人工智能导论
情感分析实验报告

齐强 2017011436 计 75

2019 年 6 月 2 日

目录

1 快速使用

1. 根据 src 文件夹下的 requirements.txt 进行相关库的安装
2. 文件准备
   (a) 数据预处理: 准备 sinanew.test sinanews.train sgns.sogounews.bigram-char 在 data 文件夹下
   (b) 跳过预处理: 保证预处理好的文件
3. 数据预处理：python3 pre-process.py (也可跳过此步，直接使用预处理好的文件)
4. 训练：(1) 修改 common.py 中的 mode 参数为'TRAIN' (2)python3 main.py
5. 测试：(1) 修改 common.py 中的 mode 参数为'TEST' (2)python3 main.py

6. 其他实验参数选项，包括 net 类型，batch_size, 训练 epoch 数量等，都可以通过更改 common.py 文件中的 Config 类进行设置

7. 多个实验时通过 git worktree 管理，可以通过 newexp.sh 脚本较快的创建新实验，参数保存在 data/res/文件夹中创建对应 branch 名称的文件夹

8. note: 如果想使用之前训练好的模型,需要保证 main.py 所在的文件夹名称与 data/res/下对应模型所在文件夹的名称相同

2 文件结构

2.1 src

1. pre-process.py: 对原始数据和词向量进行一定程度的数据处理，方便模型训练

2. common.py: 包含通用的设置，包括预处理文件和模型的路径、模型参数、训练 or 测试模式选择等

3. model.py:

   (a) Class MLP: 全连接网络结构, 运用两层全连接，主要参数为隐藏层 hidden_dim, 本次试验设为 500

   (b) Class CNN: 卷积网络结构, 运用两层卷积，一层池化，一层全连接，以及 dropout 层与 relu 层

   (c) Class LSTM: RNN 网络结构, 调用 nn.LSTM, 主要参数 hidden_dim,num_layers, 本次进行了两组 (150,2),(50,2) 实验

4. train.py 包含 eval, save, train 三个主要部分，在 main.py 中进行调用，对不同模型实现同一套训练、测试逻辑

5. main.py: 根据 config 参数选择进行训练或测试，进行 net 创建，load 已有 net 模型，开始训练或测试等

2.2 data

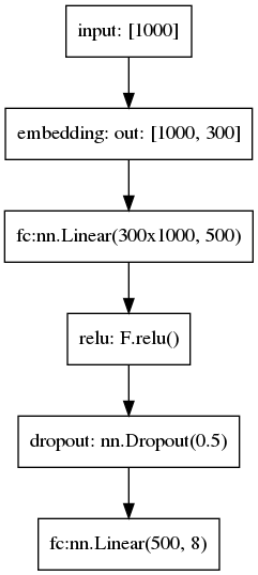
1. 原始文件：sinanew.test sinanews.train sgns.sogounews.bigram-char

2. 预处理文件: train/valid/test.arti2idx/labels.pt, weight.pt

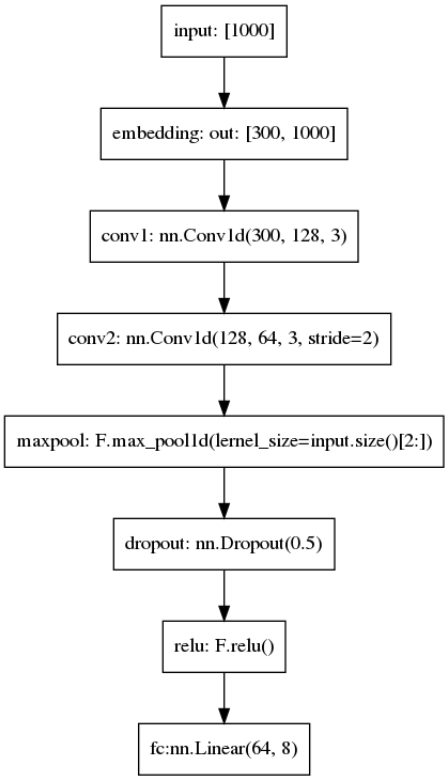
3. res: 保存各实验的训练结果，每个实验文件夹包含 best.pt curr.pt(current)

3 模型结构图与流程分析

3.1 MLP(baseline)



3.2 CNN



3.3 LSTM

4 实验结果

exp	accuracy(best)	f-score	correlation
lstm.hidden150.layer2	0.50965(0.56563)	0.27041	0.53842
lstm.hidden50.layer2	0.52068(0.53125)	0.14918	0.55862
cnn.layer2.128.64	0.53585(0.57188)	0.26962	0.56633
mlp.hidden500(baseline)	0.51838(0.55625)	0.27558	0.55282

5 参数调节

1. lr:learning rate 的大小常用的范围是 1e-2 1e-5，在实验过程中发现 lr 越小，模型训练过程中 loss 和 accuracy 变化越平缓稳定, 但是 lr 过小过大都会导致训练难以达到最佳位置并收敛，本实验中选定 lr=1e-3

2. batch_size: batch size：训练过程中，batch 的大小也会影响模型的效果，当 batch size 小于 16 的时候，模型容易出现局部过拟合，导致整体效果不理想, 受电脑内存限制，选定 batch_size = 64

3. kernal_size: 选定 kernel_size = 3

4. epoch: 在训练过程中每隔一定 steps 对验证集进行测试，当发现在验证集上收敛时，停止训练，发现 cnn, mlp epoch 大概在 500 左右即可，rnn epoch50 左右即可收敛

6 baseline 模型效果差异分析

实现了 baseline, 但是因为表现原因，并没有分析出什么..

7 问题思考

1. epoch 数量：使用固定迭代次数可能会出现过拟合现象，导致模型的准确率很低, 通过观察 test_loss 的收敛情况可以较好控制次数

2. 参数初始化：使用框架为 torch, 会在网络构建时自动根据输入网络尺寸进行参数初始化

3. 防止过拟合：

   (a) 通过 dropout 层，进行反向传播时随机失活，通过调整 dropout 参数进行调节

   (b) 对训练数据进行 shuffle 减小过拟合，数据量小是容易过拟合的重要原因

4. 分析 CNN,RNN, 全连接神经网络 (MLP) 三者的优缺点：

   (a) MLP 比 CNN、RNN 简单，较少的全连接层时，它的训练速度也更快一些，得到的最终结果也相对差一些。

   (b) CNN 更多的是从局部信息聚合得到整体信息，需要采用固定大小的输入和输出

   (c) RNN 在时间维度上有个先后顺序，可以处理任意长度的输入、输出

8 总结收获

1. 问题：对于 python 文档的使用还是缺少一定经验，正确的使用方式能够大大减小学习成本；对于神经网络这套理论很不熟悉，导致学习的很吃力
2. 学习了 pytorch 框架，明白了 CNN, RNN, MLP 的原理，感谢老师与助教的帮助