第52卷第5期
2020年10月

南京航空航天大学学报
Journal of Nanjing University of Aeronautics & Astronautics

Vol. 52 No. 5
Oct. 2020

DOI:10.16356/j.1005-2615.2020.05.009

基于函数链神经网络的深度分类器

谢润山[1,2]，王士同[1,2]

（1.江南大学人工智能与计算机学院，无锡，214122；
2.江苏省媒体设计与软件技术重点实验室（江南大学），无锡，214122）

摘要：目前的宽度学习系统（Broad learning system，BLS）通过所建立的一系列映射节点和增强节点来形成联合节点。因为联合节点与输出层的线性连接，网络权值可以用求解伪逆的方法快速求得，避免了耗时的训练过程，从而成为快速而高效的学习方法。然而在追求高精度结果的过程中，BLS对于增强节点数量的需求过于巨大，容易造成过拟合问题。为此，本文提出了基于函数链神经网络（Functional-link neural network，FLNN）的深度分类器（FLNN based deep classifier，FLNNDC），旨在提供一种更加简单却又不失精度的BLS变体结构。FLNNDC将几个轻量级的BLS子系统堆积成栈式结构，每一个轻量级的BLS子系统随机选择一部分映射节点生成增强节点，而不是全部映射节点。和原宽度结构相比，在几个主流数据集上的实验结果表明本文所提出的FLNNDC分类器具有网络结构更小且学习速度更快的优势。

关键词：函数链神经网络；宽度学习；栈式结构；深度学习

中图分类号：TP18　　**文献标志码：**A　　**文章编号：**1005-2615(2020)05-0736-10

Functional-Link Neural Network Based Deep Classifier

XIE Runshan[1,2]，*WANG Shitong*[1,2]

（1. School of Artificial Intelligence and Computer Science，Jiangnan University，Wuxi，214122，China；
2. Jiangsu Key Laboratory of Media Design and Software Technology（Jiangnan University），Wuxi，214122，China）

Abstract: The existing broad learning system（BLS）forms its union nodes by generating a series of mapping nodes and enhancement nodes. Due to the linear connection between the union nodes and the output layer，the weights of the network built by BLS can be obtained quickly by using the corresponding pseudo-inverse computation，thus avoiding the time-consuming training process. As a result，BLS becomes very fast and efficient. However，in order to achieve satisfactory performance，BLS quite often needs too many enhancement nodes，which may cause over-fitting phenomenon. A functional-link neural network（FLNN）based deep classifier，called FLNNDC，is proposed to circumvent the above drawback. FLNNDC stacks several lightweight BLS sub-systems into a stacked structure，while each lightweight BLS sub-system is built by generating enhancement nodes from randomly selected mapped features instead of all the mapped features. Experimental results on several popular datasets demonstrate the effectiveness of FLNNDC in the sense of both downsized structure and less running time.

Key words: functional-link neural network；broad learning；stacked structure；deep learning

近年来，深度学习取得了实质性的突破并在多个领域得到重要应用，特别是在计算机视觉和自然语言处理方面都取得了巨大的成功。在深度学习中，最为主流的结构是深度置信神经网络[1]、深度

基金项目：国家自然科学基金（61572236）资助项目。
收稿日期：2019-08-29；**修订日期：**2020-03-20
通信作者：王士同，男，教授，E-mail：wxwangst@aliyun.com。

引用格式：谢润山，王士同. 基于函数链神经网络的深度分类器[J]. 南京航空航天大学学报，2020，52(5)：736-745. XIE Runshan，WANG Shitong. Functional-link neural network based deep classifier[J]. Journal of Nanjing University of Aeronautics & Astronautics，2020，52(5)：736-745.

玻尔兹曼机[2]和卷积神经网络[3]。尽管深度结构展现了非凡的力量，可是几乎所有的深度结构都包含大量的参数和复杂的结构而需要非常耗时的训练过程，同时，基于反向传播和梯度下降的训练方法容易遭受收敛缓慢和陷于局部最优的问题。为了使深度结构获得更高的精确度，大部分的工作都集中在调整超参数和进一步加深结构这两方面上，尽管已经提出了许多诸如dropout[4]的优化理论，但并未从根本解决深度结构如“黑箱”的现状，这使得理论上分析深度结构变得十分困难。随机向量函数链神经网络[5-6](Random vector functional-link neural network, RVFLNN)拥有能快速求解网络训练后权值的优点，它的出现为解决传统深度结构中缓慢的训练过程问题提供了一种新的思路。

随机向量函数链神经网络有效地消除了深度结构中耗时的训练过程的缺点[7]，RVFLNN对于在紧凑集上的连续函数拥有快速的学习能力，其通用逼近能力也得到了理论证明。因此，RVFLNN已经在多个领域得到了应用，包括上下文建模和控制[8]。RVFLNN将原始数据直接进行非线性转换得到增强节点，随后原始数据和增强节点组成联合节点，联合节点和输出层建立线性连接，输出层的权值可以通过求解联合节点的伪逆方法快速得到，从而避免了耗时的训练过程。图1给出了RVFLNN的结构图。尽管RVFLNN极大增强了感知器的能力，但它在面对现当今时代出现的高维度数据上的表现却并不理想[9]，原因在于RVFLNN直接使用原始输入数据建立增强节点，数据的高维度降低了RVFLNN的学习能力。宽度学习系统[10](Broad learning system, BLS)选择先将原始输入进行特征映射，再将映射后的特征进行增强，这进一步改进了RVFLNN对高维度数据的适应能力。宽度学习系统的理念基于RVFLNN。不同的是BLS将原始输入传入映射节点中进行特征映射，这个过程可以从数据中抽离出良好的特征，为下一步生成增强节点做好准备。增强节点将映射后的特征进行非线性转换，这一过程将产生更多新的非线性特征，最后映射节点和增强节点组合成联合节点。由于联合节点和输出层建立了线性连接，通过求联合节点伪逆的方法可以快速求出网络训练后的权值，从而摆脱了耗时的训练过程。可以看出BLS于RVFLNN最大的改进在于BLS使用映射节点来对原始输入数据进行有效的特征提取，从而解决了在遇到高维输入数据时RVFLNN表现能力降低的问题。BLS的通用逼近能力已经得到了理论证明[11]，并在多个主流数据集上都表现出比深度结构更高的学习精确度和泛化能力。BLS也发展出了多个变体结构，比如fuzzy BLS[12]就是基于Takagi-Sugeno-Kang模糊系统[13-15]发展而来的新结构。但在追求高精度的学习结果过程中，BLS对于增强节点数量的需求过于巨大，这容易造成过拟合问题，也对程序运行设备造成了一定的负担。为了解决这个问题，基于FLNN的深度分类器(FLNN based deep classifier, FLNNDC)选择将几个轻量级的BLS子系统堆积成栈式结构，以期减少BLS的增强节点数量，同时保留其优良的学习能力。

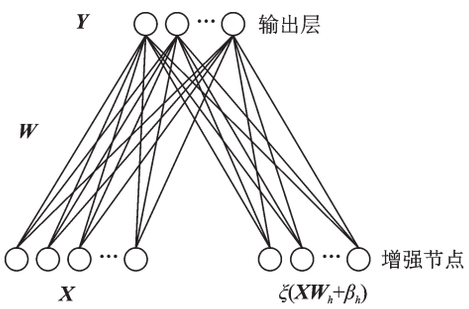

图1　随机向量函数链神经网络

Fig.1　Link neural network of random vector functional

基于FLNN的深度分类器将轻量级的BLS子系统堆积成栈式结构。这里假定每一个轻量级的BLS子系统拥有比传统的BLS少得多的映射节点和增强节点。虽然减少了隐藏节点的数量，然而按照栈式泛化原理[16]，由轻量级的BLS子系统构成的深度分类器FLNNDC的性能将随着子系统栈式堆积而不断得到提高，因而FLNNDC最终能达到令人满意的表现。具体来说：每一层的轻量级BLS子系统的学习目标设置为下层子系统学习后的“残差”，每层子系统的学习结果作为输入数据的扩维部分传入上层网络。这样，每层子系统依旧拥有BLS优良的学习能力，但网络规模却小了很多。上层子系统以下层子系统的“残差”进行学习，可以学习到下层子系统没有学习到的内容，而将下层子系统的学习结果传入上层子系统可以增强上层子系统的学习能力，这样就可以进一步减少上层子系统的增强节点数量，最终实现通过多个轻量级BLS子系统的栈式堆积减少整体结构，增强节点数量的目的。FLNNDC最终的学习结果为每一个子系统的学习结果的累加。此外，本文将原本BLS中增强节点对映射节点的“整体映射”改为随机选择映射节点的“随机映射”，以期望得到更加优秀的映射节点和增强节点。和BLS以及当前的一些主流算法相比，FLNNDC在几个主流数据集上都取得了令人满意的结果。

1 宽度学习系统

首先给出BLS结构的构建过程，然后给出BLS使用的两个优化方法：岭回归学习算法[17]和稀疏自动编码器[18]，两者都是BLS优良学习能力的重要保证。

1.1 宽度学习模型

BLS的基本结构建立在传统的RVFLNN之上，但是不同于RVFLNN直接使用原始输入数据建立增强节点，BLS首先将输入映射成一系列映射节点，再用映射节点建立增强节点，映射节点和增强节点组成联合节点，最后联合节点和输出层建立线性连接。

假定输入数据$\boldsymbol{X}$包含N个样本，每个样本的特征维度为D，即$\boldsymbol{X}\in\boldsymbol{R}^{N\times D}$。用$\boldsymbol{Y}$表示对应的目标矩阵，类别数为$C$，即$\boldsymbol{Y}\in\boldsymbol{R}^{N\times C}$。假设每组映射节点包含$k$个子节点，则第$i$组映射节点可以表示为

$$\boldsymbol{M}_i=\phi_i(\boldsymbol{X}\boldsymbol{W}_{ei}+\boldsymbol{\beta}_{ei})\quad i=1,\cdots,n \tag{1}$$

式中：ϕ_i为映射函数；$\boldsymbol{W}_{ei}$和$\boldsymbol{\beta}_{ei}$为特定分布上随机产生的权值矩阵和偏置矩阵，即$\boldsymbol{W}_{ei}\in\boldsymbol{R}^{M\times k}$和$\boldsymbol{\beta}_{ei}\in\boldsymbol{R}^{N\times k}$，则$n$组映射节点可以表示为$\boldsymbol{M}^n\equiv[\boldsymbol{M}_1,\boldsymbol{M}_2,\cdots,\boldsymbol{M}_n]$，即$\boldsymbol{M}^n\in\boldsymbol{R}^{N\times(k\times n)}$。类似地，假设每组增强节点包含$r$个子节点，则第$j$组增强节点可以表示为

$$\boldsymbol{E}_j=\xi_j(\boldsymbol{M}^n\boldsymbol{W}_{hj}+\boldsymbol{\beta}_{hj})\quad j=1,\cdots,m \tag{2}$$

式中：ξ_j为非线性激活函数；$\boldsymbol{W}_{hj}$和$\boldsymbol{\beta}_{hj}$为特定分布上随机产生的权值矩阵和偏置矩阵，即$\boldsymbol{W}_{hj}\in\boldsymbol{R}^{(k\times n)\times r}$和$\boldsymbol{\beta}_{hj}\in\boldsymbol{R}^{N\times r}$。则$m$组增强节点可以表示为$\boldsymbol{E}^m\equiv[\boldsymbol{E}_1,\boldsymbol{E}_2,\cdots,\boldsymbol{E}_m]$，即$\boldsymbol{E}^m\in\boldsymbol{R}^{N\times(r\times m)}$。相应地，联合节点表示为

$$\boldsymbol{U}\equiv[\boldsymbol{M}^n|\boldsymbol{E}^m] \tag{3}$$

由于联合节点和输出层建立了线性连接，因此整个宽度模型可以用下面的公式表达

$$\begin{aligned}\boldsymbol{Y}&=[\boldsymbol{M}_1,\cdots,\boldsymbol{M}_n|\boldsymbol{E}_1,\cdots,\boldsymbol{E}_m]\boldsymbol{W}=\\&[\boldsymbol{M}^n|\boldsymbol{E}^m]\boldsymbol{W}=\boldsymbol{U}\boldsymbol{W}\end{aligned} \tag{4}$$

式中：$\boldsymbol{W}$为连接联合节点和输出层的权值，可以求解伪逆的方式快速得到

$$\boldsymbol{W}=\boldsymbol{U}^{\dagger}\boldsymbol{Y} \tag{5}$$

式中：$\boldsymbol{U}^{\dagger}$为$\boldsymbol{U}$的伪逆。图2展示了经典的宽度学习结构。

1.2 岭回归学习算法

在宽度网络中，求解伪逆是一种求得网络输出层权值非常方便的方法。有很多不同的方法可以用来求解伪逆，比如正交投影法、奇异值分解和正交化法等[19]，但考虑到联合节点的大小，直接的求解方法开销过大，特别是面对大量的训练样本时[9]，因此岭回归算法成为求解伪逆$\boldsymbol{U}^{\dagger}$较为理想的方法，公式表达为

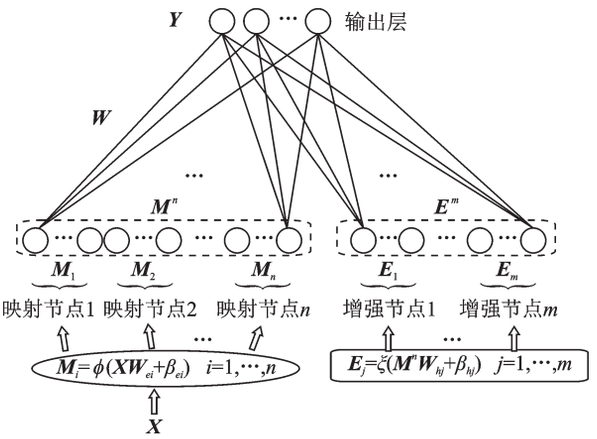

图2 经典的宽度学习结构

Fig.2 Framework of a typical BLS

$$\boldsymbol{U}^{\dagger}=\begin{cases}(\lambda\boldsymbol{I}+\boldsymbol{U}^{\mathrm{T}}\boldsymbol{U})^{-1}\boldsymbol{U}^{\mathrm{T}} & N\geqslant h\\ \boldsymbol{U}^{\mathrm{T}}(\lambda\boldsymbol{I}+\boldsymbol{U}\boldsymbol{U}^{\mathrm{T}})^{-1} & \text{其他}\end{cases} \tag{6}$$

式中：$h=k\times n+r\times m$，即隐节点数量；$\boldsymbol{I}$为单位矩阵；λ为正则化项。

1.3 稀疏自动编码器

监督学习中，原始输入数据的有效特征是获得令人满意的学习表现的重要保证。特别是稀疏化的特征可以保留关键特征而把冗余特征去除，从而达到特征选择和降低模型的复杂度[20]的目的。为了获得稀疏化特征，需要对生成映射节点的随机权值$\boldsymbol{W}_{ei}$进行合适的微调，用最优化公式表达为[21]

$$\arg\min_{\boldsymbol{W}}:\|\mathbf{M}\mathbf{W}-\boldsymbol{X}\|_2^2+\lambda\|\boldsymbol{W}\|_1 \tag{7}$$

式中：$\boldsymbol{W}$为稀疏自动编码器得到的解，即微调后的权值；$\boldsymbol{X}$为输入数据；$\boldsymbol{M}$为希望获得的稀疏化特征，满足$\boldsymbol{X}\boldsymbol{W}=\boldsymbol{M}$；$\lambda$为正则项系数。上述最优化问题可以用交替方向乘子法[22-23]求解，具体步骤如下

$$\begin{cases}\boldsymbol{W}_{k+1}=(\boldsymbol{M}^{\mathrm{T}}\boldsymbol{M}+r\boldsymbol{I})(\boldsymbol{M}^{\mathrm{T}}\boldsymbol{X}+\rho(\boldsymbol{o}_k-\boldsymbol{u}_k))\\ \boldsymbol{o}_{k+1}=S_{\lambda/r}(\boldsymbol{W}_{k+1}+\boldsymbol{u}_k)\\ \boldsymbol{u}_{k+1}=\boldsymbol{u}_k+(\boldsymbol{W}_{k+1}-\boldsymbol{o}_{k+1})\end{cases} \tag{8}$$

式中：初始值$\boldsymbol{W}_1$，$\boldsymbol{u}_1$和$\boldsymbol{o}_1$都为维度合适的零矩阵；$r>0$；S为软阈值操作，定义如下

$$S_k(a)=\begin{cases}a-k & a>k\\ 0 & |a|\leqslant k\\ a+k & a<-k\end{cases} \tag{9}$$

2 基于FLNN的深度分类器

FLNNDC的提出希望达到在保持BLS优良学习能力的前提下减少BLS增强节点数量的目的，为此，FLNNDC选择利用几个轻量级BLS子

系统堆积成为栈式结构，上层子系统对下层子系统的“残差”进行学习，随着子系统个数的增加以达到“残差逼近”的效果。为了增强上层子系统的学习能力，FLNNDC将上层的输入数据进行扩维，即包含下层子系统的学习结果，学习能力的增加意味着对增强节点的需求将减少。为了组合出更为优秀的映射节点组，将BLS的“整体增强”改为“随机增强”，以期得到表达能力更强的增强节点。这样在总体表达能力相同的前提下，所需要的增强节点数量将会减少。

为了不失一般性，假定FLNNDC的层数为L，图3展示了其结构，具体给出了第t层子系统的模型。依据图3，第t层子系统的构建过程可以清晰地阐述如下。

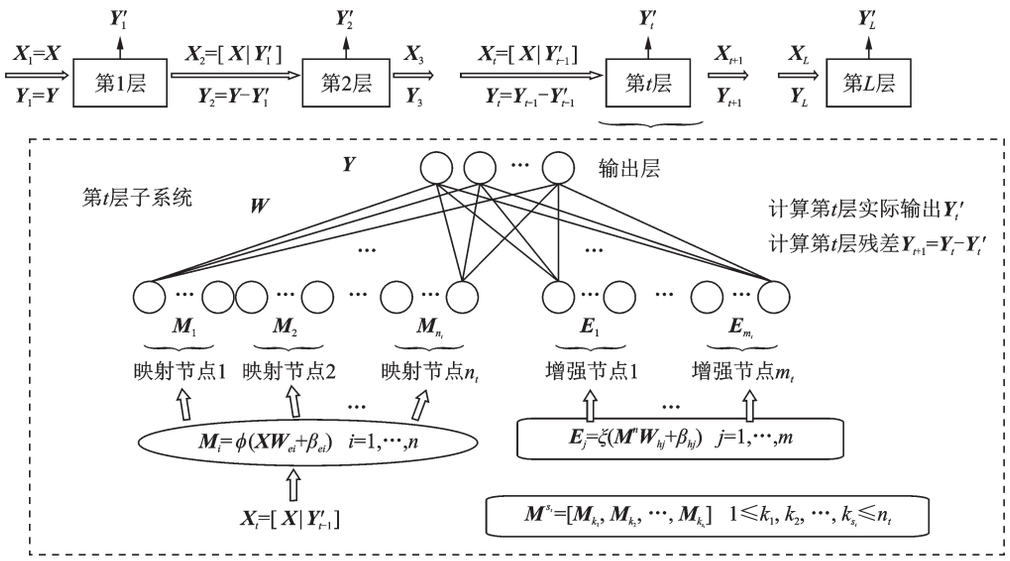

图3　基于FLNN的深度分类器的第t层结构($t=1,2,\cdots,L$)

Fig.3　The tth ($t=1,2,\cdots,L$) layer structure of FLNN based deep classifier

第t层子系统包含的参数包括：输入数据$\boldsymbol{X}_t$以及对应的目标矩阵$\boldsymbol{Y}_t$，该子系统包含n_t组映射节点，其中每组映射节点包含k_t个子节点，m_t组增强节点，其中每组增强节点包含r_t个子节点。相对于传统的BLS，所提出的FLNNDC的轻量级子系统将包含少得多的映射节点和增强节点，即满足$n_t\ll n$和$m_t\ll m$。除第1个子系统外，其余子系统的输入数据都包含上1个子系统的实际输出，即

$$\boldsymbol{X}_t=\begin{cases}\boldsymbol{X} & t=1\\ [\boldsymbol{X}|\boldsymbol{Y}_{t-1}'] & t=2,3,\cdots,L\end{cases}\tag{10}$$

同时目标矩阵相应地设置为

$$\boldsymbol{Y}_t=\begin{cases}\boldsymbol{Y} & t=1\\ \boldsymbol{Y}_{t-1}-\boldsymbol{Y}_{t-1}' & t=2,3,\cdots,L\end{cases}\tag{11}$$

式中：$\boldsymbol{Y}_{t-1}$和$\boldsymbol{Y}_{t-1}'$分别为第$t-1$层子系统的目标矩阵和实际输出。依据输入数据$\boldsymbol{X}_t$，首先建立映射节点组$\boldsymbol{M}^{n_t}\equiv[\boldsymbol{M}_1,\boldsymbol{M}_2,\cdots,\boldsymbol{M}_{n_t}]$，其中第$i$组映射节点表示为

$$\boldsymbol{M}_i=\phi_i(\boldsymbol{X}_t\boldsymbol{W}_{ei}+\boldsymbol{\beta}_{ei})\tag{12}$$

式中：权值矩阵$\boldsymbol{W}_{ei}$和偏置矩阵$\boldsymbol{\beta}_{ei}$随机产生于特定分布。不同于BLS的“整体映射”，FLNNDC使用“随机增强”来生成增强节点，即从$\boldsymbol{M}^{n_t}$中随机选择s_t组映射节点建立增强节点，其中$s_t<n_t$，这s_t组映射节点组成临时映射节点组$\boldsymbol{M}^{s_t}$。请注意，使用$\boldsymbol{M}^{s_t}$生成增强节点时，每一组增强节点都需要随机都生成新的$\boldsymbol{M}^{s_t}$。即，从映射节点组中可重复的随机选择s_t组映射节点，组成临时的映射节点

$$\boldsymbol{M}^{s_t}\equiv[\boldsymbol{M}_{k_1},\boldsymbol{M}_{k_2},\cdots,\boldsymbol{M}_{k_{s_t}}]\tag{13}$$

式中：$1\leqslant k_1,k_2,\cdots,k_{s_t}\leqslant n_t$，此时，第$j$个增强节点可以表示为

$$\boldsymbol{E}_j=\xi_j(\boldsymbol{M}^{s_t}\boldsymbol{W}_{hj}+\boldsymbol{\beta}_{hj})\tag{14}$$

式中：权值矩阵$\boldsymbol{W}_{hj}$和偏置矩阵$\boldsymbol{\beta}_{hj}$都随机产生于特定分布。重复前面步骤直到生成全部的m_t组增强节点，然后建立增强节点组$\boldsymbol{E}^{m_t}=[\boldsymbol{E}_1,\boldsymbol{E}_2,\cdots,\boldsymbol{E}_{m_t}]$，建立联合节点$\boldsymbol{U}_t=[\boldsymbol{M}^{n_t}|\boldsymbol{E}^{m_t}]$，用岭回归算法计算联合节点的伪逆$\boldsymbol{U}_t^{\dagger}$，从而得到输出层权值$\boldsymbol{W}_t=\boldsymbol{U}_t^{\dagger}\boldsymbol{Y}_t$，最后计算该子系统的实际输出

$$\boldsymbol{Y}_t'=\boldsymbol{U}_t\boldsymbol{W}_t\tag{15}$$

若$t\neq L$，还需计算第t层子系统“残差”

$$\boldsymbol{Y}_{t+1}=\boldsymbol{Y}_t-\boldsymbol{Y}_t'\tag{16}$$

式中：将$\boldsymbol{Y}_{t+1}$设置为第$t+1$层子系统的目标矩阵。至此，第t层子系统构建完成。

重复以上的子系统构建过程直到构建完所有L个子系统，此时FLNNDC的整体结构也构建完

成。FLNNDC整体结构构建完成后，网络中随机生成的$\boldsymbol{W}_{ei}$（微调后），$\boldsymbol{\beta}_{ei}$，$\boldsymbol{W}_{hj}$，$\boldsymbol{\beta}_{hj}$和“随机增强”中选择的映射节点下标都不再发生变化。进行测试时，每1层子系统都会产生1个预测结果，如第t层子系统会产生预测结果$\boldsymbol{P}_t$，第2层及以上子系统的输入数据也需要进行相应的扩维处理，如将第t层子系统的输入数据扩维为$[\boldsymbol{T}|\boldsymbol{P}_{t-1}]$。最终整个结构的预测结果$\boldsymbol{P}$同样为每一层子系统预测结果的累加，即

$$\boldsymbol{P}=\boldsymbol{P}_1+\boldsymbol{P}_2+\cdots+\boldsymbol{P}_N \tag{17}$$

从式(17)可以看出FLNNDC包含了“残差逼近”的思想，算法1表达了L层FLNNDC的构建过程。

算法1 FLNNDC的学习算法

输入：原始样本及标签$\{\boldsymbol{X},\boldsymbol{Y}\}$

输出：$\boldsymbol{W}_t$，$\boldsymbol{Y}_t'$和$\boldsymbol{Y}'$

(1) for $t=1$; $t\leqslant L$ do

(2) 计算 $\boldsymbol{X}_t=\begin{cases}\boldsymbol{X} & t=1\\ [\boldsymbol{X}|\boldsymbol{Y}_{t-1}] & \text{其他}\end{cases}$

(3) 计算 $\boldsymbol{Y}_t=\begin{cases}\boldsymbol{Y} & t=1\\ \boldsymbol{Y}_{t-1}-\boldsymbol{Y}_{t-1}' & \text{其他}\end{cases}$

(4) for $i=0;i<n_t$ do

(5) 随机生成$\boldsymbol{W}_{ei}$和$\boldsymbol{\beta}_{ei}$;

(6) 计算$\boldsymbol{M}_i=[\phi(\boldsymbol{X}_t\boldsymbol{W}_{ei}+\boldsymbol{\beta}_{ei})]$;

(7) 用稀疏编码器微调$\boldsymbol{W}_{ei}$;

(8) end

(9) 设置$\boldsymbol{M}^{n_t}\equiv[\boldsymbol{M}_1,\boldsymbol{M}_2,\cdots,\boldsymbol{M}_{n_t}]$;

(10) for $j=0;j<m_t$ do

(11) 依据式(13)设置$\boldsymbol{M}^{s_t}$;

(12) 随机生成$\boldsymbol{W}_{hj}$和$\boldsymbol{\beta}_{hj}$;

(13) 计算$\boldsymbol{E}_j=\xi(\boldsymbol{M}^{s_t}\boldsymbol{W}_{hj}+\boldsymbol{\beta}_{hj})$;

(14) end

(15) 设置$\boldsymbol{E}^{m_t}=[\boldsymbol{E}_1,\boldsymbol{E}_2,\cdots,\boldsymbol{E}_{m_t}]$;

(16) 设置$\boldsymbol{U}_t=[\boldsymbol{M}^{n_t}|\boldsymbol{E}^{m_t}]$;

(17) 依据岭回归计算$\boldsymbol{U}_t^{\dagger}$;

(18) 计算输出层权值$\boldsymbol{W}_t=\boldsymbol{U}_t^{\dagger}\boldsymbol{Y}_t$;

(19) 计算实际输出$\boldsymbol{Y}_t'=\boldsymbol{U}_t\boldsymbol{W}_t$;

(20) end

BLS单一的宽度结构使得其在面对复杂数据集时需要大量的增强节点，这无疑增大了产生过拟合的风险。FLNNDC作为BLS的变体结构，不仅继承了BLS快速学习的优点，同时减少了BLS的增强节点数量，简化了其结构。FLNNDC的创新点可以总结为以下3点。

(1) FLNNDC在BLS基础上建立了深度结构。FLNNDC的每一层结构都是1个轻量级的BLS子系统，都拥有BLS优良的学习能力。每一层子系统完成学习后会计算相应“残差”，即没有学习到内容，除第1层对原始样本进行学习外，每一层子系统仅对下层子系统的“残差”进行学习，这里包含一种“残差逼近”的思想，每一层子系统并不需要学习到目标的全部的内容，后续子系统会不断对前面没有学习到的内容进行学习，因此每一层子系统都不需要很多的增强节点，从而实现轻量级。对于高精度结果的追求方法上，FLNNDC和BLS有着明显的差异，BLS依靠大量的增强节点来提高精度，这无疑会增加过拟合风险和存储需求，而FLNNDC的选择是增加子系统个数，轻量级的子系统降低了过拟合的风险。

(2) FLNNDC对原始输入数据进行了扩维处理。与BLS只使用原始数据进行特征映射不同，为了加强层间联系，FLNNDC将下层的学习结果加入到原始数据中作为上层的输入数据，所以上层的输入数据包含了下层的信息。使用扩维后的输入数据进行特征映射能生成表现能力更强的映射节点，继而生成更为优秀的增强节点，这使得上层的子系统拥有比下层子系统更加强的学习能力，所以需要的增强节点比下层更少，这个现象在子系统数增加时表现得更加明显。所以在整体结构表现能力相同的前提下，因为得到了表现能力更强的增强节点，所以FLNNDC需要的增强节点数将会比BLS更少。

(3) FLNNDC将BLS生成增强节点的“整体增强”改为采用随机抽取方式的“随机增强”。这一改变基于以下的考量：①不断进行栈式堆积的BLS子系统中的增强节点由于包含了下一层的信息，因而本身的表达能力变得更强，故不必使用“整体增强”，亦即使用随机增强也可以达到满意的甚至更好的性能。②按照文献[11]，理论上每个轻量级BLS的通用逼近性能依赖于足够多的映射节点和增强节点。因此随机增加节点的快速增量算法由于提供了更多的增强节点而提高每个轻量级BLS的通用逼近能力。③依据栈式泛化原理[16]，随机增强后FLNNDC的逼近性能也将随着栈式堆积而不断改善。本文的实验结果也表明上述考量以及“随机增强”是有效的。

上面的3点改进相辅相成，使FLNNDC在不失精度的前提下减少了大量的增强节点的数量，并加快了训练速度。为了进一步说明FLNNDC的精简的结构和较短的训练时间的优越性，第4部分的对比实验结果将给出实验数据方面的证明。

3 实验和讨论

为了和BLS进行公平和充分的比较，参考提

出BLS的2篇文献[10-11]设计对比实验，即在文献中选择的3个数据集上进行实验，对比算法的选择和参数配置也和文献中保持一致。此外还选择3个UCI[24]数据集进行实验。为了证明FLNNDC在相同节点数量和相同数据集的前提下在运行时间上具有优势，最后还给出了BLS和FLNNDC在相同条件下运行时间的对比实验。实验环境为Intel-i3 3.40 GHz CPU，实验语言为Python。

选择在3个主流数据集：MNIST[25]，NORB[26]和Extended YaleB[27]上和其他当前的主流算法进行对比实验，实验结果均为10次取平均值，在经典数据集MNIST上还给出5层的FLNNDC每一层对总精度的影响。实验中的FLNNDC参数设置如下：$\boldsymbol{W}_{ei}$，$\boldsymbol{\beta}_{ei}$，$\boldsymbol{W}_{hj}$和$\boldsymbol{\beta}_{hj}$均取自标准正态分布，微调$\boldsymbol{W}_{ei}$的循环次数为50次，求解伪逆的岭回归中正则化参数λ设置为10^{-8}，增强节点激活函数使用tansig函数。

进行比较的其他主流算法包括：BLS、栈式自动编码器[28](Stacked autoencoder, SAE)、栈式降噪自动编码器[29](Stacked denoising autoencoder, SDA)、深度置信网络[1](Deep belief net, DBN)、基于多层感知器方法[30](Multilayer perception-based method, MLP)、极限学习机[31](Extremely learning machine, ELM)、两种基于多层感知器的极限学习机MLELM[32]和HELM[33]、扩展的模糊受限波兹曼机[34](Extended fuzzy restricted Boltzmann machine, FRBM)、支持向量机[35](Support vector machine, SVM)、最小二乘支持向量机[36](Least square support vector machine, LSSVM)和最早提出的深度卷积神经网络[37](Convolutional neural network, CNN) LeNet5[38]。

依据文献[10]给出进行对比的主流算法的参数设置，总的来说，以上方法除了BLS，HELM和MLELM都是深度结构，这些深度结构依据初始学习率0.1和每代衰减率0.95的反向传播来调整超参数。MLELM的正则化系数设置为10^{-1}，10^{3}和10^{8}，HELM的惩罚系数设置为10^{8}，MLELM和HELM的其他参数细节参考文献[34]。BLS随机权值和偏置取自[−1，1]的标准均匀分布，微调$\boldsymbol{W}_{ei}$的循环次数为50次，求解伪逆的岭回归中正则化参数λ设置为10^{-8}，增强节点激活函数使用Tansig函数。不同数据集上的具体参数设置在下面相应部分中给出。

3.1　MNIST数据集

MNIST手写数字图像是一个经典的数据集，很多主流算法都会选择在这个数据集上进行测试。MNIST包含了60 000个训练样本和10 000个测试样本，样本为0到9的10类数字，图像均为28×28的灰度图，图4展示了其中的部分图像。

图4　MNIST数据集

Fig.4　MNIST dataset

为了测试FLNNDC的精确度，使用先验知识给出每层映射节点数量、增强节点数量、选择映射节点进行增强数量以及所需层数。在深度结构中，确定网络结构大小也是最具挑战性的任务。在实验数据集MNIST中，FLNNDC的具体结构为：选用5层结构，第1层子系统选择20×8的映射节点和2 100×1的增强节点，“随机增强”选用2组映射节点；第2层子系统选择15×8的映射节点和1 995×1的增强节点，“随机增强”选用1组映射节；第3层子系统选择16×8的映射节点和1 785×1的增强节点，“随机增强”选用2组映射节点；第4层子系统选择15×8的映射节点和1 680×1的增强节点，“随机增强”选用2组映射节点；第5层子系统选择11×8的映射节点和1 575×1的增强节点，“随机增强”选用1组映射节点合计616个映射节点和9 135个增强节点。

依据文献[10]给出其他主流算法的设置：BLS使用10×10的映射节点和11 000×1的增强节点、SAE使用1 000-500-25-30的结构设置、DBN使用500-500-2 000的结构设置、DBM使用500-500-1 000的结构设置、MLELM使用700-700-15 000的结构设置以及HELM使用300-300-12 000的结构设置。前面所提及方法的测试精确度在表1给出。

表1　在数据集MNIST上的分类精确度

Table 1　Classification accuracy on MNIST dataset　%

方法	精确度
SAE	98.60
SDA	98.72
DBN	98.87
DBM	99.05
MLP	97.39
MLELM	99.04
HELM	99.13
CNN	95.63
FRBM	97.44
BLS	98.72
FLNNDC	98.72

从表1中可以看出在，FLNNDC达到了和BLS相同的98.72%精确度，相对于需要11 000个增强节点的BLS，FLNNDC只需要9 135个增强节点，且每一层平均只需要1 827个增强节点，减少了相当多的增强节点数量。为了表明参数合适的每一层都能提高FLNNDC总精度，图5给出了FLNNDC精确度随子系统数量变化趋势图。

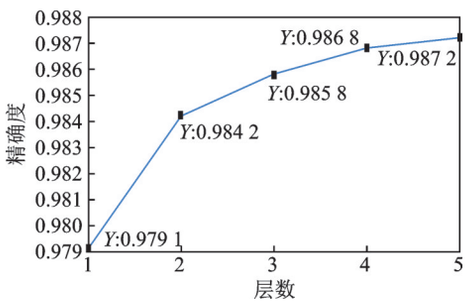

图5 在MNIST数据集上FLNNDC的精确度随层数变化图

Fig.5 Accuracies of FLNNDC's on the MNIST dataset with the increasing number of layer

3.2 NORB数据集

相比于MNIST数据集，NORB数据集是一个更加复杂的数据集，一共包含48 600张大小为2×32×32的灰度图，图片内容包含50种3D玩具物体，分别属于5大类：动物、人类、飞机、卡车和小汽车，图片中物体拍摄于不同光照条件、高度和方位，图6展示了其中的部分图像。将数据集分为24 300张包含25种物体（每类5种）的训练集和24 300张包含25种物体（每类5种）的测试集。

图6 NORB数据集

Fig.6 NORB dataset

在数据集NORB中，FLNNDC的具体结构为：选用3层结构，第1层子系统选择30×8的映射节点和900×1的增强节点，“随机增强”选用1组映射节点；第2层子系统选择22×8的映射节点和855×1的增强节点，“随机增强”选用1组映射节点；第3层子系统选择20×8的映射节点和765×1的增强节点，“随机增强”选用1组映射节点，合计576个映射节点和2 520个增强节点。

依据文献[10]给出其他主流算法的设置：BLS使用100×10映射节点加上9 000×1增强节点、DBN使用4 000-4 000-4 000结构以及HELM使用3 000-3 000-15 000结构，未给出设置的算法和在MNIST实验中使用相同参数。前面所提及方法的测试精确度在表2给出。从表2中可以看出相比于BLS的89.06%，FLNNDC达到了更高的89.69%的精确度，且一共只使用了2 520个增强节点，平均每层使用了840个增强节点，与BLS的9 000个增强节点相比大幅度减少。在和其他主流算法比较中精确度也排到了第2名，仅次于HELM的91.28%，这些实验数据有力地说明了FLNNDC的有效性。

表2 在数据集NORB上的分类精确度

Table 2 Classification accuracy on NORB dataset %

方法	精确度
SAE	86.28
SDA	87.62
DBN	88.47
DBM	89.65
MLP	84.20
MLELM	88.91
HELM	91.28
BLS	89.06
FLNNDC	89.69

3.3 Extended YaleB数据集

Extended YaleB数据集也称为扩展的耶鲁人脸数据集，由耶鲁大学制作，一共包含属于38个人的大小为32×32的2 414张剪裁过的人脸图片，图片在光照条件和面部表情上都有较大的变化，图7展示了部分图片。每个人随机选30张图片用于训练其余用于测试，即用1 140张图片训练，1 274张图片测试。

图7 Extended YaleB数据集

Fig.7 Extended YaleB dataset

在Extended YaleB数据集上，FLNNDC的具体结构为：选用4层结构，第1层子系统选择35×12的映射节点和60×1的增强节点，“随机增强”选用1组映射节点；第2层子系统选择33×12的映射节点和54×1的增强节点，“随机增强”选用1组映

射节点；第 3 层子系统选择 31×12 的映射节点和 48×1 的增强节点，“随机增强”选用 1 组映射节点；第 4 层子系统选择 30×12 的映射节点和 51×1 的增强节点，“随机增强”选用 1 组映射节点。合计 1 548 个映射节点和 213 个增强节点。

依据文献[11]给出其他参加比较的算法的具体参数：SVM 和 ELM 的参数(C,γ)使用网格搜索的方法在$\{2^{-24},2^{-23},\cdots,2^{24},2^{25}\}$区间内确定最优数值，结果为 SVM 和 ELM 的参数(C,γ)分别取$(2^{10},2^{-10})$和$(2^{11},2^{11})$；LSSVM 的最优参数(C,γ)使用 MATLAB 的工具箱 LS-SVMlab[39]自动确定，寻优结果为(4.086 4，1 433.838 4)；BLS 使用 60×30 映射节点和 3 000×1 增强节点。实验结果在表 3 给出。

表 3　在数据集 Extended YaleB 上的分类精确度

Table 3　Classification accuracy on extended YaleB dataset　%

方法	精确度
SVM	89.48
LSSVM	89.95
ELM	96.94
BLS	97.88
FLNNDC	98.04

从表 3 中可以看出相比于 BLS 的 97.88%，FLNNDC 达到了更高的 98.04% 的精确度，而且只使用了 213 个增强节点，相比于 BLS 的 3 000 个增强节点，减少幅度是巨大的，在和其他主流算法的精确度比较中也取得了最好成绩，这些数据进一步佐证了 FLNNDC 结构的优越性。

3.4　UCI 数据集

UCI 数据集是加州大学欧文分校提出的用于机器学习的数据库，是一个标准测试数据集。此处选用 Pen-Based Recognition of Handwritten Digits，Optical Recognition of Handwritten Digits 和 Page Blocks 数据集，这 3 个数据集的细节在表 4 给出。对 BLS 的最优参数进行网格搜索，具体如下：对映射节点组包含的节点数、映射节点组数量和增强节点组的数量在区间[1,30]×[1,30]×[1,3 000]进行搜索，对比算法 SVM 和 LSSVM 的最优参数确定方法和前面实验保持一致。

对于数据集 Pen-Based Recognition of Handwritten Digits，SVM 的参数(C,γ)取$(2^{2},2^{-14})$，LSSVM 的参数(C,γ)取(1.970 5，1.279 4)，BLS 使用 13×18 映射节点和 2 791×1 增强节点。FLNNDC 的具体结构为：选用 2 层结构，第 1 层子系统选择 19×22 的映射节点和 1 136×1 的增强节点，“随机增强”选用 5 组映射节点；第 2 层子系统选择 18×22 的映射节点和 1 022×1 的增强节点，“随机增强”选用 5 组映射节点。合计 2 158 个增强节点。

表 4　3 个 UCI 数据集详细信息

Table 4　Details of three UCI datasets

数据集	样本数量		特征数量	类别
	训练集	测试集		
Pen-based	7 494	3 498	16	10
Optical	3 823	1 797	64	10
Page	3 831	1 642	10	5

对于数据集 Optical Recognition of Handwritten Digits，SVM 的参数(C,γ)取$(2^{3},2^{-10})$，LSSVM 的参数(C,γ)取(174.183 7，152.909 6)，BLS 使用 8×16 映射节点和 2 717×1 增强节点。FLNNDC 的具体结构为：选用 2 层结构，第 1 层子系统选择 17×14 的映射节点和 1 097×1 的增强节点，“随机增强”选用 2 组映射节点生成增强节点；第 2 层子系统选择 16×14 的映射节点和 987×1 的增强节点，“随机增强”选用 2 组映射节点生成增强节点。合计 2 084 个增强节点。

对于数据集 Page Blocks，SVM 的参数(C,γ)取$(2^{6},2^{-15})$，LSSVM 的参数(C,γ)取(34.609 3，3 081.060 8)，BLS 使用 8×28 映射节点和 2 354×1 增强节点。FLNNDC 的具体结构为：选用 2 层结构，第 1 层结构选择 6×9 的映射节点和 201×1 的增强节点，“随机增强”选用 4 组映射节点；第 2 层结构选择 5×9 的映射节点和 180×1 的增强节点，“随机增强”选用 4 组映射节点，合计 381 个增强节点。

对比算法在 3 个 UCI 数据集上的精确度在表 5 给出。从表 5 可以看出，相对于 BLS，FLNNDC 在 3 个 UCI 数据集上都用更小的结构达到更高的精确度，与 SVM 和 LSSVM 的比较中也取得了更好的表现，这进一步佐证了 FLNNDC 的优越性。

表 5　在 3 个 UCI 数据集上的分类精确度

Table 5　Classification accuracy on three UCI datasets

方法	数据集/%		
	Pen-based	Optical	Page
SVM	97.48	96.61	95.43
LSSVM	97.80	98.11	95.80
BLS	98.31	98.05	96.38
FLNNDC	98.51	98.44	96.41

3.5　运行时间对比实验

本节对比 BLS 和 FLNNDC 在相同的隐节点数量（映射节点和增强节点）和相同的数据集的前

提下的运行时间。实验数据集选用Extended YaleB数据集。选择4种节点数量进行对比实验，分别为10×30映射节点加上1 800增强节点、10×30映射节点加上2 400增强节点、10×30映射节点加上3 000增强节点和10×30映射节点加上3 600增强节点。FLNNDC使用3层结构，每层结构为相同的映射节点数量和增强节点数量，“随机增强”选用1组映射节点。实验结果如表6所示。

表6 BLS和FLNNDC的运行时间比较

Table 6 Comparision of running time between BLS and FLNNDC s

方法	增强节点数量			
	1 800	2 400	3 000	3 600
BLS	3.538 9	5.270 2	7.985 7	11.861 7
FLNNDC	2.576 5	3.007 9	3.485 8	4.123 3

从表6可以看出，在相同的节点数量和相同数据集的前提下，相比于BLS，FLNNDC所需的运行时间更短，且在增强节点数量不断增加的情况下，FLNNDC运行时间短的优势越明显。

4 结 论

相对于BLS需要大量的增强节点来构建高精度的网络结构，FLNNDC只需要较少的增强节点数量就能达到相同精度。FLNNDC给出了减少增强节点从而精简BLS网络结构的一种思路，结构的精简可以显著减少过拟合的风险，也可以降低对程序运行设备的要求。在3个主流数据集上FLNNDC都得到了令人满意的结果，3个数据集分别代表了数字、3D物体和人脸，表明了FLNNDC对于不同对象都具有很好的分类能力。在3个UCI数据集上的实验结果也证明了FLNNDC具有良好的泛化性。在包含相同的隐节点数量时，相对于BLS，FLNNDC训练节点需要的运行时间短得多，这也是其一大优势，且在包含大量节点的情况下这个优势更加明显。FLNNDC不需要耗时的训练过程，可以通过求得伪逆从而快速得到网络训练后的权值，这是继承了FLNN的优点，深度结构的建立又增强了网络的泛化能力，这正是深度结构的优势所在，两者的结合可谓是相得益彰。为了能加快FLNNDC超参数的确定过程，下一步工作将集中于如何确定网络的最优结构，以及如何理论证明FLNNDC的栈式结构使用“随机增强”机制进行拓展时能够在期望意义上保证达到组合最优。

参考文献：

[1] HINTON G E, OSINDERO S, TEH Y. A fast learning algorithm for deep belief nets[J]. Neural Computation, 2006, 18: 0899.

[2] SALAKHUTDINOV R, HINTON G E. Deep Boltzmann machines[J]. Journal of Machine Learning Research, 2009, 5(2): 1967-2006.

[3] SIMONYAN K, ZISSERMAN A. Very deep convolutional networks for large-scale image recognition[EB/OL].(2014-09-15).https://arxiv.org/abs/1409.1556.

[4] SRIVASTAVA N, HINTON G E, KRIZHEVSKY A. Dropout: A simple way to prevent neural networks from overfitting[J]. Journal of Machine Learning Research, 2014, 15(1): 1929-1958.

[5] PAO Y, TAKEFUJI Y. Functional-link net computing: Theory, system architecture, and functionalities.[J]. Computer, 1992, 25(5): 76-79.

[6] PAO Y, PARK G, SOBAJIC D J. Learning and generalization characteristics of the random vector functional-link net[J]. Neurocomputing, 1994(2): 163-180.

[7] NARENDRA K S, PARTHASARATHY K. Identification and control of dynamical systems using neural network[J]. IEEE Trans on Neural Networks, 1990, 1(1): 4-27.

[8] TYUKIN I, PROKHOROV D. Feasibility of random basis function approximators for modeling and control[C]//Proceedings of IEEE Control Application (CCA) and Intelligent Control (ISIC). [S.l.]: IEEE, 2009: 1391-1396.

[9] CHEN C L P, ZHANG C. Data-intensive applications, challenges, techniques and technologies: A survey on big data[J]. Information Sciences, 2014, 275: 314-347.

[10] CHEN C L P, LIU Z. Broad learning system: An effective and efficient incremental learning system without the need for deep architecture[J]. IEEE Transactions on Neural Networks and Learning Systems, 2018(1): 10-24.

[11] CHEN C L P, LIU Z, FENG S. Universal approximation capability of broad learning system and its structural variations[J]. IEEE Transactions on Neural Networks and Learning Systems, 2019(4): 1191-1204.

[12] FENG S, CHEN C L P. Fuzzy broad learning system: A novel neuro-fuzzy model for regression and classification[J]. IEEE Transactions on Cybernetics, 2020(2): 414-424.

[13] JIANG Y, CHUNG F, ISHIBUCHI H, et al. Multitask TSK fuzzy system modeling by mining intertask common hidden structure[J]. IEEE Transactions on

Cybernetics, 2015(3): 548-561.

[14] GU X, CHUNG F, WANG Shitong. Bayesian takagi-Sugeno-Kang fuzzy classifier[J]. IEEE Transactions on Fuzzy Systems, 2017(6): 1655-1671.

[15] DENG Z, CAO L, JIANG Y, et al. Minimax probability TSK fuzzy system classifier: A more transparent and highly interpretable classification model[J]. IEEE Transactions on Fuzzy Systems, 2015(4): 813-826.

[16] TING K M, WITTEN I H. Stacked generalizations: When does it work?[C]//Proceedings of Fifteenth International Joint Conference on Artificial Intelligence (IJCAI). [S.l]: [s.n.], 1997: 866-871.

[17] HOERL A E, KENNARD R W. Ridge regression: Biased estimation for nonorthogonal problems[J]. Technometrics, 2000(1): 80.

[18] GONG M, LIU J, LI H, et al. A multiobjective sparse feature learning model for deep neural networks [J]. IEEE Transactions on Neural Networks & Learning Systems, 2015(12): 3263-3277.

[19] RAO C R, MITRA S K. Generalized inverse of a matrix and its applications[J]. Operational Research Quarterly, 1972, 1(4): 601-620.

[20] YANG W, GAO Y, SHI Y, et al. MRM-Lasso: A sparse multiview feature selection method via low-rank analysis[J]. IEEE Transactions on Neural Networks & Learning Systems, 2015(11): 2801-2815.

[21] OLSHAUSEN B A, FIELD D J. Sparse coding with an overcomplete basis set: A strategy employed by V1?[J]. Vision Research, 1997(23): 3311-3325.

[22] BOYD S, PARIKH N, CHU E, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers[J]. Foundation and Trends in Machine Learing, 2011, 3(1): 1-122.

[23] GOLDSTEIN T, O'DONOGHUE B, SETZER S. Fast alternating direction optimization methods[J]. SIAM Journal On Imaging Sciences, 2014(3): 1588-1623.

[24] BLAKE C L, MERZ C J. UCI repository of machine learning databases[D]. Oakland: University of California, 1988.

[25] LECUN Y, BOTTOU L, BENGIO Y, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998(11): 2278-2324.

[26] LECUN Y, HUANG F J, BOTTOU L. Learning methods for generic object recognition with invariance to pose and lighting[C]//Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. [S.l.]: IEEE, 2004: II-94-II-104.

[27] LEE K, HO J, KRIEGMAN D J. Acquiring linear subspaces for face recognition under variable lighting [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2006(5): 15.

[28] HINTON G E, SALAKHUTDINOV R R. Reducing the dimensionality of data with neural networks[J]. Science, 2006(5786): 504-507.

[29] CENT P, LAROCHELLE H, BENGIO Y, et al. Extracting and composing robust features with denoising autoencoders[C]//Proceedings of the 25th International Conference Machine Learning (ICML). New York, USA: [s.n.], 2008: 1096-1103.

[30] BISHOP C M. Pattern recognition and machine learning (Information science and statistics)[M]. Secaucus, N J, USA: Springer-Verlag, 2006.

[31] HUANG G, ZHU Q, SIEW C. Extreme learning machine: A new learning scheme of feedforward neural networks[C]//Proceedings of 2004 IEEE International Joint Conference on Neural Networks. [S.l.]: IEEE, 2004: 985-990.

[32] CAMBRIA E, HUANG G B, KASUN L L C, et al. Extreme learning machines [Trends & controversies] [J]. IEEE Intelligent Systems, 2013, 28(6): 30-59.

[33] TANG J, DENG C, HUANG G. Extreme learning machine for multilayer perceptron[J]. IEEE Transactions on Neural Networks and Learning Systems, 2016, 27(4): 809-821.

[34] FENG S, CHEN C L P. A fuzzy restricted Boltzmann machine: Novel learning algorithms based on the crisp possibilistic mean value of fuzzy numbers[J]. IEEE Transactions on Fuzzy Systems, 2018, 26(1): 117-130.

[35] CORTES C, VAPNIK V N. Support vector networks [J]. Machine Learning, 1995, 20(3): 273-297.

[36] SUYKENS J A K, VANDEWALLE J. Least squares support vector machine classifiers[J]. Neural Processing letters, 1999, 9(3): 293-300.

[37] KRIZHEVSKY A, SUTSKEVER I, HINTON G E. ImageNet classification with deep convolutional neural networks[J]. Communications of the ACM, 2017(6): 84-90.

[38] CUN Y L, BOSER B, DENKER J S, et al. Handwritten digit recognition with a back-propagation network[J]. Advances in Neural Information Processing Systems, 1997, 2(2): 396-404.

[39] PELCKMANS K, SUYKENS J A K, GESTEL V, et al. LS-SVMlab: A MATLAB/C toolbox for least squares support vector machines[M]. Leuven, Belgium: KULeuven-ESAT, 2002.

（编辑：刘彦东）