# 实验3：使⽤卷积神经⽹络TextCNN进⾏文本识别报告

# 实验背景

1.1实验数据

THUCNews中文文本分类数据集，训练集180,000条，验证集10,000 条，测试集10,000 条

1.2实验环境

Windows 11，PyCharm，python 3.10

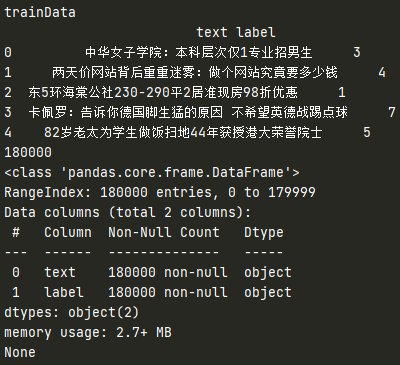
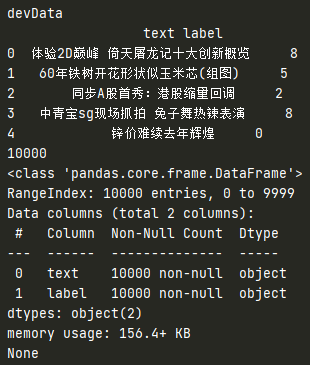
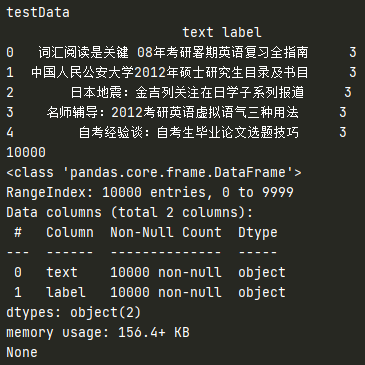
1.3实验内容

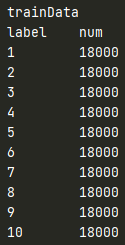
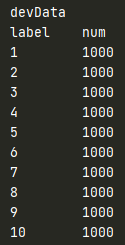
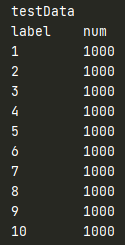
使⽤Pytorch实现⼀个TextCNN卷积神经网络，并完成对THUCNews数据集的训练和测试。

1.4数据分析

THUCNews是⼀个中文文本分类数据集，根据新浪新闻RSS订阅频道2005~2011年间的历史数据筛选过滤⽣成。在原始新浪新闻分类体系的基础上，标注者重新整合划分出14个候选分类类别：财经、彩票、房产、股票、家居、教育、科技、社会、时尚、时政、体育、星座、游戏、娱乐。原始的THUCNew数据量庞⼤，难以处理和使⽤，故本次实验对数据进⾏了简单抽样，抽取出训练集180000条，验证集10000 条，测试集10000 条。

对三个数据集进行简单的数据分析，可以发现共10种标签，各标签的text数量一致（即均分），且均无缺失值。特征text为字符串类型，标签label为数值类型。

1.5卷积神经网络图像分类和文本分类区别

（1）处理内容

CNN在图像分类中处理二维图像数据，输入数据是原始的像素值或经过标准化处理的图像像素值，像素在二维平面上排列，这种结构使得CNN能够通过卷积操作有效地提取图像中的局部特征；在文本分类中处理一维文本数据，输入数据通常是词向量或字符向量序列，文本可能需要经过词嵌入层将离散的文字转换为连续的向量表示。

（2）卷积方式（卷积核作用）

在图像分类中，CNN使用二维卷积，通过滑动窗口的方式在图像上应用卷积核，从而提取出图像的边缘、纹理、形状等局部特征；在文本分类中，CNN使用一维卷积，在文本序列上滑动，通过卷积核与文本向量的乘积运算来提取局部的n-gram特征（例如，短的词组或字符序列）。

（3）特征提取

CNN在图像分类中能够自动学习图像的低级到高级特征，从边缘、纹理到更复杂的形状和模式，这些特征是通过多层卷积和池化操作逐步提取和组合的；在文本分类中，文本特征更多地体现在词语、短语以及它们之间的组合关系上。

（4）模型结构

用于图像分类的CNN模型通常包含多个卷积层、池化层和全连接层，卷积层和池化层交替出现，用于提取和压缩图像特征，全连接层则用于将提取的特征映射到预定义的类别上；而由于文本数据的一维性使得特征提取相对简单，TextCNN模型可能包含更少的卷积层和池化层，但包含嵌入层(embedding layer)，将文本数据转换为词向量。

# 实验方案设计

2.1预处理数据集

读取数据集文件，将其分割为text和label并存储。

构建语料库，包含词表和嵌入层。遍历训练集文本中的每个字，如不在词表中，将其添加到词表，按序赋予索引（当前词表长度），将文字映射为数字特征，实现word2index；初始化嵌入层并保存，大小为词表长度和嵌入维度。

定义一个TextDataset类，继承自torch.utils.data.Dataset，并对\_\_init\_\_，\_\_getitem\_\_和\_\_len\_\_方法进行重写。在\_\_getitem\_\_方法中，截取文本为最大长度，如果文本长度小于最大长度，则用<PAD>填充，使得文本长度一致；将该文本按照词表转换为数字序列（未知字使用<UNK>的索引），最后将数字序列转换为PyTorch张量，并增加一个维度。

2.2 TextCNN基本原理和框架

卷积神经⽹络的核⼼思想是捕捉局部特征，对于文本来说，局部特征就是由若⼲单词组成的滑动窗⼝，这在NLP任务中被称为N元语法( N-gram )，N-gram已经被证明是有⽤的文本特征。TextCNN的优势在于能够⾃动地对N-gram特征进⾏组合和筛选，获得不同抽象层次的语义信息。

（1）嵌入层（Embedding Layer）

首先将文本进行分词处理，通过word2vec或者GLOV等embedding方式将每个词成映射成一个n维词向量，将自然语言数值化，便于后续的处理。

（2）卷积层（Convolution Layer）

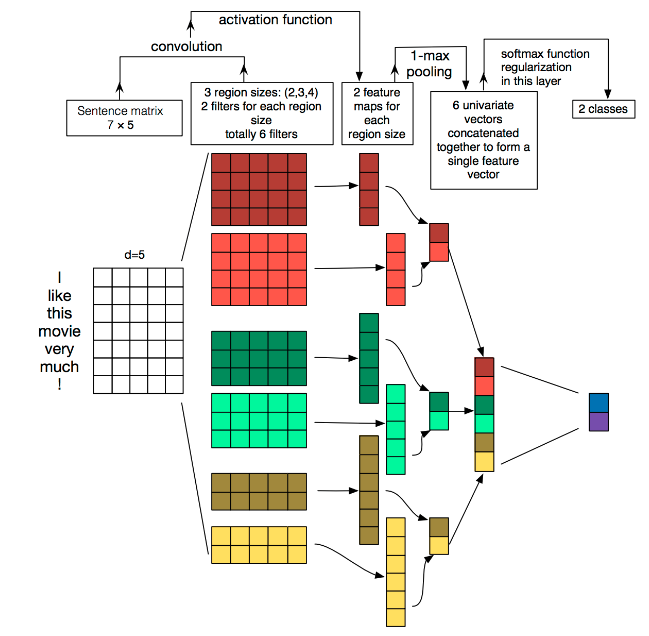
在TextCNN中，输入层是一个由词向量拼成的n\*k词矩阵，每行向量代表一个词，因此卷积核的宽度与词向量的维度(k)一致，只进行高度方向的滑动。同时相邻词位于同一句子，关联度较高。

（3）池化层（Pooling Layer）

在卷积层中，不同高度卷积核使得卷积得到的向量维度不一致，因此使用最大池化或平均池化以减少特征图的维度，同时保留最重要的特征。

（4）全连接层（Connected Layer）

包含一个或多个隐藏层，每个隐藏层通常使用激活函数（如ReLU）增加非线性因素，最后一层使用Sigmoid或Softmax激活函数得到分类的概率分布。



2.3构建卷积神经网络

定义一个Block类，继承自torch.nn.Module，包含一个卷积层，一个ReLU激活函数和一个一维最大池化层。在前向传播过程中，依次进行卷积、激活函数和池化，且过程中需要用squeeze函数对张量降维。

定义一个TextCNNModel类，也继承自torch.nn.Module，定义嵌入向量的维度，创建三个不同大小卷积核对应的Block实例，定义一个线性层作为分类器和一个交叉熵损失函数。在前向传播过程中，通过嵌入层将文本索引转换为嵌入向量，并经过三个Block实例处理，最后拼接输出并通过分类器输出预测结果。

2.4超参数选择

词向量维度决定了词向量表示空间的复杂度，维度过高可能导致过拟合，过低则可能无法充分表示词的语义。经测试embedding\_num为100时效果较好。

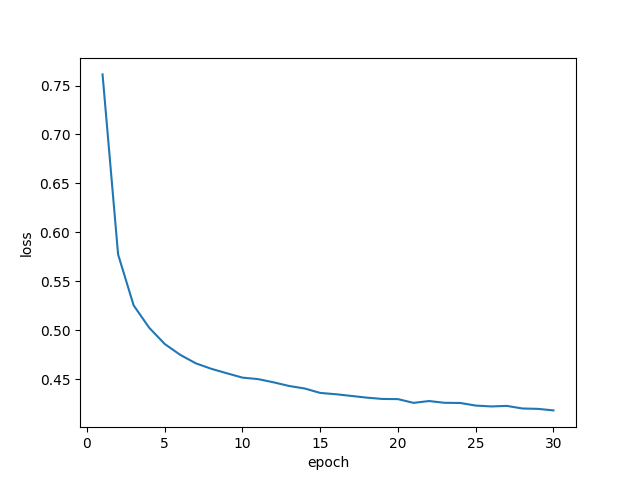
卷积核的大小决定了模型能够捕捉到的n-gram特征的范围，较小的卷积核能够捕捉到更精细的局部特征，而较大的卷积核则能够捕捉到更广泛的上下文信息，经测试使用高度为2、4、6的卷积核。适当增加卷积核数量可以增加模型的容量，使其能够学习到更多的特征，故将卷积核数量num\_filters设置为3。

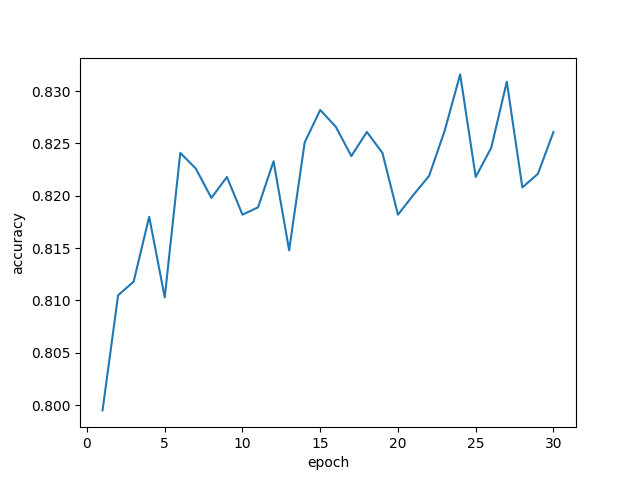
优化器选择AdamW，学习率lr从较小值（如0.001）开始，根据训练结果和运行效率进行调整，最终选择0.01作为学习率，收敛较快且准确率也较高。

较大的batch\_size可以加快训练速度，但可能会导致模型精度下降；而较小的batch\_size可以提高模型精度，但会减慢训练速度。通过测试发现batch\_size为64时可以减少训练时间，并且准确率也较高。

# 实验过程

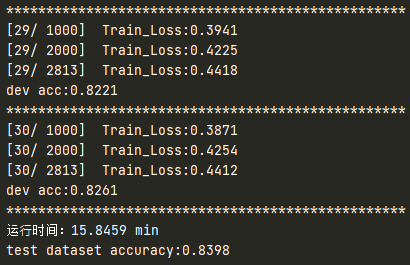
根据训练过程中开发集测试的输出信息可以发现，随着训练迭代次数增加，loss值在前5代完整训练中迅速降低，在epoch超过5后缓慢降低；测试准确率迅速升高，在大约15代完整训练后已经达到82%以上，之后在82.5%附近波动。





# 性能测试

使用验证集调整超参数，并保存最佳模型，最后使用测试集进行测试，根据结果，其在测试集上的准确率为83.98%。



# 实验习题

1. 简述文本是如何转变为word\_id，再转变为word embedding的，并说明为什么要将文本从字符转化为word embedding（提示：built\_curpus函数实现了这一过程）

（1）文本→word\_id：构造一个词表word\_2\_index（python字典），将每个字映射到唯一数值，遍历每条文本的每个字，若其不在词表中，则将其添加到词表，并以当前词表长度作为索引值(word\_id)。

（2）word\_id→word embedding：使用torch库的nn.Embedding类初始化一个嵌入层，其接受两个参数——词表大小和嵌入向量维度，完成词嵌入操作。

（3）为什么将文本从字符转化为word embedding

字符（或单词）本身不包含语义信息，仅仅是文本数据的一种原始表示，而word embedding通过嵌入向量将单词映射到高维空间中，使得语义上相似的单词在空间中彼此接近，这种表示方式有助于模型捕捉文本中的语义信息。

如果直接使用词表的大小作为特征维度，那么当词汇表很大时，会导致特征维度非常高，引发维度灾难。而word embedding将每个单词映射到一个固定长度的实数域上的向量，从而大大减少了特征的维度。

1. 为什么本次实验要构建开发集dev？开发集有什么作用？

开发集用于调整模型的参数、选择最佳特征，以及进行其他决策，以评估和优化模型性能。在本次实验中，开发集的数据接近实际应用时（即测试集）的数据分布，用于调整超参数，以使模型达到最佳预测效果，同时也可以帮助了解模型的泛化能力，以及及时发现并纠正过拟合或欠拟合现象。

1. 在Transformers等架构流行的今天，TextCNN在性能上的优势已经不显著，但是一些线上服务仍然使用TextCNN作为文本分类模型，请说明可能的理由

（1）计算资源限制：Transformer模型结构复杂，往往需要更多的计算资源来进行训练和推理；相比之下，TextCNN模型结构相对简单，对计算资源的要求较低。在一些计算资源有限或成本敏感的线上服务中，使用TextCNN可以更有效地利用现有资源。

（2）稳定性和可维护性：TextCNN模型出现较早，更为稳定，且架构简单，便于后期维护。且已经基于TextCNN模型进行开发的系统需要保持兼容性。

（3）任务场景：当数据集较小或分类任务较为简单时，Transforme模型可能出现过拟合并增加不必要计算成本。