Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №4

за 1 семестр

По дисциплине: «МиАПР»

Тема: «Нелинейные ИНС в задачах прогнозирования. Адаптивный шаг»

Выполнил:

Студент 2 курса

Группы ПО-4(1)

Калиновский Владимир  
Евгеньевич

Проверил:

Крощенко Александр Александрович

Брест 2020

Лабораторная работа №4

Нелинейные ИНС в задачах прогнозирования. Адаптивный шаг

Цель работы: Изучить обучение и функционирование нелинейной ИНС при решении задач прогнозирования с использованием адаптивного шага обучения.

Вариант 11

Задание:

Спрогнозировать нелинейный временной ряд, применяя параметры лабораторной работы №3. При этом необходимо использовать алгоритм обучения многослойной ИНС с адаптивным шагом. Для тестирования использовать функцию **

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| вариант | a | b | с | d | Кол-во входов ИНС | Кол-во НЭ в скрытом слое |
| 11 | 0.3 | 0.5 | 0.05 | 0.5 | 8 | 3 |

Код программы:

#include <iostream>

#include <math.h>

#include <iomanip>

using namespace std;

double function(double x);

double sigmoid(double x);

double output(double x, double w1[3][8], double w2[3], double T[3 + 1]);

double\* hidden(double x, double w1[3][8], double T[3]);

double get\_alpha(double w2[], double Error, double Output, double Hiddens[]);

int main()

{

double w1[3][8], w2[3], T[3 + 1], Reference, E\_min = 0.00002, alpha = 0.4, alpha2 = 0.4, x = 4, current, E = 0;

for (int i = 0; i < 3; i++)

{

for (int k = 0; k < 8; k++)

{

w1[i][k] = ((double)rand() / RAND\_MAX) \* 0.005;

}

w2[i] = ((double)rand() / RAND\_MAX) \* 0.005;

T[i] = ((double)rand() / RAND\_MAX) \* 0.005;

}

T[4] = ((double)rand() / RAND\_MAX) \* 0.005;

do

{

E = 0;

for (int q = 0; q < 400; q++)

{

current = output(x, w1, w2, T);

Reference = function(x + 8 \* 0.2);

double error = current - Reference;

double\* Hiddens = hidden(x, w1, T);

for (int j = 0; j < 3; j++)

w2[j] -= alpha \* error \* Hiddens[j];

T[4] += alpha \* error;

for (int k = 0; k < 3; k++)

{

for (int i = 0; i < 8; i++)

w1[k][i] -= alpha2 \* function(x + i \* 0.2) \* Hiddens[k] \* (1 - Hiddens[k]) \* w2[k] \* error;

T[k] += alpha2 \* Hiddens[k] \* (1 - Hiddens[k]) \* w2[k] \* error;

}

alpha2 = get\_alpha(w2, error, current, Hiddens);

x += 0.2;

E += pow(error, 2);

}

E /= 3;

cout << "error: " << E << endl;

} while (E > E\_min);

cout << "Ethalon value" << setw(15) << "Result " << setw(28) << "Delta " << endl;

for (int i = 0; i < 100; i++)

{

double Resultat = output(x, w1, w2, T), Ethalonn = function(x + 8 \* 0.2);

cout << fixed << setprecision(5) << Ethalonn << setw(21) << Resultat << setw(29) << Resultat - Ethalonn << endl;

x += 0.2;

}

system("pause");

}

double function(double x)

{

return 0.3 \* cos(0.5 \* x) + 0.05 \* sin(0.5 \* x);

}

double sigmoid(double x)

{

return 1 / (1 + pow(2, -x));

}

double output(double x, double w1[3][8], double w2[3], double T[3 + 1])

{

double Resultat = 0;

double\* hidden\_Result1 = hidden(x, w1, T);

for (int j = 0; j < 3; j++) {

Resultat += hidden\_Result1[j] \* w2[j];

}

Resultat -= T[4];

return Resultat;

}

double\* hidden(double x, double w1[3][8], double T[3])

{

double\* Result1 = new double[3];

for (int i = 0; i < 3; i++)

Result1[i] = 0;

double Inputs[8];

for (int k = 0; k < 8; k++, x += 0.2)

Inputs[k] = function(x);

for (int i = 0; i < 3; i++)

{

for (int k = 0; k < 8; k++)

Result1[i] += Inputs[k] \* w1[i][k];

Result1[i] -= T[i];

Result1[i] = sigmoid(Result1[i]);

}

return Result1;

}

double get\_alpha(double w2[], double Error, double Output, double Hiddens[])

{

double alpha = 0, A = 0, B = 0;

for (int i = 0; i < 3; i++)

{

A += pow(Error \* w2[i] \* (1 - Hiddens[i]) \* Hiddens[i], 2) \* Hiddens[i] \* (1 - Hiddens[i]);

B += pow(Error \* w2[i] \* (1 - Hiddens[i]) \* Hiddens[i], 2) \* Hiddens[i] \* Hiddens[i] \* (1 - Hiddens[i]) \* (1 - Hiddens[i]);

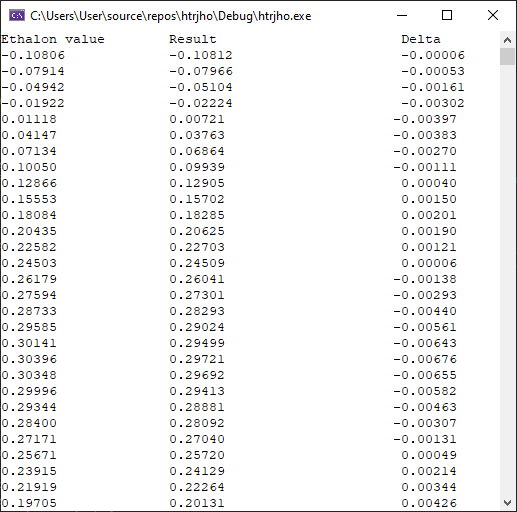
}

alpha = 4 \* A / (B \* (1 + Output \* Output));

return alpha;

}

Результат выполнения программы:



Вывод: Изучил обучение и функционирование нелинейной ИНС при решении задач прогнозирования с использованием адаптивного шага обучения.