## 模型训练流程

1. 数据预处理 python ./inputs/data\_process.py
2. 预训练词向量生成

（1）将官方提供的语料放在 ./inputs/corpus/

（2）python ./pre\_training/EmbeddingPretrain.py

这一步分别会在./inputs/embedding\_matrix 以及 ./ELMo 生成对应的预训练文件

其中/inputs/embedding\_matrix预训练词向量压缩后11G,我们已经上传到百度云盘: https://pan.baidu.com/s/1WKMIQAHmbNjTX7tcd0m\_Og

解压压缩文件 tar -xf embedding\_matrix.tar ，替换文件夹./inputs/embedding\_matrix

1. 模型训练与预测
2. python train\_elmo\_bilstm\_cnn.py
3. python train\_elmo\_bilstm\_selfatt.py
4. python train\_elmo\_gru\_locaatt.py
5. 预测数据后处理生成结果python ./outputs/result\_process.py ->./outputs/result.txt

### 根据训练好的模型生成结果

1. python pred\_result.py

完整的模型压缩之后26G，我们上传到百度云盘:

解压之后，替换文件夹./model

1. python ./outputs/result\_process.py -> ./outputs/result.txt

## 项目依赖

Python 3+, tensorflow 1.10+ keras 2.2.4+

我们将程序所跑环境打包成了requirements.txt

Conda环境安装 conda install –yes –file requirements.txt

## 引用的外部开源代码

使用了allennlp 开源的ELMo预训练代码：<https://github.com/allenai/bilm-tf>

使用了开源的Glove预训练词向量训练工具: https://github.com/maciejkula/glove-python

使用了Gensim作为预训练工具，训练了bichar，char级别的word2vec,fasttext预训练词向量: <https://github.com/RaRe-Technologies/gensim>

使用了Zhao HG开源的self-attention代码 ：<https://github.com/CyberZHG/keras-self-attention>

## 思路算法：

### 预训练

根据提供的语料分别在char级别训练了word2vec,fasttext,glove,ELMo以及bichar级别的word2vec,fasttext,glove。每一种词向量都训练了150, 200,250,300dim三种维度。

### 数据增强

随机选取训练集中8000条样本，对其实体进行同类别实体替换作为数据增强的数据。

### 模型与训练

一共使用三种模型，所有的模型的embedding层具有同样的结构，为char级别的word2vec, fasttext, glove ,ELMo 以及bichar级别的word2vec, fasttext, glove, ELMo拼接。下游结构分别为 bilstm+self-attenion+crf ， bilstm\_cnn(kernel\_size:3,5,7)+crf，GRU+local\_attention+crf.

每一个模型都采用五折交叉验证，采用早停的方式确定F1最好的模型，进行预测集预测。

### 模型集成与后处理

每一种模型分别在各个维度的embedding层上进行训练，最终得到的每一个样本的序列标签，采用加权投票的方式决定出对应的预测标签，其中每一个模型的权重来自五折交叉验证的F1做的softmax归一，最终得到生成结果。