Sentence2Vec 探索

原创 数据科学初学者 数据科学初学者 2019-07-26

需求

Word2Vec应用已经比较普遍.
基于类似思路,希望对句子级别内容实现向量化,以便后期使用时可以高效运算,比如寻找相似句子等.

目录

本篇练习主要讨论:

- 如何评价句子向量化效果
- 进一步探索必要性的背景
- 参考的一些论文及阅读思考
- 实际探索的方向
- 探索实验结果

评价标准

在无标签情况下,或可使用最近邻查找或聚类等效果用以反映向量化的效果,作为特征进行分类可考虑.
特别地,Annoy挺好用.
另外,也可试试二维或者三维可视化看看效果.

背景

基于Word2Vec,尝试了最简单的词向量平均及TFIDF加权平均,结果表明,在短句(<20左右)的情况下,效果还不错.
但是句子一旦稍长,结果就不甚可用.文章级别就更勿论.
NIPS 2013 Deep Learning Workshop:NNforText.pdf中page20处也早就提到了这一点,亲身肉测

发现确实.
(要是文档链接看不了的可以跟我吱一声)

想想其实可理解,不管是平均或者加权平均,都丢掉了句子结构信息,在短句情况下,句子结构本来也比较简单一些,所以可能信息丢失不多,尚可.
句子越长越复杂,结构越复杂,平均丢掉的信息也就越多,结果就扑街.

所以,基于Word2Vec,捕捉句子结构及顺序,可能会是一种值得探索的方向.

另外,不久之前的Bert或可考虑是否有用.当然,如果准备用Bert中间层导出,基本也就不用考虑LSTM模型了,因为结果肯定是有差距的.

论文阅读

Distributed Representations of Sentences and Documents
介绍了将文章视为另一个词的方法,文章所代表的“词”将刻画哪些未出现在当前上下文中的信息,故称:Distributed Memory Model of Paragraph Vectors(PV-DM).上下文信息通过滑动窗口得到.
论文介绍的Paragraph Vector方法从逻辑到实现上都比较通畅明晰,训练时间论文提到的是2.5万(平均230词)句子在16核机器上需要30分钟.个人觉得可以实验一下.
而且文章也间接总结了之前相关的一些论文,也可以作为索引.
ps1:在gensim的Doc2vec实现方法上用300万的中文句子做了尝试.结果表明,还是只能对短句起到比较好的效果,长句仍无法达到推进生产的标准.(所耗时间倒是跟论文所提差不多)
ps2:还试了一个Pytorch版本的Paragraph2vec,效果也不是很好,且面对新句子没法推导vec.但是思路还是可以借鉴的,或可在此基础上改进一下.

On sentence representations
是一篇介绍文章,主要介绍了句子表示方法.比较详细的介绍了最近的/较早的-监督/无监督句子表示方法.可以作为综述文章阅读.

BERT
bert-as-service
这个项目写得挺好的,耦合比较规范,但我就是不知道作者为啥把代码都写在了**init**.py中,但项目仍然是个非常好的项目.
直接可用.

探索方向

主要实验了mean Word2Vec/ paragraph vector / Bert embedding

Bert embedding的操作是输入句子,利用Bert预训练模型得到中间层结果(实验以得到的第12层为基准)
层越往前,可能效果会越有损失,但是速度可能会更快一点.

实验结果

文章从6月中开始写,但是伴随着实验结果不理想以及方法的探索,一直写到7月底,速度也是够慢的.

结果是,使用Bert embedding能够得到目前可以获取的最好结果,效果基本可以达到Word2Vec水平.
所以基本达到了可以应用的水平.
可以达到聚类可用的水平,近邻搜索用于召回或许还有点问题.

速度上来讲,120长度,基于CPU需要200ms左右,基于GPU需要30ms左右.
句子长度(需要保留的长度)越长速度也会越慢,层越往后也会越慢.

总结

目前Sentence2Vec基于最佳的Bert可以得到最好效果.
通过聚类可以发现,能够较好的刻画关键信息,即使句子比较长(<512),长度500左右基本可以覆盖中短篇章.
再往后估计需要期待下一步的NLP发展,仍需探索.

参考

[1]. Distributed Representations of Sentences and Documents
[2]. A Gentle Introduction to the Bag-of-Words Model
[3]. BERT
[4]. bert-as-service