小作业二报告

1. HMM 模型用于中文分词

1.1. TODO 1

替换自己的姓名。

1.2. TODO 2

此处要计算第一个位置的最大概率值，套用公式，检索已知的起始概率矩阵、发射概率矩阵中的参数，由转化好的首元素，套用公式，算出 dp 矩阵的边界值。

1.3. TODO 3

填写 dp 和 path 矩阵。

dp 矩阵的填写。我的想法是，对于句子中的每一个元素，取出 dp 中上一个概率值，根据转移概率矩阵，通过动态规划的思想来计算新的 dp 值，由于题目中说概率可以累乘因此我们这里采用累乘的形式。之后再取出相应的发射矩阵中的概率值累乘即可。

之后根据概率值，标记路径即可，之后的更新 label 会使用到。

1.4. TODO 4

根据 dp 和 path 矩阵的值填写 label 就可以了。这里我做了边界值的处理，因为名字不用划分开，因此可以直接打成 label = 0。

1.5. TODO 5678

前向后向算法的实现思路其实和维特比类似，因为都是使用了动态规划的方法来处理。前向算法不断用 α_{t-1} 更新 α_t，后向算法不断用 β_t 更新 β_{t-1}。

1.6. 结果

如果使用“小明是一名优秀的学生”进行测试，结果为“小明是/一名/优秀/的/学生/”。前向算法概率为 1.9261686664194958e-27，后向算法概率为 1.926168666419496e-27。

用“剡宜🄴是一名优秀的学生”进行测试，结果为“剡宜🄴/是/一名/优秀/的/学生/”。前向算法概率为 3.6571142357530564e-37，后向算法概率为 3.6571142357530564e-37。

2. BPE 算法用于英文分词

2.1. TODO 1

填写 build_bpe_vocab 函数，只需要根据需要转化成的格式，构建字典即可。

这里我们需要对每句话中的每个单词构造两两字母分开的字符串，同时要合并相同的字符串。

2.2. TODO 2

填写 get_bigram_freq 函数。

将每一个单词分割为多个 unigram，并构建可行的 bigram，同时统计每个 bigram 的个数即可。

2.3. TODO 3

填写 refresh_bpe_vocab_by_merging_bigram 函数，对给定的 bigram，遍历旧的词表，合并可以合并为给定 bigram 的 unigram，并统计个数。

2.4. TODO 4

填写 get_bpe_tokens 函数，根据词典，返回所得到的 BPE 分词列表，统计整理，将该列表按照分词长度降序排序返回即可。

2.5. TODO 5

填写 print_bpe_tokenize 函数，遍历词表，找到最长的匹配子序列，对其前后的字符进行递归调用进行分词，将结果拼接再一起。如果遍历完词表后，仍然无法匹配则返回 unknow。

2.6. TODO 6

首先统计 bigram，遍历找到找到其中频数最高的一个 bigram，并利用 refresh_bpe_vocab_by_merging_bigram 合并词表中相应 unigram。

2.7. 结果

naturallanguageprocessing 的分词结果为:
n a tur all an g u ag e pro c es s ing</w>