诗歌生成报告

1. RNN、LSTM 和 GRU 模型简介

1.1 RNN（循环神经网络）

循环神经网络（Recurrent Neural Network, RNN）是一类用于处理序列数据的神经网络。与传统的前馈神经网络不同，RNN 具有内部循环结构，允许信息在隐藏状态中传递，以捕捉时间序列的依赖关系。

RNN 结构特点

- 具有时间步（Time Step）概念，允许信息在多个时间步之间传播。
- 共享参数权重，使其能够处理变长输入序列。
- 存在梯度消失和梯度爆炸问题，导致长期依赖学习困难。

RNN 计算公式

设输入序列为 x_t，隐藏状态为 h_t，权重矩阵为 W，偏置为 b，则 RNN 的计算公式如下：

$$h_t = \tanh(W_h h_{t-1} + W_x x_t + b)$$

$$y_t = W_y h_t + b_y$$

其中，h_t 依赖于前一个时间步的隐藏状态 h_{t-1}，体现了序列信息的传递。

1.2 LSTM（长短时记忆网络）

LSTM（Long Short-Term Memory）是为了解决 RNN 中梯度消失和梯度爆炸问题而提出的一种特殊的 RNN 结构。LSTM 通过引入门控机制来控制信息流，使得模型能够有效捕捉长期依赖关系。

LSTM 结构

LSTM 由 **输入门（Input Gate）、遗忘门（Forget Gate）和输出门（Output Gate）** 组成，并引入了 **细胞状态（Cell State）**，用于存储长期信息。

LSTM 的计算公式如下：

$$f_t = \sigma(W_f [h_{t-1}, x_t] + b_f)$$

$$i_t = \sigma(W_i [h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_t = \tanh(W_C [h_{t-1}, x_t] + b_C)$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t$$

$$o_t = \sigma(W_o [h_{t-1}, x_t] + b_o)$$

$$h_t = o_t \odot \tanh(C_t)$$

其中：

- f_t 控制遗忘过去的信息。
- i_t 控制新信息的写入。
- o_t 控制输出。
- C_t 是存储长期信息的细胞状态。

1.3 GRU（门控循环单元）

GRU（Gated Recurrent Unit）是 LSTM 的一种变体，结构更简洁，计算量更小，效果与 LSTM 相近。

GRU 结构

GRU 仅包含 **重置门（Reset Gate）和更新门（Update Gate）**，省略了 LSTM 的细胞状态。

GRU 的计算公式如下：

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

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

$$\tilde{h_t} = \tanh(W_h[r_t \odot h_{t-1}, x_t] + b_h)$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h_t}$$

其中：

- z_t 控制旧信息和新信息的混合程度。
- r_t 控制遗忘旧信息的程度。
- h_t 是最终的隐藏状态。

GRU 由于参数较少，计算效率较高，适用于对计算资源要求较高的任务。

2. 诗歌生成的过程

2.1 模型架构

2.1.1 词嵌入层（Word Embedding）

- **作用**：将离散的单词（或汉字）映射为连续的向量表示，使模型能够学习词汇之间的语义关系。
- **实现方式**：
  - 使用 `nn.Embedding` 层，随机初始化词向量矩阵。
  - 输入：单词索引（ `vocab_length` 维的 one-hot 向量）。
  - 输出： `embedding_dim` 维的稠密词向量。

2.1.2 LSTM 层（长短时记忆网络）

- **作用**：捕捉诗歌文本的长期依赖关系，学习语言模式。

- **实现方式**：
  - 采用双层 LSTM 结构，增强模型表达能力。
  - 输入：词嵌入向量（`(batch_size, seq_len, embedding_dim)`）。
  - 输出：每个时间步的隐藏状态（`(batch_size, seq_len, lstm_hidden_dim)`）。

2.1.3 全连接层（Linear + Softmax）

- **作用**：将 LSTM 的输出映射回词汇空间，计算每个词的概率分布。
- **实现方式**：
  - `nn.Linear` 层将 `lstm_hidden_dim` 维向量映射到 `vocab_length` 维。
  - `LogSoftmax` 计算对数概率，便于训练时使用负对数似然损失（NLLLoss）。

2.2. 训练过程

2.2.1 数据预处理

1. **诗歌语料库**：
   - 使用中文诗歌数据集。
   - 清洗数据，去除标点符号、特殊字符。
2. **构建词汇表**：
   - 统计所有出现的字（或词），建立 `word2idx` 和 `idx2word` 映射。
3. **数据编码**：
   - 将诗歌句子转换为数字索引序列。

2.2.2 训练策略

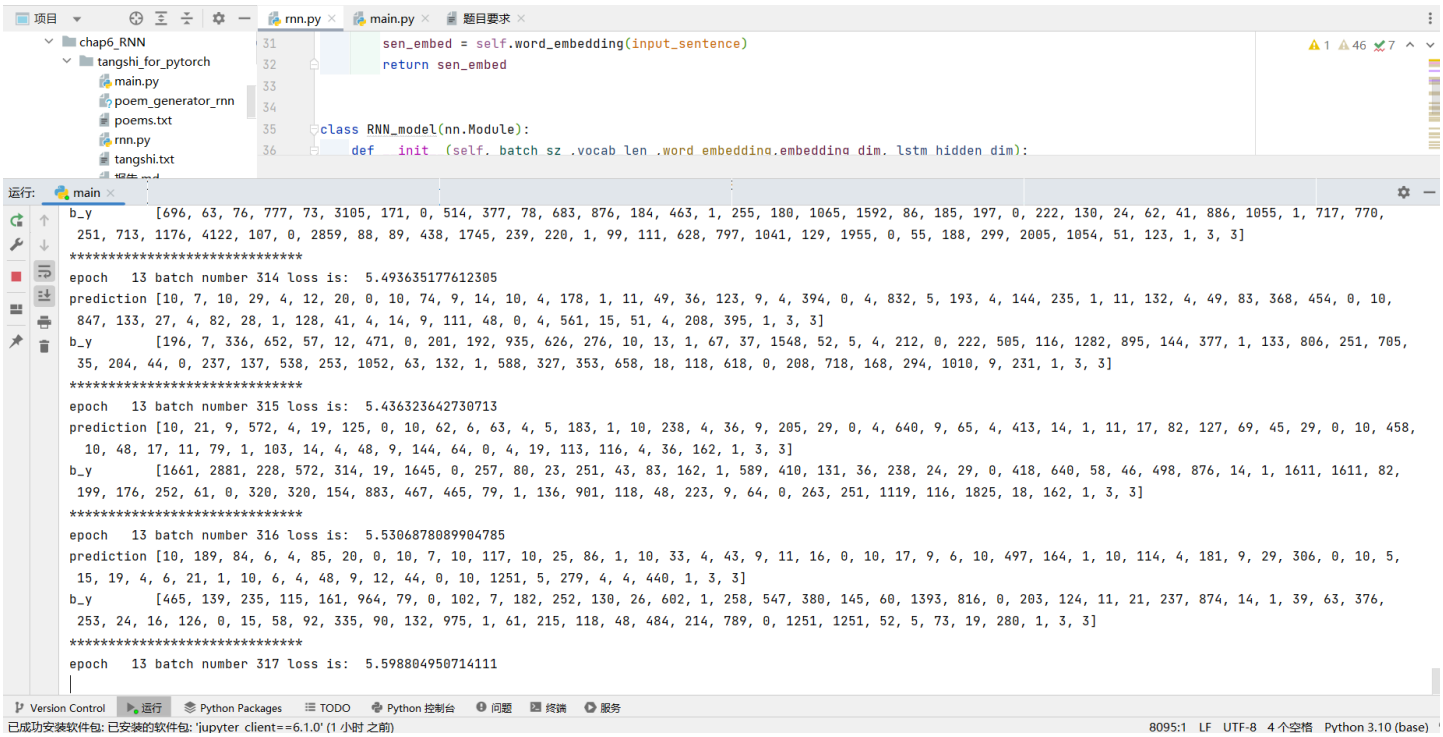
1. **损失函数**：
   - 采用 负对数似然损失，计算预测词和真实词之间的差距。
2. **优化器**：
   - 使用Adam 优化器，自适应调整学习率。
3. **训练方式**：
   - 训练时使用真实的上文词作为输入，而非模型生成的词，以加速收敛。
     -输入数据按 batch 组织，提高训练效率。

2.3. 生成阶段

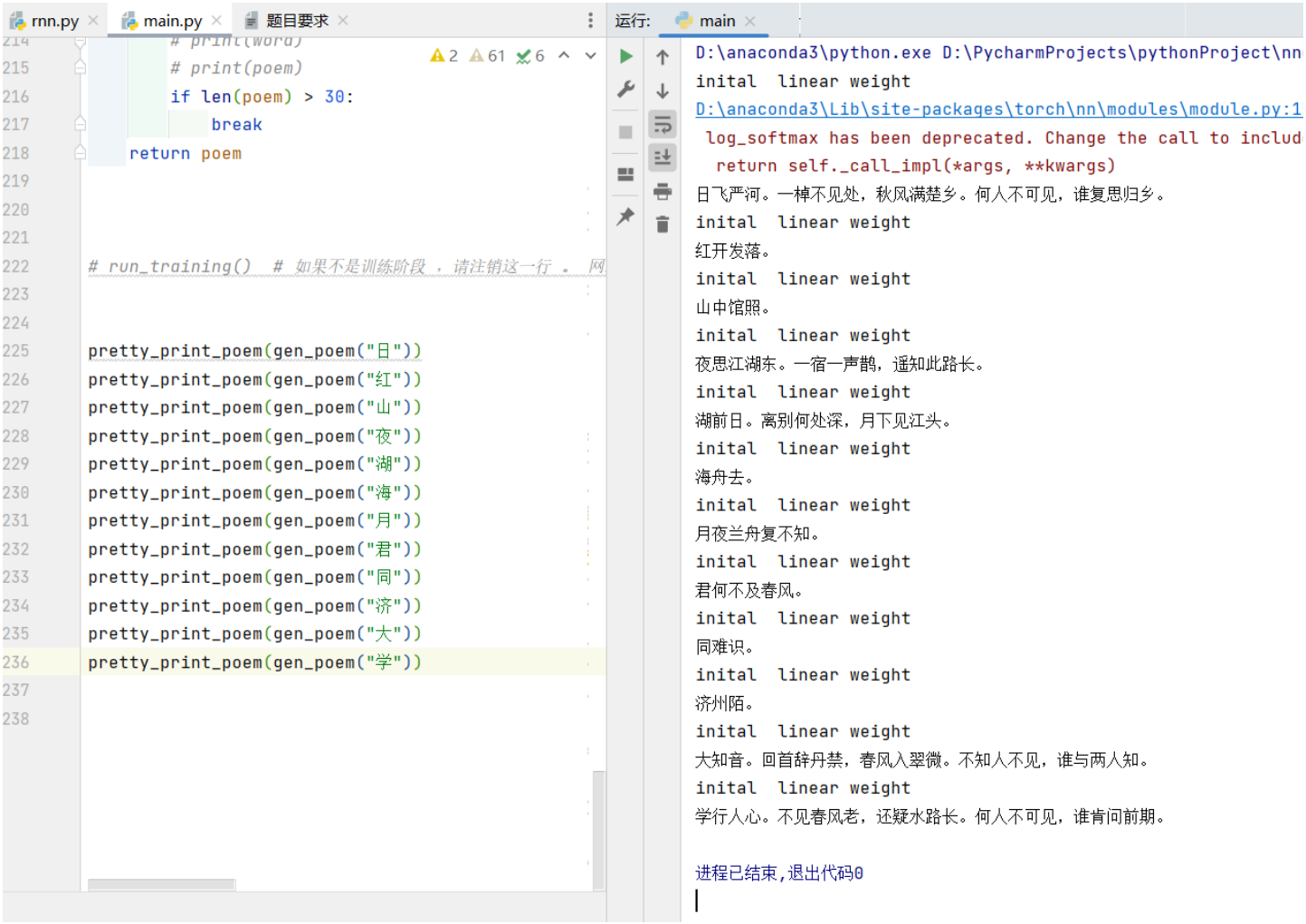
给定起始词，模型逐步预测下一个字，并作为新的输入。重复该过程，直到生成完整诗句或达到最大长度。

3. 生成结果

训练时的截图:


项目
chap6_RNN
tangshi_for_pytorch
main.py
poem_generator_rnn
poems.txt
rnn.py
tangshi.txt
rnn.py
main.py
题目要求
sen_embed = self.word_embedding(input_sentence)
return sen_embed
class RNN_model(nn.Module):
def __init__(self, batch_sz, vocab_len, word_embedding,embedding_dim, lstm_hidden_dim):
运行:
main
b_y [696, 63, 76, 777, 73, 3105, 171, 0, 514, 377, 78, 683, 876, 184, 463, 1, 255, 180, 1065, 1592, 86, 185, 197, 0, 222, 130, 24, 62, 41, 886, 1055, 1, 717, 770, 251, 713, 1176, 4122, 107, 0, 2859, 88, 89, 438, 1745, 239, 220, 1, 99, 111, 628, 797, 1041, 129, 1955, 0, 55, 188, 299, 2005, 1054, 51, 123, 1, 3, 3]
****
epoch 13 batch number 314 loss is: 5.493635177612305
prediction [10, 7, 10, 29, 4, 12, 20, 0, 10, 74, 9, 14, 10, 4, 178, 1, 11, 49, 36, 123, 9, 4, 394, 0, 4, 832, 5, 193, 4, 144, 235, 1, 11, 132, 4, 49, 83, 368, 454, 0, 10, 847, 133, 27, 4, 82, 28, 1, 128, 41, 4, 14, 9, 111, 48, 0, 4, 561, 15, 51, 4, 208, 395, 1, 3, 3]
b_y [196, 7, 336, 652, 57, 12, 471, 0, 201, 192, 935, 626, 276, 10, 13, 1, 67, 37, 1548, 52, 5, 4, 212, 0, 222, 505, 116, 1282, 895, 144, 377, 1, 133, 806, 251, 705, 35, 204, 44, 0, 237, 137, 538, 253, 1052, 63, 132, 1, 588, 327, 353, 658, 18, 118, 618, 0, 208, 718, 168, 294, 1010, 9, 231, 1, 3, 3]
****
epoch 13 batch number 315 loss is: 5.436323642730713
prediction [10, 21, 9, 572, 4, 19, 125, 0, 10, 62, 6, 63, 4, 5, 183, 1, 10, 238, 4, 36, 9, 205, 29, 0, 4, 640, 9, 65, 4, 413, 14, 1, 11, 17, 82, 127, 69, 45, 29, 0, 10, 458, 10, 48, 17, 11, 79, 1, 103, 14, 4, 48, 9, 144, 64, 0, 4, 19, 113, 116, 4, 36, 162, 1, 3, 3]
b_y [1661, 2881, 228, 572, 314, 19, 1645, 0, 257, 80, 23, 251, 43, 83, 162, 1, 589, 410, 131, 36, 238, 24, 29, 0, 418, 640, 58, 46, 498, 876, 14, 1, 1611, 1611, 82, 199, 176, 252, 61, 0, 320, 320, 154, 883, 467, 465, 79, 1, 136, 901, 118, 48, 223, 9, 64, 0, 263, 251, 1119, 116, 1825, 18, 162, 1, 3, 3]
****
epoch 13 batch number 316 loss is: 5.5306878089904785
prediction [10, 189, 84, 6, 4, 85, 20, 0, 10, 7, 10, 117, 10, 25, 86, 1, 10, 33, 4, 43, 9, 11, 16, 0, 10, 17, 9, 6, 10, 497, 164, 1, 10, 114, 4, 181, 9, 29, 306, 0, 10, 5, 15, 19, 4, 6, 21, 1, 10, 6, 4, 48, 9, 12, 44, 0, 10, 1251, 5, 279, 4, 4, 440, 1, 3, 3]
b_y [465, 139, 235, 115, 161, 964, 79, 0, 102, 7, 182, 252, 130, 26, 602, 1, 258, 547, 380, 145, 60, 1393, 816, 0, 203, 124, 11, 21, 237, 874, 14, 1, 39, 63, 376, 253, 24, 16, 126, 0, 15, 58, 92, 335, 90, 132, 975, 1, 61, 215, 118, 48, 484, 214, 789, 0, 1251, 1251, 52, 5, 73, 19, 280, 1, 3, 3]
****
epoch 13 batch number 317 loss is: 5.598804950714111
Version Control
运行
Python Packages
TODO
Python 控制台
问题
终端
服务
已成功安装软件包: 已安装的软件包: 'jupyter_client==6.1.0' (1 小时 之前)
8095:1 LF UTF-8 4 个空格 Python 3.10 (base)

生成结果图:


rnn.py
main.py
题目要求
# print(poem)
if len(poem) > 30:
break
return poem
# run_training() # 如果不是训练阶段，请注销这一行 。 网
pretty_print_poem(gen_poem("日"))
pretty_print_poem(gen_poem("红"))
pretty_print_poem(gen_poem("山"))
pretty_print_poem(gen_poem("夜"))
pretty_print_poem(gen_poem("湖"))
pretty_print_poem(gen_poem("海"))
pretty_print_poem(gen_poem("月"))
pretty_print_poem(gen_poem("君"))
pretty_print_poem(gen_poem("同"))
pretty_print_poem(gen_poem("济"))
pretty_print_poem(gen_poem("大"))
pretty_print_poem(gen_poem("学"))
运行:
main
D:\anaconda3\python.exe D:\PycharmProjects\pythonProject\nn
inital linear weight
D:\anaconda3\Lib\site-packages\torch\nn\modules\module.py:1
log_softmax has been deprecated. Change the call to includ
return self._call_impl(*args, **kwargs)
日飞严河。一棹不见处，秋风满楚乡。何人不可见，谁复思归乡。
inital linear weight
红开发落。
inital linear weight
山中馆照。
inital linear weight
夜思江湖东。一宿一声鹊，遥知此路长。
inital linear weight
湖前日。离别何处深，月下见江头。
inital linear weight
海舟去。
inital linear weight
月夜兰舟复不知。
inital linear weight
君何不及春风。
inital linear weight
同难识。
inital linear weight
济州陌。
inital linear weight
大知音。回首辞丹禁，春风入翠微。不知人不见，谁与两人知。
inital linear weight
学行人心。不见春风老，还疑水路长。何人不可见，谁肯问前期。
进程已结束,退出代码0

4. 总结

本次作业通过补充完善一个pytorch版本的基于LSTM的诗歌生成模型，采用词嵌入技术将汉字转化为稠密向量，通过双层LSTM网络学习诗歌的语义和韵律特征，最终使用全连接层和Softmax输出生成结果。实验使用负对数似然损失和Adam优化器进行训练。结果表明，该模型能够学习到古典诗歌的基本语言模式，可以生成具有一定连贯性和诗意的文本。