东南大学自动化学院

实 验 报 告

课程名称：模式识别原理

实验名称：大作业：电影评论情感分析

姓　　名：陈垚鑫、曾庆翔、甘玉琪、辛均浩

学　　号：**171356、171355、171357、171348**

评定成绩：　　审阅教师：

目录

一、问题描述

情感分析也称观点挖掘(opinion mining)，是机器学习中自然语言处理(NLP)领域一个非常流行的，具有挑战性的分支，它主要是分析文档的情感倾向。人们用语言表达自己的情绪，而真实的情绪往往被讽刺，模棱两可，和一些有深层含义的词语所掩盖，这些对人类和计算机来说都是非常具有误导性的。

本次实验来自于 Kaggle 竞赛，主要任务是对电影评论文本进行情感分类。随着互联网的蓬勃发展，越来越多的人喜欢在网络上对电影进行评论，网络上积累了大量的影片评论数据，这些评论所表述的观影感受和电影情感特征对于建立个性化的电影推荐系统具有重要价值。本次实验数据集是来自 IMDB 中的电影评论，数据集一共包含了 50000 条评论以及其对应的情感分类（正面评论/负面评论），正面评论代表对于影片的评分超过了 6 颗星，而负面评论则表示评分低于 5 颗星。50000 条评论中一半用于训练集，一半用于测试集，分别整合成表格，表格分为三列，第一列表示评论的 id，第二列表示评论的文本内容，第三类表示文本的分类结果（1 表示正面，0 表示负面）。

二、实验目标

kaggle 网站上（https://www.kaggle.com/c/word2vec-nlp-tutorial#description）有一个 tutorial competition，内容是对 IMDB 的 50000 条电影评论进行情感分析，其中 25000 条作为训练集，剩下 25000 条作为测试集。在该 tutorial 上所提供的分类器 baseline 是 0.8421 的准确率。为此，我们着力于提升分类器的分类效果，能更好判别出电影评论对应的情感。分别用传统的词袋模型和 word2vec 模型初步提取文本特征，结合逻辑回归，多元贝叶斯等传统机器学习算法和 RNN，CNN 等深度学习算法进行分类，测试集准确率需要超过 Kaggle 上的 baseline，即 0.8421。在理解，实现各类算法的过程中体会其各自的优缺点与适用范围。掌握一些基础的调参方法与模型融合方法。

三、实验方法

3.1 数据预处理

原始的电影评论不能直接拿来当成分类器的输入，需要做一定的处理。

- 去掉所有的 stop word。Stop word 是指那些经常出现在句子中，但对句子的实际意义影响不大的词。例如 I、he、you、the、this、that 等。在 python 中有 Natural Language Toolkit (NLTK)的库，里面包含了英语的 stop word 库。通过 NLTK 的库将原始 raw data 中的 stop word 除去。
- 所有的词语进行词型还原。在英语中包含了很多时态的变换，例如单数变复数，动词变成动名词等。不同的时态对语意并没有多大的改变，因此我们需要将所有词还原成最原始的样子。
- 在所有否定词之后的动词、形容词前加上前缀 not_，成为一个新词。例如 I don't love this movie。Love 是一个很强烈的带有正面情感的词，显然在统计时不能直接使用，而是应用 not_love 替代，将 not_love 成为一个新的词，带有强烈的负面情感。改写规则如下：
  1. 遇到标点符号时就取消添加前缀。
  2. 在一个句子内第奇数个否定词后加前缀
  3. 在一个句子内第偶数个否定词后不添加前缀，为了解决双重否定词
- 最后去掉所有的标点、数字。我们认为数字、标点对电影评论的情感没有什么影响，因此除去。

通过上述的处理，raw data 已经转换为可以统计的数据了。下面以一个例子来说明处理结果。

```
原文：   this movie isn't good at all,and I don't love it,i hate it
处理后: movie notgood notlove hate
```

图 1 数据预处理示例

3.2 词袋模型

词袋模型就是基于统计词频的模型，分为以下 3 个方法。

1. 词频统计

简而言之，我们将所有评论中出现的词做成一个单词表，词频统计就是统计词表中所有单词在每条电影评论中出现的次数。显然这是一个非常稀疏的矩阵，大部分元素都是 0。

2. 词是否出现统计

原理和 1 相似。不同点在于统计单词是否出现。统计出的矩阵不仅稀疏，还是一个二值矩阵。

3. tf-idf 统计

在统计词频的基础上，还除去每个单词的逆文档频率。所谓逆文档频率指的是该单词在所有文档中出现的频率。这是文档处理中常用的技巧，是一个对词语权重赋值的方法，对于出现在很多文档的词语，其就不具有太大的代表性，因此不具备太多信息。赋予一个较小的权重。

可以看出，词袋模型仅考虑的词的出现频率，而不考虑词的空间位置。为了进一步考虑空间位置，我们统计词频不仅仅基于单个词语，而是基于句子中的连续 2/3 个单词构成的短语，简称 bigram/tirgram 短语统计。

另外，除了 tf-idf 使用逆文档频率对单词分配权重的方法，我们还研究使用了另外两种权重分配方法。

$$w1 = \left| \frac{cw^{+} - cw^{-}}{cw^{+} + cw^{-}} \right|$$

$$w2 = \left| \log\left(\frac{cw^{+}}{cw^{-}}\right) \right|$$

上式中 cw+表示在训练集正类所有电影评论中该单词出现次数,cw-表示在训练集负类所有电影评论中该单词出现的次数。可以直观理解为，如果某个词经常出现在正类中，而几乎不怎么出现在负类中，那么这个词具有很好的区分正负类效果，因此应该赋予更高的权重。W1 和 w2 为两种不同的计算方法，但他们思想都是一样的，其中 w2 权重为著名的 NBSVM 中所使用的方法。

最后，词频统计中可能出现一句话中有多个词重复，因此我们使用 log 函数对其进行了削减，使不同单词的词频不会相差太大。

基于上述的方法，我们使用了四个分类器，分别是 logistic regression（LR）、多元贝叶斯（MNB），基于 hurbe loss 的 SGD 方法，GBDT（使用 lightgbm）。在上述特征中分别使用四个分类器，并基于投票方法和概率方法进行融合，查看最后的结果。

3.3 word2vec 模型

3.3.1 word2vec

word2vec 也叫 word embeddings， 中文名词向量，是单词从稀疏 1-V 编码（这里 V 是词汇量大小）通过隐藏层投影到较低维矢量空间上，本质上是特征提取器，是初步对单词的语义特征进行编码。在这样的密集表示中，语义相近的词距离（比如余弦距离）也相近。

得到词向量的方式有两种：

- 随机初始化词嵌入矩阵，直接在模型中训练。
- 利用预训练好的词嵌入模型。对于预训练中有的单词直接取得其词向量，对于没有的单词，其词向量可以用 0 或者随机小的正数来填充。该词向量矩阵可以在之后的训练中固定或微调。

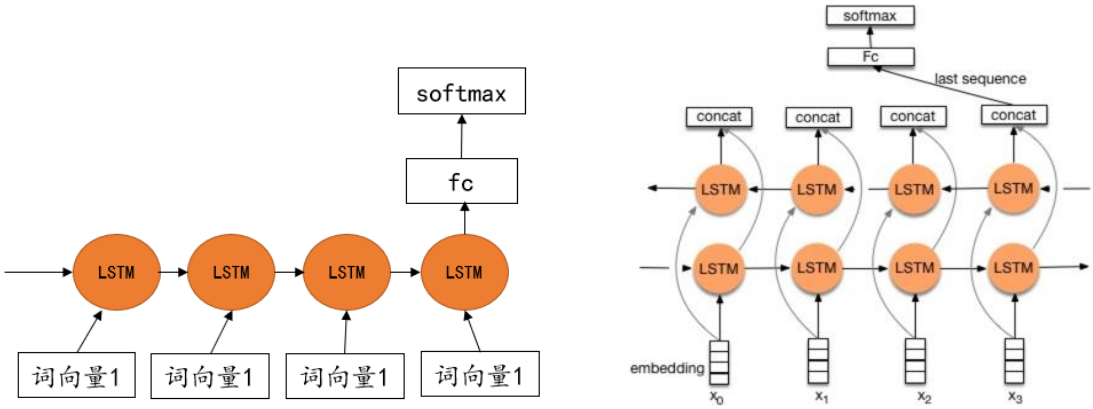
提取到词向量特征后，就可以用深度学习算法进行进一步地处理。本次实验主要实现了三种：RNN,

CNN 和 fastText。需要说明的是，这三种结构都可以处理不定长的句子，但是为了进行批训练，将句子进行填充或者截取到设定的最大长度，将不定长的序列变成定长的序列。具体来说，如果序列没有达到最大长度，则在前部补 0，如果超过最大长度，则从后面截取最大长度的序列。

3.3.2 RNN

RNN，即循环神经网络，其主要用途是处理和预测序列数据。不同于前馈神经网络的是，RNN 层内节点的输出还会重新输入本层，即 RNN 利用它内部的记忆来处理任意时序的输入序列，这让它可以更容易处理如不分段的手写识别、语音识别等。LSTM(长短期记忆网络)是 RNN 的一个改进，是为了解决 RNN 的长期依赖问题。LSTM 设计了门控机制，增加了 cell，适合于处理和预测时间序列中间隔和延迟相对较长的重要事件。在此基础上，通过两层或者多个 LSTM 层的堆叠，可以实现双向循环神经网络及深层循环神经网络。

本次实验采用了单向 LSTM 和双向 LSTM 两种结构，如图 2(a）(b)所示，单向 LSTM 是把最后一个词的隐藏层结果直接连接全连接层，双向 LSTM 则是把最后一个词的正反向隐藏层输出串联（concat）后再送入全连接层。



(a)单向 LSTM　　(b)双向 LSTM

图 2 RNN 文本分类结构图

RNN 的优点是顺序处理结构使得其能够捕捉到单词之间的位置关系，能够更好的表达上下文信息。缺点是下一个时刻的输出要依赖于上一个时刻的输出，无法在整个序列上进行并行处理，引起训练时间过长，训练速度非常慢。对于一个长度为 N 的序列，要建立长期依赖性，需要经过 O(n)次运算，运算复杂度高。此外，对于输入序列中的每一个元素，RNN（单向 RNN）会对序列中的第一个元素进行 n 次非线性运算，而对最后一个元素则仅进行 1 次非线性运算。

3.3.3 CNN

CNN,即卷积神经网络，可以捕捉局部相关性，擅长于图像处理。在处理文本数据时，将句子进行词嵌入后经过一层一维卷积与一层一维最大池化，最后送入全连接层，结构如图 3 所示。本次实验中 CNN 分别用了用预训练的嵌入矩阵和随机初始化的嵌入矩阵做了对比实验。对结构解释如下，其中超参数的设定均是随机初始化嵌入矩阵时设定的值：

输入层：句子中的词语对应的 word vector 依次（从上到下）排列的矩阵，假设句子有 n 个词，词嵌入的维数为 k，那么这个矩阵就是 n×k 的。本次实验中 n = 700，k = 400。

卷积层：输入层通过卷积操作得到若干个 Feature Map，卷积窗口的大小为 h×k，其中 h 表示纵向词语的个数，而 k 表示 word vector 的维数。使用多个 filter 来获取多个特征，得到的是若干个列数为 1 的 Feature Map。本次实验中 h = 5，filter 的数量为 800。

池化层：使用 Max-over-time Pooling 的方法。即简单地从之前一维的 Feature Map 中提出最大的值，最大值代表着最重要的特征。最终池化层的输出为各个 Feature Map 的最大值的集合。

全连接 + softmax 层：池化层的输出通过全连接的方式，连接一个 softmax 层，计算属于每个分类的概率。另外，由于本次实验是二分类任务，还尝试了 sigmoid 分类。

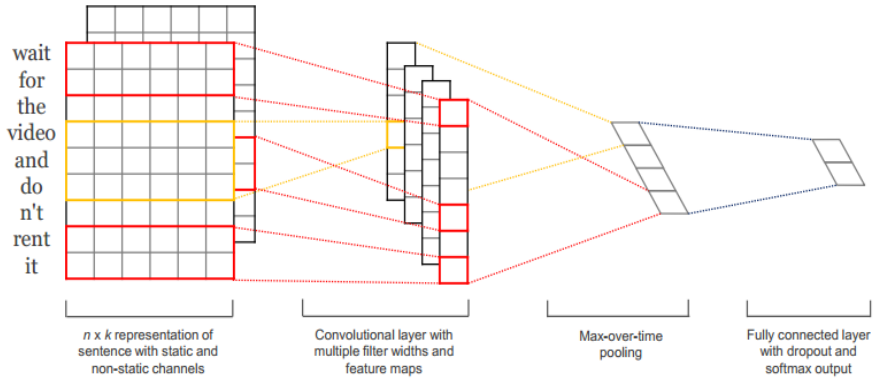


图 3 CNN 文本分类结构图

CNN 的优点是能够并行处理数据，计算更加高效，速度比 RNN 快很多。CNN 对每个单词的非线性操作是恒定的。缺点是对位置信息的捕捉不如 RNN，超参数调节繁琐。

3.3.4 fastText

fastText 的结构如图 5 所示，其基本思路是给定一个输入序列，首先提取 N-gram 特征得到 N-gram 特征序列，然后对每个特征做词嵌入操作，再把该序列的所有特征词向量相加做平均，作为模型的隐藏层，最后送入全连接层进行分类。这个思路类似于平均化的 Sentence Embedding。

N-gram 特征如图 4 所示，从上至下依次是 3gram，2gram 和 1gram。举个例子，对于句子“I like this movie very much”，它的 2gram 集合是{I-like, like-this, this-movie, movie-very, very-much}，也就是将连续的两个单词组合成一个整体，并为每个个体创建一个向量。

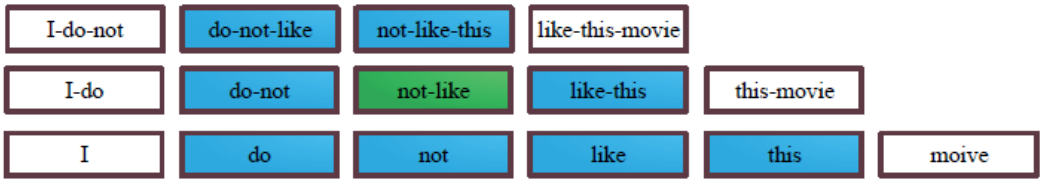


图 4 N-gram 特征

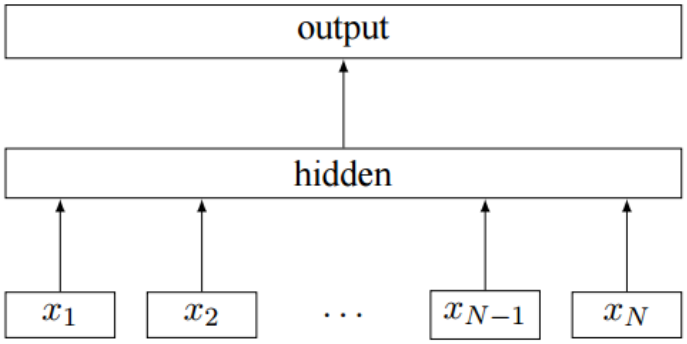


图 5 fastText 文本分类结构图

fastText 的优点是速度非常快，缺点是对于大的数据集复杂的分类任务准确性不如前两类。

3.4 模型融合

针对不同模型产生的结果，一般可以采用模型融合的方法使结果获得提升。模型融合就是综合考虑不同模型的情况，并将它们的结果融合到一起。最简单便捷的方式就是从竞赛的提交结果文件中进行融合，因为这样做并不需要重新训练模型，只需要把不同模型的测试结果获取后，通过采取某些措施得出最终结果，这种融合方法属于“晚融合”方法，即融合发生在各个模型预测结果之后，具有较好的效果。

3.4.1 改进投票法融合

根据不同模型输出的结果，最常见的融合方法是采用投票法，即根据所有模型对某一输入的预测结果进行投票，之后选择其票数最高的结果作为最终结果。本文中采用了改进的投票法进行模型融合，其流程如图 6 所示。

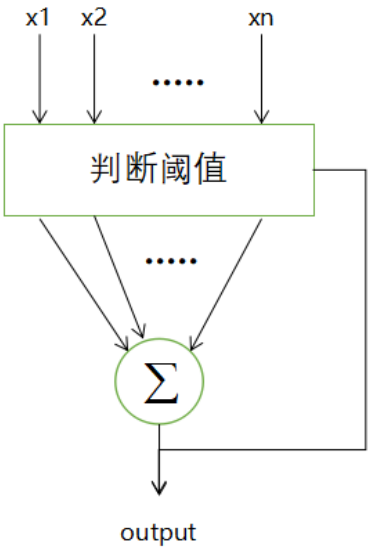


图 6 改进的投票法融合结构

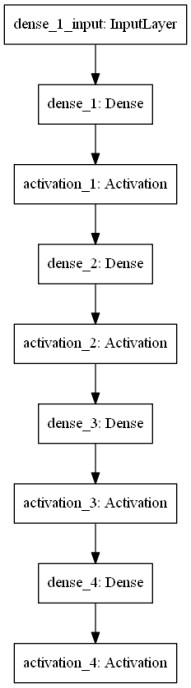


图 7 神经网络融合模型结构

图中 n 种模型的输入在进行传统投票前首先经过了一个阈值判断的结构。这是由于在模型进行预测时通过验证集上的预测结果我们发现在模型输出预测概率较高时，该预测结果也往往较为准确。当使用阈值过滤器将预测结果超过 0.99999 的结果筛选出来时，该部分结果数据准确率将有大幅的提升（对于 cnn 模型，这一结果可由 89%提升至 99%左右），但所选出的结果只占全部预测结果的一小部分。因此在进行模型融合时，可首先遍历所有模型的输出结果，通过阈值对结果进行过滤，挑选出置信度较高的模型预测结果，将其直接作为最终输出。

若所有模型均无法得到置信度较高的预测结果，则这部分预测结果再采用投票法进行加权求和，选出票值最高的结果作为最终输出。

经试验表明，在进行模型融合后，最终预测准确率针对单模型有较大提升，证明该融合方法确实较为有效。

3.4.2 神经网络模型融合

投票法是人为经验所设定的融合方法，并不一定适用于所有的问题。因此，本文采用了另一种方法对模型进行融合，即采用神经网络对多个模型的输出结果进行训练融合，进行融合的神经网络如图 7 所示。

进行神经网络融合时，将训练集划分为训练集和验证集两部分。首先在训练集中对各个单模型进行训练，之后利用 n 个单模型对验证集结果进行预测，之后将 n 组不同的预测结果输入图 7 所示神经网络结构

中，输出则与验证集真实输出进行损失求取，用以训练整个网络模型。与投票法相比，该方法不拘泥于模型融合时的加权平均方法，对于不同问题具有较好的普适性，同时也更容易找到更佳的融合方法。实验证明该方法能够获得精度更高的预测结果。

四、实验结果

4.1 词袋

词袋与 LR，MNB，SGD，GBDT 结合分类的效果如表 1 所示。

表 1 词袋模型与传统机器学习算法结合分类结果

特征	LR	MNB	SGD	GBDT	概率融合	投票融合
词频统计	0.8862	0.8697	0.8710	0.8830	0.8937	0.8913
词频统计+权重 w1	0.8963	0.8874	0.8713	0.8825	0.8965	0.8980
词频统计+权重 w2	0.8936	0.8742	0.8713	0.8825	0.8975	0.8954
词是否出现	0.8911	0.8760	0.8710	0.8830	0.8951	0.8932
词是否出现+权重 w1	0.9012	0.8950	0.8710	0.8835	0.8968	0.9002
词是否出现+权重 w2	0.9002	0.8925	0.8710	0.8835	0.8968	0.8998
词频 log 处理+权重 w1	0.9003	0.8929	0.8714	0.8825	0.8974	0.9004
TF-IDF 统计	0.8901	0.8969	0.8713	0.8782	0.8953	0.8954

基于上述表格可知以下几点结论：

- LR 在高维空间分类中效果非常好
- 添加权重比不添加权重效果更好，权重相当于一个特征选择的手段，因此会带来正面的效果。同时也可以看出，权重 w1 比权重 w2 要更好一点。
- 当单模型效果彼此差距不大时，融合方法在一定程度上能提升分类效果。然而当某个模型特别突出，而其他模型效果一般时，融合已经不能提升效果了，相反会有所下降。
- MNB 和 GBDT 对权重赋值不太敏感的原因是，这两类模型都不在乎特征数值的绝对大小，而只在乎该特征上样本的数值相对大小，因此同一个特征乘上同一个值，并不会引起相对大小的改变，因此没作用。而另外两类模型对数值的绝对大小有比较高的要求，因此权重赋值非常有用。
- tf-idf 在这里并不是很有用。因为逆文档频率可以反映词语对该文档的代表性程度，并不能反映对该种类的代表性程度，因此作用不大。
- 统计词是否出现比统计词的出现频率更有意义。

4.2 word2vec

word2vec 与 RNN，CNN，fastText 结合分类的结果如表 2 所示，其中 word2vec 分别采用了随机初始化的嵌入矩阵与预训练后微调的嵌入矩阵；CNN 最后的分类分别采用了 softmax 和 sigmoid；RNN 分别采用了单向 LSTM 和双向 LSTM；fastText 采用了 2-gram(1-gram 效果一般，没有做出展示，3-gram 由于机器内存有限，无法实验)。最后模型融合展示了投票法和神经网络融合两种方式的结果。

表 2 word2vec 模型与深度学习算法结合分类结果

特征嵌入矩阵	CNN（softmax）	CNN（sigmoid）	LSTM	Double-LSTM	fastText（2-gram）	投票融合	网络模型融合
随机初始化	0.89928	0.89944	0.83296	0.8296	0.90276	0.9178	0.92093
预训练+微调	0.88272	0.89784	-	-	-		

由表格可以看出，word2vec 模型在 fastText 上的效果最好，CNN 效果与 fastText 接近，RNN 最差。使用预训练的参数与直接随机初始化嵌入矩阵效果接近。使用神经网络做模型融合的效果略优于传统的投票法融合。

需要说明的是，这些结果的很大一部分原因是数据集小，分类任务简单。RNN 准确率差除了是因为数据集小没有发挥其优势外，还是因为训练时间过长，没有进行精细地调参。当面对大数据集以及复杂的任务时，从目前的多篇论文可知，预训练的嵌入矩阵与 CNN 的组合是综合考虑时间与准确率的优先选择。

五、实验总结与成员贡献

词袋模型在 LR 的效果最好，统计词是否出现比词出现的频率效果更好。然而词袋模型完全忽视位置信息。word2vec 模型在 fastText 上的效果最好，CNN 效果与 fastText 接近，RNN 最差。使用预训练的参数与直接随机初始化嵌入矩阵效果接近。word2vec 模型可以选择后续特征处理方式来决定怎样使用位置信息。

成员贡献见表 3。

表 3 成员贡献表

成员	完成任务	贡献率
陈垚鑫	词袋模型(LR 和 SGD)	25%
曾庆翔	word2vec 模型(CNN 和模型融合)	25%
甘玉琪	word2vec 模型(RNN 和 fastText)	25%
辛均浩	词袋模型(GBDT 和 MNB)	25%