“自然语言处理”实验报告

实验 2：命名实体识别

姓名：石翔宇　　学号：1190200523

Email：xyu.shi@hit.edu.cn

1. 实验概述

本次实验学习使用 HMM、ME、CRF 和深度学习等不同的命名实体识别方法，并在两个不同的数据集上进行实验。实验需要用到百度 AI Studio 平台和华为云计算平台。

2. 实验目标

- 通过不同方法的结果对比，掌握不同实体识别方法的优缺点。
- 通过对不同数据集的使用，掌握命名实体识别需要的数据预处理、模型训练、模型测试和评价方法。
- 通过对百度 AI Studio 平台和华为云平台的使用，了解国产主流人工智能平台提供的学习资源和计算资源，并能使用这些平台完成特定的自然语言处理任务。
- 通过实验报告的撰写，提高分析能力、写作能力和表达能力。

3. 命名实体识别的评价

命名实体识别的评价指标及计算公式

本次实验学习使用 HMM、ME、CRF 和深度学习等不同的命名实体识别方法，并在两个不用的数据集上进行实验。实验需要用到百度 AI Studio 平台和华为云计算平台。

命名实体识别的评价指标包括准确率、召回率和 F1 值。评价指标的计算公式如下：

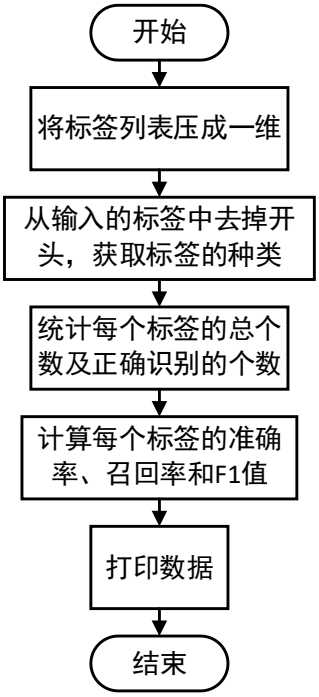
$$\text{准确率} = \frac{\text{识别出正确的实体数}}{\text{识别出的实体数}}$$

$$\text{召回率} = \frac{\text{识别出正确的实体数}}{\text{样本的实体数}}$$

$$\text{F1 值} = \frac{2 \times \text{准确率} \times \text{召回率}}{\text{准确率} + \text{召回率}}$$

实体级命名实体识别的评价程序的实现思路

实体级命名实体识别的评价程序实现的总体思路如下图所示：



“统计每个标签的总个数”的实现具体思路如下：

对每个合法的当前标签的区间，记录区间的开始和结束位置。区间的个数即为标签的个数。将预测结果的区间与标准答案的区间取交得到新的区间集合，集合中元素的个数即为正确识别的个数。

获取当前标签的每个合法区间的程序伪代码如下：

```
def to_region(tags_line, tag):
    if tag == 'O':
        return set([(p, p) for p in range(len(tags_line)) if tags_line[p] == 'O'])
    B, M, E, S = 'B-' + tag, 'M-' + tag, 'E-' + tag, 'S-' + tag

    point, begin_p, region = 0, None, set()
    while point < len(tags_line):
        if begin_p == None:
            if tags_line[point] == B:
                begin_p = point
            elif tags_line[point] == S:
                begin_p = point + 1
                region.add((point, point))
        else:
            if tags_line[point] == E:
                region.add((begin_p, point))
                begin_p = None
            elif tags_line[point] == B:
                begin_p = point
            elif tags_line[point] != M:
                begin_p = None
            elif tags_line[point] == S:
                begin_p = point + 1
                region.add((point, point))
        point += 1
    return region
```

对于每个标签的准确率、召回率和 F1 值的计算，使用上面的公式即可。

评价程序的调用方法

评价程序为 class Metrics_object(object)，参数包括 golden_tags 和 predict_tags，其中 golden_tags 标准答案，predict_tags 为预测的输出。外部能够访问的方法有 report_scores()，用于输出评价结果。

调用方法如下：

```
metrics = Metrics_object(test_tag_lists_clue, pred_tag_lists)
metrics.report_scores()
```

评价 HMM 模型的实体识别系统的效果

对 HMM 模型在 ner_char_data 目录下的 train.txt 文件训练模型，在 test.txt 文件上进行测试。

测试结果按词级别评价结果如下表所示：

	precision	recall	f1-score	support
B-NAME	0.98	0.875	0.9245	112
M-NAME	0.9459	0.8537	0.8974	82
E-NAME	0.9	0.8036	0.8491	112
O	0.9568	0.9177	0.9369	5190
B-PRO	0.5581	0.7273	0.6316	33
E-PRO	0.6512	0.8485	0.7368	33
B-EDU	0.9	0.9643	0.931	112
E-EDU	0.9167	0.9821	0.9483	112
B-TITLE	0.8811	0.8925	0.8867	772
M-TITLE	0.9038	0.8751	0.8892	1922
E-TITLE	0.9514	0.9637	0.9575	772
B-ORG	0.8422	0.8879	0.8644	553
M-ORG	0.9002	0.9327	0.9162	4325
E-ORG	0.8262	0.868	0.8466	553
B-CONT	0.9655	1	0.9825	28
M-CONT	0.9815	1	0.9907	53
E-CONT	0.9655	1	0.9825	28
M-EDU	0.9348	0.9609	0.9477	179
B-RACE	1	0.9286	0.963	14
E-RACE	1	0.9286	0.963	14
B-LOC	0.3333	0.3333	0.3333	6
M-LOC	0.5833	0.3333	0.4242	21
E-LOC	0.5	0.5	0.5	6

M-PRO	0.449	0.6471	0.5301	68
avg/total	0.9149	0.9122	0.913	15100

测试结果按实体级别评价结果如下表所示：

	precision	**recall**	**f1-score**	**support**
NAME	0.8491	0.8036	0.8257	112
PRO	0.5581	0.7273	0.6316	33
EDU	0.8917	0.9554	0.9224	112
TITLE	0.8747	0.886	0.8803	772
ORG	0.7543	0.7884	0.771	553
CONT	0.9655	1	0.9825	28
RACE	0.8667	0.9286	0.8966	14
LOC	0.3333	0.3333	0.3333	6
avg/total	0.8263	0.8491	0.8372	1630

词级别评价的细粒度更高，使得当整个实体未被正确识别，但单个实体中部分词被正确识别时，部分被正确识别词被计入正确的个数。而实体级别评价时需要整个实体中每个词都被正确识别才计入正确的个数。这使得词级别评价结果相比于实体级别评价结果较高。

4.基于最大熵模型的实体识别

最大熵模型原理的简单介绍

最大熵模型假设分类模型是一个条件概率分布$P(Y|X)$，其中X为特征，Y为输出。给定一个训练集$(x^{(1)},y^{(1)}),(x^{(2)},y^{(2)}),\dots,(x^{(m)},y^{(m)})$，其中$x$为$n$维特征向量，$y$为类别输出。我们的目标就是用最大熵模型选择一个最好的分类类型。

在给定训练集的情况下，我们可以得到总体联合分布$P(X,Y)$的经验分布$\overline{P}(X,Y)$，和边缘分布$P(X)$的经验分布$\overline{P}(X)$。$\overline{P}(X,Y)$即为训练集中X,Y同时出现的次数除以样本总数m, $\overline{P}(X)$即为训练集中X出现的次数除以样本总数m。

用特征函数$f(x,y)$描述输入x和输出y之间的关系。定义为：

$$f(x,y)=\begin{cases}1, & x与y满足某个关系\\ 0, & 否则\end{cases}$$

可以认为只要出现在训练集中出现的$(x^{(i)},y^{(i)})$， 其$f(x^{(i)},y^{(i)})=1$。 同一个训练样本可以有多个约束特征函数。

特征函数$f(x,y)$关于经验分布$\overline{P}(X,Y)$的期望值，用$E_{\overline{P}}(f)$表示为：

$$E_{\overline{P}}(f) = \sum_{x,y} \overline{P}(x,y) f(x,y)$$

如果模型可以从训练集中学习，我们就可以假设这两个期望相等。即：

$$E_{\overline{P}}(f) = E_P(f)$$

上式就是最大熵模型学习的约束条件，假如我们有M个特征函数$f_i(x,y)(i = 1,2 \dots, M)$就有$M$个约束条件。可以理解为我们如果训练集里有$m$个样本，就有和这$m$个样本对应的$M$个约束条件。

这样我们就得到了最大熵模型的定义如下：

假设满足所有约束条件的模型集合为

$$E_{\overline{P}}(f_i) = E_P(f_i)(i = 1,2, \dots M)$$

定义在条件概率分布$P(Y|X)$上的条件熵为

$$H(P) = -\sum_{x,y} \overline{P}(x) P(y \mid x) \log P(y \mid x)$$

我们的目标是得到使$H(P)$最大的时候对应的$P(y \mid x)$， 这里可以对$H(P)$加了个负号求极小值，这样做的目的是为了使$-H(P)$为凸函数，方便使用凸优化的方法来求极值。

自己实现的基于最大熵模型的实体识别系统描述

我们使用 sklearn 中的 LogisticRegression 来作为最大熵模型。我们首先需要将输入数据的 features 和 tags 压平，成为一维的 list。再利用 sklearn 中的 DictVectorizer 将 features 转为词向量，将转换好的词向量和 tags 调用 LogisticRegression.fit()训练模型。

测试时也需要将测试数据的 features 压平，再利用 DictVectorizer 将 features 转为词向量。将转换好的词向量调用 LogisticRegression.predict()获得模型对测试数据的输出。

基于最大熵模型的实体识别系统利用的特征

使用的特征包括：

- 前一个词
- 当前词
- 后一个词
- 前一个词+当前词
- 当前词+后一个词
- 前一个词+当前词+后一个词

评价自己实现的基于最大熵模型的实体识别系统的效果

利用 ner_char_data 目录下的 train.txt 文件训练模型，在 test.txt 文件上进行测试。

测试结果按词级别评价结果如下表所示：

	precision	**recall**	**f1-score**	**support**
B-NAME	0.8203	0.9375	0.875	112
M-NAME	0.9762	0.5	0.6613	82
E-NAME	0.9478	0.9732	0.9604	112
O	0.9606	0.9435	0.952	5190
B-PRO	0.85	0.5152	0.6415	33
E-PRO	0.775	0.9394	0.8493	33
B-EDU	0.9292	0.9375	0.9333	112
E-EDU	0.9908	0.9643	0.9774	112
B-TITLE	0.9119	0.8718	0.8914	772
M-TITLE	0.9069	0.8663	0.8861	1922
E-TITLE	0.9909	0.987	0.989	772
B-ORG	0.8893	0.8571	0.8729	553
M-ORG	0.8894	0.9466	0.9171	4325
E-ORG	0.9019	0.8807	0.8911	553
B-CONT	0.6	0.9643	0.7397	28
M-CONT	1	1	1	53
E-CONT	1	0.9643	0.9818	28
M-EDU	0.9556	0.9609	0.9582	179
B-RACE	1	0.9286	0.963	14
E-RACE	1	0.9286	0.963	14
B-LOC	0	0	0	6
M-LOC	0.6	0.1429	0.2308	21
E-LOC	1	0.8333	0.9091	6
M-PRO	0.5479	0.5882	0.5674	68
avg/total	0.9228	0.9219	0.9211	15100

测试结果按实体级别评价结果如下表所示：

	precision	**recall**	**f1-score**	**support**
NAME	1	0.5714	0.7273	112
PRO	1	0.4242	0.5957	33
EDU	0.9903	0.9107	0.9488	112
TITLE	0.9693	0.8174	0.8869	772

ORG	0.9308	0.6564	0.7699	553
CONT	1	0.9286	0.963	28
RACE	1	0.8571	0.9231	14
LOC	0	0	0	6
avg/total	0.9576	0.7436	0.8329	1630

5. HMM、ME 与 CRF 的效果对比

利用 ner_clue_data 目录下的数据训练并测试模型

首先读取数据文件，对于每一行都有三层循环，首先循环每个出现的标签，再循环标签对应的每个词，最后循环词出现的每个位置。

对于每个位置，若位置长度为 1，则直接将新标签设置为“'S' + tag”；若长度大于 1，则开始和末尾的位置的新标签分别设置为“'B' + tag”和“'E' + tag”，中间的位置的新标签则设置为“'M' + tag”。

对于每一行中没有出现的那些位置，则将其标签设置为“O”。

最后将数据存储成与 ner_char_data 相同的结构。对 train.txt 和 dev.txt 分别做上述操作。

与 ner_char_data 相同地，还需要将将每个 word 和 tag 转成数字 id 的形式。

我们可以运行下面的代码来获取 ner_clue_data 目录下的数据

```
1. train_word_lists_clue, train_tag_lists_clue, word2id_clue, tag2id_clue = data_build
   _clue(file_name="train.txt", make_vocab=True)
2. test_word_lists_clue, test_tag_lists_clue = data_build_clue(file_name="dev.txt", ma
   ke_vocab=False)
```

将 HMM、ME 和 CRF 模型的输入改为上述读取的数据，并用上述读取的数据进行测试即可。

HMM、ME 和 CRF 模型的评价效果

利用 ner_clue_data 目录下的 train.txt 文件训练模型，在 dev.txt 文件上进行测试。

HMM 模型的结果如下：

	precision	**recall**	**f1-score**	**support**
name	0.5802	0.6065	0.5931	465
address	0.3668	0.3727	0.3697	373
organization	0.5237	0.5422	0.5328	367
game	0.6906	0.7186	0.7043	295

scene	0.4439	0.4354	0.4396	209
book	0.6263	0.4026	0.4901	154
company	0.5124	0.5476	0.5294	378
position	0.6413	0.6605	0.6507	433
government	0.5052	0.5951	0.5465	247
movie	0.5449	0.6424	0.5897	151
avg/total	0.5437	0.5605	0.5502	3072

ME 模型的结果如下：

	precision	**recall**	**f1-score**	**support**
name	0.8739	0.4172	0.5648	465
address	0.6857	0.193	0.3013	373
organization	0.8908	0.5559	0.6846	367
game	0.8702	0.6136	0.7197	295
scene	0.7826	0.1722	0.2824	209
book	0.8372	0.2338	0.3655	154
company	0.8671	0.3624	0.5112	378
position	0.9046	0.5912	0.7151	433
government	0.9036	0.3036	0.4545	247
movie	0.9	0.2384	0.377	151
avg/total	0.8518	0.3994	0.5293	3072

CRF 模型的结果如下：

	precision	**recall**	**f1-score**	**support**
name	0.8082	0.7247	0.7642	465
address	0.5932	0.4692	0.524	373
organization	0.8062	0.7139	0.7572	367
game	0.8161	0.8271	0.8215	295
scene	0.6733	0.4833	0.5627	209
book	0.7869	0.6234	0.6957	154
company	0.7955	0.7407	0.7671	378
position	0.8191	0.7113	0.7614	433
government	0.7846	0.7814	0.783	247
movie	0.6818	0.6954	0.6885	151
avg/total	0.7642	0.6839	0.7203	3072

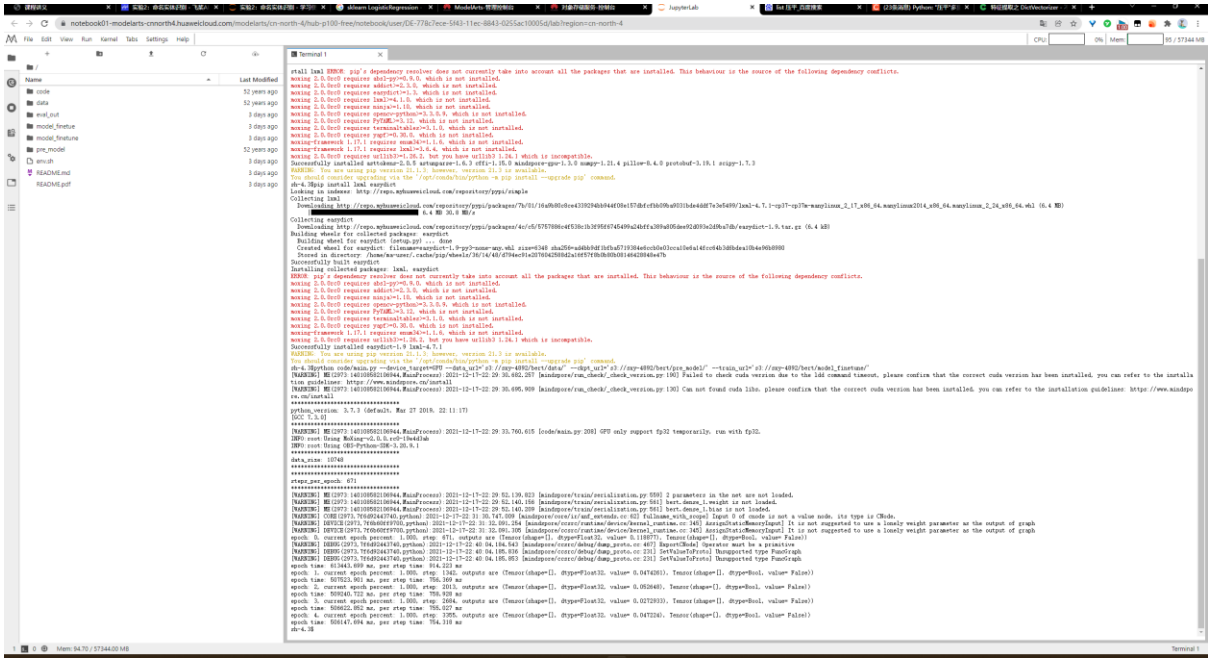
HMM 模型的假设前提在比较小的数据集上是合适的，但实际上在大量真实语料中观察序列更多的是以一种多重的交互特征形式表现，观察元素之间广泛存在长程相关性。在命名实体识别的任务中，由于实体本身结构所具有的复杂性，利用简单的特征函数往往无法涵盖所有的特性，这时 HMM 的假设前提使得它无法使用复杂特征，使得 HMM 效果不好。

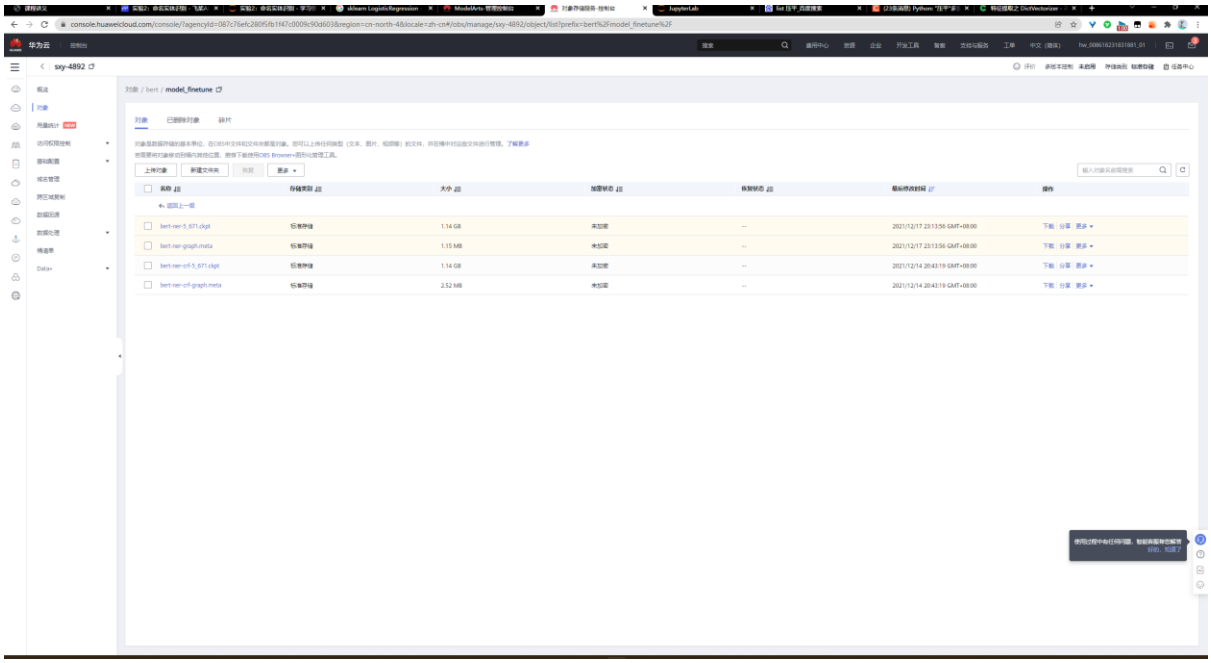
ME 模型可以使用任意的复杂相关特征，在性能上最大熵分类器超过了贝叶斯分类器，它获得的是所有满足约束条件的模型中信息熵极大的模型。这使得 ME 模型的效果要好于 HMM 模型。

CRF 模型是在给定需要标记的观察序列的条件下，计算整个标记序列的联合概率分布，而不是在给定当前状态条件下，定义下一个状态的状态分布。CRF 模型由于其自身在结合多种特征方面的优势和避免了标记偏置问题。但 CRF 需要训练的参数更多，与 ME 相比，它存在训练代价大、复杂度高的缺点，这使得虽然 CRF 模型理论上效果要好于 ME 模型，但实际结果并不理想。

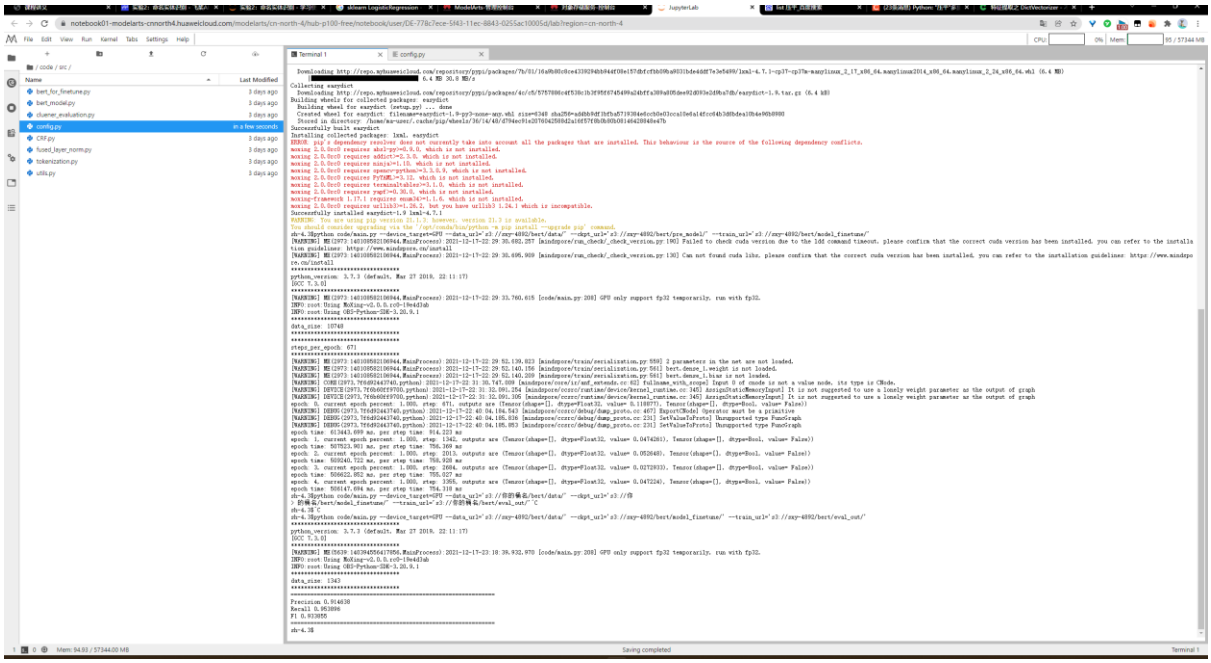
6. 在华为云上的计算资源进行命名实体识别

1. 启动微调训练（无 CRF）



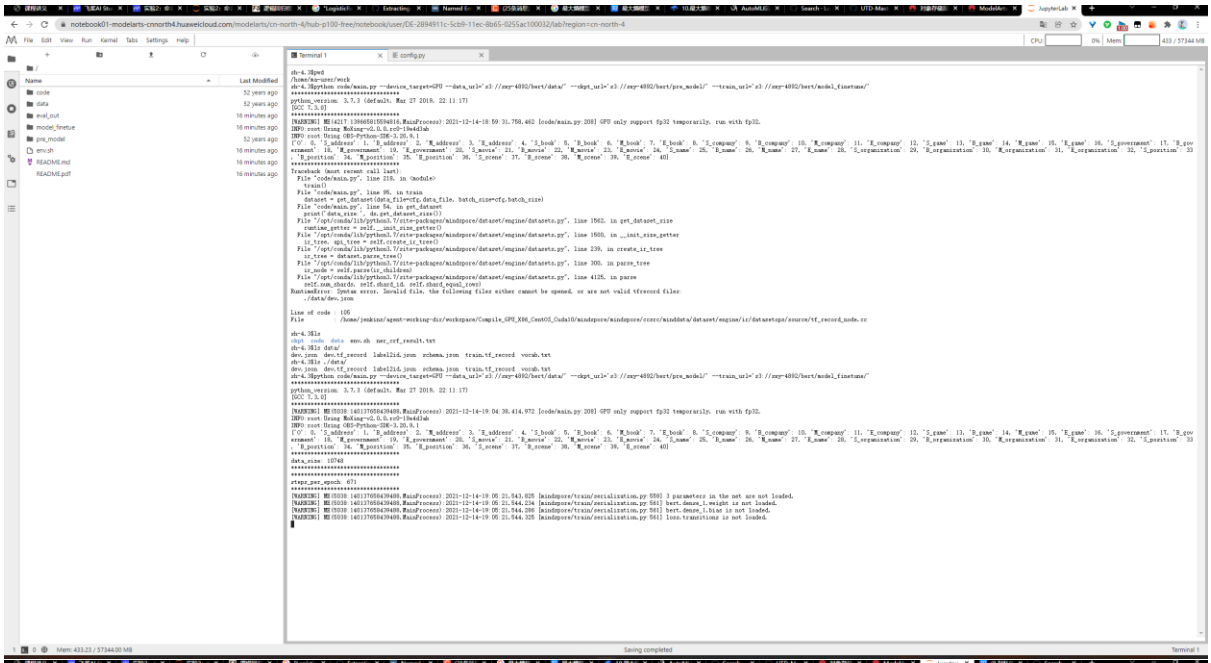


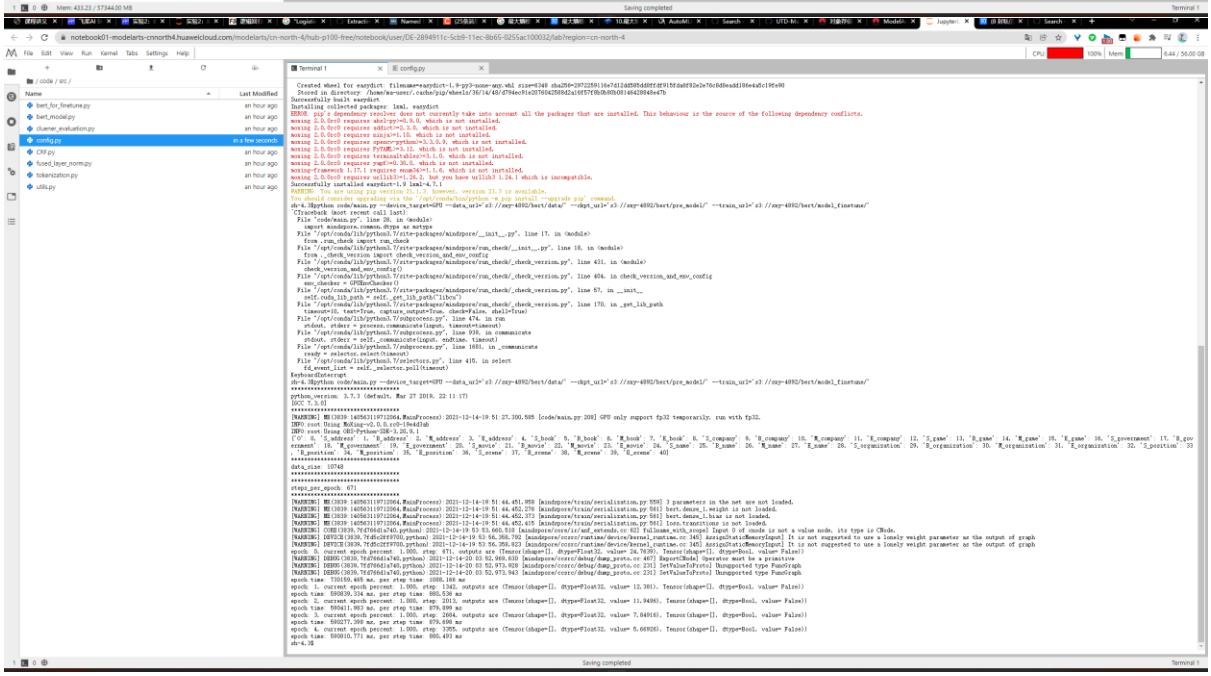
2. 测试评价指标

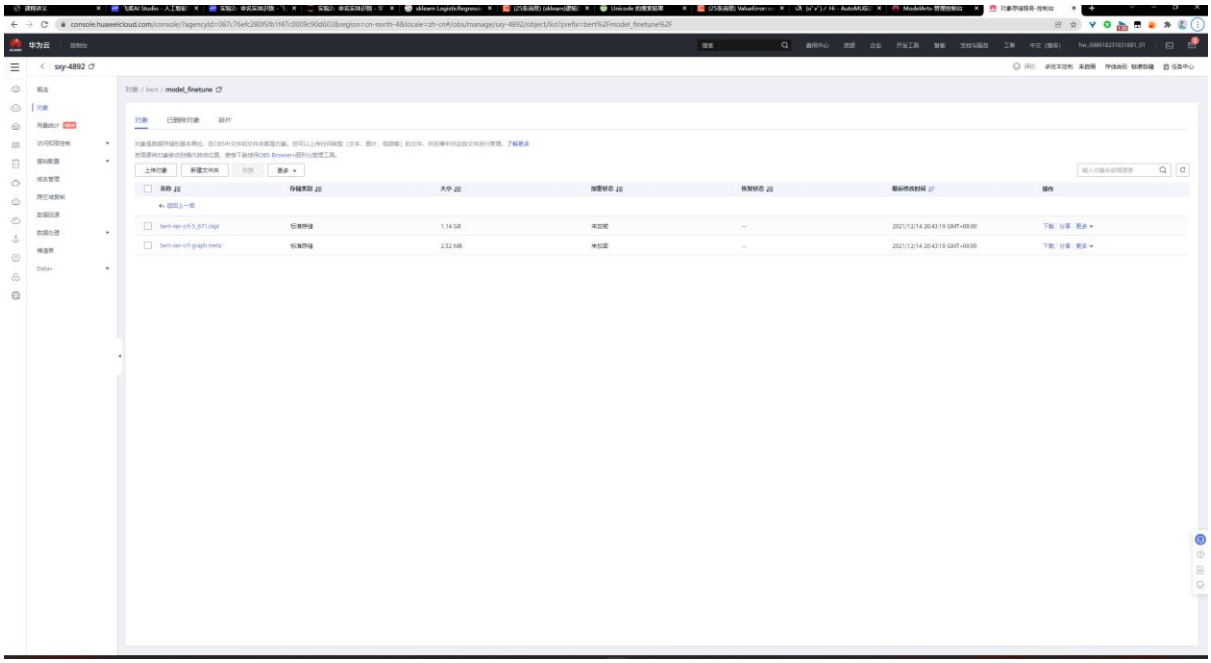


3. 启动微调训练（有 CRF）

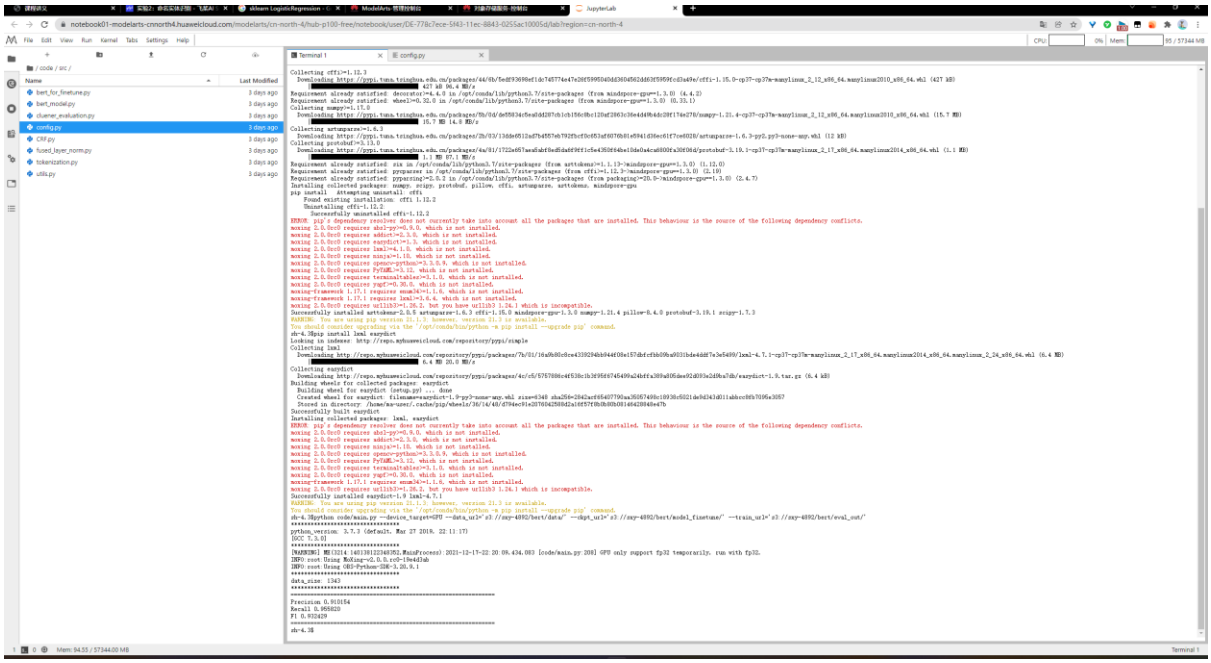
哈尔滨工业大学　　2021 学年秋季学期　　自然语言处理



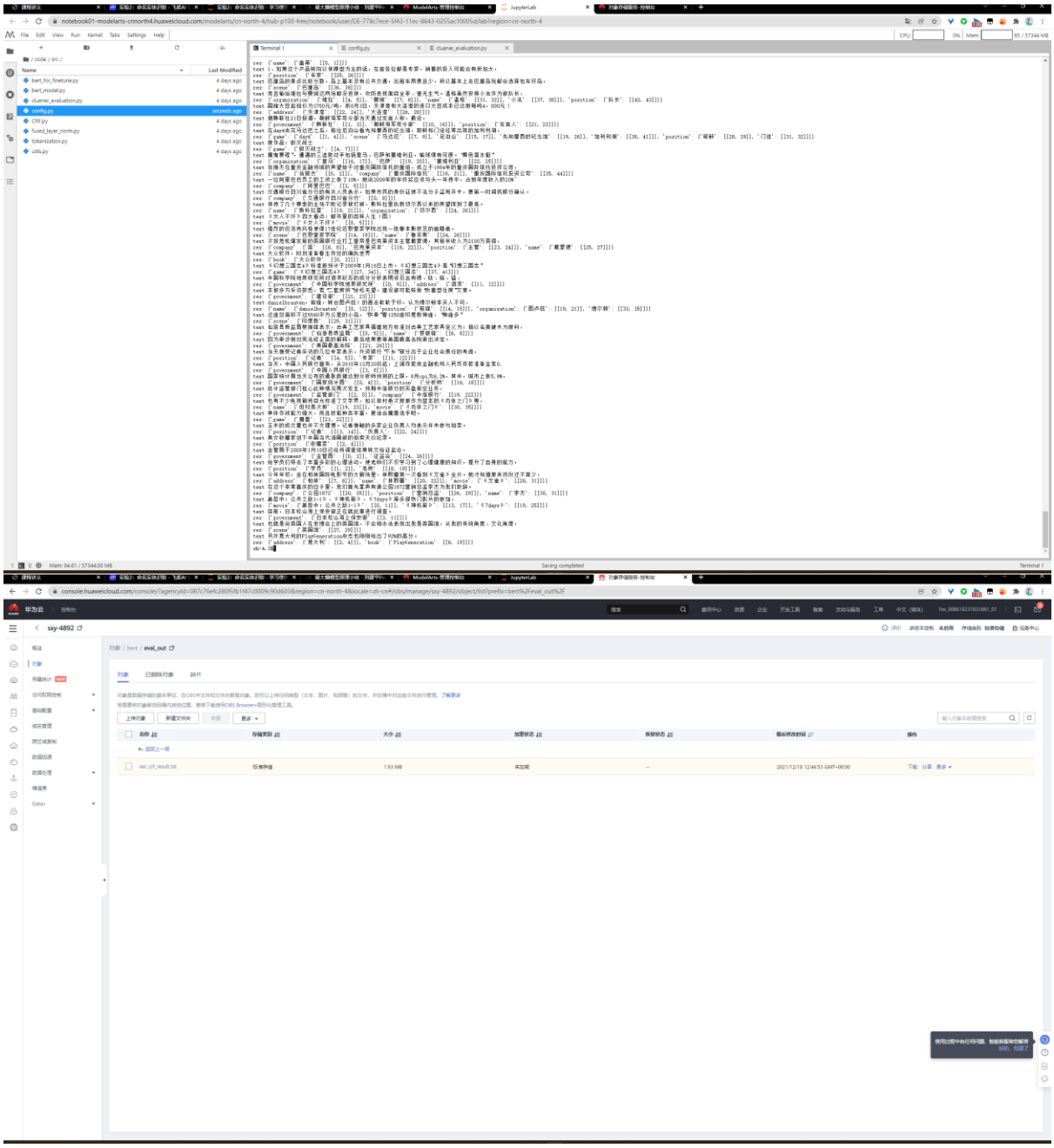




4. 测试评价指标



5. 测试结果



7. 实验的收获和体会

通过本次实验，我实现了 HMM、ME、CRF 等不同的命名实体识别方法，对各种模型的特点有了更加深刻的理解。与此同时，我实现了实体级别的评价程序，体会到了词级别与实体级别评价程序的差异。我还在两个不同的数据集上进行了实验，经历了数据预处理的任务过程，对命名实体识别任务从数据预处理、模型训练，到模型测试和评价方法有了全方位的训练体验。

参考文献

1. https://www.cnblogs.com/pinard/p/6093948.html
2. https://www.cnblogs.com/Eva-J/articles/7291842.html

3. https://www.cnblogs.com/549294286/archive/2013/06/06/3121761.html
4. https://blog.csdn.net/ccblogger/article/details/81843304
5. https://blog.csdn.net/houyanhua1/article/details/87895394
6. https://blog.csdn.net/firewall5788/article/details/78214631
7. https://blog.csdn.net/Jon_Sheng/article/details/79693971
8. https://www.pythonheidong.com/blog/article/1158746/a1884b4c4f86926ce3fa/
9. https://www.jianshu.com/p/21f83a1b33ae