BERT 环境配置与实验

全未祺

2023.11.13

对动手实践利用机器学习方法分析大规模数据有进一步了解,并学习如何利用远程环境进行工程代码的调试。

1 BERT 环境配置

1.1 服务器环境配置

a 使用 SSH 连接远程服务器

b 从 Anaconda 的官网下载 for Linux 的安装包, 用 FTP 传输至服务器上想要的安装位置, 并使用 bash 命令在该位置进行安装

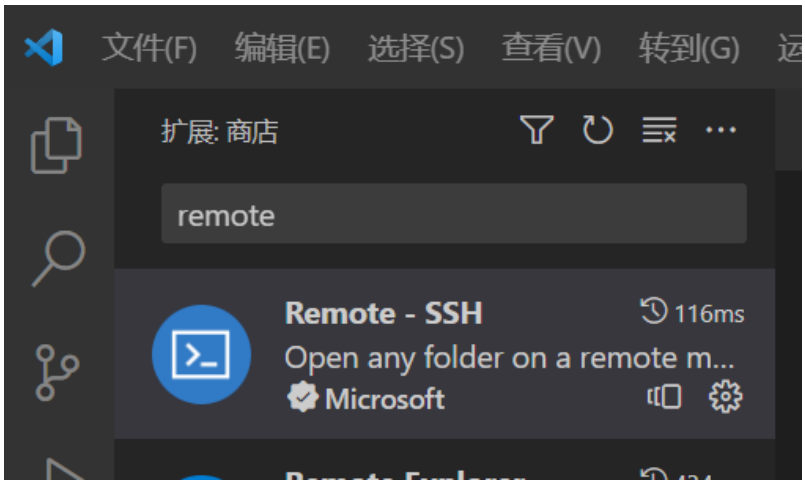
c 使用 conda create 命令创建 base 环境以外的虚拟环境

d 使用 conda activate 命令进行虚拟环境的切换

e 安装所需要的包

1.2 VSC 连接服务器

a 下载 Remote-SSH



b 添加服务器连接配置

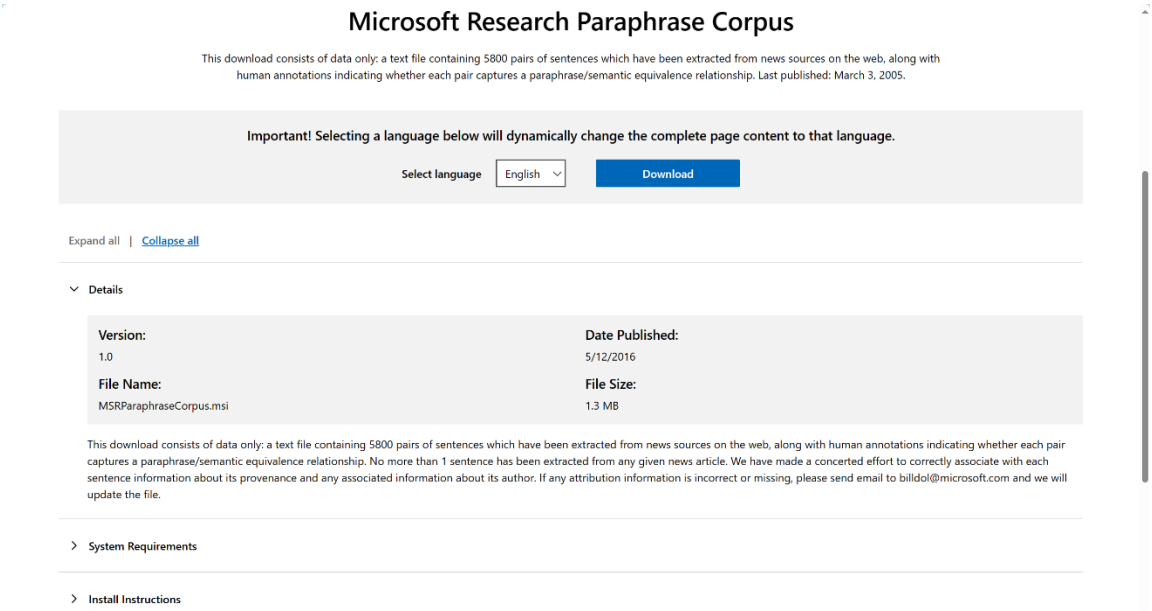
2 BERT 实践

熟悉 PyTorch 框架下，利用预训练的 transformers 的预训练 BERT 模型对 MRPC 数据集进行同义预测的 pipeline. 尝试理解数据是如何预处理，模型是怎么读入数据,是如何进行推理，如何进行评价的。

2.1 数据集

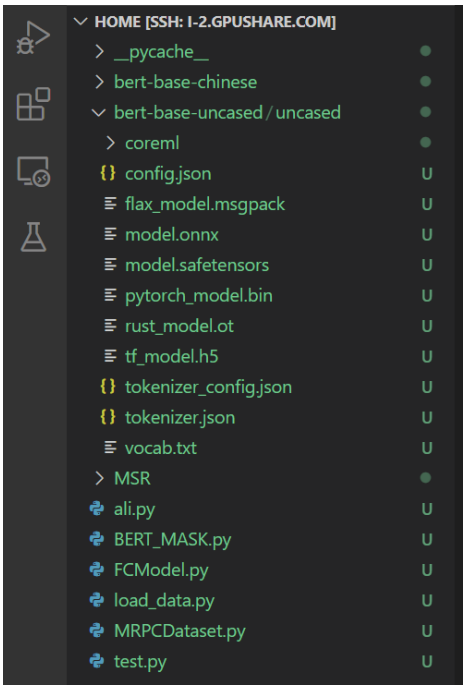
MRPC（Microsoft Research Paraphrase Corpus）包含了 5800 个句子对，有的是同义的，有的是不同义的，是否同义由一个二元标签进行描述。

下载链接 https://www.microsoft.com/en-us/download/details.aspx?id=52398



2.2 模型下载

可以在阿里云的相关镜像上下载模型，及其配置，并上传到服务器上。



2.3 代码逻辑

对 BERT 进行微调，每个句子对用 BERT 指定分隔符［SEP］连接后,通过 BERT 得到合成句子的 representation. 再通过通过一个两层的多层感知机得到分类结果。这里预训练 BERT 模型使用的是 HuggingFace 的 BERT-base-uncased。

2.4 模型运行及其结果

可以调整优化器的相关参数。

```
#定义优化器&损失函数
# optimizer = torch.optim.ASGD(model.parameters(),lr=0.001)
optimizer =torch.optim.Adam(model.parameters(),lr=0.02)
bert_optimizer=torch.optim.Adam(bert_model.parameters(),lr=0.001)
crit =torch.nn.BCELoss()
```

运行结果，同时输出损失和准确率。

```
(base) root@I16279d6f2202801760a:/home# conda activate base
(base) root@I16279d6f2202801760a:/home#  /usr/bin/env /usr/local/miniconda3/bin/python /root/.vscode-server/extensions/ms-python.python-2023.20.0/pytho
nFiles/lib/python/debugpy/adapter/../../debugpy/launcher 55937 -- /home/BERT_MASK.py
数据载入完成
设备配置完成
bert 层模型创建完成
全连接层模型创建完成
Asking to truncate to max_length but no maximum length is provided and the model has no predefined maximum length. Default to no truncation.
batch 0 loss:0.788278 accuracy:0.375000
batch 1 loss:3.280364 accuracy:0.437500
batch 2 loss:3.502980 accuracy:0.562500
batch 3 loss:0.812215 accuracy:0.312500
batch 4 loss:0.650065 accuracy:0.562500
batch 5 loss:0.967223 accuracy:0.250000
batch 6 loss:2.904629 accuracy:0.625000
batch 7 loss:2.412966 accuracy:0.625000
batch 8 loss:1.550910 accuracy:0.562500
batch 9 loss:0.934825 accuracy:0.562500
batch 10 loss:1.390372 accuracy:0.500000
batch 11 loss:0.965017 accuracy:0.375000
batch 12 loss:0.912256 accuracy:0.562500
batch 13 loss:0.502468 accuracy:0.812500
batch 14 loss:0.857591 accuracy:0.625000
batch 15 loss:0.424309 accuracy:0.875000
batch 16 loss:0.507727 accuracy:0.812500
batch 17 loss:0.566039 accuracy:0.750000
batch 18 loss:0.848211 accuracy:0.562500
batch 19 loss:0.626216 accuracy:0.750000
batch 20 loss:0.676265 accuracy:0.625000
batch 21 loss:0.672632 accuracy:0.625000
batch 22 loss:0.837454 accuracy:0.437500
batch 23 loss:0.725416 accuracy:0.312500
batch 24 loss:0.621320 accuracy:0.687500
batch 25 loss:0.896715 accuracy:0.500000
batch 26 loss:0.557282 accuracy:0.750000
batch 27 loss:0.822771 accuracy:0.312500
batch 28 loss:1.016589 accuracy:0.375000
batch 29 loss:1.045303 accuracy:0.125000
batch 30 loss:0.390892 accuracy:0.875000
batch 31 loss:1.379304 accuracy:0.625000
batch 32 loss:2.112521 accuracy:0.500000
batch 33 loss:1.080678 accuracy:0.687500
batch 34 loss:0.839892 accuracy:0.625000
batch 35 loss:0.679079 accuracy:0.750000
batch 36 loss:0.882959 accuracy:0.312500
batch 37 loss:0.928953 accuracy:0.187500
batch 38 loss:0.536666 accuracy:0.875000
batch 39 loss:0.622850 accuracy:0.750000
batch 40 loss:0.745133 accuracy:0.750000
batch 41 loss:0.814808 accuracy:0.750000
```

观察结果，可知准确率维持在一个不错的范围内，但个别情况仍然需要改进，模型整体比较优秀。