实验三 近似抽取

计 24 2012011335 柯均洁

一、 实验目的

从一段文本中抽取出与字典中的关键词相似的部分

二、 设计思路

方法：Trie-based Method

参考文献：Deng D, Li G, Feng J. An efficient trie-based method for approximate entity extraction with edit-distance constraints[C]//Data Engineering (ICDE), 2012 IEEE 28th International Conference on. IEEE, 2012: 762-773.

思路：

1. 若 ED 阈值为 th，则将词典中的 e 划分为 th+1 份（采用 even-partition）。若文档 d 与某个 e 相似，则至少有 e 的某个子串 em。匹配所有的 trie 即可得到候选 entity

2. 当检测到文档的某子串 d[i, j]对应的 trie 节点包含候选 entity e，则要找一个更长的子串 d[k, l]使得 d[k, l]与 e 的编辑距离小于 th

所以有

$$ED(e,d[k,l]) = ED(e_l,d[k,i-1]) + ED(e_m,d[i,j]) + ED(e_r,d[j+1,l])$$
$$= ED(e_l,d[k,i-1]) + ED(e_r,d[j+1,l]) \le th$$

因此，可以将左右两部分分开考虑，首先找到与 er 编辑距离<=th 的部分 d[j+1, l]，再从这些 er 对应的 el 中找出与其编辑距离<=th 的部分 d[k,i-1]，将两者组合起来，使得总编辑距离<=th。即可找到所有的串

3. 在计算左右两半边 ED 的时候，使用一种 soft-based extension method，先将候选字符串按字母序排序，在动态计算 ED 的同时将每一步结果都保留下来，然后计算时先求出当前 er/el 与前一个计算的 er/el 的最大公共前缀/后缀，这样就可以利用前面已经计算过的结果

4. 在匹配字符串的时候可以采用类似 KMP 方法的字符串比对方式，每个 trie 节点计算一个 next，减少主串的回退。（虽然实现了，本地测不出问题，但是调了好久都未能完整通过服务器端的测试…）

三、 代码框架

1. Node 类

   Trie 节点类。

   - 每个 Node 保存有一个孩子节点的数组 children。为了加快运算速度，直接将 10 个数字、26 个字母、空格、句号、逗号映射成为 26+10+3 个数字，每个孩子都对应一个字符。
   - 每个 Node 保存有 leftStrs 和 rightStrs。也就是每个 trie 节点对应 e 的 el 和 er 部分，可以调用 sort 对其进行字母排序

2. buildTrie(int threshold)

   根据 threshold 建立 trie 树

3. extension(Node* p, string& D, int left, int right, int threshold, vector<EDExtractResult> &result)

   当文档 D 和 trie 节点 p 匹配后，调用该函数计算候选字符串。首先将与 p 中 er 编辑距离<=threshold 的，然后再计算 er 对应 el 编辑距离<=threshold 的