

毕业设计说明书

|  |  |  |  |
| --- | --- | --- | --- |
| 作 者: | 赵斯蒙 | 学 号： | 913106840231 |
| 学 院: | 计算机科学与工程学院 | | |
| 专业(方向): | 计算机科学与技术 | | |
| 题 目: | 基于卷积神经网络的文本的情感分析 | | |
|  |  | | |

|  |  |
| --- | --- |
| 指导者： | 贾修一 副教授 |
|  |  |
| 评阅者： | \*\*\* 副教授 |

2017 年 5 月

声 明

我声明，本毕业设计说明书及其研究工作和所取得的成果是本人在导师的指导下独立完成的。研究过程中利用的所有资料均已在参考文献中列出，其他人员或机构对本毕业设计工作做出的贡献也已在致谢部分说明。

本毕业设计说明书不涉及任何秘密，南京理工大学有权保存其电子和纸质文档，可以借阅或网上公布其部分或全部内容，可以向有关部门或机构送交并授权保存、借阅或网上公布其部分或全部内容。

学生签名：

年 月 日

指导教师签名：

年 月 日

目录

[1 绪论 4](#_Toc481926518)

[1.1 工程背景及问题 4](#_Toc481926519)

[1.2 相关技术的现状 4](#_Toc481926520)

[1.3 需解决的工程问题 5](#_Toc481926521)

[1.4 论文章节安排 5](#_Toc481926522)

[2 卷积神经网络原理 6](#_Toc481926523)

[2.1 神经网络的基本原理 6](#_Toc481926524)

[2.2 训练算法 7](#_Toc481926525)

[2.3 卷积 9](#_Toc481926526)

[2.4 逻辑函数 10](#_Toc481926527)

[3 算法设计 10](#_Toc481926528)

[3.1 特征选择与预处理 11](#_Toc481926529)

[3.1.1 word to vector词向量模型 11](#_Toc481926530)

[3.2 模型搭建 13](#_Toc481926531)

[3.2.1 卷积神经网络 13](#_Toc481926532)

[3.2.2 Pooling汇合操作 14](#_Toc481926533)

[3.2.3 正则化 14](#_Toc481926534)

[3.2.4 归一化指数函数（softmax） 15](#_Toc481926535)

[3.2.5 交叉熵与权重衰减 15](#_Toc481926536)

[3.3 算法实现 17](#_Toc481926537)

[3.3.1 卷积操作 17](#_Toc481926538)

[3.3.2 神经网络 17](#_Toc481926539)

[3.3.3 梯度下降 17](#_Toc481926540)

[4 系统实现以及测试 17](#_Toc481926541)

[4.1 框架搭建 17](#_Toc481926542)

[4.2 数据集 18](#_Toc481926543)

[4.3 数据预处理 18](#_Toc481926544)

[4.4 模型搭建与训练 20](#_Toc481926545)

[4.5 实验结果分析 20](#_Toc481926546)

[致谢 21](#_Toc481926547)

# 绪论

卷积神经网络作为近几年再次兴起的机器学习算法，受到了学术界的广泛关注，随之而来的深度学习热潮以及其在各个领域上的应用，更是为人类掀起了了技术革命的新篇章

## 工程背景及问题

短文本的分类一直是机器学习领域一个比较困难的问题，由于短文本自身所具有的特点：上下文信息匮乏、字词的歧义性，致使短文本的分类本身就具有一定的不确定性，而微博文本，微博评论就是典型的短文本，对此类短文本进行情感分析以及更进一步的情感分类，对于舆情研究，满意度反馈，证券投资[[1]](#endnote-1)等领域，具有十分巨大的潜力。对于此类文本，传统文本分析的方法无法提供足够的精确度，面对微博文本这种时效性和口语性较强的文本类型时，更是无法给出让人足够满意的结果，所以，利用机器学习方法对此类短文本并进行分类就成为了一个潜力巨大的新方向。

## 相关技术的现状

目前，实现文本分类的机器学习技术主要有支持向量机(SVM)[[2]](#endnote-2)，朴素贝叶斯[[3]](#endnote-3)，以及本文中提到的神经网络等，在实际使用中，也有结合了传统的词典分类以及情感标签[[4]](#endnote-4)的分类方法，都取得了一定的成效，随着机器学习领域的不断进步以及近几年卷积神经网络和深度学习的发展，对于短文本分析的精度被不断刷新；除了分类器(classifier)的算法为文本分类提供了有力支撑，随着词向量[[5]](#endnote-5)的出现，文本数据有了更好的特征选择和特征表示方法；中文文本分词技术的提高则更好的利用了词向量所带来的语义上的优势；此外，计算能力的提升数据源的增加也为卷积神经网络在文本分类上的应用的成功做出的不可忽视的贡献。

卷积神经网络（CNN）是一种常见的深度学习架构，受生物自然视觉认知机制启发而来。1959年，Hubel & Wiesel [[6]](#endnote-6) 发现，动物视觉皮层细胞负责检测光学信号。受此启发，1980年 Kunihiko Fukushima 提出了CNN的前身——neocognitron [[7]](#endnote-7)。

20世纪 90 年代，LeCun[[8]](#endnote-8)等人发表论文，确立了CNN的现代结构，后来又对其进行完善。他们设计了一种多层的人工神经网络，取名叫做LeNet-5，可以对手写数字做分类。和其他神经网络一样， LeNet-5 也使用 backpropagation 算法训练。

CNN能够得出原始图像的有效表征，这使得CNN能够直接从原始像素中，经过极少的预处理，识别视觉上面的规律。然而，由于当时缺乏大规模训练数据，计算机的计算能力也跟不上，LeNet-5 对于复杂问题的处理结果并不理想。

2006年起，人们设计了很多方法，想要克服难以训练深度CNN的困难。其中，最著名的是 Krizhevsky 等人提出了一个经典的CNN 结构[[9]](#endnote-9)，并在图像识别任务上取得了重大突破。其方法的整体框架叫做 AlexNet，与 LeNet-5 类似，但要更加深一些。

AlexNet 取得成功后，研究人员又提出了其他的完善方法，其中最著名的要数 ZFNet[[10]](#endnote-10), VGGNet[[11]](#endnote-11), GoogleNet 和 ResNet[[12]](#endnote-12) 这四种。从结构看，CNN 发展的一个方向就是层数变得更多，ILSVRC 2015 冠军 ResNet 是 AlexNet 的20 多倍，是 VGGNet 的8 倍多。通过增加深度，网络便能够利用增加的非线性得出目标函数的近似结构，同时得出更好的特性表征。但是，这样做同时也增加了网络的整体复杂程度，使网络变得难以优化，很容易过拟合。

虽然在实验的测量中，CNN获得了巨大的成功，但是，仍然还有很多工作值得进一步研究。首先，鉴于最近的CNN变得越来越深，它们也需要大规模的数据库和巨大的计算能力，来展开训练。人为搜集标签数据库要求大量的人力劳动。所以，大家都渴望能开发出无监督式的CNN学习方式。

同时，为了加速训练进程，虽然已经有一些异步的SGD算法[[13]](#endnote-13)，证明了使用CPU和GPU集群可以在这方面获得成功，但是，开放高效可扩展的训练算法依然是有价值的。在训练的时间中，这些深度模型都是对内存有高的要求，并且消耗时间的，这使得它们无法在手机平台上部署。如何在不减少准确度的情况下，降低复杂性并获得快速执行的模型，这是重要的研究方向。

其次，超参数的选择对于CNN的性能有着较大的影响，比如学习率、卷积过滤的核大小、层数等等，这需要大量的技术和经验。这些超参数存在内部依赖，而且随着模型规模的增长，调整的代价汇编的很高。最近的研究显示，在学习式深度CNN架构的选择技巧上，存在巨大的提升空间。

## 需解决的工程问题

要实现文本分类，第一步的工作就是提取特征，使要输入的文本变成可以被模型所识别的数值形式，鉴于近年来词向量在文本分类领域的优异表现，本实验中决定采用的特征提取方式就是词向量，但是从源文本转化成词向量还需经历两个步骤，分别是“词”和“向量”，显然，中文文本和英文文本有着明显的区别英文以及其他诸多语言都有着天然的分词模式，而中文的分词则没有这种特点，有些时候甚至会出现一词多义、一句多分的情况，这就需要在特征选择阶段对文本进行分词，随着时代的变化，更多新名词的出现，分词的工具也需要不断更新迭代；在分词结束后，要给每一个分好的词映射对应的向量值，根据词向量5的解释，在获得实际向量值之前还需要另外训练一个词向量的模型，从而得到每一个词对应的向量值；在得到了一个句子的每一个词的向量值后，就可以近似的把这个句子作为一个图片，至此我们就得到了一个句子的数值矩阵形式，而且这种形式能很好的反映出这个句子所具有的语义特征；得到了可以输入的数据之后，接下来的工作就是把数据置入模型进行训练，但是要进行训练现在的数据维度还是太大，为了解决这个问题，我们引入了卷积操作，在合适的卷集合的大小之下，目标文本的序列特征将得到足够的体现。但是在进入神经网络模型训练前，还需要进行一个步骤，因为神经网络对于输入数据的大小有确定性要求，所以需要进行池化(pooling)操作，使每一个卷积操作的结果只有一个会被传输到输入端。在上述步骤完成后，到达神经网络的输入端的数据一定是确定数量的卷积核的输出值，这个确定的数量就是卷积核的数量；之后，要搭建适合的神经网络，在搭建神经网络的过程中，要选择适当的训练算法和学习率等超参数，以较好的适应学习的情景。在训练的过程结束之后，就需要进行验证的步骤，对模型的准确率进行评估，所以，在训练之前还要注意原始数据集的划分工作，划分成用于训练的训练集和用于测试的测试集。

最后，需要使用图形化的技术实现一个机器学习的简易系统，实现简单易用的调整各种学习过程中所需的参数，还要能自定义选择训练集，监视训练情况等功能。

## 论文章节安排

本篇论文主要介绍了借助卷积神经网络对微博文本进行情感分析的理论基础和系统流程，具体章节安排如下：

第二章主要介绍了卷积神经网络的基本原理以及理论基础，对必要的公式进行了演算和推导，主要包括神经网络的概念，卷积的概念以及训练算法的简单说明。第三章则详细的介绍了本次实验中所需要的具体的算法以及实现方式，对核心模块的实现过程进行了详细的说明和公式推导。第四章则介绍了构建系统的详细流程，关于数据预处理的方法，模型搭建的结构，训练与验证的方式等等，最后对实验的结果进行分析，并对本次实验使用的机器学习系统进行简单的介绍。

# 卷积神经网络原理

神经网络算法最早源于上世纪40年代，其主要思想是模仿人类神经元的工作特性，每一个输出值都与每一个输入值呈线性关系，通过增加模型的复杂度和参数的数量，增加模型的精确性和可靠性，随着计算能力的增加，可以训练的神经网络的规模也越来越大，在各个机器学习的应用领域取得了越来越好的结果。

## 神经网络的基本原理

人工神经网络(artificial neural network)ANN，简称神经网络，最基本的单位是神经元：[[14]](#endnote-14)如图2-1所示

其中，神经元的每一个输入都乘以相应的权重并相加，与神经元自身的阈值进行对比，这个过程需要一个激活函数来进行，决定是否向下一层传播，从而得到用于输出的，成为模型的输出或者是进入下一层继续训练，用数学公式表达起来就是如下形式：

……

Sigmoid

图 ‑1

理想中的激活函数应当是如下图2-2的阶跃函数，但是由于在进行后续操作的过程中，会出现求导，求积分等运算，使用阶跃函数就无法很好的适应这样的计算，因为显然，该函数在0处不可导，而且在0处不光滑，这就导致在涉及到求导以及积分的时候必须要进行分类，增加了模型的复杂度。所以，当前主要采用的激活函数石是另一种更加平滑，连续的函数，这就是被称为sigmoid函数的激活函数，典型的sigmoid函数如图2-3所示，这种函数也被称为挤压函数，因为其目的就是尽可能的模拟阶跃函数的性质，尽可能的减少在函数值中间区域（0-1）的输入的范围，从而实现激活的效果。

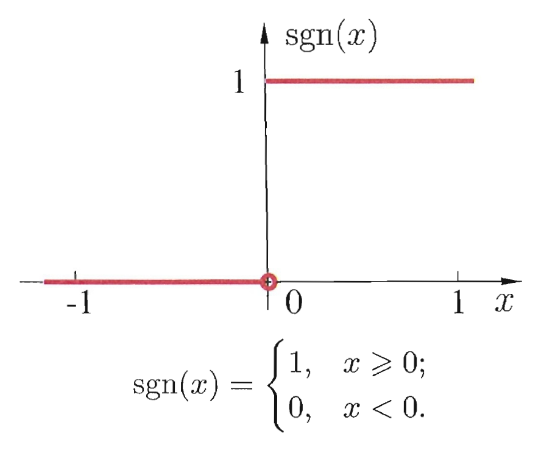
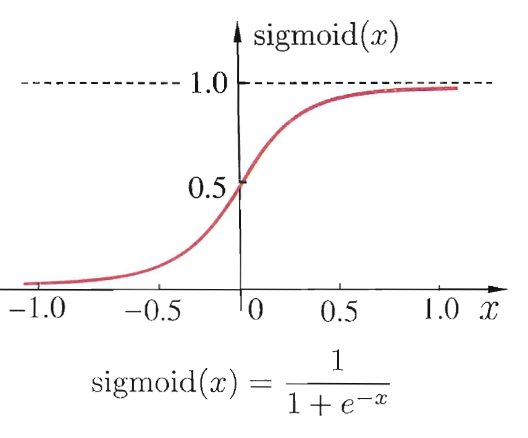
由多个神经元按一定的层次结构组合起来的模型就成为神经网络，简单来说，神经网络可以看做由若干线性模型嵌套并列组成的数学模型。

图 ‑2

图 ‑3

## 训练算法

假设现有一层神经网络模型，输入为x，正确的输出结果应当是y，而我们的模型输出的结果是，结果不正确，需要对模型的参数进行调整，使：

其中η表示学习率，在学习率合适的时候，模型会向着给定输入输出的模式拟合。

但是，这只是在一层网络的情况下，在多层网络的情况之中，这样的训练方式就无法使用了，需要引入梯度下降算法。

误差逆传播算法

给定训练集即输入端含有*d*个属性，输出端由*l*个属性，而且这个神经网络由输入层，隐层和输出层三个层组成，便可以得到下图的神经网络图，*d*个输入层，*q*个隐层，*l*个输出层，其中输出岑第*i*个神经元的阈值由表示，隐层第*h*个神经元的阈值由表示，输入层第*j*个神经元和隐层第*h*个神经元之间的权值用表示，隐层第*h*个神经元和输出层第*i*个神经元之间的连接权值用表示，可以得到隐层第*h*个神经元的输出，输出层第*i*个神经元的输出,其中为隐层的第*h*个神经元的输出，假设隐层和输出层的神经元都使用了sigmoid函数,假定神经网络在上述训练集中的输出为即：

由*y*与所产生的均方误差，就称为神经网络之中的代价函数，其值为：

与一层的模型类似，每一层的权重值都需要进行更新操作，更新的估计式为：

接下来就是要计算的值了，这就要使用到梯度下降算法，可以看做由模型中所有待训练参数作为自变量的未知函数的值，所以，由梯度的概念可知，在当前位置求出所有权重参数关于代价函数的梯度值表示的是代价函数在这一个自变量上的偏导数：

由于代价函数表示训练模型和实际值之间的差别，所以训练的目的也就是让的值越来越小，所以，当目前的梯度值是正的，就表示随着的增加，也会不断增加，而反之，如果梯度值是负的，那么，随着的增加，则会不断变小，所以为了让不断变小，就有：

由于首先影响的是第*i*个输出层的神经元的输入值再影响到的，所以有：

显然

另外，sigmoid函数具有一个很好的性质：

于是，可以推出：

经过推倒，可得到

类似的，可以得到隐层和输入层权值的变化量，隐层阈值，输出层阈值的表达式，分别为：

## 卷积

卷积是利用两种函数生成第三个函数的一种数学算子，其主要分为两种，分别是对于连续函数和离散函数进行运算的形式，由于神经网络之中所有的输入源都是离散的，所以下面只对卷积操作的离散形式进行介绍。

对于定义在整数上的函数*，*我们定义卷积函数为：

其中，*m*表示卷积的窗口大小，默认为，而*n*表示进行卷积操作的基本点对于所有域外的取值，都设成零，当的支撑集为有限长度*M*时，上式就变成了对有限数量的函数进行求和：

在具体情境中，会视情况改变卷积核的大小，而且在二维形式下，公式会有变化一般的，有：

其中，*k*表示卷积核（kernel），*m*表示原始矩阵，而卷积核的大小则是该式表示的值是在处的卷积操作值，对所有的原始矩阵元素进行操作得到的新的矩阵，就是卷积操作所期望得到的结果。

## 逻辑函数

\begin{align}
h_\theta(x) = \frac{1}{1+\exp(-\theta^Tx)},
\end{align}在进行分类的过程中，需要将模型的输出属性确定为若干个输出，其中，最简单的分类应当是二分类问题，在进行二分类问题之前，首先假设，在输出层的假设函数（激活函数）为：


\begin{align}
J(\theta) = -\frac{1}{m} \left[ \sum_{i=1}^m y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)}) \log (1-h_\theta(x^{(i)})) \right]
\end{align}
而代价函数为：

根据梯度下降算法可以计算出每一个参数的调整值，逻辑回归的本质上是线性回归，只不过根据二分类的问题调整了代价函数以及激活函数的形式。

# 算法设计

卷积神经网络由两部分组成，分别是卷积层和神经网络层，其中，神经网络的设计源于人类神经元的突触，激活的方式，让各个神经元之间彼此相连，通过调整参数来实现模型的拟合，而卷积层的操作可近似作为一个参数稀疏的神经网络层，其主要功能是降维和挖掘相邻像素(在本例中是词汇)之间的关系。

## 特征选择与预处理

在进行神经网络训练之前，首先要做的是将文本信息转化成有意义的数字信息，目前常见的编码方式有词袋模型（BOVW），独热码（one hot encoding）以及词向量（word to vector），也有使用随机分配的方式进行赋值的方法，经过比对，还是包含了语音信息的词向量的效果最好。

### word to vector词向量模型

文本处理不同于图像处理以及其他直接得到的数值处理任务，文本本身的计算机编码是不具有任何意义的，相邻的数据所具有的含义可能较大，从模型的层面上，相近的输入得到的输出结果应该是相似的，但是原文本所包含的含义却截然不同，对预测结果势必有一定的影响，对于这一点，英文的影响要明显大于汉语的影响，因为无论是utf-8还是各种传统的汉字编码，基本上都是按照文字的偏旁部首排序的，由于汉字本身所具备的特点，偏旁部首都是有一定的意义的，所以相近的字在语义上也存在着一定的相似性，而英文的情况就有所不同了，因为英文本身的字母不具有任何含义，拼成的词汇也很难发现其内在逻辑，所以，找到一种合适的表示文本特征的方法就是提高神经网络模型计算准确度的思路之一。

word embedding是在2003年由Bengio等人提出5，旨在从大量的数据源样本之中找到词元的分布特性，进而量化其语义特性，在一定条件下还可以进行分类，在本实验中采用的方法是由Tomas Mikolov团队所提出的Word to vector方法 ，下面将对其进行适当介绍。

* Skip-gram 模型

Skip-gram模型是基于这样一个假设：在有足够大数量的训练文本的情况下，根据特定的上下文语境，在任意空缺的地方填入一个词，所有的单词中，可能性最高的几个单词之间一定具有相近的含义；这样的假设有一定的实际道理，比如，在上下文文本“一只白色的\_\_\_\_穿过了马路”，我们一般都会联想到应该填入“狗”，“猫”之类的词，而如果在训练集之中存在这样类似的文本，那么，由上下文所影响的关于单词的属性值就能反映出这个单词所具有的含义。所以，为了得到这种映射，首先要对这个映射进行初始化，给每一个单词对应的向量进行随机赋值，接下来，就可以看成一个神经网络的训练问题，每一个词的向量值可以看作是神经网络之中的权值信息，而代价函数则是由下式表示：

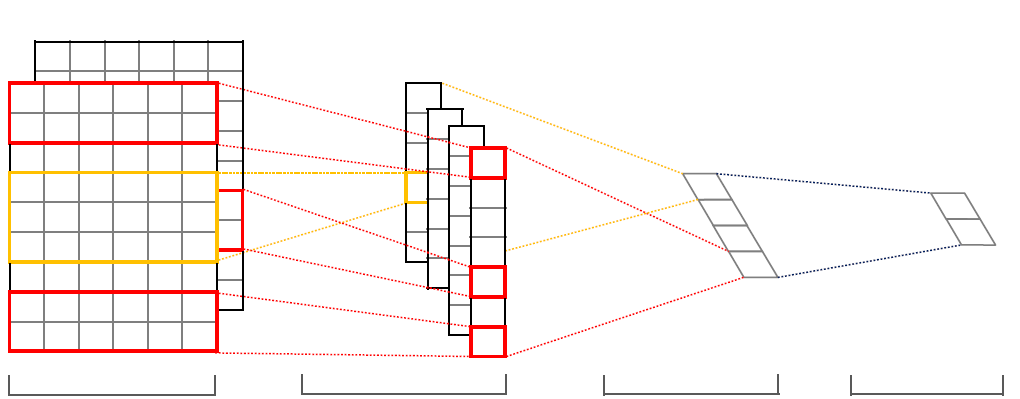
其中表示在出现的情况下之后第*j*个单词出现的概率，而该模型训练的目的就是让这个代价的值尽可能的小，接下来就要计算*。*

其中，代表对应*w*的向量值，而分别表示的输入单词和输出单词，对于word embedding 中所训练的神经网络的具体细节，在此就不做过多的展开，只需知道由这个操作得到的词向量映射包含一定的语音信息，而且具有确定的长度，这对于卷积神经网络的输入是非常重要的。

## 模型搭建

卷积神经网络的模型如下图所示，首先是卷积核进行的卷积，然后是pooling的汇合操作，接着就是数据进入隐含层，最终到输出层要决定是采用多个二分类分类器还是softmax分类器，根据学习问题的情景不同，做出的选择也有所不同，计算出衰减函数后就可以应用梯度下降进行训练，但是利用传统的均方误差学习方法又存在一些问题，因此引入交叉熵和权重衰减操作，更好的规范模型。

图 3‑1



文本的词向量序列，其中每个单词映射出一行，一个句子由一个矩阵表示

每一个卷积核都对全部的词矩阵进行操作，得到对应的一维值序列

Pooling操作，每一个卷积核得到的值序列，都只取其中的一个值，以保证输入神经网络的数据大小是确定的

全连接层，在神经网络中计算之后得到用于分类的值数据

### 卷积神经网络

经过上一部分的工作，假设我们已经得到了所有需要的单词的词向量，而且每个单词的词向量长度为*k*，现在需要输入一个句子，令：作为与句子中第*i*个单词有关的单词向量，一个长度为*n*的句子（必要时需要进行填补）就可以表示为：

其中是并置算符，就是让作为单词的并置。同时一个卷积操作还应当包括一个卷积核，这个卷积核的作用在上面已经描述过，就是在卷积操作中的第一个函数，注意，这个卷积核的大小一般是，其中，*k*是向量的长度，而*h*则是小于*n*的任意正整数，这就表明，每一次卷积操作都会包含若干个单词的所有向量信息，其含义就是应用于一个规模为h的局部语块，从而产生一个新的特征。举个例子是一个从单词中生成的新特征，那么的生成公式为：

其中：是运算偏项，*f*是一个非线性函数（比如说双曲正切函数），这个滤波器应用于句子中的每一个局部语块，从而产生一个特征映射

### Pooling汇合操作

可以看出上式得到的结果*c*的大小与源向量长度*n*有关，这对于神经网络的输入来说是难以接受的，一般的，我们可以采用扩展或者是压缩的办法来统一输入的大小，但一般都是应用于图像处理的课题中，使用了拉伸和压缩，是否对语义没有影响，还难以有确切的定论，而且本文主要讨论的是卷积神经网络的应用价值，所以在汇合的过程之中就采用了最简单也是目前比较主流的max over time池化操作，对每一个卷积核生成的向量

将其中的最大值作为与特定滤波器绑定的特征值。仅仅保留每一个映射的最重要的特征——具有最高值。这种池化策略自然地可以处理可变长度的句子。

在一定程度上也减少了参数数量和计算的复杂度

### 正则化

为了防止函数的过拟合，需要在进行pooling操作之前进行正则化操作，正则化操作的方法为：

### 归一化指数函数

由于之前对神经网络的基本原理进行了详细的解释，所以隐藏层的结构就不做详细介绍了，但是对于输出函数的选择还是需要解释一下。

在监督学习中，常见的分类模式有二分类，n-分类，二分类可以看作n-分类的特殊形式，任何n-分类问题也都可以由最多二分类问题解决，如使用卷积神经网络求解的有名的手写数字识别问题，其中分类器所分出的可能的类别有10种，且这十个类别之间互斥，那么对于特定输入，就需要得到划分在每一个类别上的概率，概率最大的结果将是最有可能的结果，与之对应的，若果每一个类别之间不是互斥的，那么使用n个二分类器就可以的到关于每一个类别的分类结果。所以，在前文中，我们提到了二分类问题采用的逻辑函数回归，而softmax回归就是这种回归在多分类问题上的推广，由于本文中讨论的情感分析系统仅仅进行的是二分类问题，所以对于softmax分类不做详细介绍。

### 交叉熵与权重衰减

在介绍神经网络的时候，我们采用的是来L2范式作为代价函数：

但是许多实验证明，用L2范式作为代价函数会出现一个问题；首先，对该函数求导，可以得到：

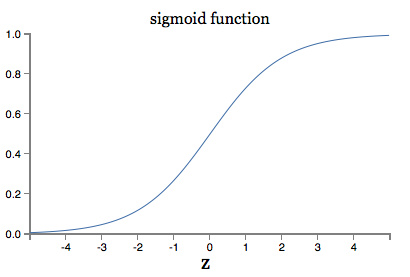
由sigmoid函数的图像（见图3-2），可以看出该函数所具有的性质，在*z*的值较大时，趋于平缓，而对应的的值也就趋于0，上式所表示的梯度值也就非常小。这就是为什么有的时候尽管模型的代价函数值很大，但学习速率还是很慢的原因。

图 3‑2

在这样的情境下，就需要引入新的代价函数，以谋求改善梯度值收敛缓慢的问题，在这样的背景下，用交叉熵表示的代价函数出现了。

其中，*a*表示训练数据的结果，而*y*表示实际的标签，这个加和式覆盖了所有的训练输入*x*，根据上式的处理方法，对此式进行处理，用带入上式，得到：

化简之后得到：

利用sigmiod函数的定义：，进行若干步推倒可以将上式化简为：

由此可以发现，梯度值与成线性关系，也就是目标值与实际值的差距越大，进行参数学习时的步长也就越大，也就从根本上避免了收敛缓慢的情况。

另外，权重衰减也是代价函数中的一个重要的组成部分，在加入了权重衰减的损失函数中，损失值表示为：

其中，表示要进行训练的参数，表示该参数所对应的权重衰减的值。该式的目的在于增加大参数的惩罚，从而促进模型训练更小的参数，因为一般来说，较小的参数值具有较少的过拟合性，而且如果不加以限制的话，权值将有可能向着越来越大的方向发展，事实证明，权重衰减对于模型的提升还是值得认可的。

## 算法实现

### 卷积操作

卷积操作的边缘填充策略有两种，一种是全部补零，而另一种则是不对超出变缘的像素进行卷积计算，在第二种情况下，卷积所得到的新矩阵大小是一定小于原始大小的，另外一个影响生成向量大小的参数就是卷积操作进行的步长，也就是每隔多少个元素进行一次卷积操作。在本例中，卷积的边缘策略为第二种，算法如下。

1. w,h = 卷积核的宽度和高度
2. x,y = 文本向量矩阵的长和宽
3. sx,sy = 卷积操作的步长
4. m[x,y] = input(text)//将文本转化成词向量
5. k[w,h] = rand()//随机初始化卷积核矩阵
6. out[ceiling((x-w)/sx),ceiling((y-h)/sy)]//输出矩阵
7. for i from 0 upto x step sx:
8. for j from 0 upto y step sy:
9. if i+w≤x and j+h≤y then
10. sum = 0//求和
11. for q from 0 upto w
12. for p from 0 upto h
13. sum += m[i+q][j+p]\*k[q][p]
14. end for//第12步的循环结束
15. end for//第11步的循环结束
16. out[q][p] = sum
17. end if//第9步的判断结束
18. end for//第8步的循环结束
19. end for//第7步的循环结束
20. 输出：out[]

可以看出卷积操作中共有两次两重循环的嵌套，时间复杂度为O(x\*y\*h\*w)，但是随着硬件计算的发展，卷积操作的计算速度远比一般的神经网络要快。

### 神经网络

神经网络的基本单位是神经元，每一层神经网络的计算就是将每一个神经元的输出值相组合，计算方法如下：

1. 输入：I[len\_in]//其中len\_in为输入向量的长度，I数组表示来自上一层或者输入层的数据
2. w[len\_out][len\_in] = 该层神经网络的所有参数权重值组成的矩阵
3. b[len\_out] = 参数阈值矩阵
4. out[len\_out] = 输出向量
5. for i from 0 upto len\_out
6. sum = 0//用于计算参数的累加和
7. for j from 0 upto len\_in
8. sum += w[i][j]\*I[j]
9. end for//第7步的循环结束
10. sum+=b[i]
11. out[i] = sigmoid(x)//激活函数
12. end for//第5步的循环结束
13. 输出：out[]

简单来说每一层的神经网络就是进行一次矩阵的乘法，偏移层也可以合并入权重向量中，只不过对应的输入向量要加入一个恒为1的元素。

### 梯度下降

从2-2章中推出的公式可以得到：

这是在损失函数为L2范式的情况下的梯度下降公式，在使用交叉熵作为损失函数的时候，其形式为

所以，可以得出其计算的算法为：

1. 输入：logits[len\_out],label[len\_out]//
2. cost = 0
3. learning\_rate = 学习率
4. for i from 0 upto len\_out
5. cost += logit[i]-label[i]
6. end for
7. cost = cost\*learning\_rate/len\_out
8. weight[wn]//表示需要学习的权重参数全集
9. delta [wn] //wn表示权重参数的个数，delta表示权重的变化量，用于输出
10. for i from 0 to wn
11. delta[i] = weight\*cost
12. end for
13. 输出：delta

以上为交叉熵代价函数下权值的梯度计算方式，输出的是对每一个权值修改量的数组，要实现学习与进化，还需要对权值应用这些修改，修改参数的方式也分为多种，一般来说，权值的修改会添加移动平均的算法以减少抖动，在本次实验中我们采用的方法是指数平均法：

每一次得到的新变量都要与存储的旧变量进行加权平均，这样能保证尽可能的减少参数学习中的抖动，却又能给新的学习变量最高的权重。

# 系统实现以及测试

## 框架搭建

对于机器学习，目前有很多成熟的机器学习框架，诸如caffe，scikit-learn，tensorflow等等，而且大部分都是开源的，在编程语言上，python因其简单，可靠，功能强大的特点成为目前主流的机器学习语言，虽然python也存在着性能较低的问题，但是与c语言的良好兼容性一定程度上弥补了这种缺陷，而且，很多进行复杂计算的机器学习库都采用c语言编写核心部分，从而获得了极高的性能，也使得更复杂的机器学习模型得以运行。

* Tensorflow

TensorFlow是谷歌基于DistBelief进行研发的第二代人工智能学习系统，其命名来源于本身的运行原理。Tensor（张量）意味着N维数组，Flow（流）意味着基于数据流图的计算，TensorFlow为张量从流图的一端流动到另一端计算过程。TensorFlow是将复杂的数据结构传输至人工智能神经网中进行分析和处理过程的系统。

TensorFlow可被用于语音识别或图像识别等多项机器深度学习领域，对2011年开发的深度学习基础架构DistBelief进行了各方面的改进，它可在小到一部智能手机、大到数千台数据中心服务器的各种设备上运行。TensorFlow的使用完全开源，任何人都可以用。

在本次实验中采用的机器学习框架便是tensorflow，因为其结构清晰，效率高，控件丰富，能满足本次实验的所有需要。

## 数据集

本次试验所使用的短文本数据来源是微博，但是由于目前大部分微博的内容都包含图片，纯文字的博文很少，而相对的，对微博的评论却大部分都是纯文本，且具有丰富的情感倾向，所以，本次试验的数据集就采用的以微博评论为主的数据集，其中，训练集大小为6000，验证集大小为1000，其中正负情感各含1/2

## 数据预处理

对文本形式的输入，需要进行分词和建立词向量模型，在本次实验中采用的词向量模型是从互联网上下载的经过训练的模型，其语料库是来自维基百科的100万个词条，而之所以要使用来自维基百科的语料库而不是大量爬取的微博文本，是因为微博的文本时代性太强，词汇含义变化的很快，在语义上存在不稳定性，利用某一个时间段内的语义解释训练出来的词向量显然是不恰当的，随着时间变化准确度也会下降。而使用百科中的文本就可以基本语义的稳定性，尽管缺少了一些“新词”、“潮词”，但是由于其语料库的大部分文本在语义表示上都是正确的，所以能够处理好，三者之间的关系如图4-1所示。

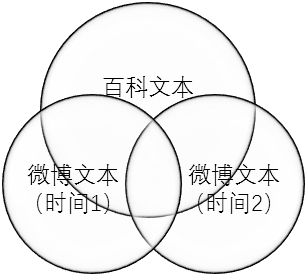


图 4‑1

在选择了向量集和训练文本之后，就需要将训练的数据集转变成输入模型的矩阵，这个过程由三个过程组成（见图4-2）：1.首先对原始文本进行分词，这个步骤由python自带的第三方库完成；2.把分好词的文本中的词一一映射成词向量模型中的向量；3.将生成的向量按照词序排好，组合成矩阵。在其中涉及到了三个问题，首先是第一步中的分词，对于一些新词和网络语，分词系统无法给出正确的分词方式，在这个问题上我的解决方式是跳过，有的分词系统确实是可以添加自定义单词进入，实现尽量正确的分词，但如若添加的单词不在向量模型中，这就引发了第二个问题，假如分出的词在向量模型中无法找到准确的映射匹配，也就是所谓的“未初始化的向量问题，一般来说，初始化的方法有两种，全部置零和随机初始化，我选择的方式是全部置零，因为从人脑思考问题的方式来看，在一个句子中出现的不理解的单词，一般会有三种做法，一是根据字形来推测，二是根据上下文来猜测含义，都没有结果就直接跳过，接着读下面的句子，前两种方法都需要进行额外的学习，而第三种方法对应的操作正是置零；最后一个问题就是向量的大小的问题，由于tensorflow中卷积神经网络要求输入向量必须具有确定的长度，所以必须要把文本向量全部转化成统一的大小，在上文有讨论过将变长向量归一化的方法，所以这里我采用补齐的方法，将向量长度补成微博文本最长限制的140，填充的值为全零。至此我们得到了输入所需的向量。

图 4‑2

## 模型搭建与训练

本机器学习模型主要分为四个模块，预处理模块，模型构建模块，训练和监督模块，web交互模块。

预处理模块的作用就是将文本转化成需要的向量，具体方式在4.3中已经详细介绍过，考虑到训练的时候的数据集具有整体性，而且调试的时候需要多次使用同一个数据集，如果每次使用都需要将文本数据转化成向量，将会带来很多计算量的损耗，所以在预处理模块中，文本文件被指间转化成tfrecords形式，这是tensorflow特有的数据保存的二进制形式，可以快速高效的传输给训练模型。

模型构建模块搭建了数据训练的流程图如图4-3所示，首先是卷积层，卷积层包含若干卷积核，每一个卷积核都由随机初始化的参数构成，卷积核的大小和数量，但是可变的只是在词维度，因为卷积核的宽度固定为词向量的长度，这样，经过卷积操作后得到的就是一个一维向量，在进入第二层pooling层之前，要对向量进行*l2*正则化，以减少过拟合的情况；在pooling层中，每一个卷积核产生的一维向量只产生一个最大值进入到下一层的比较；之后的操作是dropout，从上一层的神经元传递过来的值都有一定几率被舍弃，这个比率值设为*p*，每一个神经元的值可以这样表示：

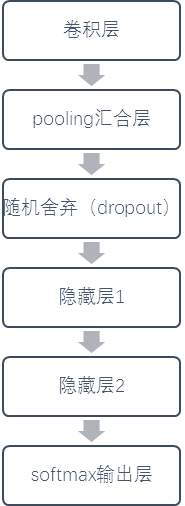


图 4‑3

其中rand是随机函数，？表示对前面的语句进行判断，若是真值则取第一个值，假则取第二个值。

进行这个操作的目的是增加不确定性，减少过拟合，一般来说在训练的时候，*p* 的值设为0.5，即一半的神经元不被激活，而在进行验证的时候*p*值调整为1，激活全部的神经元。

由上一部生成的值作为输入进入隐藏层，这里的两层隐藏层全都是传统意义上的神经网络层，每一个神经元都由一组权值和一个阈值*b*组成，并且在计算过输出值后，要使用sigmoid函数进行激活，然后传递到下一个隐藏层或者输出层，最终的输出层其实形式上就是一个普通的神经网络层，为了节省计算资源，tensorflow一般将激活函数合并在计算代价函数的操作之中。在完成上述操作之后，输出数据将作为输入计算交叉熵，同时累加所有变量的权重衰减，得到函数最终的代价值。

在得到代价值之后，模型进入到训练操作，tensorflow提供了一个梯度下降计算器，去计算所有变量的梯度值，随后要进行指数平均操作(exponential moving average)，因为在进行变量修改的时候要考虑到异常数据导致计算梯度值不正常的情况，减少过拟合的同时对新输入的值给予足够高的权重。

至此，一轮神经网络的学习全部结束了。在进行多代的迭代时，学习率的选择对于学习的结果是极其重要的，因为学习率太高，函数收敛的速度快，但是到临界点的时候会发生抖动，而学习率太低，收敛的速度过慢，效率低，所以找到一个合适的学习率也是需要经过多次试验的摸索和经验。

下面，对我们的训练系统进行简要的介绍。

本实验系统主要由两个部分组成，分别是实验设定部分和实验监控部分，其中，实验设定部分的作用就是提供上传数据集和设定实验参数的功能，界面如图，每一次训练都可以为其提供如下参数，卷积核的数量，卷积核的大小，隐藏层1,2的大小，机器学习的训练学习率，训练的最大次数，每批样本的数量以及指数移动平均的衰减参数值。并且可以指定已经上传的数据集。

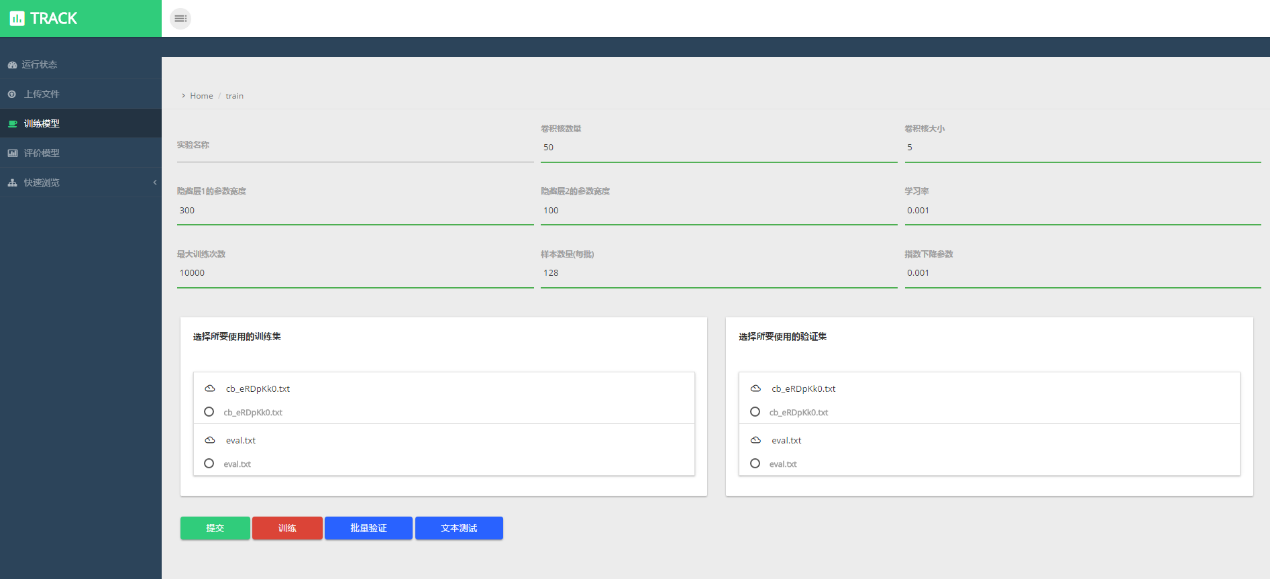
图4-4所示的是系统中的建立训练界面，通过输入参数和指定数据集，让训练进入到准备状态，保存后可以再主页找到训练的列表。（图4-5）

图 4‑4

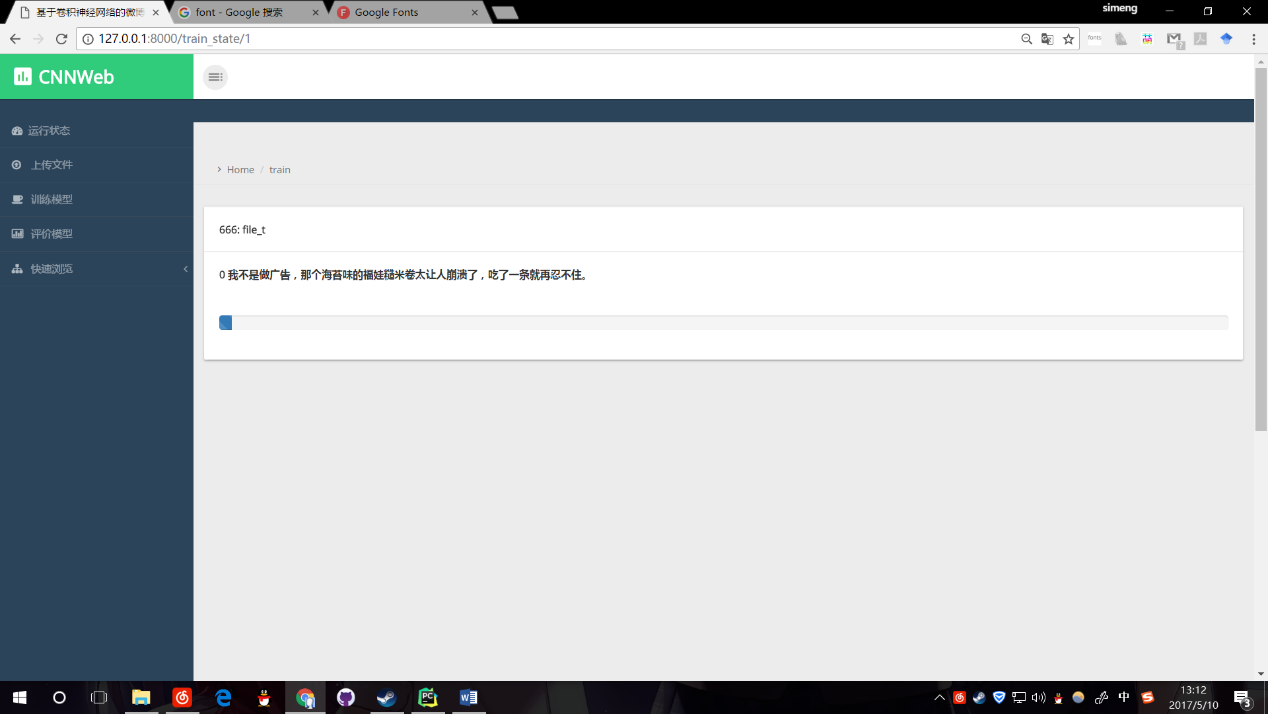
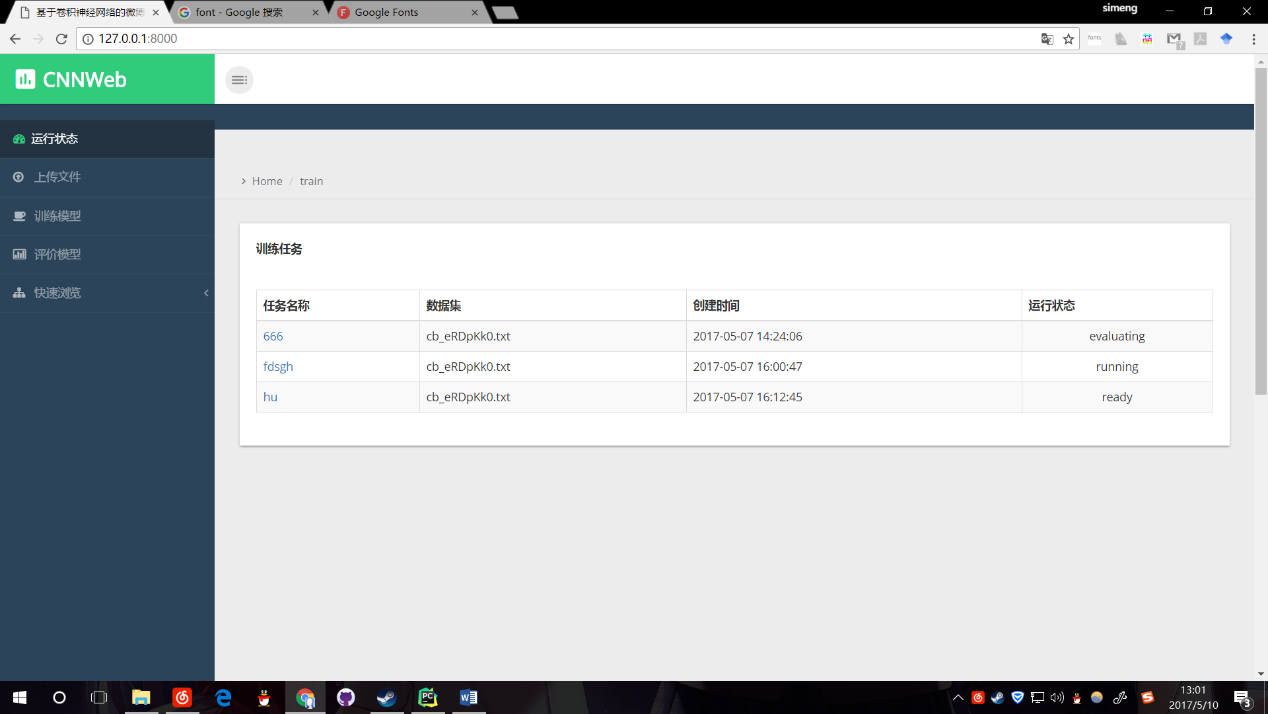
在点击了每一个训练之后，可以进入到训练额修改界面，进行修改或者直接开始进行训练，修改界面总共有四个选项，分别是修改的提交，训练，批量验证和单独测试。点击训练之后，就可以进入到训练的界面了，在后台的服务器端已经开始运行训练程序，从前端看则是一个显示进度的进度条。

图 4‑5

图 4‑6

训练的过程分为两步，一是生成原始数据的tfrecord格式的数据集，二是进行实际的训练过程。



图 4‑7

验证的部分也是类似的界面首先对进行验证的额文件进行转化操作，然后实时显示 ，默认的验证次数是1000次



图 4‑8

当然，也可以使用文本验证模式测试训练出来的模型是否真的具有实用价值：

图 4‑9

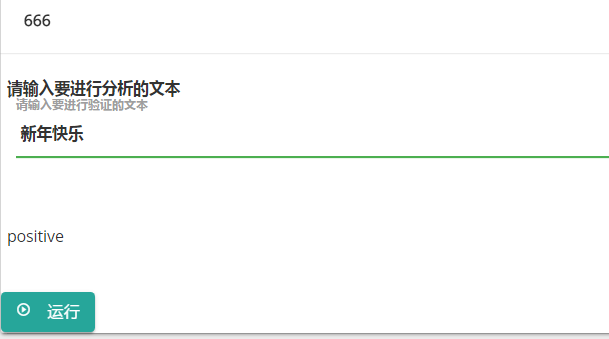


图 4‑10

直接输出文本的积极与消极分类结果。

本系统的目的旨在更好的演示模型的训练结果和操作，本身设计有很多不完善的地方，bug也比较多，这也是改进的方向之一

# 实验结果分析

建好模型与系统之后，我进行了一次数据训练，训练集的大小为6000，其中正、负文本各占1/2，训练总步数为1000步， 卷积核的数量为150个，含有两层隐含层，第一层中有300个神经元，第二层中有100个神经元，每一批的样本数量为128，学习率为0.01，指数平均的衰减速率为0.999，所得到的代价值和准确度的关系如图5-1所示，注意，此处的准确度不是真正意义上的衡量模型性能的准确度，而是每一批训练的数据的准确度，其意义是该模型与当前的数据集的拟合程度，可以看出，在运行到400步左右时，模型已经能够基本上完整拟合训练数据了，时间上每一批的训练只需要0.2s，总的学习时间大约在200s左右，这证明卷积神经网络具有较为良好的性能。

但是在使用外部集合进行验证的时候，正确率大概只用75%，

# 致谢

图 5‑1

1. 陆宇杰. 中文微博情感分析及其应用[D]. 华东师范大学, 2013. [↑](#endnote-ref-1)
2. Mccallum A, Nigam K. A Comparison of Event Models for Naive Bayes Text Classification[J]. IN AAAI-98 WORKSHOP ON LEARNING FOR TEXT CATEGORIZATION, 1998, 62(2):41--48. [↑](#endnote-ref-2)
3. Tong S, Koller D. Support vector machine active learning with applications to text classification[J]. Journal of Machine Learning Research, 2002, 2(1):45-66. [↑](#endnote-ref-3)
4. 孙建旺, 吕学强, & 张雷瀚. (2014). 基于词典与机器学习的中文微博情感分析研究. *计算机应用与软件,* *31*(7), 177-181. [↑](#endnote-ref-4)
5. Bengio Y, Ducharme R, Vincent P, et al. A neural probabilistic language model[J]. Journal of machine learning research, 2003, 3(Feb): 1137-1155. [↑](#endnote-ref-5)
6. Hubel D H, Wiesel T N. Visual area of the lateral suprasylvian gyrus (Clare—Bishop area) of the cat[J]. The Journal of Physiology, 1969, 202(1): 251. [↑](#endnote-ref-6)
7. Fukushima K. Neocognitron: A hierarchical neural network capable of visual pattern recognition[J]. Neural networks, 1988, 1(2): 119-130. [↑](#endnote-ref-7)
8. LeCun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86(11): 2278-2324. [↑](#endnote-ref-8)
9. Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks[C]//Advances in neural information processing systems. 2012: 1097-1105. [↑](#endnote-ref-9)
10. Zeiler M D, Fergus R. Visualizing and understanding convolutional networks[C]//European conference on computer vision. Springer International Publishing, 2014: 818-833. [↑](#endnote-ref-10)
11. Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition[J]. arXiv preprint arXiv:1409.1556, 2014. [↑](#endnote-ref-11)
12. He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016: 770-778. [↑](#endnote-ref-12)
13. LeCun Y, Boser B, Denker J S, et al. Backpropagation applied to handwritten zip code recognition[J]. Neural computation, 1989, 1(4): 541-551. [↑](#endnote-ref-13)
14. Zhou, Z. (n.d.). 机器学习. 1st ed. 清华大学出版社, pp.98-99. [↑](#endnote-ref-14)