**论文研究内容**

在细粒度观点挖掘中，运用RNN模型及词向量(word embedding)

介绍论文研究内容的时候，尽量精确一些，具体到每个子任务，重点介绍论文实验中所涉及的任务。

像这里，“细粒度观点挖掘”其实是一个比较大的概念，正如文中所说：

Fine-grained opinion mining involves identifying the opinion holder who expresses the opinion, detecting opinion expressions, measuring their intensity and sentiment, and identifying the target or aspect of the opinion

实际上，可以分成四个子任务

1. identifying the opinion holder（观点持有人识别）
2. detecting opinion expressions（观点抽取）
3. measuring their intensity and sentiment（观点情感分析）
4. identifying the target or aspect of the opinion（评价对象抽取）

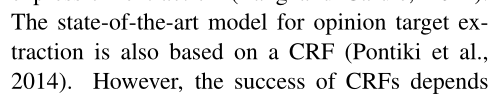
这篇论文实际上是在d这个任务上做的实验

**目前方法及存在问题**

1、CRFs (比如state-of-art model)

问题：由于对良好的特征集(appropriate feature set)和对特征函数(feature function)的阐述(感觉翻译的不太准...)，并且会消耗大量的人力在每个训练上。

这里就有点尬了，我还好奇地找了找，这里是原文



state-of-the-art 其实是个英文短语啊，意思是“最先进的、最新的”。这句话的意思是“目前在评价对象抽取方面最先进的模型也是基于条件随机场（CRF）的（Pontiki等人发表在2014年）”

这里涉及到CRF，你们可能没接触过，可以简单地上网查一查。推荐你看一下周志华的《机器学习》或者李航的《统计学习方法》，这两本属于机器学习入门必看的书，建议反复阅读。

2、candidate aspects

3、hidden Markov model

4、LDA(Latent Dirichlet Allocation) and its variants

（后三个的存在的问题在论文里并没有找到对应内容）

机器学习实际上是一个比较大的概念，广义上，它可以指。。。（具体定义我怕描述的有偏差，容易误导你，那两本书里应该都有详细定义）。“深度学习”实际上是“机器学习”中的一个大类，属于人工智能领域的state-of-the-art方法。但是在我们日常学习生活中，“机器学习”往往用的是它狭义的一面，指的是“传统机器学习”，往往与“深度学习”区别开来。

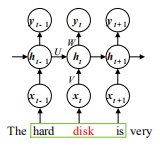
在这里，你列出来的这些个方法，实际上都是属于“传统机器学习”领域的，所以你在第一点里提到的“存在问题”基本上是普遍存在于这类方法中的。另外，1和3是重点要去看的，1比较难，可以先看3（hmm，隐马尔科夫模型）。

**模型**

**1、simple RNNs**

**特点：适用于基于时间的序列，如语音识别、文章分析等需要结合上下文的学习**

**(1)、Elman-type RNN模型**



x(t)代表在序列索引号t时训练样本的输入。

h(t)代表在序列索引号t时模型的隐藏单元。

y(t)代表在序列索引号t时训练样本序列的输出(可以有任意数量的输出)。

U,W,V这三个矩阵是模型每一层及每个神经元之间的权值，在整个RNN网络中是共享且相同的，神经网络模型通过训练“学“到的东西就蕴含在“权值“中。

**模型运行步骤：**

**第一步：**

为模型随机分配权重(weight matrices)和偏移量(bias)

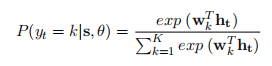
**第二步：**

开始训练，将词序列导入输入层，x(t)进入隐藏层，并结合h(t-1)，在隐藏层内按如下公式进行计算，其中U,V为权值(weight matrices)，b为偏移量(bias)。

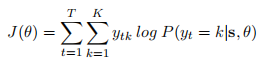
其中f为激活函数（activation function，如sigmoid，tanh）是用来扭曲坐标系空间，方便进行线性的分割，从而区分出正确答案。

**第三步：**

隐藏层结果h(t)和权值W输入到输出层y(t)，在输出层中运用softmax函数得出最后答案。

softmax函数：用于多分类过程，它将多个神经元的输出，映射到（0,1）区间内，通过类似概论的方法进行多分类。

**第四步：**

运用BPTT(backpropagation through time)方法，并结合cross-entropy损失函数训练模型参数，使模型预测值与目标值的损失达到最小，则训练成功。其中损失函数：

**\*simple RNN的训练方法——SGD(stochastic gradient descent)加BPTT**

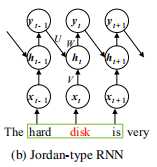
BPTT（back-propagation through time）算法是常用的训练RNN的方法，其实本质还是BP算法，只不过RNN处理时间序列数据，所以要基于时间反向传播，故叫随时间反向传播。BPTT的中心思想和BP算法相同，沿着需要优化的参数的负梯度方向不断寻找更优的点直至收敛。

在偏导求解过程中会有累乘运算，累乘会导致激活函数导数的累乘，因为激活函数的导数位于1,0之间，所以会导致“梯度消失“和“梯度爆炸“现象的发生。

可以通过缩短BPTT所包含的步数来解决，但这样就导致在处理大范围的文字的时候失去RNN的时间记忆功能，可以用LSTM解决

看起来，你理解得还是挺不错的

**(2)、Jordan-type RNN：**

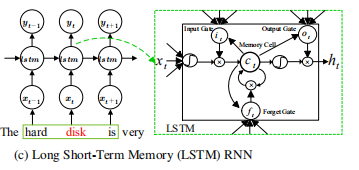


其他与Elman-type RNN相同，只是隐藏层变化了：

主要是隐层参数传递方式发生了改变，事实上，用得多的还是第一种Elman-type RNN（通常说rnn，都是指这个）

**2、LSTM (Long Short-Term Memory RNN)**

**特点：LSTM网络通过精妙的门控制将短期记忆与长期记忆结合起来，并且一定程度上解决了梯度消失的问题，可以学习长期依赖信息。**



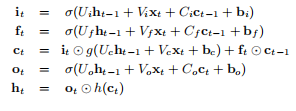
c：记忆细胞

i：输入门

o：输出门

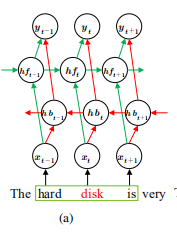
f：遗忘门

这些隐藏层中嵌入的函数决定了记忆细胞中的记忆如何更新：



**3、Bidirectionality RNNs**

**特点：当前的输出y(t)不仅仅与前面的序列hf(t-1)有关，并且还与后面的序列hb(t-1)有关。预测一个语句中缺失的词语那么就需要根据上下文来进行预测。Bidirectional RNNs是由两个RNNs上下叠加在一起组成的。输出由这两个RNNs的隐藏层的状态决定的。**



隐藏层由两个子隐藏层决定：



关于模型方面有什么不懂的，可以看看这个订阅号，是赵妍妍老师那个实验室的学生们在运营的

https://mp.weixin.qq.com/s/Kr9jnNuYJB0zmfhRQZM2gg

**实验结果分析**

1、Word embedding对处理性能有显著的作用；

2、相比于CRF，RNN有更好的自动学习的特性，并且可以更加智能的处理标签的依赖关系；

3、大多是时候，LSTM和双向RNN会增加RNN中的参数数量，导致一些任务参数配置过度，性能低于ElmanRNN

4、加入语言特征(Google Embeddings)可以补充word embedding，增加性能

5、微调(fine-tuning)可以使word embedding的嵌入效果达到最好