大数据与深度学习应用

NLP 项目分类问题

李梓铉

学号 3118103163

指导老师: 胡飞虎教授

2019 年 5 月 31 日

1 问题背景

1.1 背景介绍

文本分类作为 NLP 的常见任务，具有很高的实际应用价值。本文将采用 LSTM 模型，训练一个能够识别文本不同类别项目描述的分类器。项目描述包含项目名称，项目简介，主要产品，业主方，和所在地。项目类别包含 7 类如建筑化工等。本文没有考虑不同项目描述之间的区别，直接将所有描述整合为同一类进行分类。意思就是将项目名称项目简介等对应的 string 连接为同一个 string 作为输入。本文使用 keras 深度学习库实现了基本的 LSTM 做分类任务的流程，但由于时间有限的只是完成了一个标准的流程，并熟悉了调参的过程。在本文提到的模型之上，还想到有一些可以改进的地方，以后有空的话，笔者可以再做优化，比如针对不同描述各训练一个模型并进行投票得到一个集成模型。

（Keras 是一个高层神经网络 API，Keras 由纯 Python 编写而成并基 Tensorflow、Theano 以及 CNTK 后端。Keras 为支持快速实验而生，能够把 idea 迅速转换为结果，）

如何使用训练好的模型请看 README.md

2 解题思路

2.1 文本表示与特征提取

使用 word2vec 算法，用高维向量（词向量，Word Embedding）表示词语，并把相近意思的词语放在相近的位置，而且用的是实数向量（不局限于整数）。我们只需要有大量的某语言的语料，就可以用它来训练模型，获得词向量。从而实现文本的表示与特征提取，并且词向量很方便做聚类，定义距离即可得到相近意思的词语，通过训练得到词向量的转化，就可以将句子分词后得到多个词向量并使用 LSTM 处理。或者用传统方法如 SVM 直接根据词向量进行分类。本文使用 jieba 分词与 Gensim 库实现分词和词向量的处理。

2.2 分类算法

分类算法部分使用了两种分类算法，第一种是支持向量机，对将句子转化为词向量后直接使用 SVM 求距离直接进行分类。第二类是 LSTM 神经网络。将矩阵形式的输入（原始句子）转化为多个较低维度一维向量（词向量），并利用 LSTM 的特性，使用遗忘门控制训练时候梯度的收敛性的同时，也能够保持长期的记忆性，实现语义的理解分类。

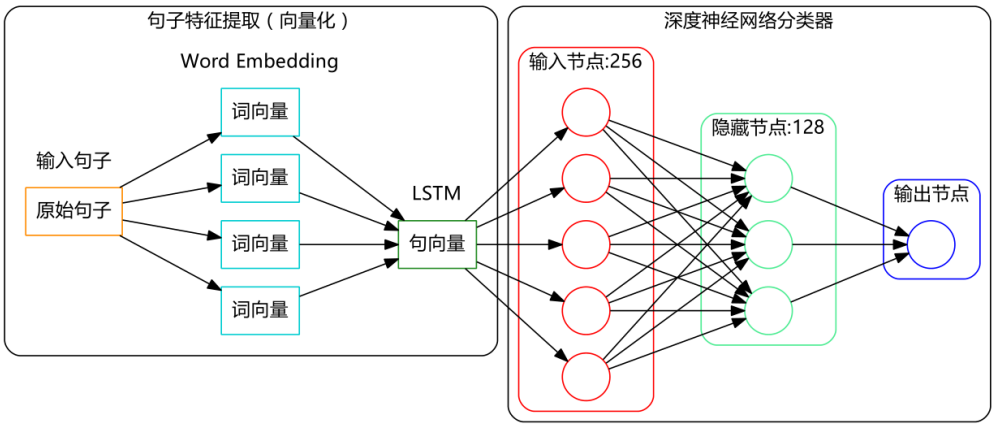


图 1: LSTM 流程图

3 结果与讨论

3.1 支持向量机 SVM

去掉包含空值 nah 的数据共有 2405 条可用数据，训练集包含 1923 个数据，测试集包含 481 个数据，个数比为 4：1，共经过 159 次迭代后，SVM 分类可以在达到 0.7 的分类准确度。

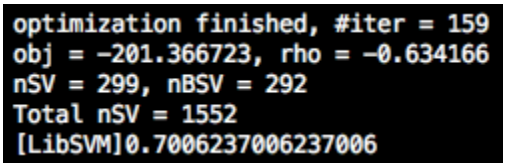


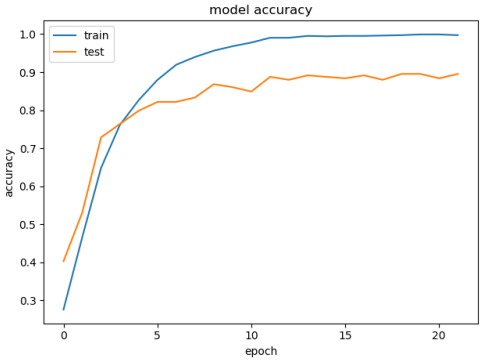
图 2: SVM 结果

这一步可以确定词向量的训练应该是没有问题的，并且对深度学习的效果提了一个 benchmark，至少高于 70% 的准确率才可以说明这个训练的神经网络是足够高效的。

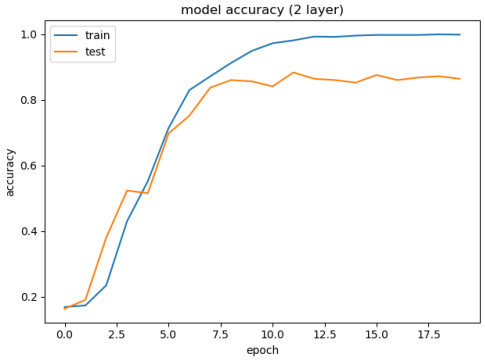
3.2 长短时记忆网络 LSTM

数据预处理部分考虑不同类别的项目个数区别过大会影响模型的准确性，因此在预处理的时候，统计了每个类别的个数，并以个数最少的那类项目的个数为基准，在其他类中取同样个数的数据来构成训练集。经统计 [建筑', ' 化工医药', ' 能源矿产', ' 材料制造', ' 其他', ' 电力', ' 交通运输'] 类别对应个数为 [344, 209, 280, 183, 446, 706, 236]，因此取最少的类别个数 183，每一类中取 183 个数据，得到共 1281 个数据，按 4:1 划分训练集和测试集。

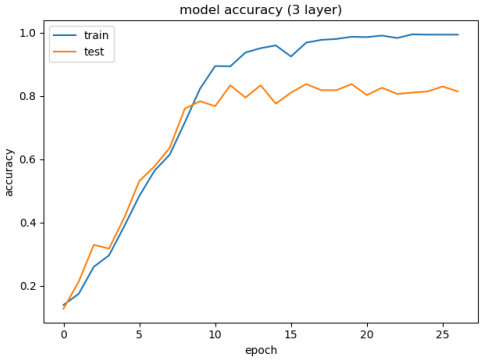
lstm 激活函数使用 sigmoid，并使用 EarlyStopping 回调函数（效果是在 val_accuracy 10 个 epoch 后仍然不上升的情况下终止训练），以及 dropout 层减轻过拟合。



(a) 1 layer LSTM



(b) 2 layer LSTM



(c) 3 layer LSTM

图 3: 不同层数 LSTM 训练过程

调参主要从不同词向量个数，不同网络层数，不同 dropout 参数。通过调参得到如下训练结果。训练结果按照分类准确度从高到低排列如下。颜色表示模型的优劣，绿色最优，红色最差。

神经元层数与每层个数	词向量个数	drop层	分类准确度
1layer256	256voc	0.3drop	0.9031
1layer256	512voc	0.3drop	0.899
1layer256	512voc	0.6drop	0.895348837
1layer512	512voc	0.6drop	0.895348837
2layer256,128	256voc	0.3drop	0.88372093
2layer128,128	256voc	0.3drop	0.875968992
2layer512,512	512voc	0.6drop	0.864341085
1layer32	64voc	0.3drop	0.841085271
3layer256,256,256	256voc	0.3drop	0.813953488

图 4: LSTM 计算结果总结

从训练结果可以看出

- drop out 对于收敛过程有影响，但基本不影响模型最终的结果
- 词向量个数不可以太少，增加词向量个数可以在文本转化过程中损失更少的信息，当调到 256 个以上后，对准确度提升不再明显
- 单层神经元就可以实现很好的分类效果，在本例中增加神经层数反而会导致过拟合，其中 3 层神经元的结果就显著弱于单层

因此根据奥卡姆剃刀原则，应选取准确率高的尽可能简单的模型，所以选取单层 256，词向量 256 的 LSTM 模型。在测试集上精确度可以达到 90.31%.

并且我还尝试了 Relu 激活函数，但是结果非常差如图所示。

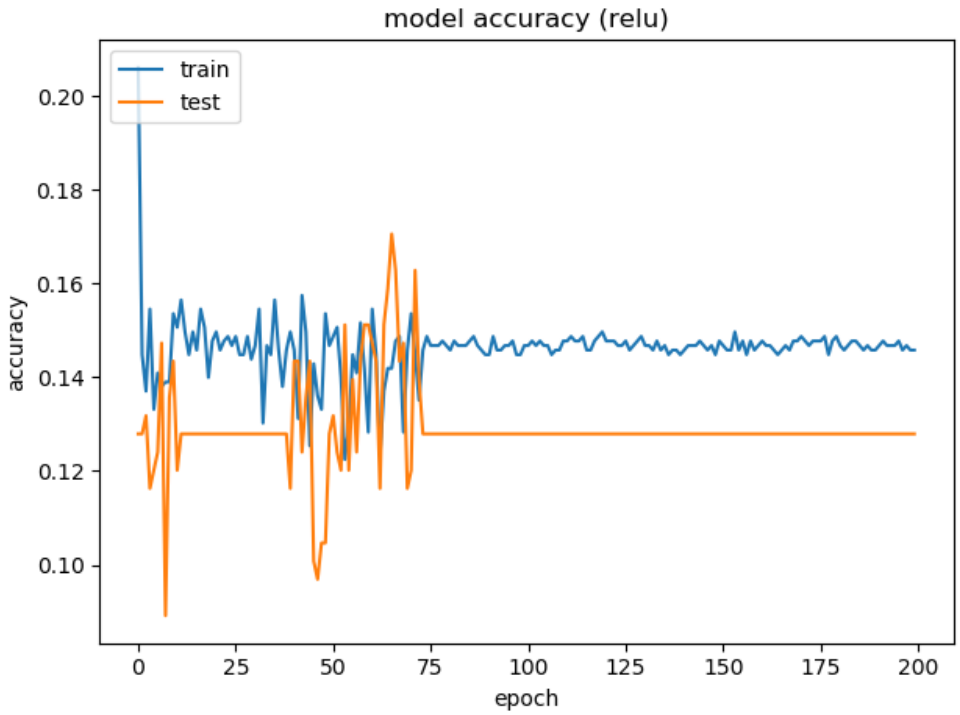


图 5: Re_lu 激活函数结果

经过查找资料，虽然 ReLu 激活函数有很多好处，比如 ReLUs 的好处之一是它们避免了梯度的消失。但是 LSTM 一开始就被设计避开了这一点。假设没有消失梯度的问题。所以这一点并不能对 lstm 神经网络有提升。而从另一个方面，relu 原则上是设计来训练深层的神经网络的，它的主要优势是更容易训练。而 LSTM 最初不是被设计成“深的”神经网络。所以反而可能会导致很多问题。

4 心得体会

通过完成这次 NLP 作业让我对神经网络以及 NLP 有了更深的了解，对于处理 NLP 问题的一般流程有了了解，以后有时间还想继续做一些可以进一步提高准确率的工作。并且在训练过程中发现 CPU 占用率是满的而 GPU 占用率较低，估计是 cpu 处理 batch 是分开处理分部交给 GPU 的，如果可以并行处理应该可以大幅提高训练的速度，并也找了相关的文章，但由于时间限制没有在作业中包含这一部分。最后感谢胡教授这学期的教学，让我收获很多，顺颂时祺。