同济大学

TONGJI UNIVERSITY

《神经网络与深度学习》

实验报告

实验名称	RNN
实验成员	范诗棋（2252320）
日　　期	2025 年 3 月 25 日

1、模型介绍

1.1 RNN 模型

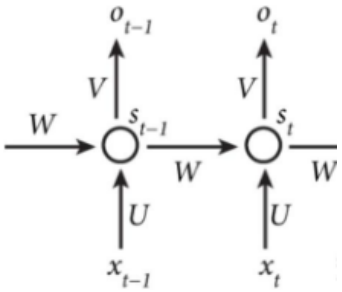
循环神经网络(Recurrent Neural Network，RNN)，是一种能够挖掘具有序列特性的数据中的时序信息以及语义信息的神经网络模型。RNN 的网络结构包括输入层、隐藏层、输出层和权重矩阵。输入序列后，从输入层到隐藏层存在一个参数矩阵 U；从隐藏层到输出层可产生该时刻的输出，两层之间存在参数矩阵 V；从隐藏层到下一个隐藏层存在一个与时间相关的参数矩阵 W，从而将历史信息传递到下一时刻。t 时刻的输出为

$$O_t = g(V \cdot S_t)$$

其中O_t代表 t 时刻的输出，S_t代表 t 时刻的隐藏层的值。S_t的表示为

$$S_t = f(U \cdot X_t + W \cdot S_{t-1})$$

S_t的值由该时刻的输入层和上一时刻的隐藏层决定。



1.2 LSTM 模型

长短期记忆网络(Long short-term memory，LSTM)是一种 RNN 的变体，能够通过门控装置有效缓解基础版本 RNN 的梯度消失和爆炸的问题，并且能够记住长期信息，在适当情况下更新或以往，从而在处理长序列数据时表现出色。

LSTM 的核心结构为细胞状态和门控装置。原始的 RNN 隐藏层只有一个状态 h 用于保存短期状态，在此基础上，LSTM 增加了状态 c 用于保存长期状态。

门控装置包括 3）遗忘门 Forget Gate：决定上一时刻的c_{t-1}有多少保存到当前时刻c_t；1）输入门 Input Gate：决定某一时刻是否会有信息输入到 Memory Cell，即网络输入x_t有多少保存到当前时刻c_t;2）输出门 Output Gate：决定某一时刻 Memory Cell 的输出，即c_t有多少参与当前时刻输出h_t。

三种门控装置的数学表达为：

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$
$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$
$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

当前输入的单元状态为

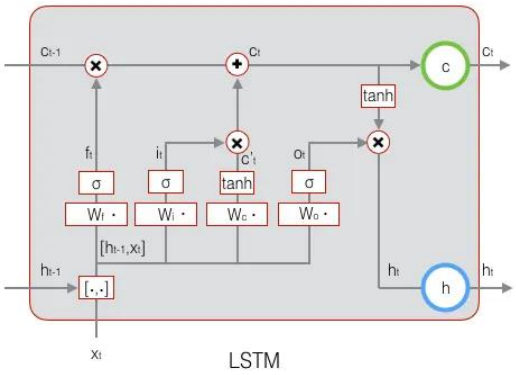
$$\widehat{c_t} = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$$

t 时刻下的单元状态为

$$c_t = f_t \odot c_{t-1} + i_t \odot \widehat{c_t}$$

最终的输出为

$$\mathrm{h}_t = o_t \odot \tanh(c_t)$$



1.3 GRU 模型

门控循环单元(Gate Recurrent Unit，GRU)是 RNN 的一种变体，于 LSTM 类似地能够解决 RNN 不能长期记忆以及梯度相关的问题。但该模型结构较 LSTM 更为简单。

模型的核心结构包括重置门和更新门。重置门控制新的输入信息与过去记忆的结合程度，更新门控制前一时刻信息对当前状态的影响程度。

重置门的数学表达为：

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t])$$

通过重置门可计算候选隐藏状态

$$\widehat{\mathrm{h}_t} = \tanh\left(W \cdot [r_t \odot h_{t-1}, x_t]\right)$$

重置门的输出值r_t越大，新的输入信息与记忆的结合程度越高。

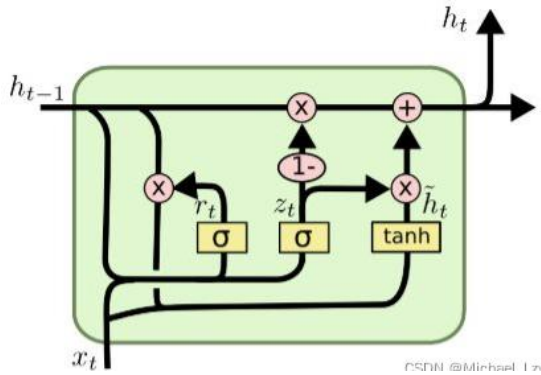
更新门的数学表达为

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t])$$

更新记忆表达式为

$$\mathrm{h}_t = (1 - z_t) \odot h_{t-1} + z_t \odot \widehat{\mathrm{h}_t}$$

更新门的输出值z_t越大，记忆的成分越多，反之则遗忘得越多。



2、诗歌生成过程

本实验采用了原始的 RNN 模型完成诗歌生成任务。利用训练好的 RNN 模型生成诗句时，对应的方法中，首先根据 RNN 单元的隐藏状态维度随机生成一个初始状态，与预定义的起始字符在字符集中的索引一同输入模型。

模型先通过嵌入层将输入的索引转换为固定大小的稠密向量，再将嵌入向量和当前状态输入 RNN 单元，获得输出 h 以及下一状态。将 h 输入到一个全连接层，将其映射为字符集大小的 logits 向量，其分量对应了下一个字符每种取值的概率；在 logits 向量中取概率最大值即为模型预测的下一个 token 的字符集索引，将对应字符存入诗句列表中，并将其与下一状态作为下一个 token 生成的输入。

重复上述过程，直至诗句列表达到要求长度为止。

3、实验结果

实验分别采用开头词汇“日”、“红”、“山”、“夜”、“湖”、“海”、“月”生成诗句，检验模型效果，实验结果如图。

```
def gen_sentence_for_begin_word(begin_word):
    state = [tf.random.normal(shape=(1, 128), stddev=0.5), tf.random.normal(shape=(1, 128), stddev=0.5)]
    cur_token = tf.constant([word2id[begin_word]], dtype=tf.int32)
    collect = [cur_token.numpy()[0]]
    for _ in range(50):
        cur_token, state = model.get_next_token(cur_token, state)
        collect.append(cur_token.numpy()[0])
    return [id2word[t] for t in collect]
print(''.join(gen_sentence_for_begin_word('日')) + '\n')
print(''.join(gen_sentence_for_begin_word('红'))+ '\n')
print(''.join(gen_sentence_for_begin_word('山'))+ '\n')
print(''.join(gen_sentence_for_begin_word('夜'))+ '\n')
print(''.join(gen_sentence_for_begin_word('湖'))+ '\n')
print(''.join(gen_sentence_for_begin_word('海'))+ '\n')
print(''.join(gen_sentence_for_begin_word('月'))+ '\n')
```

✓ 0.5s

```
日日，一声何处不知。eos子不知何处事，不知何处是何人。eos道不知何处事，不知何处是何人。eos子不知何处事，不

红叶滴红花满花。eos有一时无处处，一年何处不知人。eos来不得无人事，不得人间不可知。eos道不知何处事，不知何

山边雨满江风。eos落花声落，风风落月深。eos来无处处，不见此中时。eos子无人事，何人不可知。eos来无处处，不见

夜暮，一片月中春。eos色不知，此时何。eos子不知，此时何。eos子不知，不得之》）eos，一时不可知。eos中无处事，

湖上，无人在何人。eos来不得无人事，不得人间不可知。eos道不知何处事，不知何处是何人。eos子不知何处事，不知

海，今日无人不可知。eos道不知何处事，不知何处是何人。eos子不知何处事，不知何处是何人。eos子不知何处事，不

月侵苔叶，风雨满花声。eos客无人事，何人不可知。eos来无处处，不见此中时。eos子无人事，何人不可知。eos来无处
```

5、实验总结

通过实验学习并熟悉了 RNN 神经单元的功能以及 RNN 神经网络的构建、训练和生成过程，亲自体验了生成诗歌的乐趣，仿佛教小孩说话一样有趣。

诗歌文本是满足序列特征的数据；如果使用经典神经网络完成任务，预测结果仅取决于训练文本出现的频率，会失去语义；而 RNN 能够有效捕捉诗歌文本各个字符与上下文之间的关系，即所谓“语境”，能够运用该规律生成有一定逻辑性和可读性的诗歌文本。