工作周报

时间：2018年10月22日星期一~2018年10月28日星期一

|  |
| --- |
| 工作内容 |
| 1. **学习并完成HMM与序列标注** 2. **学习并理解维特比算法**   下面是我对HMM维特比算法的具体理解：  一、隐马尔可夫模型（HMM）是一种统计模型，它用来描述一个含有隐含未知参数的马尔可夫过程。其难点是从可观察的参数中确定该过程的隐含参数，然后利用这些参数来进一步的分析。  任何一个HMM都可以用下列五元组来描述：  例子：老王每天根据天气（下雨，天晴）决定当天的活动（呆在寝室打扫，逛街，散步）中的一种，我每天只能在微信上看见她给我发的消息，我前天待在寝室，昨天和卢思童在大商场，那么我就可以分析前天在下雨，昨天天晴，在这个例子中，显状态是活动，隐状态是天气。  ：param observation:观测序列  observations = ('walk', 'shop', 'clean')  ：param states:隐状态  states = ('Rainy', 'Sunny')  ：param start\_probability:初始概率（隐状态）  start\_probability = {'Rainy': 0.6, 'Sunny': 0.4}  ：param transition\_probability:转移概率（隐状态）  transition\_probability = {  'Rainy': {'Rainy': 0.7, 'Sunny': 0.3},  'Sunny': {'Rainy': 0.4, 'Sunny': 0.6}, }  ：param emission\_probability:发射概率（隐状态表现为显状态的概率）  emission\_probability = {  'Rainy': {'walk': 0.1, 'shop': 0.4, 'clean': 0.5},  'Sunny': {'walk': 0.6, 'shop': 0.3, 'clean': 0.1}, }  二、求解最可能的天气  求解最可能的隐状态序列是HMM的三个典型问题之一，通常用维特比算法解决。维特比算法就是求解HMM上的最短路径（-log(prob)，也即是最大概率）的算法。  因为第一天我的朋友去散步了，所以第一天下雨的概率：  V[第一天][下雨] = 初始概率[下雨] \* 发射概率[下雨][散步] = 0.6 \* 0.1 = 0.06。  同理可得：  V[第一天][天晴] = 0.24 。  从直觉上来看，因为第一天朋友出门了，她一般喜欢在天晴的时候散步，所以第一天天晴的概率比较大，数字与直觉统一了。  从第二天开始，对于每种天气Y，都有：  前一天天气是X的概率 \* X转移到Y的概率 \* Y天气下朋友进行这天这种活动的概率。  因为前一天天气X有两种可能，所以Y的概率有两个，选取其中较大一个作为V[第二天][天气Y]的概率，同时将今天的天气加入到结果序列中  3. 比较V[最后一天][下雨]和[最后一天][天晴]的概率，找出较大的哪一个对应的序列，就是最终结果。  HMM模型在NLP方向上得到了较为深入的应用，特别是序列标注方面，其中分词、词性标注、命名实体识别都属于这一类。中文分词是将中文自然语言文本划分成词语序列，目前主流方法为序列标注，即用BMES这个四个标签去标注句子中的每一个字(B是词首，M是词中，E是词尾，S是单字词)。 |
| 问题记录 |
| 1. 通过了解维特比算法的基本概念然后用代码实现并给出了自己的注释   # 打印路径概率表 def print\_dptable(V):  print(" ",)  for i in range(len(V)): print("%5d" % i,)  print   for y in V[0].keys():  print("%.5s: " % y,)  for t in range(len(V)):  print("%.7s" % ("%f" % V[t][y]),)  print  def viterbi(obs, states, start\_p, trans\_p, emit\_p):   # 路径概率表 V[时间][隐状态] = 概率  V = [{}]  # 一个中间变量，代表当前状态是哪个隐状态  path = {}   # 初始化初始状态 (t == 0)  for y in states:  V[0][y] = start\_p[y] \* emit\_p[y][obs[0]]  path[y] = [y]   # 对 t > 0 跑一遍维特比算法  for t in range(1, len(obs)):  V.append({})  newpath = {}   for y in states:  # 概率 隐状态 = 前状态是y0的概率 \* y0转移到y的概率 \* y表现为当前状态的概率  (prob, state) = max([(V[t - 1][y0] \* trans\_p[y0][y] \* emit\_p[y][obs[t]], y0) for y0 in states])  # 记录最大概率  V[t][y] = prob  # 记录路径  newpath[y] = path[state] + [y]   # 不需要保留旧路径  path = newpath   print\_dptable(V)  (prob, state) = max([(V[len(obs) - 1][y], y) for y in states])  return (prob, path[state]) |
| 待办事项 |
| 1.继续深入学习。  2.分别用CRF和HMM完成分词与词性标注。 |