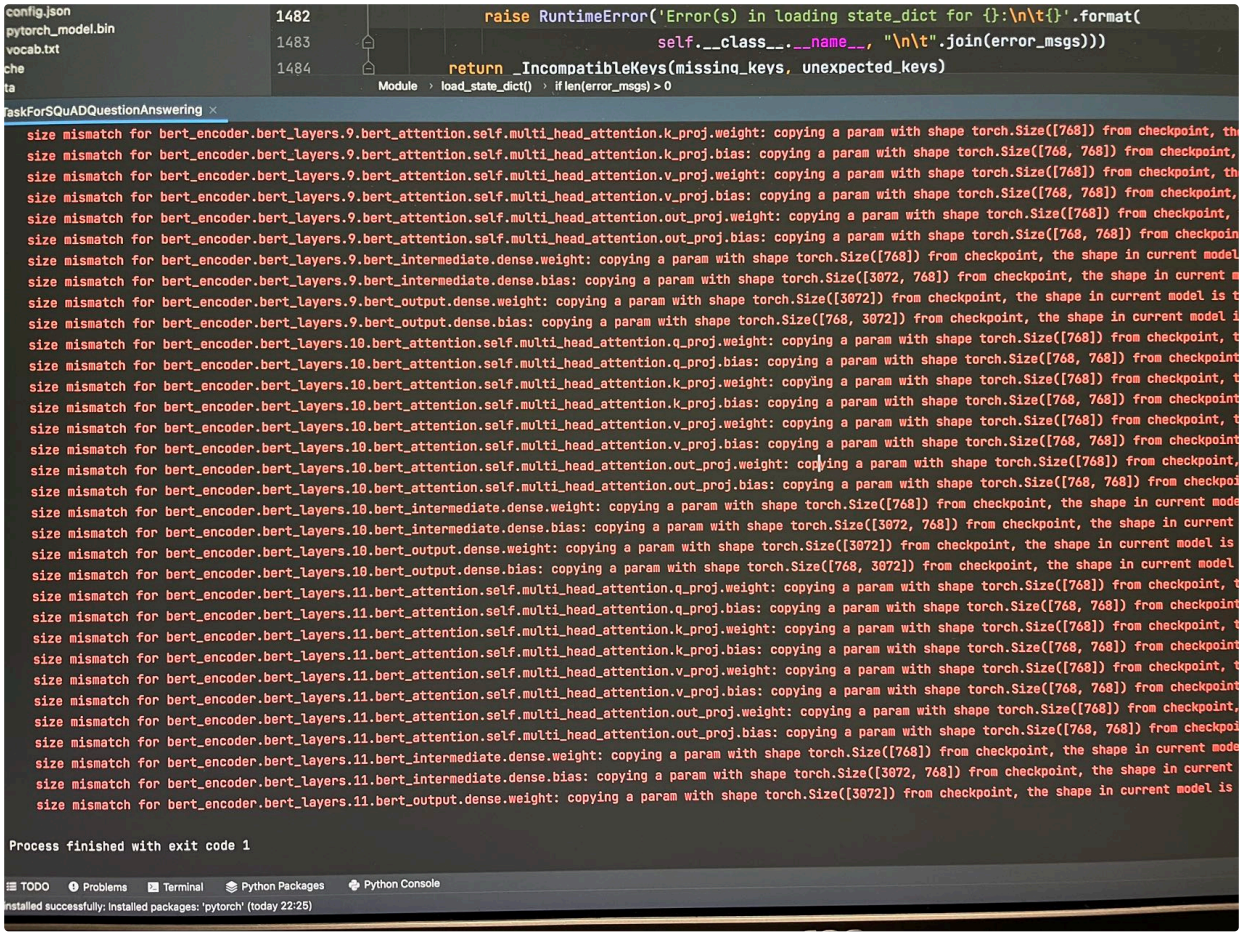
遇到的问题及解决

环境配置问题

在终端通过Anaconda 创建虚拟环境后安装了pytorch ，但在pycharm中进入虚拟环境后使用conda list 指令却没有显示pytorch 包。

解决办法：在pycharm -> preferences -> python Interpreter -> package 中点击 '+' 号，手动安装pytorch 包后解决。

size mismatch问题



查阅资料后，发现是因为使用的预训练模型选择错误。

解决办法：重新从 https://storage.googleapis.com/bert_models/2018_10_18/uncased_L-12_H-768_A-12.zip下载**BERT-Based-Uncased** 模型后解决。

torch.load() 函数问题

出现报错：torch.load with map_location=torch.device(“cpu”) to map your storages to the CPU.

解决办法：因为我的电脑是macbook ，其GPU 不支持torch 训练神经网络，所以必须强制规定该模型在CPU 上运行。在torch.load()中加入参数：**map_location=torch.device('CPU')** 即可。

个人思考理解

在这一部分，我打算从BERT 模型的演变历程，自底向上地谈谈我个人对过程的理解和思考。

Attention

对于我们熟悉的语言（如母语），在我们人脑处理自然语言的过程中，我们总是会在不经意间关注句子中的关键词，从而帮助我们达到理解整个句子的目的。对于复杂的句式，我们也总能通过一些前后文关键词来理解某个词语在句子中的特定含义。下面我举个例子：

> 我儿子说今天在他舍友桌子上看到了新发布的苹果，他也想买一部，但我拒绝了。

如果我们想要理解句中“苹果”一词的含义，就需要结合前后文本。在此长句中，为了理解“苹果”这个词语，我们会优先关注形容词“新发布的”和量词“一部”，这样可以帮助我们判断出此句中的“苹果”指的是Apple的电子产品而不是一种水果名称。而对于句子中“今天”、“桌子”等词语，我们会认为它们对于理解“苹果”一词不那么重要，也就是**权重**很低，而这种权重就是Attention。Attention 机制就是通过**抑制无关元素**达到精确化模型的目的。

Attention 机制从数图领域被引入NLP 领域后，改进了先前基于RNN 的Encoder-Decoder 模型，通过在每一时刻的隐层状态中加入每个词语的权重，得到不同的编码，最后通过训练得到最好的Attention 分布。这样可以让模型在每一时刻都能动态地看到全局信息，将注意力集中到权重高的context 中，从而提高模型精度。

但Attention 机制也是有缺点的。通过对比BERT 模型可以发现，Attention 无法学习输入序列的顺序关系，而句子中词语的语序往往又是非常重要的，这也是NLP 和数字图像处理的一个区别所在，而在数图领域诞生的Attention 机制没有很好地驾驭这一方面。

Transformer

Transformer 模型的核心就是**串并联堆叠**和**self-attention** 机制。它由编码器（encoder）和解码器（decoder）两大部分组成，每个部分当中都由6个编（解）码器堆叠而成。这个模型的创新之处在于抛弃了RNN 和CNN 模型，不用循环和卷积过程，只基于attention 机制，采用**以空间换时序**的思路，通过简单粗暴的模型堆叠达到了很好的效果。也就是说Transformer 就是将多个encoder 串联后和多个串联的decoder 并联，实现了并行运算。

Transformer 引入Self Attention 后会更容易捕获长句子中单词间的特征，因为如果是RNN ，需要依次序序列计算，对于远距离的两个单词之间的关联特征，要经过若干步骤的信息累积才能将两者联系起来，而距离越远，有效捕获的可能性越小。但是**Self Attention 在计算过程中会直接将句子中任意两个单词的联系通过一个计算步骤直接联系起来，这就相当于每个单词之间的距离都缩短为1，所以远距离依赖特征之间的距离被极大缩短**，有利于有效地利用这些特征。

在编码过程中，对于句子中的每个单词，首先先用算法把单词**向量化**，在其中嵌入这个单词在句子中的**位置信息**（很关键，因为在自然语言处理领域句子的语序很重要，这也在一定程度上缓解了attention 的顽疾），再通过一系列的矩阵操作嵌入其他单词的权重信息，得到self-attention 值，最后通过前馈神经网络将其转换为结果Z 向量（先升维4倍后再降维，最后结果维度和原单词向量维度相同）。在解码过程中，不仅要考虑self-attention 值，还要考虑encoder-decoder attention值，也就是说

在接码时也要考虑编码过程中各个单词的权重。

最后，因为解码器输出的是一个矩阵，Transformer 会通过线性层将输出矩阵转换为一个一维向量（其中包含所有可能的单词），再通过softmax层找出其中可能性最大的，成为最后结果。

总的来说，Transformer 就是将词语的context 信息和位置信息嵌入词向量（Embedding 过程），将自然语言信息数字化后输入模型运算，得到最后结果。

BERT

BERT 是由Transformer 演化而来的。由于Transformer 的堆叠思路使得它具有良好的性能，人们试着将多个Transformer 的编码器（encoder）进行组合，以达到更好效果，有点类似俄罗斯套娃。因为就像人类聚集讨论后产生智慧一样，数个编码器进行组合、交流可能也会产生让人意想不到的效果。在预训练阶段，BERT 读取海量未标注的句子对进行无监督学习，这些句子中有一些缺失单词，而BERT 将预测这些缺失的单词，最后将结果和真实值进行对比。在预训练阶段，句子中的词被转换为embeddings 向量，它是由Token embeddings（词向量）、Segment embeddings（识别句子的向量）、**Position embeddings（位置编码）**相加而成的，**这使得BERT 能同时识别词语和句子两种特征**。

BERT 还可以根据不同任务进行**微调**。例如对于本次的SQuAD 数据集，BERT 可以将输入的无监督文本改为“问题—答案对”，通过模型预测出答案，再与真实答案进行比较，达到有针对性训练网络的目的。

下一步工作

- 受机器配置影响，实验受到阻力，下一步将租用GPU 进行实验，提高效率。
- 调整BERT 模型参数，比较模型运行过程中Train loss 和Train accuracy值的振荡情况，且对最终得到的F1 值和EM 值做比较。