

神经网络训练中的迭代扩展卡尔曼粒子滤波算法

张应博

(大连理工大学 城市学院, 辽宁 大连 116600)

摘 要: 基本粒子滤波算法已被成功用于训练神经网络, 但该算法在建议分布的选择上并没有考虑当前时刻观测值的影响, 针对该问题提出在神经网络训练中, 使用迭代扩展卡尔曼滤波器来生成建议分布. 由于迭代扩展卡尔曼滤波器在传递近似建议分布的均值和协方差的过程中, 充分利用了观测值信息, 从而可以更好地描述神经网络权值的后验概率分布. 实验结果表明, 在训练神经网络时, 迭代扩展卡尔曼滤波器作为建议分布的粒子滤波算法训练性能明显优于基本粒子滤波算法及扩展卡尔曼粒子滤波算法(EKPF).

关键词: 神经网络训练; 迭代扩展卡尔曼滤波; 迭代扩展卡尔曼粒子滤波

中图分类号: TP391

文献标识码: A

文章编号: 1000-7180(2010)08-0103-05

Iterated Extended Kalman Particle Filter for Neural Network Training

ZHANG Ying-bo

(City Institute, Dalian University of Technology, Dalian 116600, China)

Abstract: The generic particle filter has been applied with success to neural network training, but the proposal distribution chosen by the generic particle filter doesn't incorporate the latest observations which can deteriorate the performance of the algorithm. In this paper, we propose to use the iterated extended Kalman filter to generate proposal distribution in particle filtering framework. The iterated extended Kalman filter can make efficient use of the latest observation, and the generated proposal distribution can approximate the posterior distribution of neural network weights much better and improve the performance of particle filter. The experimental results show that the proposed particle filter outperforms the generic particle filter and the EKPF.

Key words: neural network training; iterated extended Kalman filter; iterated extended Kalman particle filter

1 引言

多层感知器(Multilayer perceptrons, MLP)是一种得到广泛应用的人工神经网络结构, MLP一般采用BP算法来训练. BP算法对于小型网络以及规模相对简单的问题能够取得较好的效果, 但是对于复杂问题或者网络中节点非常多的问题. S. Singhal等人^[1]提出将多层感知器的训练看作非线性动态系统的辨识问题, 使用状态空间法建立神经网络训练模型, 并应用扩展卡尔曼滤波算法(Extended Kalman Filter, EKF)来进行训练, 实验结果表明与传统的BP算法相比, EKF训练算法的收敛性能明

显优于BP算法.

然而, EKF算法对输出方程进行局部线性化的过程中常常会丢失一些重要的信息, 对于非线性输出模型而言, 使用EKF进行线性化以后得到的分布不能够很好地逼近真实的非线性系统分布. 为解决EKF算法存在的问题, Freitas等人^[2]提出使用SIR算法(Sequential Importance Sampling-Resampling)对神经网络进行训练. SIR算法, 即粒子滤波算法, 是近年来新兴的一种解决非线性滤波问题的方法, 其基本思想是用一个带有权重(也称重要性比率)的样本集合(也称粒子)来逼近感兴趣的概率分布. 但是SIR算法却存在严重的退化问题, 退化现象严重

影响粒子滤波算法的性能^[3]. 有效减少退化现象影响的方法之一是选择一个好的建议分布来产生样本粒子^[4-6]. 文献[7]提出的基于扩展卡尔曼粒子滤波算法的神经网络训练算法(EKPF)就是基于该思想. 为了提高训练的性能, 本文提出使用 IEKF 算法来构造近似高斯后验概率密度函数, 以此产生迭代扩展卡尔曼粒子滤波算法, 并应用该算法训练多层感知器网络, 在实验中与 SIR 算法和 EKPF 算法对比, 验证算法的训练性能.

2 背景知识

2.1 神经网络的状态空间模型

首先, 使用状态空间表示法为神经网络建模, 以描述神经网络随时间变化的过程. 使用一个状态迁移方程描述神经网络权值的变化, 一个测量方程描述某一具体物理过程的输入和输出之间的非线性关系, 方程如下:

$$w_{k+1} = w_k + d_k \quad (1)$$

$$y_k = g(w_k, x_k) + v_k \quad (2)$$

式中, $y_k \in \mathcal{R}$ 表示输出测量, $x_k \in \mathcal{R}^d$ 表示输入测量, $w_k \in \mathcal{R}^m$ 表示神经网络的权值. 非线性测量函数 $g(\cdot)$ 使用一个多层感知器来逼近. 假设系统测量噪声为 v_k , 假设为 0 均值高斯噪声, 协方差为 R . 假设第 $k+1$ 时刻的网络权值依赖于 k 时刻的网络权值以及系统随机过程噪声 d_k . 过程噪声可用来描述系统参数变化的不确定性、系统建模误差或者未知的输入量, 如目标机动等. 假设过程噪声为 0 均值高斯噪声, 协方差为 Q .

2.2 粒子滤波算法

粒子滤波算法的基本思想是使用一个带权粒子集合来逼近在推理过程中需要进行的积分运算, 这些粒子取自于神经网络权值的后验概率密度函数.

$$\hat{p}(w_{0:k} | y_{1:k}) = \frac{1}{N_s} \sum_{i=1}^{N_s} \delta(w_{0:k} - w_{0:k}^i) \quad (3)$$

式中, $w_{0:k}^i$ 表示用来近似后验概率密度函数的粒子, $\delta(\cdot)$ 为 delta 函数. 将后验概率密度函数用这种方法逼近之后, 关于神经网络权值的期望函数 $E(f_k(w_{0:k})) = \int f_k(w_{0:k}) p(w_{0:k} | y_{1:k})$ 中涉及的积分运算可以近似为求和运算, 如下:

$$E(f_k(w_{0:k})) = \frac{1}{N_s} \sum_{i=1}^{N_s} f_k(w_{0:k}^i) \quad (4)$$

式中, N_s 表示使用的粒子数目, 粒子 $w_{0:k}^i$ 从后验概率密度函数中抽取.

在实际应用中, 直接从后验概率密度中抽取样本粒子是很难做到, 因此, 有学者提出从一个已知的概率分布密度, 即建议分布密度 $\pi(w_{0:k} | y_{1:k})$ 中抽取样本粒子, 使用这些粒子来近似后验概率密度^[4]. 引入建议分布以后, 神经网络权值的期望函数 $E(f_k(w_{0:k}))$ 计算如下:

$$\begin{aligned} E(f_k(w_{0:k})) &= \int f_k(w_{0:k}) \frac{p(w_{0:k} | y_{1:k})}{\pi(w_{0:k} | y_{1:k})} \pi(w_{0:k} | y_{1:k}) dw_{0:k} \\ &= \int f_k(w_{0:k}) \frac{p(y_{1:k} | w_{0:k}) p(w_{0:k})}{p(y_{1:k}) \pi(w_{0:k} | y_{1:k})} \pi(w_{0:k} | y_{1:k}) dw_{0:k} \\ &= \int f_k(w_{0:k}) \frac{q_k(w_{0:k})}{p(y_{1:k})} \pi(w_{0:k} | y_{1:k}) dw_{0:k} \quad (5) \end{aligned}$$

上式中, $q_k(w_{0:k})$ 称为非归一化重要性比率,

$$q_k(w_{0:k}) = \frac{p(y_{1:k} | w_{0:k}) p(w_{0:k})}{\pi(w_{0:k} | y_{1:k})} \quad (6)$$

$$\begin{aligned} E(f_k(w_{0:k})) &= \frac{1}{p(y_{1:k})} \int f_k(w_{0:k}) q_k(w_{0:k}) \pi(w_{0:k} | y_{1:k}) dw_{0:k} \\ &= \frac{\int f_k(w_{0:k}) q_k(w_{0:k}) \pi(w_{0:k} | y_{1:k}) dw_{0:k}}{\int q_k(w_{0:k}) \pi(w_{0:k} | y_{1:k}) dw_{0:k}} \\ &= \frac{E_\pi(q_k(w_{0:k}) f_k(w_{0:k}))}{E_\pi(q_k(w_{0:k}))} \quad (7) \end{aligned}$$

因此, 通过从建议分布函数 $\pi(\cdot)$ 中抽取粒子, 感兴趣的期望值的计算就可以近似为下式:

$$\begin{aligned} E(f_k(w_{0:k})) &\approx \frac{\frac{1}{N_s} \sum_{i=1}^{N_s} f_k(w_{0:k}^i) q_k(w_{0:k}^i)}{\frac{1}{N_s} \sum_{i=1}^{N_s} q_k(w_{0:k}^i)} \\ &= \sum_{i=1}^{N_s} f_k(w_{0:k}^i) \tilde{q}_k(w_{0:k}^i) \quad (8) \end{aligned}$$

式中, $\tilde{q}_k^i$ 为归一化重要性比率,

$$\tilde{q}_k^i = \frac{q_k^i}{\sum_{j=1}^{N_s} q_k^j} \quad (9)$$

如果粒子集合 $w_{0:k}^i$ 独立同分布, 当 N_s 趋于无穷大时, 后验概率密度函数可以近似为

$$\hat{q}(w_{0:k} | y_{1:k}) = \sum_{i=1}^{N_s} \tilde{q}_k^i \delta(w_{0:k} - w_{0:k}^i) \quad (10)$$

为了序贯估计某时刻 k 的后验概率密度函数

而不用修改 k 时刻以前的状态,可以将建议分布函数分解为以下形式:

$$\pi(w_{0:k} | y_{1:k}) = \pi(w_0) \prod_{j=1}^k \pi(w_j | w_{0:j-1}, y_{1:j}) \quad (11)$$

如果我们假定系统状态服从马科夫过程且系统观测值在给定系统状态的情况下相互独立,即:

$$p(w_{0:k}) = p(w_0) \prod_{j=1}^k p(w_j | w_{j-1}),$$

$$p(y_{1:k} | w_{0:k}) = \prod_{j=1}^k p(y_j | w_j) \quad (12)$$

将以上式(10)~(12)带入重要性比率的计算公式,可得到递归计算粒子重要性比率的公式,

$$q_k = \frac{p(y_{1:k} | w_{0:k}) p(w_{0:k})}{\pi(w_{0:k-1} | y_{1:k-1}) \pi(w_k | w_{0:k-1}, y_{1:k})}$$

$$= q_{k-1} \frac{p(y_{1:k} | w_{0:k}) p(w_{0:k})}{p(y_{1:k-1} | w_{0:k-1}) p(w_{0:k-1})} \cdot \frac{1}{\pi(w_k | w_{0:k-1}, y_{1:k})}$$

$$= q_{k-1} \frac{p(y_k | w_k) p(w_k | w_{k-1})}{\pi(w_k | w_{0:k-1}, y_{1:k})} \quad (13)$$

上式为我们提供了一种递归更新粒子重要性比率的机制.粒子滤波算法可参考文献[2].

3 迭代扩展卡尔曼粒子滤波神经网络训练算法

迭代扩展卡尔曼粒子滤波算法(iterated extended Kalman particle filter, IEKPF)[8]使用迭代扩展卡尔曼滤波产生重要采样密度,更新粒子的过程中可以充分运用观测值信息,提高算法的训练性能.

3.1 迭代扩展卡尔曼滤波器(IEKF)

假定所有随机变量为高斯的,那么关于神经网络权值的 $w_{0:k}$ 的后验概率密度 $p(w_{0:k} | y_{1:k})$ 可以近似为

$$p(w_{0:k} | y_{1:k})$$

$$= p(w_{0:k} | y_k, y_{1:k-1})$$

$$= \frac{1}{c} p(y_k | w_{0:k}) p(w_{0:k} | y_{1:k-1})$$

$$= \frac{1}{c} N(y_k; g(w_{0:k}, x_k), R)$$

$$N(w_{0:k}; \hat{w}_{k:k-1}, P_{k:k-1}) \quad (14)$$

在 $w_{0:k}$ 处最大化式(14),等价于最小化下式:

$$J(w_{0:k}) = 1/2 [y_k - g(w_{0:k}, x_k)]^T R^{-1} [y_k - g(w_{0:k}, x_k)] + 1/2 (w_{0:k} - \hat{w}_{k:k-1})^T P_{k:k-1}^{-1} (w_{0:k} - \hat{w}_{k:k-1}) \quad (15)$$

应用 Newton-Raphson 算法递归地将上式最小

化,就可以得到关于 $w_{0:k}$ 的最大后验估计值.预先确定迭代次数 m ,假设在第 j 次迭代时($j = 1, 2, \dots, m$),得到的神经网络权值估计值为 w^j ,然后将 $J(w)$ 在 w^j 展开成泰勒展式,如下:

$$J = J^j + J'^j (w - w^j) + \frac{1}{2} (w - w^j)^T J''_{ww} (w - w^j) \quad (16)$$

其中,

$J' = J |_{w=w^j}, J''_w = \nabla_w J |_{w=w^j}, J''_{ww} = \nabla_w \nabla_w^T J |_{w=w^j}, J'_w$ 和 J''_{ww} 分别为 J 关于 w_{k+1} 的梯度和 Hessian 矩阵.令式(16)关于 w 的梯度为 0,使得式(15)极小,迭代产生关于 w 的下次迭代的估计值为

$$w^{j+1} = w^j - (J''_{ww})^{-1} J'_w \quad (17)$$

其中,关于 w 的梯度 J'_w 通过下式计算:

$$J'_w = -g^j(\hat{w}_{k+1|k+1}^j, x_{k+1})^T \cdot R^{-1} [y_{k+1} - g(\hat{w}_{k+1|k+1}^j, x_{k+1})] + P_{k+1|k}^{-1} (\hat{w}_{k+1|k+1}^j - \hat{w}_{k+1|k}) \quad (18)$$

取 g 的一阶导数, J 的 Hessian 矩阵变为

$$J''_{ww} = g^j(\hat{w}_{k+1|k+1}^j, x_{k+1})^{-1} R^{-1} g(\hat{w}_{k+1|k+1}^j, x_{k+1}) + P_{k+1|k}^{-1}$$

$$= H_{k+1}^{j,T} R^{-1} H_{k+1}^j + P_{k+1|k}^{-1} \quad (19)$$

其中,

$$H_{k+1}^j = g(\hat{w}_{k+1|k+1}^j, x_{k+1}) \quad (20)$$

然后利用矩阵求逆,有

$$(J''_{ww})^{-1} = P_{k+1|k}^{-1} - P_{k+1|k} H_{k+1}^{j,T}$$

$$(H_{k+1}^j P_{k+1|k} H_{k+1}^{j,T} + R^{-1} H_{k+1}^j P_{k+1|k}) = P_{k+1|k+1}^{-1} \quad (21)$$

将式(19)和式(21)代入式(17),可得到迭代扩展卡尔曼滤波器的状态更新方程,

$$\hat{w}_{k+1|k+1}^{j+1} = \hat{w}_{k+1|k+1}^j + P_{k+1|k+1}^j H_{k+1}^{j,T} R^{-1} (y_{k+1} - g(\hat{w}_{k+1|k+1}^j, x_{k+1})) - P_{k+1|k+1}^j P_{k+1|k}^{-1} (\hat{w}_{k+1|k+1}^j - \hat{w}_{k+1|k}^j) \quad (22)$$

协方差矩阵的传递公式为

$$P_{k+1|k+1}^j = P_{k+1|k}^j + P_{k+1|k}^j H_{k+1}^{j,T}$$

$$(H_{k+1}^j P_{k+1|k}^j H_{k+1}^{j,T} + R)^{-1} H_{k+1}^j P_{k+1|k}^j \quad (23)$$

经过 m 次迭代之后,得到 $k+1$ 时刻的估计量 $\hat{w}_{k+1}$ 和 $P_{k+1|k+1}$.

3.2 迭代扩展卡尔曼粒子滤波算法

在粒子滤波算法的框架内,将后验概率密度函数近似为高斯密度函数,使用均值和方差即可描述该高斯函数.

$$p(w_{0:k} | y_{1:k}) \approx p_N(w_{0:k} | y_{1:k}) = N(\bar{w}_k,$$

$$P_k) \quad (24)$$

因此,我们只需要递归地计算权值的均值和相应的协方差即可,将近似高斯分布向前传递.在粒子滤波算法的框架内,对每一个粒子,使用 IEKF 产生并传递近似高斯的建议分布,即:

$$q(w_i^i | w_{0:k-1}^i, y_{1:k}) = N(\bar{w}_k^i, P_k^i) \quad (25)$$

$$i = 1, \dots, N_s$$

在时刻 k , 根据最新的观测值信息, 使用 IEKF 方程计算每一个粒子的均值和协方差的估计, 然后从构成的近似建议分布中采样第 i 个粒子, 该过程递归进行下去.

IEKPF 算法步骤总结如下:

步骤 1: 初始化神经网络权值 ($k = 0$ 时刻):

FOR $i = 1, \dots, N_s$,

从网络第一层先验分布 $p_1(w_0)$ 和第二层先验分布 $p_2(w_0)$ 中抽取粒子 w_0^i .

计算重要性比率:

$$q_0^i = p(y_0 | w_0^i)$$

归一化重要性比率:

$$\bar{q}_0^i = \frac{q_0^i}{\sum_{j=1}^{N_s} q_0^j}$$

步骤 2: FOR $k = 1, \dots, L$

步骤 2.1: 采样阶段

FOR $i = 1, \dots, N_s$,

计算雅可比矩阵 F_k^i & G_k^i ,

使用 IEKF 方程更新粒子:

$$\hat{w}_{k|k-1}^i = w_{k-1}^i$$

$$P_{k|k-1}^i = F_k^i P_{k-1}^i F_k^{iT} + G_k^i$$

For $j = 1, \dots, m$

计算观测模型的雅可比矩阵:

$$H_k^i \text{ \& \& } U_k^i;$$

用式(23)更新协方差矩阵 P_k^i , 用式(22)更新

均值 $\bar{w}_k^i$;

Endfor

构造重要采样密度, 抽取粒子 $w_k^i \sim N(\bar{w}_k^i,$

$P_k^i)$

计算重要性比率, 根据式(13).

归一化重要性比率

$$\bar{q}_k^i = \frac{q_k^i}{\sum_{j=1}^{N_s} q_k^j}$$

步骤 2.2: 再采样阶段

去除重要性比率较小的粒子, 同时复制重要性比率较大的粒子, 得到 N_s 个随机粒子 w_{k+1}^i , 为每个粒子分配大小相同的重要性比率, $q_k^i = 1/N_s$.

4 仿真实验

本节将通过实验验证 EKPF 算法的训练性能. 在实验中, 输入输出数据由下列非线性非平稳状态空间模型产生^[7],

$$x_{k+1} = 0.5x_k + \frac{25x_k}{(1+x_k)^2} + 8\cos(1.2k) + \text{process_noise} \quad (26)$$

$$y_k = \frac{x_k^2}{20} + 6\sqrt{0.05(k-1)} + 3 + \text{measurement_noise} \quad (27)$$

实验中, 使用一个多层感知器来逼近上述模型产生的数据. 多层感知器网络为三层前馈网络, 隐层神经元数为 10 个, 输出层为 1 个线性神经元. 用 SIR 粒子滤波算法、EKPF 算法、IEKPF 算法训练该多层感知器. 粒子滤波算法的粒子数目 N_s 为 50.

表 1 所示为两种算法的一步预测平均误差及运行时间对比, 实验进行 100 次独立运行, 每一次运行 120 个时刻. 从表中可以看出, 使用 IEKPF 训练算法的一步预测误差约为 4.58, 低于 SIR 训练算法的 6.84 和 EKPF 训练算法的 6.23, 在运行时间方面, SIR 为 109s, EKPF 为 114s, IEKPF 为 118s, 相差不多, IEKPF 算法训练运行时间稍高于 SIR 和 EKPF 算法. 图 1 所示为某一次运行中, 三种算法训练感知器对输出的预测误差曲线, 从图中可以看出, IEKPF 算法明显优于 SIR 和 EKPF 算法.

表 1 平均一步预测误差表

	SIR	EKPF	IEKPF
RMS error	6.84	6.23	4.58
Time/s	109	114	118

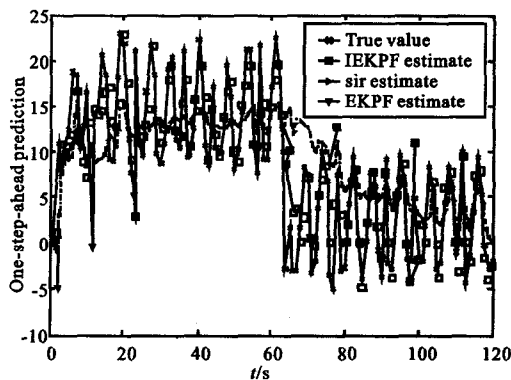


图 1 一步预测值与真实值对比曲线

5 结束语

本文提出一种新型的神经网络训练算法,即 IEKPF 算法. 在粒子滤波算法框架内,使用 IEKF 算法传播近似高斯建议分布,在迭代过程中有效利用观测值提供的信息,从而可以更好地描述神经网络权值的分布函数,本算法不仅可用于多层前馈网络,也可用于递归神经网络、径向基神经网络等的训练.

参考文献:

- [1] Singhal S, Wu L. Training multilayer perceptrons with the extended kalman algorithm[J]. *Advance in Neural Information Processing Systems*, 1998(1):133-140.
- [2] Freitas J F G, Johnson S E, Niranjana M, et al. Global optimization of neural network models via sequential sampling - importance resampling[C]// *Proc. of the 5th International Conference on Spoken Language Processing*. Sydney, Australia, 1998.
- [3] 于金霞,汤永利,刘文静. 粒子滤波自适应机制研究综述

[J]. *计算机应用研究*, 2010, 27(2):417-422.

- [4] Arulampalam M S, Maskell S, Gordon N, et al. A tutorial on particle filters for on-line nonlinear/non-gaussian bayesian tracking[J]. *IEEE Transactions on signal processing*, 2002, 50(2):174-188.
- [5] Doucet A, Godsill S, Andrieu C. On sequential monte carlo sampling methods for bayesian filtering[J]. *Statistics and Computing*, 2000(10):197-208.
- [6] Cappe O, Godsill S J, Moulines E. An overview of existing methods and recent advances in sequential monte carlo[J]. *Proceedings of the IEEE*, 2007, 95(5):899-924.
- [7] 王法胜,郭权. 基于扩展卡尔曼粒子滤波算法的神经网络训练[J]. *计算机工程与科学*, 2010, 32(5):47-49.
- [8] 李良群,姬红兵,罗军辉. 迭代扩展卡尔曼粒子滤波器[J]. *西安电子科技大学学报:自然科学版*, 2007, 34(2):233-238.

作者简介:

张应博 男,(1973-),双硕士,副教授.研究方向为人工智能、软件工程.

(上接第 102 页)

参考文献:

- [1] Estrin D, Srivastava M. Wireless sensor networks(Tutorial)[M]. USA: ACM MOBICOM, 2002.
- [2] Reed I S, Solomon G. Polynomial codes over certain finite fields[J]. *SIAM Journal of Applied Math.*, 1960(8):300-304.
- [3] MacKay D J C. Fountain codes[J]. *Capacity Approaching Codesdesign and Implementation Special Section*, 2005, 152(6):1062-1068.
- [4] Luby M, LT-codes[C]// *Proceedings of the ACM Symposium on Foundations of Computer Science (FOCS)*. Canada: Uancouver, 2002.

- [5] Levis P, Madden S, Gay D, et al. The emergence of networking abstractions and techniques in TinyOS[C]// *Proceedings of the First USENIX/ACM Networked System Design and Implementation*. San Francisco, 2004:1-14.
- [6] Levis P, Lee N, Welsh M, et al. TOSSIM: accurate and scalable simulation of entire tinyos applications[C]// *Proceedings of the 1st international conference on Embedded networked sensor system*. Los Angeles, 2003:126-137.

作者简介:

杜军朝 男,(1974-),博士,副教授.研究方向为操作系统、无线传感器网络.