《软件实践》课程实验报告

暑期学校实验项目：高考志愿填报助手

小组名称	知识图谱构建 A 组						
姓　　名	李春澍	专业	人工智能	班级	091181	学号	09118116
实验时间	2020.8.31-2020.9.23		指导教师	孔祥龙		成绩	

一、实验背景和目的

常言道，“考得好不如填得好，”高考志愿填报是高校录取中至关重要的一环，直接影响考生能否被录取上适合、理想的大学，甚至影响到考生的一生。然而，许多考生往往只顾考到高的分数，无暇研究高考志愿，只凭借道听途说或顾名思义等方式进行填报。这就产生了对高考志愿填报辅助系统的需求。

本课程的实验项目高考志愿填报助手，针对这一庞大的需求，旨在设计一个 Web 高考志愿推荐网站，根据历年高考数据以及用户数据（如高考成绩和排名），为用户提供志愿填报的合理建议。网站应该提供一个清晰友好的界面，并能够为考生提供查询和可视化功能，基于知识图谱和人工智能算法等技术进行高考志愿推荐，给出关于志愿填报的建议，帮助考生决定合理的高考志愿填报策略。

二、小组任务和个人任务

我所在的小组为“知识图谱构建 A 组”。本小组的任务为，以高校为中心构建包含高校、专业及其一级学科相关知识的知识图谱，从而使其他小组负责的知识推理算法成为可能。

在小组中，我的个人任务是和另外两位小组成员李浩天、庄祎共同构建各个专业到一级学科的映射关系。我们尝试 BERT 向量相似度和 Jaccord 相似度匹配两种方式。我的具体分工是，利用 BERT 模型计算各专业和一级学科的向量表示，从而根据向量相似度构建从专业到一级学科的映射词典。

三、个人任务需求分析

总体来说，高校中专业的设置是基于学科的。然而，对于同一学科，专业名称可能在不同高校各不相同，在同一高校也会有不同的班级；与此同时，近年来流行大类招生、宽口径培养，一个专业可能涉及多个学科。这种歧义性为知识图谱的构建带来了困难。

国家一级、二级学科目录提供了一个比较权威的标准学科命名，为消歧义带来了可能。不过，对于大部分情况，二级学科的划分过于细致，适用于研究生专业。对于本科专业而言，更加适合采用一级学科。我和李浩天、庄祎要做的就是建立起一个从专业名称到一级学科（可能有多个）的词典。

小组尝试使用多种方法构建这样的词典。由于涉及到语义的匹配，而现代预训练语言模型可以做到用一个向量来表示一个短语的语义。一个自然的想法就是利用语言模型计算学科和专业的向量表示，然后进行相似度匹配，即选择和专业相似度高的学科作为匹配。对于语言模型，我采用中文 BERT 模型。对于相似度，我选择使用余弦相似度而非欧几里得距离。

在得到匹配结果之后，与另一组李浩天进行的 Jaccard 相似度匹配结果进行比对，再经过人工修订得到最后的结果。人工修订由三人共同完成。

四、实验过程（需附上关键代码及相关说明）

1. 语言模型的选择和载入

BERT 模型是应用非常广泛成熟的语言模型，而且有着许多中文预训练后的版本。经过对比，我选择哈工大讯飞联合实验室（HFL）发布的中文预训练模型 BERT-wwm。不同于 Google 发布的原始版本 BERT 中，中文以字为粒度进行切分，BERT-wwm 在进行预训练时使用全词掩膜技术，即对组成同一个词的汉字全部进行掩膜。我选用这个模型的 ext 版本，使用中文维基和高达 5.4B 词数的通用语料库进行预训练。

在下载完这个模型的 PyTorch 版本后，我使用 huggingface transformers 库进行载入。

```
from transformers import BertModel, BertTokenizer

model_path = './chinese-bert-wwm-ext'
tokenizer = BertTokenizer.from_pretrained(model_path)
model = BertModel.from_pretrained(model_path, return_dict=False)
```

2. 载入专业名称和学科名称的词表

经过自己整理，我得到了一个 txt 格式的一级学科列表，使用 pandas.read_csv()进行读取，存为一个 pandas.Series。除此之外，对于二级学科列表也进行了类似处理，但由于需求变更，后续并没有用上。

```
subject1 = pd.read_csv("一级学科.txt")
print(subject1)

#%%

subject1 = subject1["一级学科"]
```

同理，使用 pandas.read_csv()读取第一组整理出来的专业 csv 表格，只用到其中的第 5 列，即专业名称。使用 unique()方法进行去重复。

```
majors = pd.read_csv("majors.csv")
majors = majors.iloc[:, 4]
majors = majors.unique()
print(majors)
```

3. 得到 BERT 向量表示

为了得到专业名称和学科名称的向量表示，先使用 BertTokenizer 对短语进行预处理，也就是将短语变成许多 token，替换为词汇表中的编号，并且加上所需的特殊 token，如[CLS]

和[SEP]，形成一个 tensor。然后将 tensor 输入到模型中得到输出。模型输出是最后一层 hidden state，是一个 tensor，选用其中属于[CLS]标签的那一维作为整个短语的向量表示。

```
subject1_reps = []
for subject in subject1:
    input_ids = torch.tensor([tokenizer.encode(subject, add_special_tokens=True)])
    with torch.no_grad():
        outputs = model(input_ids)
        vector = outputs[0][0][0].numpy()
        subject1_reps.append(vector)

subject1_reps = np.array(subject1_reps)
```

```
major_reps = []
for major in majors:
    input_ids = torch.tensor([tokenizer.encode(major, add_special_tokens=True)])
    with torch.no_grad():
        outputs = model(input_ids)
        vector = outputs[0][0][0].numpy()
        major_reps.append(vector)

major_reps = np.array(major_reps)
```

4. 得到从专业、学科到向量表示的词典

```
major_dict = dict(zip(majors, major_reps))
major_dict
```

```
subject1_dict = dict(zip(subject1, subject1_reps))
subject2_dict = dict(zip(subject2, subject2_reps))
```

得到向量表示的词典之后，将词典发给庄祎，由他进行相似度匹配，最终得到一个从专业名称到一级学科的词典 disamb_dict。

5. 人工比对和校订

得到词典后，将如此得到的词典和李浩天用 Jaccard 相似度匹配得到的词典进行比对。在三人之中分配任务，进行人工比对和校订，用人工改正错误的匹配结果，并且补充更全面的匹配结果。

五、实验结果与分析

1. 通过 BERT 得到一个短语的向量表示，以“我爱你”为例。

```
input_ids = torch.tensor([tokenizer.encode("我爱你", add_special_tokens=True)])
with torch.no_grad():
    outputs = model(input_ids)
    vector = outputs[0][0][0]
    print(vector)
    print(vector.shape)
```

```
tensor([ 3.3570e-01, -5.4012e-01, -1.9958e-01,  5.4008e-01,  9.2046e-01,
        -9.2604e-01, -1.5489e-01, -6.3178e-01, -8.8748e-01,  4.8947e-01,
        -3.5463e-01, -9.6442e-02, -2.6993e-01, -9.0201e-01,  5.5577e-01,
        -5.7829e-01,  1.1385e+00, -7.1449e-01, -8.6087e-01, -6.1813e-01,
         1.7531e-01,  3.2738e-01, -6.2459e-01, -3.7135e-01,  6.2036e-01,
        -3.4575e-01, -4.7123e-02,  1.5857e-01,  8.6580e-01,  1.2950e+00,
         2.2524e-01, -3.8432e-01, -3.8576e-01])
torch.Size([768])
```

得到一个 768 维的向量。可见，通过这种方式可以得到一个短语的向量表示。

2. 得到从专业名称、一级学科到向量表示的词典。这里的 all 是未细分的专业。

```
major_dict = dict(zip(majors, major_reps))
major_dict
```

```
{'all': array([-1.83136001e-01,  2.19828442e-01,  3.28633100e-01,  4.18238603e-02,
         3.72874469e-01, -1.15956366e+00,  4.56586361e-01, -3.88357639e-01,
        -5.45881987e-01,  3.85034949e-01, -2.83389837e-01,  2.61822194e-01,
         2.14988232e-01, -5.61588764e-01,  6.18504882e-01, -2.10637748e-01,
         4.04448241e-01, -5.31205297e-01, -3.59614134e-01, -1.37617677e-01,
```

```
with open('subject1_dict.pickle', 'rb') as subject1_f:
    subject1_dict = pickle.load(subject1_f)

subject1_dict
```

```
{'哲学': array([ 0.9999499 ,  0.9709993 ,  0.99999934,  0.9163759 ,  0.9337068 ,
         0.7637681 , -0.08838896, -0.7852978 ,  0.989223  , -0.9998451 ,
         0.99999964,  0.9999998 , -0.4823873 , -0.22836743,  0.99985397,
        -0.99999565,  0.698988  ,  0.68366665,  0.9896527 , -0.27733815,
         0.99553937, -0.99985945,  0.31961295, -0.9939031 ,  0.8252924 ,
```

```
print(len(subject1_dict))
print(len(major_dict))
```

```
89
2584
```

两个词典的键值对数量与专业名称数量、一级学科数量相一致。

3. 庄祎编写匹配算法，得到一个从专业名称到一级学科的词典。

```
with open('disamb_dict_bert_pooler_subject1.pickle', 'rb') as f:
    disamb_dict = pickle.load(f)
disamb_dict
```

```
{'all': '力学（可授工学、理学学位）',
 '外国语言文学类': '社会学',
 '财政学类': '公共管理',
 '法学': '法学',
 '工商管理类': '工商管理',
 '国际政治': '社会学',
 '金融学类': '理论经济学',
 '经济学类': '社会学',
 '理科试验班': '光学工程',
 '人力资源管理': '农业资源利用',
 '人文科学试验班': '光学工程',
 '社会科学试验班（管理学科类）': '心理学（可授教育学、理学学位）',
 '社会学类': '社会学',
```

可以发现，匹配结果：

a) 部分字面很接近的专业匹配得比较好，例如“工商管理类”匹配到“工商管理”、“电气类”匹配到“电气工程”；

b) 对于字面上不是很接近的专业，有的也有着较好的匹配结果，说明 BERT 模型确实可以学习到超越字面的语义，例如“金融学类”匹配到“理论经济学”；

c) 对于较长较复杂、带括号，或具有一些“单列”“定向”等词语修饰的专业，匹配结果不好。有的流于字面，例如“风景园林”匹配到“林业工程”；有的根本不相干，例如“数学类(含数学与应用数学(基地班)、信息与计算科学)”匹配到“环境科学与工程”。

总的来说，结果不如使用 Jaccard 相似度的匹配结果。我分析主要原因可能有以下几点：

a) 受笔记本电脑机能限制，无法对预训练模型进行训练微调。如果能在高等教育相关的语料库上对模型进行微调，BERT 模型学习专业和学科相关语义的能力应该会提高很多。

b) 缺少对数据的预处理。事实证明，李浩天先对数据进行预处理（如删除停用词、删除括号）之后的效果更好。我用 BERT 模型的结果也表明，结果会受到例如“单列”“定向”等没有意义的词语的干扰。

六、实验总结与心得体会

1. 实验总结

在本次实验中，我参与的工作是为知识图谱的构建做预处理，为专业名称作一级学科的标注。其中，我具体负责的是用 BERT 模型得到向量表示，用于进行从专业名称到一级学科的相似度匹配。这样，我们得到了一个有一定意义的结果。再经过与 Jaccard 相似度匹配得到的结果进行比对，以及人工校对，我们最终得到了一个较好的映射。

2. 心得体会

本次实践课程项目的主线是开发一个高考志愿填报助手网站。从这一角度看，我的工作——为知识图谱的构建做预处理，离主线较远。那些做 Web 前后端开发、知识推理和 AI 算法的同学，他们的工作才是离主线最近。这不能不说是一种遗憾

所幸，软件实践的目的也可以说达到了。因为这次在实验过程中使用人工智能中的自然语言处理技术进行预处理，也是用 Python 去进行软件编程；上学期数据库课程的大作业中，我使用 Django 开发了网上书店网站，对使用 Python 进行 Web 开发的技术已经学过了。

值得反思的是，我用 BERT 模型向量表示相似度匹配做出来的结果差强人意，不如李浩天用 Jaccard 相似度匹配的结果——这仅仅是字面匹配而已，并不像 BERT 一样涉及到语义。这提醒我，尽管很多人工智能学者夸夸其谈，在实际应用中，仍然需要考虑传统的方法，这些传统方法没有花哨的 CUDA、并行计算，却效率高、不吃内存。

当然，我们不能否认现代人工智能算法能做到许多很有意义的事情。但是在使用时，尤其是 NLP 应用领域，我们必须重视许多枯燥的工程问题，例如预处理、调参……只有这样，才能真正发挥人工智能算法的强大威力。

2020 年 9 月制