徽标

描述已自动生成

卡通画

描述已自动生成

**数智NLP组2022第二阶段考核报告**

**textCNN模型实现文本分类**

学 院 计算机学院

专 业 计算机

年级班别 2021级（6）班

学 号 3121004890

学生姓名 赵继业

2022年5月1日

目录

[一、数据预处理 2](#_Toc102242830)

[1.过滤无用信息 2](#_Toc102242831)

[2.subword算法 2](#_Toc102242832)

[二、网络构建 3](#_Toc102242833)

[1.构建原理 3](#_Toc102242834)

[1.1.词嵌入层（Embedding Layer 3](#_Toc102242835)

[2.网络搭建 6](#_Toc102242836)

[三、实验记录 8](#_Toc102242837)

[参考文献 15](#_Toc102242838)

### 一、数据预处理

1.过滤无用信息

在数据预处理中，需要过滤掉无用的数据，自然语言处理，无用的数据就是停用词(stop words)，数字以及标点等数据，因此，在预处理数据时，需要先将一些常见的停用词如人称，介词，疑问等，以及标点符号，数字去除掉

2.subword算法

与传统空格分隔tokenization技术的对比

* 传统词表示方法无法很好的处理未知或罕见的词汇（OOV问题）
* 传统词tokenization方法不利于模型学习词缀之间的关系
* Character embedding作为OOV的解决方法粒度太细
* Subword粒度在词与字符之间，能够较好的平衡OOV问题

BPE算法

1. 准备足够大的训练语料
2. 确定期望的subword词表大小
3. 将单词拆分为字符序列并在末尾添加后缀“ </ w>”，统计单词频率与字符的频率
4. 统计每一个连续字节对的出现频率，选择最高频者合并成新的subword
5. 重复第4步直到达到第2步设定的subword词表大小或下一个最高频的字节对出现频率为1

例如：

I like the delicious hamburger and do not like tomato

统计词频

‘I </w>’:1,’ l i k e </w>’:2, ’t h e </w>’, ‘d e l i c i o u s</w>’:1, ‘h a m b u r g e r</w>’:1,

‘a n d </w>’:1, ‘d o </w>’:1 ‘n o t </w>’:1, ‘t o m a t o </w>’:1

将每个每个以空格分隔的字符(即token)统计频率并且加入词表

统计每一个连续字节对的出现频率，此次频率最高的为e</w>,次数为2+1=3

将e</w>加入词表，并且e与</w>减去e</w>出现的次数

文本变为

‘I </w>’:1,’ l i k e</w>’:2, ’t h e</w>’, ‘d e l i c i o u s</w>’:1, ‘h a m b u r g e r</w>’:1,

‘a n d </w>’:1, ‘d o </w>’:1 ‘n o t </w>’:1, ‘t o m a t o </w>’:1

接着重复上面的步骤

在得到了subword的词表后，对该词表按照子词长度由大到小排序，每个字词对应一个索引。编码时，对于每个单词，遍历排好序的子词词表寻找是否有token是当前单词的子字符串，如果有，则该token是表示单词的tokens之一。

我们从最长的token迭代到最短的token，尝试将每个单词中的子字符串替换为token的索引。 最终将迭代所有tokens，并将所有子字符串替换为tokens。 如果仍然有子字符串没被替换但所有token都已迭代完毕，则将剩余的子词替换为0。最终得到由数字索引组成的序列。

二、网络构建

1.构建原理

1.1.词嵌入层（Embedding Layer**）**

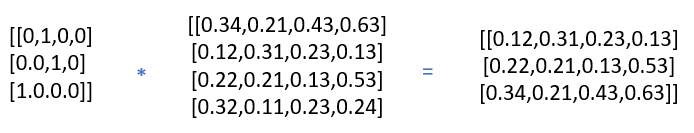
为什么使用embedding？

Embedding ,即词嵌入，Embedding 是一个将离散变量转为连续向量表示的一个方式。在神经网络中，它不光可以减少离散变量的空间维数，还可以表示不同词之间关系的远近。使用one-hot编码，每个词都能在一个一维数组中用0和1表示出来，表达特征的能力很强，但每个词之间的距离都相同，无法体现词与词之间的联系，并且当词语数量多时，维度会很高，这时，embedding的优势就体现出来了。

embedding的实现

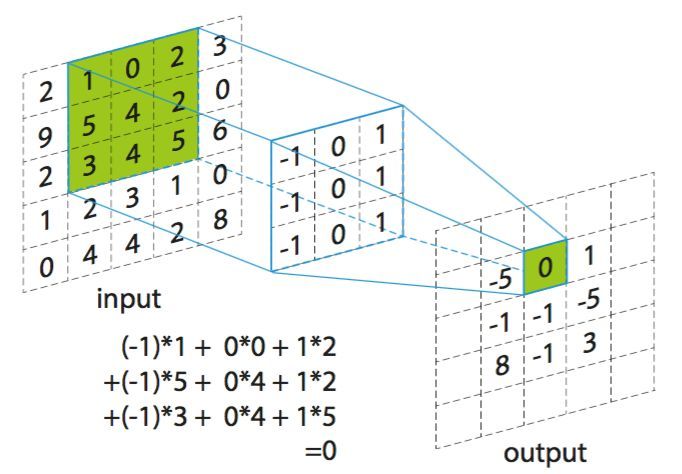
将一个one-hot表示的词转换为embedding表示实际上是用one-hot矩阵与一个权值矩阵相乘，也就是一个全连接层，这个权值矩阵是通过神经网络训练得来的

例如：



在keras的Embedding layer中，需要文本替换为对应token的索引，再输入

**1.2.卷积层(Convolutional Layer)**



卷积的过程

输入矩阵与卷积核对应数字的乘积之和为输出矩阵对应位置的值，步幅为卷积核在输入矩阵中每次计算滑动的距离。为了保持卷积后矩阵的尺寸与原来一致可以在输入矩阵周围补0一般情况下, 输入的图片矩阵以及后面的卷积核, 特征图矩阵都是方阵, 这里设输入矩阵大小为 w, 卷积核大小为 k, 步幅为 s, 补零层数为 p, 则卷积后产生的特征图大小计算公式为:



卷积的作用

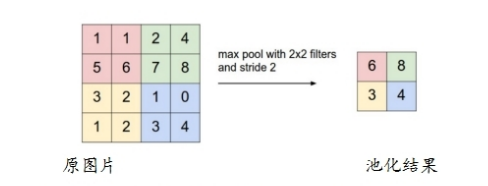
卷积可以提取输入数据的特征，并且其权值共享的特性大大地减少了神经网络的参数

**1.3.池化层（Pooling Layer）**

与卷积相似，如果是最大池化则是输出结果取对应区域中的最大值

池化层的作用

减小了下一层输入大小，进而减小计算量和参数个数，并且在一定程度上减少了过拟合



1.4.全连接层(Fully Connected Layer)

全连接层的每个单元可以理解为用一个尺寸与输入矩阵相同的卷积核与输入矩阵进行卷积，例如输入矩阵为5X5，全连接层有100个单元，则全连接层的输出为一个1X100的向量。

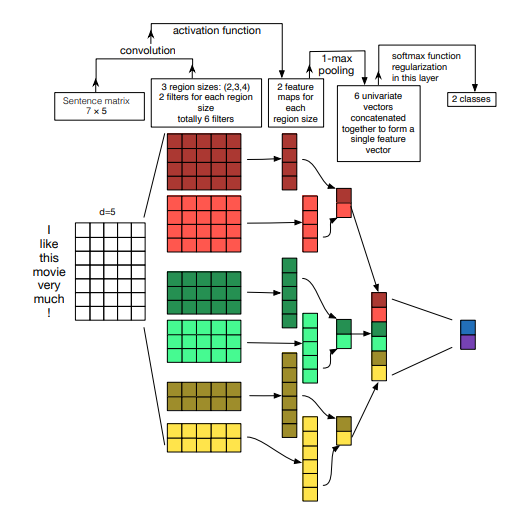
全连接层的作用

全连接层在整个网络卷积神经网络中起到“分类器”的作用。如果说卷积层、池化层和激活函数等操作是将原始数据映射到隐层特征空间的话（特征提取+选择的过程），全连接层则起到将学到的特征表示映射到样本的标记空间的作用。换句话说，就是把特征整合到一起（高度提纯特征），方便交给最后的分类器或者回归

**1.5.随机失活（drop out）**

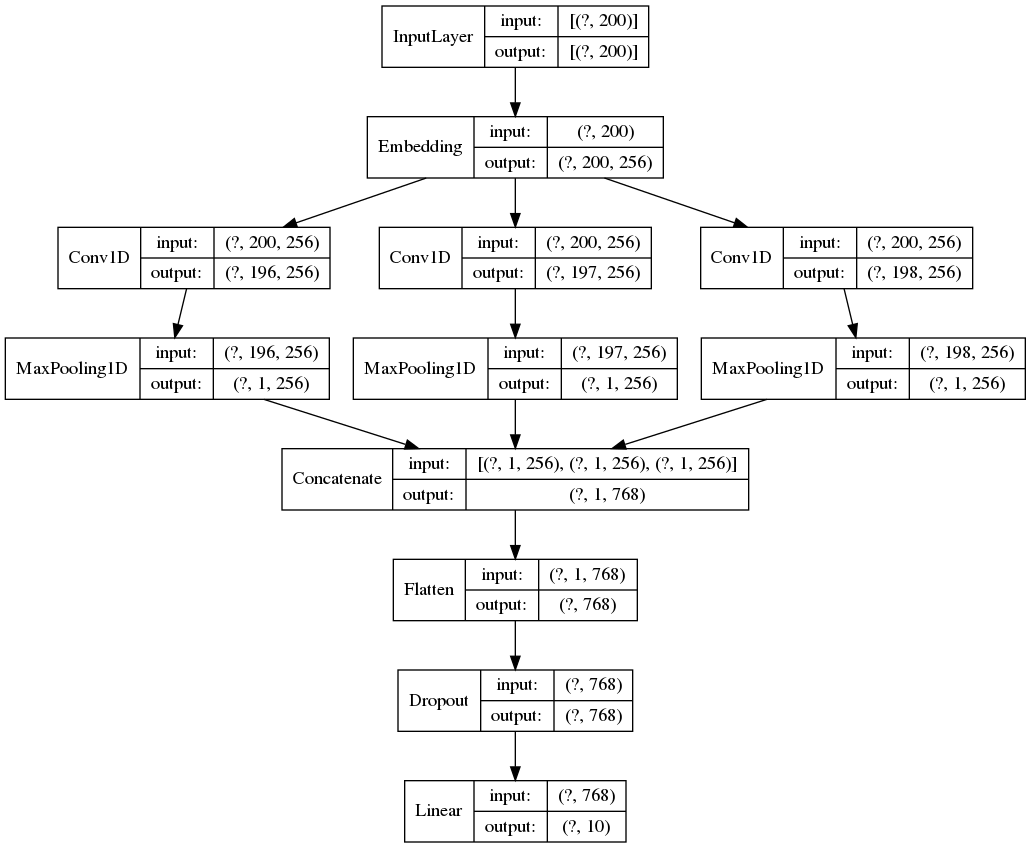
如果模型参数过多，而训练样本过少，容易陷入过拟合。随机失活t是通过遍历神经网络每一层的节点，然后通过对该层的神经网络设置一个节点保留概率p，即该层的节点有p概率被保留，概率的取值范围在0到1之间，设置为1则为全部保留，即不使用随机失活。通过设置神经网络该层节点的保留概率，使得神经网络不会去偏向于某一个节点，从而使得每一个节点的权重不会过大。

2.网络搭建



《A Sensitivity Analysis of (and Practitioner Guide to) Convolutional Neural Networks for Sent》中的模型结构

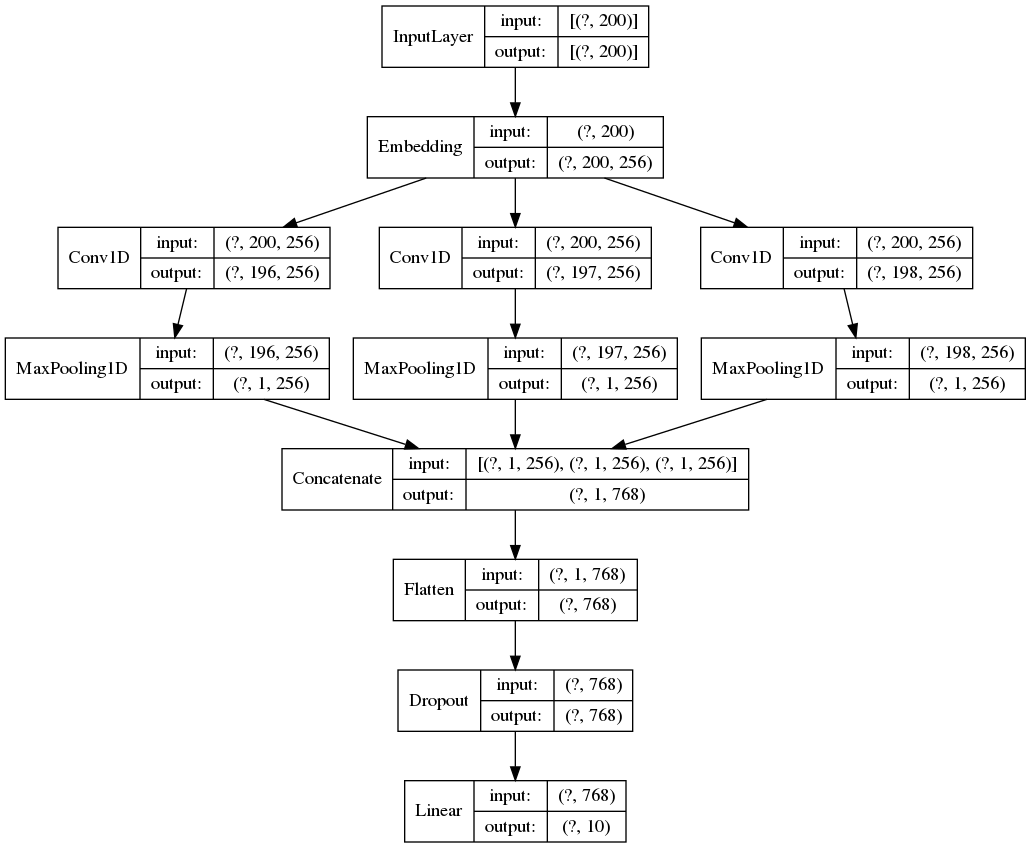
在论文中的textCNN模型中使用三种不同尺寸的卷积核进行1D卷积，得到多个列向量，不同于2D卷积，1D卷积从上往下卷，因为对相同位置上的词语进行卷积没有什么意义。经过最大池化层后，列向量变为一个数，拼接在一起变成一个列向量，输入全连接层进行分类，对全连接层中的权值矩阵进行L2正则化，以减少过拟合。



本次实验的文本经过预处理后，每个样本变为200维的向量输入神经网络，词嵌入层输出200X256的矩阵

卷积层同样使用三种不同尺寸的卷积核进行1D卷积，卷积和的尺寸分别为3，4，5，每种卷积核的数量为256，经过最大池化层后从最后一个维度进行拼接，得到1X768的矩阵，再展开变成一个向量，再进行随机失活，随机失活的概率设为0.4，即每个单元有0.4的概率被丢弃，最后经过一个全连接层进行分类

三、实验记录

词表大小：1500

词向量维度：256

卷积层激活函数：relu

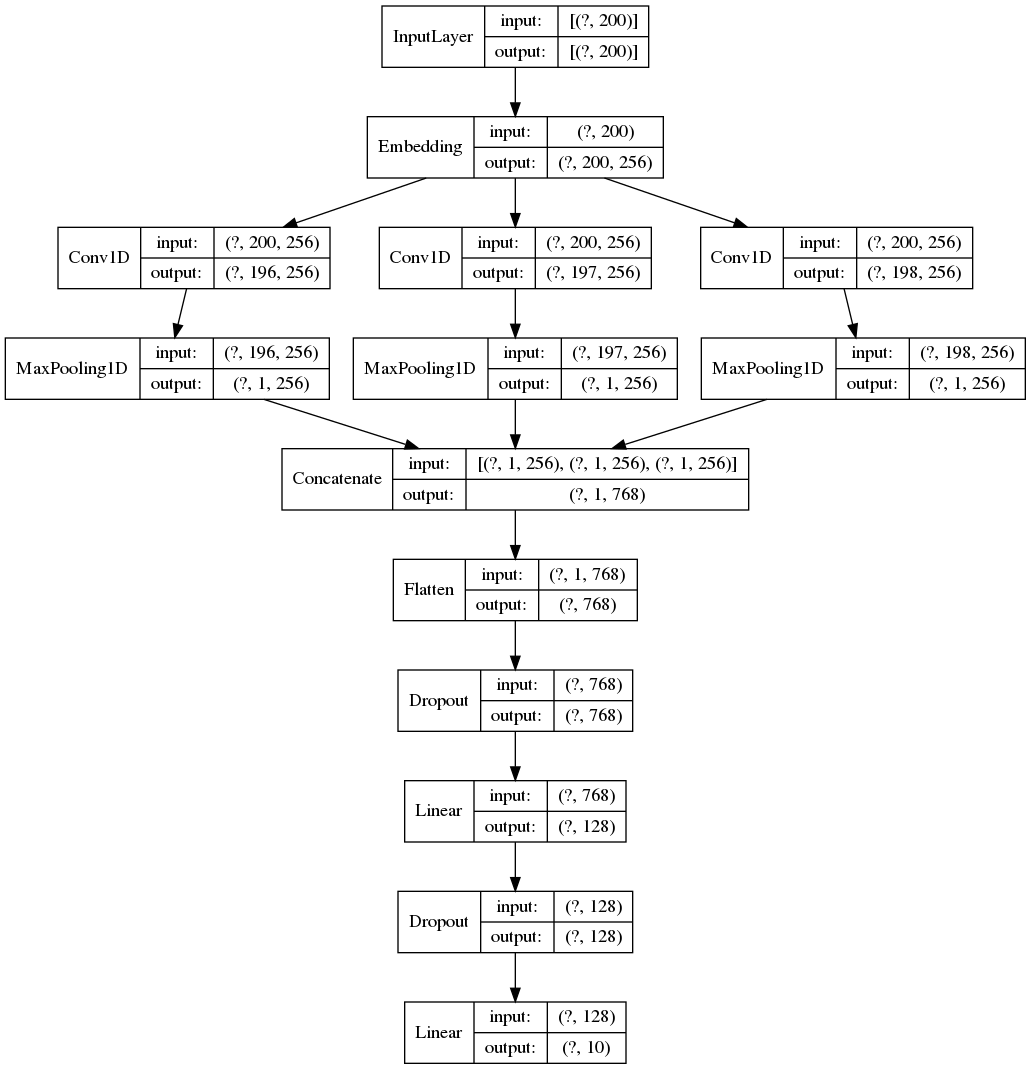
词数：200

dropout：0.4

全连接层l2：0.02

训练次数：10

准确率：0.49629



词表大小：1500

词向量维度：256

卷积层激活函数：relu

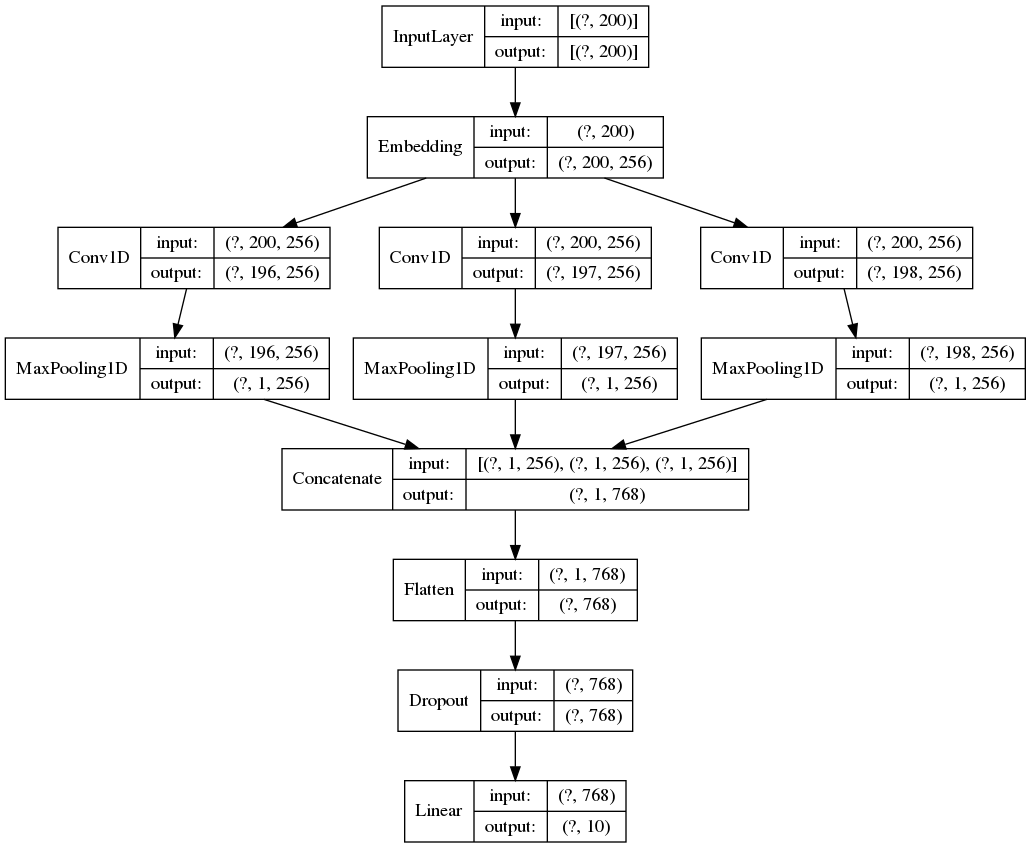
词数：200

dropout：0.4

全连接层l2：0.02

训练次数：10

准确率：0.48252

词表大小：1500

词向量维度：256

卷积层激活函数：relu

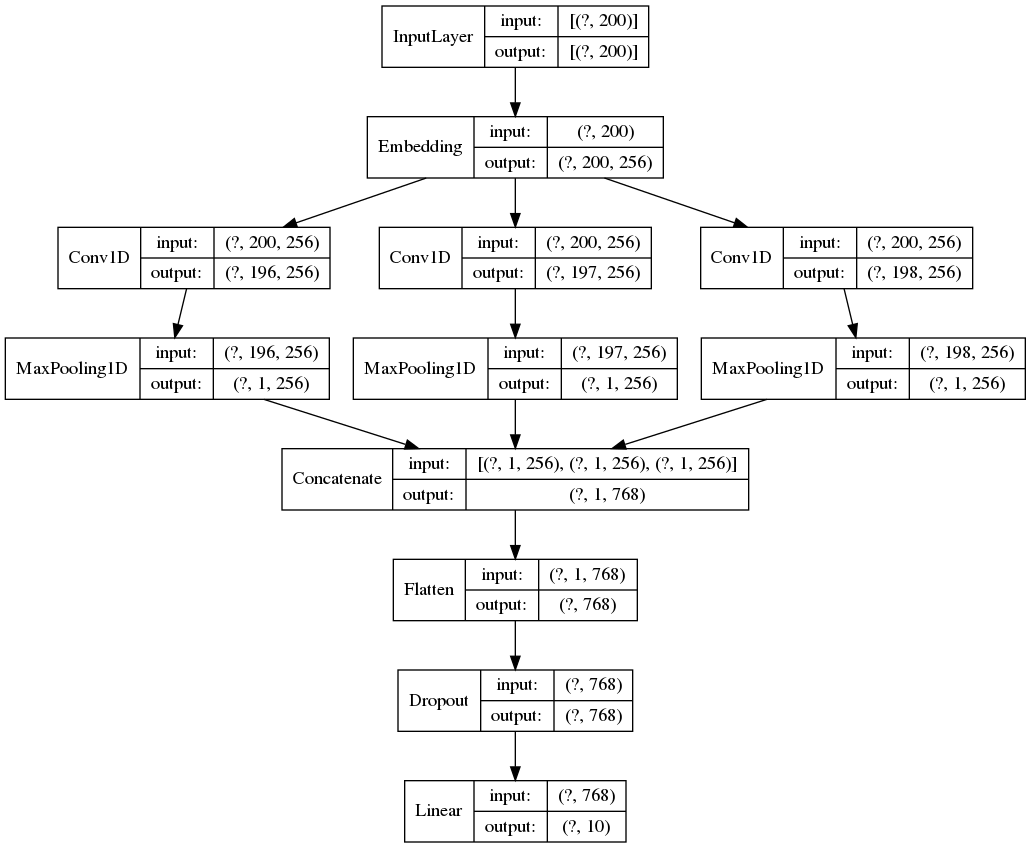
词数：200

dropout：0.4

全连接层l2：0.01

训练次数：10

准确率：0.4658

词表大小：1000

卷积层激活函数：relu

词向量维度：256

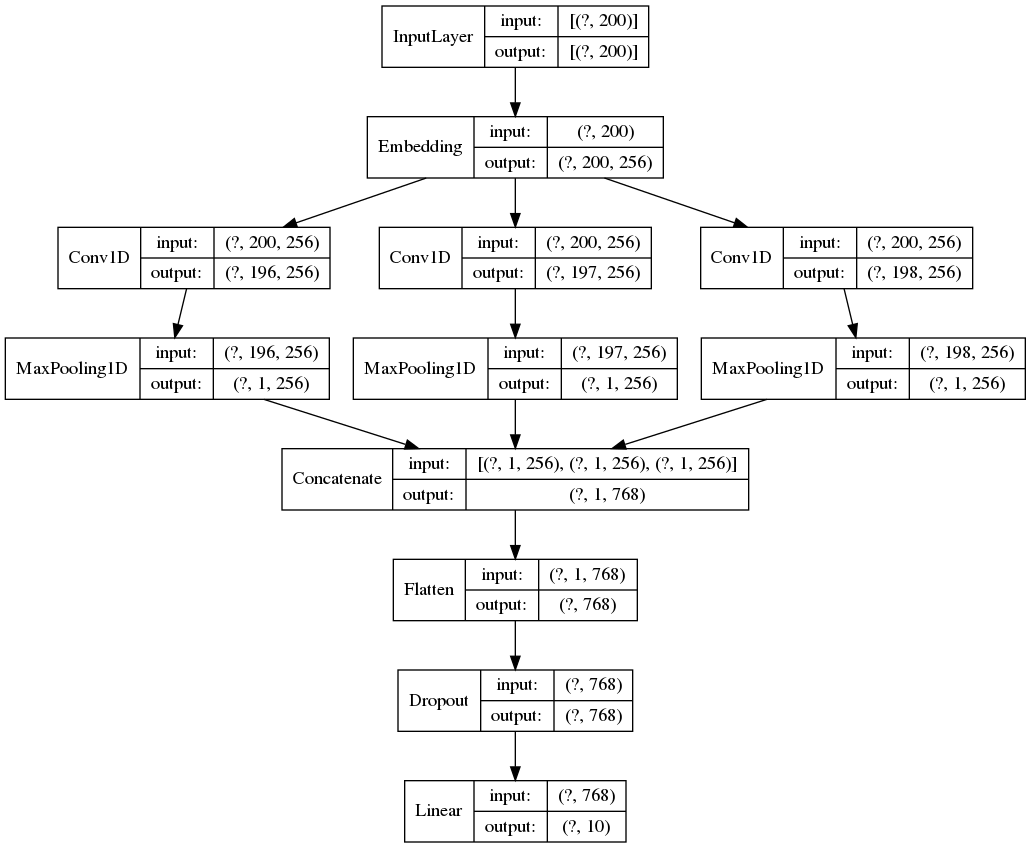
词数：200

dropout：0.4

全连接层l2：0.02

训练次数：10

准确率：0.50689

词表大小：1000

卷积层激活函数：relu

词向量维度：256

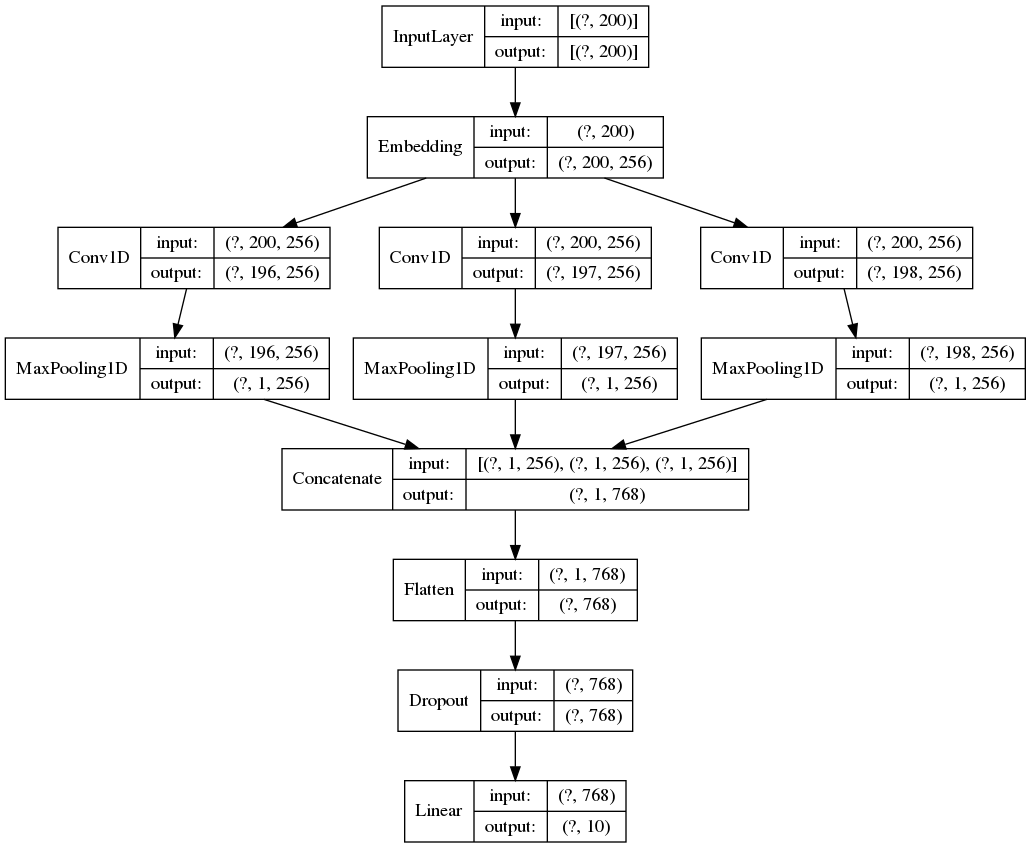
词数：200

dropout：0.4

全连接层l2：0.02

训练次数：5

准确率：0.49114

词表大小：1000

卷积层激活函数：tanh

词向量维度：256

词数：200

dropout：0.4

全连接层l2：0.02

训练次数：10

准确率：0.49053

# 参考文献

[1] [主页 - Keras 中文文档](https://keras.io/zh/)

[2] 《A Sensitivity Analysis of (and Practitioner Guide to) Convolutional Neural Networks for Sent》

[3] [深入理解NLP Subword算法：BPE、WordPiece、ULM - 知乎 (zhihu.com)](https://zhuanlan.zhihu.com/p/86965595)