情感分析 实验报告

计 82 郑凯文 20181011314

2020 年 5 月 31 日

1 Model Introduction

情感分析（Sentiment Analysis）是 NLP 领域的研究重点之一。本实验解决的是其中的情感分类（Sentiment Classification）问题，具体而言是将一篇已分词的文章归为八个类别中的一个：感动、同情、无聊、愤怒、搞笑、难过、新奇、温馨。

1.1 Datasets Manipulation

使用词嵌入将原数据集预处理为词向量构成的列表，并保存在 json 文件中。模型训练时，直接加载 json 文件而无需多次转化。即，使用静态的词向量，而不在训练过程中对词向量进行调整。事实证明，这样无论对于提升准确率还是减缓过拟合均有帮助。

1.1.1 Pretrained Vectors for Word-embedding

使用`https://github.com/Embedding/Chinese-Word-Vectors` 提供的预训练词向量用于嵌入。其中

- 表示方式（Representation）为 SGNS（Skip-Gram with Negative Sampling）
- 语料库（Corpus）为搜狗新闻（Sogou News）
- 上下文特征（Context Features）为 Word
- 维数为 300

1.1.2 Missing Words

纵使上述 GB 量级的词向量库拥有 32 万左右的词，文章的分词仍有可能在库中不出现。对于不存在的词，可以将其填充为 0 或随机一个库中的向量代替。本实验采用后者。

1.1.3 Ambiguous Labels

这是一个多分类问题，而提供的标签信息只有用户的投票。自然可以使用 majority voting 来确定 groud truth，但面临票数最多的标签不唯一的情况。此时，提供的信息是不足以甄别其真实类别，无论用哪一个标签进行训练均有失偏颇。在本实验中，这样的样本被直接舍弃。

1.2 Netword Architecture

本实验以 MLP 作为 baseline，除此之外实现了 CNN、RNN、BiLSTM。对于每个模型，均选用了一种代表性网络结构。

1.2.1 MLP

在并不复杂的任务上，深层神经网络由于难以训练往往表现弱于浅层神经网络。本实验使用了简易的网络结构

- Flatten
- FC(in_channels=300*padding, out_channels=512)
- Sigmoid
- Dropout(dropout_rate)
- FC(in_channels=512, out_channels=8)
- Softmax

1.2.2 CNN

如图 1所示，采取的 CNN 结构为 TextCNN

- conv1=Conv2d(in_channels=1, out_channels=100, kernel=(3,300))
- conv2=Conv2d(in_channels=1, out_channels=100, kernel=(4,300))
- conv3=Conv2d(in_channels=1, out_channels=100, kernel=(5,300))
- x1=MaxPool1d(conv1(x))
- x2=MaxPool1d(conv2(x))
- x3=MaxPool1d(conv3(x))

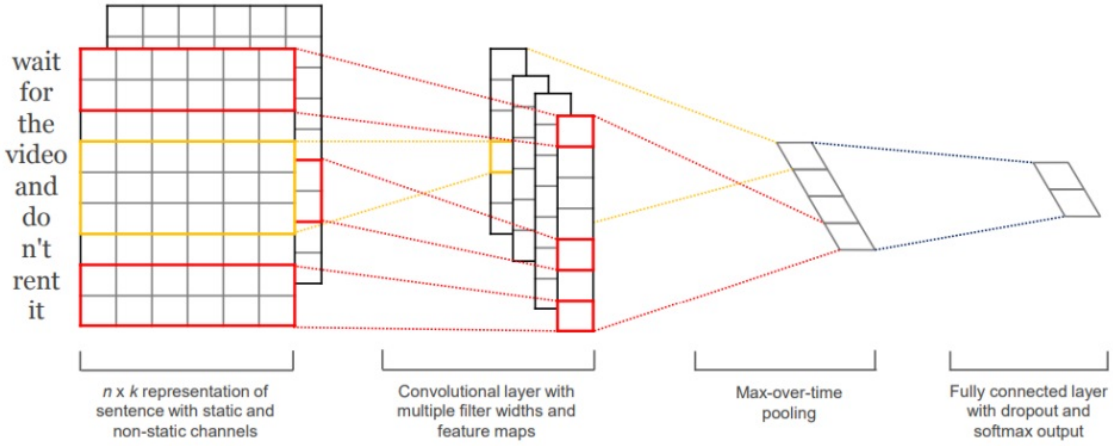

图 1: TextCNN

- Concate(Flatten(x1), Flatten(x2), Flatten(x3))
- Dropout(dropout_rate)
- FC(in_channels=300, out_channels=8)
- ReLU
- Softmax

1.2.3 RNN

利用 Pytorch 自带的 RNN 单元，接上全连接层

- RNN(hidden_size=64, num_layers=2, dropout=dropout_rate)
- FC(in_channels=64*padding, out_channels=8)
- Softmax

1.2.4 BiLSTM

利用 Pytorch 自带的双向 LSTM 单元，接上全连接层

- BiLSTM(hidden_size=64, num_layers=2, dropout=dropout_rate)

- FC(in_channels=64*padding*2, out_channels=8)
- Softmax

1.3 Implementation Details

1.3.1 Padding

由于是序列数据，文章的长度并不是固定的，而这给 Pytorch 中 Dataloader 的使用和 mini-batch 的训练方式带来诸多不便。于是设定一个 padding 值，长度小于 padding 的文章被 0 填充至 padding 长度，多余的部分则被截断。

对于 MLP 和 CNN，padding 设置为 800，这保证了大部分文章不被截断。对于 RNN 和 BiLSTM，由于模型本身参数极多，设置 padding 为 300 并对部分文章进行裁剪以加速训练和减小内存/显存占用。经过测试，保留 300 的长度不会损失过多信息，甚至对测试准确率有帮助。

1.3.2 Loss Function

使用交叉熵作为损失函数，这是两种分布间距离的常用衡量方式。我也尝试了 MSE，但效果不佳。

1.3.3 Class Weights

由于训练集的类别十分不平衡，在训练中极易陷入局部极小，模型将大多数输入的分类预测为愤怒。在 CNN、MLP 中，我通过合理大小的正则化解决了这一问题；而在 RNN 和 BiLSTM 中，梯度消失的问题依旧存在。

为使模型更愿意输出训练集中较为稀有的类别，在 RNN 和 BiLSTM 中，我在损失函数中为每类分配了权重，反比于在训练集中出现的次数。在 CNN、MLP 中，带权与否对分类准确率影响不大，最终采用不带权的损失用于最终训练。

1.3.4 Optimizer

优化器的选择上,SGD、Momentum 和 Nesterov 是基本和略作修改的梯度下降法。Adadelta、RMSprop、Adam、Adamax 等则进行了各种策略的学习率自适应调整，整体效果相差不大。本实验使用了 Adam 优化器，学习率为 Pytorch 默认的推荐值 1e-3。

1.3.5 Seed

随机种子决定着权重的初始化。不同的种子下，模型的表现会有略微差异（个别情况有较大差异）。为保证结果的可复现性，设置种子为 0。实验在 GPU 上进行，若使用 CPU 结果可能不同，随机因素也可能导致复现失败，但大体而言结果应相差不大。

1.3.6 Regularization

正则化是防止模型过拟合的方法，本实验中主要通过优化器的 weight decay 以及 dropout 层的 rate 来控制。weight decay 对应于 L2 正则化，经尝试对于 CNN、RNN 和 BiLSTM 设置为固定值 0.05，而 MLP 由于参数较多，0.05 会导致欠拟合，因此改为 0.03。Dropout Rate 进行多种取值下的调参。

1.3.7 Training Methods

严格来讲，数据集应被划分为训练集、验证集和测试集，使用验证集评估不同模型的好坏以及评估训练进程，测试集用于最终测试。但此次实验提供的数据集存在严重的类别不平衡、样本不足的情况，在测试集中划分出验证集的成本过高。因此，本实验将训练轮数固定为 100 个 epoch，并在每个 epoch 后计算模型在测试集上的各种指标，持续跟踪模型训练情况。训练时 batch size 为 64。

2 Hyperparameter Tuning

由于模型训练（尤其是 RNN、LSTM）十分耗时，这里只对 Dropout Rate 进行粗粒度的调参。对于其它超参数，简单地固定为较为合理和常见的取值。

2.1 Dropout Rate

Dropout 作为和正则化类似的方法，用以降低模型复杂度，预防过拟合。Dropout Rate 较低时，模型过拟合较为严重，随训练轮数增加，在测试集上的性能逐渐下降。Dropout Rate 过高时，模型则训练缓慢，难以收敛。取 Dropout Rate 为 0、0.2、0.4、0.6、0.8，四种模型在测试集上的准确率如表 1。其中，RNN 和 BiLSTM 的准确率在训练中波动较大，但最高不超过 56%。为公平起见，这里只以第 100 个 epoch 时的表现为准。

表 1: 四种模型在不同 Dropout Rate 下第 100 个 epoch 的测试准确率

模型 \ Rate	0	0.2	0.4	0.6	0.8
MLP	56.53%	54.71%	49.44%	55.92%	**57.54%**
CNN	62.20%	**64.88%**	64.17%	64.07%	63.46%
RNN(weighted)	50.00%	40.23%	42.46%	51.27%	**51.77%**
BiLSTM(weighted)	51.97%	50.71%	**52.68%**	52.13%	50.71%

3 Results and Analysis

3.1 Experiment Results

对于四种模型，取固定初始条件下，调参过程中测试集准确率最高的 Dropout Rate 用于最终训练和评估。结果如表 2所示。

表 2: 四种模型在第 100 个 epoch 的评价指标

模型 \ 指标	Accuracy	Macro F1 Score	Pearson Correlation Coefficient
MLP	57.54%	0.1663	0.6366
CNN	64.88%	0.3116	0.6911
RNN(weighted)	51.77%	0.2805	0.5841
BiLSTM(weighted)	52.68%	0.3203	0.5910

3.2 Learning Curve

学习曲线是一种判断拟合情况的简易方法。在图 2和图 3中，给出了 CNN 模型和 MLP 模型在最优 Dropout Rate 下每轮的训练损失和测试损失。可以看到，CNN 中发生了过拟合现象，这是由于较低的 Dropout Rate。而 MLP 由于 Dropout Rate 较高，主要发生的是欠拟合，且波动较大。一个重要的事实是，并非开始发生过拟合时就应该停止训练，过拟合发生后，测试集的损失仍有可能继续下降（如 CNN 的学习曲线所显示的）。发生了过拟合和欠拟合的模型也不代表性能不好，实践中应从验证集上的表现来判别。

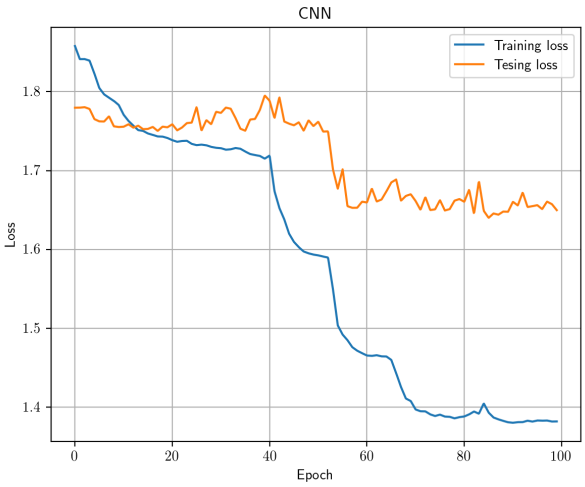

图 2: CNN 的学习曲线

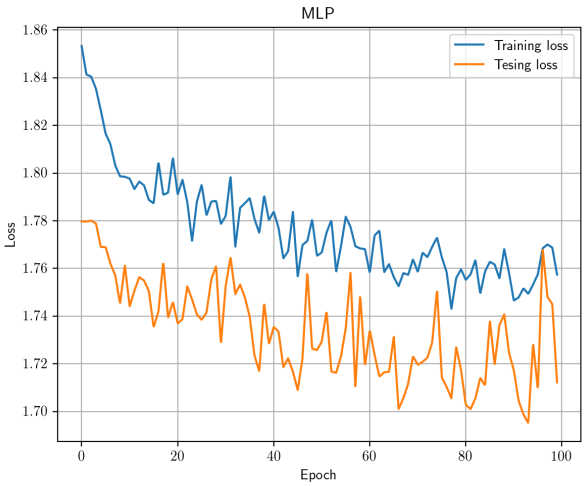

图 3: MLP 的学习曲线

3.3 Comparison and Analysis

从结果来看，TextCNN 模型的效果是最好的，其相比 MLP 具有明显的优势，这是由于其权值共享的结构使得网络具有更少的参数量，更易学习，且更容易抓住和提取局部特征。

最令人诧异的是，以处理序列数据著称的 RNN 系列甚至准确率不如 MLP，我推测这是由于情感分析中，上下文信息没那么重要，往往是情感色彩很强的局部决定着最终的分类。此外，BiLSTM 相比于普通 RNN，准确率更加稳定。而 RNN 系列相比 CNN 和 MLP，虽然准确率差了不少，但 F1 Score 却毫不逊色，这说明 RNN 系列模型没有那么“盲目”地一股脑预测为愤怒，也进一步表明了训练集和测试集具有严重的类别不平衡问题，使得模型难以训练，准确率很难提升。

4 Questions

1. **实验训练什么时候停止是最合适的？简要陈述你的实现方式，并试分析固定迭代次数与通过验证集调整等方法的优缺点。**

   在机器学习的训练过程中，往往通过训练集和验证集上的损失来判断拟合情况和决定停止时刻：当训练集上的损失小于验证集时，模型开始过拟合；当验证集损失不再下降时，需及时终止训练。本实验中，由于样本数不足和覆盖不全面，没有进行验证集的划分，而是进行固定轮数的训练。我尝试对模型进行较长轮数的训练，但由于模型较小，在 100 个 epoch 以内足以收敛，且更多轮数会导致过拟合。

固定迭代次数较为简单，但对于一般的模型，难以确定最优轮数，模型可能欠拟合或过拟合，错过训练过程中的最优解。通过验证集调整具有较强的自适应性，能够在陷入过拟合时进行 early stop，但模型性能可能在波动或略有降低后才逐渐跳出局部极小，达到更优的解，这种方法同样会错过最优解。

本实验中，固定训练轮数为 100，主要发生的是过拟合的情况。为此，我将模型在每轮后在训练集和测试集上的损失记录下来，持续跟踪模型性能的变化，这样也易于对比分析。

2. **实验参数的初始化是怎么做的？不同的方法适合哪些地方？（现有的初始化方法为零均值初始化，高斯分布初始化，正交初始化等）**

   本实验中采用的是 Pytorch 的默认初始化方式，即采用 Kaiming Uniform 按照与维数相关的均匀分布初始化参数 Tensor。对于其它初始化方式，我没有进行进一步尝试。

   需要注意的是，神经网络的权重不可进行 0 初始化，这会使模型的参数永远对称，大幅降低表达能力。非对称的参数初始化方式可以做到 symmetry breaking，零均值初始化、Xavier Normal、Xavier Uniform、Kaiming Normal、Kaiming Uniform、正交初始化都是常用的选择。对于 RNN 系列，由于易出现梯度消失和梯度爆炸，可以采用正交初始化或 Xavier 这种保证了方差一致性的方法。

3. **过拟合是深度学习常见的问题，有什么方法可以方式训练过程陷入过拟合。**

   增大训练集、正则化（L1、L2）、Dropout、降低参数量、优先增加模型深度而非宽度、数据增强、early stop。

4. **试分析 CNN，RNN，全连接神经网络（MLP）三者的优缺点。**

   MLP 稠密、参数量大、训练和收敛较快，但容易陷入过拟合。

   CNN 通过权值共享降低参数量和预防过拟合，卷积运算便于提取局部特征，但训练速度略慢于 MLP。

   RNN 适合处理不同长度的序列信息，但参数数量过多，训练极慢，且容易产生梯度消失和梯度爆炸（从 RNN 和 BiLSTM 不进行 reweight 便会被困在局部极小，将几乎所有输入分类为愤怒即可看出）。

5 Summary

本次实验主要在较为简易的模型结构下进行调参，使我体会了“玄学炼丹”的滋味。但即使随机种子不同、超参数可以调节，带来的对模型性能的改变远小于不同模型结构间的固有差

异，如 CNN 在每种 Dropout Rate 下均有 60% 以上的准确度，其它模型难以简单通过调参企及。

情感分析是 NLP 中的重要主题，但 65% 的最高准确率在应用中也几乎没什么意义。就我的观点而言，这主要是样本不足、类别不平衡的训练集质量过低，难以启发有价值的结论，60% 多的准确率也是这个训练集上的极限了。