深入讨论纠错系统

深度学习自然语言处理 2020-09-13

以下文章来源于CS的陋室 ，作者机智的叉烧

CS的陋室

陋室，用知识装点。房主主要谈论与数学和计算机相关的知识，不定时推送和个人学习...

来自：CS的陋室

纠错框架的基本结构

虽然纠错只是一个看着简单的任务，但是实际上已经构建成了一个非常完整的系统，根据这个系统兼顾很多事情，举几个点：

- 充分缩小范围，防止过纠，毕竟纠错是NLP系统的上游，过纠的代价非常大。
- 充分挖掘可能错误的位置，在词汇支持的情况，找到可能正确的结果，保证召回率。
- 各种抽取特征，能通过更加严谨的方式在多个候选中找到最优的结果。

然后我们来看看一个比较OK的纠错系统结构是什么样的，三大步骤：

- 错误检测：检测句子错误的部分，后续只对这个部分进行错误纠正。
- 候选召回：根据识别的错误进行针对性的修改，这块依赖候选集。
- 候选排序：错误可能有很多。召回的结果也有很多，那个才是最优解，这步需要通过一定的方式得到最优结果。

错误检测

错误检测是文本进入纠错体系的一个大门，设立他的目的有这几个：

- 缩小纠正范围，降低后续流程的压力。
- 减少过纠，保证准确率，用户自己输对了却改错了体验非常差。

换言之，我们需要在错误检测中做的是，找到句子中可能出现错误的位置，提取出来任务就完成了，那么，这块有什么方法呢。

- 最简单的一种方法，就是结合词典去做，这个词典其实已经有比较通用的，那就是jieba的词典（idf.txt），对于绝大部分人而言，输入的东西一般都是TOP的，那么一些未见过的，即未登录词，就很可能是错误的内容了。但需要注意的是，领域内的词汇我们需要补充，词典覆盖率要足够的高，这样识别的准确率才会够高。
- 第二个方法也是一个无监督的方法——用语言模型。语言模型能评判一个句子出现的概率，换言之，如果句子出现的频次足够低，那这里面就很可能有错误的词汇，再精确到句子中的每个位置，那就是一个局部的n-gram的条件概率了，如果概率比整个句子明显低，那就说明这个位置或者说这个位置附近可能存在错误点，我们可以拿出来。其实这个应该这里几个方法中门槛最低的一个了，只需要语料，不需要挖掘覆盖率足够大的词典，也不需要标注样本，直接可以做。
- 第三个想说的方法就是序列标注的方法。分词和NER其实都可以抽象为序列标注问题，错误检测也可以，简单的其实就是整个句子中，有错误的标注为1，没错误的标注为0，然后通过CRF之类的方法来进行预测，从而完成抽取。这个能很好的把控准确性，效果还是会比较好的，但问题在于这种标注样本，可能比常规的ner样本更难拿到。

pycorrector我前面提到过，就用了上面1、2两种方法，在开放域里面其实效果不错，但是在垂域，我们就需要更多的语料甚至是重新构建里面涉及的模型和词典。

而在我的实践中，又有如下的经验，大家可以参考：

- 错误检测这块由于只是纠错系统中的其中一部分，后续还有大量的步骤可以控制，所以我们并不需要对这步做非常高的准确性的要求，抱着"宁可错杀也不放过"的思路去做，**保证真正错的部分能被拿出来即可，对准确率可以很大程度的放松。**
- 无论是上面哪种方法，检测错误的时候都要注意，检测出错误的位置可能不是真的问题点，而可能识别出来的未知的附近，因此要扩大召回的话，附近的可以都挑出来试试一起处理。

候选召回

在指导错误的位置以后，我们就要开始对症下药了，那么，什么是可能的药，我们就要开始找了，这就是候选召回的主要任务，针对错误点，我们找可能正确的结果。要找到正确的结果，主要是两种方式：基于词典的和基于NLG的。

基于词典的方式是比较经典而且在现在还是比较常用的方法，说白了我们就要去找一些词汇，我们叫做"混淆集"，也就是一个简单的kv对，遇到什么词，我们就给出一些候选的结果，这个的结果非常简单，但是挖掘会非常困难，搜索领域常用的方式就是共现query，大部分情况下，用户会在没有得到正确结果的时候修正结果重新搜索，所以共现query是一个非常好的挖掘资源。

基于词典的方式纠错的量总有上限，但是总有一些难以召回的情况，因此借助一些NLG的方式，可以扩大召回，这个NLG，是一种文本生成的方式，可以根据上下文纠正的句子，给出一些可能的结果。但是这个方案的缺点是非常依赖平行样本，即一个错误、一个正确的样本，这个获取往往会比较难。

候选排序

现在对一个我句子，我们手里都有很多候选的结果，这里的候选排序主要有两个目的：

- 判断这么多候选结果中选出最好的几个。
- 最好的几个相比原来的句子要足够好，才能被纠。

这里，我们需要持续思考的是，这个排序规则改怎么定。

最简单的方式就是使用语言模型的perplexity，即混淆度，这是用来一个句子他真的是句子的打分，一般而言这个正常无错的句子ppl就会比较小，有错的句子是ppl的比较大，可以用这个指标来衡量最佳的纠错结果是什么。

$$ppl = \sqrt[N]{\Pi_{i=1}^{N} \frac{1}{p(\omega_i|\omega_1...\omega_{i-1})}}$$

光一个ppl的评判是不够的，不仅仅是ppl的相对量，还有绝对量，还有就是pl虽然下降但是还是很高，还有和ppl无关的因素，如拼音的相似度、和原句的相似度等，因此可以先升级为机器学习，把前面提到的指标抽取为特征，通过简单的机器学习进行计算。

进一步地，同样可以使用平行样本，通过深度学习的方式来衡量是否需要纠正。

其他相关

纠错只是一个系统，我们要在里面添加很多的零件完成各个我们拆解的任务，我们来看看有什么需要做的事情：

- 语言模型。语言模型在纠错中起到了至关重要的作用，因此一个好的语言模型非常重要，而影响语言模型效果的很大一块因素就是数据，尤其是统计语言模型，通过调整数据集的分布，例如使用特定垂域的语料进行训练，能有效提升最终的效果，但要注意不要把错误句子过多的引入到模型中。
- 混淆集。混淆集用于候选召回，如果正确结果无法被召回，则效果会受到很大影响，因此我们需要通过多渠道挖掘，在github、知乎等网站，加上一些论文提到的数据中收集外部数据，同时通过用户query，尤其是共现query来获取一些用户容易混淆的错误。有一篇文章提到了，混淆集是纠错的上限，正确答案召回不到，好的错误检测和候选排序都没用。
- 规则。纠错系统中需要大量的规则，错误检测阶段衡量错误的阈值我们要用规则卡，排序阶段我们也需要一些提权降权保证最终我们需要的内容能排在前面，例如一些专有名词的保护，”电池“不能被改为”滇池“，”赢政“不能被改为”行政“。这些规则看着简单，但是要想提出这些规则，必须对数据有足够的了解。

小结