**基于TextCNN网络的新闻文本分类**

本文主要通过**TextCNN**网络来对给定的**新闻文本**进行**分类**。TextCNN是一种用于文本分类的卷积神经网络，在自然语言处理领域应用广泛。它通过卷积神经网络提取文本特征，再将特征输入到全连接层进行分类。

在本文中，我们首先介绍数据预处理中的数据预处理中的**文本清洗**，**jieba分词**，**woed2vec词嵌入**。然后介绍TEXTCNN网络的结构以及原理，最终进行模型评估和参数调优。

关键词：TextCNN 文本清洗 jieba分词 word2vec 词嵌入

目录

[一、 深度学习的三种模型 3](#_Toc133881935)

[1. Textcnn 3](#_Toc133881936)

[2. RNN 3](#_Toc133881937)

[3. LSTM 3](#_Toc133881938)

[4. 三种模型的优缺以及联系 4](#_Toc133881939)

[二、 数据预处理 5](#_Toc133881940)

[1. 文本清洗 5](#_Toc133881941)

[2. jieba分词 5](#_Toc133881942)

[3. 词嵌入 6](#_Toc133881943)

[3.1 Word2Vec介绍 6](#_Toc133881944)

[三、 TEXTCNN网络的构建 7](#_Toc133881945)

[1. 卷积层 7](#_Toc133881946)

[2. 池化层 7](#_Toc133881947)

[3. Dropout层 7](#_Toc133881948)

[4. 全连接层 8](#_Toc133881949)

[5. 激活函数 8](#_Toc133881950)

[5.1 Relu函数 8](#_Toc133881951)

[四、 模型的评估与参数调优 9](#_Toc133881952)

[1. 句子长度对结果的影响 9](#_Toc133881953)

[2. 词语维度对结果的影响 10](#_Toc133881954)

[3. Dropout组合对结果的影响 10](#_Toc133881955)

[4. L2系数对结果的影响 11](#_Toc133881956)

[5. Epoch对结果的影响 11](#_Toc133881957)

[6. 学习率对结果的影响 12](#_Toc133881958)

[7. 不同优化器对结果的影响 13](#_Toc133881959)

[8. 是否使用动态学习率对结果的影响 13](#_Toc133881960)

[9. 是否使用dropout层 14](#_Toc133881961)

# 深度学习的三种模型

## Textcnn

TextCNN是一种用于文本分类的卷积神经网络，在自然语言处理领域应用广泛。它通过卷积神经网络提取文本特征，再将特征输入到全连接层进行分类。

具体来说，TextCNN首先将输入的文本序列通过embedding层将每个词转换为向量形式，然后使用不同大小的卷积核进行卷积，得到一系列特征图。接着对每个特征图进行max pooling操作，得到该特征图中最大值作为该特征的表示。最后将所有特征拼接起来，输入到全连接层进行分类。

TextCNN的优点在于其结构简单、易于实现和解释，并且在文本分类等任务上取得了较好的效果。同时，由于卷积神经网络可以捕获文本中的局部信息，因此TextCNN在文本分类等NLP任务中表现优异。

## RNN

循环神经网络（Recurrent Neural Network，简称RNN）是一种能够处理序列数据的神经网络，广泛用于自然语言处理、语音识别等领域。与传统的前馈神经网络不同，RNN具有循环结构，可以将历史时刻的信息通过循环传递到当前时刻。

RNN的基本结构由一个输入层、一个隐藏层和一个输出层组成。其中隐藏层上的神经元在时间上是连续的，每个神经元会接收到当前时刻的输入以及上个时刻的隐藏状态，并计算出当前时刻的隐藏状态。这样，在序列数据中，RNN可以通过隐藏层中的记忆单元来捕获历史信息。

RNN有许多变种，例如长短时记忆网络（LSTM）和门控循环单元（GRU），它们通过设计隐藏层的状态更新方式来提高网络的记忆能力和表达能力，进一步改进了RNN的性能。

## LSTM

长短时记忆网络（Long Short-Term Memory Network，简称LSTM）是一种深度学习模型，常用于处理时序数据。相较于传统的循环神经网络（RNN），LSTM有更强的记忆能力，能够有效地处理长时依赖问题。

LSTM的基本单元包含输入门、遗忘门、输出门和细胞状态。在每个时间步，输入门控制信息输入到细胞状态的程度，遗忘门控制细胞状态中需要忘记的信息，输出门控制从细胞状态到输出值的信息流。同时，LSTM中细胞状态可以选择性地记忆或忘记某些信息。

## 三种模型的优缺以及联系

TextCNN、RNN、LSTM是常用于自然语言处理领域的神经网络模型，它们在文本分类、情感分析等任务中取得了显著的效果。

TextCNN是一种卷积神经网络模型，与传统的RNN或LSTM相比，其输入不需要按照时间顺序，而是基于词嵌入向量进行卷积和池化。TextCNN通过卷积滤波器捕捉词汇间的局部相关性，可以快速地处理长文本，并且由于无需考虑时序关系，具有并行计算的优势。但是，它无法捕捉到文本中的长期依赖关系。

RNN是一种循环神经网络模型，通过控制权重共享，可以对变长序列数据进行建模。RNN可以处理任意长度的输入序列，并将历史信息传递到当前时刻，因此具有良好的记忆能力。然而，RNN容易出现梯度消失或梯度爆炸的问题，导致训练过程困难。

LSTM是对RNN的改进，通过引入门机制来控制细胞状态中的信息流。LSTM可以有效地解决梯度消失和梯度爆炸的问题，并且具有更强的记忆能力和更好的长期依赖建模能力。然而，LSTM的计算量和参数较大，较为耗时。

TextCNN、RNN和LSTM之间具有一定的联系，实际应用中也可以进行组合使用。例如，可以将TextCNN和LSTM结合起来，分别捕获局部和全局的特征信息，在处理长文本分类任务时取得较好的效果。另外，也可以使用注意力机制（如TextCNN with Attention）来进一步优化模型性能，使模型更关注重要的词汇信息，同时减少噪声影响。

以下重点介绍Textcnn的数据预处理，模型搭建以及模型的评估与参数调优。

# 数据预处理

## 文本清洗

文本清洗通常包括以下几个步骤：去除非字母字符、转换大小写、去除停用词、词干提取和标记化。其中，停用词是指在文本中频繁出现但通常不携带很多信息的词语，如“的”、“是”、“一”等。在textcnn中，去除停用词可以有效减少噪声和冗余信息，从而提高模型的性能。

此外，还可以对文本进行词干提取（stemming），即将不同形态的词语转换为它们的基本形式，如将“running”、“runner”、“runs”等形式的单词都转换成“run”。这也有助于减少数据维度和噪声，并提高模型的效率和准确率。

本文使用的是如下四个停用词表 baidu\_stopwords.txt，cn\_stopwords.txt，hit\_stopwords.txt，scu\_stopwords.txt。在处理文本时，我去除了数字字母以及特殊符号。

## jieba分词

结巴分词是一款中文分词工具，由建立在Python上的工具包jieba实现。结巴分词采用了基于前缀词典和动态规划算法的分词方法，因此具有较高的分词准确率和分词速度。

结巴分词主要包含三种分词模式：

* 精确模式：针对文本进行全模式扫描，将句子精确地切分成词语。
* 全模式：针对文本进行全模式扫描，在不考虑上下文的情况下尽可能地切分出所有可能的词语。
* 搜索引擎模式：在精确模式的基础上，对长词再次进行二次切分，保留所有可能的词语。

除了分词功能外，结巴分词还具有词性标注、关键词抽取、文本摘要、并行分词和自定义词典等功能，可以满足不同应用场景的需求。

本文使用的是精确模式。

## 词嵌入

词嵌入（Word embedding）是自然语言处理（NLP）中一种表征学习技术，它是将每一个单词映射到一个实数值向量上。该向量通常比较稠密，维度也比较低。这个映射过程是从单词高维空间到低维连续向量空间的一个过程。也就是说，通过词嵌入技术，我们可以把文本中的每个单词表示为一个实数值向量，这样就能够在模型中使用，从而更好地进行文本处理和分析。

常用的词嵌入方法有One-hot Representation、Word2Vec和GloVe等。

本文主要使用的是Word2Vec模型。

### Word2Vec介绍

CBOW和Skip-gram是Word2Vec模型中的两种经典算法。

CBOW（Continuous Bag-of-Words）算法是通过给定上下文词来预测目标词汇的概率。它将文本序列中的每个词的上下文作为输入，并尝试预测当前词汇。CBOW在训练时，通过将各个上下文词汇的词向量进行加和，得到了一个平均向量，再将该平均向量通过线性变换映射成目标词汇的概率输出。CBOW相对于Skip-gram模型，更加适用于训练数据集较大、高频词多的情况下。

Skip-gram（跳字模型）算法则是基于目标词汇预测其上下文单词的概率。即从中央单词中预测窗口内的周围单词。Skip-gram模型训练时，通过将中央词汇的词向量输入神经网络，得到与之关联的上下文词汇的概率分布。Skip-gram相对于CBOW模型，更加适用于训练数据集规模较小、低频词多的情况下。

Negative Sampling和Hierarchical Softmax是Word2Vec模型中的两种优化方法。

Negative Sampling算法通过一个二分类模型学习到每个单词的嵌入向量。在训练时，它只针对当前上下文中出现的负样本进行更新，以此优化效率。该优化方法一般适用于大规模数据集并且高频词汇较多的情况下。

Hierarchical Softmax是一种哈夫曼树结构，被应用到Word2Vec中以实现更加高效的计算。在训练时，通过对目标单词在哈夫曼树中进行分布式表示，并采用分层的方式计算其概率。和Negative Sampling相比，Hierarchical Softmax适用于低频词汇较多、数据集较小的情况下。

本文使用的模型是Skip-gram模型和Negative Sampling优化方法。

# TEXTCNN网络的构建

## 卷积层

TextCNN的卷积层采用不同大小的卷积核对输入的文本进行特征提取，这些卷积核的窗口大小可以是1、2、3、...、n（n为卷积层的输入长度）。具体来说，每个卷积核窗口会在输入的文本中滑动，提取出相邻词语的n-gram特征。这些特征能够捕捉到词语的局部性和顺序性，并且通过不同大小的卷积核得到的特征可以捕捉到不同尺度的文本结构。

## 池化层

实施池化的目的有以下几点 ：

* + 降低数据维度，减少计算量和减小过拟合问题。
  + 对输入数据的位置、大小等变化具有一定的不变性。
  + 可以通过池化操作增强模型的鲁棒性，提高模型的泛化能力。

常见的池化层有多种不同形式的非线性池化函数，其中最大池化(Max pooling)是最为常见的。它是将输入的图像划分为若干个矩形区域，对每个子区域输出最大值。直觉上，这种机制能够有效地原因在于，在发现一个特征之后，它的精确位置远不及它和其他特征一起存在的事实更为重要，因此可以忽略掉一些位置信息的同时保留特征信息。其他池化方法还包括平均池化、随机池化等。

## Dropout层

每个神经元在每次训练中都有一定概率P被设置为0，这个过程相当于给神经网络引入了噪声，从而迫使神经网络不得不寻找更加鲁棒和泛化的解决方案。在测试时，神经元不再被随机舍弃，并且通过缩放权重来保持输出不变，从而可以得到更加稳定的结果。

与其他正则化方法（如L1或L2正则化）相比，Dropout的优点在于可以作用于神经网络的任何层。因此，它可以被广泛地应用于各种类型的神经网络中，包括全连接网络、卷积神经网络、循环神经网络等。

## 全连接层

全连接层（Fully Connected Layer），也称为密集连接层（Dense Layer)，是深度学习中最常用的一种神经网络层，它被用于将前一层的所有神经元都连接到当前层中的每一个神经元。在前馈神经网络和卷积神经网络中，全连接层通常被用作网络的最后一层，用于将低级别特征转换为高级别特征，以便进行分类、回归等任务。

## 激活函数

激活函数是一种非线性函数，被广泛应用于神经网络中，主要作用是给神经元的输出添加非线性因素。简而言之，激活函数通过对神经元输入进行非线性映射，使神经网络可以拟合各种复杂的函数关系。

常见的激活函数包括：Sigmoid、Tanh、ReLU、Leaky ReLU、ELU等。其中，Sigmoid和Tanh是饱和激活函数，容易出现梯度消失的问题；而ReLU以及其变种（如Leaky ReLU和ELU）则是非饱和激活函数，可以有效地避免梯度消失的问题，并且具有计算速度快、实现简单等优点。

### Relu函数



ReLU函数的优点在于它具有简单、计算速度快等特点，容易实现并且不会出现梯度消失的问题。此外，由于ReLU函数的输出是稀疏的，即只有一部分神经元被激活，因此它还可以有效地缓解过拟合的问题，提高模型的泛化能力。

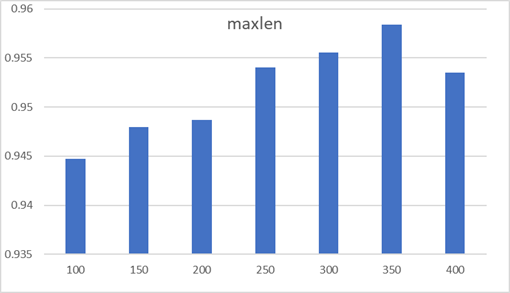
值得注意的是，当 x 小于 0 时，ReLU函数的导数为 0，这意味着在反向传播过程中，对应的梯度值也为 0，这会导致一些神经元的权重无法更新。为了解决这个问题，又出现了一种被称为带泄露的ReLU函数（Leaky ReLU），它在x小于0时，输出一个非零的小值，以保持梯度的流动性。

# 模型的评估与参数调优

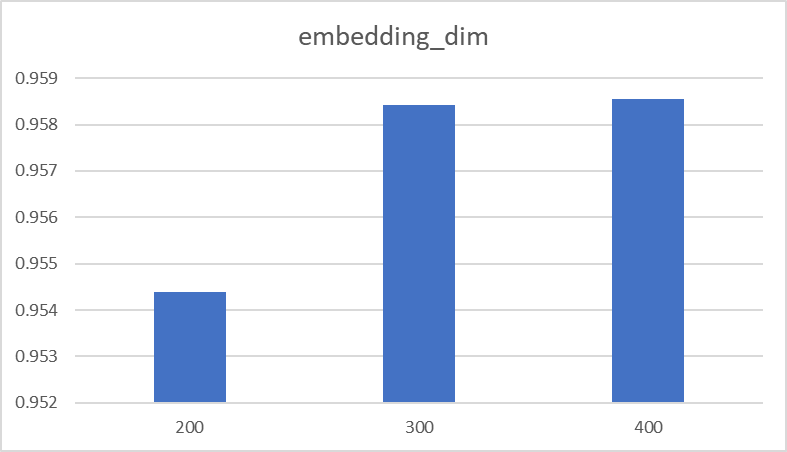
**初始条件：**

|  |  |  |  |
| --- | --- | --- | --- |
| **Max\_len** | **Embedding\_dim** | **Dropout组合** | **L2系数** |
| **350** | 400 | 0.5+0.3+0.2 | 1e-3 |
| **Epoch** | 学习率 | 是否使用动态学习率 | 优化器 |
| **50** | 1e-4 | 是 | Nadam |

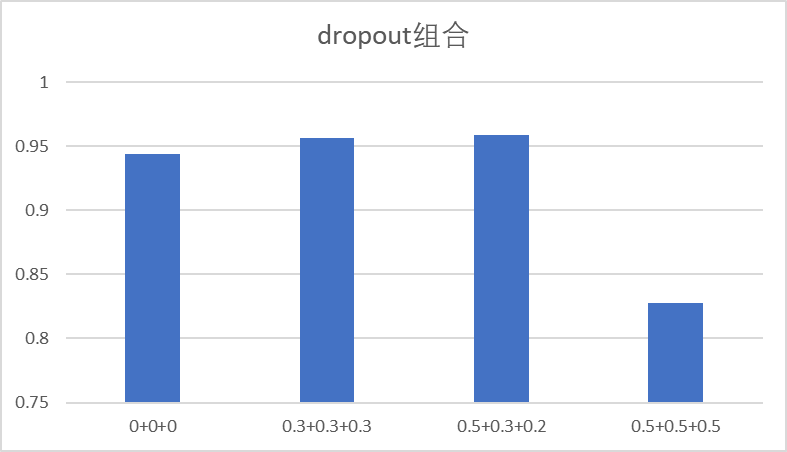
## 句子长度对结果的影响



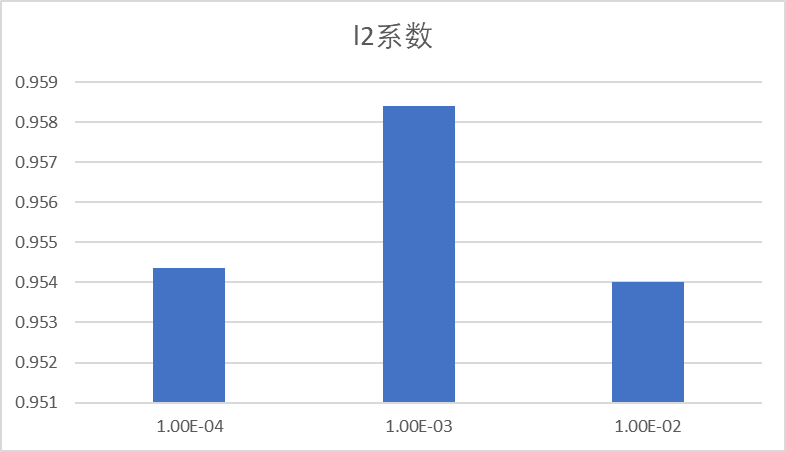
## 词语维度对结果的影响



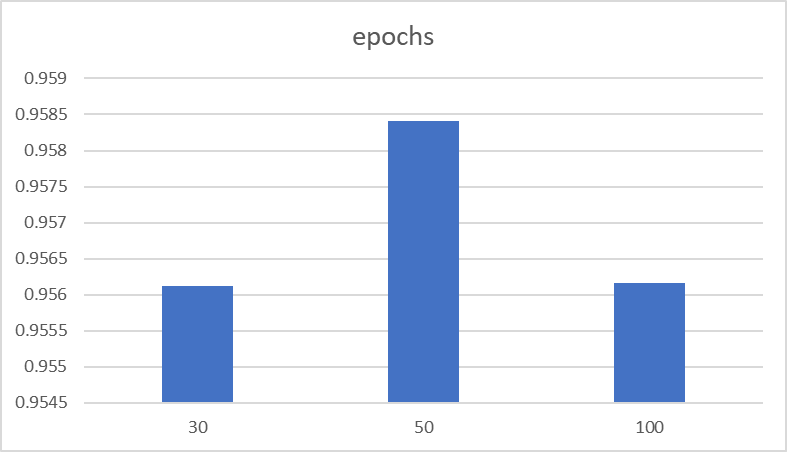
## Dropout组合对结果的影响



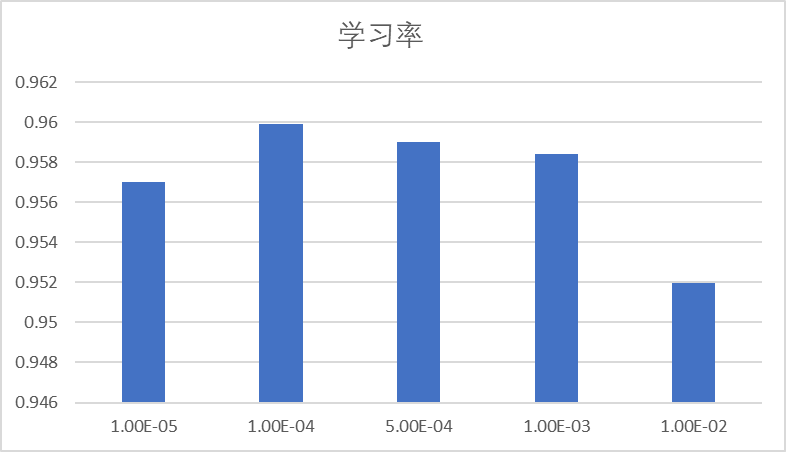
## L2系数对结果的影响



## Epoch对结果的影响



## 学习率对结果的影响



## 不同优化器对结果的影响

|  |  |
| --- | --- |
| **SGD** | **Nadam** |
|  |  |

## 是否使用动态学习率对结果的影响

|  |  |
| --- | --- |
| **不使用** | **使用** |
|  |  |

## 是否使用dropout层

|  |  |
| --- | --- |
| **不使用** | **使用** |
|  |  |