深圳大学期末考试试卷

开/闭卷 开卷 A/B 卷 _____

课程编号 1504660001 课序号 01 课程名称 自然语言处理 学分 2.5

命题人(签字)_____ 审题人(签字)_____ _____年____月____日

题号	一	二	三	四	五	六	七	八	九	十	基本题总分	附加题
得分												
评卷人												

座号_____ 学号_____ 姓名_____ 专业_____ 学院_____

（密封线内不答题）密封线

一、随堂测试 2 RNN 计算案例：字符级文本生成 (共 2 题，第 1 题 50 分，第 2 题 50 分，共 100 分)
（姓名： 学号： ）

给定一句话：The quick brown fox jumps over the lazy dog.

1. RNN 的前向传播：
1）输入数据：将句子 "The quick brown fox jumps" 转换为独热编码。

```
# 准备数据
sentence = "The quick brown fox jumps over the lazy dog."
# 按单词分割句子，保留标点符号
words = re.findall(r"\b\w+\b|[^\w\s]", sentence)
print(f"句子分割为单词: {words}")

# 创建多个训练样本：每个样本是(前缀序列，目标单词)
samples = []
for i in range(1, len(words)):
    prefix = words[:i]   # 前缀序列
    target = words[i]    # 目标单词
    samples.append((prefix, target))
for i, (prefix, target) in enumerate(samples):
    print(f"样本 {i+1}: 前缀={prefix}，目标={target}")

# 提取所有唯一单词并创建映射
vocab = sorted(list(set(words)))
word_to_idx = {word: i for i, word in enumerate(vocab)}
idx_to_word = {i: word for i, word in enumerate(vocab)}
vocab_size = len(vocab)

# 将单词序列转换为独热编码，并输出独热编码
def words_to_one_hot(words_seq, vocab_size, word_to_idx, verbose=False):
    """将单词序列转换为独热编码序列"""
    one_hot_vectors = []
    for word in words_seq:
```

```
        # 创建独热向量
        one_hot = np.eye(vocab_size)[word_to_idx[word]]
        one_hot_vectors.append(one_hot)

        # 如果需要，输出独热编码信息
        if verbose:
            idx = word_to_idx[word]
            print(f"   单词 '{word}' 的独热编码: {one_hot}")

    return np.array(one_hot_vectors)
```

2）初始化权重：设置输入权重 W1，隐藏层权重 W2，输出权重 W3。

```
# 初始化 RNN 权重
hidden_size = 16   # 隐藏层大小

# 输入到隐藏层的权重 (词汇表大小 × 隐藏层大小)
W1 = np.random.randn(vocab_size, hidden_size) * 0.01
# 隐藏层到隐藏层的权重 (隐藏层大小 × 隐藏层大小)
W2 = np.random.randn(hidden_size, hidden_size) * 0.01
# 隐藏层到输出层的权重 (隐藏层大小 × 词汇表大小)
W3 = np.random.randn(hidden_size, vocab_size) * 0.01

# 偏置项
b1 = np.zeros((1, hidden_size))   # 输入层到隐藏层的偏置
b2 = np.zeros((1, vocab_size))    # 隐藏层到输出层的偏置
```

2. 梯度计算：

1) 损失函数：使用交叉熵损失函数计算模型输出与真实标签之间的损失。

```
# RNN 前向传播
def rnn_forward(inputs, W1, W2, W3, b1, b2, hidden_size):
    """
    RNN 前向传播
    inputs: 输入序列的独热编码，形状为(seq_len, vocab_size)
    返回: 输出序列，隐藏状态序列
    """
    seq_len, vocab_size = inputs.shape
    hidden_states = np.zeros((seq_len + 1, hidden_size))   # 初始隐藏状态为 0
    outputs = np.zeros((seq_len, vocab_size))

    for t in range(seq_len):
        # 计算当前时间步的隐藏状态
        hidden_states[t+1] = np.tanh(
            np.dot(inputs[t], W1) +
            np.dot(hidden_states[t], W2) +
            b1
        )
```

```
        # 计算当前时间步的输出
        outputs[t] = np.dot(hidden_states[t+1], W3) + b2

    return outputs, hidden_states

# 交叉熵损失函数
def cross_entropy_loss(predictions, target):
    """计算交叉熵损失"""
    exp_preds = np.exp(predictions - np.max(predictions))  # 防止数值溢出
    probs = exp_preds / np.sum(exp_preds)
    loss = -np.sum(target * np.log(probs + 1e-10))  # 加小值防止 log(0)
    return loss, probs

# 反向传播
def rnn_backward(inputs, outputs, hidden_states, target, W2, W3, hidden_size):
    """RNN 反向传播计算梯度"""
    seq_len, vocab_size = inputs.shape
    dW1 = np.zeros_like(W1)
    dW2 = np.zeros_like(W2)
    dW3 = np.zeros_like(W3)
    db1 = np.zeros_like(b1)
    db2 = np.zeros_like(b2)

    # 最后一个时间步的输出误差
    exp_preds = np.exp(outputs[-1] - np.max(outputs[-1]))
    probs = exp_preds / np.sum(exp_preds)
    delta_output = probs - target  # 输出层误差

    # 隐藏层误差
    delta_hidden = np.dot(delta_output, W3.T) * (1 - hidden_states[-1]**2)
    delta_hidden = delta_hidden.reshape(1, -1)  # 确保是二维数组

    # 计算梯度
    dW3 += np.dot(hidden_states[-1].reshape(-1, 1), delta_output.reshape(1, -1))
    db2 += delta_output

    dW1 += np.dot(inputs[-1].reshape(-1, 1), delta_hidden)
    dW2 += np.dot(hidden_states[-2].reshape(-1, 1), delta_hidden)
    db1 += delta_hidden

    return dW1, dW2, dW3, db1, db2
```

2) 使用梯度下降法：使用学习率 η 更新权重更新权重

```
# 训练模型
learning_rate = 0.01
epochs = 1000  # 训练轮次

for epoch in range(epochs):
```

```
total_loss = 0

# 遍历所有样本进行训练
for prefix, target_word in samples:
    # 准备输入和目标（训练时不打印独热编码，避免输出过多）
    input_one_hot = words_to_one_hot(prefix, vocab_size, word_to_idx, verbose=False)
    target_idx = word_to_idx[target_word]
    target_one_hot = np.eye(vocab_size)[target_idx]

    # 前向传播
    outputs, hidden_states = rnn_forward(
        input_one_hot, W1, W2, W3, b1, b2, hidden_size
    )
    final_output = outputs[-1]

    # 计算损失
    loss, _ = cross_entropy_loss(final_output, target_one_hot)
    total_loss += loss

    # 反向传播计算梯度
    dW1, dW2, dW3, db1, db2 = rnn_backward(
        input_one_hot, outputs, hidden_states, target_one_hot, W2, W3, hidden_size
    )

    # 更新权重
    W1 -= learning_rate * dW1
    W2 -= learning_rate * dW2
    W3 -= learning_rate * dW3
    b1 -= learning_rate * db1
    b2 -= learning_rate * db2

# 每 100 轮打印一次损失
if (epoch + 1) % 100 == 0:
    print(f"轮次 {epoch+1}/{epochs}, 平均损失: {total_loss/len(samples):.4f}")
```

运行结果：

```
句子分割为单词: ['The', 'quick', 'brown', 'fox', 'jumps', 'over', 'the', 'lazy', 'dog', '.']

训练样本:
样本 1: 前缀=['The'], 目标=quick
样本 2: 前缀=['The', 'quick'], 目标=brown
样本 3: 前缀=['The', 'quick', 'brown'], 目标=fox
样本 4: 前缀=['The', 'quick', 'brown', 'fox'], 目标=jumps
样本 5: 前缀=['The', 'quick', 'brown', 'fox', 'jumps'], 目标=over
样本 6: 前缀=['The', 'quick', 'brown', 'fox', 'jumps', 'over'], 目标=the
样本 7: 前缀=['The', 'quick', 'brown', 'fox', 'jumps', 'over', 'the'], 目标=lazy
样本 8: 前缀=['The', 'quick', 'brown', 'fox', 'jumps', 'over', 'the', 'lazy'], 目标=dog
样本 9: 前缀=['The', 'quick', 'brown', 'fox', 'jumps', 'over', 'the', 'lazy', 'dog'], 目标=.

词汇表: ['.', 'The', 'brown', 'dog', 'fox', 'jumps', 'lazy', 'over', 'quick', 'the']
词汇表大小: 10
单词到索引的映射:
  '.': 0
  'The': 1
  'brown': 2
  'dog': 3
  'fox': 4
  'jumps': 5
  'lazy': 6
  'over': 7
  'quick': 8
  'the': 9
```

```
轮次 100/1000, 平均损失: 2.2499
轮次 200/1000, 平均损失: 2.1991
轮次 300/1000, 平均损失: 1.9324
轮次 400/1000, 平均损失: 0.5396
轮次 500/1000, 平均损失: 0.1114
轮次 600/1000, 平均损失: 0.0528
轮次 700/1000, 平均损失: 0.0335
轮次 800/1000, 平均损失: 0.0243
轮次 900/1000, 平均损失: 0.0189
轮次 1000/1000, 平均损失: 0.0154
```

```
预测测试及独热编码:

输入前缀: ['The']
  单词 'The' 的独热编码: [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
预测: quick, 实际: quick → 正确

输入前缀: ['The', 'quick']
  单词 'The' 的独热编码: [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
  单词 'quick' 的独热编码: [0. 0. 0. 0. 0. 0. 0. 0. 1. 0.]
预测: brown, 实际: brown → 正确

输入前缀: ['The', 'quick', 'brown']
  单词 'The' 的独热编码: [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
  单词 'quick' 的独热编码: [0. 0. 0. 0. 0. 0. 0. 0. 1. 0.]
  单词 'brown' 的独热编码: [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
预测: fox, 实际: fox → 正确

输入前缀: ['The', 'quick', 'brown', 'fox']
  单词 'The' 的独热编码: [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
  单词 'quick' 的独热编码: [0. 0. 0. 0. 0. 0. 0. 0. 1. 0.]
  单词 'brown' 的独热编码: [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
  单词 'fox' 的独热编码: [0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
预测: jumps, 实际: jumps → 正确

输入前缀: ['The', 'quick', 'brown', 'fox', 'jumps']
  单词 'The' 的独热编码: [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
  单词 'quick' 的独热编码: [0. 0. 0. 0. 0. 0. 0. 0. 1. 0.]
  单词 'brown' 的独热编码: [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
  单词 'fox' 的独热编码: [0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
  单词 'jumps' 的独热编码: [0. 0. 0. 0. 0. 1. 0. 0. 0. 0.]
预测: over, 实际: over → 正确
```

```
输入前缀: ['The', 'quick', 'brown', 'fox', 'jumps', 'over']
  单词 'The' 的独热编码: [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
  单词 'quick' 的独热编码: [0. 0. 0. 0. 0. 0. 0. 0. 1. 0.]
  单词 'brown' 的独热编码: [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
  单词 'fox' 的独热编码: [0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
  单词 'jumps' 的独热编码: [0. 0. 0. 0. 0. 1. 0. 0. 0. 0.]
  单词 'over' 的独热编码: [0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]
预测: the, 实际: the → 正确

输入前缀: ['The', 'quick', 'brown', 'fox', 'jumps', 'over', 'the']
  单词 'The' 的独热编码: [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
  单词 'quick' 的独热编码: [0. 0. 0. 0. 0. 0. 0. 0. 1. 0.]
  单词 'brown' 的独热编码: [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
  单词 'fox' 的独热编码: [0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
  单词 'jumps' 的独热编码: [0. 0. 0. 0. 0. 1. 0. 0. 0. 0.]
  单词 'over' 的独热编码: [0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]
  单词 'the' 的独热编码: [0. 0. 0. 0. 0. 0. 0. 0. 0. 1.]
预测: lazy, 实际: lazy → 正确

输入前缀: ['The', 'quick', 'brown', 'fox', 'jumps', 'over', 'the', 'lazy']
  单词 'The' 的独热编码: [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
  单词 'quick' 的独热编码: [0. 0. 0. 0. 0. 0. 0. 0. 1. 0.]
  单词 'brown' 的独热编码: [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
  单词 'fox' 的独热编码: [0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
  单词 'jumps' 的独热编码: [0. 0. 0. 0. 0. 1. 0. 0. 0. 0.]
  单词 'over' 的独热编码: [0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]
  单词 'the' 的独热编码: [0. 0. 0. 0. 0. 0. 0. 0. 0. 1.]
  单词 'lazy' 的独热编码: [0. 0. 0. 0. 0. 0. 1. 0. 0. 0.]
预测: dog, 实际: dog → 正确

输入前缀: ['The', 'quick', 'brown', 'fox', 'jumps', 'over', 'the', 'lazy', 'dog']
  单词 'The' 的独热编码: [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
  单词 'quick' 的独热编码: [0. 0. 0. 0. 0. 0. 0. 0. 1. 0.]
  单词 'brown' 的独热编码: [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
  单词 'fox' 的独热编码: [0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
  单词 'jumps' 的独热编码: [0. 0. 0. 0. 0. 1. 0. 0. 0. 0.]
  单词 'over' 的独热编码: [0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]
  单词 'the' 的独热编码: [0. 0. 0. 0. 0. 0. 0. 0. 0. 1.]
  单词 'lazy' 的独热编码: [0. 0. 0. 0. 0. 0. 1. 0. 0. 0.]
  单词 'dog' 的独热编码: [0. 0. 0. 1. 0. 0. 0. 0. 0. 0.]
预测: ., 实际: . → 正确
```