NER-Project

1. 问题描述

给定一个中文文本作为输入，期望模型扫描中文文本的每一个词，识别出其中的实体（例如人物，地点，概念，时间等等），这是一个实体识别（NER）任务。

2. CRF模型

2.1 介绍

CRF全称为条件随机场（CRF, Conditional Random Fields）模型，是一种用于标注和分割序列数据的概率图模型。它属于判别式模型，即直接建模给定输入序列时输出标注序列的条件概率，而不建模输入的整体分布。

简单来说，CRF通过考虑前后标签之间的依赖关系，来给每个位置分配最合理的标签。

2.2 模型训练与推理

简要介绍

训练阶段：

- 给定训练集 $(X^{(i)}, Y^{(i)})$，最大化条件概率：

$$\max \sum_i \log P(Y^{(i)}|X^{(i)})$$

- 可以使用 **梯度下降** 或 **L-BFGS** 优化特征权重 λ_k。

推理阶段：

- 给定输入序列 X，寻找最可能的标签序列：

$$\hat{Y} = \arg\max_Y P(Y|X)$$

- 通常使用 **维特比算法**（Viterbi）求解最优序列。

代码下载

使用老师提供的压缩包中的代码。

如何训练

把训练的文本放进crf文件夹内，假设路径为./train.txt，然后在crf文件夹内运行命令行终端，输入以下命令：

```
训练的命令
1   crf_learn -p 8 template ./train.txt model
```

如何测评

（1）获取批量输出

把测试的文本放进crf文件夹内，假设路径为./test.txt，然后在crf文件夹内运行命令行终端，输入以下命令：

```
获取批量输出
1   crf_test -m model ./test.txt >output.txt
```

运行后会得到批量输出文件output.txt。

（2） 计算token级别的分数和entity级别的分数

与下面第三章的方法一样，得到批量输出文件后运行对应的脚本（需要修改输入文件的路径）即可计算token级别的分数和entity级别的分数。

3. BERT模型

3.1 介绍

3.1.1 BERT-Tiny

使用https://huggingface.co/ckiplab/bert-tiny-chinese-ner的开源模型，该模型是由CKIP Lab基于 BERT Tiny Chinese 微调而来，支持繁体中文。

- Bert模型基础信息：12M参数量，4层、隐藏层维度大小为312；
- 额外的分类层：312维度 -> 73维度，将token分类成73个实体标签；

3.1.2 BERT-Base

使用https://huggingface.co/ckiplab/bert-base-chinese-ner的开源模型，该模型由 CKIP Lab 基于 BERT Base Chinese 微调而来，支持繁体中文。

- Bert模型基础信息：110M参数量，12层、隐藏层维度大小为768；
- 额外的分类层：768维度 -> 73维度，将token分类成73个实体标签；

3.2 模型训练与推理

简要介绍

由于我们的数据集只有17类实体的标签，而原模型的分类头是分成73类，因此需要修改原模型的分类头，重新初始化参数，根据我们的数据集做微调训练。

代码下载

代码仓库地址：https://github.com/Wenz0101/NER-Zh，该代码使用pytorch进行训练，其中不包括训练的模型（需要额外下载），包括了训练的数据集。

依赖的python包都放在了requirements.txt文件中，为了下载所有的依赖库，运行：

```
下载项目依赖的python包
1   pip install -r requirements.txt
```

数据集下载

上述代码仓库已经包括了需要用到的数据集（其中包括train，validation和test），位于/dataset文件夹下，该数据集的地址：https://github.com/lancopku/Chinese-Literature-NER-RE-Dataset，是关于中国文学的数据集。

模型下载

主要使用两个原始模型（bert-tiny-chinese-ner和bert-base-chinese-ner）进行微调训练，由于大小限制，不包括在代码仓库中，其中原始模型下载后需要放在/raw_model文件夹下，下载地址：

- bert-tiny-chinese-ner：https://huggingface.co/ckiplab/bert-tiny-chinese-ner
- bert-base-chinese-ner：https://huggingface.co/ckiplab/bert-base-chinese-ner

文件说明

- data.py：用于对数据集做预处理，提取数据集
- train.py：
- batch-output.py：
- single-output.py：
- entity-eval.py：
- dataset/：
- raw_model/：
- trained_model/：

如何训练

确保数据集已经在dataset文件夹下，确保原始模型已经在raw_model文件夹下，在确保命令行在代码仓库的源路径下（即.../NER-Zh），运行如下指令：

- 对于bert-tiny-chinese-ner这个较小的模型，以训练1个epoch，batch_size=4，学习率=1e-5为例，其中save_path是模型训练完之后模型保存的路径，可以自己修改。

基于bert-tiny-chinese-ner进行训练

```
python train.py --train_dataset ./dataset/train.txt --model_path ./raw_model/bert-tiny-chinese-ner --save_path ./trained_model/bert-tiny-train-test --epochs 1 --batch_size 4 --lr 1e-5
```

- 对于bert-base-chinese-ner这个较大的模型，以训练1个epoch，batch_size=1，学习率=1e-5为例，其中save_path是模型训练完之后模型保存的路径，可以自己修改。

基于bert-base-chinese-ner进行训练

```
python train.py --train_dataset ./dataset/train.txt --model_path ./raw_model/bert-base-chinese-ner --save_path ./trained_model/bert-base-train-test --epochs 1 --batch_size 1 --lr 1e-5
```

注：如果训练过久，可以适当减少训练集的条目数量。

下载训练好的模型

由于本地资源受限，用服务器资源对两种bert的基础模型做了微调训练，具体性能见第四章，下载地址为：https://huggingface.co/wenz20101/bert-chinese-ner/tree/main，将下载好的模型放在trained_model文件夹中。

如何测评

（1）获取模型的批量输出

确保数据集已经在dataset文件夹下，确保训练好的模型已经在trained_model文件夹下，在确保命令行在代码仓库的源路径下（即.../NER-Zh）；

假设训练好的模型路径为：./trained_model/bert-tiny-test，输出的文件路径为./output/bert-tiny-output.txt，运行如下命令即可得到指定模型的批量输出：

获取模型批量输出

```
python batch-output.py --model_path ./trained_model/bert-tiny-test --test_file ./dataset/test.txt --output_file ./output/bert-tiny-output.txt
```

（2）根据模型输出计算token级别的分数

确保已经在output文件夹下生成了模型的批量输出，假设批量输出文件路径为：/output/bert-tiny-output.txt，运行如下命令即可得到token级别的分数：

```
计算token级别的分数
python token-eval.py --file ./output/bert-tiny-output.txt
```

（3）根据模型输出计算entity级别的分数

确保已经在output文件夹下生成了模型的批量输出，假设批量输出文件路径为：/output/bert-tiny-output.txt，运行如下命令即可得到entity级别的分数：

```
计算entity级别的分数
python entity-eval.py --file ./output/bert-tiny-output.txt
```

如何推理

确保数据集已经在dataset文件夹下，确保训练好的模型已经在trained_model文件夹下，在确保命令行在代码仓库的源路径下（即.../NER-Zh），确保已经把需要分析的文本写入指定的文件（例如./dataset/input.txt）；

假设训练好的模型路径为：./trained_model/bert-tiny-test，输入的文件路径（即要分析的文本）为./dataset/input.txt，输出的文件路径为./output/test.txt，运行如下命令即可得到指定模型的批量输出：

```
单一文本的推理
python single-output.py --model_path ./trained_model/bert-tiny-test --input_file ./dataset/input.txt --output_file ./output/output.txt
```

4. 结果对比

4.1 训练与推理耗时

- CRF 模型的训练环境（使用CPU）：Intel(R) Core(TM) i5-9300H CPU @ 2.40GHz（8G）
- bert-tiny 和 bert-base 模型的微调环境（使用GPU）：NVIDIA GeForce RTX 4090（24G）
- 训练集数量：约4800句，共100万行；

模型	CRF	bert-tiny	bert-base

训练耗时	约15min	约15min	约1h
推理速度（token/s）	约9407	约14940	约3715

4.2 token分类分数

指标 \ 模型	CRF	bert-tiny	bert-base
Accuracy	0.8526	0.8698	0.8939
Precision	0.5619	0.5648	0.6299
Recall	0.4883	0.5248	0.6024
F1-score	0.5156	0.5420	0.6011

4.3 实体分类分数

指标 \ 模型	CRF	bert-tiny	bert-base
Accuracy	0.8526	0.8698	0.8939
Precision	0.6926	0.6037	0.6908
Recall	0.6049	0.6370	0.7242
F1-score	0.6458	0.6199	0.7071