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»

Выполнил:

Студент 2 курса

Группы ПО-4(1)

Боровский М.В.

Проверил:

Крощенко А.А.

Брест 2020

Лабораторная работа №4

Нелинейные ИНС в задачах прогнозирования

Цель работы: Изучить обучение и функционирование нелинейной ИНС при решении задач распознавания образов.

Вариант 3

Задание:

**Задание*.***

1. Спрогнозировать нелинейный временной ряд, применяя параметры лабораторной работы №3. При этом необходимо использовать алгоритм обучения многослойной ИНС с адаптивным шагом.
2. Результаты представить в виде отчета содержащего:
3. Титульный лист,
4. Цель работы,
5. Задание,
6. График прогнозируемой функции на участке обучения,
7. Результаты обучения: таблицу  со столбцами: эталонное значение, полученное значение, отклонение; график изменения ошибки в зависимости от итерации.
8. Результаты прогнозирования: таблицу  со столбцами: эталонное значение, полученное значение, отклонение.
9. Выводы по лабораторной работе.

В выводах сравнить полученные результаты с результатами лабораторной работы №3.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № варианта | a | b | с | d | Кол-во входов ИНС | Кол-во НЭ в скрытом слое |  |
| 3 | 0.3 | 0.3 | 0.07 | 0.3 | 10 | 4 |  |

Код программы:

#include <iostream>

#include <math.h>

#include <iomanip>

#define input\_layer 10

#define hidden\_layer 4

using namespace std;

double sigmoid(double x)

{

return 1 / (1 + pow(2.7, -x));

}

double function(double x)

{

double a = 0.3, b = 0.3, c = 0.07, d = 0.3;

return a \* cos(b \* x) + c \* sin(d \* x);

}

double\* hidden(double x, double w1[hidden\_layer][input\_layer], double\* T)

{

double\* result = new double[hidden\_layer];

for (int i = 0; i < hidden\_layer; i++)

result[i] = 0;

double Inputs[input\_layer];

for (int k = 0; k < input\_layer; k++, x += 0.1)

Inputs[k] = function(x);

for (int i = 0; i < hidden\_layer; i++)

{

for (int k = 0; k < input\_layer; k++)

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

result[i] -= T[i];

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

}

return result;

}

double get\_alpha(double\* w2, double Error, double Output, double\* Hiddens)

{

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

for (int i = 0; i < hidden\_layer; 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;

}

double output(double x, double w1[hidden\_layer][input\_layer], double\* w2, double\* T)

{

double Result = 0;

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

for (int j = 0; j < hidden\_layer; j++) {

Result += hidden\_result[j] \* w2[j];

}

Result -= T[2];

return Result;

}

int main()

{

setlocale(LC\_ALL, "rus");

int epox = 0;

double w1[hidden\_layer][input\_layer], w2[hidden\_layer], T[hidden\_layer + 1], Reference, E\_min = 0.00002, alpha = 0.4, alpha1 = 0.4, x = 4, current, E = 0;

for (int i = 0; i < hidden\_layer; i++)

{

for (int k = 0; k < input\_layer; k++)

{

w1[i][k] = ((double)rand() / RAND\_MAX) \* 0.05;

}

w2[i] = ((double)rand() / RAND\_MAX) \* 0.05;

T[i] = ((double)rand() / RAND\_MAX) \* 0.05;

}

T[2] = ((double)rand() / RAND\_MAX) \* 0.05;

do

{

E = 0;

for (int q = 0; q < 350; q++)

{

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

Reference = function(x + 10 \* 0.1);

double error = current - Reference;

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

for (int j = 0; j < hidden\_layer; j++)

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

T[2] += alpha \* error;

for (int k = 0; k < hidden\_layer; k++)

{

for (int i = 0; i < input\_layer; i++)

w1[k][i] -= alpha1 \* function(x + i \* 0.1) \* Hiddens[k] \* (1 - Hiddens[k]) \* w2[k] \* error;

T[k] += alpha1 \* Hiddens[k] \* (1 - Hiddens[k]) \* w2[k] \* error;

}

alpha1 = get\_alpha(w2, error, current, Hiddens);

x += 0.01;

E += pow(error, 2);

}

E /= 2;

cout << "Error " << E << endl;

epox++;

} while (E > E\_min);

cout << epox << endl;

cout << "Эталон" << setw(23) << "Прогноз" << setw(20) << "Отклонение1" << endl;

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

{

double Result = output(x, w1, w2, T), Ethalonn = function(x + 10 \* 0.1);

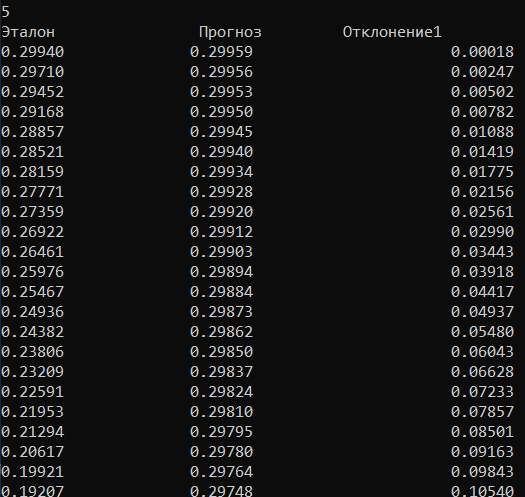
cout << fixed << setprecision(5) << Ethalonn << setw(21) << Result << setw(29) << Result - Ethalonn << endl;

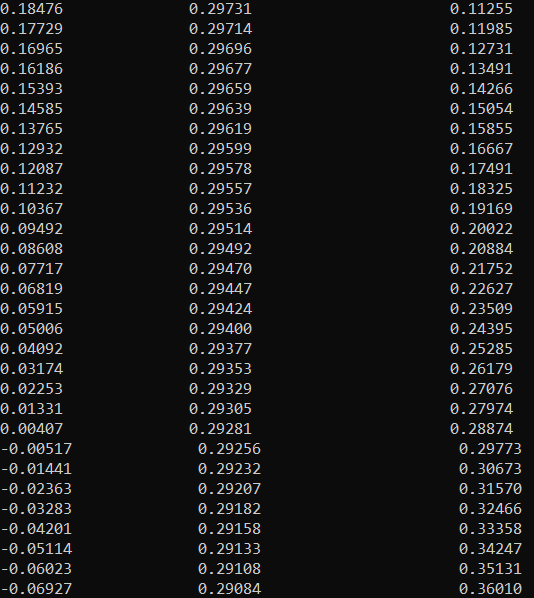
x += 0.1;

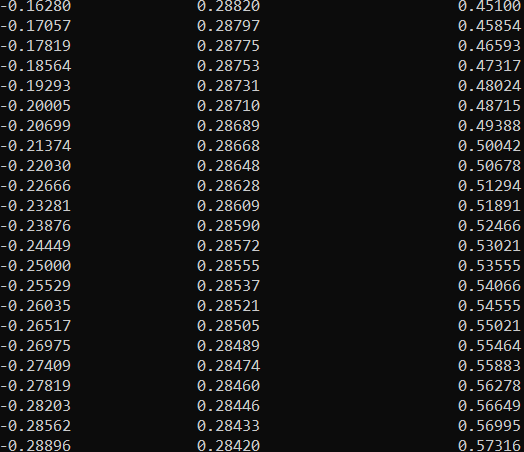
}

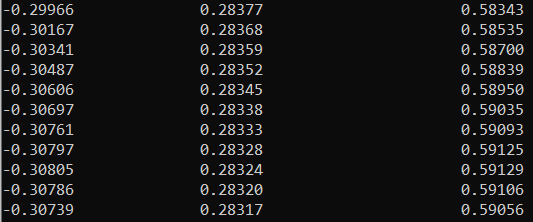
system("pause");

}









Вывод: В ходе работы разработал нелинейную ИНС для задач распознавания образов.