第三章 基于改进的SMC算法以及BP神经网络的带参数马尔科夫模型验证框架

3.1框架概述

为了解决带参数的马尔科夫模型的验证问题，本章首先提出一种新的统计模型检验算法以解决已知参数下马尔科夫模型的检验问题，然后利用模型在若干参数下的验证数据使用基于神经网络的函数拟合的方法计算出模型在未知参数下的模型检验值。

SMC算法将模型检验看成一个参数估计问题予以解决，即假设马尔科夫模型满足某一时态逻辑公式的概率为， SMC算法的任务是计算符合可信度和精度要求的的估计值。

与其他SMC算法不同，我们在取样的时候引入了旨在减小估计值方差的对偶变量（antithetic variables）取样法，估计值方差的减小可以提高估计的精度。而在使用常规SMC算法时，为了提高精度则需提高样本大小。因而对偶变量取样法间接减少了随机路径取样数，提高了SMC算法的效率。

在得到了模型在特定参数点下的模型检验值之后，我们针对带参数模型的满足函数进行函数拟合。由于事先无法确定满足函数的解析形式，于是我们采用神经网络对满足函数进行拟合。不同于常规的神经网络拟合算法，本文从加权最小二乘法中借鉴了估计权值的思想，通过为模型在不同参数下的概率检测值赋予不同的权值，进而表征模型在不同的参数下模型检验值的可靠性，对可靠性高的检验值赋予更高的权重，从而优化了函数拟合的效果。

下面我们主要分为两个部分来叙述本框架采用的算法：基于改进的SMC算法的采样算法，以及基于BP神经网络的满足函数拟合算法。拟合算法采用SMC算法的样本点作为输入，拟合出一条与样本点尽量接近的曲线，从而预测模型在未知参数下的模型检验值。

3.2 基于对偶变量采样法的SMC算法

经典的蒙特卡罗方法通过重复性的取样获取对某个值的估计值。通过中心极限定理（central limit theorem），我们可以给出估计值的置信区间。但是对于同一个待估计的值，却可以有多个估计值，每个估计值的方差有大有小。估计值的可信度随着估计值的方差的增大而降低。常见的减小估计值的方差的方法有对偶变量法，共同变量法以及控制变量法等[5]。本文吸取了对偶变量并将其运用到SMC算法中。下面我们首先给出对偶变量的定义。

**定义1. 对偶变量** 称两个定义在实数域的随机变量和为一对对偶变量如果和服从同一分布且两者是负相关的关系。

关于对偶变量我们有如下定理[6]。

**定理1**. 设为一个偶数，为相互独立的对偶变量对，其中对于任意的整数，和均服从于某一特定分布，则

是的无偏估计，并且其方差为

其中，为和之间的相关系数。

从以上定理我们可以看出，由于和之间的对偶性使得恒成立，所以

恒成立，即通过对偶变量取样法得出的估计值的方差要小于标准蒙特卡罗方法的估计值。

下面通过一个简单的例子来说明对偶变量是如何减小估计值方差的 。

假设存在随机变量，的表达式为

其中，参数未知。现在需要对进行估计，即运用蒙特卡罗方法计算出一个尽可能准确的的估计值。

我们将看成是一个黑盒对参数进行估计。标准蒙特卡罗方法的思路是产生个独立且服从于均匀分布的随机变量，然后通过函数得到的个样本，则的估计值为

我们可以得出的方差为

由于相互之间彼此独立，所以对任意恒成立，上式可以写成

采用对偶变量的蒙特卡罗方法和上述方法略有不同。 假设样本大小为偶数，即。我们首先产生个独立且服从于均匀分布的随机变量，剩下的个随机变量则取作，的上标表示该随机变量是的对偶变量。类似地，我们给出的表达式：

类似的我们给出的方差

其中，等于

可以看出，由于和按照上述方法获取并且当两者不相等时，恒有即恒成立，所以我们有下式成立

下面分析方差的减小将如何影响样本的大小。

根据中心极限定理 ，均值为，方差为的独立同分布的随机变量的算数平均，当充分大时近似服从均值为，方差为的正态分布。

另外关于样本方差和均值我们还有以下定理[7]：

定理1 设分别是来自于总体的样本，，分别是样本的期望和方差，则有

上述定理表示近似服从参数为的t分布。

所以，我们有

即

于是我们得到的一个置信水平为的置信区间为

因为为的无偏估计，而估计值，于是我们有

所以上述置信区间也可以写作

分析上述置信区间我们可以发现，估计值的方差的减小能有效减小置信区间的宽度，这也从理论上反映出估计值的方差的减小能提高估计值的精度。进一步，假设估计值的方差减小为原来的，则为达到同样的精度，假设估计值的方差不变，样本大小必须增到原来的倍，因此我们有以下结论，即

结论： 假设现在要求大小为的置信区间为未知参数进行估计，如果使用标准蒙特卡罗方法需要大小为的样本，而使用对偶变量法使得估计值的方差是标准估计值的倍，则使用对偶变量法需要的样本大小为。

下文将阐述如何在SMC算法中运用上述思想。首先我们给出一条有限长的随机路径的对偶路径的定义，接着我们证明，通过利用对偶路径改进传统的SMC算法，可以有效减小估计值的方差，从而提高估计的精度。

通过回顾在第二章中描述的SMC算法可知，马尔科夫模型中的一条随机路径唯一对应于一个由随机数组成的向量。 借助于这一点，我们给出适用于DTMC和CTMC的对偶路径的定义。

定义1 马尔科夫模型的对偶路径 对应于任意一条马尔科夫模型产生的随机路径，其对偶路径为。

以下是完整的产生对偶路径的算法。

|  |
| --- |
| 算法2. 基于对偶变量的DTMC模型随机路径产生算法  输入：模型DTMC，路径长度d  输出：长度为d的两条对偶路径  if d < 0 then  return empty path  else  path1 = path()  path2 = path()  for i = 1 to d do  rnd = uniformRandom() // 产生[0,1]上服从均匀分布的随机数  antiRnd = 1 – rnd  nextState = chooseNextState(rnd)  nextState2 = chooseNextState(antiRnd)  path1.append(nextState)  path2.append(nextState2)  end for  return path1, path2 |

以上算法的输入同算法1，但是不同于算法1只返回一条随机路径，上述会返回两条互为对偶路径的随机路径。观察以上算法可以发现，算法2在为产生下个状态而生成随机数的同时，会将作为第二条随机路径选择下个状态的依据，从而产生了两条互为对偶路径的随机路径。

通过对SMC算法的阐述我们知道，马尔科夫模型的验证问题本质上是一个0-1分布的参数的估计问题。假设模型满足公式的概率为，我们的任务是计算出一个尽可能接近的估计值。

SMC算法的思路可以用下式来表示

其中，函数为

通过对0-1分布参数估计的问题的阐述我们发现，只要保证两条路径的验证结果和不同就可以减少估计值的方差。然而通过观察对偶路径的定义，我们发现上述定义无法保证互为对偶路径的两条路径的验证结果不同，此问题可以通过定义状态之间的一种偏序关系得以解决。

定义2 状态之间的偏序关系 对于马尔科夫模型，LTL公式，对于中的任意状态，我们定义为所有经过且满足公式的路径的总条数，为所有经过的路径的总条数。对于任意中两个状态和，我们称和满足偏序关系当且仅当

下面借助一个简单的模型阐述上述偏序关系的使用。

假设模型的状态转移图如下图所示：

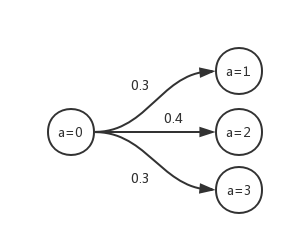


图1 模型状态转移图[此图需要修改，absorbing state]

假设我们的任务是求模型满足公式的概率，根据算法1[DTMC有限长随机路径产生算法]， 只需产生n个服从0到1之间的均匀分布的随机数即可，假设产生的随机数为。可以发现，当路径的最后一个状态为a=2时，该路径满足公式。同样是根据算法1[DTMC有限长路径产生算法]，我们发现，系统产生的两条对偶路径的路径的验证结果和并不一定不同，于是我们使用上述定义的偏序关系重新排列初始状态的所有使能状态（enabled state）。

要对状态进行排序，首先需要知道每个状态的和，所以我们首先需要产生一定数量的路径并进行验证，然后利用这些路径对模型中的状态进行成功路径和失败路径的计数，最后根据对状态进行排序，完整算法见下图所示。

|  |
| --- |
| 算法1 状态重排序算法  输入：n条随机路径组成的数组paths以及相应的验证结果组成的数组results，模型M  输出：None  for p, r in zip(paths, results) // 同时遍历路径数组和验证结果数组  for state in p  if r == true:  state.s\_cnt += 1 // s(S)加一  state.c\_cnt += 1 // c(S)加一  end for  end for  for state in M  states = next(state) // 获取状态state的全部使能状态  sort(states, key=state.s\_cnt/state.c\_cnt) // 根据state.s\_cnt/state.c\_cnt对状态进行排序  end for |

我们利用上述算法对上述模型进行状态重排序，实验对模型总共产生了10条路径，其中有4条为满足路径，其余的6条为失败路径，经过计数后可以发现状态1, 2, 3[换一种表述?]的值依次是0，1和0，于是排序后，状态2将排在初始状态全部使能状态的最后。

可以发现，经过使能状态重排序后，模型产生的两条互为对偶路径的路径的验证结果和不同的可能性大为提升，从而可以提升估计值的方差。

以下是完整的基于对偶路径的SMC算法。

|  |
| --- |
| 算法3. 基于对偶路径的SMC算法  输入：DTMC模型，bounded LTL公式，置信区间参数，  输出：DTMC模型在初始状态下满足特定PCTL公式的概率  n = 0 // 抽样总个数  positive = 0 // 验证结果为true的路径总数  samples = computeSamples(, d)  length = getLength() // 解析bounded LTL公式，获取取样路径长度  if samples % 2 != 0 then  samples += 1  end if  for i = 1 to samples / 2  path1, path2 = genRandomPathAnti(length) // 产生长度为length的两条随机路径  if checkLTL(path1, ) == true then  positive += 1  end if  if checkLTL(path2, ) == true then  positive += 1  end if  n += 2  end for |

可以看出，上述算法在传统SMC算法的基础上，通过采用对偶算法生成随机路径，除了可以减小随机数的生成数量外，还可以提高算法估计的准确性。

3.3 估计值加权的神经网络函数拟合方法

理论表明，神经网络可以以任意精度拟合任意函数[1]。所以，在很多传统函数拟合方法无法适用的场景，就可以采用神经网络作为拟合特定曲线的手段。例如，在生产生活中已知变量之间存在相互关系，但是无法确定其解析表达式时，则无法采用传统的最小二乘法对曲线进行拟合。

在本节中，我们首先会阐释神经网络的基本原理，先后阐释用于分类问题和回归问题的神经网络。接着，我们会引入在加权最小二乘法中引入的估计权值的概念，并阐述在神经网络的训练算法中如何引入这一概念帮助提高神经网络的拟合性能，最后我们给出完整算法并用例子说明。

在阐述本章将要介绍的算法之前，回顾我们在第二章中介绍的理论背景。本质上来讲，系统中的每条路径均对应一个实数值，这个实数值反应了这条路径在所有路径中被抽中的概率。而一个PCTL公式的验证结果可以用所有满足相应LTL公式的路径的概率之和来表示，即

根据定理1[神经网络可以拟合任何连续函数]，任意一个连续函数都可以被一个包含任意多个神经元的三层BP神经网络以任意精确度拟合。于是，我们首先定义一个带参数的离散马尔科夫模型的满足函数，接着证明该满足函数在定义域内是连续函数。

定义1. 满足函数 给定带参数的离散马尔科夫模型，PCTL公式，假设的取值范围为，定义定义域为，值域为的函数

即，给定内的一个点，我们定义了满足函数的值为带参数模型在参数取时模型满足公式的概率值。

下面证明上述定义的满足函数在定义域内是连续的。

定理2. 给定一个带参数的马尔科夫模型，假设模型中任意两个状态之间的转化概率关于均是连续函数的话，那么其对应的满足函数在定义域内是连续函数。

证明：由前所述，模型满足PCTL公式的概率可以用满足相应LTL公式的路径概率之和来表示。假设用来表示该集合。可以发现，与参数的具体取值无关。假设用表示该集合的路径概率之和。所以，我们只要证明是关于的连续函数即可。

又由于，所以我们只要证明任何一条随机路径被抽中的概率值是的连续函数即可。

由第二章中的理论背景可知，

其中是随机路径的长度，是某个转化发生的概率。由已知条件，关于是连续的，由连续函数的性质可知，关于是连续的，即结论得证。

结合定理1和定理2可知，我们可以使用BP神经网络拟合模型对应的满足函数。

3.3.1 用于分类和回归的BP神经网络

下面通过一个简单的分类问题说明神经网络的基本工作原理。

考虑识别手写体数字的问题[?]。下图展示了一些手写体数字的例子。



图1 手写体数字

在阐释BP神经网络之前，我们先给出S型神经网络的定义。

下图表示一个S型神经元。



图2 S型神经元

上述S型神经元接受作为输入，是其输出。的表达式如下所示：

其中，表示S型神经元的权重，表示其偏置。其中，（即逻辑函数，也叫S型函数）的形状酷似英文字母’s’，这也是S型神经元命名的由来。S型神经元是BP神经网络的基本组成单元。

一个典型的BP神经网络由三层神经元组成，分别为输入层，隐藏层以及输出层。输入层和输出层包含的神经元的个数与具体的输入以及问题的类型有关，隐藏层的神经元的个数属于超参数，一般是事先给定或者根据问题规模进行选择。

使用神经网络解决特定问题一般大致分为两步：训练与预测。前者是神经网络的训练阶段，最常见的算法是随机梯度下降算法（stochastic gradient descent, SGD）。后者包含了一个逐层前向传播的过程，所以一般也称为前向传播算法。下面依次叙述上述算法。

神经网络的训练问题可以这样描述，假设用表示神经网络的所有神经元的权重组成的向量，表示所有神经元的偏置，我们定义如下的代价函数：

其中，表示样本的个数，表示每个训练样本的输入，表示对应的训练样本的输出，表示神经网络在输入为的情况下的输出值，表示向量的模。我们一般把称之为二次代价函数。

有了代价函数，我们这样叙述BP神经网络的训练问题，即给定一组训练数据，我们需要找到使得代价函数最小的和。

如何找到合适的以及使得最小？这里经典的BP神经网络使用随机梯度下降。一个多元函数的梯度的定义如下：

从梯度的定义可以看出 ，多元函数的梯度的每个分量是函数关于每个分量的偏导数。  
关于梯度有如下显而易见的定理：

定理1. 对于多元函数来说，当自变量的增量时，。

上述定理可以由模的非负性证明。根据微积分，当函数的自变量发生了一个很小的变化时，如，函数值的变化量可以用下式近似表示：

结合上述两式，我们可以得出当时，函数值增量恒成立。所以，通过不断计算函数在当前点的梯度，从而计算出自变量的增量，就可以不断地使得函数值减小，直至达到最小值。

那么如何运用梯度下降算法求解式[?代价函数]呢？观察式[?代价函数]，我们可以发现是和的函数，对于任意的与 ，我们均使用下式进行更新：

根据定理1以及其推论，我们有恒成立。我们只需反复运用上述两式对中的每个以及进行迭代，直至达到某个迭代停止条件即可。

虽然从理论上分析梯度下降算法有较好的效果，但是从实践运用上来看仍存在不少问题，其中最严重的就是算法的运行效率问题。从上述两式可以看出，在每次迭代过程中，算法都要遍历训练样本中的每个进而求平均值，这样的效率不高。如果每次迭代只以少样样本作为迭代，效率可大大提高，而这也正是随机梯度下降的思路。

完整的随机梯度下降算法如下所示。

|  |
| --- |
| 算法1. 随机梯度下降算法  输入：训练样本  输出：代价函数的最小值  initialize and  repeat until convergence {  randomly choose m samples from      }  return and |

在上述算法中我们可以发现，随机梯度下降算法的每次迭代中，主要的工作就是计算代价函数关于神经网络中的权重和偏置的偏导数，BP神经网络使用反向传播（back propagation）算法对神经网络中每个神经元的权重和偏置进行更新。反向传播算法本质上使用导数的链式法则对神经网络进行逐层求偏导，与经典的神经网络训练速度相比，效率得到了极大的提高[3]。

神经网络的前向传播算法主要用于预测与分类，给定神经网络的输入，利用前向传播算法对神经网络进行逐层求值，从而得出最后的输出。具体的，假设神经网络中第层中第个神经元到第层中第个神经元的链接上的权重 为，第层上第个神经元的偏置为，第层上第个神经元的激活值为，那么第层上的第个神经元的激活值就可以用下式表示：

当采用向量表示每一层的各个神经元的偏置，用矩阵表示层和层之间链接的权重时，我们可以将上式写成

迭代地利用上式，我们就可以由输入层逐层向后计算，从而计算出输出层的值。

至此我们将神经网络的主要的基本原理叙述完毕，概括地说，神经网络使用反向传播算法计算代价函数关于权重和偏置的偏导数，利用随机梯度下降算法对权重矩阵和偏置向量进行迭代更新，从而求解使得代价函数最小的权重矩阵和偏置向量。得到这两个值之后，就可以利用训练好的神经网络对模型在未知输入下的输出进行预测，预测采用前馈算法。对于分类问题，输出层神经元的激活函数一般采用逻辑函数，逻辑函数将映射到，因此，对于二分类问题，可以如下定义：

而对于回归问题，输出层神经元的激活函数一般采用线性函数，即。

3.3.2. 加权神经网络拟合算法

但是，上述算法没有考虑到每个样本的可靠性。由于SMC算法本质上是一种蒙特卡罗方法，其得到的结果本身就带有一定偏差。而且，模型在不同参数下运行SMC算法获取的数据的偏差也不同。神经网络在训练时应该考虑到训练数据的可靠性。对于神经网络来说，如果已知观测值的可靠性要高于，那么应该给赋予更大的权重。换句话说，因为的可靠性要高于，因此其对函数拟合的影响也更大。下面，我们给出训练数据可靠性权重的概念 。

定义1.可靠性权重 可靠性权重表达了我们对于一个训练数据可靠性的度量，可靠性越高的观测数据，其在神经网络训练中的可靠性权重也越高。

上述定义仅从功能性方面阐述了可靠性权重，并未给出了可靠性权重应该如何定义。从直观上讲，可靠性权重应该由估计值的标准差唯一决定，即估计值的标准差越大，其可靠性程度越低。但是，如果要为每个估计值均计算它的标准差，计算代价又更大。因此本文提出一种分段计算标准差的方法。它首先将轴进行分段，在每一段中取若干个样本点，然后用直线对这些样本点进行线性拟合，拟合后可以计算出这些样本点到直线的平均偏移量，这个偏移量实际上表达了该段数据点的总体不确定性，平均偏移量越大，说明该段数据点的不确定性也越大。

针对第个子段中的个数据点，我们使用直线对数据点进行拟合。任意一个观测值与这条直线的偏差反映了每一段观测值的不确定性，它可以由下式表示：

基于上式，我们可以给出该段内所有样本点的可靠性权重为

下面给出完整的权重计算算法。

|  |
| --- |
| 算法2. 样本点权重计算算法  输入： 第段内所有样本点，  输出：该段内所有样本点可靠性权重  k = 1.0 // 直线斜率  b = 0.0 // 直线截距  = leastSquare( // 执行标准拟合直线最小二乘算法，得出斜率和截距的估计值  standardDiv = standardDiv(  return 1.0 / |

在实际的运用中，为了保证权重的引入不会影响到神经网络的训练，需要对权重采用归一化措施，即

在介绍加权神经网络拟合算法之前，我们需要回顾一下经典的反向传播算法。

首先让我们定义以下符号：假设我们定义从层第个神经元到第层的第个神经元的链接的权重为，第层中第个神经元的偏置为，第层中第个神经元的激活值为，第层上第个神经元的带权输入为，用向量形式表示即为。

我们定义第层上第 个神经元的误差为

经典的反向传播算法包含四个方程，首先，反向传播算法需要计算输出层的误差，即

上式的推倒依赖于偏导数的链式法则，其中，为逻辑函数。

上式用向量形式重写即为

其中，，，。

然后，我们利用神经网络下一层的误差计算当前层的误差，假设我们需要计算第层神经元中第个神经元的误差，即。由于上一层某个神经元的误差会传递到下一层所有神经元，所以需要对下一层的神经元的误差进行求和。

由于

故上式可以改写为

而上式的向量形式为

通过上述两个公式，我们可以计算出神经网络每一层的误差。而通过误差，我们可以求出代价函数关于神经网络中每个权重以及偏置的偏导数。

具体地，我们有代价函数关于第层神经元中第个神经元的偏置的偏导数为：

而代价函数关于某个权重的偏导数为

综上所述，反向传播算法通过计算输出层的误差然后逐层往后计算每一层神经元的误差，并通过链式法则计算代价函数关于权重和偏置的偏导数。

回顾代价函数，

如前所述，不同的样本具有不同的可靠性权重，因此我们改写上式，即

可以发现，可靠性越高的样本，在代价函数中所占的比重也就越大，其对权重和偏置的影响也就越大。而可靠性越低的样本，因为被赋予了更小的权重，对神经网络的影响也就越小。

下面给出完整的加权神经网络梯度下降算法。

|  |
| --- |
| 算法4. 加权神经网络随机梯度下降算法  输入：神经网络的权重向量以及偏置向量，个训练样本，样本的权重向量  输出：训练好的神经网络的权重以及  initialize and  repeat until convergence {  randomly choose m samples from      }  return and |

3.4 完整算法描述

下面通过一个例子描述算法的整个运行过程。

我们以如下模型为例。

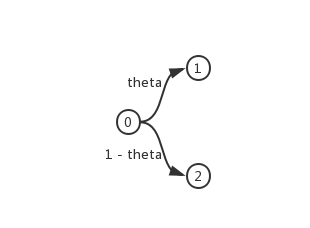


图4. 包含三个状态的带参数马尔科夫模型

从上述状态转移图我们可以看出，模型在状态0下的转移依赖于参数，从状态0转移至状态1的概率为，转移到状态2的概率为，状态1和状态2均为吸收状态。模型需要验证的公式为

表示系统在5个时间单位内到达状态1 的概率。

由于系统的行为取决于参数的具体值，所以上述模型是一个带参数的马尔科夫模型。这里，我们对参数在内进行验证。

首先我们对区间进行分段，假设将其均匀分为10段。依据算法[?加权随机梯度下降]，我们需要在每一段内取若干个样本点，运行SMC算法，假设每段取100个样本点。对每个参数点，我们运行基于对偶路径的SMC算法，得到100个样本点，。以下是我们在区间段内得到的100个样本点。

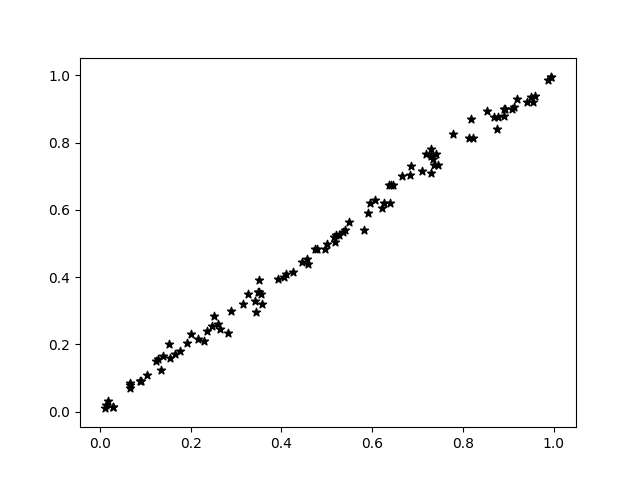


图5. 区间内100个样本点分布图

接着，我们需要为这100个样本点确定可靠性权重，将这100个样本点作为输入，运行线性拟合的最小二乘算法，得到直线，求出样本到直线的平均偏差，求出每个样本点的可靠性权重。重复为每个区间运行上述算法，计算出每个样本区间的可靠性权重。

下表为各个区间段的可靠性权重。

表1. 各区间段的可靠性权重

|  |  |
| --- | --- |
| 区间 | 权重 |
| [0.0, 0.1) | 2.005 |
| [0.1, 0.2) | 0.362 |
| [0.2, 0.3) | 0.880 |
| [0.3, 0.4) | 0.368 |
| [0.4, 0.5) | 1.287 |
| [0.5, 0.6) | 1.251 |
| [0.6, 0.7) | 0.4253 |
| [0.7, 0.8) | 0.865 |
| [0.8, 0.9) | 0.509 |
| [0.9, 1.0) | 2.045 |

观察图5以及表1可以发现，区间内的样本点越集中意味着区间中的样本点的可靠性越高，代表着这些样本点的权重也应该越高。得到了1000个样本点以及每个样本点对应的可靠性权重后，将这些信息输入给神经网络进行训练，就可以得到训练好的神经网络模型。下图是训练好的神经网络预测结果和利用PRISM软件对模型进行求解的对比图。

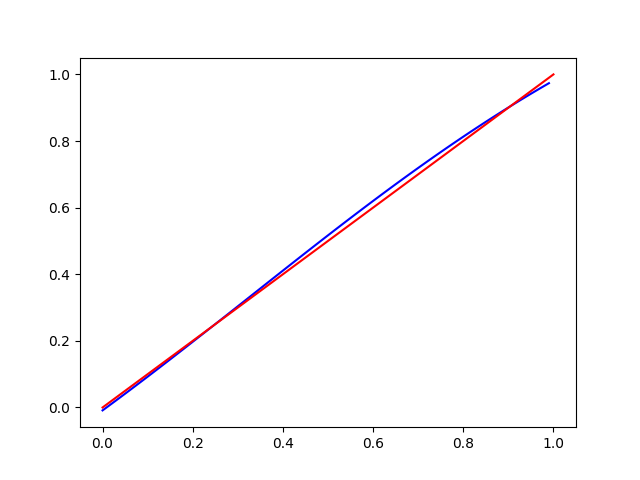


图5. 预测结果与PRISM结果对比图

从上图可以看出，神经网络预测的结果与PRISM软件通过解析计算出的结果相差无几。

可以认为PRISM工具的验证的结果为真实值，通过计算可以得出， 神经网络和PRISM在未知点（依旧位于[0，1]之间）的预测结果的误差为0.0096，满足精确度要求。

3.5 小结

本章分别阐述了基于对偶路径改进的SMC算法以及基于加权的神经网络训练算法，并结合具体案例给出了完整的算法描述。

[1] Hornik K, Stinchcombe M, White H. Multilayer feedforward networks are universal approximators[M]. Elsevier Science Ltd. 1989.

[1]Simulation and the monte carlo method

[3] Rumelhart D E, Hinton G E, Williams R J. Learning representations by back-propagating errors.[J]. 1986, 323(6088):399-421.

[5] <http://www.qest.org/qest2009/tutorial/tuffin1.pdf>

[6] Kroese D P, Taimre T, Botev Z I. Handbook of Monte Carlo Methods[M]. 2011.