基于 RUSBoost 和积矩系数的神经网络优化算法*

尹化荣，陈 莉†，张永新，陈丹丹
（西北大学 信息科学与技术学院，西安 710127）

摘 要：针对单个神经网络分类准确率低、RUSBoost 算法提高 NN 分类器准确率耗时长的问题，提出了一种混合 RUSBoost 算法和积矩系数的分类优化算法。首先，利用 RUSBoost 算法生成 m 组训练集；然后，依据 Pearson 积矩系数计算每组训练集属性的相关程度消除冗余属性，生成目标训练集；最后，新的子训练集训练神经网络分类器，选择最大准确率分类器作为最终的分类模型。实验中使用了四个 Benchmark 数据集来验证所提算法的有效性。实验结果表明，提出的算法的准确率相较于传统的算法最多提升了 8.26%，训练时间最高降低了 62.27%。

关键词：神经网络；RUSBoost；积矩系数；集成学习
中图分类号：TP391 **文献标志码**：A **文章编号**：1001-3695(2018)09-2592-05
doi:10.3969/j.issn.1001-3695.2018.09.007

Optimization algorithm for neural network based on RUSBoost and correlation coefficient

Yin Huarong, Chen Li†, Zhang Yongxin, Chen Dandan
(*School of Information Science & Technology, Northwest University, Xi'an* 710127, *China*)

Abstract: Referring to the current problems that a single neural network classification accuracy is low, and RUSBoost is a time-consuming algorithm for improving the NN classifier accuracy. This paper proposed a hybrid algorithm which combined RUSBoost with the Pearson correlation coefficient to optimize the classifier of neural network. First of all, it generated m by using RUSBoost algorithm for group training. Then, according to the Pearson product-moment coefficient of correlation calculated the property of each group and eliminated redundant properties in the set, generating target training sets. Finally, it trained neural network classifiers by the new training set, and selected the maximum accuracy classifier as the ultimate model. It chose 4 Benchmark data sets to verify the effectiveness of the algorithm. Experimental results show that the accuracy of the algorithms presented compared to traditional algorithm for maximum lifting 8.26% , and the training timer reduced 62.27%.

Key words: neural network; RUSBoost; correlation coefficient; ensemble learning

0 引言

在众多的分类算法中，神经网络(neural network，NN)是一个重要的研究分支，已成功地应用于互联网文本挖掘、图像识别、经济金融预测、舆情分析和国家安全等领域。通常情况下，训练一个 NN 分类模型，包括数据预处理、训练模型和模型评估三个部分。尽管一些训练 NN 分类器的方法，如遗传算法、蚁群算法和粒子群等，通过模拟生物的特性训练神经网络分类器，以一种新的方式推动了神经网络的发展。但是这些方法存在容易陷入局部最优、在迭代后期收敛速度慢[1]、产生的 NN 分类能力较弱、泛化能力不强的问题，不能够最大限度地提高神经网络分类器的性能。Gutierrez-Osuna 等人[2]指出分类器模型的性能除了与模型本身有关外，还取决于数据预处理阶段所使用的方法。因此，在构建神经网络分类器时，预处理算法的选择是非常重要的。

在如何提高神经网络分类性能方面，集成学习吸引了越来越多学者的关注。集成学习通过训练多个神经网络的分类器共同作用的策略，能获得比单独使用任何一种学习算法都好的预测性能[3~5]。作为集成学习的重要算法——Boosting 算法近年来得到了很大的发展。Freund 等人[6]提出的 Discrete AdaBoost 算法使用均方误差函数作为损失函数训练分类器，并输出每个样本属于{-1,1}的概率。尽管该方法可以降低分类器模型的误差，且产生的分类器性能优于随机抽样方法，但 Discrete AdaBoost 算法每一个分类器输出的结果是 +1 或者 -1，不能精确地输出样本记录属于每个类的概率。Real AdaBoost 算法作为 AdaBoost 算法的改进算法，该方法使用对数函数作为损失函数，并通过训练权值输出每个弱分类器在(0,1)区间上样本属于某类的概率，成功地改进了 Discrete AdaBoost 算法输出结果不精确的问题[7~9]。虽然 Discrete AdaBoost 和 Real AdaBoost 算法的分类结果非常准确，但都过度增加了错误分类样本的权重，导致分类模型的泛化能力变差[10]。针对此问题，Gentle AdaBoost 算法通过使用牛顿迭代法平滑地调整权值，改善了负类别样本容易过拟合的问题。但是 Discrete AdaBoost、Real AdaBoost 和 Gentle AdaBoost 算法都存在错误分类越多精度损失就越大的问题。为此，Friedman 等人[11]提出了基于逻辑回归的 LogitBoost 算法，其使用最大似然函数求解方程，避免了分类错误越多损失就越大的问题。但是该方法的不足之处在于对不平衡数据集的分类能力较弱。Chawla 等人[12]提出了 SMOTEBoost 算法，通过抽样改变数据集的分布，

收稿日期：2017-05-08；**修回日期**：2017-06-21 **基金项目**：国家自然科学基金资助项目(61502219)；博士后基金资助项目(2015M582697)

作者简介：尹化荣(1990-)，男，陕西渭南人，硕士，主要研究方向为大数据与数据挖掘、机器学习；陈莉，女(通信作者)，博导，教授，主要研究方向为智能信息处理、大数据与数据挖掘(chenli@nwu.edu.cn)；张永新，男，副教授，博士，主要研究方向为图像智能处理；陈丹丹，女，硕士，主要研究方向为图像处理.

改进了 AdaBoost 算法容易对不平衡数据集过拟合的问题。Seiffert等人[13]优化了 SMOTEBoost 算法,提出 RUSBoost 算法,通过正则化技术训练神经网络的权值,简化了 SMOTEBoost 算法的训练过程。但是该方法也存在训练时间过长的问题。

综上可知,RUSBoost 算法虽然可以将神经网络的分类结果映射到(0,1)概率区间上,而且避免了神经网络在训练过程中精度损失过高的情况,改善了不平衡数据集容易过拟合的问题,提高了神经网络的识别率。但是在神经网络训练阶段,RUSBoost 算法要经过多次迭代计算权值,尤其是对高维数据集,需要耗费大量时间。

考虑到在众多降低维度的算法中 Pearson 积矩系数具有易于计算的特点,该方法通过计算高维度中每组属性的相关程度,选择合适的阈值消除冗余属性,以达到降低维度和运行时间的效果。针对 RUSBoost 算法存在的问题,本文提出了一种混合 RUSBoost 和 Pearson 积矩系数[14]的神经网络优化方法。该方法通过使用 Pearson 积矩系数降低 RUSBoost 算法中产生的子训练集维度,在神经网络分类器性能较高的情况下减少了运行时间。

1　相关工作

1.1　神经网络

感知器是最简单的神经网络模型,最早由 Rosenblatt 提出[15]。通常情况下,一个感知器模型就可以解决分类问题,但是感知器无法解决一些具有异或特性的问题,因此一般意义上的神经网络为一个两层神经网络,其结构如图 1 所示。设 p、q、r 分别表示输入层、隐含层和输出层的神经元个数,x_1,x_2,…,x_p 为输入层,l_1,l,…,l_q 为隐含层,y_1,y_2,…,y_r 为输出层。图 1 中 $w_{i,j}^{[1]}$ 表示第 1 层从 x_j 到 l_i 连接的权值,$w_{i,j}^{[2]}$ 表示第 2 层从 l_j 到 Y_i 连接的权值。设隐含层和输出层的转移函数分别为 $f^{[1]}(x)$ 和 $f^{[2]}(x)$,则输出层计算公式为

$$y_j = f^{(2)}\left(\sum_{i=1}^{q}(l_i \times w_{ji}^{[2]} + b_j^{[2]})\right) \tag{1}$$

其中:l_i 由式(2)计算。

$$l_j = f^{(1)}\left(\sum_{i=1}^{p}(l_i \times w_{ji}^{[1]} + b_j^{[1]})\right) \tag{2}$$

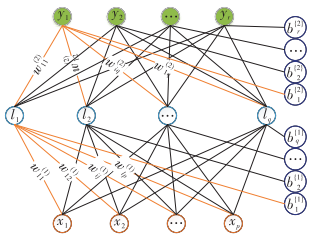


图1　两层神经网络结构

一般情况下,隐含层的节点越多,神经网络模型的准确率越高,但是训练时间也会越长。文献[16]中指出,隐含层神经元个数 q 计算公式如下:

$$q < \sqrt{p+r} + a \quad a \in [0,10] \tag{3}$$

在上述模型中,神经元与神经元之间连接的权值是未知的,无法将该模型用于实际的分类问题中,因此需要选择一种学习算法以确定神经元之间连接的权值。在神经网络学习算法中,网络中权值的调整方式为

$$\Delta W_{ij} = \eta s[w_{ij}, X, d_j]X \tag{4}$$

其中:w_{ij}是神经元 i 到 j 的权值;X 为输入向量;s 为学习信号,由选择的学习算法确定;d 为期望输出;η 为学习率。更加详细的学习过程在 1.3 节中说明。

1.2　RUSBoost 算法

在分类器的改进方法之中,RUSBoost 算法较好地改善了单个分类器分类能力较弱的问题,同时该方法也较好地解决了非平衡数据集分类能力差的问题。RUSBoost 算法的基本思想是:若训练集有 n 条记录,将数据集分割为 t 份;接着产生 n/t 数量的随机权值作为子数据集中每个记录初始的权值;然后使用每个子数据集训练模型 h_t,并计算 pseudo-loss 值 ϵ_t;最后使用正则化的方法更新下一次迭代需要的训练集记录的权重,直到满足限制条件为止。每个子数据集重复上述过程训练分类器,最后从训练好的所有模型中选择最好的作为最终模型。该算法的详细计算过程如算法 1 所示。

算法 1　RUSBoost 算法

输入:

集合 $S,(x_1,y_1),(x_2,y_2),\cdots,(x_m,y_m)$;

弱分类器,WeakLearner;

迭代次数 T。

a)初始化每个记录的权值 $D_i = \dfrac{1}{m}$

b)for $t = 1,2,\cdots,T$

(a)由 a)创建临时训练集 S'_t,并随机产生 S'_t 对应的权值 D'_t

(b)调用 WeakLearner,并传入参数 S'_t 和 D'_t

(c)返回模型 h_t

(d)计算 pseudo-loss

$$\epsilon_t = \sum_{(i,y):y_i \neq y} D_t(i)(1 - h_t(x_i,y_i) + h_t(x_i,y))$$

(e)计算权值更新参数

$$\alpha_t = \frac{\epsilon_t}{1-\epsilon_t}$$

(f)更新 D_t:

$$D_{t+1}(i) = D_t(i)\alpha_t^{\frac{1}{2}(1+h_t(x_i,y:y\neq y_i))}$$

(g)正则化 D_{t+1}:

$$Z_t = \sum_i D_{t+1}(i), D_{t+1}(i) = \frac{D_{t+1}(i)}{Z_t}$$

end for

输出:最终分类器为

$$H(x) = \underset{y\in Y}{\operatorname{argmax}} \sum_{t=1}^{T} h_t(x,y)\log\frac{1}{\alpha_t}$$

1.3　Pearson 积矩系数

在数据预处理的过程中,通常存在训练集中样本的维度过高、使用原训练集训练模型需要花费的时间较长问题。因此,消除数据集中的冗余属性成为一个备受关注的问题。常见的消除冗余的方法有 χ^2 相关检验法、Pearson 积矩系数法和协方差分析法等。考虑到 Pearson 积矩系数的计算方式简洁、易于计算机的计算、有利于减少 RUSBoost 算法训练时间的优点,本文采用 Pearson 积矩系数法来消除冗余属性,降低数据集的维度。具体如下:

设 X、Y 分别为两个属性,$r_{X,Y}$表示 X、Y 两个维度的相关程度,且 $-1 \leqslant r_{X,Y} \leqslant +1$。$|r_{X,Y}|$的值越高,说明 X、Y 属性相关性越强,表明属性 X 相对于属性 Y 来说是冗余属性,或者属性 Y 相对于属性 X 来说是冗余属性,可以通过删除其中一个属性进行降维。Pearson 积矩系数的计算公式为

$$r_{X,Y} = \frac{\sum_{i=1}^{n}(x_i-\bar{X})(y_i-\bar{Y})}{n\sigma_X\sigma_Y} = r_{X,Y} = \frac{\sum_{i=1}^{n}(x_iy_i) - n\bar{X}\bar{Y}}{n\sigma_X\sigma_Y} \tag{5}$$

其中:$\sum_{i=1}^{n}(x_i \cdot y_i)$是 X、Y 的点积;n 是样本元组的个数;σ_X、σ_Y 分别是属性 X 和 Y 的标准差;$\bar{X}$ 和 $\bar{Y}$ 分别为属性 X 和 Y 的均值;x_i 和 y_i 为元组 i 在属性 X 和 Y 上的值。但是在实际应用中,通过统计得到的样本缺乏代表性,不能代表全部样本的分布,因此 $r_{X,Y}$的值计算公式变为

$$r_{X,Y} = \frac{\sum_{i=1}^{n}(x_i-\bar{X})(y_i-\bar{Y})}{nS_XS_Y} = r_{X,Y} = \frac{1}{n-1}\frac{\sum_{i=1}^{n}(x_iy_i) - n\bar{X}\bar{Y}}{nS_X\sigma S_Y} \tag{6}$$

其中:S_X 和 S_Y 分别是属性 X 和 Y 的方差。

1.4　评估指标

评估分类器预测性能的主要度量方法包括准确率、误分类

率、敏感度、特效性和精度。其中,准确率(accuracy)指分类器正确分类元组在数据集中的百分比,它反映了分类器对各类元组的正确识别情况。准确率的计算公式[17]为

$$\text{accuracy} = \frac{TP + TN}{P + N} \tag{7}$$

从式(7)中可看出,当数据集分布平衡时,使用 accuracy 评估模型最有效。衡量分类器性能另一组指标为精度(precision)和召回率(recall)。Precision 为查准率,表示由分类器分类为正元组的数目占实际正类的百分比。Recall 是查全率,表示分类器标记为正元组的百分比。Precision 和 recall 的计算公式为

$$\text{precision} = \frac{TP}{TP + FP} \tag{8}$$

$$\text{recall} = \frac{TP}{TP + FN} = \frac{TP}{P} \tag{9}$$

最后,均方误差(mean squared error,MSE)是衡量平均误差的一种较方便的方法,可以评价数据的变化程度,指估计值与真实值之差平方的期望值。MSE 的值越小,预测模型的精确度越好。MSE 的计算公式为

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\text{observed}_t - \text{predicted}_t)^2 \tag{10}$$

考虑到不同的评估方法适用于不同的情况,采用 accuracy、precision、recall、MSE 四个指标综合衡量分类器性能。

2 基于 RUSBoost 和积矩系数的神经网络分类优化算法

为了解决 RUSBoost 算法训练神经网络模型用时较长的问题,本文提出了一种 RUSBoost 与 Pearson 积矩系数相结合的神经网络优化算法,记为 RPN 算法。该算法不仅充分考虑了单个神经网络分类能力较弱的问题,同时克服了构建多个神经网络训练时间较长的问题。RPN 算法的思想是:a)利用 RUSBoost 算法生成 m 组训练集;b)依据 Pearson 积矩系数计算每组训练集属性的相关程度消除冗余属性,产生新的子训练集;c)新的子训练集结合 RUSBoost 生成分类器,选择最大准确率分类器作为最终的分类模型。对属性消除冗余的过程参考算法 2,RPN 算法流程如图 2 所示。

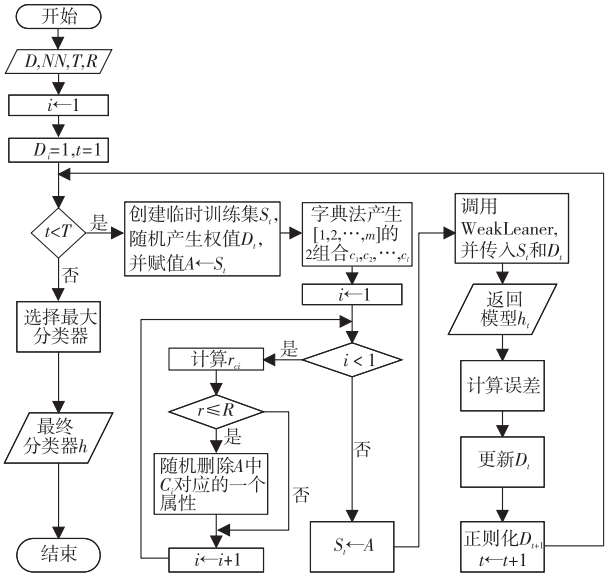


图2 RPN算法流程

需要特别说明的是,在 RPN 算法采用 m 组数据集分别训练模型时,利用了 BP 学习法[17,18]的思想。但是 RPN 算法与 BP 算法不同,RPN 算法通过选择 ReLU 函数[19,20](式(11))作为神经网络的转移函数,避免了 BP 算法中 Sigmoid 函数求偏导数速度慢的问题,加速了权值的学习过程。此外,输出层的神经元选择 TanH 函数作为激活函数,其方程为式(12),两种函数对应的图形如图 3 所示。

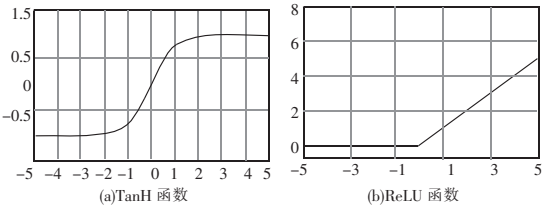


图3 转移函数

$$f(x) = \begin{cases} 0 & x < 0 \\ x & x \geqslant 0 \end{cases} \tag{11}$$

$$f(x) = \tanh(x) = \frac{2}{1 - e^{-2x}} - 1 \tag{12}$$

在 RPN 算法中,学习信号 s 的计算公式为

$$s = \lfloor d_j - f(W_j^T W) \rfloor f'(W_j^T W) = (d_j - Y_j) f'(\text{net}) \tag{13}$$

其中:f 为可导转移函数;$f'(\text{net}_j)$ 与具体所选的激活函数有关。由式(5)和(13)可以推出

$$\Delta W_{ij} = \eta (d_j - Y_j) f'(\text{net}) X \tag{14}$$

考虑到通过似然方程法求解权值 W 是非常困难的,通常情况下,采用下面的等价公式来调整权值:

$$E(W) = \frac{1}{2}(d_j - Y_j)^2 = \frac{1}{2}\sum [d_j - f(W_j^T W)]^2 \tag{15}$$

$$\nabla E = \frac{\partial}{\partial w} E(w) = (d_j - Y_j) f'(\text{net}_j)(-X) \tag{16}$$

其中:E 是 w_{ij} 的函数;$f'(\text{net}_j)$ 与具体所选的激活函数函数有关。从式(15)(16)中可以看出,E 与误差负梯度方向成正比,通过求 E 的最小值,可以求出 W。最后结合式(4)和(13)~(16)可推出

$$\nabla E = s(-X) \tag{17}$$

$$\Delta W_{ij} = -\eta \nabla E \tag{18}$$

$$W^{\text{new}} = W^{\text{old}} + \Delta W_{ij} = W^{\text{old}} - \eta \nabla E \tag{19}$$

令 $Z(x) = w_0 + \sum w_j \times x_j$,将 ReLU 函数的方程代入式(12)中,可得 RPN 算法输出层的梯度计算公式为

$$\nabla E = \frac{\partial}{\partial w} E(w) = \frac{\partial}{\partial y} E(w) \times \frac{\partial y}{\partial z} \times \frac{\partial z}{\partial w} = (d_j - Y_j) \times 1 \times (-X) \tag{20}$$

而由式(1)可知,Y 是权值 W 的函数,因此给定初值 W_0 和步长 η 即可迭代求出 E 的局部最小值,并求出对应的 W。最终通过多次迭代并评估迭代结果,选出最佳的权值 W 生成神经网络。

算法 2 RPN 算法

输入:

集合 $D,(x_1,y_1),(x_2,y_2),\cdots,(x_m,y_m)$

分类器,BP-NN

迭代次数,T

积矩系数阈值,R。

a)基于字典序法产生[1,2,…,m]的 2 组合 $c_1,c_2,\cdots,c_l$

b)Pearson 积矩系数降维训练集 A

for $i = 1,2,\cdots,l$

根据式(5)计算 c_i 对应属性的积矩系数 r

if $r \geqslant R$

then 随机删除 c_i 对应的一个属性

end for

赋值:$S \leftarrow A$

输出集合 S

c)将集合 S、神经网络 BP-NN 和迭代次数 T 输入到 RUSBoost 算法计算

输出:最终的神经网络。

3 结果与分析

为了更好地评价提出的 RPN 算法,在实验过程中采用四个基准数据集,分别是 Chronic Kidney Disease[21]、Parkin-

sons[22]、Vertebral Column[23]、Connectionist Bench[24],并与 AdaBoost、LogitBoost、RUSBoost 算法作了对比。实验中的参数设置为相关系数阈值 $R=0.6$,迭代次数 $T=1\ 000$,学习率 $\eta=0.1$,隐含层神经元的个数 q 由式(3)计算。具体的实验运行环境见表 1。

表 1　实验运行环境

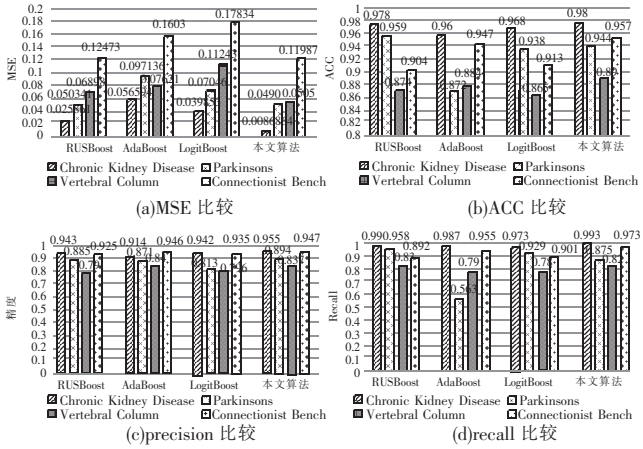
设备	配置	设备	配置
OS	Windows 10 专业版 64 bit	显卡	ATI HD 6770
CPU	Xeon X5670 2.93 GHz 两颗 12 核 24 线程	编程工具	MATLAB 2016b x64
RAM	96 GB	是否并行	MATLAB 并行编程

3.1　RPN 算法的对比实验分析

本节对比了提出的 RPN 算法与传统的 RUSBoost、AdaBoost 和 LogitBoost 算法在四个 Benchmark 数据集上的性能,并通过表 2 和图 4 显示了实验结果。实验结果表明,在数据集 Chronic Kidney Disease 上,RPN 算法的均方误差比其他三种算法小一个数量级,并且准确率、精度和召回率分别最大提高了 2.08%、4.49% 和 2.06%;在 Parkinsons 数据集上,RPN 算法比较原始算法降低了近 50% 均方误差值,准确率和精度分别最大提高了 8.26% 和 9.96%;在 Vertebral Column 数据集上,RPN 算法的 MSE 值均比其他三种算法低,准确率、精度和召回率最大提高了 2.89%、5.95% 和 5.13%;在 Connectionist Bench 数据集上,本文 RPN 算法的准确率、精度和召回率分别最大提高了 5.89%、2.38% 和 9.08%;在剩余的对比实验中,提出的 RPN 算法也有着不同程度的提升。更多的实验数据参见表 2 和图 4。

表 2　RPN 算法性能比较

比较项	数据集	RUSBoost	AdaBoost	LogitBoost	本文算法
MSE	Chronic Kidney Disease	0.025 861	0.056 594	0.039 853	0.008 686
	Parkinsons	0.050 341	0.097 136	0.070 46	0.049 356
	Vertebral Column	0.068 98	0.076 21	0.112 43	0.05
	Connectionist Bench	0.124 73	0.160 3	0.178 34	0.119 87
ACC	Chronic Kidney Disease	0.978	0.96	0.968	0.98
	Parkinsons	0.959	0.872	0.938	0.944
	Vertebral Column	0.874	0.884	0.865	0.89
	Connectionist Bench	0.904	0.947	0.913	0.957
precision	Chronic Kidney Disease	0.943	0.914	0.942	0.955
	Parkinsons	0.885	0.871	0.813	0.894
	Vertebral Column	0.79	0.84	0.796	0.837
	Connectionist Bench	0.925	0.946	0.935	0.947
recall	Chronic Kidney Disease	0.99	0.987	0.973	0.993
	Parkinsons	0.958	0.563	0.929	0.875
	Vertebral Column	0.83	0.79	0.78	0.82
	Connectionist Bench	0.892	0.955	0.901	0.973



(a)MSE 比较　(b)ACC 比较

(c)precision 比较　(d)recall 比较

图 4　RPN 算法性能比较

表 3 说明了 RPN 算法的神经网络结构设置。从表中的数据可以看出,RPN 算法网络中隐含层神经元个数皆小于 11,而且在四个数据集上的准确率可达到 97.80%、94.40%、87.40% 和 95.70%,召回率和精度最高也可达到 97% 以上,说明 RPN 算法可以在简单的模型结构中获得较好的性能。

表 3　RPN 算法的网络结构

数据集	MSE	ACC/%	precision/%	recall/%	layer number
Chronic Kidney Disease	0.008 686	97.80	97.30	96.70	7
Parkinsons	0.049 35	94.40	89.40	87.50	7
Vertebral Column	0.689 8	87.40	79.00	83.00	5
Connectionist Bench	0.119 87	95.70	94.70	97.30	10

本节的实验结果说明,在四个数据集上进行四种算法的比较实验中准确率较高的情况下,模型的精度和召回率也保持在较高的水平。召回率越高表示本文算法在实际样本测试中越准确,精度高说明测试样本准确率在输出中所占比率大,两者从不同的角度说明了本文提出的 RPN 算法是有效的,可以提升神经网络的分类能力。

3.2　RPN 算法复杂度分析

在本节将进行 RPN 算法与表 4 中的四个 Benchmark 数据集上所用时间的比较。实验结果表明,在 Chronic Kidney Disease数据集的实验结果中,RPN 算法较传统的 RUSBoost、AdaBoost和 LogitBoost 算法的运行时间分别减少了 13.52%、36.51% 和 30.04%;在同样的对比实验设置下,RPN 算法在 Parkinsons 数据集上比传统算法最大降低了 62.27%,平均降低了 40.88% 的运行时间;在 Vertebral Column 数据集上,RPN 算法比传统的三种算法分别缩短了 19.48%、54.38% 和 20.87% 的运行时间;在 Connectionist Bench 数据集上,提出的 RPN 算法虽然降低幅度较小,但最小也降低了 7.34% 的运行时间。

表 4　RPN 算法运行时间比较实验　/s

数据集	RUSBoost	AdaBoost	LogitBoost	本文算法
Chronic Kidney Disease	8.315	11.327	10.279	7.191
Parkinsons	4.246	10.951	9.767	4.132
Vertebral Column	5.379	9.493	5.473	4.331
Connectionist Bench	4.763	4.116	4.632	3.814

由上文可知,RPN 算法的时间复杂度与 RUSBoost 分割子集数 m、迭代次数 T 和训练集大小 n 有关。由于字典法生成组合的时间复杂度为 $n!$,所以本文算法的时间复杂度为 $T\times((n/m)!)$;又因为 m 和 T 的大小是常数,所以 RPN 算法的时间复杂度为 $n!$。

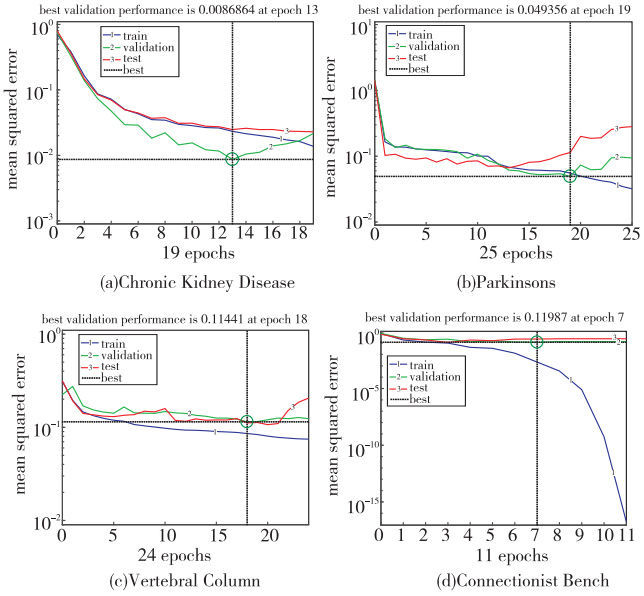
通过图 5 展示了 PRN 算法的迭代过程。在四个数据集的实验结果中,RPN 算法分别在第 19 次迭代、第 25 次迭代、第 20 次迭代和第 11 次迭代后得到最佳的神经网络模型,同时 RPN 算法获得较小训练误差和测试误差所用的迭代次数也较小,说明提出的 RPN 算法可以在短时间内获得最佳的神经网络模型。

本节的实验结果表明,在 16 组对比实验中,RPN 算法的运行时间均比其他三种算法少,同时 RPN 算法也可以在较小的迭代次数下获得较小的分类误差。总体来看,使用 RPN 算法获得的分类器在保证较小的误差情况下,也可以实现神经网络模型的快速生成。

4　结束语

针对单个神经网络分类器准确率低,且 Boosting 算法用于神经网络用时长的问题,本文基于 RUSBoost 和积矩系数提出了一种新的神经网络优化算法(RPN 算法)。RPN 算法与现有方法的不同之处在于,在利用 RUSBoost 方法提升神经网络的识别率时,结合了积矩系数降维来降低算法的总体运行时间。在实验分析中,本文选取了四个 Benchmark 数据集,并与 AdaBoost、LogitBoost、RUSBoost 算法作了对比,从多个层面验证了 RPN 算法的有效性。RPN 算法的性能对比实验分析表明,提出的 RPN 算法能够在简单的网络结构下有效地提高神经网络

模型的准确率、精度和召回率，降低神经网络模型的均值误差。RPN 算法运行时间的实验结果分析表明，RPN 算法比传统算法的运行时间最高降低了 62.27%，说明本文算法可以减少生成模型的时间。另外，本文提出的 RPN 算法存在实验流程较为复杂的问题，这将是下一步的工作和研究的方向。



(a)Chronic Kidney Disease (b)Parkinsons

(c)Vertebral Column (d)Connectionist Bench

图5 RPN算法MSE迭代过程

参考文献：

[1] 张超群，郑建国，王翔. 蜂群算法研究综述[J]. 计算机应用研究，2011，28(9)：3201-3205，3214.

[2] Gutierrez-Osuna R，Nagle H T. A method for evaluating data-preprocessing techniques for odour classification with an array of gas sensors [J]. IEEE Trans on Systems，Man，and Cybernetics，Part B：Cybernetics，1999，29(5)：626-632.

[3] Polikar R. Ensemble based systems in decision making[J]. IEEE Circuits and Systems Magazine，2006，6(3)：21-45.

[4] Rokach L. Ensemble-based classifiers[J]. Artificial Intelligence Review，2010，33(1-2)：1-39.

[5] Wang Xizhao，Xing Hangjie，Li Yan，*et al*. A study on relationship between generalization abilities and fuzziness of base classifiers in ensemble learning[J]. IEEE Trans on Fuzzy Systems，2015，23(5)：1638-1654.

[6] Freund Y，Schapire R E. Experiments with a new boosting algorithm [C]//Proc of the 13th International Conference on Machine Learning. 1996.

[7] Freund Y，Schapire R E. A desicion-theoretic generalization of on-line learning and an application to boosting[C]//Proc of the 2nd Annual European Conference on Computational Learning Theory. Orlando，FL：Academic Press，Inc，1995.

[8] Huang Chang，Wu Bo，Haizhou A I，*et al*. Omni-directional face detection based on real adaboost[C]//Proc of International Conference on Image Processing. Piscataway，NJ：IEEE Press，2004.

[9] Wu Shuqiong，Nagahashi H. Parameterized AdaBoost：introducing a parameter to speed up the training of real AdaBoost[J]. IEEE Signal Processing Letters，2014，21(6)：687-691.

[10] Schapire R E，Singer Y. Improved boosting algorithms using confidence-rated predictions[J]. Machine Learning，1999，37(3)：297-336.

[11] Friedman J，Hastie T，Tibshirani R. Additive logistic regression：a statistical view of boosting (with discussion and a rejoinder by the authors)[J]. The Annals of Statistics，2000，28(2)：337-407.

[12] Chawla N V，Lazarevic A，Hall L O，*et al*. SMOTEBoost：improving prediction of the minority class in boosting[C]//Proc of European Conference on Principles of Data Mining and Discovery in Databases. 2003：107-119.

[13] Seiffert C，Khoshgoftaar T M，Van Hulse J，*et al*. RUSBoost：a hybrid approach to alleviating class imbalance[J]. IEEE Trans on Systems，Man，and Cybernetics，Part A：Systems and Humans，2010，40(1)：185-197.

[14] Benesty J，Chen Jingdong，Huang Yiteng，*et al*. Pearson correlation coefficient[M]. Berlin：Springer，2009：1-4.

[15] Zhong Kai，Yang Qiqi，Zhu Song. New algebraic conditions for ISS of memristive neural networks with variable delays[J]. Neural Computing & Applications，2017，28(8)：2089-2097.

[16] 王小川. MATLAB 神经网络 43 个案例分析[M]. 北京：北京航空航天大学出版社，2013.

[17] Chen Yishi，Zhao Xing，Jia Xiuping. Spectral-spatial classification of hyperspectral data based on deep belief network[J]. IEEE Journal of Selected Topics in Applied Earth Observations & Remote Sensing，2015，8(6)：2381-2392.

[18] Chen Yushi，Jiang Hanlu，Li Chunyang，*et al*. Deep feature extraction and classification of hyperspectral images based on convolutional neural networks[J]. IEEE Trans on Geoscience & Remote Sensing，2016，54(10)：6232-6251.

[19] Krizhevsky A，Sutskever I，Hinton G E. ImageNet classification with deep convolutional neural networks[C]//Proc of the 25th International Conference on Neural Information Processing Systems. 2012：1097-1105.

[20] Jia P，Zhang M，Yu W，*et al*. Hyperspectral image feature extraction method based on sparse constraint convolutional neural network[Z]. 2017.

[21] Dr Soundarapandian. Chronic_kidney_disease[EB/OL]. [2017-03-06]. https://archive. ics. uci. edu/ml/datasets/Chronic_Kidney_Disease.

[22] Denver C. Parkinsons[EB/OL]. [2017-03]. https://archive. ics. uci. edu/ml/datasets/Parkinsons.

[23] Mota H D. Vertebral + Column[EB/OL]. [2017-03]. https://archive. ics. uci. edu/ml/datasets/Vertebral + Column.

[24] Venkat V，Chan K. A neural network methodology for process fault diagnosis[J]. AiChE Journal，2010，35(12)：1993-2002.

(上接第 2587 页)

[25] Pham Q H，Nguyen M L，Nguyen B T，*et al*. Semi-supervised learning for vietnamese named entity recognition using online conditional random fields[C]//Proc of the 7th ACL-IJCNLP Named Entities Workshop. 2015.

[26] Manamini S A P M，Ahamed A F，Rajapakshe R A E C，*et al*. Ananya：a named-entity-recognition (NER) system for Sinhala language [C]//Proc of Moratuwa Engineering Research Conference. Piscataway，NJ：IEEE Press，2016：30-35.

[27] Nongmeikapam K，Shangkhunem T，Chanu N M，*et al*. CRF based name entity recognition (NER) in Manipuri：a highly agglutinative Indian language[C]//Proc of the 2nd National Conference on Emerging Trends and Applications in Computer Science. Piscataway，NJ：IEEE Press，2011：1-6.

[28] Konkol M，Konopík M. CRF-based czech named entity recognizer and consolidation of czech NER research[C]//Proc of International Conference on Text，Speech，and Dialogue. Berlin：Springer，2013：153-160.

[29] Agerri R，Rigau G. Robust multilingual named entity recognition with shallow semi-supervised features[J]. Artificial Intelligence，2016，238(9)：63-82.

[30] 金明，杨欢欢，单广荣. 藏语命名实体识别研究[J]. 西北民族大学学报：自然科学版，2010，31(3)：49-52.

[31] 李捷译注. 百家姓[M]. 呼和浩特：远方出版社，2007.

[32] Lafferty J D，McCallum A，Pereira F，*et al*. Conditional random fields：probabilistic models for segmenting and labeling sequence data[C]//Proc of the 18th International Conference on Machine Learning. 2001：282-289.