基于深度学习的情感分析方法在评论处理的应用

Krse Lee/ 11,6, 2014

Guide Line

深度学习

情感分析

统计工作

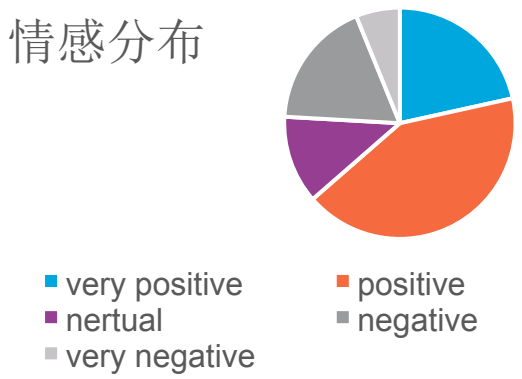
引言

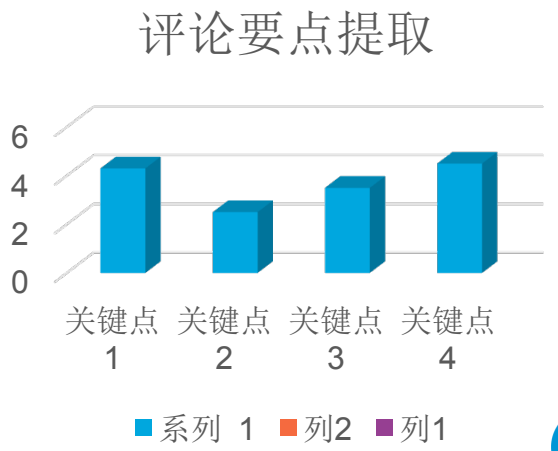
企业产品的生产与市场需求状况息息相关。

了解用户的喜好分布，生产出受消费者欢迎的产品是我们追求的目标。

为了了解消费者对产品喜好情况，我们抓取了Amazon上各个HP笔记本型号的用户评论，并设计了相应的程序对这些评论进行分析，从而了解用户喜好的分布情况。

- 准确了解用户在使用HP的产品时的具体体验，特别是各个功能细节，例如对于笔记本：预装的OS，屏幕，硬盘...
- 对于体验不佳的方面，HP可以在开发新型号时有针对性的改进。
- 对于体验好的方面，HP可以重点向消费者宣传。





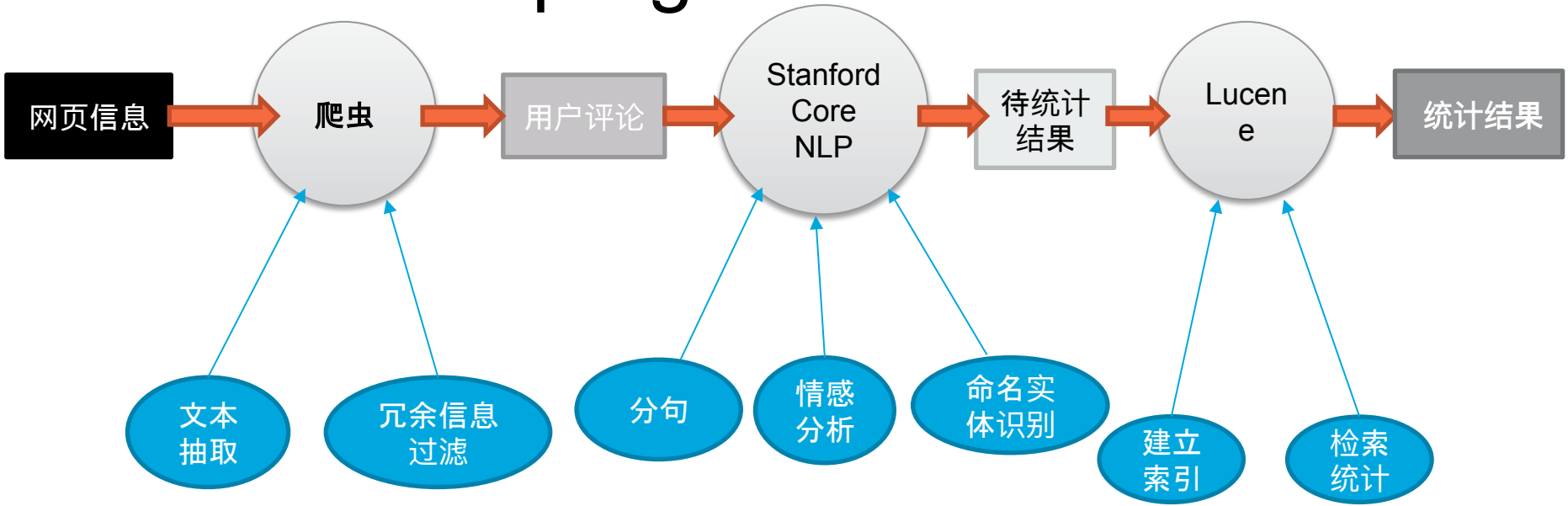
情感分析的应用领域

股市预测：Derwent Capital Markets推出了一支基于Twitter的对冲基金，通过分析大量推文中反映出的情绪来判断市场涨跌，并承诺每年15-20%的高回报率。

幸福感调查：佛蒙特大学计算实验室的Hedonometer项目组通过自然语言处理，对过去五年中每天发布的千万条推文进行情感分析，统计出一年什么时间人们的幸福感最强。

其他：电影评论、产品评价、用户反馈等。

How does our program work?



使用我们的方法，可以了解到对某一产品，用户的总体评价分布（好评、中评、差评）、针对某一特定部分的喜好分布、众多评论中具有代表性的评价，并且可以从用户评论中自动提取评价的关键点进行统计。

深度学习

向量空间模型的不足

深度学习概述

基于递归神经网络的语法树的建立方法

传统向量空间模型的不足

传统的向量空间模型中，本文向量的维度值是由一个个单词确定的，例如：I am a software engineer. 那么，确定的向量为：

I	am	a	software	engineer
1	1	1	1	1

我们使用机器学习算法进行文本挖掘大多是用的这样的向量，当比较文本相似度是，比较的是两篇文档中相同词汇出现的频率，出现的词频越相近，两篇文档越相似。

考虑这样两个句子

The country of my birth

The place where I was born

这是两个意思相近的句子，但是采用传统向量比较，它们的相似度差异就比较大。

传统向量空间模型的不足

句子1：The country of my birth
句子2：The place where I was born

$$Sim(D,Q) = \frac{d \cdot q}{\|d\| \times \|q\|} = \frac{\sum_{k=1}^{t}(d_{ik} \times q_k)}{\sqrt{\sum_{k=1}^{t} {d_{ik}}^2 \times \sum_{k=1}^{t} {q_k}^2}}$$

	The	country	of	my	birth	place	where	I	was	born
句子1	1	1	1	1	1					
句子2	1					1	1	1	1	1

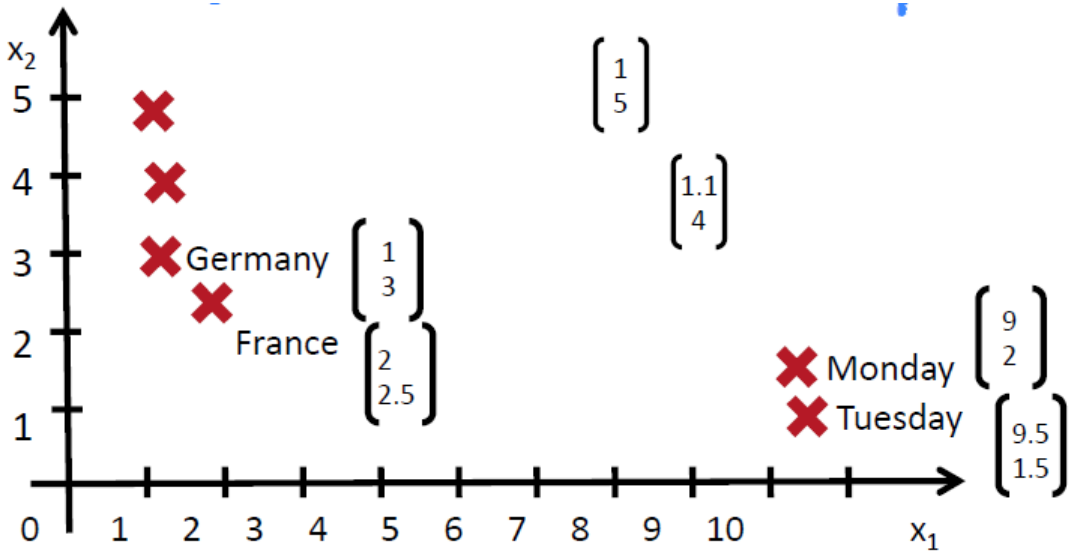
计算出两个句子的余弦相似度为$1/\mathrm{sqrt}(30) \approx 0.1825$

即两个句子相似度非常的低，这个结果显然是我们不希望看到的，这种方法进行文本聚类效果可想而知。

深度学习及 word vector space model

如果我们有一种方法，能够从单词开始构建向量，将意思相近的单词构造成相似的向量，那么这些相似单词构成的句子向量也就会相似。

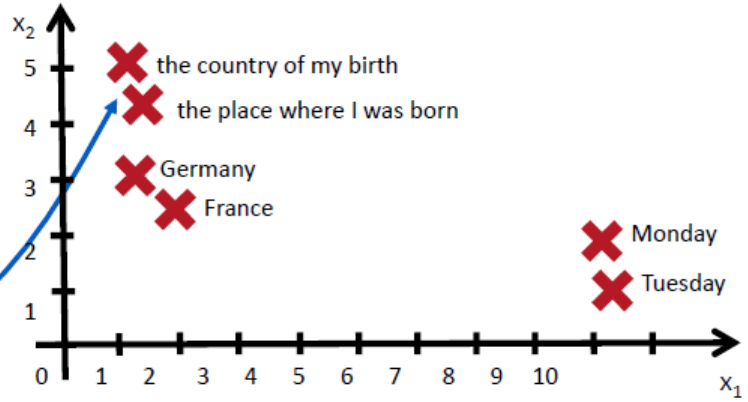
这就是Stanford NLP Group提出的基于深度学习的自然语言处理方法，深度学习也是机器学习的一个范畴。

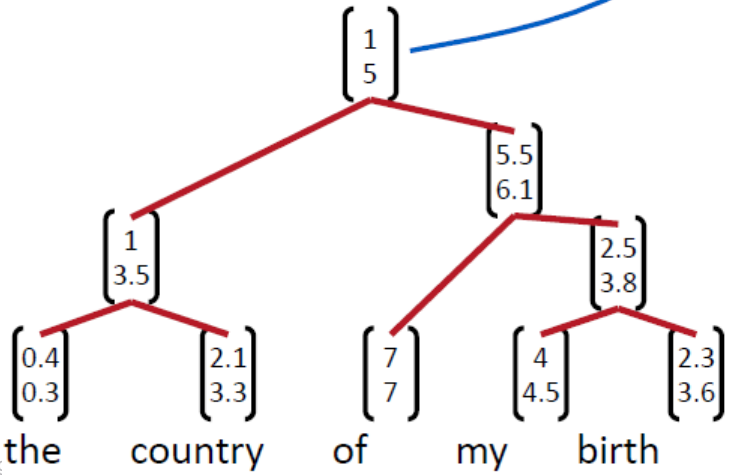


深度学习及 word vector space model

Use principle of compositionality

The meaning (vector) of a sentence is determined by
(1) the meanings of its words and
(2) the rules that combine them.





Models in this section can jointly learn parse trees and compositional vector representations

深度学习的优点

抓住单词之间的语义关系，句子不再由独立的单词组成，而是一个有意义的实体。

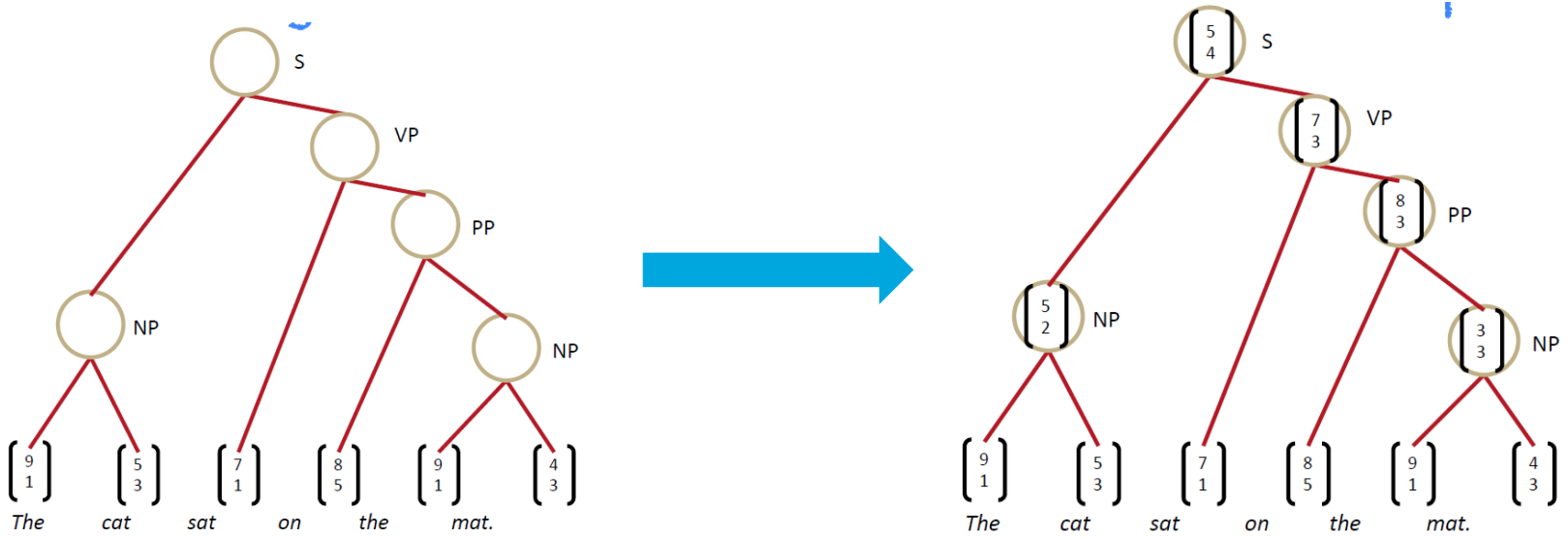
可以从从语义层面进行文本挖掘

掌握句子的语法结构和各部分的依存关系

强大的算法支持

问题：

一个句子如何构造适合深度学习的语法树？
单词向量如何叠加成短语向量，再到句子向量？



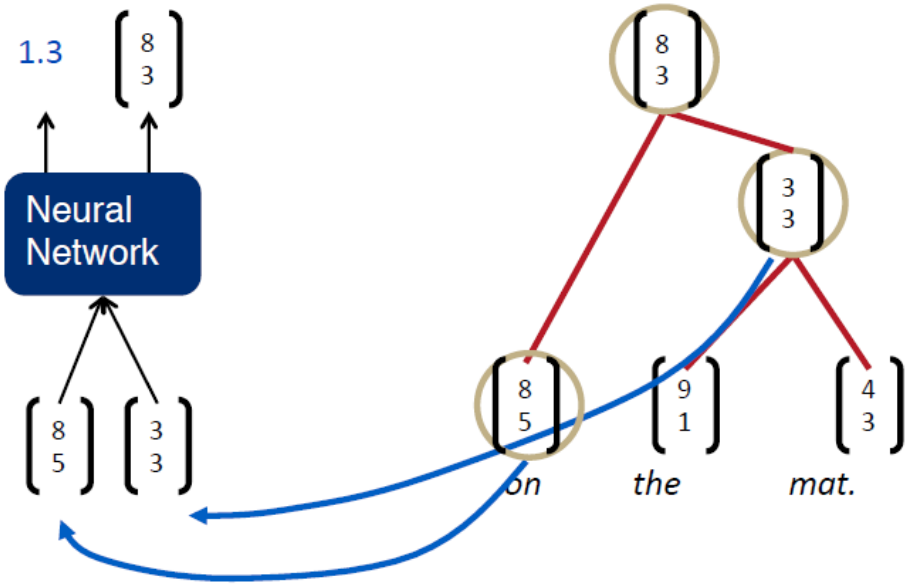
方法:递归神经网络

输入:
两个候选子节点

输出:
1.两个子节点合并后的合适度
2.两个子节点合并后的语义向量

$$p = \tanh\left(W \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} + b\right),$$

$$\text{score} = U^T p$$



神经网络

神经网络是一个能够学习，能够总结归纳的系统，也就是说它能够通过已知数据的实验运用来学习和归纳总结。

a1~an：输入向量的各个分量

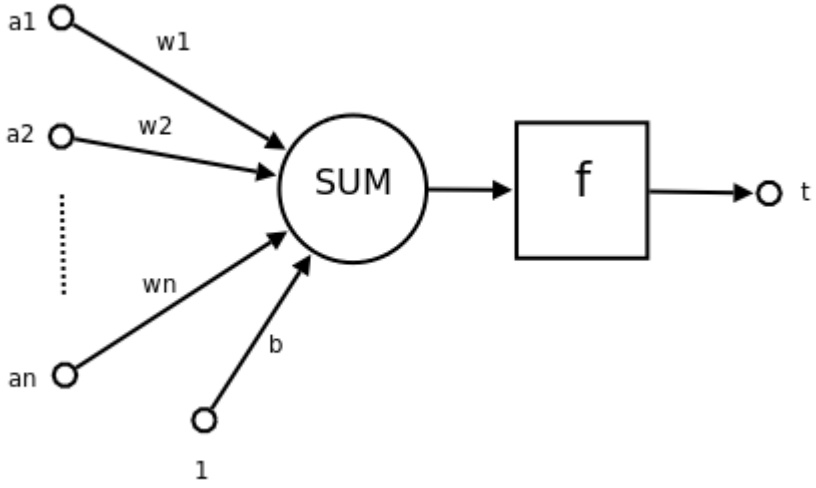
w1~wn：神经元各个突触的权值

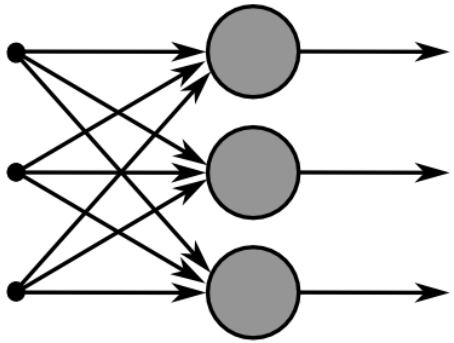
b：偏置

f：传递函数，通常为非线性函数。

t：神经元输出

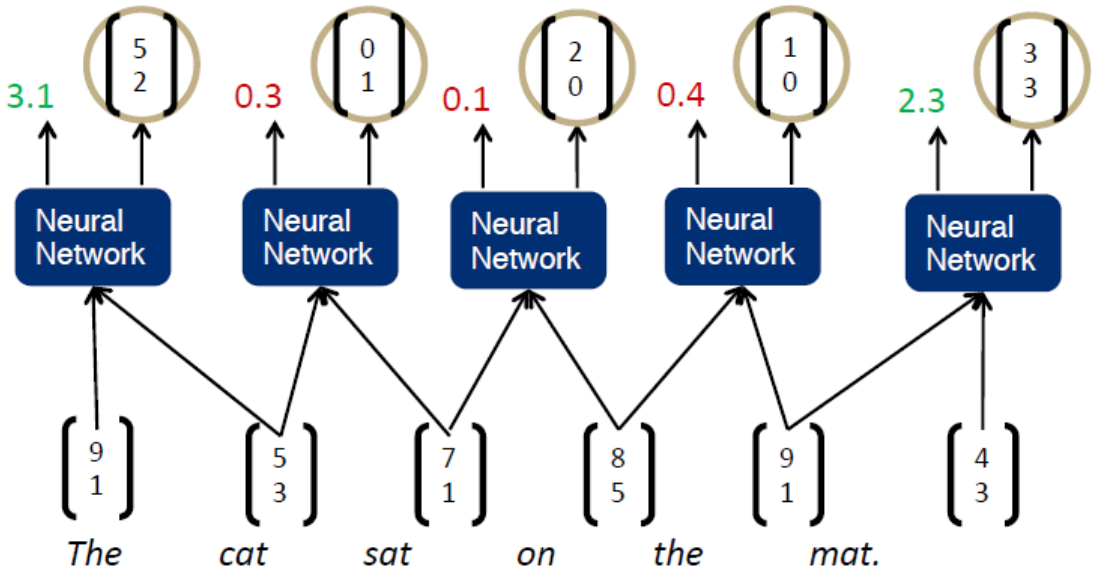
$$t = f(\vec{W}\vec{A'} + b)$$



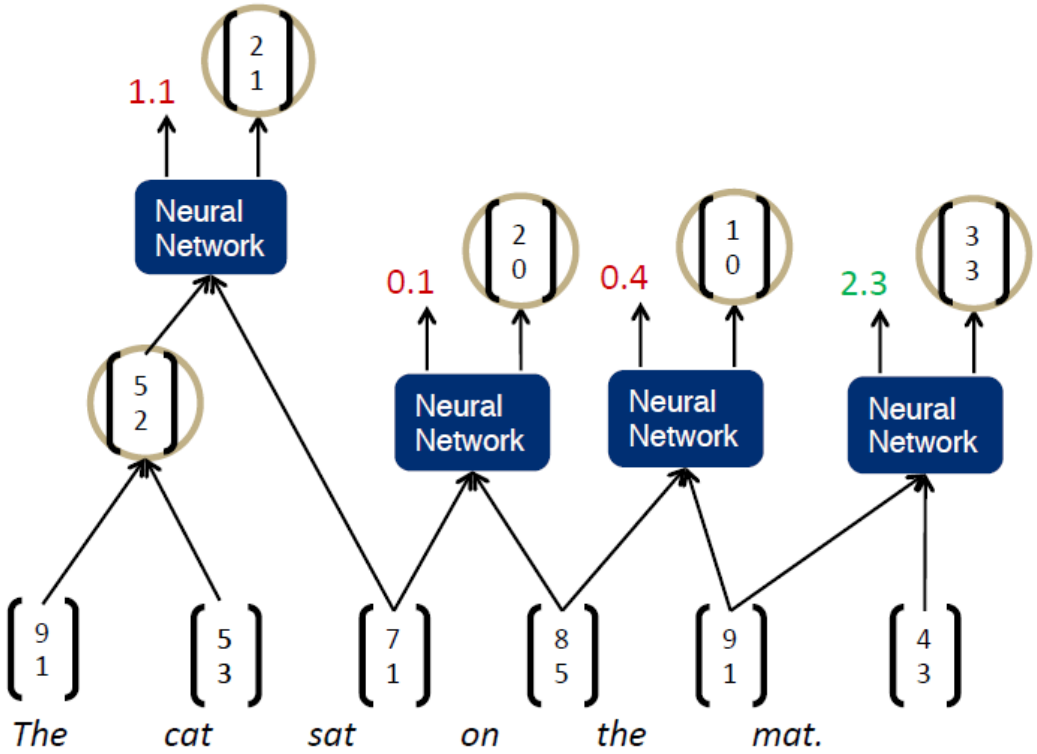


output layer

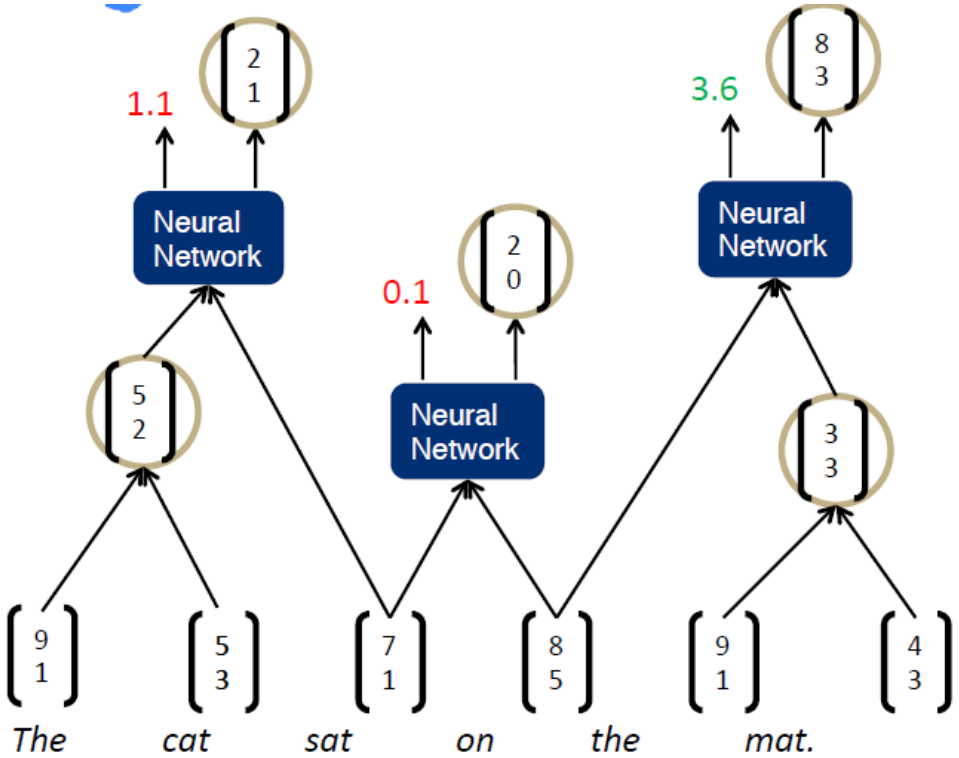
语法树建立过程



语法树建立过程



语法树建立过程



情感分析

情感分析基本方法

基于递归神经伸张网络的情感分析

情感分析

情感分析是指输入一段文字，通过其中的特征词和一定的算法，分析出该段文字的情感是积极、消极还是中性的方法。

常见的情感分析算法有：

贝叶斯

SVM（支持向量机）

递归神经网络

递归神经伸张网络

……

Model	Fine-grained		Positive/Negative	
	All	Root	All	Root
NB	67.2	41.0	82.6	81.8
SVM	64.3	40.7	84.6	79.4
BiNB	71.0	41.9	82.7	83.1
VecAvg	73.3	32.7	85.1	80.1
RNN	79.0	43.2	86.1	82.4
MV-RNN	78.7	44.4	86.8	82.9
RNTN	**80.7**	**45.6**	**87.6**	**85.4**

情感分析被广泛运用于趋势预测、舆情监控等方面，而对于用户评论，我们也可以采用情感分析方法来统计用户对产品各个方面的评价分布。

情感分析步骤

情感词典的建立:

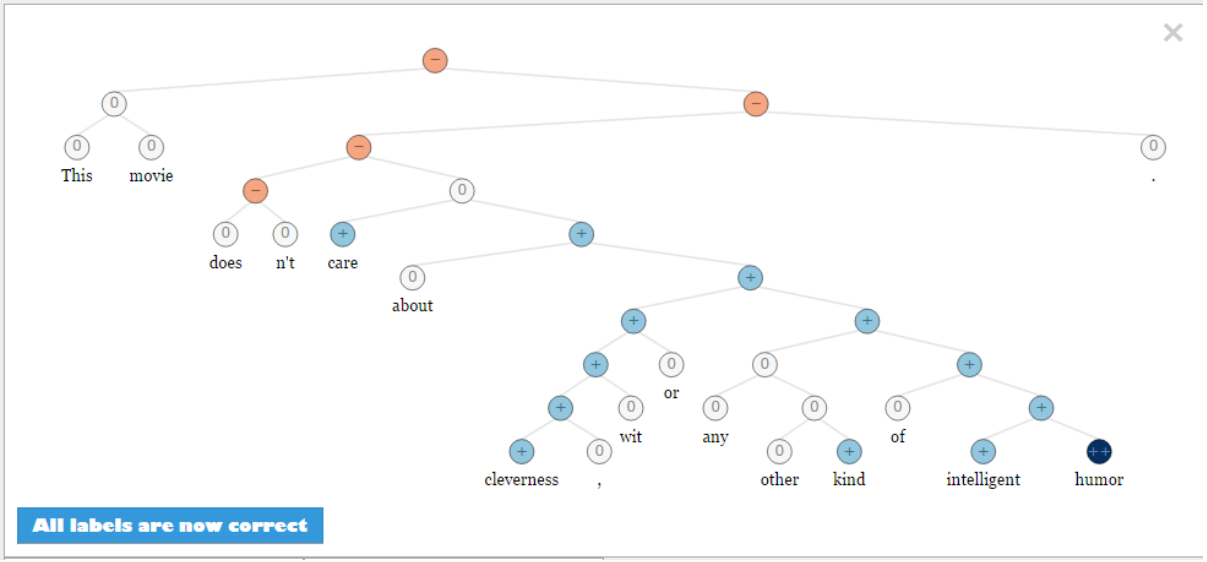
情感词典用于标注句子中每个单词的情感向量。

语法分析:

用于建立语法树。

递归伸张:

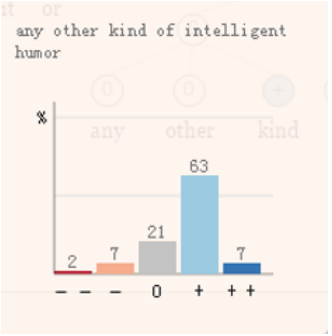
从语法树的最右叶子节点开始，向上递归伸张累加，计算每个节点的情感向量值，直到根节点。整个过程是一个递归神经伸张网络的计算过程。



句子的情感判断:对于根节点

Score > positive 阈值	positive sentence
Score < negative 阈值	negative sentence
Else	neutral sentence

递归神经伸张网络(RNTN)



递归定义:

对于任意长度的短语输入，短语都被表示为一个word vector和一棵语法树，从叶子节点开始，始终使用同一个**神经伸张复合函数** p = g(x , y) 来计算更高层节点的word vector。

优点:

与词袋模型不同的是，RNTN(Recursive Neural Tensor Network)可以准确地捕获句子的情感变化和否定词作用域。

详情可见如下论文:

http://nlp.stanford.edu/~socherr/EMNLP2013_RNTN.pdf

递归神经网络模型

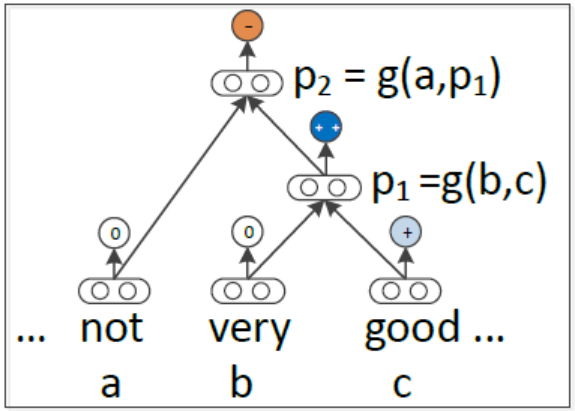
递归神经伸张网络是由递归神经网络演变来的。

任意给定一个短语或句子，利用递归神经网络的方法构造成一棵语法树，每一个叶子节点代表一个单词，其有一个word vector和一个情感向量。

我们已经讨论过word vector的叠加方法，接下来主要是情感向量的叠加。

同样，使用bottom up的方法，两个叶子节点作为神经伸张复合函数 g(x,y)的输入，计算出其父节点情感向量p的值，不断往上计算，直到语法树的根节点。

注意：单词、短语、句子都有一个相同类型的情感向量，短语的情感向量由单词计算得到，句子的由短语和单词的计算得到。



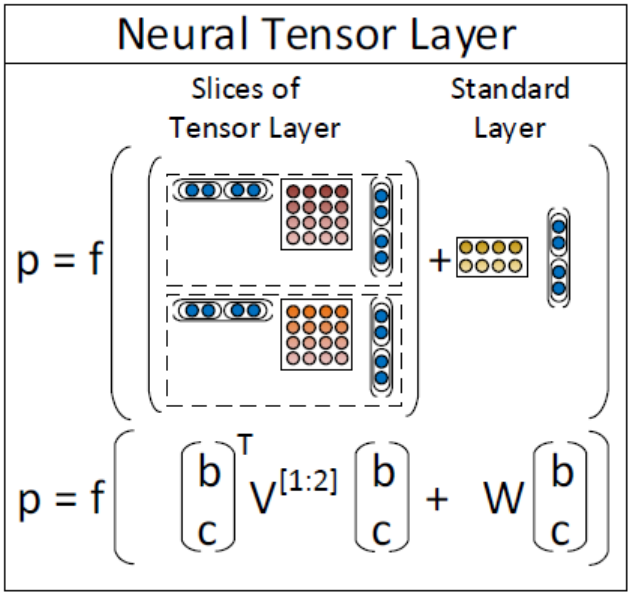
递归神经伸张网络

递归神经伸张网络在计算时与递归神经网络有所不同，在于有一个“伸张层”的计算。

- 在该模型中，单词情感是一个维度为d的向量；
- 用一个矩阵L（d×|V|）存放所有的单词，其中|V|是词汇表中单词的个数。
- W（d×2d）是一个参数，叫做情感分类矩阵，经过机器学习训练获得。

于是，我们可得到神经伸张复合函数的形式如下：

$$p_1 = f\left(\begin{bmatrix} b \\ c \end{bmatrix}^T V^{[1:d]} \begin{bmatrix} b \\ c \end{bmatrix} + W \begin{bmatrix} b \\ c \end{bmatrix}\right)$$



$$t = f(\vec{W}\vec{A'} + b)$$

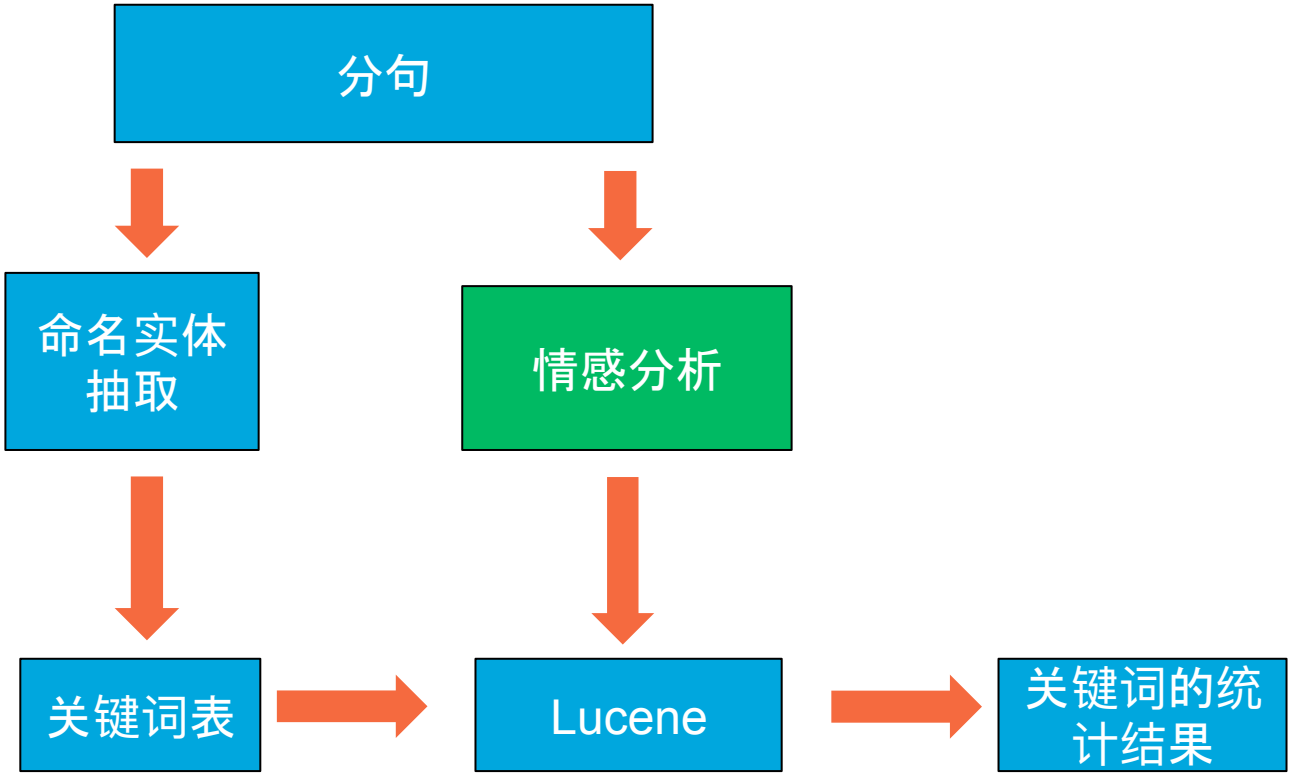
剩余工作

分句

命名实体抽取

基于Lucene的统计分析

评论处理工作流程

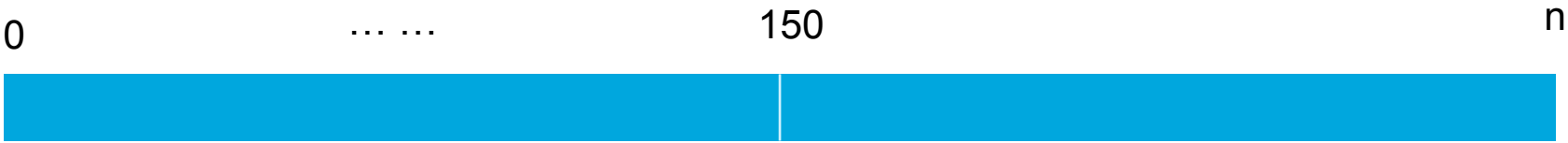


分句

Stanford core nlp 虽然提供了分句功能，但是对于长度超过80个单词的长文本，执行效率非常的慢，并且容易发生内存溢出。因此在进行分句工作之前，需要进行一些预处理：

从文本起始处开始，进行如下迭代：

```
count++;
if    count>=150
      向前寻找最近一个句子终结符，将文段拆成两个部分
      原文本 = 拆分后剩下的文本
      count = 0
```



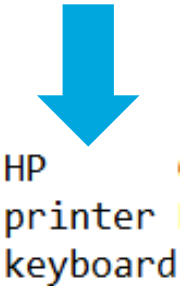
向前寻找最近一个句子终结符，将文段拆分

命名实体抽取

使用Stanford core NLP中的NER组件可以轻松完成，其实现原理是条件随机场，可以根据周围出现词的特征判断该词是不是命名实体。

其效果如下：

HP is a very big company , it produce not only computer and printer , but also keyboard !



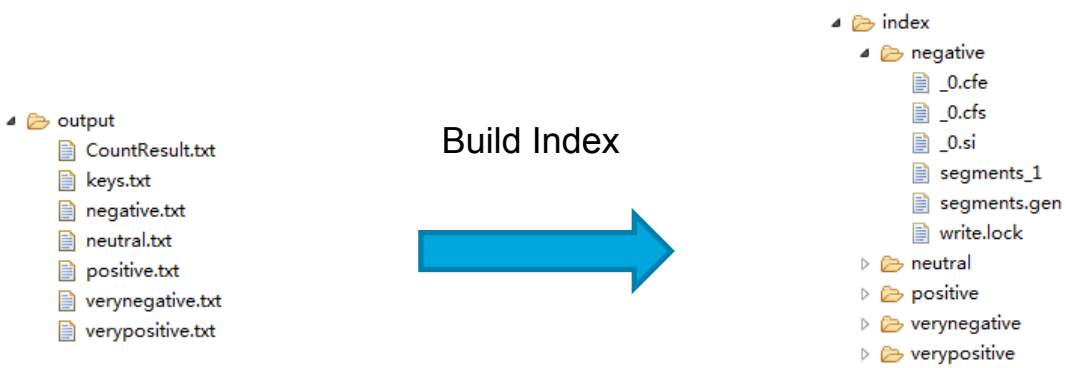
总共可以分离出7类命名实体，包括日期、时间、机构名、人名、地名、专有名等。

考虑到实际运用效果，我们需要对结果进行一定的人工整理，得到关键词表。

使用Lucene进行分析结果统计

Apache Lucene 是一个信息检索和文本处理工具，利用它可以快速、准确统计出我们需要的结果。

经过情感分类的评论被分别存储在了5个文件中，对5个文件分别建立索引，完成后便可以利用上一步得到的关键词表进行检索，并统计各个关键词中的情感分布。



结果统计

图示为对ChromeBook的检索结果。

我们得到谈论关于Chrome Book的总评论句数、各个情感下的分布情况、各个情感下的代表评论……

```
44 ================ Chromebook ==================
45 totalnum :  605
46 veryposi :  15  percentage :    2.48 %
47 positive :  133 percentage :    21.98 %
48 neutral  :  69  percentage :    11.4 %
49 negative :  360 percentage :    59.5 %
50 verynage :  28  percentage :    4.63 %
51 ---------------------veryposi-----------------------
52 This is the best Chromebook yet!
53 Chromebook 11 by HP is amazing!
54 I really enjoy my Chromebook immensely!
55 We were very disappointed.What is good about this Chromebook?The screen is fantastic, th
56 It seems much better and more solid than the HP Chromebook.
57 ---------------------positive-----------------------
58 Great Chromebook.
59 The chromebook is beautiful.
60 The Chromebook in general looked great.
61 Still, it is an excellent Chromebook.
62 The Chromebook itself is great.
63 ---------------------neutral-----------------------
64 It's a Chromebook.
65 Chromebook is my primary computer.
66 Now what about this Chromebook.
67 Under the Chromebook is the charger.
68 :  Get a different Chromebook.
```