情感分类作业报告

计 81 包涵 2018011289

2020 年 5 月 29 日

1 模型简介

该实验中，我实现了 Multilayer Perceptron (MLP)、Convolutional Neural Network (CNN)、Recurrent Neural Network (RNN)、Deep Pyramid CNN (DpCNN) 和 Attention-Based RNN (AttRnn) 几个网络。以 MLP 作为实验的 baseline，我主要对前两个网络进行了调整和测试，出于好奇对后两个网络进行了简单的测试。

在每个模型中，均使用了预训练的 300 维中文词向量 (Li et al. 2018)，对中文词语进行编码。由于新闻长度不一，对于每条新闻预处理成了相同的长度（下称 pad_size），新闻较长则截断，不够长则做 padding。每个模型的 pad_size 可以不同。于是每条新闻就被处理成 “pad_size* 向量维度” 大小的矩阵。对于标签，我使用了将最大值转为单标签预测的方法，所以每个网络的输出神经元数量与分类数目相同，并取 activation 最大的神经元作为输入新闻的分类结果。

下面对几个网络模型进行简单的介绍。

1.1 Multilayer Perceptron

该网络的结构如下图所示

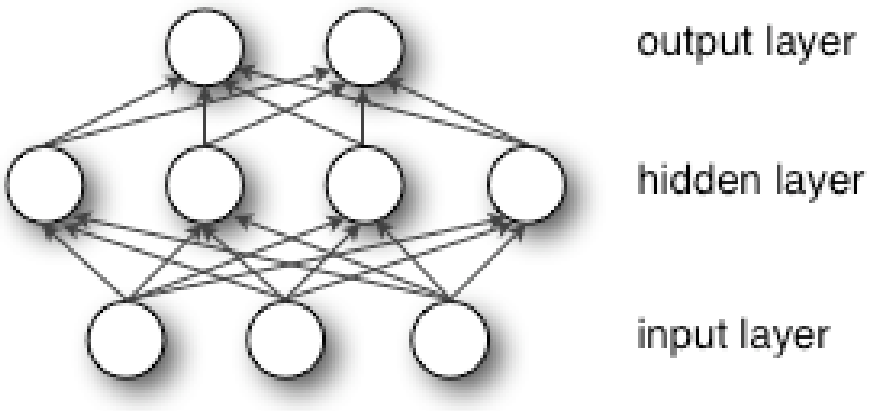

图 1: MLP Network Structure

如图所示，网络由输入层、隐层和输出层三层构成。输入层与隐层、隐层与输出层均为全连接。输出层的数量与需要分类的类别数相同，隐层的维度可以调节。将经过预处理的新闻，排列成 1 维向量，作为输入层。经过第一个全连接层后，对输出做 ReLU，再输入第二个全连接层。为了降低 overfitting 的情况，对于隐层的神经元做了 dropout 的处理。

1.2 Convolutional Neural Network

网络结构如下图所示 (Kim 2014)

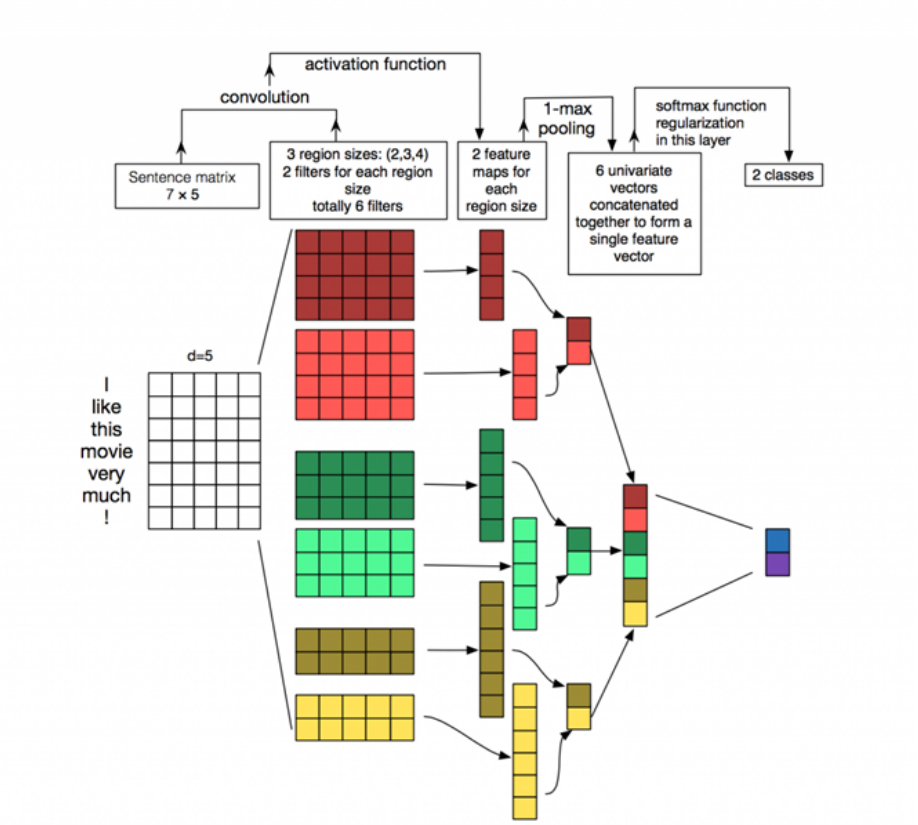

图 2: CNN Structure

该网络有若干个不同高度的 kernel，将输入的词向量矩阵沿着行方向进行卷积，得到一系列的列向量。将相同维度的 kernel 生成的列向量进行池化操作并排列成一个列向量，最后通过一个全连接层输出。实际使用中，省略了池化操作，而是将卷积后得到的所有列向量直接通过一个全连接层输出。

1.3 Recurrent Neural Network

该网络的结构如下图所示 (Liu, Qiu, and Huang 2016)

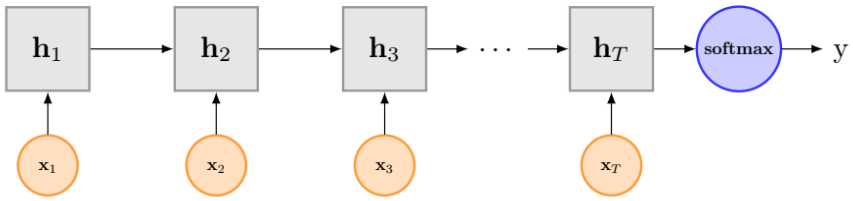

图 3: RNN Structure

该网络是典型的 RNN，如图为展开后的网络示意图，h_i 代表同一个 LSTM 网络，其输出维度为预设的隐层维度。x_i 是输入词向量矩阵的第 i 行，即为新闻中的第 i 个词。将新闻中的每个词依次输入该网络后，得到正向和反向的两个特征向量。随后生成分类结果的部分在上图中没有表示出来，具体过程为将两个特征向量与词向量连接，通过一个卷积和最大池化网络，再通过全连接层输出。

1.4 Deep Pyramid CNN

该网络的结构如下图所示 (Johnson and Zhang 2017)

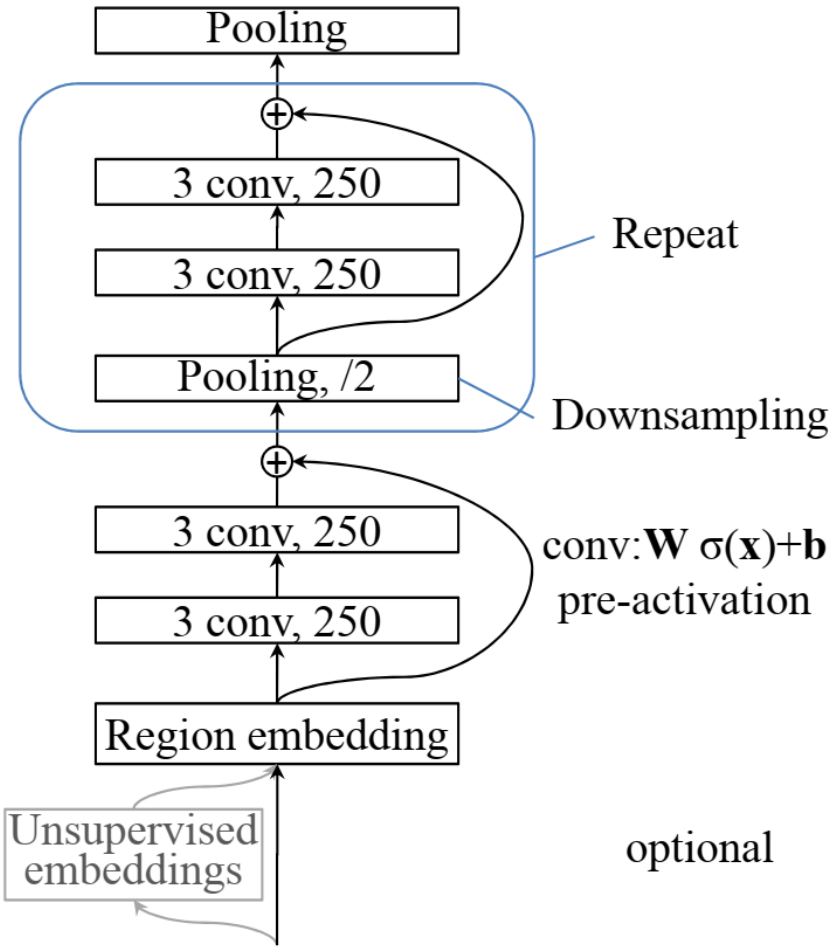

图 4: Deep Pyramid CNN Structure

输入的词向量首先经过 Region embedding 的卷积层，kernel 大小为”(3, 词向量维度)”，输出特征维度为 250。在经过两个 kernel 为”(3,1)” 的卷积层后，进入 Downsampling 阶段，重复经过 kernel 为”(3,1)”, stride 为 2 的最大池化层和两个 kernel 为”(3,1)” 的卷积层，直到每个特征层的维度是 2。为了确保维度恰好降为 2，对 pad_size 有一定要求。然后将特征层依次排列，经过一个全连接层输出。通过 Downsampling 阶段，网络高效地提取输入语料中更高层次的信息，并且可以较方便地用于不同长度的输入。

1.5 Attention-Based RNN

该网络的结构如下图所示 (Zhou et al. 2016)

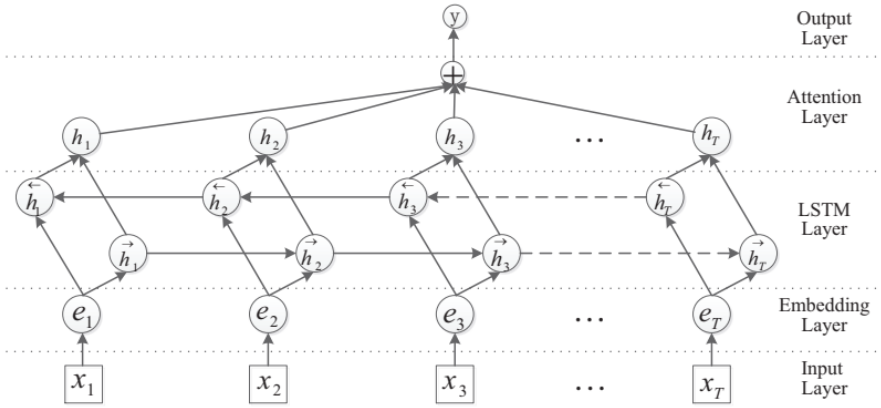

图 5: Attention-Based RNN Structure

该网络首先使用双向的 LSTM 层，把输入转为词级别的特征。将正向和反向的到的两个特征相加，得到 Attention 层的输入。然后对输入计算加权的向量和，再通过全连接层输出。通过计算词级别特征的加权向量和，网络可以学习到句子级别的特征，从而可能提高性能。

2 实验效果

在 test 集上的实验结果如下：

模型	准确率 (%)	F1-宏平均 (%)	F1-微平均 (%)	相关系数
MLP	52.87	0.1940	0.5287	0.3087
CNN	58.75	0.2581	0.5875	0.4027
RNN	58.66	0.2948	0.5866	0.3982
DpCNN	56.46	0.2206	0.5646	0.3377
AttRNN	57.45	0.2320	0.5745	0.3751

可以看出经过参数调整的 CNN 和 RNN 网络表现最好，AttRNN 网络在使用了和 RNN 网络类似的参数的情况下，表现也不错。DpCNN 表现稍

差，一方面可能由于网络深度较大，而训练集较小，所以网络得不到充分的训练；另一方面可能和没有调参有关系。MLP 网络作为 baseline，即便经过参数调整，与其余更复杂的网络结构的效果仍有较大差距。

各个评价指标得出的结论基本一直，除了采用宏平均的 F1 值。RNN 网络虽然准确率等略低于 CNN 网络，但是宏平均的 F1 值比 CNN 高很多。这可能是因为数据集非常不平衡，CNN 网络为了降低 loss，对“同情”和“温馨”两类的学习不充分，没有得到任何正类，所以 F1 值都未定义，设为了 0，因而经过算术平均之后，F1 的宏平均较低。而 RNN 网络只有“温馨”类没有正类，所以宏平均 F1 较高。可见如果模型对于某个类别的识别能力很差，宏平均的 F1 值会偏低，而微平均的 F1 值相对反应模型各个类的整体识别能力。并且可以看出，对于分类问题，微平均的 F1 值与准确率是一样的。

所给训练和测试数据类的分布非常不均衡。以训练数据为例，类分布为

类别	数量	比例 (%)
无聊	145	6.2
温馨	27	1.2
愤怒	984	42.0
搞笑	367	15.7
难过	180	7.7
同情	124	5.3
感动	416	17.8
新奇	99	4.2

训练中使用对交叉熵损失函数加入权重进行了调整，但是效果不明显。

3 参数调整

只针对 MLP，CNN 和 RNN 进行了参数调整的实验。由于没有固定随机数的 seed，每次实验都有一定的随机性，结果可能有一定波动，但是能反应基本的趋势。

3.1 Learning Rate

Learning rate 控制 BP 算法中对权重更新的步长，对于获得好的学习效果非常重要。这次作业中我使用了固定的 learning rate，并且对各个模型

进行了测试。这里以 CNN 模型的测试数据为例进行说明。Learning rate 对训练过程中 loss 的影响如下图所示，

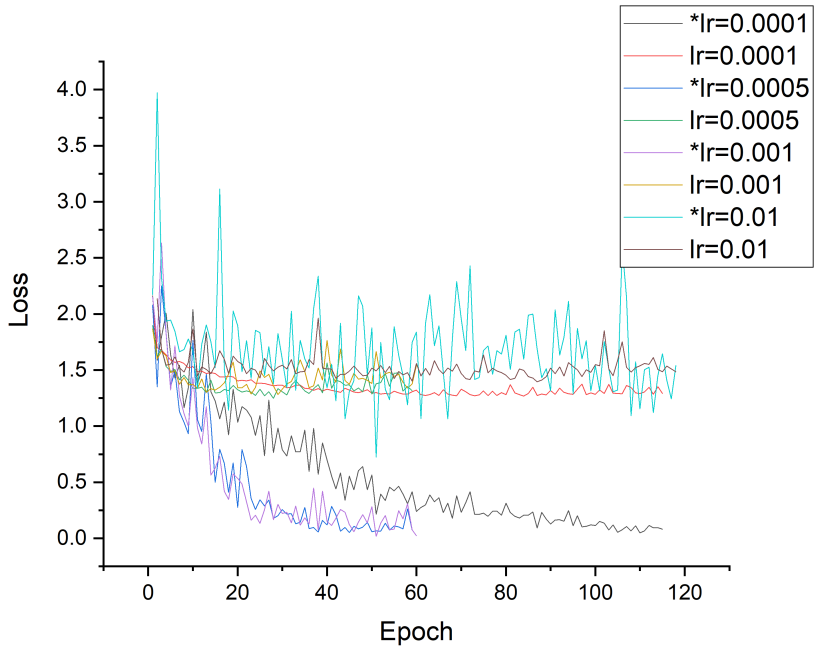

图 6: CNN 网络 Loss 与 lr 的关系

其中带有 “*” 号的是训练集上的 loss，没有 “*” 号的是验证集上的 loss。各条曲线截至的 Epoch 数不同，因为训练停止的条件是验证集的 loss 超过 1000 个 batch 没有下降。

Learning rate 对于测试集准确率的影响如下表所示

Learning Rate	准确率
0.0001	0.5844
0.0005	0.5925
0.001	0.5871
0.01	0.5620
0.1	0.4704

Learning rate 为 0.1 时的 loss 曲线波动过大，没有画在图 6 中。

可以看出，当 learning rate 较大时，loss 的波动很大，而且模型很难收敛，最终的训练效果也较差。当 learning reate 设置合理时，模型可以较快收敛，并且达到较好的训练效果。

3.2 文本截取长度 (pad_size)

文本截取的长度如果过短，模型不能得到足够的信息，对文本做出准确的判断。如果截取长度过长，训练时间会增加，存储压力也较大，而且对于长度小于截取长度的文本，需要增加 padding，可能影响分类准确率。对训练数据的文本词数统计如下：

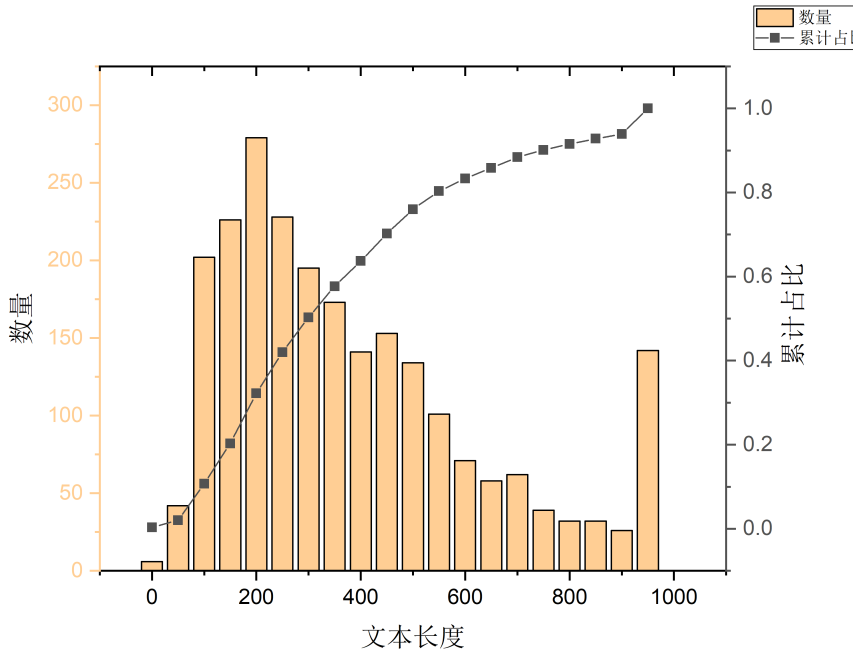

图 7: 新闻词数的频数分布

可以看出，长度为 200 左右的新闻数量较多，50% 的新闻长度都在 300 以下。

对于进行了测试的网络，pad_size 与模型准确率的关系如下表所示: (第一行为 pad_size，表中数据为准确率)

模型	50	100	200	300	500
MLP	52.65	51.75	52.38	52.42	47.76
CNN	57.45	58.39	57.90	58.66	58.62
RNN	57.23	58.17	57.50	58.98	58.12

对于 MLP 模型，pad_size 过大时，可能由于模型无法收敛，训练效果很差，其余长度对于模型效果没有明显影响。对于 CNN 和 RNN 模型，从数据中可以看出，较大的 pad_size 对模型表现没有负面的影响，而且 pad_size 在 300 左右时应当可以让模型达到理想的效果，这与新闻的长度分布也相对应。所以在各个模型中均取 300 左右的 pad_size.

3.3 Dropout Rate

Dropout 用于降低过拟合的情况，但是由于训练集很小，即便使用 dropout，也不能避免过拟合，尤其对于简单的 MLP 网络，如下图所示：

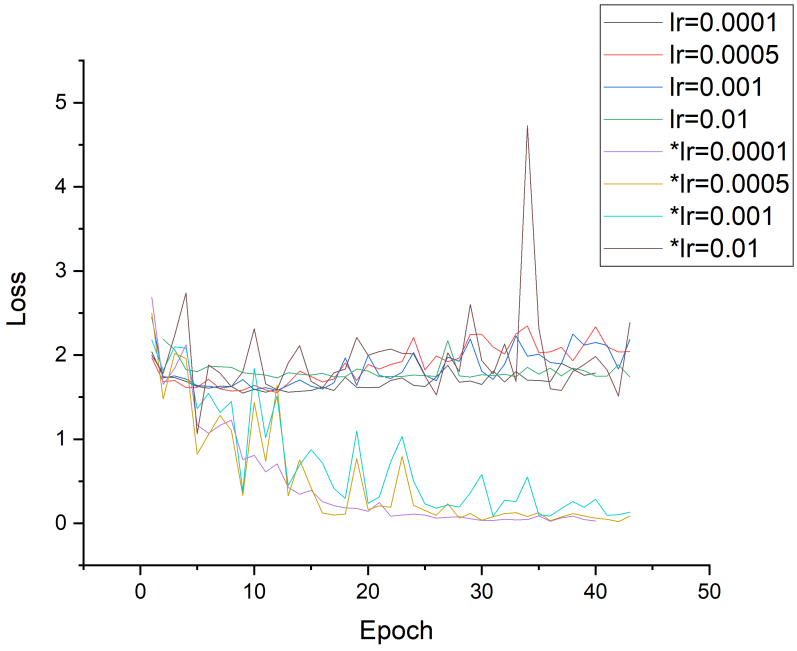

图 8: MLP 网络 Loss 与 lr 的关系

训练时都使用了 0.5 的 dropout rate. 但是仍然有较严重的过拟合问题。对于各个模型，dropout rate 与准确率的关系如下表所示：

模型	0.3	0.4	0.5	0.6	0.7
MLP	51.89	51.53	53.37	52.51	52.96
CNN	57.54	58.98	58.44	58.89	58.12
RNN	58.39	58.98	58.44	58.35	55.16

由于模型的结构不同，dropout 对于模型训练的影响也不尽相同，难以总结出规律。但是可以看出对于特定的问题和模型，设置合适的 dropout 可以提高模型准确率。如果不能进行大量的测试，那么 0.5 左右的 dropout 应该比较安全。

3.4 CNN: Kernel Sizes

CNN 网络使用不同大小的 kernel，试图获取句子层面的信息。kernel 的大小对于网络表现应该有一定影响。我测试了几个不同大小的 kernel，结果在下面的表格中展示。其中"Kernel sizes" 一列，以"(2,3,4)" 为例，意为使用大小为 2*300, 3*300 和 4*300 的 kernel 进行卷积。

Kernel sizes	准确率
(2,3,4)	58.71
(2,4,8)	58.84
(2,3,6,12)	57.41

该任务的文本长度较长，所以适当增大 kernel size 对于模型效果有提高。但是过大的 kernel size 对于准确率可能有负面的影响，这可能是因为过大的 kernel size 会把空间上相隔较远的词放在一起进行卷积，导致很难学习出规律，影响模型表现。

3.5 Hidden Layer Size

隐层在 CNN 之外各个网络中都有使用，隐层维度也是设计网络时非常重要的一个因素。这里以 RNN 和 MLP 网络为例，进行隐层维度对模型准确率的实验。

RNN 模型实验结果：

模型	准确率
32	57.23
64	58.08
128	58.39
192	57.94

MLP 模型实验结果：

模型	准确率
50	52.33
100	52.96
150	51.97
200	47.93
250	48.02

两个模型都呈现了相似的趋势，就是隐层维度适中时模型表现较好，过大时模型表现下降明显。隐层维度过高时，模型不易收敛，特别是对于训练数据量较小的情况。隐层维度高也会引入更多的参数，提高训练模型的成本。RNN 模型在 LSTM 结构之后，信息的综合层次已经较高，所以需要的隐层维度较低，就可以得到好的效果；而 MLP 模型由于需要隐层来综合输入词向量的信息，所以需要更高的隐层维度来达到效果。

4 模型比较

使用各个模型测试结果中，各类的 F1 值对模型表现进行更细致的比较。

类别	MLP	CNN	RNN
感动	0.5398	0.6521	0.6658
同情	0.0000	0.0000	0.0345
无聊	0.0305	0.1611	0.1351
愤怒	0.6873	0.7328	0.7429
搞笑	0.1030	0.1611	0.2523
难过	0.0893	0.1333	0.2714
新奇	0.1026	0.2247	0.2564
温馨	0.0000	0.0000	0.0000

MLP 和 CNN 都未能识别“同情”和“温馨”两类，MLP 比 CNN 在各个类的 F1 值都低，说明 MLP 对各类的识别能力都比较差。RNN 可以正确识别少量的“同情”类，而且在权重比较大的类中（愤怒、搞笑和难过），识别能力要强于 CNN. 综合来看，CNN 和 RNN 的分类效果各有优势，都强于 MLP.

5 问题思考

5.1 训练停止的方式

应当在模型拟合训练集较好，又没有过拟合时，也就是模型泛化能力最强的时候停止训练。在本次作业中，我使用训练集训练模型，并每隔若干个 batch 在验证集上测试模型表现。如果模型在验证集上的 loss 超过 1000 个 batch 没有下降，那么停止训练。

固定迭代次数的方法实现较为简单，缺点是为了寻找合适的迭代次数需要多次实验来探索。验证集调整的方法需要记录验证结果，并根据模型在验证集的表现决定是否继续训练。该方法更加灵活，可以更有效地避免过拟合或者训练不足。缺点是实现稍复杂，停止条件设置不合理会导致训练效果差。

5.2 参数初始化

实验使用了 Pytorch 中 Xavier Normal 初始化的方法。从调整了标准差的正态分布中随机选取权重。这里使用 RNN 网络，对 Xavier, Kaiming 和普通高斯初始化几种方法进行了简单的实验。使用每种初始化方法，对 RNN 网络进行三次训练，记录准确率和迭代次数的平均值如下：

初始化方法	准确率	迭代次数
Xavier Normal	58.83	18
Kaiming Uniform	58.95	16.33
Normal	训练失败	训练失败

从 RNN 网络来看，Kaiming Uniform 的初始化方法似乎可以让网络更快收敛，并且的准确率有 marginal 的提高。

Xavier 在初始化权重时，考虑了梯度消失和对称的激活函数带来的影响，应该更适合主要使用对称激活函数（如 tanh 等）的网络。Kaiming 的

方法考虑了非对称激活函数（如 ReLU）的影响，可能更适合主要使用这些函数的网络。高斯分布初始化的方法适合深度较浅的网络，否则容易产生梯度消失的问题。

5.3 避免过拟合

主要的方法有：

- 在网络中加入 dropout，随机抑制某些神经元。
- 使用验证集来提前停止训练，避免过拟合。
- 在损失函数中加入 regularization 项。
- 数据增强，对于图像的训练效果较好。对图像进行对称变换、扭曲、添加噪声等操作，让网络的泛化能力更强。

5.4 网络的优缺点

CNN 网络只能使用固定的窗口大小来提取上下文的信息，然后再通过池化合并，通过全连接输出。这个方法训练速度快，效果也较好，但是窗口大小的选择需要测试。而且由于 CNN 只能接受固定长度的文本，需要对文本进行截取操作，对长文本的效果可能变差。

RNN 网络利用输入的时序性的特点，可以利用前后文的关系，在局部总结出有用的信息，非常适合于语言处理、时间序列分析等任务。缺点在于训练速度慢。

MLP 网络全连接的特点，决定了每个神经元都对输入的整体信息进行综合，当输入内容少且局部性不强时比较适合。但是当输入内容多，信息层次低时，MLP 网络由于深度有限，无法高效地提取输入中高层次的信息，所以难以有较好的表现。本次实验中，MLP 网络没有着重利用信息的局部性，即上下文，训练速度快但是表现一般。

6 总结

通过本次实验，我学习并使用了深度学习框架 Pytorch，体验了处理数据、搭建神经网络、训练和调整参数的过程，收获良多。即加深了对神经网络的理解，也增加了训练神经网络的经验。神经网络属于经验性较强的领

域，既需要理论基础的支撑，又需要大量的实验来探索，只有积累了足够的知识和经验，才能在面对新问题时，又快又好地解决它。深度学习现今作为一个重要的方法，在计算机的各个分支以及其他领域都有广泛的应用，我们也应当学习掌握，利用好这个工具。

References

Johnson, Rie and Tong Zhang (2017). "Deep pyramid convolutional neural networks for text categorization". In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 562–570.

Kim, Yoon (2014). *Convolutional Neural Networks for Sentence Classification*. arXiv: 1408.5882 [cs.CL].

Li, Shen et al. (2018). "Analogical Reasoning on Chinese Morphological and Semantic Relations". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Melbourne, Australia: Association for Computational Linguistics, pp. 138–143. URL: http://aclweb.org/anthology/P18-2023.

Liu, Pengfei, Xipeng Qiu, and Xuanjing Huang (2016). *Recurrent Neural Network for Text Classification with Multi-Task Learning*. arXiv: 1605.05101 [cs.CL].

Zhou, Peng et al. (2016). "Attention-based bidirectional long short-term memory networks for relation classification". In: *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 2: Short papers)*, pp. 207–212.