中文平均信息熵的计算

陆旭军 ZY2203320
18376486@buaa.edu.cn

摘要

信息熵常被用来作为一个系统的信息含量的量化指标，从而可以进一步用来作为系统方程优化的目标或者参数选择的判据。本文通过马尔科夫链计算中文的平均信息熵，以字和词组为单位进行比较。

引言

1948 年，香农提出了信息熵的概念，解决了对信息的量化度量问题。如今熵 (Entropy)，信息熵，已经是机器学习中绕不开的一个概念。信息熵的公式：

$$\mathrm{H}(\mathrm{X}) = -\sum_{\mathrm{x}\in\mathrm{X}} \mathrm{p}(\mathrm{x}) \log \mathrm{p}(\mathrm{x})$$

克劳德·香农给出了信息熵的三个性质：单调性，发生概率越高的事件，其携带的信息量越低；非负性，信息熵可以看作为一种广度量，非负性是一种合理的必然；累加性，即多随机事件同时发生存在的总不确定性的量度是可以表示为各事件不确定性的量度的和，这也是广度量的一种体现。[1]

一元模型信息熵计算公式为：[2]

$$\mathrm{H}(\mathrm{X}) = -\sum_{\mathrm{x}\in\mathrm{X}} \mathrm{p}(\mathrm{x}) \log \mathrm{p}(\mathrm{x})$$

其中 p(x)为每个词在语料库中出现的频率，可以看作是单个词所携带的信息熵。

二元模型的信息熵计算公式为：[2]

$$H(X|Y) = -\sum_{x\in X, y\in Y} P(x,y) log P(x|y)$$

其中联合概率 P(x,y)可近似等于每个二元词组在语料库中出现的频率，条件概率 P(x|y)可近似等于每个二元词组在语料库中出现的频率与以该二元词组的第一个词为词首的二元词组的频数的比值。

实验过程

1.语料库预处理

语料库预处理的目的是读取所有的文本文件，并按自然段分割好。由于一元模型与二元模型在处理非汉字字符、空格、段落时稍有不同，因此预处理阶段不消除这些乱码文本，仅读入文本。

2.一元模型信息熵计算

一元模型仅涉及单个汉字，因此在预处理过程中需要将字母、数字、标点符号、特殊字符等全部去除。在消去这些噪声后用 jieba 分词得到单个汉字，这些所有汉字的字数为文本库的总字数。写一个 for 循环以统计每一个字出现的次数，这时用字典的方式进行记录{“str”:”int”}，分别记录的是字与字频。用 sorted()函数按字频从高到低重新排序。计算字的

平均信息熵时，代入一元模型信息熵计算公式即可，需要说明的是概率 P(x)=字频/总字数。

3.二元模型信息熵计算

二元模型至少涉及两个汉字，在预处理过程中需要将字母、数字、特殊字符等全部去除，但标点符号必须予以保留，否则会导致句末与句首重组后出现新的无意义词组。在消去这些噪声后导入汉字停词表用 jieba.lcut_for_search 分词得到词组，但得到的词组会含有大量的停词、空格，需要手动二次清除这些词，此时这些所有词数为文本库的总词数。

本文二元模型使用的是可以很方便地统计词频的 Counter() 函数，有了这个函数就不用手动的使用 for 循环来手动统计词频。Counter() 是 collections 库中的一个函数，可以用来统计一个 python 列表、字符串、元组等可迭代对象中每个元素出现的次数，并返回一个字典。

计算字的平均信息熵时，先计算联合概率分布 P(xy)=词频/总词数和条件概率 P(x|y)=词频/首字为 y 的词频数总和，最后代入二元模型信息熵计算公式即可。

值得注意的是，在计算条件概率 P(x|y)时，首字为 y 的词频数总和的计算可以简化，可以先将所有词组的首字提取出来作为一个 list，单独对这个 list 统计词频。这样做比每次都比较所有不同词组首字效率并循环累加提高 38 倍。

实验结果

1.一元模型

字库总字数： 3540274

不同字的个数： 151124

信息熵：12.925112873560831

出现频率前 10 的一元词语：“的”——97054、“了”——88280、“他”——53611、“是”——53178、“道”——48465、“我”——47012、“你”——46428、“在”——36445、“也”——27199、“这”——26514

2.二元模型

词库总词数： 2410399

不同词的个数： 150175

信息熵： 4.4074388039969925

出现频率前 10 的二元词语：

“叫说道”——11137、“一个”——8882、“小宝”——8772、“自己”——8700、“韦小”—— 8687、“韦小宝”—— 8657、“什么”——7483、“不是”——6699、“令狐”——6431、“一声”——6106

3.结果分析

一元模型是基于单个字出现的频率来计算概率的模型，而二元模型则是基于相邻的两个词组合出现的频率来计算概率的模型。

对于一元模型，给定一个字库，总共有 3540274 个字，其中不同字的个数为 151124 个。通过对字出现的频率进行计算，可以得到该模型的信息熵为 12.93。这个信息熵值表明，在一个字出现的概率分布下，信息的平均不确定度为 12.93。

对于二元模型，给定一个词库，总共有 2410399 个词，其中不同词的个数为 150175 个。通过对相邻词组合出现的频率进行计算，可以得到该模型的信息熵为 4.41。这个信息熵值表明，在一个相邻词组合出现的概率分布下，信息的平均不确定度为 4.41。

通过对比可以发现，二元模型的信息熵值远小于一元模型的信息熵值，说明二元模型比一元模型更能准确地描述语言的规律和结构。这是因为二元模型考虑了相邻的两个词组合出

现的频率，可以更好地反映词与词之间的语义关系和上下文信息，从而提高了预测的准确性。而一元模型只考虑了单个字出现的频率，缺乏上下文信息的约束，因此无法准确地预测下一个字的出现概率。

总结

N-gram 模型是一种基于统计的语言模型，它可以根据给定的文本序列建立起一个基于前 N 个词的概率模型，用于对下一个词或词组进行预测。而 N 值的大小则决定了模型考虑前后文关系的长度。

同时，随着 N 取值变大，文本的信息熵会越小，这是因为 N-gram 模型在计算概率时需要考虑更多的前后文关系，进而可以更准确地预测下一个词或词组。因此，N-gram 模型可以帮助我们理解文本的结构和语言规律，同时也可以应用于自然语言处理中的诸多任务，如文本分类、机器翻译、语音识别等。

参考文献

[1] https://blog.csdn.net/weixin_44966965/article/details/124007507

[2] Peter F. Brown, Vincent J. Della Pietra, Robert L. Mercer, Stephen A. Della Pietra, and Jennifer C. Lai. 1992. An estimate of an upper bound for the entropy of English. Comput. Linguist. 18, 1 (March 1992).