**天 津 师 范 大 学**

**《机器学习/自然语言处理》考查报告**

题目：基于**深度学习的文本分类实验**

**学 院： 电子信息与工程学院-人工智能学院**

**学生姓名：****于瀛**

**学 号：2030330041**

**专 业：智能科学与技术**

**年 级：智能2001**

**完成日期：2023.12.28**

**授课教师：李正涛**

摘 要：文本分类是自然语言处理领域的一个重要任务，旨在将文本数据划分为预定义的类别。近年来，基于深度学习的文本分类方法取得了显著的进展。本实验使用不同深度学习模型的文本分类方法，通过利用深度学习模型的强大表示能力，实现了高精度的文本分类。实验结果表明，这几种方法在Wiki文本分类数据集上均取得了优异的性能。

关键词：文本分类，深度学习，自然语言处理，神经网络

1. 引言

文本分类是一种广泛应用于自然语言处理领域的任务，它可以用于垃圾邮件检测、情感分析、新闻分类等应用。文本分类是自然语言处理（NLP）中的一个重要任务，它涉及到将文本数据分配到一个或多个预先定义的类别中。尽管文本分类在许多应用中都非常有用，但在实际操作中，它面临着多种困难：文本数据通常包含大量的词汇和语言结构，这导致了数据的高度多样性。这种多样性使得很难找到能够捕捉所有相关信息的特征。语义复杂性：文本中的语义信息可能非常复杂，包括多义词、同义词、反义词、短语和句子结构等。这些复杂性给文本分类带来了挑战。上下文依赖性：文本中的许多词汇和表达都依赖于上下文，相同的词汇在不同的上下文中可能有不同的含义。这要求分类系统能够理解上下文信息。语言变化：语言是不断发展变化的，新的词汇和表达方式不断出现，这要求文本分类系统能够适应这种变化。非结构化数据：文本数据通常是半结构化或非结构化的，没有固定的格式或结构，这给数据的预处理和特征提取带来了困难。数据稀疏性：对于某些特定的文本类别，可能很难收集到足够的训练数据，这导致模型可能无法学习到足够的特征来区分不同的类别。类别不平衡：在许多文本分类问题中，不同的类别可能有不同数量的样本，这可能导致模型偏向于数量较多的类别，而忽略数量较少的类别。模型泛化能力：文本分类模型需要在训练集上学习到的特征泛化到未见过的数据上。这要求模型具有良好的泛化能力，以应对新的数据分布。计算资源：文本分类模型通常需要大量的计算资源来处理大量的文本数据，这在大规模应用中可能成为一个问题。随着深度学习技术的不断发展，基于神经网络的文本分类方法逐渐成为研究热点。本实验提出了多种基于深度学习模型的文本分类方法，并对其进行了详细的实验验证。

2. 相关工作

近年来，许多研究者开始探索基于深度学习的文本分类方法。早期的文本分类方法主要采用传统的机器学习模型，如朴素贝叶斯、支持向量机等。然而，这些方法无法捕捉文本中的深层语义信息。随着深度学习技术的发展，研究者开始将神经网络应用于文本分类任务。其中，卷积神经网络（CNN）、循环神经网络（RNN）和Transformer模型是三种常用的神经网络模型。这些模型具有强大的表示能力，能够捕捉文本中的深层语义信息。因此，本实验使用了不同的基于深度学习的模型进行文本分类。

3. 方法

3.1 数据预处理

在进行文本分类任务之前，首先对原始文本数据进行预处理。预处理主要包括分词、去停用词、词干提取等操作。本实验采用wiki数据集，将数据集分为6万训练集，1万测试集和1万验证集，进行合理的预处理可以提高文本分类的准确率。

3.2 深度学习模型

本实验采用深度学习模型进行文本分类。深度学习模型具有强大的表示能力，能够捕捉文本中的深层语义信息。文本分类任务中，模型选择至关重要。常见的文本分类模型有朴素贝叶斯、支持向量机、随机森林、深度学习等。不同模型具有不同的优缺点，需要根据具体任务和数据选择合适的模型。在实验中，我选择了以卷积神经网络（CNN）、循环神经网络（RNN）和Transformer模型为基础的模型进行对比实验。使用TextCNN（经典的CNN文本分类模型）、TextRNN（BiLSTM）、TextRNN\_Att（BiLSTM+Attention）、DPCNN、Transformer以及bert这些模型进行文本分类，将输入序列输入到深度学习模型中，能够有效地提取文本的特征表示，并高效进行文本分类。

3.3模型评估

在文本分类任务中，模型评估是检验分类性能的重要手段。常见的评估指标有准确率、精确率、召回率、F1值等。评估指标可以了解模型的分类效果，为模型优化提供依据。

3.4模型优化

在文本分类任务中，模型优化是提高分类性能的关键。优化方法包括调整模型参数、使用集成学习、添加外部知识等。采用交叉熵损失函数，通过反向传播算法对模型进行训练和优化。通过不断优化，可以提高模型的分类效果。

4. 实验与分析

为了验证本论文提出的基于深度学习模型的文本分类方法的有效性，我在Wiki数据集上进行了实验。实验结果表明，各个方法在Wiki数据集上均取得了优异的性能，明显优于传统的文本分类方法。

Wiki数据集类别包括：哲学、经济学、法学、教育学、文学、历史学、理学、工学、农学、医学、管理学、军事学、艺术学、交叉学科14个一级学科。

4.1使用TextCNN进行文本分类

数据处理：所有句子padding成一个长度：seq\_len

1.TextCNN模型输入：[batch\_size, seq\_len]

2.经过embedding层：加载预训练词向量或者随机初始化, 词向量维度为embed\_size：

[batch\_size, seq\_len, embed\_size]

3.卷积层：NLP中卷积核宽度与embed-size相同，相当于一维卷积。3个尺寸的卷积核：(2, 3, 4)，每个尺寸的卷积核有100个。卷积后得到三个特征图：[batch\_size, 100, seq\_len-1]、[batch\_size, 100, seq\_len-2]、[batch\_size, 100, seq\_len-3]

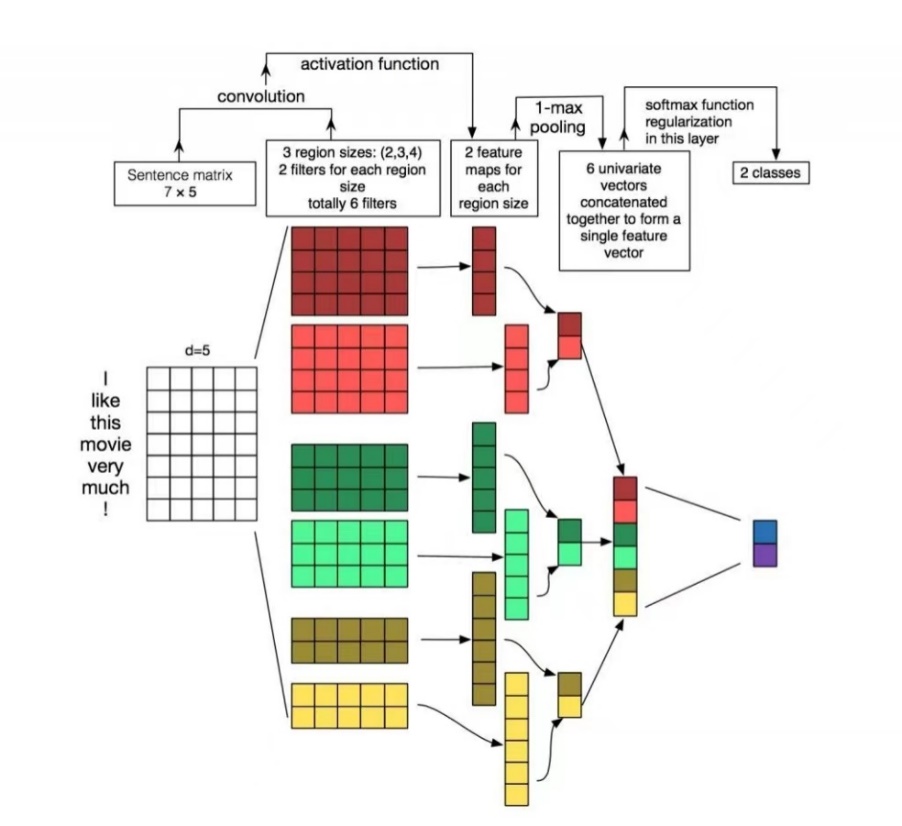
4.池化层：对三个特征图做最大池化：[batch\_size, 100]、[batch\_size, 100]、[batch\_size, 100]

5.拼接：[batch\_size, 300]

6.全连接：num\_class是预测的类别数、[batch\_size, num\_class]

7.预测：softmax归一化，将num\_class个数中最大的数对应的类作为最终预测[batch\_size, 1]

分析：卷积操作相当于提取了句中的2-gram，3-gram，4-gram信息，多个卷积是为了提取多种特征，最大池化将提取到最重要的信息保留。



**TextCNN**

4.2使用TextRNN进行文本分类

1.模型输入： [batch\_size, seq\_len]

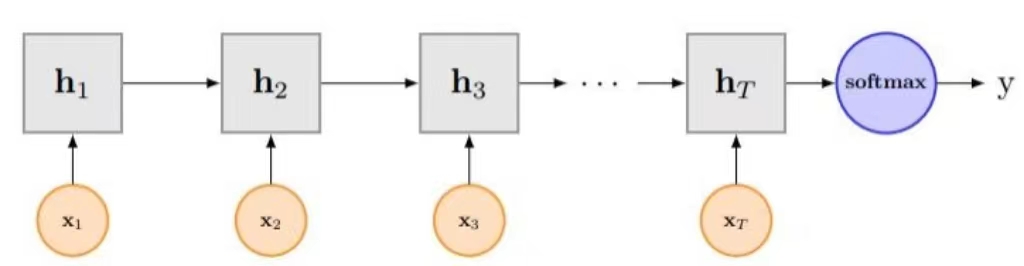
2.经过embedding层：加载预训练词向量或者随机初始化, 词向量维度为embed\_size： [batch\_size, seq\_len, embed\_size]

3.双向LSTM：隐层大小为hidden\_size，得到所有时刻的隐层状态(前向隐层和后向隐层拼接)[batch\_size, seq\_len, hidden\_size \* 2]

4.拿出最后时刻的隐层值：[batch\_size, hidden\_size \* 2]

5.全连接：num\_class是预测的类别数[batch\_size, num\_class]

6.预测：softmax归一化，将num\_class个数中最大的数对应的类作为最终预测[batch\_size, 1]



分析：LSTM能更好的捕捉长距离语义关系，但是由于其递归结构，不能并行计算，速度慢。

4.3使用TextRNN+Attention进行文本分类

1.模型输入： [batch\_size, seq\_len]

2.经过embedding层：加载预训练词向量或者随机初始化, 词向量维度为embed\_size： [batch\_size, seq\_len, embed\_size]

3.双向LSTM：隐层大小为hidden\_size，得到所有时刻的隐层状态(前向隐层和后向隐层拼接) [batch\_size, seq\_len, hidden\_size \* 2]

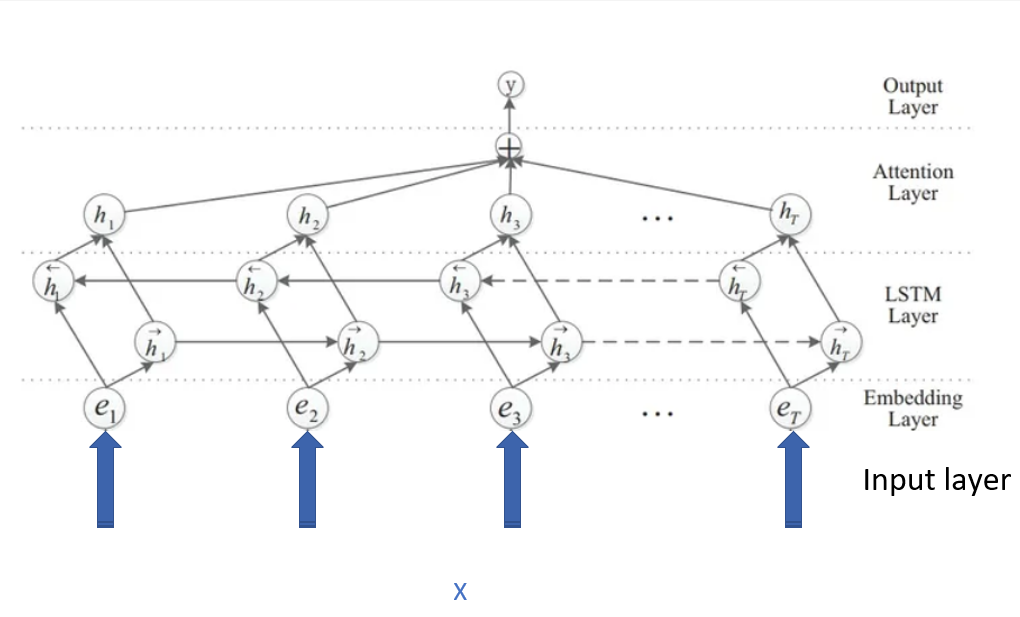
4.初始化一个可学习的权重矩阵w：w=[hidden\_size \* 2, 1]

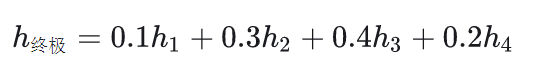
5.对LSTM的输出进行非线性激活后与w进行矩阵相乘，并经行softmax归一化，得到每时刻的分值：[batch\_size, seq\_len, 1]

6.将LSTM的每一时刻的隐层状态乘对应的分值后求和，得到加权平均后的终极隐层值[batch\_size, hidden\_size \* 2]

7.对终极隐层值进行非线性激活后送入两个连续的全连接层[batch\_size, num\_class]

8.预测：softmax归一化，将num\_class个数中最大的数对应的类作为最终预测[batch\_size, 1]



分析：其中4-6步是attention机制计算过程，其实就是对lstm每刻的隐层进行加权平均。比如句长为4，首先算出4个时刻的归一化分值：[0.1, 0.3, 0.4, 0.2]，然后

4.4使用DPCNN进行文本分类

1.模型输入： [batch\_size, seq\_len]

2.经过embedding层：加载预训练词向量或者随机初始化, 词向量维度为embed\_size： [batch\_size, seq\_len, embed\_size]

3.进行卷积，250个尺寸为3的卷积核，论文中称这层为region embedding。[batch\_size, 250, seq\_len - 3 + 1]

4.接两层卷积(+relu)，每层都是250个尺寸为3的卷积核，(等长卷积，先padding再卷积，保证卷积前后的序列长度不变)[batch\_size, 250, seq\_len - 3 + 1]

5.接下来进行上图中小框中的操作

I. 进行 大小为3，步长为2的最大池化，将序列长度压缩为原来的二分之一。（进行采样）

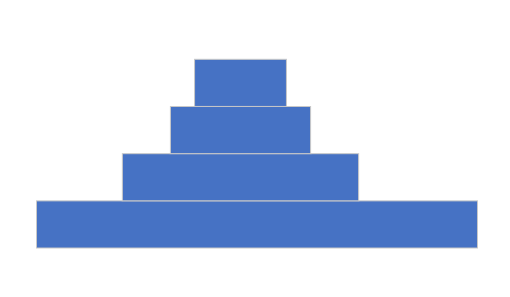
II. 接两层等长卷积(+relu)，每层都是250个尺寸为3的卷积核。

III. I的结果加上II的结果。（残差连接）

重复以上操作，直至序列长度等于1，[batch\_size, 250, 1]

6.全连接+softmax归一化：[batch\_size, num\_class]==>[batch\_size, 1]

分析：TextCNN的过程类似于提取N-Gram信息，而且只有一层，难以捕捉长距离特征。

反观DPCNN，可以看出来它的region embedding就是一个去掉池化层的TextCNN，再将卷积层叠加。

每层序列长度都减半(如上图所示)，可以这么理解：相当于在N-Gram上再做N-Gram。越往后的层，每个位置融合的信息越多，最后一层提取的就是整个序列的语义信息。

综合上述方法的结果得到准确率对比如下表所示。实验结果表明，这两种模型在文本分类任务上都表现出了优秀的性能。通过对实验结果的分析，我发现深度学习模型在文本分类任务上具有以下优点：首先，深度学习模型能够自动学习和提取文本的特征，无需人工设计特征；其次，深度学习模型能够处理大规模的文本数据，具有良好的扩展性；最后，深度学习模型能够处理复杂的文本结构，如长短文本和多主题文本。本实验验证了深度学习在文本分类任务上的有效性。

|  |  |  |
| --- | --- | --- |
| 模型 | Acc | 备注 |
| TextCNN | 91.22% | 经典的CNN文本分类 |
| TextRNN | 91.12% | BiLSTM |
| TextRNN\_Att | 90.90% | BiLSTM+Attention |
| DPCNN | 91.25% | 深层金字塔CNN |

5. 结论

本使用通过基于深度学习模型的文本分类方法，通过利用深度学习模型的强大表示能力，实现了高精度的文本分类。实验结果表明，该方法在Wiki数据集上均取得了优异的性能。未来，我们将进一步探索深度学习模型在文本分类任务中的应用，以提高分类性能和泛化能力。