一个呆萌的Query纠错算法

# 1、预想

我所理解的Query纠错是在拥有一个在海量文本上训练的单向语言模型，或者双向的，例如Bert，并在对应任务场景的海量数据集上预训练之后，对于给定的query，以beam search的方式做query生成。但实际情况是，对应任务场景并没有海量的数据集。一种简单直接的方式是把query转成pinyin，并与候选的query集合计算编辑距离，取改写最小的，根据候选的query的热度做排序选最优的。这样做效果不易于保证，其一，query改写往往是多段的，当该query与候选query有很多相似段的时候，这个候选集合就很难以有效的手段评估；其二，这种方式没法做到查漏补缺；其三，这种方式计算量随着热门query的增长而激增。

基于此，有一个新的想法，既可以保证query能够多段改写并且效果更优，又可以保证能够查漏补缺，还可以保证计算量不会随着热门query的增长而激增。最重要的意义是，该方法可以在对应任务场景没有海量数据集的情况下，做冷启动，为后续上模型积累自动标注的数据。

该想法便是用ngram和masked ngram做检索，然后依据ngram拼音的编辑距离来做小段的ngram纠错，通过beam search的方式保存最优的K个结果，这里的K为beam size。

# 2、算法简概

## 2.1 step 1

从对应任务场景的线上日志里聚合热门query，同时需要统计出每条query的频次，可以用于后面的决策过程。

## 2.2 step 2

基于热门query构建ngram检索词表和masked ngram检索词表，在本例中，N=3。例如：

你知道五道口地铁站怎么走吗

加入开始标记和结束标记，本例以\_为标记，改写query为：

\_你知道五道口地铁站怎么走吗\_

创建ngram检索词表

|  |
| --- |
| \_你知  你知道  知道五  道五道  五道口  道口地  口地铁  地铁站  铁站怎  怎么走  么走吗  走吗\_ |

创建masked ngram检索词表

|  |
| --- |
| \_你知[mask] => \_你知  你知[mask] => 你知道  知道[mask] => 知道五  道五[mask] => 道五道  五道[mask] => 五道口  道口[mask] => 道口地  口地[mask] => 口地铁  地铁[mask] => 地铁站  铁站[mask] => 铁站怎  怎么[mask] => 怎么走  么走[mask] => 么走吗  走吗[mask] => 走吗\_ |

## **2.3 step 3**

给定query，查询query的ngram是否存在于ngram检索词表，如果存在，则不用纠错，反之，mask掉该ngram最后一个字，查询该ngram是否存在于masked ngram检索词表里，如果不存在，则不做纠错，反之，则找到最多beam size个候选。伪代码如下：

|  |
| --- |
| # ngram纠错，返回候选ngram集合  rewrite\_ngram(ngrams):  candidates = []  for ngram in ngrams:  if ngram in ngram检索词表:  ngram加入候选  else:  new = mask掉ngram最后一个字  if new in masked ngram检索词表:  计算候选改写的ngram和ngram的编辑距离  依据编辑距离排序候选集合  编辑距离最小的都加入到候选集合里  else:  ngram加入候选  shuffle(candidates)  return candidates[:beam\_size]  # 这里K表示beam size  query\_correct(query):  q = '\_' + query + '\_'  query\_cands.add(q[:K-1])  # 已完成纠错的query候选  complete\_cands = set()  while query\_cands:  ngrams = 根据query\_cands和query构建ngrams  new\_ngrams = rewrite\_ngram(ngrams)  依据new\_ngrams和query\_cands再构建query\_cands  并将已完成纠错的qeury加入complete\_cands    从complete\_cands获取最优候选  根据该候选的长度，原query的长度，以及编辑距离决定是否做改写  返回改写后的query |

# 3、例子以及代码

本例随机抽取了QA训练集的100条query

|  |
| --- |
| 吃什么长睫毛 吃什么长睫毛快 1  自己做葡萄酒可以用电饭锅发酵吗 做葡萄酒的比例 0  再见美人鱼电影歌曲 再见美人鱼电影插曲歌名 1  宾馆的wifi怎么连接 酒店的wifi怎么连接?? 1  ...  ...  ...  湛江去海南多少公里 湛江到海南有多远 1  虚拟号段是什么意思 虚拟号码是什么意思? 1  墙头马上的孩子叫 笔记本电脑开机怎样进入安全模式 0  小米起了虫子还能吃吗 总有一天我要撕去这虚假的星空日语怎么读 0 |

生成ngram检索词表如下：

|  |
| --- |
| \_吃什 2  吃什么 2  什么长 2  么长睫 2  长睫毛 2  睫毛\_ 1  睫毛快 1  毛快\_ 1  ...  ...  ...  虚假的 1  假的星 1  的星空 1  星空日 1  空日语 1  日语怎 1  语怎么 1  么读\_ 1 |

生成masked ngram检索词表如下(其中^A和^B是单个字符)：

|  |
| --- |
| \_吃^A \_吃了^B2^A\_吃的^B1^A\_吃什^B2  吃什^A 吃什么^B2  什么^A 什么颜^B1^A什么穿^B1^A什么关^B1^A什么号^B1^A什么意^B3^A什么品^B1^A什么\_^B4^A什么?^B4^A什么灯^B1^A什么长^B2^A什么马^B1^A什么是^B1^A什么证^B1^A什么原^B2^A什么叫^B1^A什么我^B1^A什么牌^B1  么长^A 么长睫^B2  长睫^A 长睫毛^B2  睫毛^A 睫毛快^B1^A睫毛\_^B1  毛快^A 毛快\_^B1  ...  ...  ...  撕去^A 撕去这^B1  去这^A 去这虚^B1  这虚^A 这虚假^B1  虚假^A 虚假的^B1  假的^A 假的星^B1  的星^A 的星空^B1  星空^A 星空日^B1  空日^A 空日语^B1  日语^A 日语怎^B1  语怎^A 语怎么^B1 |

尝试纠错Query：你知道吗:

|  |
| --- |
| query: 你知道吗  ngrams: ['\_你知']  ngram: \_你知  replace new ngram [\_你] failed  candidates: ['\_你知']  ngrams: ['你知道']  ngram: 你知道  replace new ngram [你知] failed  candidates: ['你知道']  ngrams: ['知道吗']  ngram: 知道吗  rewrite [知道吗] to [知道手]  candidates: ['知道手']  ngrams: ['道手\_']  ngram: 道手\_  rewrite [道手\_] to [道手机]  candidates: ['道手机']  ngrams: ['手机\_']  ngram: 手机\_  rewrite [手机\_] to [手机实]  candidates: ['手机实']  ngrams: ['机实\_']  ngram: 机实\_  rewrite [机实\_] to [机实名]  candidates: ['机实名']  ngrams: ['实名\_']  ngram: 实名\_  rewrite [实名\_] to [实名制]  candidates: ['实名制']  ngrams: ['名制\_']  ngram: 名制\_  has ngram: 名制\_  candidates: ['名制\_']  new query: 你知道手机实名制 |

算法最终推导出的改写Query是你知道手机实名制，与原Query不等长，并且与原query的编辑距离很大，会在最终的决策中过滤掉，所以最终对该Query不会做Query改写。

尝试纠错Query宾馆的Wafi怎么链接:

|  |
| --- |
| query: 宾馆的Wafi怎么链接  ngrams: ['\_宾馆']  ngram: \_宾馆  has ngram: \_宾馆  candidates: ['\_宾馆']  ngrams: ['宾馆的']  ngram: 宾馆的  has ngram: 宾馆的  candidates: ['宾馆的']  ngrams: ['馆的W']  ngram: 馆的W  rewrite [馆的W] to [馆的w]  candidates: ['馆的w']  ngrams: ['的wa']  ngram: 的wa  rewrite [的wa] to [的wi]  candidates: ['的wi']  ngrams: ['wif']  ngram: wif  has ngram: wif  candidates: ['wif']  ngrams: ['ifi']  ngram: ifi  has ngram: ifi  candidates: ['ifi']  ngrams: ['fi怎']  ngram: fi怎  has ngram: fi怎  candidates: ['fi怎']  ngrams: ['i怎么']  ngram: i怎么  has ngram: i怎么  candidates: ['i怎么']  ngrams: ['怎么链']  ngram: 怎么链  rewrite [怎么链] to [怎么连]  candidates: ['怎么连']  ngrams: ['么连接']  ngram: 么连接  has ngram: 么连接  candidates: ['么连接']  ngrams: ['连接\_']  ngram: 连接\_  has ngram: 连接\_  candidates: ['连接\_']  new query: 宾馆的wifi怎么连接 |

该纠错结果非常NICE！！！！！！！！

算法代码见<https://github.com/jpegbert/search_coding/tree/master/query_correction>