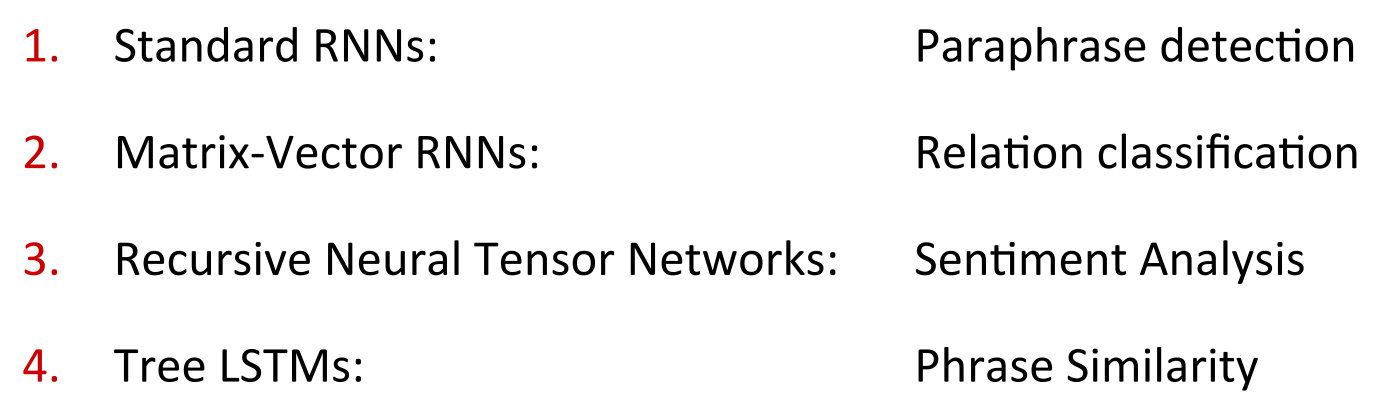
## Advanced Recursive Neural Networks

本章主要介绍四个模型



现分别叙述如下

## Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection

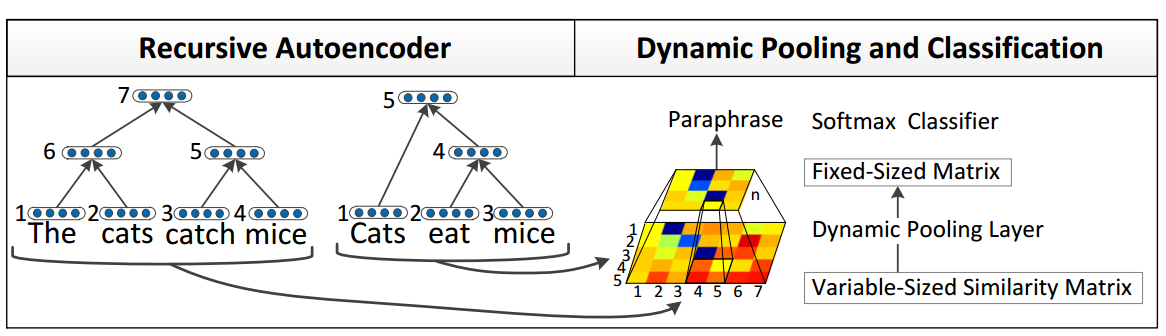
该模型主要用来检测俩个句子之间的释义的相似程度。

比如对于以下两个句子，它们释义的相似程度有多高

S1 The judge also refused to postpone the trial date of Sept. 29

S2 Obus also denied a defense motion to postpone the September trial date

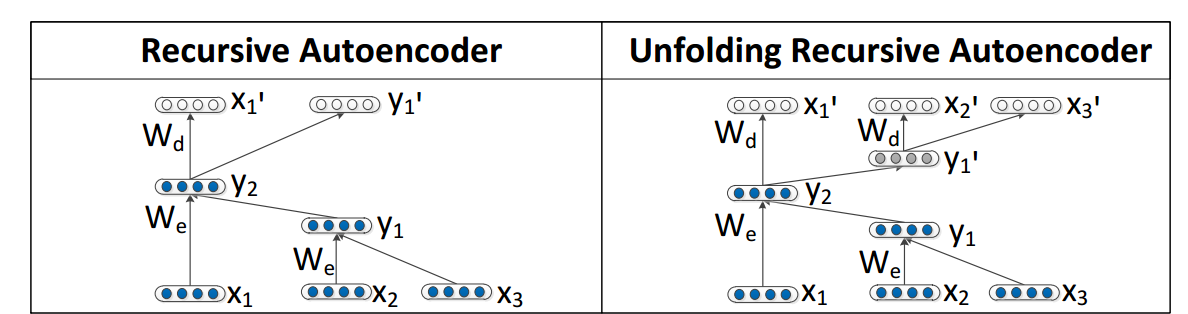
先来一张模型的整体结够图



从图中可以看出，模型分成两个部分，左侧为递归自编码部分，右侧的为动态池化和分类部分。

### **Recursive Autoencoders**

先对自编码（RAE）部分进行介绍，有两种方案进行自编码，网络结构分别如下



首先，我们因该思考一个问题，将一个句子编码之后，我们该如何评价这个编码的好坏？一个简单的想法就是将编码后的句子再进行解码，将解码后的结果与对应的输入向量进行比较，它们的差值越小，那么编码的结果也就越可靠。因此上图中两种自变码模型只是在解码过程中有所不同，左侧图片中，只解码到了根节点的直接子节点，右侧模型中，则解码到所有的叶子节点。到这里，你是不是还有一点疑问呢？这棵语法树是如何建立的？其实这棵语法树是通过一个语法解析器建立的（例如RNN，PCFG等），在有了这颗语法树之后，就能开展接下来的工作了。下面先以左图位列对该模型进行说明：

给定一个句子的词向量序列，树中的任意一个分支都可以表示成一个三元组每一孩子节点要么是一个词向量xi，要么是一个非终端结点。上图中的树可以表示如下



拥有树的结够后，我们可以定义生成父节点的运算



[c1;c2]是孩子节点向量c1，c2的串接，是一个2n维的列向量，

在解码过程中，使用下式



Erec（p）是重构误差，采用欧式距离计算，通过等式（2）我们可以先通过y2计算出得到通过类似的运算，我们能得到，在非展开的模型中，给定一棵树我们用所有非终端节点的重构误差的和作为总的重构误差



在本例中

这里需要注意的一点就是，如果在正向训练过程，我们将所有非终端结点压缩成0向量，可以想象，如果解码过程也将所有向量压缩成0向量自然会得到一个最小的重构误差，但这并不是我们想要的。因此，在编码时，我们要对每个非终端结点进行标准化。

### Unfolding Recursive Autoencoder

说完了非展开的自编码（标准RAE），再来阐述展开式的自编码就好说多了，在展开式的自编码中，正向的编码过程与标准RAE相同，所不同的是UARE对重构误差做了新的定义，在计算一个包含了从xi到xj的子树的重构误差时，不在使用它的直接子节点，而是用它的全部叶子节点去计算重构误差：

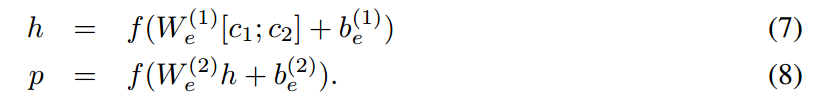


重构的叶子的计算方式与标准RAE相同。

与标准RAE相比，URAE不存在压缩非终端结点时所产生的问题，同时，对于标准RAE来说，树根的两个子节点一个可能是一个很庞大的子树，该子树包含了由子树构成的短语的意思，而另一个子节点可能是一个词向量，显然，两者在语义丰富程度上不对等的。但是，在反向传播时在公式中所占位置确实相同的，因此，训练出来的We可能会趋向于对这两者的意义做平均分配。而URAE是通过所有叶子节点进行计算误差的，并不存在这个问题。

### Deep Recursive Autoencoder

为了使网络更深一层，可以对每一个节点的输出向量再添加一个影藏层，也就是使用如下两个式子替换等式（1）



## An Architecture for Variable-Sized Similarity Matrices

现在完成模型的无监督学习部分。接下来就是计算俩个句子之间相似性了

### Computing Sentence Similarity Matrices

构建一个矩阵S来捕获俩个句子之间的相似程度，对于两个长度分别为n和m的句子，矩阵S的维度是（2n-1）\*(2m-1)，首先将第一个句子按原始的句子中词的的顺序排成一列，之后再将所有第一个句子的解析树中的节点排在第一个句子的词之后，用类似的方法将第二个句子排成一行，矩阵中的i，j元就是相对应的欧式距离。

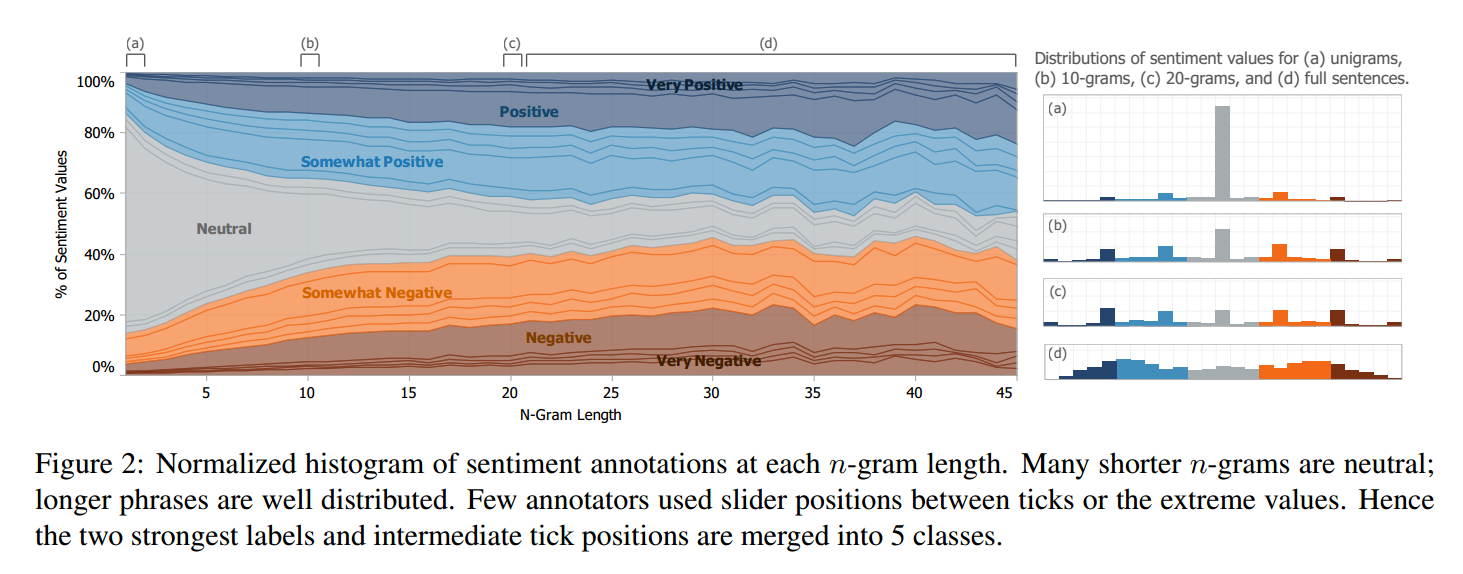
### Dynamic Pooling

思考这样一个问题，不同长度的两个句子生成的S矩阵的大小也不一样，因此，难以直接将其喂食到网络之中。

因此，提出了动态池化的概念，我们期望得到一个固定大小的（）矩阵，因此每次在池化时要动态的改变每一个池化范围。

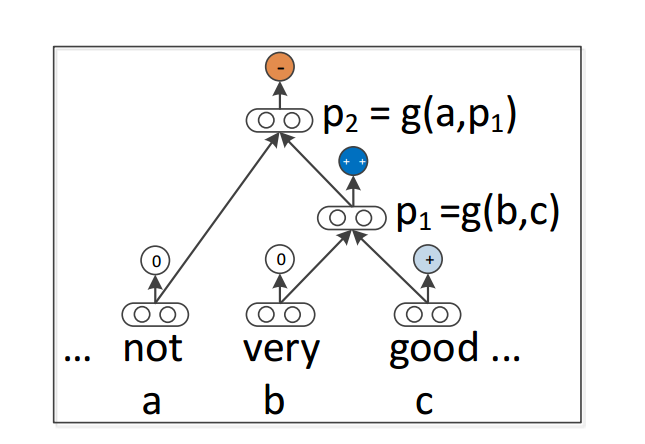
## 情感分析

情感分析是指分析一个句子的情感表达强度，一般将情绪由小鸡到积极划分为多个等级，情感分析的任务就是将一个句子划分到一个正确的情感等级，在介绍情感分析的模型之前，先简单介绍一下Stanford Sentiment Treebank，这是一个已经生成好的树库，原始的Treebank包含了10662个句子，这些句子中一半的感情强度可以归属为消极，一半的感情强度可以归属为积极。在此基础上，通过斯坦福的解析器将10662个句子解析成了215154个更小的短语，解析后的句子是一个解析树。之后，在这些短语上进行情感等级标注。下图是一个情感等级分布表



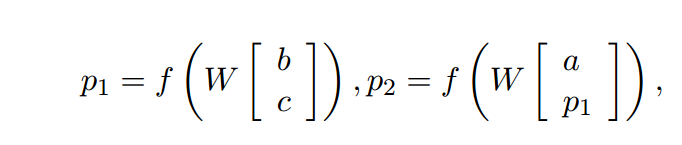
### RNN

在传统RNN模型之上，每一个节点额外输出一个softmax分类，就构成了最简单的RNN模型，等式（1）描述了该输出





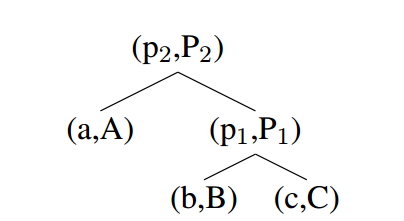
上式中,C是情感强度的总类数，d是词向量的维度。



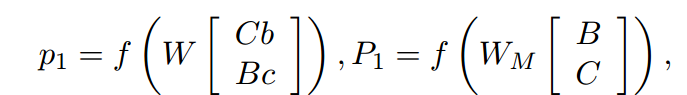
需要注意的是，在情感强度分析中，我们连同词向量一同进行训练，初始时词向量置为，r=0.0001的随机值

### MV-RNN: Matrix-Vector RNN

正常的语句中，每对词之间应该都包含一些交互信息，但传统的RNN网络只是将两个词串接一起乘以了一个权值矩阵，缺少更进一步的交互。在MV-RNN模型中，一个词以（词矩阵，词向量）的形式表现。词矩阵的初值为单位阵加一点高斯噪音



在计算一个父节点时，通过交叉的此矩阵与词向量之间的乘积作为串接对象，具体形式如下

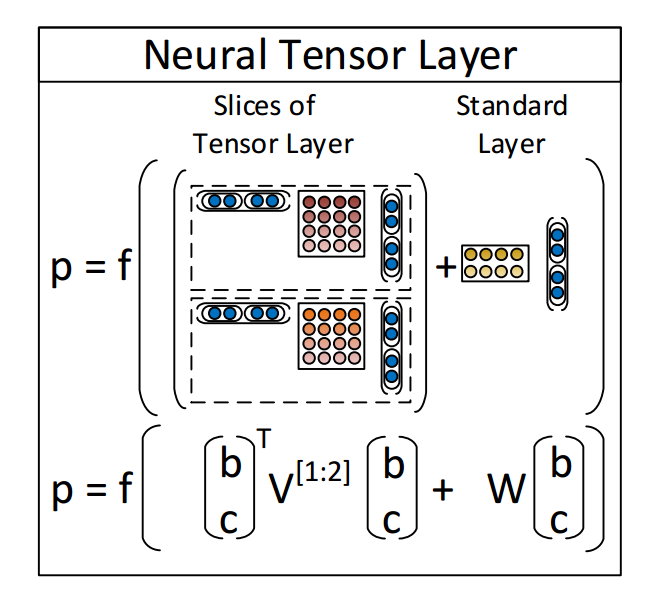


式中，，。

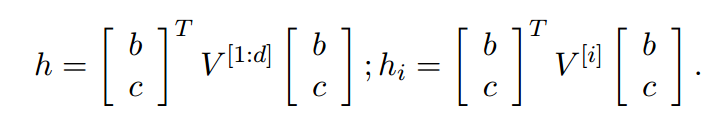
### RNTN:Recursive Neural Tensor Network

MV-RNN中的一个问题就是参数过于庞大，并且依赖于词汇表的大小。因此，提出一个新的基于张量的模型。

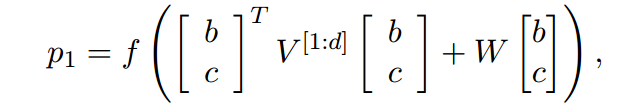
在RNTN模型中，新增一个三维张量，张量的第k片记为，整个张量用表示。张量模型可以通过下图来表示

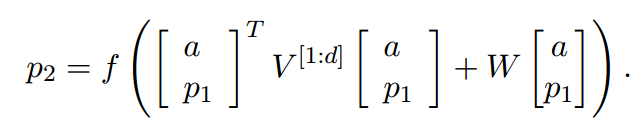


其中关于张量的预算可以从下式中理解

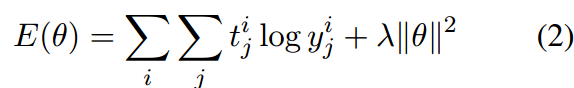


，p1，p2的计算公式这样定义



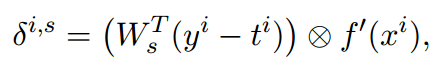


现在，根据Treebank中的数据信息，我们已经有了每个节点情感强度的真实分类，是一个one –hot向量。没一个节点都会输出一个预测分类。因此，节点都可以写一个交叉熵。



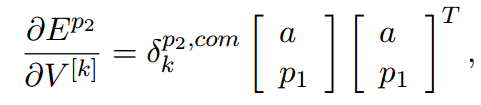
是第i个样本的y节点输出。是对应该节点的真实分类。是改模型的参数。

定义，是各层激活函数的输入（注意这里与原文的定义不一致，目前看来应该是原文有问题），即, 是本层交叉熵对的梯度，则

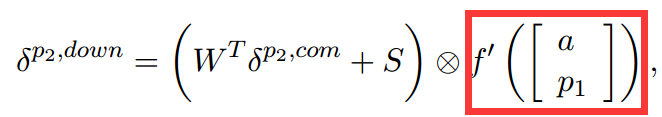


定义是对本层激活函数的输入的完全的梯度，即它包含来自本层的交叉熵梯度和高层传递下来的梯度

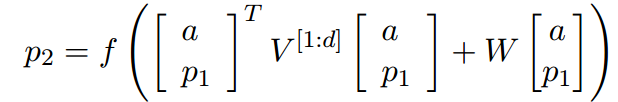
对于P2节点来说，它不存在来自高层的梯度。因此，，在p2节点处对的梯度由以下公式给出



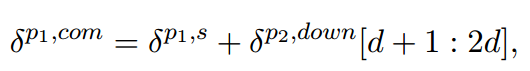
从节点p2传至其子节点的误差由下式定义



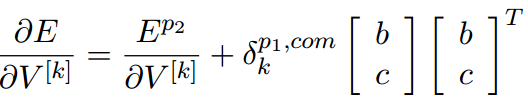
其中，S定义如下



需要注意的是，红框中的应该替换为该节点a，和节点p1的激活层输入，但a其实是没有激活层的，红框内的内容应该改成，（1…1）表示对a为线性激活函数的求导结果，表示p1的激活层输入。实际求导至子节点的激活层输入。此时，由两部分组成，来自p2的误差和本层的交叉熵，他们之间是和的关系



表示取得该梯度的后d个元素组成的向量。的总的梯度为



权值矩阵W的梯度求解方式与的类似，不再叙述。