LSTM 文本生成

王铭泽 ZY2203115

wmz20000729@buaa.edu.cn

1 Introduction

LSTM 是一种特殊的循环神经网络（RNN），它在处理序列数据时比传统的 RNN 模型更加优秀。这是因为在传统的 RNN 中，信息会不断地在神经网络中传递，但是随着时间步骤的增加，信息会逐渐消失，导致长时间依赖关系无法被有效地捕捉。而 LSTM 通过引入门控机制和记忆单元，可以在很长的时间范围内保持信息的连续性和一致性。LSTM 的核心是一个 LSTM 单元，它包含三个门：输入门、输出门和遗忘门。输入门用于控制新的输入信息是否进入记忆单元，输出门用于控制记忆单元中的信息是否输出，遗忘门用于控制哪些信息需要被遗忘。每个门都有一个 sigmoid 激活函数，用于决定门的打开和关闭程度。此外，LSTM 单元还有一个记忆单元，用于存储和传递信息。在 LSTM 中，每个时间步骤都会输入一个序列数据，然后通过一系列的 LSTM 单元进行处理，最终输出一个序列数据。每个 LSTM 单元都会根据当前的输入、上一个时间步骤的输出和当前时间步骤的记忆单元状态来计算输出和更新记忆单元状态。这样，LSTM 可以有效地捕捉时间序列中的长程依赖关系，从而在很多应用场景中取得了良好的效果。LSTM 被广泛应用于自然语言处理、语音识别、音乐生成等领域。在自然语言处理中，LSTM 常用于文本分类、语言模型、机器翻译等任务。在语音识别中，LSTM 可以用于音频信号的特征提取和序列建模。在音乐生成中，LSTM 可以用于学习音乐序列的模式和结构，进而生成新的音乐作品。

LSTM 的原理如下：LSTM 的关键在于记忆细胞，它可以有效地保存长序列数据中的信息。在每个时间步，LSTM 会根据当前输入和上一个时间步的隐藏状态，计算遗忘门、输入门和候选记忆细胞状态，然后根据这些状态计算当前时间步的记忆细胞状态。在计算过程中，遗忘门可以控制哪些信息需要被保留或遗忘，输入门可以控制哪些新的信息需要被加入，候选记忆细胞可以计算当前时刻的记忆状态。记忆细胞可以通过梯度反向传播算法来更新权重，以使得模型可以自动学习到最优的权重参数。每个时间步，LSTM 会根据当前时间步的记忆细胞状态和输出门状态，计算当前时间步的隐藏状态，然后将隐藏状态作为模型的输出。

2 Method

2.1 数据读取与编码

首先定义字典类，在代码中 Dictionary(object) 得以实现。其中，word2idx 是一个字典，将单词映射到它们在字典中的索引；idx2word 是一个字典，将索引映射回对应的单词；idx 是当前词典中单词的数量。len 方法返回词典中单词的数量。add-word 方法用于将单词添加到词典中。如果单词不在词典中，它将被添加到 word2idx 字典中，并分配一个新的索引。同时，idx2word 字典也会被更新。如果单词已经在词典中，add-word 方法不会做任何事情。

再定义语料库类，在代码中 Corpus(object) 得以实现。统计单词数和构建字典：遍历所有文本文件，使用 jieba 分词工具将每行文本分词，并在字典中记录每个词出现的次数。将文本转化为 id 序列：遍历所有文本文件，将每行文本分词后，将每个词转化为字典中对应的 id。将 id 序列划分为 batch-size 大小的 batch：将 id 序列划分为 batch-size 大小的 batch，并返回一个 Tensor 对象。在实现 LSTM 模型时，我

们需要将文本转化为 id 序列，并将其划分为 batch 进行训练。这个类可以方便我们实现这些操作，并且在实现过程中使用了 jieba 分词工具，可以提高我们处理中文文本的效率。

2.2 LSTM 算法创建

首先定义了模型的各个组件：嵌入层 (nn.Embedding)、LSTM 层 (nn.LSTM) 和线性层 (nn.Linear)。nn.Embedding 用于将输入序列编码为固定尺寸的向量，常用于将 one-hot 向量压缩成低维稠密向量表示；nn.LSTM 用于实现 LSTM 模型，其中 num-layers 表示 LSTM 的层数，batch-first=True 表示输入的数据维度按 (batch-size, seq-length, input-size) 排列，即 (batch-size, time-steps, feature-dim)；nn.Linear 用于输出最终的预测结果。

在 forward 方法中，前向传播时我们首先将输入序列用嵌入层进行向量化，然后将有关的信息传递给 LSTM 层。h 是 LSTM 的隐藏状态。接下来通过将输出 out 变形为 (batch-size*seq-length, hidden-size) 的形状，然后通过将 out 乘以线性层的权重矩阵，来输出最终的预测结果。因此，forward 方法返回的是输出 out 和 LSTM 的隐藏状态 h 的元组，使得我们可以在后续的训练和预测过程中使用。

2.3 训练

定义损失函数和优化器为交叉熵损失函数和 Adam 优化器，用于在训练过程中计算损失和更新模型参数

交叉熵函数是一种常用于分类模型中的损失函数，用于计算预测值和真实标签之间的差异。在机器学习中，我们希望最大限度地减少预测错误，因此我们需要一种可用于衡量预测错误的方式。交叉熵损失函数计算预测概率分布和实际概率分布之间的差距，具体地，它通过将实际标签（标签的值为 1，其余为 0）与预测的概率分布（值在 [0,1] 之间）之间的距离进行比较，来评估模型的性能。在分类任务中，我们通常使用交叉熵函数来目标最小化预测概率分布与实际概率分布之间的差距，以获得最佳的模型效果。

Adam 优化器是一种常用的梯度下降优化算法，它的主要思想是自适应地调整学习率，从而更快地收敛到最优解。Adam 优化器的具体实现方式如下：i. 初始化模型参数和梯度累积变量、为 0。ii. 在每个 batch 中，计算参数的梯度。iii. 计算梯度的一阶矩估计（平均梯度）和二阶矩估计（平方梯度的平均）。iv. 对一阶矩估计和二阶矩估计进行偏差矫正。v. 根据一阶矩估计和二阶矩估计自适应地调整学习率，并使用学习率更新模型参数。vi. 重复步骤 2-5，直到模型收敛或达到指定的迭代次数。Adam 优化器在深度学习中被广泛应用，它具有以下优点：自适应调整学习率，可以更快地收敛到最优解，同时避免了手动调整学习率的麻烦；对梯度的一阶矩估计和二阶矩估计分别进行了偏差矫正，使得估计更加准确；在处理大规模数据集和高维参数空间时表现优秀。

接下来，进入训练的主循环，循环次数为指定的 num-epochs。在每个 epoch 中，我们首先初始化 LSTM 模型的隐藏状态为全 0 向量，然后对于每个时间步，我们将输入和输出都送入模型中得到模型的输出和隐藏状态，并计算模型的损失。接着，我们使用 PyTorch 内置的 backward() 函数计算模型参数的梯度，并使用 optimizer.step() 函数更新模型的参数。

2.4 评估

每个 epoch 结束后计算困惑度：在每个 epoch 结束后，我们调用 evaluate() 函数计算模型在训练集上的困惑度，并将其打印出来。这个困惑度可以用来评估模型的性能。困惑度（perplexity）是一种常用于评估语言模型好坏的指标。它可以用来衡量模型在预测下一个词时的不确定性。简单来说，困惑度越小，模型的预测能力就越好。困惑度是一个在对数空间内计算的指标，它表示给定一个测试集，模型对该测试集的预测能力。

首先初始化模型的状态（states）为全零，并初始化总损失（total-loss）和总单词数（total-words）为 0。然后，使用 torch.no-grad() 上下文管理器进入计算图的“无梯度”模式，以避免在评估阶段不必要地

计算梯度。接下来，对于输入序列中的每个序列片段，将其转换为张量并送入模型中计算输出和状态。然后，根据损失函数计算预测值（outputs）和目标值（targets）之间的损失，并将损失乘以序列长度加入总损失中。同时，将序列长度加入总单词数中，以便计算平均困惑度。

3 Results and conclusion

首先，先试用金庸语料库进行生成文本生成，生成结果如下：

赤老温站他跃上屋顶，对何铁手道：“原来如此，咱们上去截住！”郭靖眼，吴平：“你……你我也得好？”

血刀僧大怒，突然高举，叫道：“你怕我挡住了你么？”洪七公踌躇笑道：“就是找你吃的么？还当真割得起？”正要向郭靖道：“什么自知之明？”戚芳道：“正是。”黄蓉格道：“师父，要怎样不要别人？”小女孩道：“我也他输得很，但给你们想去去我一笔里。重大意义是他毕命，不过是你叔叔的罪名，也就是我跟了强人，分一份儿的胡涂。”郭靖听得了头，道：“我可不在乎。我就爱知道，教你们受了甚么东西，拿起内力，哪一个也敢？如此试招。”王处一道：“也不是这么容易好的，我不想，已自就这般沉重。”黄蓉听到此处境界，突然站起，细细打量，见她已是马钰，便点头答应。

黄药师道：“你瞧，是以江湖上最大的冤家，你发兄长的剑法。”洪七公道：“是啊，我算到既如此见到他？”那老人仰头向天，反来着地道：“好罢，我在两湖武林中抢到了一十五年罢。”这等幸运许多资质的海，料想自己奇怪，听他说这句话来，明知难就十分自负。完颜康虽然不送女儿，但他却丝毫瞧相求，何以又怎会上行在宝应杀退，只听裘千丈又道：“哼，在桃花岛前，第二件便奔了出去。咱们这老儿去微末功夫，江湖上人称有这等为难，全只不知这少年西毒之人被咱们打死了老顽童？”急忙上前，自己却只是个美貌少女，指着他这般狼狈之言，只觉他未及辱大了，心想你着这翡翠功夫，莫不是将黄蓉轻发作。郭靖心想：“看来他还是几个人？”马钰道：“我向他们还要请真吧。”次日一早，一灯大师径随即大喜，抹在主梳上，缓缓的一望着父亲吹奏的清晰之人，袁承志纵身问：“怎么？”黄蓉道：“你和他们干吗？是全真门下么？

同时发现，随着训练次数的提升，困惑度逐渐下降说明训练次数越高文本生成的匹配度越高。

同时分析不同 seqlength 的影响，在 LSTM 中，seq-length 指的是输入序列的长度。seq-length 的大小会对模型的训练和预测产生影响，具体如下：训练时间：较长的 seq-length 需要更多的计算和内存，会导致模型训练的时间变长。因此，在实际应用中，需要根据计算资源和模型性能平衡来选择合适的 seq-length。梯度消失/爆炸问题：在训练深度 LSTM 模型时，较长的 seq-length 可能会导致梯度消失或爆炸问题，特别是在使用反向传播算法时。为了解决这个问题，通常会采用一些技术，如梯度裁剪、权重初始化、残差连接等。模型泛化能力：较长的 seq-length 可能会使模型过拟合训练数据，从而降低模型的泛化能力。因此，需要根据具体任务和数据集的情况，选择合适的 seq-length 以获得较好的模型泛化能力。预测性能：较长的 seq-length 通常可以提高模型的预测性能，因为它可以为模型提供更多的上下文信息，从而更好地捕捉序列中的长期依赖关系。但是，过长的 seq-length 也可能会降低预测性能，因为它可能会使模型在处理某些输入时出现“记忆短路”现象，从而影响模型的准确性。

最终实现了一个基于 LSTM 的文本生成模型，可以生成一定长度的文章。这个模型包含了字典类、语料库类和 LSTM 模型类。在训练模型之前，我们需要先创建一个语料库对象，并将文本转换成一系列的整数。然后我们创建了一个 LSTM 模型，使用 CrossEntropyLoss 作为损失函数和 Adam 作为优化器来训练模型。训练完成后，我们可以使用模型生成指定长度的文章，模型的输出将会自动写入一个文件中。并对于影响模型训练效果的元素进行分析验证其影响。