# 实验3（附加）：词嵌入的原理与预训练词向量使用报告

# 实验背景

1.1实验数据

sentence polarity dataset v1.0，影评情感分析英文数据集，包含5331条积极文本和5331条消极文本

1.2实验环境

Windows 11，PyCharm，python 3.10

1.3词嵌入和预训练词向量

词向量是⽤来表⽰词的向量，也可被认为是词的特征向量或表征。将词汇映射到实数向量的技术即为词嵌入（word embedding）。概念上⽽⾔，它是指把⼀个维数为所有词的数量的⾼维空间嵌入到⼀个维数低得多的连续向量空间中，每个单词或词组被映射为实数域上的向量。词嵌入的⽅法包括⼈⼯神经⽹络、对词语同现矩阵降维、概率模型以及单词所在上下文的显式表⽰等。

机器无法直接理解和判断自然语言的意义及其之间的关联性，而通过词嵌入这种方式将词语转变为词向量，机器便可对词语进行计算，通过计算不同词向量之间的距离和夹角余弦值而得出词语之间的相似性。

1.4 Elmo词向量技术背景

无论是VSM、LSA还是skip-gram、CBOW所获得的词向量都是静态的，并没有考虑不同上下中词的不同语义。例如使用gensim训练好word2vec向量后，固定的词，不论该词所在的上下文，获得的输出向量都是一样的。而动态词向量的提出可以解决这一问题。

ELMo（Embeddings from Language Models），是一种动态词向量表示方法，能够对单词的语法特征以及语义特征进行建模，其最终训练得到一个语言模型。当需要文本s中某个词w的词向量表示时，ELMo通过将s作为输入，得到文本的向量表示，继而获得词w在文本s语境下的词向量表示。同时，其动态特性能够让其适应不同的上下文，称为上下文相关词向量或者动态词向量。

# 实验方案设计

实验3的TextCNN中，在构建语料库时，首先将文本中的每个字映射为唯一数字，创建word\_2\_index词表，然后利用nn.Embedding函数构建词向量，将以词表长度为维数的高维空间嵌入到一个指定的维数低得多的连续向量空间中，embedding = nn.Embedding(len(word\_2\_index), embedding\_num)。

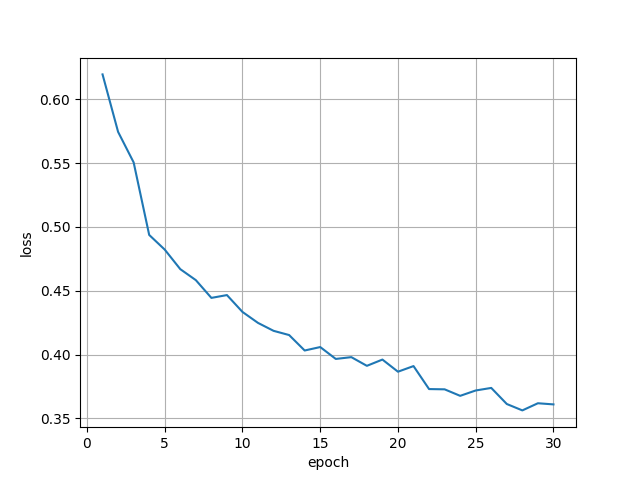
而在该实验中，需要初始化一个allennlp库的Elmo类的实例，对待转换的分好词的原始句⼦列表（例如sentence\_lists=[ [I, have, a,dog ], [ I, have, an, apple ]]）进行预训练。

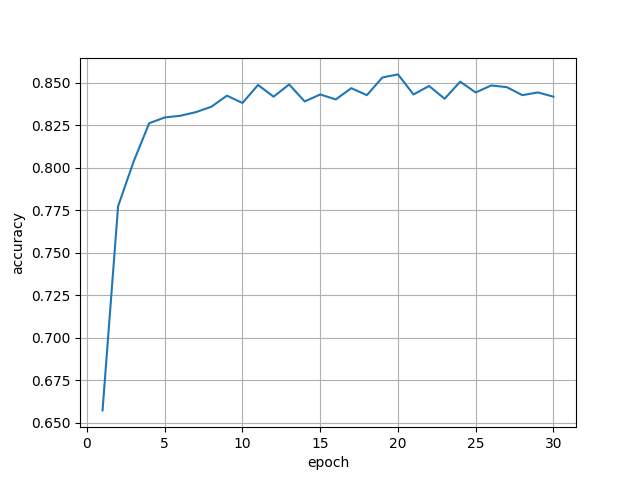
由于Elmo为预训练词向量，词表在预训练时已经固定，不能随意更改，因此使⽤Elmo时，不能使⽤之前代码中word\_2\_index计算词表映射，⽽需要与预训练时的词表对齐，所以使⽤AllenNLP库中提供的batch\_to\_ids来进行转换。完成转换后，直接调⽤elmo模型即可获得词向量。

实验的其他部分，如训练、测试环节，CNN的设置等与实验3基本一致。

# 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15代完整训练中迅速降低，在epoch超过15后缓慢降低；测试准确率迅速升高，大约10代完整训练后已经达到83%以上，之后在84%附近波动。

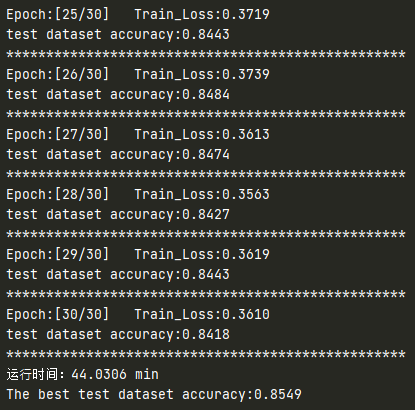




与nn.Embedding性能相比，在其他超参数相同的情况下，使用ELMo词嵌入方法的性能更佳，原因是ELMo是一种动态词向量表示方法，每个词没有固定的词向量，是根据词的上下文环境来动态产生当前词的词向量，可以较好解决一词多义的问题。

# 性能测试

随机划分数据集，70%作为训练集，30%作为测试集，经过30次完整训练后，最佳模型在测试集上的预测准确率为85.49%。



# 实验习题

1. 简述为何要执行embeddings = torch.unsqueeze(embeddings, dim=1)，包括unsqueeze操作的效果，及对embedding进行unsqueeze操作的理由

（1）效果：torch.unsqueeze 是 PyTorch 中的一个升维函数，用于在指定的维度上插入一个大小为1的维度。embeddings = torch.unsqueeze(embeddings, dim=1) 是在 embeddings 的第二个维度上插入一个大小为1的维度，例如[ ‘I’, ‘like’, ‘apples’ ]→[ [‘I’], [‘like’], [‘apples’] ]。

（2）理由：对embeddings进行维度转换，将embeddings转化为适合输入TextCNN中Block层的维度。

Block层内部使用了卷积层（nn.Conv2d），该层期望的输入张量通常具有四个维度，分别是[ batch\_size, channels, height, width ]。在文本处理中，channels通常被设置为1（通道数），而height和width可以分别对应于文本的长度和嵌入向量的维度。因此，通过unsqueeze操作在第二个维度插入一个大小为1的维度，可以使得batch\_emb的形状符合Block层的输入要求。