自然语言处理系列（8）：RCNN

这篇文章翻译自Recurrent Convolutional Neural Networks for Text Classification

文本分类是许多NLP应用的基础任务。传统的文本分类器通常依赖于许多人设计的特性，如字典、知识库和特殊的树内核。与传统的方法相比，我们引入了一个递归的卷积神经网络来进行文本分类，而没有人为设计的特征。在我们的模型中，我们应用了一个经常的结构，在学习单词表示法时尽可能地捕捉上下文信息，这可能大大减少了与传统的基于窗口的神经网络相比的噪音。我们还使用了一个自动判断哪些单词在文本分类中扮演关键角色，以在文本中捕获关键组件的方法。我们对四个常用的数据集进行实验。实验结果表明，该方法在多组数据集上优于最先进的方法。

一、Introduction

文本分类是许多应用的重要组成部分，如web搜索、信息过滤和情绪分析(Aggarwal和Zhai 2012)。因此，它引起了许多研究者的关注。

文本分类中的一个关键问题是特征表示，这通常是基于单词bagof -words (BoW)模型，其中的unigrams、bigrams、n-grams或一些精心设计的模式通常被提取为特征。此外，还应用了频率、MI (Cover and Thomas 2012)、pLSA (Cai and Hofmann 2003)、LDA (Hingmire et al. 2013)等几种特征选择方法，以选择更具鉴别性的特征。

然而，传统的特征表示方法往往忽略文本中的上下文信息或词序，对于捕捉词的语义仍然不满意。例如，在句中，“沿着南岸的夕阳漫步提供了一系列令人惊叹的优势。”当我们分析“Bank”(unigram)这个词时，我们可能不知道它是指金融机构还是河旁。此外，“South Bank”(bigram)，尤其是考虑到两个大写字母，可能会误导那些对伦敦不太了解的人，把它当作金融机构。当我们获得更大的上下文“沿着南岸漫步”(5-gram)，我们就能很容易地辨别出它的意思。虽然高阶n-grams和更复杂的特性(如树内核(Post和Bergsma 2013))被设计用于捕获更多的上下文信息和单词序列，但它们仍然存在数据稀疏问题，这严重影响了分类的准确性。近年来，经过预先训练的word embedding和深层神经网络的快速发展，给各种NLP任务带来了新的启发。word embedding是单词的一种分布式表示，极大地缓解了数据稀疏问题(Bengio et al. 2003)。Mikolov、Yih和Zweig(2013)表明，预先训练的词嵌入可以捕捉有意义的句法和语义规律性。在word embedding的帮助下，人们提出了一些基于合成的方法来获取文本的语义表示。

Socher et al .(2011 a;2011 b;2013年提出递归神经网络(RecursiveNN)，在构建句子表示方面已被证明是有效的。然而，递归通过树结构捕获了一个句子的语义。它的性能很大程度上取决于文本树结构的性能。此外，构造这样一种文本树的时间复杂度至少为$O(n^2)$，其中n为文本的长度。当模型遇到长句或文档时，这将是非常耗时的。此外，两个句子之间的关系很难用树结构来表

示。因此，递归不适合对长句或文档建模。

另一种模型，循环神经网络（RNN），模型时间复杂度为$O(n)$。该模型通过逐字分析一个文本单词，并将所有先前文本的语义存储在一个固定大小的隐藏层中(Elman 1990)。RNN的优点是能够更好地捕捉上下文信息。这可能有利于捕获长文本的语义。然而，RNN是一个有偏倚的模型，在这个模型中，后面的单词比先前的单词更具优势。因此，当它被用于捕获整个文档的语义时，它可能会降低效率，因为关键组件可能出现在文档中的任何地方，而不是最后。为了解决偏置问题，我们引入了卷积神经网络(CNN)，将一个不带偏见的模型引入到NLP任务中，它可以很好地确定文本中带有最大池化层的识别性短语。因此，与递归或循环神经网络相比，CNN可以更好地捕捉文本的语义。CNN的时间复杂度也是$O(n)$。然而，以前对CNNs的研究倾向于使用简单的卷积核，如固定窗(bert et al. 2011;Kalchbrenner和Blunsom 2013)。使用这样的内核时，很难确定窗口大小:小窗口大小可能导致一些关键信息的丢失，而大的窗口会导致巨大的参数空间(这可能很难训练)。因此，它提出了一个问题:我们能否比传统的基于窗口的神经网络学习更多的上下文信息，更准确地表示文本的语义。

为了解决上述模型的局限性，我们提出了一个循环卷积神经网络(RCNN)，并将其应用于文本分类的任务。首先，我们应用一个双向的循环结构，与传统的基于窗口的神经网络相比，它可以大大减少噪声，从而最大程度地捕捉上下文信息。此外，该模型在学习文本表示时可以保留更大范围的词序。其次，我们使用了一个可以自动判断哪些特性在文本分类中扮演关键角色的池化层，以捕获文本中的关键组件。我们的模型结合了RNN的结构和最大池化层，利用了循环神经模型和卷积神经模型的优点。此外，我们的模型显示了$O(n)$的时间复杂度，它与文本长度的长度是线性相关的。

我们用英语和汉语四种不同的任务来比较我们的模型和以前的最先进的方法。类别分类包含主题分类,情感分类和写作风格分类。实验表明，我们的模型在四种常用数据集的三种情况下比以往的先进方法更出色。

二、Related Work

2.1 Text Classification

传统的文本分类工作主要集中在三个主题:特征工程、特征选择和不同类型的机器学习算法。对于特征工程来说，最广泛使用的功能是bag-of-words。此外，还设计了一些更复杂的功能，比如词性标签、名词短语(Lewis 1992)和tree kernels (Post and Bergsma 2013)。特征选择旨在删除噪声特征，提高分类性能。最常见的特征选择方法是去停词(例如，“The”)。先进的方法使用信息增益、相互信息(Cover和Thomas 2012)或L1正则化(Ng 2004)以选择有用的特性。机器学习算法常用分类器，如逻辑回归(LR)、朴素贝叶斯(NB)和支持向量机(SVM)。然而，这些方法都存在数据稀疏问题。

2.2 Deep neural networks

最近，深度神经网络(Hinton and Salakhutdinov 2006)和表示学习(Bengio, Courville，和Vincent 2013)提出了解决数据问题的新思路，并提出了许多学习单词表示的神经模型(Bengio et al. 2003;Mnih and Hinton 2007;Mikolov 2012;Collobert et al . 2011;huanget al . 2012;Mikolov et al . 2013)。一个单词的神经表示被称为单词嵌入，它是一个实值向量。“嵌入”一词使我们能够简单地利用两个嵌入向量之间的距离来测量单词的相关性。通过预先训练的词嵌入，神经网络在许多NLP任务中表现出色。Socher等人(2011b)使用半监督递归自动编码器来预测句子的情绪。Socher等(2011a)提出了一种用循环神经网络进行语义检测的方法。Socher等人(2013)引入递归神经张量网络分析短语和句子的情绪。Mikolov(2012)利用循环神经网络构建语言模型。Kalchbrenner和Blunsom(2013)提出了一个新的对话行为分类网络。conbert等人(2011)引入了卷积神经网络的语义角色标记。

三、Model

我们提出了一个深度神经模型来捕获文本的语义。图1显示了我们模型的网络结构。网络的输入是一个文档D，它是一个序列的单词$w_1, w_2 \ldots w_n$。网络的输出为类别。我们使用$p(k \mid D\ ,\ \theta)$来表示文档类别k的概率,θ是网络参数。

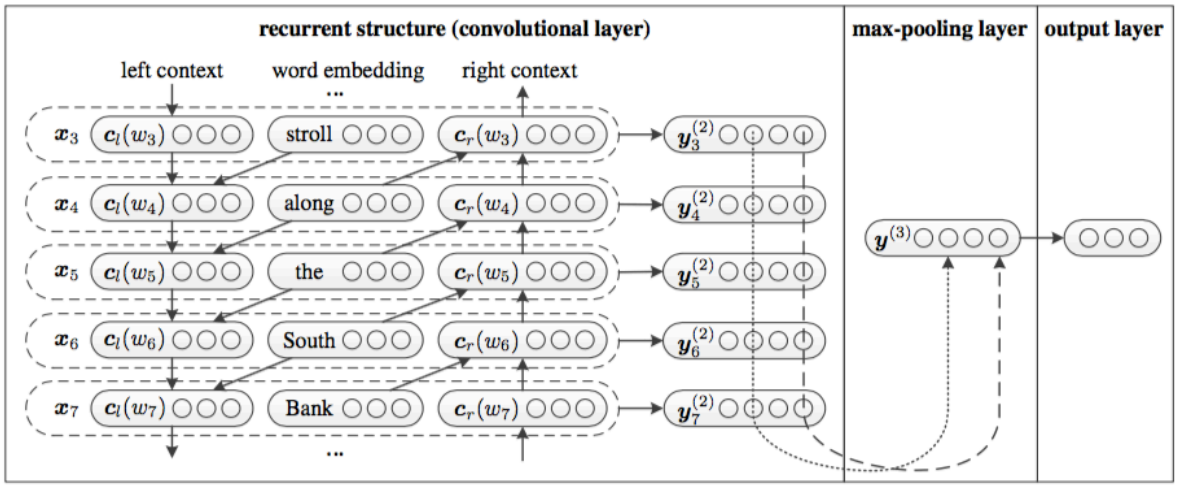

Figure 1: The structure of the recurrent convolutional neural network. This figure is a partial example of the sentence “A sunset stroll along the South Bank affords an array of stunning vantage points”, and the subscript denotes the position of the corresponding word in the original sentence.

3.1 Word Representation Learning

我们结合一个词和它的上下文来呈现一个词。语境帮助我们获得更准确的词义。在我们的模型中，我们使用一个循环结构，它是一个双向的循环神经网络，用来捕获上下文。我们将$c_l(w_i)$定义为词w_i左边的文本，将$c_r(w_i)$定义为词w_i右边的文本。$c_l(w_i)$和$c_r(w_i)$都是具有|c|个实值元素的稠密向量。使用式（1）来计算词w_i左边的文本，$e(w_{i-1})$是词w_{i-1}的词嵌入，它是一个长度为$|e|$的实值向量。$c_l(w_{i-1})$是上一个词w_{i-1}的左半部分文本。任何文档的第一个词的左半边文本使用相

同的参数$c_l(w_1)$。$W^{(l)}$是一个将隐藏层(context)转换为下一个隐藏层的矩阵。$W^{(sl)}$是一个矩阵，用来将当前单词的语义与下一个单词的左上下文结合起来。f是非线性激活函数。右半边文本$c_r(w_i)$用相同的方式计算，如公式（2）所示。一个文档的最后一个词的右半边文本共享参数$c_r(w_n)$

$$c_l(w_i) = f(W^{(l)}c_l(w_{i-1}) + W^{(sl)}e(w_{i-1})) \quad (1) c_r(w_i) = f(W^{(r)}c_r(w_{i+1}) + W^{(sr)}e(w_{i+1})) \quad (2)$$

如式(1)和(2)所示，上下文向量捕获了所有左和右上下文的语义。例如，在图1中，$c_l(w_7)$编码了左侧上下文的语义“沿着南方漫步”以及前面的所有文本，$c_r(w_7)$编码了右侧上下文的语义“提供了一个……””。然后，我们定义单词w_i的表示形式为式(3)，即左侧上下文向量$c_l(w_i)$，词嵌入表示$e(w_i)$和右侧上下文向量$c_r(w_i)$的连接。用这种方式，我们的模型可以更好地消除“w_i”这个词的含糊含义，而不是只使用固定窗口的传统神经模型，（例如他们只使用关于文本的部分信息）

$$x_i = [c_l(w_i); e(w_i); c_r(w_i)] \quad (3)$$

循环结构可以在文本的向前扫描时获取所有的c_l，在反向扫描时获取所有的c_r。时间复杂度为$O(n)$。当我们获得了单词w_i的表示x_i后，我们将一个线性变换与tanh激活函数一起应用到x_i，并将结果传递到下一层。

$$y_i^{(2)} = tanh(W^{(2)}x_i + b^{(2)}) \quad (4)$$

y是一个潜在的语义向量，每一个语义因素都将被分析，以确定代表文本的最有用的因素。

3.2 Text Representation Learning

我们模型中的卷积神经网络是用来表示文本的。从卷积神经网络的角度来看，我们以前所提到的重复结构是卷积层。当计算所有单词的表示时，我们应用一个max-pooling层。

$$y^{(3)} = max_{i=1}^{n} y_i^{(2)} \quad (5)$$

max函数是一个按元素的函数。$y^{(3)}$的第k个元素是$y_i^{(2)}$的所有向量的第k个元素的最大值。池化层将不同长度的文本转换为固定长度的向量。通过使用池化层，我们可以在整个文本中捕获信息。还有其他类型的池层，比如平均池层(Collobert et al. 2011)。我们这里不使用平均池，因为这里只有几个单词和它们的组合对于捕获文档的含义非常有用。在文档中，最大池化层试图找到最重要的潜在语义因素。池化层接受循环结构的输出作为输入。池化层的时间复杂度为$O(n)$。整体模型是一个循环结构和一个最大池化层的级联，因此，我们的模型的时间复杂度仍然是$O(n)$。我们模型的最后一部分是一个输出层。与传统的神经网络相似，它被定义为

$$y^{(4)} = W^{(4)}y^{(3)} + b^{(4)} \qquad (6)$$

最后，将*softmax*函数应用于$y^{(4)}$。它可以把输出数字转换成概率。

$$p_i = \frac{exp(y_i^{(4)})}{\sum_{k=1}^{n} exp(y_k^{(4)})} \qquad (7)$$

四、Training

4.1 Training Network parameters

我们定义的所有训练参数为θ。

$$\theta = \{E, b^{(4)}, c_l(w_1), c_r(w_n), W^{(2)}, W^{(4)}, W^{(l)}, W^{(r)}, W^{(sl)}, W^{(sr)}\}$$

具体来说，参数$E \in R^{|e| \times |V|}$是词嵌入，偏差向量$b^{(2)} \in R^{|H|}$,$b^{(4)} \in R^{O}$，初始文本$c_l(w_1), c_r(w_n) \in R^{(H \times (|e|+2c))}$，转换矩阵$W^{(2)} \in R^{H \times (|e|+2|c|)}$,$W^{(4)} \in R^{O \times H}$,$W^{(4)} \in R^{(O \times H)}$,$W^{(l)}, W^{(r)} \in R^{|c| \times |c|}$,$W^{(sl)}, W^{(sr)} \in R^{|e| \times |c|}$，其中$|V|$是词库中词的数量，$H$是隐藏层的规格，$O$是文档类别数。

网络的训练目标是使得θ满足对数似然最大化：

$$\theta \rightarrow \sum_{D \in D} logp(class_D | D, \theta) \qquad (9)$$

其中D是训练文档集，$class_D$是文档D的真实类别。

我们使用随机梯度下降法(Bottou 1991)来优化训练目标。在每一步中，我们随机选择一个样本(D, classD)进行一次梯度下降：

$$\theta \leftarrow \theta + \alpha \frac{\partial logp(class_D | D, \theta)}{\partial \theta} \qquad (10)$$

其中α是学习率。
在训练神经网络的训练过程中，我们采用了一种常用的方法来训练神经网络。我们将神经网络中的所有参数服从均匀分布初始化。最大或最小值的大小等于“fan-in”的平方根(Plaut和Hinton 1987)。数值是我们模型中前面一层的网络节点。该层的学习速率除以“fan-in”。

4.2 Pre-training Word Embedding

五、Experiments

5.1 Datasets

5.2 Experiment Settings

5.3 Comparison of Methods

5.4 Results and Discussion

5.5 Contextual Information

六、Conclusion