|  |  |
| --- | --- |
| **自然语言处理及应用**  **实验报告** | |
|  | |
|  |  |
| **名称** | 基于HMM的中文分词和语义标注 |
| **姓名** | 黄隆宁 |
| **班级** | 硕1224班 |
| **学号** | 3121155049 |
| Email | hln18773372567@stu.xjtu.edu.cn |
| **日期** | 2020-10-6 |

# 实验目的

利用HMM工具，实现一个简单的汉语分词和词性标注程序。

# 实验环境

Ubuntu 18.04

Python3.8

调用build-in库：

time：用于算法运行时间统计

math：用于数学函数的计算

pip依赖：

numpy：绝大部分的数据表示和数学计算

panda：用于DP矩阵的存储和迭代计算

sklearn：数据集划分及预测结果分析

# 实验方法

所有的实验代码都更新在我的Github，可在Github在线查看：<https://github.com/HLNN/NLP>

本实验要求基于以HMM工具实现汉语分词和词性标注，实验要求中推荐了HTK工具，。HTK是剑桥大学于上世纪九十年代推出的一款隐马尔可夫模型工具，该工具基于C语言实现。但因现在Python语言的相关支持工具更加丰富，开发更为便捷，因此在本次实验中，将使用Python语言编写隐马尔可夫模型基类，并在此基础上派生实现汉语分词和词性标注的子类。

我们在本次实验中实现的HMM类，需要完成的工作为隐马尔可夫模型的解码问题。通过一组观测状态序列求解最大可能性的隐含状态序列。

模型的训练部分，我们需要统计以下的数据：

首先是转移状态矩阵A，该矩阵表示马尔可夫链状态q1转变为状态q2的概率，通过统计q1之后紧接着出现q2的频次即可得到。

发射矩阵B，该矩阵表示马尔可夫链状态q对应观察序列中观测值o的概率。

另外还需统计初始状态概率，即每个马尔可夫链状态q，作为马尔可夫链的初始状态的概率。

对数据集完成参数统计之后，对这些数据进行平滑及归一化处理即可。模型的训练部分比较简单，运算复杂度较小。

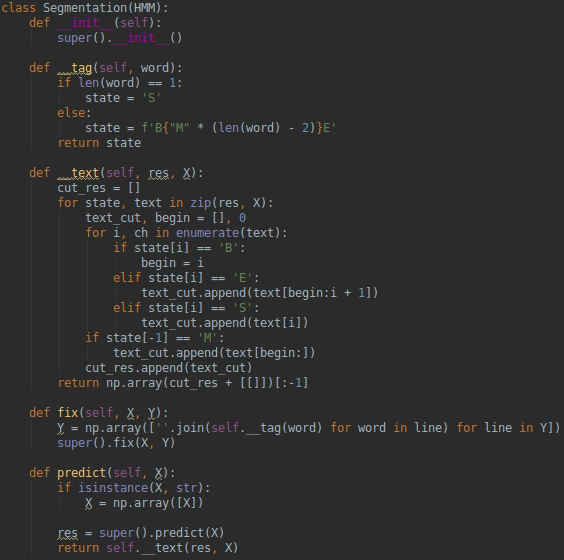
推理部分使用Viterbi算法提高运算效率。如果直接使用暴力求解，需要叠加的隐状态数量高达N^T个，其中N为隐状态的数量，T为数据序列的长度。算法的复杂度随数据长度指数增加，当数据过长时，暴力求解算法将难以在有限时间能得解。 Viterbi算法使用DP的思想，将全句求解全局最优转化为一系列的局部最优求解。通过记录t时刻，计算并存储每个状态的最大概率，省去大量不必要的计算。

正向迭代完成之后，只需反向回溯，比较每一个时刻t时，选择所有状态中概率最大的即可。

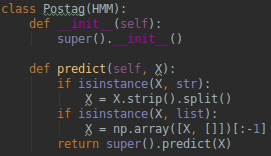
本实验的代码实现部分，按照功能不同分为以下几个部分：

hmm.py：实现核心HMM框架，包括fix和predict两部分。fix部分统计训练集数据，得到状态转移矩阵、发射矩阵和初始状态矩阵等参数。predict部分实现Vitervi算法，通过输入的观测序列求解最大概率的隐状态序列。

segmentation.py：在HMM类的基础上派生实现Segmentation类，并重写HMM类中的fix和predict函数。使用4-Tag的方式表示分词结果，其中B、M、E、S分别表示开始字、中间字、结束字和单字词。fix函数中，将数据集中以空格分隔的分词结果，转化为HMM类能处理的BMES序列。predict函数将HMM类返回的BMES序列转化为以空格分隔的分词结果。



postag.py：在HMM类的基础上派生实现Postag类，并重写HMM类中的fix和predict函数，以实现一些数据处理于转换。



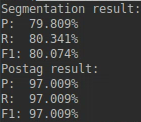
utils.py：实现了一系列的工具函数。包括数据集读取函数，汉语分词和词性标注的P、R、F1计算函数等。

evaluation.py：实现了数据集划分和模型性能评估。使用sklearn中的KFold模块，对数据集进行5折交叉验证，计算每一轮测试集的准确率P，召回率R以及F1指标，并计算5折的平均值。

1. **实验结果**

数据集采用北京大学计算语言研究所的1998年人民日报语料库，并对数据进行清洗。为了减小隐状态数量，将组合实体名进行了合并。针对词性标注，仅保留了26个基本词性，将拓展词性归类到对应的基本词性中。

分别对汉语分词和词性标注进行5折的交叉验证之后，平均的P、R、F1指标如下图所示：



汉语分词准确率、召回率和F1均在80%上下，词性标注的准确率为97%。整体来看，本次汉语分词和词性标志的结果不错。

1. **遇到问题及解决思路**

在一些汉语分词测试中，HMM输出的结果的最后一个隐状态为M，而这种状态时不可能出现的。通过分析状态转移矩阵，该情况只有可能是最后一位的隐状态E被错误预测为M。因此在4-Tag转化为分词结果时，手动判断组后一位是否为M，即可避免因最后一位错误预测而影响分词效果。

