南京理工大学
NANJING UNIVERSITY OF SCIENCE & TECHNOLOGY

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)：智能科学与技术

题　　目：基于卷积神经网络的文本分类

指导者：贾修一　　副教授

(姓　名)　　(专业技术职务)

(姓　名)　　(专业技术职务)

评阅者：　　副教授

(姓　名)　　(专业技术职务)

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

毕 业 设 计 说 明 书 中 文 摘 要

微博文本的情感分类，是自然语言处理在短文本分类上的一个应用，其具有的上下文信息少，文本长度短，口语化及网络化语言成分高等特点使传统分类方法难以实现，本文应用卷积神经网络实现微博的情感分类，首先对卷积神经网络的基本概念及原理进行了阐述，对核心模块的实现过程进行详细的说明和公式推导，在 6000 条微博文本上进行实验，并基于 tensorflow 构建框架，最后使用图形化的技术实现一个机器学习的简易系统，实验结果表明：模型在在测试集上运行的准确率为 75%。

关键词 机器学习 文本分类 卷积神经网络 自然语言处理

毕业设计说明书外文摘要

Title Text Classification

Based on Convolution Neural Network

Abstract

The emotional classification of weibo text is an application of natural language processing in short text classification. The text from weibo has the characteristics of less context information, short text length, colloquial and networked language composition, which makes the traditional classification method difficult to realize, This paper applies the convolution neural network on the essential classification of weibo text; first, It introduces the basic concepts and principles of the convolution neural network, after that it offers the detailed implementation of the core module and the formula derivation. The framework is built based on tensorflow, with 6000 weibo text on the experiment. Meanwhile, I apply the graphical technology to build a simple machine learning system, the experimental results show that the accuracy rate of the model can reach 75% on the test set.

Keywords Convolution Neural Network, Machine learning, Text Classification, Natural Language Processing

目 次

1 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，更是为人类掀起了技术革命的新篇章

1.1 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]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。

1.2 相关技术的现状

目前，实现文本分类的机器学习技术主要分为两个方向，一个是由专家制定相应的规则，应用这些规则进行分类，这种方式带来的明显的问题就是对人工的依赖，需要人为的制订规则，且在不同的应用领域，制定的规则也有所不同，这就需要耗费大量的人力和时间；人们就希望从繁琐的工作中解放出来，试图寻找文本本身中存在的规律，因此，便产生了基于统计学的分析方法，有支持向量机(SVM)[3]，朴素贝叶斯[2]，以及本文中提到的神经网络等，这些方法的特点是通过建立数学模型并人工标注大量的语言库，把这些标注好的数据集（也称训练集）输入该数学模型中，通过若干操作让模型学习到数据中所包含的规律，这种方法能较好的减少人的工作量，在实际使用中，也有结合了传统的人工制定规则方式的词典分类以及情感标签的分类方法[4]，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]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。1959 年，Hubel & Wiesel [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]。

20 世纪 90 年代，LeCun[8]等人发表论文，确立了 CNN 的现代结构，后来他使用 CNN 搭建了手写识别的模型，使用 backpropagation 算法训练，取名为 LeNet-5。

CNN 对于图像的处理能力是强大的，由于卷积操作的特性，它能够极其方便的从原始像素中提取相邻像素的规律特征。但是由于训练所需的计算能力不足，在当时，CNN 模型并没有取得较好的表现。

2006 年起，人们开始关注 CNN 在机器学习领域的潜力，并开始研究新的结构和算法。其中，在图像处理领域的最著名的是 Krizhevsky 等人提出了一个经典的 CNN 结构[9]，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], VGGNet[11], GoogleNet 和 ResNet[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。网络深度的增加带来的是网络拓扑结构的变化，更复杂的结构往往具有更强的非线性映射，但是这也带来一个问题，那就是在训练数据较少的时候，很容易出现过拟合的问题。

虽然在实验的测量中，CNN 的成绩已经证明了他的正确性，但是 CNN 还有许多需要提高的地方，首先，随着神经网络深度的增加，模型训练对于计算能力的要求不断提高，随之而来的，对于训练数据量的要求也在不断提高，而

人为搜集标签数据库要求大量的人力劳动。所以不需要标注数据，自动学习的无监督学习就成为了一个更有潜力也更有价值的研究方向。

同时，为了加速训练进程，虽然已经有一些异步的 SGD 算法[13]，证明了使用 CPU 和 GPU 集群可以在这方面获得成功。在训练的过程中，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。开发出开放，高效的学习系统，将对机器学习的进步有着巨大的价值。

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，虽然目前已经有学者已经搭建起了超参数自学习的模型，但是尚无较大的进展。所以，在学习式深度 CNN 架构的选择技巧上，存在巨大的提升空间。

1.3 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[14]，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[15]。

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。

1.4 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：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：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：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。

2 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。

2.1 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：[16]如图 2-1 所示

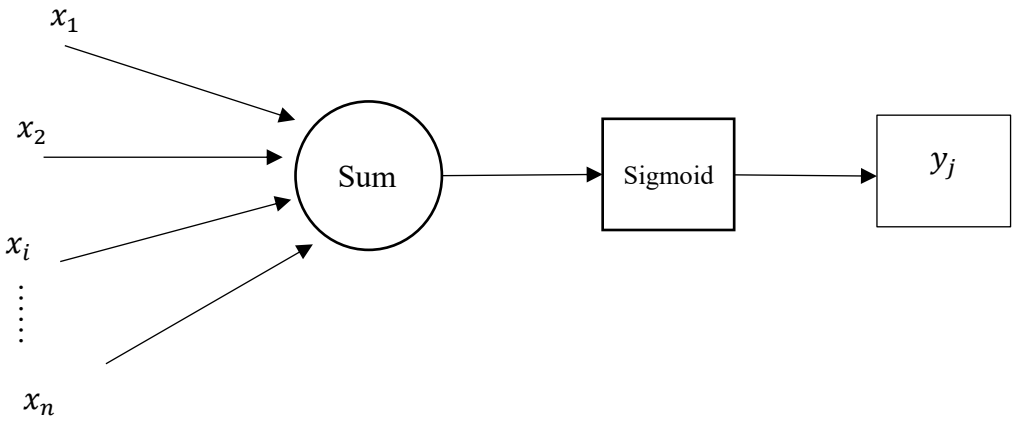


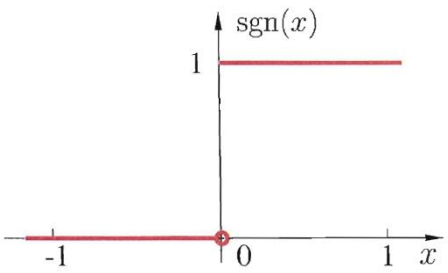
图 2-1 神经网络中神经元的结构

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y_j，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：

$$y_j = f(\Sigma_i w_i x_i - \theta) \qquad (1)$$

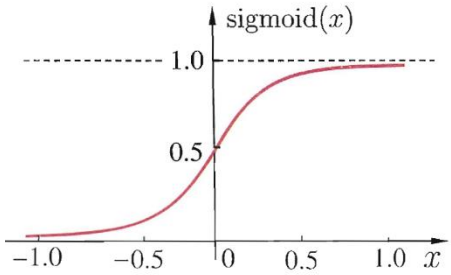
理想中的激活函数应当是如下图 2-2 的阶跃函数，但是由于在进行后续操作的过程中，会出现求导，求积分等运算，使用阶跃函数就无法很好的适应这样的计算，因为显然，该函数在 0 处不可导，而且在 0 处不光滑，这就导致在涉及到求导以及积分的时候必须要进行分类，增加了模型的复杂度。所以，当前主要采

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。



$$\operatorname{sgn}(x)=\begin{cases}1, & x\geqslant 0;\\ 0, & x<0.\end{cases}$$

图 2-2 阶跃函数的图像



$$\operatorname{sigmoid}(x)=\frac{1}{1+e^{-x}}$$

图 2-3 sigmoid 函数图像

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。

2.2 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$\hat{y}$，结果不正确，需要对模型的参数进行调整，使：

$$w_i \leftarrow w_i + \Delta w \qquad (2)$$

$$\Delta w = \eta(y - \hat{y}) \qquad (3)$$

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$D=\{(x_1,y_1),(x_2,y_2),\ldots,(x_m,y_m)\},x_i\in R^d,y_i\in R^l$即输入端含有$d$个属性，输出端有$l$个属性，为了方便说明，在模型中仅包含一个隐含层，可以得到下图的神经网络图[16]。

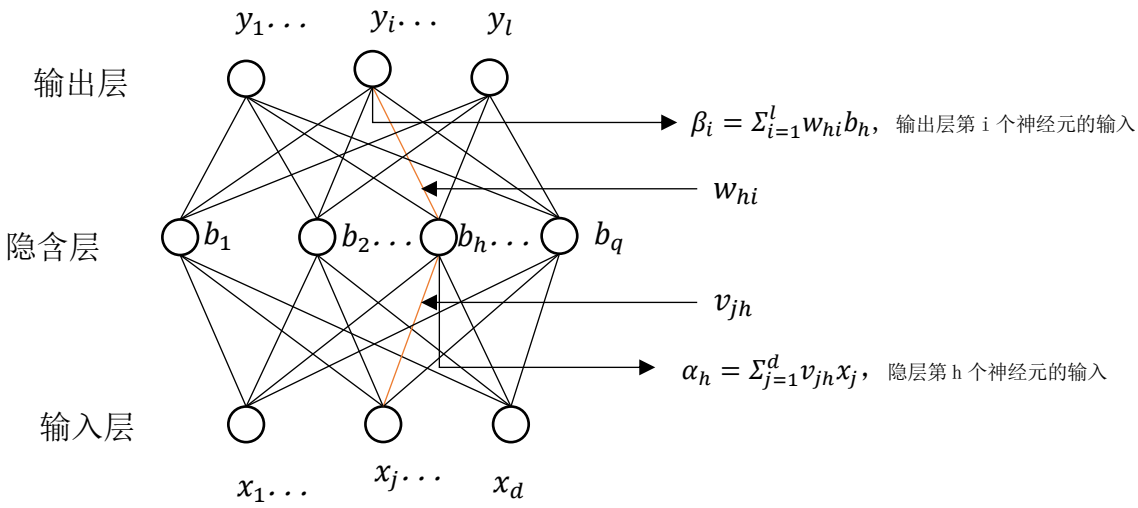


图 2-4 神经网络的基本结构

d个输入层，q个隐含层，l个输出层，其中输出层第i个神经元的阈值由θ_i表示，隐层第h个神经元的阈值由γ_h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v_{jh}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w_{hi}表示，可以得到隐层第h个神经元的输出$\alpha_h = \Sigma_{j=1}^{d} v_{jh} x_j$，输出层第$i$个神经元的输出$\beta_i = \Sigma_{i=1}^{l} w_{hi} b_h$,其中$b_h$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,神经网络在上述训练集中的输出为$\hat{y}_k = (\hat{y}_1^k, \hat{y}_{2,\ldots,}^k \hat{y}_l^k)$即：

$$\hat{y}_j^k = f(\beta_i - \theta_i) \qquad (4)$$

由 y 与$\hat{y}$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：

$$E_k = \frac{1}{2}\sum_{i=1}^{L}\left(\hat{y}_i^k - y_i^k\right)^2 \qquad (5)$$

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：

$$v = v + \Delta v \quad (6)$$

接下来就是要计算Δv的值了，这就要使用到梯度下降算法，E_k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：

$$\frac{\Delta E_k}{\Delta w_{hi}} = \frac{\partial E_k}{\partial w_{hi}} \quad (7)$$

由于代价函数E_k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E_k的值越来越小，所以，当目前的梯度值是正的，就表示随着w_{hi}的增加，E_k也会不断增加，而反之，如果梯度值是负的，那么，随着w_{hi}的增加，E_k则会不断变小，所以为了让E_k不断变小，就有：

$$\Delta w_{hi} = -\eta \frac{\partial E_k}{\partial w_{hi}} \quad (8)$$

由于w_{hi}首先影响的是第i个输出层的神经元的输入值β_i再影响到的E_k，所以有：

$$\frac{\partial E_k}{\partial w_{h_i}} = \frac{\partial E_k}{\partial \hat{y}_i^k} \cdot \frac{\partial \hat{y}_i^k}{\partial \beta_i} \cdot \frac{\partial \beta_i}{\partial w_{hi}} \quad (9)$$

显然

$$\frac{\partial \beta_i}{\partial w_{hi}} = b_h \quad (10)$$

另外，sigmoid 函数具有一个很好的性质：

$$f'(x) = f(x)\big(1 - f(x)\big) \quad (11)$$

于是，可以推出：

$$g_i = -\frac{\partial Ek}{\partial \hat{y}_i^k} \cdot \frac{\partial \hat{y}_i^k}{\partial \beta_i}$$

$$= -\big(\hat{y}_i^k - y_i^k\big) f'(\beta_i - \theta_i)$$

$$= \hat{y}_i^k\big(1 - \hat{y}_i^k\big)\big(y_i^k - \hat{y}_i^k\big) \quad (12)$$

经过推导，可得到

$$\Delta w_{hi} = \eta g_i b_h \quad (13)$$

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：

$$\Delta \theta_i = -\eta g_i \quad (14)$$

$$\Delta v_{jh} = \eta e_h x_j \quad (15)$$

$$\Delta \gamma_h = -\eta e_h \quad (16)$$

2.3 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$\mathbb{Z}$上的函数f，g我们定义卷积函数为：

$$(f * g)[n] \stackrel{\text{def}}{=} \sum_{m=-\infty}^{\infty} f[m]g[n-m]$$

$$= \sum_{m=-\infty}^{\infty} f[n-m]g[m] \quad (17)$$

其中，m表示卷积的窗口大小，默认为$(-\infty, +\infty)$，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$g[n]$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：

$$(f * g)[n] = \sum_{m=-M}^{M} f[n-m]g[m] \quad (18)$$

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：

$$(k * n)[x \cdot y] = \sum_{i=-\frac{a}{2}}^{\frac{a}{2}} \sum_{j=-\frac{b}{2}}^{\frac{b}{2}} (M[x+i, y+j] * k[i,j]) \quad (19)$$

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$a * b$, 该式表示的值是在$[x, y]$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。

2.4 逻辑函数

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：

$$h_\theta(x) = \frac{1}{1 + exp(-\theta^T x)} \quad (20)$$

而代价函数为：

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)}\, log\, h_\theta\left(x^{(i)}\right) + \left(1 - y(i)\right) log\left(1 - h_\theta\left(x^{(i)}\right)\right)\right] \quad (21)$$

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。

2.5 Dropout 操作

由于大规模的神经网络一般都会存在对训练样本集的过分适应，这也被称作过拟合，而为了减少模型与数据的过拟合，常见的操作是训练多组模型并且将他们混合起来，显而易见，这种做法是费时费力的，所以，在训练的时候就要对模型进行一定的变化，而 dropout 就是这么一个操作，它的基本思路是：每一个单元都有一定的概率被忽略，即设置一个概率p，每一个神经元的值可以这样表示：

$$d_i = a_i * \left((rand(0,1) < p)?\, 0 : 1\right) \quad (22)$$

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，这样对一个规模为M的神经网络，由于随机性，每一次训练的网络结构都不一样，无形中相当于增加到了X个神经网络，而这个X跟p以及训练的次数有关，。

3 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。

3.1 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。

3.1.1 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[4]。，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：

$$\frac{1}{T}\sum_{i=1}^{T}\sum_{-C\le j\le C} log\, p(w_{t+j}|w_t) \quad (23)$$

其中$p(w_{t+j}|w_t)$表示在出现w_t的情况下w_t之后第 j 个单词出现w_{t+j}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$p(w_{t+j}|w_t)$。

$$p(w_O|w_I) = \frac{e^{{v'_{w_O}}^{\top} v_{w_I}}}{\bar{z}_{w=1}^{W} e^{{v'_w}^{\top} v_{wI}}} \quad (24)$$

其中，v_w代表对应 w 的向量值，而w_O, w_I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。

3.2 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。

3.2.1 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：$x_i \in R^k$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：

$$x_{1:n} = x_1 \oplus x_2 \oplus \ldots \oplus x_n \quad (25)$$

其中$\oplus$是并置算符，就是让$x_{i:i+j}$作为单词$x_i, x_{i+1} \ldots, x_{i+j}$的并置。同时一个卷积操作还应当包括一个卷积核$w \in \mathbb{R}^{nk}$，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$h * k$，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

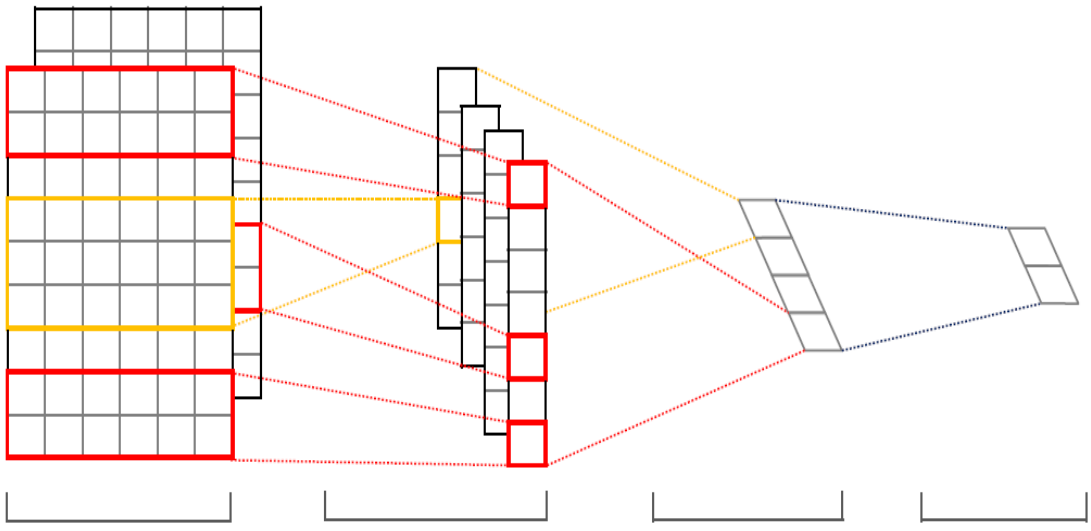


图 3-1 卷积神经网络的结构示意图

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c_i是一个从单词$x_{\mathrm{i:i+h-1}}$中生成的新特征，那么$\mathrm{c_i}$的生成公式为：

$$c_i = f\left(w \cdot x_{i:i+h-1} + b\right) \quad (26)$$

其中：$b \in R$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$\left\{x_{1:n}, x_{2:h+1,\ldots}x_{n-h+1:n}\right\}$中的每一个局部语块，从而产生一个特征映射

$$c = \left[c_1, c_{2,\ldots}c_{n-h+1}\right] \quad (27)$$

3.2.2 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

$$c = \left[c_1, c_{2,\ldots}c_{n-h+1}\right] \quad (28)$$

将其中的最大值$\hat{c} = max\{c\}$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

3.2.3 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：

$$l_2 = \frac{\boldsymbol{c}}{\sqrt{\sum {c_i}^2}} \quad (29)$$

其中$\boldsymbol{c}$表示原向量，而c_i表示向量中的第i个元素，该式所表示的含义就是用原始向量除以其所有元素的 L2 范式，该操作去除了输入向量中的线性元素，即同一个向量被任意放大或缩小，经过 L2 正则化之后，产生的向量是相同的，而不是判断成两个不同的结果，增强了模型的泛化能力。

3.2.4 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，如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。

3.2.5 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：

$$E_k = \frac{1}{2}\sum_{i=1}^{L}\left(\hat{y}_i^k - y_i^k\right)^2 \quad (30)$$

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：

$$\frac{\partial C}{\partial w} = (a-y)\sigma'(z)x = a\sigma'(z) \quad (31)$$

由 sigmoid 函数的图像（见图 2-3），可以看出该函数所具有的性质，在 z 的值较大时，$\sigma(z)$趋于平缓，而对应的$\sigma'(z)$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。

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。

$$C = -\frac{1}{n}\sum_{x}(y\,ln\,a + (1-y)\,ln(1-a)) \quad (32)$$

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$a = \sigma(z)$带入上式，得到：

$$\frac{\partial C}{\partial w_j} = -\frac{1}{n}\sum_{x}\left(\frac{1}{\sigma(z)} - \frac{(1-y)}{1-\sigma(z)}\right)\frac{\partial \sigma}{\partial w_j}$$

$$= -\frac{1}{n}\sum_{x}\left(\frac{1}{\sigma(z)} - \frac{(1-y)}{1-\sigma(z)}\right)\sigma'(z)x_j \quad (33)$$

化简之后得到：

$$\frac{\partial C}{\partial w_j} = \frac{1}{n}\sum_{x}\frac{\sigma'(z)x_j}{\sigma(z)\big(1-\sigma(z)\big)}(\sigma(z) - y) \quad (34)$$

利用 sigmoid 函数的定义：进行若干步推倒可以将上式化简为：

$$\frac{\partial C}{\partial w_j} = \frac{1}{n}\sum_{x} x_j(\sigma(z) - y) \quad (35)$$

由此可以发现，梯度值与$\sigma(z) - y$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：

$$C = -\frac{1}{n}\sum_{x}(y\,ln\,a + (1-y)\,ln(1-a)) + \sum w_i {v_i}^2 \quad (36)$$

其中，v_i表示要进行训练的参数，w_i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。

3.3 算法实现

3.3.1 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边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。

输入：原始向量/原始矩阵 m[x,y]， k[w,h] = rand()

输出：经过卷积操作的向量或者矩阵

step(1). initialize//初始化输出矩阵，其中 m[x,y]是输入的原始图像，k[w,h]随机生成，作为卷积核输入

step(2). for i from 0 upto x step sx,j from 0 upto y step sy:

a) if i+w≤x and j+h≤y then//判断该选择情况是不是

b) 求得在这一点的卷积操作，卷积核与原始图像的矩阵相乘

i. for q from 0 upto w, p from 0 upto h

ii. sum += m[i+q][j+p]*k[q][p]

iii. end for

step(3). end for

step(4). 输出：out 所指向的生成矩阵

其中，x,y 分别是输入矩阵的长和宽，h,w 分别是卷积核的长和宽，sx,sy 分别是卷积核在 x 方向和 y 方向上的步长，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。

3.3.2 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：

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

输出：经过神经网络计算的值

step(1). Initialize//w[len_out][len_in] = 该层神经网络的所有参数权重值组成的矩阵，b[len_out] = 参数阈值矩阵 out[len_out] = 输出向量

step(2). for i from 0 upto len_out

step(3). 计算该神经元的输出

a) for j from 0 upto len_in

b) sum += w[i][j]*I[j]

c) end for//第 7 步的循环结束

d) sum+=b[i]

step(4). end for

step(5). out[i] = sigmoid(sum)//对该神经元的输出应用激活函数

step(6). 第二步的循环结束

step(7). 输出：out[len_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。

3.3.3 梯度下降

从 2-2 章中推出的公式可以得到：

$$\Delta w_{hi} = \eta g_i b_h = \eta b_h \hat{y}_i^k \left(1 - \hat{y}_i^k\right)\left(y_i^k - \hat{y}_i^k\right) \quad (37)$$

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

$$\frac{\partial C}{\partial w_j} = \frac{1}{n}\sum_x x_j(\sigma(z) - y) \quad (38)$$

$$\Delta w_j = \eta \frac{\partial C}{\partial w_j} = \frac{1}{n}\eta \sum_x x_j(\sigma(z) - y) \quad (39)$$

所以，可以得出其计算的算法为：

输入：logits，label 分别时模型的输出结果和实际的标记结果

输出：delta，表示每一个参数应当的变化值

step(1). 输入：logits[len_out],label[len_out],wn,weight[wn] ,delta [wn] //weight 表示需要学习的权重参数全集，wn 表示权重参数的个数，delta 表示权重的变化量，用于输出

step(2). cost = 0

step(3). learning_rate = 学习率

step(4). 计算每一个输出神经元的输出值和实际值的差

a) for i from 0 upto len_out

b) cost += logit[i]-label[i]

c) end for

step(5). cost = cost*learning_rate/len_out

step(6). 计算每一个神经元的变化量

a) for i from 0 to wn

b) delta[i] = weight*cost

c) end for

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：

$$v_i = v_{i-1} * d + v_{new} * (1 - d) \quad (40)$$

其中 d 表示每一次计算移动平均时原参数的的衰减值，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。

4 系统实现以及测试

4.1 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 是谷歌基于最新研发的第二代人工智能学习系统，他的运行主要包括两个部分。Tensor（张量）表示数组，而且数组的大小没有限制，Flow（流）则意味着基于数据流图的计算，在使用 tensorflow 进行训练的时候，编程人员只需要使用他提供的库绘制自己的计算系统的图，并提供输入数据，模型便可以开始训练，tensorflow 所提供的各种库都是经过优化和测试的代码，具有较高的运行效率。

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。

4.2 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，测试集大小为 1000，其中正负情感各含 1/2

4.3 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爬取到的微博文本，是因为微博的文本所具有的时代性太强，词汇含义变化的很快，在语义

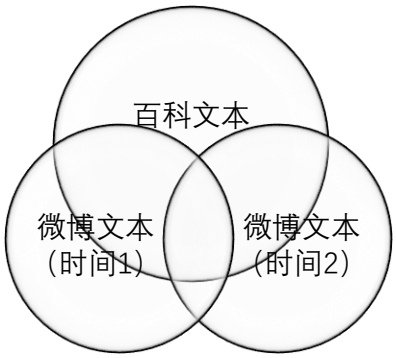


图 4-1 文本特征与采集时间的关系示意图

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。

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。

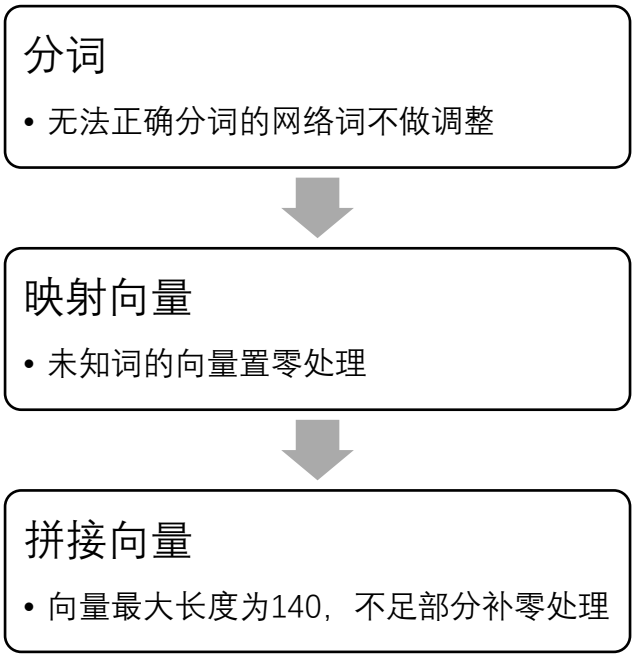


图 4-2 文本预处理流程示意图

4.4 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。

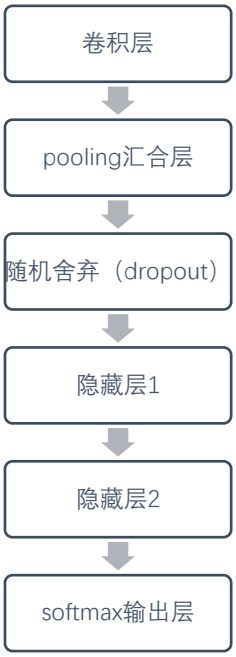


图 4-3 机器学习模型处理流程示意图

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，其具体步骤在 2.5 中已经介绍过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$[w_1, w_2, \dots, w_n]$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

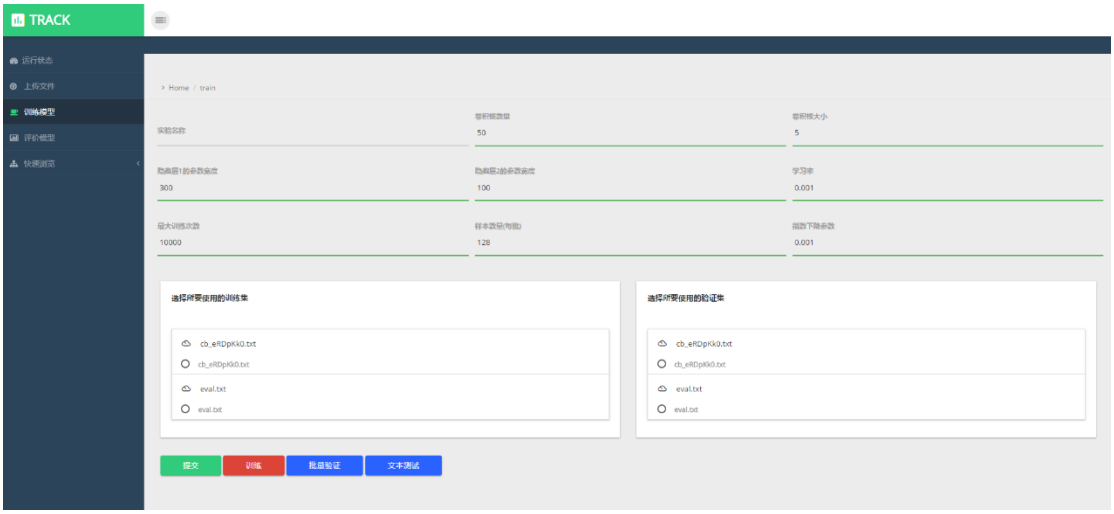


图 4-4 建立机器学习向量的训练任务界面

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）

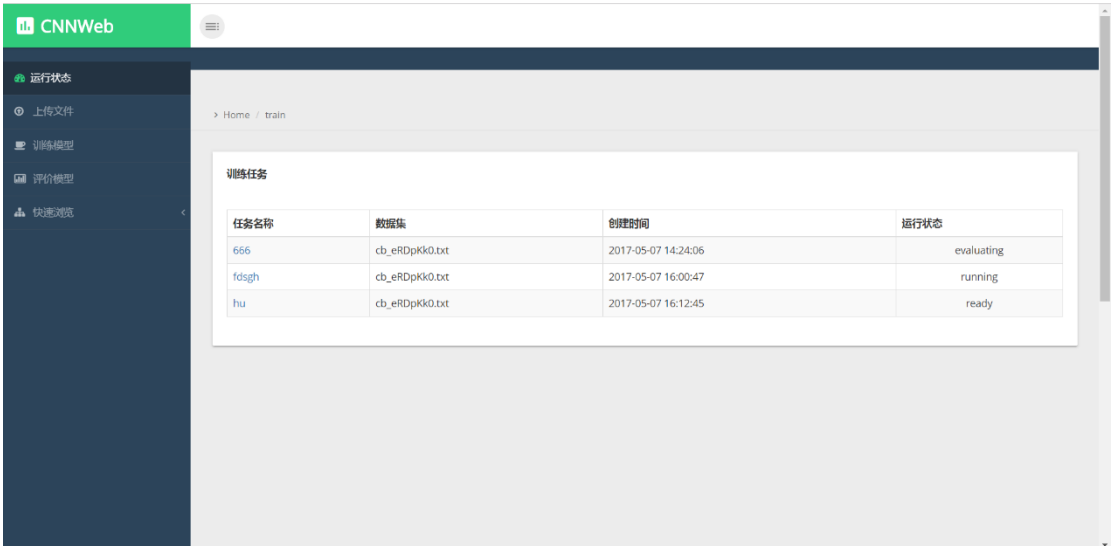


图 4-6 选择训练任务的界面

在点击了每一个训练之后，可以进入到训练修改界面，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。

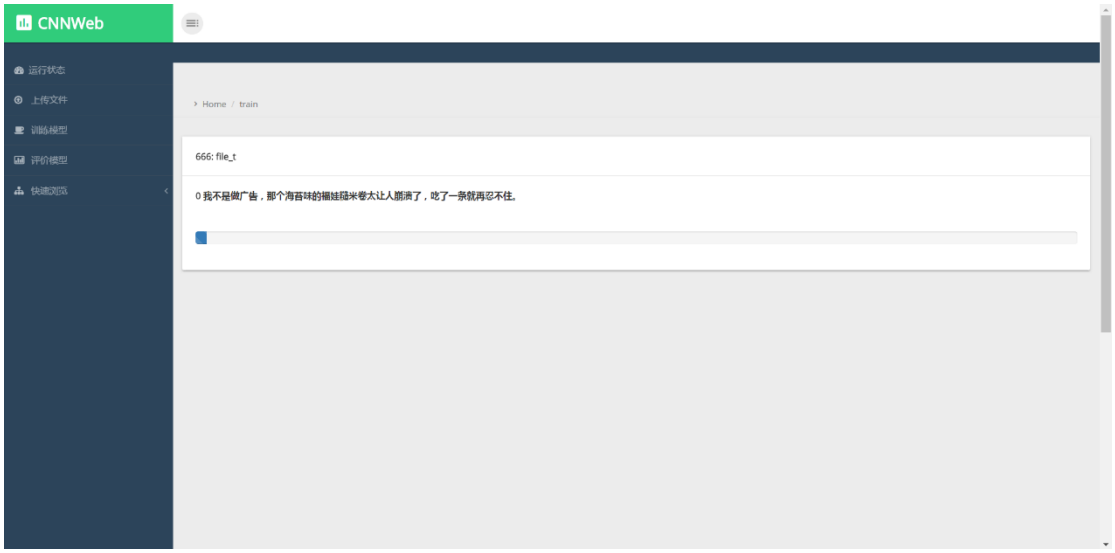


图 4-5 训练的过程界面

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。

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

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 4-8 验证过程监视界面

图 4-7 训练过程监视界面

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

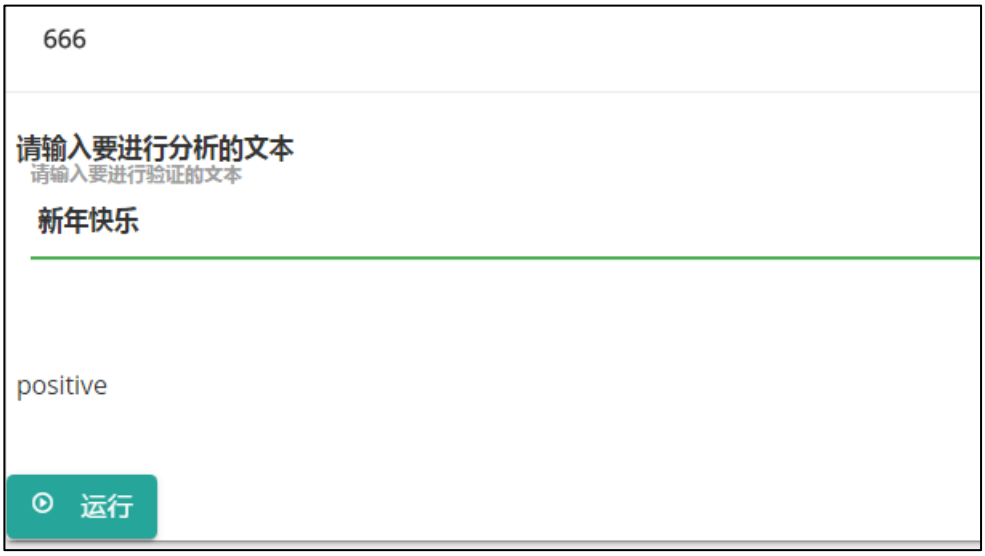


图 4-9 实时输入文本并进行验证

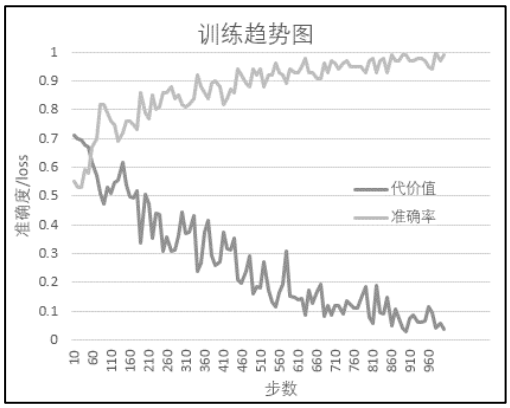
4.5 实验结果分析

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学习程度，可以看出，训练过程中，模型的准确率的值产生了一些抖动，但是并不影响整体的趋势是向上的，而且最终的准确率可以达到 0.99。

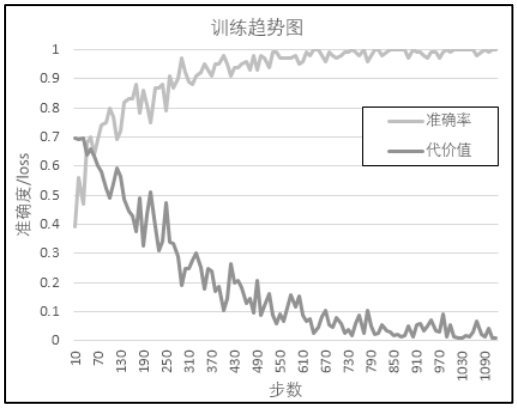
但是在使用外部集合进行验证的时候，正确率大概只有 77%，但是程序本身与训练数据的拟合度达到了极高的程度，对此，存在的可能有两种，一是训练数据本身存在缺陷，分类上存在模棱两可的情况，即同一个文本在特定的语境条件下会有两种不同的分类方式，而在测试集中这种语境可能会产生变化，从而使判断标准有了偏差，

二是本实验中的卷积神经网络训练的过程中出现了过拟合的情况。

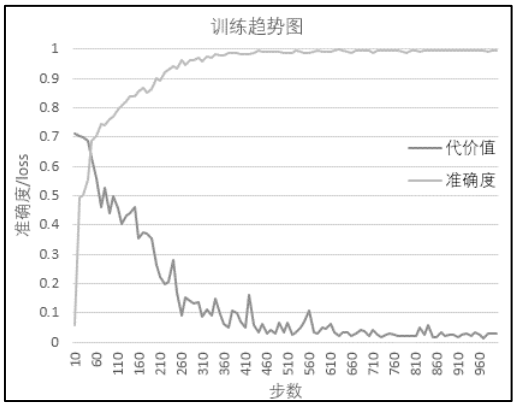
以此次试验为基准，通过调整不同的参数，我又进行了两次试验如下图(a)(b)所示。



(a)



(b)



(c)

图 4-10 实验数据

第二次实验中，我将模型的大小进行了调整，减少了所有隐藏层的神经元数量，卷积层数量不变，隐藏层 1 数量变为 100，隐藏层 2 数量变为 5，得到结果如图(b)所示，在测试集上得到的准确率值为 0.75

第三次实验，我们把 dropout 层的 p 设为 1，即所有的神经元都被激活，可以看出，模型在后期拟合的非常平滑，而且准确率非常的高，如图(c)，但是验证的结果却一般，可以看出模型确实发生了过拟合，而且正确率也有所下降，只有 75%。

通过比对(a)(b)两图可以发现，在当前的数据量和模型复杂度条件下，增加和减少模型的复杂度对于模型的性能是有一定影响的，但是这种影响并不是太大。

通过上述实验可以发现，现阶段卷积神经网络的性能提升的着眼点在于减少训练中的过拟合程度，由于数据集有限无法进行更多的比对实验，但是根据卷积神经网络的性质，在足够的训练集的条件下，模型的性能将会越来越好。

结　　论

文本的分类是人类近几年在机器学习领域着眼的热门问题，而卷积神经网络作为基于统计学的方法中最有潜力的方向，也受到了巨大的关注，以其优异的性能和良好的表现成为自然语言处理的有力工具。

本文主要介绍了使用卷积神经网络进行文本分类的基本流程和具体的实现方法，论述了其理论上的原理和工作方式，可以得出如下结论：卷积神经网络在进行机器学习的过程中具有理论上无限逼近训练模型的能力，只要具有足够多的参数和足够的训练能力，但这也势必会带来一些问题，比如数据的过拟合，虽然在实验中，我们已经采取了足够多的方式来避免过拟合的产生，但是从结果上来看，模型的过拟合现象还是存在并且比较严重。在撰写此文的时候，还尚无好的对策，所以，如何解决模型泛化能力弱的问题，将是未来工作的方向。

另一方面，我们可以发现卷积神经网络具有的一些良好的性质：训练一个中等规模的神经网络所需要的时间是分钟级的，即使对卷积层以及隐藏层的大小进行调整，计算的效率依然很高，这得益于卷积操作以及训练算法的高效性，当然，隐藏层的数量以及每层单元的增加都会提高计算复杂度，但是从实验的结果来看，这种单纯规模上的增加并不能显著提高模型的正确率，相对的，在训练数据的选择上，分词系统，以及词向量的系统等涉及到文本的语义分析的部分将是模型效率提高的着眼点，从实验的结果上看，模型的学习能力是无需质疑的，主要的矛盾应当就是由于特征提取的工作不能成分的提取文本具有的关键语义信息，进而导致模型的泛化能力弱，由于个人能力和精力有限，这些工作并没有完成，希望在将来有机会继续完成这些工作。

致　　谢

在刚刚接到这个题目的时候，我的感觉是踌躇满志又有些迷茫的，我对于机器学习这个领域很感兴趣，在文本分析中，一行行的文字是怎么让机器识别出来并找出其中的规律的，这一切让我很好奇，然而在实际操作的过程中一切并不是这么顺利，对编写代码的框架不熟悉、没有足够的知识储备、没有足够的训练数据、没有好的机器来进行运算，这些困难让我困恼过一阵，但是最终都解决了，在最后跑出我希望的计算结果的时候，我内心的喜悦和自豪是难以言喻的；然而我能做到这些，离不开各位老师，学长和同学们的帮助。

首先要感谢的是我的指导老师贾修一老师，他给出的学习书目让我在尽可能的短的掌握所需的知识，并且学会自学的途径；然后要着重感谢刘军煜学长，他耐心的为我讲解很多基础的问题，也为我 的模型搭建和系统构成提出了很多中肯的建议，这些建议让我少走了很多弯路，在论文的修改上也给出很多修改意见，让我论文逐渐完善；最后要感谢室友王浩宇同学，他提供的高性能机器让我的训练得以非常顺利的进行，节省了很多调试和训练的时间。

在这次的毕业设计中，有很多迷茫不懂得地方得到了很多同学的支持，大家一起进步一起学习，虽然研究的方向不一样，但是能帮忙的地方都尽量帮了一些忙，比如数据集的分享，爬虫软件的使用，感谢这次毕业设计的机会，让我能认认真真的做完这么一项完整的工作，也体会到了做学术的严谨和艰辛，以后会知难而上，继续努力。

参 考 文 献

[1] 陆宇杰. 中文微博情感分析及其应用[D]. 华东师范大学, 2013.

[2] McCallum A, Nigam K. A comparison of event models for naive bayes text classification[C]//AAAI-98 workshop on learning for text categorization. 1998, 752: 41-48.

[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.

[4] 孙建旺, 吕学强, 张雷瀚. 基于词典与机器学习的中文微博情感分析研究[J]. 计算机应用与软件, 2014, 31(7): 177-181.

[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.

[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.

[7] Fukushima K. Neocognitron: A hierarchical neural network capable of visual pattern recognition[J]. Neural networks, 1988, 1(2): 119-130.

[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.

[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.

[10] Zeiler M D, Fergus R. Visualizing and understanding convolutional networks[C]//European conference on computer vision. Springer International Publishing, 2014: 818-833.

[11] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition[J]. arXiv preprint arXiv:1409.1556, 2014.

[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.

[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.

[14]胡耀斌. 网络舆论主题探测、追踪与分析关键技术研究[D]. 山东财经大学, 2013.

[15]王玉斌. 数据挖掘技术在辅助决策中的应用研究[J]. [D]. 重庆大学, 2008.

[16] 周志华. 机器学习及其应用[M]. 清华大学出版社有限公司, 2006.

[17] Kim Y. Convolutional Neural Networks for Sentence Classification[C]. empirical methods in natural language processing, 2014: 1746-1751.

[18] Johnson R, Zhang T. Convolutional Neural Networks for Text Categorization: Shallow Word-level vs. Deep Character-level[J]. arXiv preprint arXiv:1609.00718, 2016.

[19] Sebastiani F. Machine learning in automated text categorization[J]. Acm Computing Surveys, 2002, 34(1):1-47.

[20] Pang B., Lee L., Vaithyanathan S. Thumbs up: sentiment classification using machine learning techniques[C]//Proceedings of ACL, 2002: 79-86.

[21] Xu R.F, Wong K.F,Xia Y. Coarse-Fine opinion mining-WIA in NTCIR-7 MOAT task[C]//Proceedings of NTCIR, 2008: 307-313.

[22] Tan S., ZhangJ. An empirical study of sentiment analysis for Chinese documents[J]. Expert Systems with Applications, 2008, 34(4): 2622-2629

[23] Socher R., Perelygin A., Wu J.Y., et al. Recursive deep models for semantic compositionality over a sentiment Treebank[C]//Proceedings of EMNLP 2013: 1631-1642.

[24] Collobert R, Weston J, Bottou L, et al. Natural language processing (almost) from scratch[J]. Journal of Machine Learning Research, 2011, 12(Aug): 2493-2537.

[25] Bollegala D, Weir D, Carroll J. Using multiple sources to construct a sentiment sensitive thesaurus for cross-domain sentiment classification[C]//Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human

Language Technologies-Volume 1. Association for Computational Linguistics, 2011: 132-141.
[26] 谢丽星, 周明, 孙茂松. 基于层次结构的多策略中文微博情感分析和特征抽取[J]. 中文信息学报, 2012, 26(1): 73-83.
[27] Mnih A, Hinton G E. A scalable hierarchical distributed language model[C]//Advances in Neural Information Processing Systems. 2009: 1081-1088.
[28] Mikolov T, Sutskever I, Chen K, et al. Distributed representations of words and phrases and their compositionality[C]//Advances in Neural Information Processing Systems. 2013: 3111-3119.
[29] Mikolov T, Chen K, Corrado G, et al. Efficient estimation of word representations in vector space[J]. arXiv preprint arXiv:1301.3781, 2013.
[30] Socher R., Pennington J., Huang E.H, et al. Semi-supervised recursive autoencoders for predicting sentiment distributions[C]//Proceedings of ACL, 2011: 151-161.
[31] Johnson R., Zhang T. Effective use of word order for text categorization with convolutional neural networks[J]. Computing Research Repository, 2014: 1412.1058.
[32] Maas A.L, Daly R.E, Pham P.T, et al. Learning word vectors for sentiment analysis[C]//Proceedings of ACL, 2011: 142-150.
[33] Tang D., Wei F., Yang N., et al. Learning sentiment-specific word embedding for twitter sentiment classification[C]//Proceedings of ACL, 2014: 1555-1565.
[34] Faruqui M., Dodge J., Jauhar S.K, et al. Retrofitting word vectors to semantic lexicons[J]. Computing Research Repository, 2014: 1441.4166.
[35] Pang B, Lee L. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales[C]//Proceedings of the 43rd annual meeting on association for computational linguistics. Association for Computational Linguistics, 2005: 115-124.
[36] Collobert R, Weston J, Bottou L, et al. Natural language processing (almost) from scratch[J]. Journal of Machine Learning Research, 2011, 12(Aug): 2493-2537.