# 基于 NAG 的 BP 神经网络的研究与改进

景立森[1] 丁志刚[2,3,4] 郑树泉[2,3,4] 廖 威[1]

[1](上海市计算技术研究所 上海 200040)

[2](上海产业技术研究院 上海 201206)

[3](上海计算机软件技术开发中心 上海 201112)

[4](上海嵌入式系统应用工程技术研究中心 上海 201112)

**摘 要** 针对当前 BP 神经网络存在的问题,在 NAG(Nesterov Accelerated Gradient)动量更新的基础上,建立基于黄金分割比梯度动量 GNAG(Golden ratio Nesterov Accelerated Gradient)更新策略。并把黄金分割比运用到隐含层神经元的确定上,进一步提高了 BP 神经网络的性能。将该算法应用于 MNIST 手写字体识别,得到了较好的收敛速度和预测评估结果。应用实例表明,基于黄金分割比梯度动量更新策略的 BP 神经网络收敛速度更快,泛化能力更强。

**关键词** BP 神经网络 NAG 梯度下降 学习率

**中图分类号** TP181 **文献标识码** A DOI:10.3969/j.issn.1000-386x.2018.11.046

## STUDY AND IMPROVEMENT OF BP NEURAL NETWORK BASED ON NAG

Jing Lisen[1] Ding Zhigang[2,3,4] Zheng Shuquan[2,3,4] Liao Wei[1]

[1](*Shanghai Institute of Computing Technology*, *Shanghai 200040*, *China*)

[2](*Shanghai Industrial Technology Institute*, *Shanghai 201206*, *China*)

[3](*Shanghai Development Center of Computer Software Technology*, *Shanghai 201112*, *China*)

[4](*Shanghai Embedded System Engineering Research Center*, *Shanghai 201112*, *China*)

**Abstract** To solve the problems existing in the current BP neural network, on the basis of NAG momentum update, we built an update strategy based on golden ratio nesterov accelerated gradient. The golden section ratio was applied to the determination of hidden layer neurons, thus further improving the performance of BP neural network. The algorithm was applied to MNIST handwriting recognition, which obtained better convergence speed and prediction evaluation results. Application examples show that BP neural network based on GNAG update strategy has faster convergence speed and stronger generalization ability.

**Keywords** BP neural network NAG Gradient descent Learning rate

## 0 引 言

近年来,随着计算机硬件和人工智能的迅速发展,涌现出丰富的机器学习算法。由文献[1]提出的对神经网络影响深远的误差反向传播算法(Back Propagation Algorithm)作为最流行的算法之一,也是应用最广泛的算法,其研究价值是不言而喻的。对于神经网络的优化,梯度下降也是最常用的优化方法之一。基于梯度下降的误差反向传播算法,在学术界和工业界都得到了很大程度的关注,对研究和改进此算法也是呈现极大的热情。

基于 vanilla 策略是梯度下降算法中最简单、方便的参数更新策略。参数沿着梯度变化的反方向更新,

收稿日期:2018-06-06。2016 年上海市科学技术委员会基金资助项目(16DZ1110101)。**景立森**,硕士生,主研领域:数据挖掘,机器学习。**丁志刚**,研究员。**郑树泉**,教授级高工。**廖威**,硕士生。

但此策略面对等高线或马鞍面时学习速度较慢。为了克服这一缺点,文献[2]提出了基于动量的更新策略[2]。在一定程度上提高了梯度下降的泛化能力和收敛速度,但是仅考虑到前一次梯度向量和当前梯度向量,没有突出当前梯度的重要性。NAG 算法[3]考虑到了前梯度向量迈出一步之后的梯度变化,使得梯度下降可以具有预测性,NAG 虽然比较智能,但是当前梯度动量和迈出一步之后梯度方向动量没有统一的标准,泛化能力还可以进一步优化。本文在动量更新策略和 NAG 的基础上,通过确定当前梯度和预测梯度呈黄金分割比例来优化,泛化能力得到了进一步的提高。

对于 BP 神经网络隐含层神经元的确定,以往都是凭经验确定,数目过多会过拟合,数目过少特征提取又不充分。经验主义的学者们都是基于定量的,这样也不能很好地得到较优的 BP 网络。

本文对于当前误差反向传播神经网络存在的两个问题,建立了基于黄金分割比的 GNAG 动量更新梯度策略,以及基于黄金分割比的隐含层神经元确定策略,使得 BP 神经网络的性能得到进一步优化。将此算法应用于手写数字识别,得到了较好的收敛速度和评估结果。

## 1　BP 神经网络

BP 神经网络是目前应用最广泛的神经网络[4],对于经典的三层结构其拓扑结构如图 1 所示。

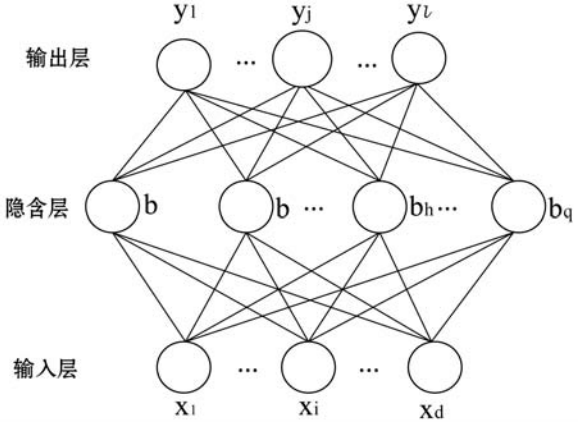


图 1　BP 神经网络

BP 算法不仅可以用于多层前馈神经网络,也可以用于其他类型的神经网络,例如递归神经网络。对于 BP 算法,设给定训练集 $T=\{(x_1,y_1),(x_2,y_2),\cdots,(x_m,y_m)\}$,$x_i\in R^d$,$y_i\in R^l$,也就是说输入变量有 $d$ 个维度,输出变量有 $l$ 个维度,隐含层有 $q$ 个神经元维度。记输出层第 $j$ 个神经元的阈值用 $\theta_j$ 表示,隐含层第 $h$ 个神经元的阈值用 $\gamma_h$ 表示,输入层第 $i$ 个神经元与隐含层第 $h$ 神经元之间的连接权为 $v_{ih}$,隐含层第 $h$ 个神经元与输出层第 $j$ 个神经元之间的连接权重为 $w_{hi}$;记隐含层第 $h$ 个神经元接收到的输入为 $\alpha_h=\sum_{i=1}^{d}v_{ih}x_i$,输出层第 $j$ 个神经元接受到的输入为 $\beta_j=\sum_{h=1}^{q}w_{hj}b_h$,其中 $b_h$ 为隐含层第 $h$ 个神经元的输出[5]。

### 1.1　经典梯度下降

对于训练实例 $(x_k,y_k)$,假设神经网络的输出为 $O_k=(O_1^k,O_2^k,\cdots,O_l^k)$,隐藏层和输出层的激活函数采用 Sigmoid 函数,则:

$$O_j^k=s(\beta_j-\theta_j) \tag{1}$$

网络在 $(x_k,y_k)$ 上的均方误差为:

$$E_k=\frac{1}{2}\sum_{j=1}^{l}(O_j^k-y_j^k)^2 \tag{2}$$

对于图 1 的网络,需要确定的参数有:输入层到隐含层的 $d\times q$ 个权值,隐含层到输出层的 $q\times l$ 个权值,$q$ 个隐含层神经元的阈值,$l$ 个输出层神经元的阈值,总共有 $(d+l+1)q+l$ 个参数需要确定[5]。

BP 算法基于梯度下降策略,以目标的负梯度方向对参数进行优化,对式(2)的误差 $E_k$,给定学习率 $\eta$,由偏微分的性质,则有:

$$\Delta w_{hj}=-\eta\frac{\partial E_k}{\partial w_{hj}}=-\eta\frac{\partial E_k}{\partial O_j^k}\cdot\frac{\partial O_j^k}{\partial\beta_j}\cdot\frac{\partial\beta_j}{\partial w_{hj}} \tag{3}$$

由 $\beta_j$ 的定义,显然可以得到:

$$\frac{\partial\beta_j}{\partial w_{hj}}=b_h \tag{4}$$

根据式(1)和式(2),有:

$$\begin{aligned}g_i&=\frac{-\partial E_k}{\partial O_j^k}\cdot\frac{\partial O_j^k}{\partial\beta_j}=\\&-(O_j^k-y_j^k)s'(\beta_j-\theta_j)=\\&-(O_j^k-y_j^k)s(\beta_j-\theta_j)[1-s(\beta_j-\theta_j)]=\\&(y_j^k-O_j^k)O_j^k(1-O_j^k)\end{aligned} \tag{5}$$

将式(4)和式(5)代入式(3),即可得到 BP 算法关于 $w_{hj}$ 的更新公式:

$$\Delta w_{hj}=\eta g_j b_h \tag{6}$$

对于输出层神经元阈值有:

$$\begin{aligned}\Delta\theta_j&=-\eta\frac{\partial E_k}{\partial\theta_j}=-\eta\frac{\partial E_k}{\partial O_j^k}\cdot\frac{\partial O_j^k}{\partial\theta_j}=\\&-\eta(O_j^k-y_j^k)s'(\beta_j-\theta_j)(-1)=\\&\eta(O_j^k-y_j^k)s(\beta_j-\theta_j)[1-s(\beta_j-\theta_j)]=\\&-\eta(y_j^k-O_j^k)O_j^k(1-O_j^k)=-\eta g_j\end{aligned} \tag{7}$$

对于输入层和隐含层的权值,有:

$$\Delta v_{ih}=-\eta\frac{\partial E_k}{\partial v_{ih}}=-\eta\frac{\partial E_k}{\partial b_h}\cdot\frac{\partial b_h}{\partial\alpha_h}\cdot\frac{\partial\alpha_h}{\partial v_{ih}}=$$

$$-\eta\frac{\partial E_k}{\partial \beta_j}\cdot\frac{\partial \beta_j}{\partial b_h}\cdot s'(\alpha_h-\gamma_h)\cdot x_i=$$

$$-\eta\sum_{j=1}^{l}(-g_j)w_{hj}s'(\alpha_h-\gamma_h)\cdot x_i=$$

$$\eta\sum_{j=1}^{l}g_j w_{hj}f(\alpha_h-\gamma_h)[1-f(\alpha_h-\gamma_h)]\cdot x_i=$$

$$\eta\sum_{j=1}^{l}g_j w_{hj}b_h(1-b_h)\cdot x_i=$$

$$\eta x_i b_h(1-b_h)\sum_{j=1}^{l}g_j w_{hj}=\eta e_h x_i \tag{8}$$

式中：$e_h=b_h(1-b_h)\sum_{j=1}^{l}g_j w_{hj}$。

对于隐含层的阈值，有：

$$\Delta\gamma_h=-\eta\frac{\partial E_k}{\partial \gamma_h}=$$

$$-\eta\frac{\partial E_k}{\partial b_h}\cdot\frac{\partial b_h}{\partial \gamma_h}=$$

$$-\eta\frac{\partial E_k}{\partial b_h}\cdot s'(\alpha_h-\gamma_h)\cdot(-1)=$$

$$\eta\frac{\partial E_k}{\partial b_h}\cdot b_h(1-b_h)=$$

$$\eta b_h(1-b_h)\frac{\partial E_k}{\partial \beta_j}\cdot\frac{\partial \beta_j}{\partial b_h}=$$

$$\eta b_h(1-b_h)\cdot\sum_{j=1}^{l}(-gj)w_{hj}=$$

$$-\eta b_h(1-b_h)\cdot\sum_{j=1}^{l}g_j w_{hj}=-\eta e_h \tag{9}$$

式中：$e_h=b_h(1-b_h)\sum_{j=1}^{l}g_j w_{hj}$。

至此，根据式(6)－式(9)，BP 网络中$(d+l+1)q+l$ 个参数即可更新迭代 BP 网络。

### 1.2 BP 神经网络隐含层数目的确定

为了抑制欠拟合和过拟合，对于 BP 神经网络的隐含层的数目确定，既不能太多也不能太少[6]，之前一般根据算法工程师的经验，去做相应的确定和优化。

随着人们对 BP 隐含层的深入研究，找到了一种相对来说有效的确定方法。对于图 1 所示的 BP 神经网络，隐含层神经元的数目 $q$ 可以用以下公式来确定[7]：

$$q=\sqrt{d+l}+a \tag{10}$$

式中：$d$ 是输入层神经元的维度；$l$ 是输出层神经元的维度；$a$ 是一个取 1 到 10 之间的调节常数。但是这种确定方法对于 $a$ 的确定也是凭经验来调节的，需要很多尝试，特别是对于维度比较小的 BP 网络，$a$ 的确定偶然因素太大。

## 2 BP 算法改进

### 2.1 梯度下降算法动量更新策略

对于 BP 神经网络的梯度下降算法，可以用类似于式(3)的 $\Delta w_{hj}=-\eta\frac{\partial E_k}{\partial w_{hj}}$方程表示。为了求得最优解在目标误差函数上运动，步长就是学习率。类似地，对于粒子在保守力的作用下的粘性介质中运动，假设摩擦系数为$\mu$，潜在的能量为 $E(w)$，根据牛顿第二定律有：

$$m\frac{\mathrm{d}^2 w}{\mathrm{d}t^2}+\mu\frac{\mathrm{d}w}{\mathrm{d}t}=\frac{-\mathrm{d}E(w)}{\mathrm{d}w} \tag{11}$$

式中：$w$ 是粒子的坐标矢量。可以把式(3)看作是一种没有质量的特殊粒子的情况。

通过对比梯度下降算法的更新策略和运动物体在保守力作用下在粘性介质中运动的对比，如果牛顿第二定律可以在梯度下降算法中起到某种作用，那将是很好的更新策略。下面来阐释怎么把牛顿定律类比地用在梯度更新策略上的。为了方便，把式(11)离散化一下：

$$m\frac{w_{t+\Delta t}+w_{t-\Delta t}-2w_t}{(\Delta t)^2}+\mu\frac{w_{t+\Delta t}-w_t}{\Delta t}=\frac{-\partial E(w)}{\partial w} \tag{12}$$

对式(12)，左右调整变形一下可得：

$$w_{t+\Delta t}-w_t=\frac{-(\Delta t)^2}{m+\mu\Delta t}\frac{\partial E(w)}{\partial w}+\frac{m}{m+\mu\Delta t}(w_t-w_{t-\Delta t}) \tag{13}$$

在物理学中，一个运动的物体具有惯性，把这个思想运用到梯度下降算法中，也就是说在梯度更新时，既在一定程度上保留先前更新的梯度方向的同时，也利用当前梯度方向做相应的调整[2]。根据式(13)可以设：

$$\Delta w=-p\frac{\partial E(w)}{\partial w}+\varepsilon\Delta w_{t-1} \tag{14}$$

式中：

$$p=\frac{(\Delta t)^2}{m+\mu\Delta t}\quad \varepsilon=\frac{m}{m+\mu\Delta t}$$

到此，使用最速梯度下降的动量更新策略和物体的运动规律有机地结合在一起，可以看出，当 $\varepsilon=0$ 意味着 $m=0$，反之亦然。

### 2.2 NAG 动量更新策略

当使用动量更新梯度算法时，可以看成是把一个小球扔下山坡，小球积累动量，速度变得越来越快，直到达到谷底。类似地，在动量更新策略沿着梯度最大的反方向往谷底滚去，直到抵达误差函数的最低点。

对于式(14)的动量更新策略,小球从山上滚下去的时候,盲目地沿着斜率方向走,并不会得到最优解。我们希望有一个智能的小球,可以知道自己要滚到哪里去,在下坡的过程中加速,在稍微有上坡的趋势的时候减速。此策略不但加快了收敛速度,同时可以抑制摇摆[8]。NAG 是一种能够给动量这种预测能力的一种算法[9]。对于普通的动量更新策略如图 2 所示。

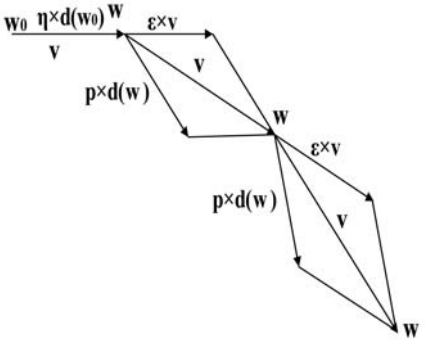


图 2　普通动量更新策略

$\varepsilon$ 为更新先前一次梯度方向上的动量学习率,$p$ 为更新当前点梯度方向上的动量学习率。观察式(14)可以把参数迭代分为两步来走,如图 3 所示。

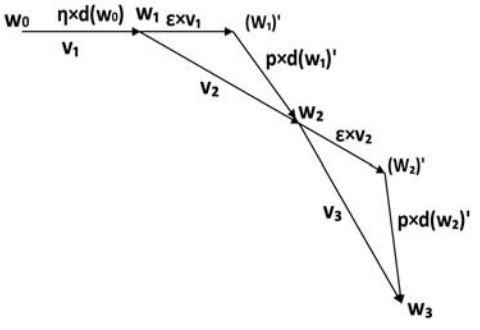


图 3　NAG 动量更新策略

先沿着先前的更新方向继续前进 $\varepsilon \times v_{i-1}$,到达点 $(w_{i-1})'$,再沿着 $(w_{i-1})'$ 的梯度方向前进 $p \times d((w_{i-1})')$,到达 $w_i$,此方法即为 NAG。由于在更新动量之前,在点 $(w_{i-1})'$ 做梯度方向的调整,属于超前预测,这种做法是有一定的"向前看"的优势[10],适应性比较好。

## 2.3　黄金分割比隐含层 BP 神经网络

黄金分割比是把一条线段分割为两部分,使其中一部分与全长之比等于另一部分与这部分之比,如图 4 所示。

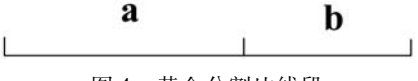


图 4　黄金分割比线段

黄金分割比等式:

$$\frac{a+b}{a}=\frac{a}{b}=\frac{1+\sqrt{5}}{2}\approx 1.618 \tag{15}$$

黄金分割具有严格的比例性、艺术性、和谐性,蕴藏着丰富的美学价值。由于按此比例设计的造型十分美丽,因此称为黄金分割比,也成为中外比。这个数值的作用不仅仅体现在诸如绘画、雕塑、音乐建筑等艺术领域,而且在管理、工程设计也有着不可忽视的作用。

本文把这个完美的比例,用在神经网络隐含层神经元的数目确定和 NAG 动量更新中。

对于 BP 神经网络的隐含层的数目,过多过少都不合适,会呈现不同的欠拟合或过拟合结果,但是有一点是很明确的,即总的来说,隐含层神经元的数目肯定要在某种程度上比输入层和输出层神经元多,至于多多少,一般都是根据项目经验去尝试从而确定合适的数目。即使用式(10)来确定,$a$ 也是凭经验来调节的,需要很多尝试,偶然因素也比较大。

因此我们引入基于黄金分割比的隐含层的确定方法。对于图 1 所示的 BP 网络,令:

$$\frac{q_1}{d}=\frac{q_2}{l}=\frac{1+\sqrt{5}}{2}=gr \tag{16}$$

根据式(16),求输入层和输出层确定的隐含层的平均,令隐含层神经元的个数为:

$$q=\frac{q_1+q_2}{2}=\frac{gr\cdot(d+l)}{2} \tag{17}$$

基于式(17)所确定的隐含层神经元数目,蕴藏着丰富的美学,能有效规避经验主义,又能很好地提取数据特征,达到很好的特征提取和数据转换工作,泛化能力进一步提高。

## 2.4　黄金分割比 NAG 动量更新策略

对于 NAG 算法,先沿着先前的更新方向继续前进 $\varepsilon \times v_{i-1}$,到达点 $(w_{i-1})'$,再沿着 $(w_{i-1})'$ 的梯度方向前进 $p \times d((w_{i-1})')$,到达 $w_i$,故有:

$$w_t=\varepsilon w_{t-1}+p\frac{\partial E_k}{\partial (w_{t-1})'} \tag{18}$$

动量首先计算当前梯度,接着沿着梯度负方向迈出一大步,计算梯度值,然后做修正。这个具有预见性的更新防止前进得太快,同时增强了算法的响应能力。和普通动量更新策略作对比,如图 5 所示。

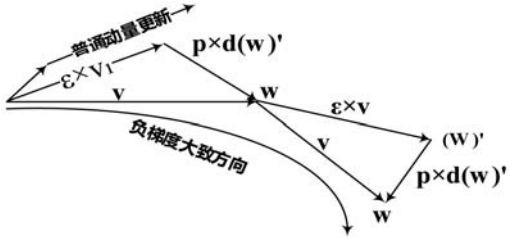


图 5　NAG 动量和普通动量对比图

虽然 NAG 有前瞻性,也有很好的预测性,收敛速度比较快,但是对于学习率 $\varepsilon$ 和 $p$,还是凭经验主义来人为设定的,泛化能力弱。通常情况下需要多次运行模型才能预估比较合适的学习率,偶然因素太大。再

次引入黄金分割比,令:

$$\frac{\varepsilon+p}{\varepsilon}=\frac{\varepsilon}{p}=\frac{1+\sqrt{5}}{2}\approx 1.618 \tag{19}$$

称满足式(19)的 NAG 动量为黄金 NAG 动量算法,记为 GNAG。由于原来需要两个学习率的确定,现在让其比值符合黄金分割比,有效地利用黄金分割的自然美学,融合到工程应用中。当前梯度负方向与下一步梯度负方向不一致时,两个动量叠加会得到有效抑制和调整,当两个动量方向大致一致时,二者叠加使得迈出一大步的学习步长。因此不但可以泛化学习算法,还可动态调整学习率,接下来利用实验来验证其有效性。

## 3 应用实例

### 3.1 实验验证

MNIST 数据集是目前测试神经网络最流行的数据集之一,相对于 $N$ 维奇偶问题[11]更具有代表性。采用 MNIST 数据集,来识别 0 ~ 9 的 10 个手写数字,图片大小是 28 × 28,故输入维度是 784 的变量,输出维度是 10,利用式(17),故隐含层神经元数目为:

$$q=\frac{gr\cdot(d+l)}{2}=\frac{1+\sqrt{5}}{2}\cdot\frac{784+10}{2}\approx 245$$

利用 google 的 tensorflow 开源框架来实现只含有一个隐含层的神经网络,设计并实现 NAG 的两个动量向量大小之比为黄金分割比,从而达到构建成 GNAG 的 BP 神经网络,利用 matplotlib 来可视化显示数据分析结果。构建的神经网络如图 6 所示。

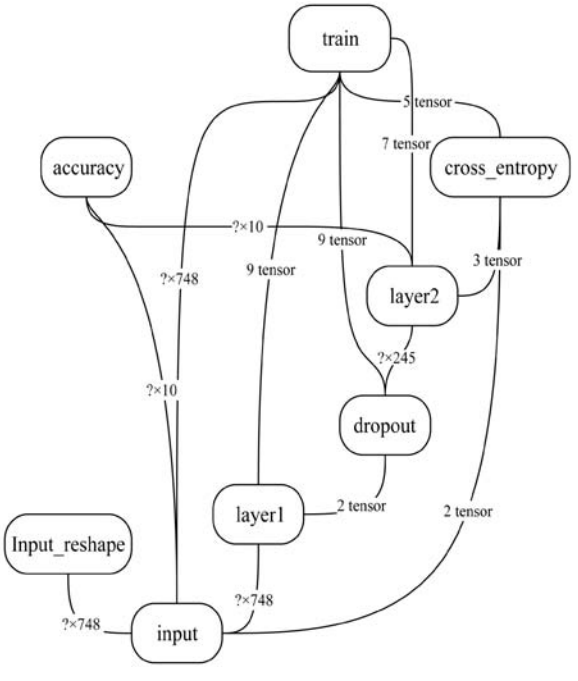


图 6 GNAG-BP 神经网络

### 3.2 结果分析

智能的 NAG 算法可用性已很高,而 GNAG 的性能比 NAG 更好,两者在精确度的对比如图 7 所示。

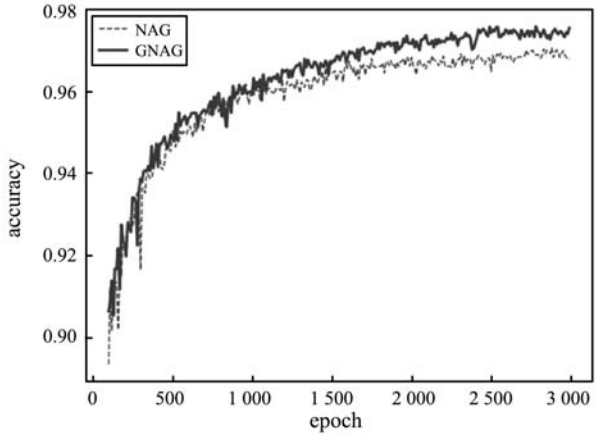


图 7 NAG 和 GNAG 精确度对比

由图 7 可知,GNAG 的精确度更高,收敛速度更快,误差平均损失函数更小,故学习能力更好,泛化能力更强。

对于 GNAG 优化算法的精确度和交叉熵代价函数分别如图 8 和图 9 所示。

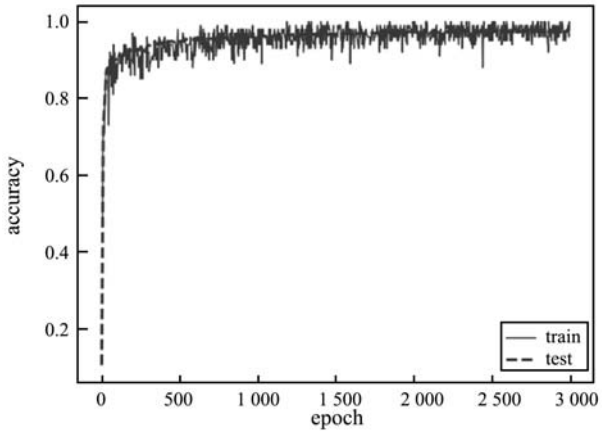


图 8 GNAG 精确度

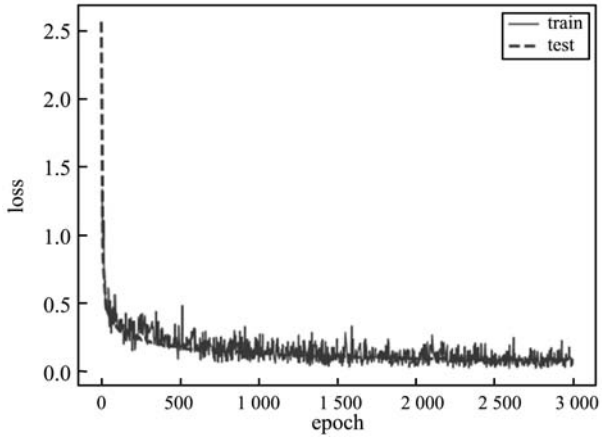


图 9 GNAG 损失函数

根据图 8 和图 9,可以看出,训练的精度和误差都很陡峭,说明 GNAG 的学习速度极快,GNAG 训练结果

围绕预测结果上下浮动,基本上预测结果在训练集的中心部位,没有出现过拟合或梯度消失等现象。虽然只是用的简单的三层黄金分割比例的 BP 神经网络,可是对于 MNIST 数据集我们的正确率已经达到了 97.55%,比人眼识别率还要高,可见黄金比例运用到 BP 网络中效果特别好。

## 4　结　语

本文在基于 NAG 算法的基础上,引入了黄金分割比,有机地结合了自然美学和工程应用,在泛化能力和收敛速度上,都得到了一定程度的提高。应用实例表明,基于黄金分割比梯度动量更新策略的 BP 神经网络误差更小,收敛速度更快,因此泛化能力更强。对于别的数据集,有待进一步研究和实验。

## 参　考　文　献


[1] Williams D, Hinton GE. Learning representations by back-propagation by errors[J]. Nature, 1986, 323(9): 533-536.

[2] Qian N. On the momentum term in gradient descent learning algorithms[J]. Neural Netw, 1999, 12(1): 145-151.

[3] Nesterov Y. Gradient methods for minimizing composite objective function[J]. Core Discussion Papers, 2007, 140(1): 125-161.

[4] 王磊,王汝凉,曲洪峰,等. BP 神经网络算法改进及应用[J]. 软件导刊,2016,15(5):38-40.

[5] 周志华. 机器学习[M]. 北京:清华大学出版社,2016:97-106.

[6] 王磊,王汝凉. 基于改进的 BP 神经网络方法的数据挖掘[J]. 广西师范学院学报(自然科学版),2016,33(1):79-84.

[7] 蔡荣辉,崔雨轩,薛培静. 三层 BP 神经网络隐层节点数确定方法探究[J]. 电脑与信息技术,2017,25(5):29-33.

[8] Alli-Oke R O, Heath W P. A secant-based Nesterov method for convex functions[J]. Optimization Letters, 2017, 11(1): 81-105.

[9] Ninomiya H. A novel quasi-Newton-based optimization for neural network training incorporating Nesterov's accelerated gradient[J]. Nonlinear Theory and Its Applications, IEICE, 2017, 8(4): 289-301.

[10] Botev A, Lever G, Barber D. Nesterov's accelerated gradient and momentum as approximations to regularised update descent[C]//2017 International Joint Conference on Neural Networks (IJCNN). IEEE, 2017: 1899-1903.

[11] 于涛. BP 网络自适应学习率算法分析[D]. 大连:大连理工大学数学学院,2011.


(上接第 162 页)

作总结如下:

(1) 经检验,本文建立的分区质量评价函数能够反映分区的质量。网格质量与分区质量相关度较高。

(2) 遗传算法能够有效处理结构化网格分区问题。与传统手工划分网格方法相比,极大提高了网格生成的成功率与自动化程度。

(3) 在采用遗传算法分区前,需要预先设定对应面上的分割点数目,尚未解决全自动处理复杂几何结构分区的问题。

## 参　考　文　献


[1] 戴萍, 林枫. 燃气轮机叶片气膜冷却研究进展[J]. 热能动力工程, 2009, 24(1): 1-6.

[2] 龚勋. 涡轮冷却叶片结构网格参数化方法研究[D]. 南京:南京航空航天大学, 2016.

[3] 董平. 航空发动机气冷涡轮叶片的气热耦合数值模拟研究[D]. 哈尔滨:哈尔滨工业大学, 2009.

[4] Lai Y K, Kobbelt L, Hu S M. Feature aligned quad dominant remeshing using iterative local updates[J]. Computer-Aided Design, 2010, 42(2): 109-117.

[5] 周天孝, 白文. CFD 多块网格生成进展[J]. 力学进展, 1999, 29(3): 344-368.

[6] 岳孟赫, 刘勇, 赵璐,等. 涡轮叶片结构化网格自动分区策略研究[J]. 航空计算技术, 2017,47(5): 68-72.

[7] 周明,孙树栋. 遗传算法原理及应用[M]. 北京:国防工业出版社,2002.

[8] 张洪梅. 三维六面体网格自适应生成算法研究及其应用[D]. 济南:山东大学,2007.

[9] Thompson J F, Bharat K S, Weatherill N P. Handbook of grid generation[M]. CRC Press, 2009.


(上接第 219 页)


[9] 何国正. 分布式智能网络爬虫的设计与实现[D]. 北京:中国科学院大学,2016.

[10] Mun H J, Li Y. Secure Short URL Generation Method that Recognizes Risk of Target URL[J]. Wireless Personal Communications, 2017, 93(1): 269-283.

[11] Sumathia P, Manickachezianb R. Semantic Web Mining using Web Crawler and DOM Tree with EsvmModified SOM for Advanced Medical Information Retrieval System[J]. International Journal of Control Theory and Applications, 2017, 10(12): 247-264.

[12] 姜子林. 层次聚类的方法及应用[J]. 电子技术与软件工程,2018(1): 179-180.