一种基于LM的量子神经网络训练算法

张翼鹏 陈 亮 郝 欢
(解放军理工大学通信工程学院 南京 210007)

摘 要 针对量子神经网络的训练结果易陷入局部极小值的问题,将Levenberg-Marquardt(LM)算法引入到原训练算法中,从而提高网络收敛速度与训练效果。并通过改进原训练算法的迭代步骤,解决训练过程中网络权值与量子间隔不同的目标函数相互冲突引起的输出均方误差和波动的问题。实验结果表明,相比原训练算法,引入LM后的训练算法可以大幅减少迭代次数,显著降低网络收敛值,提高量子神经网络的分类效果。

关键词 量子神经网络,Levenberg-Marquardt算法,最速下降,量子间隔
中图法分类号 TP183 **文献标识码** A

LM Based Training Algorithm for Quantum Neural Networks

ZHANG Yi-peng CHENG Liang HAO Huan
(Institute of Communications Engineering, PLA University of Science & Technology, Nanjing 210007, China)

Abstract Aiming at the question that the result trained by quantum neural networks is easy to fall into the local least value, the Levenberg-Marquardt algorithm was introduced into the original training algorithm to increase the training speed and improve the performance of the network. In addition, the conflict between different objective functions used for the training synaptic weights and quantum intervals can cause mean square error fluctuates. In order to solve this problem, the iteration order of the original training algorithm was adjusted. The experimental results show that, compared with the original training algorithm, the algorithm using LM can significantly reduce the number of iterations, significantly decrease the mean square error when the network convergences, and improve the classification results of quantum neural network.

Keywords Quantum neural networks, Levenberg-marquardt algorithm, Steepest descent, Quantum intervals

1 引言

人工神经网络(Artificial Neural Network, ANN)是实现并行分布式处理的有效工具,现已广泛应用于模式识别、信号处理、组合优化、自动控制等诸多领域[1]。

量子神经网络是学者Purushothaman和Karayiannis提出的一种新的单隐层前向型神经网络[2]。该神经网络由输入层、隐层和输出层构成。其中输入层和输出层与传统的前向神经网络相同,而隐层神经元借鉴了量子态叠加思想,将多个sigmoid函数线性叠加,构建成具有多层结构的激励函数。通过调整量子间隔的位置,可以使不同类的数据映射到不同的量级上,从而使分类获得更大的自由度。

量子神经网络需要训练两部分参数:1)网络的权值与偏置;2)多层激励函数中的量子间隔。根据文献[2],权值与偏置的训练采用基于输出误差最小的梯度下降法。而量子间隔则采用了不同的目标函数:同类输入数据的隐层神经元输出方差的总和最小,其训练算法依然是基于目标函数的梯度下降法。在网络的训练中首先对权值进行调整,使输入数据映射到不同的类空间中,其次对隐层神经元的量子间隔进行调整,从而体现数据的不确定性。由于这种固有的模糊特性,使得量子神经网络在数据分类中具有感知分类数据间模糊性的特性。通过剔除模糊类数据还可以进一步提高分类效果[3],这使得量子神经网络在文字识别、人脸识别、说话人识别等领域有着广泛的应用[4-6]。

尽管量子神经网络具有优良的性能,但其训练算法却存在明显的缺陷。根据文献[7],由于在网络训练中对权值和量子间隔采用了不同的目标函数,且两种参数的训练交替独立进行,因此在训练过程中两者训练的结果会相互冲突:针对一个目标函数的优化,可能会引起另一目标函数的恶化。针对这一问题,孙健等借鉴约束优化理论原理,在两个目标函数的梯度优化过程中引入惩罚函数,从而解决了这一问题[7]。但是,网络训练算法依旧采用梯度下降法,该算法不能保证训练结果的全局最优性,且收敛速度较慢。

Levenberg-Marquardt(LM)算法是介于牛顿法与梯度下降法之间的一种非线性优化方法,它能使目标函数陷入局部极小值的机率大大降低,且训练的迭代次数少,能有效节省训练时间。学者张鸿燕等[8]从理论上证明了LM算法本质上是在迭代过程中把原问题化为多个最小二乘(Least Square, LS)问题来求解,并通过分解矩阵结构揭示了LM算法优良特性的根源。学者Indrajit等[9]通过大量实验同样验证了LM算

到稿日期:2012-11-14 返修日期:2013-03-14 本文受国家自然科学基金项目(61072042)资助。
张翼鹏 硕士生,主要研究方向为语音信号处理与智能计算。

法的优良特性。学者 Wilamowski 等[10]改进了 Hessian 矩阵的计算过程，在一定程度上解决了 LM 算法需要较大存储量的问题，并且有效提高了运算速度。

本文针对量子神经网络的原始训练算法存在的不足，将 LM 算法引入到量子神经网络的训练中。以 LM 算法解决原始算法易使训练结果陷入局部最极小值的问题，并在算法设计上优化了迭代次序，从而解决原始算法训练中两个目标函数相互冲突的问题。本文接下来将介绍量子神经网络基于梯度下降的原始训练算法以及孙健等在文献[7]中给出的改进算法，然后给出本文设计的基于 LM 算法的量子神经网络训练策略，最后将所述 3 种训练算法进行对比仿真实验以验证本文算法性能。

2 基于梯度下降的训练算法及其改进

2.1 基于梯度下降的网络原始训练算法

量子神经网络的结构如图 1 所示。传统 sigmoid 以及三层 sigmoid 激励函数响应如图 2 所示。假设量子神经网络有 n_i 个输入神经元、n_h 个隐层神经元、n_o 个输出神经元。

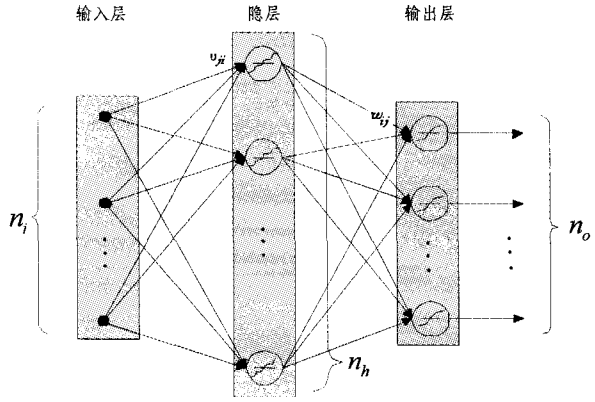

图 1 量子神经网络结构图

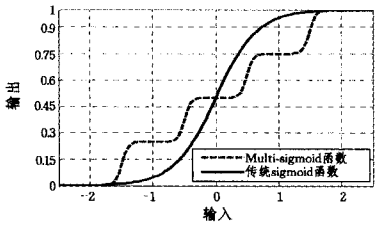

图 2 传统激励与多层激励函数响应图

训练时，首先训练权值 v_{ji} 和 ω_{ij}，再训练隐层神经元激励函数的量子间隔。网络权值的训练目标是使量子神经网络输出值与目标值的误差平方和最小，如式(1)所示：

$$\{v_{ji},\omega_{ij}\}=\mathrm{argmin}\{E=\sum_{k=1}^{K}\sum_{i=1}^{n_o}(y_{i,k}-\hat{y}_{i,k})^2\} \tag{1}$$

式中，E 表示网络权值 v_{ji} 和 ω_{ij} 的目标函数，K 为训练的样本个数。$y_{i,k}$ 表示当输入为 x_k 时，输出神经元 i 的目标值，$\hat{y}_{i,k}$ 表示实际值。量子间隔的训练目标是使同类输入数据的隐层神经元输出方差的总和最小，如式(2)所示：

$$\begin{aligned}\{\theta_j^r\}&=\mathrm{argmin}\{G=\sum_{j=1}^{n_h}\sum_{m=1}^{n_o}\sigma_{j,m}^2\}\\&=\mathrm{argmin}\{\sum_{j=1}^{n_h}\sum_{m=1}^{n_o}\sum_{x_k:x_k\in C_m}(\langle\tilde{h}_{j,C_m}\rangle-\tilde{h}_{j,k})^2\}\end{aligned} \tag{2}$$

式中，G 表示量子间隔 θ_j^r 的目标函数，$\tilde{h}_{j,k}$ 表示第 j 个隐层神经元的输出。C_m 表示由第 m 类输入数据构成的集合。$\langle\tilde{h}_{j,C_m}\rangle$ 表示由第 m 类输入数据所产生的，第 j 个隐层神经元输出结果的均值，如式(3)所示：

$$\langle\tilde{h}_{j,C_m}\rangle=\frac{1}{|C_m|}\sum_{x_k:x_k\in C_m}\tilde{h}_{j,k} \tag{3}$$

式中，$|C_m|$ 表示集合 C_m 的势，即集合 C_m 中的数据个数。

两类参数求解均采用针对目标函数的梯度下降法。首先调整网络的权值，ω_{ij} 与 v_{ji} 的更新可分别由式(4)、式(5)所示：

$$\omega_{ij,k}=\omega_{ij,k-1}+\Delta\omega_{ij,k-1}=\omega_{ij,k-1}-\alpha*\partial\frac{E}{\omega_{ij}} \tag{4}$$

$$v_{ji,k}=v_{ji,k-1}+\Delta v_{ji,k-1}=v_{ji,k-1}-\alpha*\partial\frac{E}{v_{ji}} \tag{5}$$

式中，α 表示网络权值的学习速率。然后调整量子间隔，θ_j^r 的更新可由式(6)所示：

$$\theta_{j,k}^r=\theta_{j,k-1}^r+\Delta\theta_{j,k}^r=\theta_{j,k-1}^r-\beta*\partial\frac{G}{\theta_j^r} \tag{6}$$

式中，β 表示量子间隔的学习速率。这样，经过多次迭代即可训练得到所需的权值以及量子间隔。目标函数 E 对权值的梯度、G 对量子间隔的梯度的推导以及迭代的具体过程可以参见文献[2]，这里不再赘述。

2.2 加入惩罚函数的改进算法[7]

孙健等借鉴约束优化理论，在原算法中引入惩罚函数，有效解决了 E 值和 G 值冲突的问题。将约束优化理论运用到量子神经网络的训练迭代中，首先需要在目标函数中加入限制条件，即：

$$\begin{aligned}&\{v_{ji}',\omega_{ij}'\}=\mathop{\mathrm{argmin}}_{\{v_{ji},\omega_{ij}\}}\{E=\sum_{k=1}^{K}\sum_{i=1}^{n_o}(y_{i,k}-\hat{y}_{i,k})^2\}\\&\text{s. t. } G(\{v_{ji}',\omega_{ij}'\})-G(\{v_{ji},\omega_{ij}\})\leqslant 0\end{aligned} \tag{7}$$

$$\begin{aligned}&\{\theta_j^r\}=\mathop{\mathrm{argmin}}_{\{\theta_j^r\}}\{G=\sum_{p=1}^{n_h}\sum_{m=1}^{n_o}\sigma_{p,m}^2\}\\&\text{s. t. } E(\{\theta_j^r\})-E(\{\theta_j^r\})\leqslant 0\end{aligned} \tag{8}$$

式中，$\{v_{ji}',\omega_{ij}'\}$ 和 $\{\theta_j^r\}$ 分别表示原参数集一次迭代更新后的结果。但在式(7)和式(8)的限制条件中，更新后的参数集是未知的，因此无法直接由限制条件构建惩罚函数。文献[7]将 E 值和 G 值分别作为对方的惩罚函数，由此得到两个新的目标函数：

$$\{v_{ji},\omega_{ij}\}=\mathrm{argmin}\{E+\beta_E G\} \tag{9}$$

$$\{\theta_j^r\}=\mathrm{argmin}\{G+\beta_G E\} \tag{10}$$

式中，β_E，β_G 为两个权值。在迭代过程中，这两个权值将会影响参数训练对 E 和 G 的偏重程度。

$$\begin{aligned}\beta_E&=\frac{E_p}{2G_p}\\\beta_G&=\frac{G_p}{2E_p}\end{aligned} \tag{11}$$

式中，E_p 和 G_p 分别表示迭代更新前的 E 值和 G 值，将比值再除以 2 是为了保证更新的重点在目标函数中的前一项。针对新的目标函数，采用梯度下降方法即可实现对量子神经网络的训练。其推导的详细过程可以参见文献[7]，这里不再详细推导。这样，通过引入惩罚函数，孙健等有效解决了 E 值和 G 值相互冲突的问题。

3 基于 LM 的量子神经网络训练算法

Levenberg-Marquardt(LM)算法非常适合性能指数是均方误差的神经网络训练。本节将详细阐述采用 LM 算法训练

量子神经网络的具体流程。

对于量子神经网络，目标函数 E 可以表示为 K 个样本集合的平方误差之和：

$$E=\sum_{k=1}^{K}\sum_{i=1}^{n_o}(y_{i,k}-\hat{y}_{i,k})^2=\sum_{k=1}^{K}\sum_{i=1}^{n_o}(e_{i,k})^2$$
$$=\sum_{i=1}^{N}(v_i)^2=\boldsymbol{v}^{\mathrm{T}}(\boldsymbol{x})\boldsymbol{v}(\boldsymbol{x}) \tag{12}$$

其中：

$$\boldsymbol{v}^{\mathrm{T}}(\boldsymbol{x})=[v_1 \quad v_2 \quad \cdots \quad v_N]$$
$$=[e_{1,1} \quad e_{2,1} \quad \cdots \quad e_{n_o,1} \quad e_{1,2} \quad \cdots \quad e_{n_o,K}],N=K\times n_o \tag{13}$$

将量子神经网络的权值和偏置合并成一个参数向量：

$$\boldsymbol{x}^{\mathrm{T}}=[x_1 \quad x_2 \quad \cdots \quad x_n]=[v_{1,1} \quad v_{1,2} \quad \cdots \quad v_{n_h,n_i} \quad b_1^1 \quad \cdots \quad b_{n_h}^1 \quad \omega_{1,1} \quad \cdots \quad \omega_{n_o,n_h} \quad b_1^2 \quad \cdots \quad b_{n_o}^2] \tag{14}$$

式中，$n=n_h(n_i+1)+n_o(n_h+1)$。若利用牛顿法更新 $\boldsymbol{x}$，即网络权值与偏置，则：

$$\boldsymbol{x}_{k+1}=\boldsymbol{x}_k-\boldsymbol{A}_k^{-1}\boldsymbol{g}_k \tag{15}$$

式中，$\boldsymbol{A}_k\equiv\nabla^2E|_{x=x_k}$，$\boldsymbol{g}_k\equiv\nabla E|_{x=x_k}$。目标函数 E 对 $\boldsymbol{x}$ 的梯度为：

$$\nabla E=2\boldsymbol{J}^{\mathrm{T}}(\boldsymbol{x})\boldsymbol{v}(\boldsymbol{x}) \tag{16}$$

其中：

$$\boldsymbol{J}(\boldsymbol{x})=\begin{bmatrix}\frac{\partial v_1(\boldsymbol{x})}{\partial x_1} & \frac{\partial v_1(\boldsymbol{x})}{\partial x_2} & \cdots & \frac{\partial v_1(\boldsymbol{x})}{\partial x_n}\\ \frac{\partial v_2(\boldsymbol{x})}{\partial x_1} & \frac{\partial v_2(\boldsymbol{x})}{\partial x_2} & \cdots & \frac{\partial v_2(\boldsymbol{x})}{\partial x_n}\\ \cdots & \cdots & \cdots & \cdots\\ \frac{\partial v_N(\boldsymbol{x})}{\partial x_1} & \frac{\partial v_N(\boldsymbol{x})}{\partial x_2} & \cdots & \frac{\partial v_N(\boldsymbol{x})}{\partial x_n}\end{bmatrix} \tag{17}$$

$\boldsymbol{J}(\boldsymbol{x})$ 为目标函数 E 的雅克比矩阵，其具体求解方法参见文献[1]。E 的 Hessian 矩阵为：

$$\nabla^2E=2\boldsymbol{J}^{\mathrm{T}}(\boldsymbol{x})\boldsymbol{J}(\boldsymbol{x})+2\boldsymbol{S}(\boldsymbol{x}) \tag{18}$$

其中：

$$\boldsymbol{S}(\boldsymbol{x})=\sum_{i=1}^{N}v_i(\boldsymbol{x})\nabla^2v_i(\boldsymbol{x}) \tag{19}$$

$S(x)$ 一般较小，可以忽略不计。则将式(16)和式(18)代入式(15)可得高斯-牛顿法参数向量 $\boldsymbol{x}$ 的更新算式：

$$\boldsymbol{x}_{k+1}=x_k-[\boldsymbol{J}^{\mathrm{T}}(\boldsymbol{x}_k)\boldsymbol{J}(\boldsymbol{x}_k)]^{-1}\boldsymbol{J}^{\mathrm{T}}(\boldsymbol{x}_k)\boldsymbol{v}(\boldsymbol{x}_k) \tag{20}$$

高斯—牛顿法不用计算二阶导数，但问题是 $\boldsymbol{H}=\boldsymbol{J}^{\mathrm{t}}\boldsymbol{J}$ 可能不可逆，LM 算法对其进行了改进，通过在 Hessian 矩阵 $\boldsymbol{H}$ 中加入修正因子 $\mu\boldsymbol{I}$，保证构造的矩阵 $\boldsymbol{G}$ 可逆，且修正后 $\boldsymbol{H}$ 的特征向量保持不变。

$$\boldsymbol{G}=\boldsymbol{H}+\mu\boldsymbol{I}=\boldsymbol{J}^{\mathrm{T}}\boldsymbol{J}+\mu\boldsymbol{I} \tag{21}$$

将 $\boldsymbol{G}$ 代入式(20)中的 $\boldsymbol{H}$，得到 LM 算法的更新公式：

$$\boldsymbol{x}_{k+1}=\boldsymbol{x}_k-[\boldsymbol{J}^{\mathrm{T}}(\boldsymbol{x}_k)\boldsymbol{J}(\boldsymbol{x}_k)+\mu_k\boldsymbol{I}]^{-1}\boldsymbol{J}^{\mathrm{T}}(\boldsymbol{x}_k)\boldsymbol{v}(\boldsymbol{x}_k) \tag{22}$$

当 $\mu_k=0$ 时，LM 算法退化为高斯—牛顿法。当 μ_k 增加时，LM 算法近似于较小学习速率的梯度下降法：

$$\boldsymbol{x}_{k+1}=\boldsymbol{x}_k-\frac{1}{2\mu_k}\nabla E(\boldsymbol{x}) \tag{23}$$

这样，便将 LM 算法引进到了量子神经网络权值与偏置的训练更新中。本文对量子间隔的训练采用了可变学习速率的梯度下降法，θ_j^r 的更新公式为：

$$\Delta\theta_{j,k}^r=\gamma\Delta\theta_{j,k-1}^r-(1-\gamma)\beta*\partial\frac{G}{\theta_j^r} \tag{24}$$

式中，γ 为动量因子，β 为量子间隔的学习速率。

结合 LM 算法和可变学习速率的梯度下降法的迭代步骤，本文设计的采用 LM 算法的量子神经网络训练算法的迭代步骤为：

(1)初始化训练所需参数：μ(默认值为 0.001)，β(默认值为 0.1)，$\gamma(0\leqslant\gamma<1)$，$\xi$(典型值为 1%至 5%)，$\rho(0<\rho<1)$，$\eta(\eta>1)$，$\theta(\theta>1)$。

(2)将样本集输入网络，得到网络输出 $\hat{\boldsymbol{y}}$ 和误差 $\boldsymbol{e}=\boldsymbol{y}-\hat{\boldsymbol{y}}$，计算所有样本的平方误差和 E 以及同类输入数据的隐层神经元输出方差的总和 G，并计算出矩阵 $\boldsymbol{J}$。

(3)由式(22)计算 $\Delta\boldsymbol{x}_k$；

(4)用 $\boldsymbol{x}_k+\Delta\boldsymbol{x}_k$ 带入网络计算新的平方误差和 E'。若 $E'<E$，则将 μ 除以 θ，并设 $\boldsymbol{x}_{k+1}=\boldsymbol{x}_k+\Delta\boldsymbol{x}_k$，转到步骤(5)；否则将 μ 乘以 θ，转到步骤(3)。

(5)由式(24)计算量子间隔的增量 $\Delta\boldsymbol{\theta}_k$，将 $\boldsymbol{\theta}_k+\Delta\boldsymbol{\theta}_k$ 代入网络计算此时的平方误差和 E'' 以及同类输入数据在隐层的输出方差总和 G'。

(6)如果 $G'/G>1+\xi$，将学习速率 β 乘以 ρ，动量因子 γ 置零；如果 $G'/G<1$，将学习速率 β 乘以 η，如果动量因子 γ 被置零，则恢复到以前的值；如果 $1<G'/G<1+\xi$，学习速率 β 保持不变，如果动量因子 γ 被置零，则恢复到以前的值。

(7)如果 $E''<E'$，$\theta_{k+1}=\theta_k+\Delta\theta_k$；否则 $\theta_{k+1}=\theta_k$。转回步骤(2)继续迭代。

当式(16)计算的梯度的模小于给定值，或平方误差总和 E 减小到某个目标误差，或迭代次数达到给定上限时，算法停止迭代，网络被认为收敛。由算法迭代步骤可以看出，不同于文献[2]算法流程，在调整量子间隔时，只有 E 值和 G 值同时下降，算法才会接受量子间隔的更新。这样就避免了在训练量子间隔时对 E 值造成的波动，使算法更加稳定。

4 实验与结果

为验证算法效果，从 TIMIT 语音库中随机选取 20 名说话人的语音作为实验对象，并选取较为稳定的浊音部分作为训练与测试语音。其中，每名说话人的训练语音长 4s，测试语音长 2s，每人有 8 段不同的测试语音。首先将语音数据以 16ms 为一帧进行分帧处理，然后对每帧数据计算 16 维梅尔倒谱系数(MFCC，Mel Frequency Cepstrum Coefficient)和 12 维线性预测系数(LPC，Linear Prediction Coefficient)，合成 28 维特征向量作为量子神经网络的输入进行仿真实验。采用两种参数的混合有助于提升识别结果的稳定性。输出数据为 10 维向量，用于指示输入数据所属的类别。当输入数据属于第 m 个说话人时，此向量中第 m 个元素为 1，其余元素均为 0。这样，对每个说话人将有 250 帧数据作为训练数据，8×125 帧数据作为测试数据。

根据输入输出数据的维数，构建具有 28 个输入神经元、10 个输出神经元的量子神经网络，隐层神经元选用层数为 3 的 multi-tansig 函数，量子间隔和斜率参数的初始值分别为[−2.5～2.5]和 2.5。输出神经元选用 purelin 线性函数。初始量子间隔的学习速率为 0.1，迭代次数上限设为 500。

图 3 给出了 3 种网络训练算法在不同隐层神经元下网络训练的收敛值。为了克服初始网络权值与偏置的随机性，图中数据值为 10 次实验的平均值。

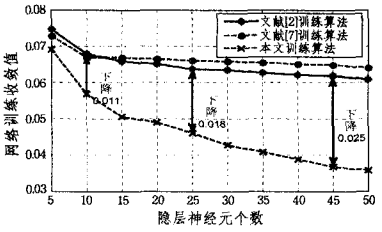

图 3　3 种训练算法收敛值的比较图

从图 3 可以看出，本文训练算法得到的网络训练收敛值更小，表示网络具有更高的拟合度，随着隐层神经元的增加，训练效果的提升更为明显。文献[7]训练算法为解决 E 值和 G 值的冲突，在 E 值中引入 G 为惩罚函数，牺牲了部分性能，故其网络训练收敛值略高于文献[2]算法得到的收敛值。

图 4 给出了在 40 个隐层神经元时，3 种训练算法的性能曲线图。

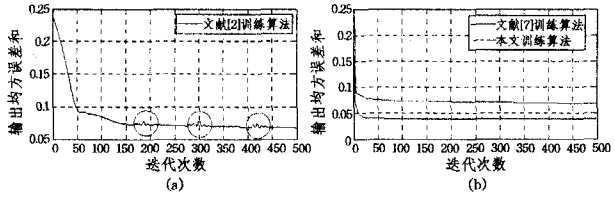

图 4　各算法的 E 值随迭代次数的变化图

比较图 4(a)(b)可以发现，本文算法与文献[7]得到的训练收敛值更低，文献[2]的训练算法达到的收敛值基本相当。本文训练算法和文献[7]训练算法与文献[2]训练算法相比，可以在较少的迭代步数后达到稳定状态。文献[2]算法得到的性能曲线图在红圈处存在较为明显的波动，本文训练算法和文献[7]训练算法的迭代过程则较为平稳。

量子神经网络具有感知数据间存在模糊性的能力，对于处于几类交叠区域的模糊数据，其对应于各类输出神经元的输出值就会非常接近。所以，对应于各类输出神经元的值也可以理解为输入数据属于该类的隶属度，当其中两个最大值的相差超过小时，说明输入数据处于这两类的交叠区域中。通过剔除这部分模糊数据，即可降低剩余数据中的错误分类概率。

图 5 给出了隐层神经元为 40 时，数据剔除门限从 0 到 0.3 时，3 种训练算法所得结果针对 16ms 帧的分类正确率随模糊数据剔除门限的变化情况。

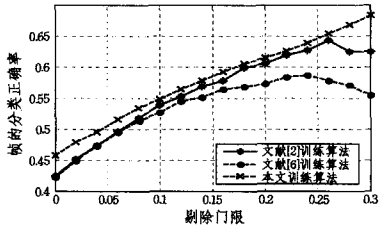

图 5　帧的分类正确率随剔除门限的变化图

从图 5 可以看出，本文算法具有良好的训练效果，帧正确率明显高于另两种算法。文献[2,7]算法对应的帧正确率在门限高于 0.25 时出现了一定程度的下降。这是因为，当门限为 0.25 时，其数据提出比例均超过 70%，且其训练算法不能保证训练结果的全局最优性，这样任何一个小的分类错误都会被放大，造成分类正确率的下降。

上述 3 种算法针对 2s 时说话人的分类正确率为 125 帧数据的联合处理结果，有一些浮动。其结果如表 2 所列。

表 2　3 种算法针对说话人的分类正确率

所用算法 \ 分类正确率	最小值	最大值	平均值
文献[2]训练算法	0.875	0.975	0.9125
文献[7]训练算法	0.8875	0.975	0.925
本文训练算法	0.9375	1	0.975

从表 2 可以看出，利用本文算法训练的网络可以得到更高的分类正确率，表明了本文训练算法具有更加优良的训练效果。

结束语　本文针对量子神经网络训练结果易陷入局部极小值的问题，将 LM 算法引入到量子神经网络的训练中，使网络收敛值明显下降，有效提高了训练速度与训练效果。并通过改进原训练算法的迭代步骤，在调整量子间隔时，只有 E 值和 G 值同时下降时，算法才会接受量子间隔的更新，从而解决了两个目标函数相互冲突引起的输出均方误差和波动的问题。最后通过仿真验证了本文所述训练算法的性能。

目前，量子神经网络的应用局限于数据分类上，这是因为量子间隔的目标函数是针对数据分类构造的。这就限制了量子神经网络进一步的发展与应用。下一阶段的研究将集中在量子间隔目标函数的改进上，使量子神经网络可以应用在函数逼近、分析预测等更多领域。

参 考 文 献

[1] Hagan M T, Demuth H B, Beale M H. Neural Network Design [M]. Boston: PWS Pub Co., 1995

[2] Purushothaman G, Karayiannis N B. Quantum neural networks (QNNs): inherently fuzzy feedforward neural networks [J]. Neural Networks, IEEE Transactions on, 1997, 8(3): 679-693

[3] Karayiannis N B, Xiong Y. Training Reformulated Radial Basis Function Neural Networks Capable of Identifying Uncertainty in Data Classification[J]. IEEE Transactions on Neural Networks, 2006, 17(5): 1222-1234

[4] 肖婧，谭阳红. 基于新特征提取法和量子神经网络的手写数字识别[J]. 电子测量技术，2009(06)：84-87

[5] 盖怀存，张小锋，江泽涛. 基于量子神经网络的人脸识别技术研究[J]. 计算机工程与应用，2010，46(8)：187-189

[6] 王金明，王耿，郑国宏，等. 一种量子神经网络说话人识别方法[J]. 解放军理工大学学报：自然科学版，2012，13(3)：242-246

[7] 孙健，张雄伟，孙新建. 一种新的量子神经网络训练算法[J]. 信号处理，2011，27(9)：1306-1312

[8] 张鸿燕，耿征. Levenberg-Marquardt 算法的一种新解释[J]. 计算机工程与应用，2009，45(19)：5-8

[9] Indrajit M, Srikanta R. Comparing the performance of neural networks developed by using Levenberg-Marquardt and Qusai-Newton with the gradient descent algorithm for modeling a multiple response grinding process[J]. Expert Systems with Applications, 2012, 39(3): 2397-2407

[10] Wilamowski B M, Hao Yu. Improved Computation for Levenberg-Marquardt Training [J]. IEEE Transactions on Neural Networks, 2010, 21(6): 930-937