自然语言处理实验报告

基于RNN与CNN的词向量训练与对比分析

姓名：张宗梓

学号：2023K8009991013

专业：人工智能

2025 年 12 月 21 日

目录

1 实验目的与任务

本实验旨在通过构建神经网络语言模型，利用《人民日报》语料库训练汉语词向量，并对比分析循环神经网络（RNN）与卷积神经网络（CNN）在特征提取和词向量生成上的差异。

主要任务包括：

1. 数据预处理：清洗语料，构建词表，将文本转换为索引序列。
2. 模型构建：分别搭建 RNN 和 CNN 模型，完成下一词预测任务。
3. 词向量提取：从 Embedding 层获取词向量。
4. 对比分析：计算并对比两个模型生成的词向量在语义相似度上的表现。

2 实验原理与设置

2.1 模型架构

2.1.1 RNN 模型

RNN 模型通过捕捉序列的时间依赖性来学习语义。本实验采用单向 RNN，结构如下：

- **Embedding 层**：将词索引映射为稠密向量。
- **RNN 层**：处理输入序列，保留上下文信息。
- **全连接层**：将最后一个时间步的隐状态映射为词表概率分布。

2.1.2 CNN 模型

CNN 模型通过滑动窗口提取局部特征。本实验结构如下：

- **Embedding 层**：同上。
- **Conv1d 层**：一维卷积，核大小为 2 (类似于 Bi-gram 特征)。
- **Max Pooling 层**：全局最大池化，提取最显著特征。
- **全连接层**：输出预测概率。

2.2 实验参数设置

表 1: 实验超参数设置

参数名	数值
词表大小 (Vocab Size)	1000
词向量维度 (Embed Dim)	16
输入序列长度 (Seq Len)	5
批大小 (Batch Size)	64
学习率 (Learning Rate)	0.001
训练轮数 (Epochs)	15

3 实验步骤与具体实现

3.1 数据预处理

利用 Python 对《人民日报》1998年1月语料进行清洗。

- **去噪：**剔除词性标记、数字、标点符号及非中文字符，仅保留纯汉字实词。
- **截断：**根据词频统计，保留前 1000 个高频词，其余标记为 `<UNK>`。
- **数据集构建：**采用滑动窗口法，取前 5 个词预测第 6 个词。

3.2 核心代码实现

以下为 RNN 模型定义的关键代码片段：

```
class RNNModel(nn.Module):
    def __init__(self, vocab_size, embed_dim, hidden_dim):
        super(RNNModel, self).__init__()
        self.embedding = nn.Embedding(vocab_size, embed_dim)
        self.rnn = nn.RNN(embed_dim, hidden_dim, batch_first=True)
        self.fc = nn.Linear(hidden_dim, vocab_size)

    def forward(self, x):
        embeds = self.embedding(x)
        out, _ = self.rnn(embeds)
        return self.fc(out[:, -1, :])
```

4 实验结果与分析

4.1 训练过程对比

在训练过程中观察到，CNN 模型的 Loss 收敛速度较快，说明其在提取局部固定搭配（如“经济发展”）上具有优势；而 RNN 模型在初期震荡后逐渐平稳，最终两者的 Loss 均收敛至 3.0 左右。

4.2 词向量相似度分析 (Top-10)

利用训练好的词向量计算余弦相似度 (Cosine Similarity)，随机选取目标词进行对比分析。

表 2: RNN与CNN词向量相似词召回对比

目标词	RNN Top-5 相似词	CNN Top-5 相似词
中国	美国, 国家, 地区, 发展, 关系	北京, 国际, 合作, 世界, 建设
经济	建设, 社会, 发展, 改革, 市场	金融, 增长, 技术, 结构, 全球
工作	会议, 任务, 活动, 领导, 部门	决定, 精神, 组织, 开展, 深入

4.3 结果讨论

1. **语义捕捉能力**：从结果可以看出，RNN 和 CNN 都能学习到一定的语义信息。例如“中国”的近义词中出现了“国家”、“北京”等词汇，符合直觉。

2. **模型差异**：

   - **RNN** 倾向于找到**语法功能相似**的词（如名词找名词，动词找动词），这是因为它在时序预测中学习到了词性规律。
   - **CNN** 倾向于找到**局部共现频率高**的词，受限于窗口大小（Window Size=2），它更关注短语搭配。

3. **局限性分析**：由于词表仅限制在 1000 词，且过滤了大量低频词，导致部分目标词的相似词列表中出现了不相关的通用词（如“进行”、“的”等）。这属于 OOV (Out-Of-Vocabulary) 带来的信息损失，可以通过扩大词表和增加训练语料来改善。

5 实验总结

本次实验成功利用 PyTorch 框架实现了基于 RNN 和 CNN 的语言模型，并提取了汉语词向量。实验表明，即便是在较小的数据集和简单的模型结构下，神经网络依然能够通过上下文预测任务有效地捕捉词汇之间的语义关系。通过对比发现，CNN 在训练效率上略优于 RNN，而 RNN 在处理长距离序列依赖上理论上具有优势。

附录：文件清单

- `corpus_cleaned.txt`: 清洗后的语料库
- `vocab.json`: 词表字典
- `train_rnn.py`: RNN 训练脚本
- `train_cnn.py`: CNN 训练脚本
- `analyze.py`: 相似度分析脚本
- `model/`: 存放训练好的模型与向量文件