МИНЕСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧЕРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра ИИТ

**ЛАБОРАТОРНАЯ РАБОТА №3**

За первый семестр

«Линейная искусственная нейронная сеть. Адаптивный шаг обучения»

Выполнил

Студент ФЭИС

2-го курса, группы ПО-7(2)

Практика Д.О

Проверил

Крощенко А.А

Брест, 2021

**ЛАБОРАТОРНАЯ РАБОТА №3**

«Нелинейные ИНС в задачах прогнозирования»

Цель работы: изучить обучение и функционирование нелинейной ИНС при решении задач прогнозирования.

Вариант 7

Задание:

Написать на любом ЯВУ программу моделирования прогнозирующей линейной

ИНС. Для тестирования использовать функцию 

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № варианта | a | b | с | d | Кол-во входов ИНС | Кол-во НЭ в скрытом слое |
| 5 | 0.1 | 0.5 | 0.09 | 0.5 | 8 | 3 |

Алгоритм обратного распространения ошибки минимизирует среднеквадратичную ошибку нейронной сети. Для этого с целью настройки синаптических связей используется метод градиентного спуска в пространстве весовых коэффициентов и порогов нейронной сети.



График функции 0.3 \* cos(0.1\* x) + 0.06 \* sin(0.1 \* x):



Текст программы:

#include <iostream>

#include <math.h>

#include <iomanip>

using namespace std;

double Sigmoid(double x) {

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

}

double func(double x) {

double a = 0.3, b = 0.1, c = 0.06, d = 0.1;

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

}

double\* hidden(double x, double Wes1[2][6], double T[2]) {

double\* result\_value = new double[2];

for (int i = 0; i < 2; i++) {

result\_value[i] = 0;

}

double entrances[6];

for (int k = 0; k < 6; k++, x += 0.1) {

entrances[k] = func(x);

}

for (int i = 0; i < 2; i++) {

for (int k = 0; k < 6; k++) {

result\_value[i] += entrances[k] \* Wes1[i][k];

}

result\_value[i] -= T[i];

result\_value[i] = Sigmoid(result\_value[i]);

}

return result\_value;

}

double output(double x, double Wes1[2][6], double Wes2[2], double T[2 + 1]) {

double result = 0;

double\* hidden\_result\_value = hidden(x, Wes1, T);

for (int j = 0; j < 2; j++) {

result += hidden\_result\_value[j] \* Wes2[j];

}

result -= T[3];

return result;

}

int main() {

setlocale(0, "");

int stage = 0;

double Wes1[2][6], Wes2[2], T[2 + 1], reference\_value, E\_min = 0.00001, alpha = 0.7, x = 4, current, E = 0;

for (int i = 0; i < 2; i++) {

for (int k = 0; k < 6; k++) {

Wes1[i][k] = ((double)rand() / RAND\_MAX);

}

Wes2[i] = ((double)rand() / RAND\_MAX);

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

}

T[3] = ((double)rand() / RAND\_MAX);

do {

E = 0;

for (int q = 0; q < 500; q++) {

current = output(x, Wes1, Wes2, T);

reference\_value = func(x + 6 \* 0.1);

double Error = current - reference\_value;

double\* Hiddens = hidden(x, Wes1, T);

for (int j = 0; j < 2; j++)

Wes2[j] -= alpha \* Error \* Hiddens[j];

T[3] += alpha \* Error;

for (int k = 0; k < 2; k++) {

for (int i = 0; i < 6; i++) {

Wes1[k][i] -= alpha \* func(x + i \* 0.1) \* Hiddens[k] \* (1 - Hiddens[k]) \* Wes2[k] \* Error;

}

T[k] += alpha \* Hiddens[k] \* (1 - Hiddens[k]) \* Wes2[k] \* Error;

}

x += 0.1;

E += pow(Error, 2);

}

E /= 2;

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

stage = stage + 1;

} while (E > E\_min);

cout << " Количество эпох: " << stage << endl;

cout << "N" << "\t" << "Эталон" << setw(20) << "Получ.знач." << setw(15) << "Отклонение" << endl;

for (int i = 0; i < 15; i++) {

double result = output(x, Wes1, Wes2, T), ethelon\_value = func(x + 6 \* 0.1);

cout << i + 1 << "\t" << ethelon\_value << right << setw(15) << result << right << setw(15) << result - ethelon\_value << endl;

x += 0.1;

}

system("pause");

return 0;

}

Результаты тестирования программы:



Вывод: изучил обучение и функционирование нелинейной ИНС при решении задач прогнозирования.