Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №1

за 1 семестр

По дисциплине: «МиАПР»

Тема: «Линейная искусственная нейронная сеть. Правило Видроу-Хоффа»

Выполнила:

Студент 2 курса

Группы ПО-4(2)  
Маевский А.В.

Проверил:

Крощенко А.А.

2020Лабораторная работа №1

Линейная искусственная нейронная сеть. Правило Видроу-Хоффа

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

Вариант 5

Задание:

Написать на любом ЯВУ программу моделирования прогнозирующей линейной ИНС. Для тестирования использовать функцию

y = a\*sin(bx) + d

a = 1, b = 9, d = 0.5, кол-во входов ИНС = 4.

Обучение и прогнозирование производить на 30 и 15 значениях соответственно табулируя функцию с шагом 0.1. Скорость обучения выбирается студентом самостоятельно, для чего моделирование проводится несколько раз для разных . Результаты оцениваются по двум критериям - скорости обучения и минимальной достигнутой ошибке. Необходимо заметить, что эти критерии в общем случае являются взаимоисключающими, и оптимальные значения для каждого критерия достигаются при разных .

Код программы:

#include <iostream>

#include <Windows.h>

#include <iomanip>

#include <ctime>

#include <cmath>

using namespace std;

double function(int a, int b, double x, double d);

void print\_result(int n, double T, double\* Y, int Num\_Learning, int N, int n\_protected, double\* W);

int main() {

SetConsoleOutputCP(1251);

system("color f0");

srand(time(0));

int a = 1, b = 9, n = 4, Num\_Learning = 30, Num\_Predicted = 15;

double d = 0.5, step = 0.1, x = 0, E, Em = 0.001, T = 1;

double\* W = new double[n];

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

W[i] = 1.0 / (double)rand();

}

int N = Num\_Learning + Num\_Predicted;

double\* Y = new double[N];

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

x += step;

Y[i] = function(a, b, x, d);

}

double y1, //выходное значение нейронной сети

step\_learning = 0.3; //скорость обучения

do {

E = 0;

for (int i = 0; i < Num\_Learning - n; i++) {

y1 = 0;

for (int j = 0; j < n; j++) { //векторы выходной активности сети

y1 += W[j] \* Y[i + j];

}

y1 -= T;

for (int j = 0; j < n; j++) { //изменение весовых коэффициентов

W[j] -= step\_learning \* (y1 - Y[i + n]) \* Y[i + j];

}

T += step\_learning \* (y1 - Y[i + n]); //изменение порога нейронной сети

E += 0.5 \* pow(y1 - Y[i + n], 2); //расчет суммарной среднеквадратичной ошибки

}

} while (E > Em);

print\_result(n, T, Y, Num\_Learning, N, Num\_Predicted, W);

delete[] Y;

delete[] W;

return 0;

}

double function(int a, int b, double x, double d)

{

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

}

void print\_result(int n, double T, double\* Y, int Num\_Learning, int N, int Num\_Predicted, double\* W)

{

cout << "РЕЗУЛЬТАТЫ:" << endl;

cout << "1) Обучение:" << endl;

cout << setw(30) << left << "Эталонные значения" << setw(38) << left << "Полученные значения" << "Отклонение" << endl;

double\* predict = new double[N];

for (int i = 0; i < Num\_Learning; i++) {

predict[i] = 0;

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

predict[i] += W[j] \* Y[j + i];

}

predict[i] -= T;

cout << "y[" << i << "] = " << setw(30) << left << Y[i + n] << setw(30) << left << predict[i] << pow(Y[i + n] - predict[i], 2) << endl;

}

cout << "2) Прогнозирование:" << endl;

cout << setw(30) << left << "Эталонные значения" << setw(38) << left << "Полученные значения" << "Отклонение" << endl;

for (int i = 0; i < Num\_Predicted; i++) {

predict[i + Num\_Learning] = 0;

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

predict[i + Num\_Learning] += W[j] \* Y[i + j + Num\_Learning - n];

}

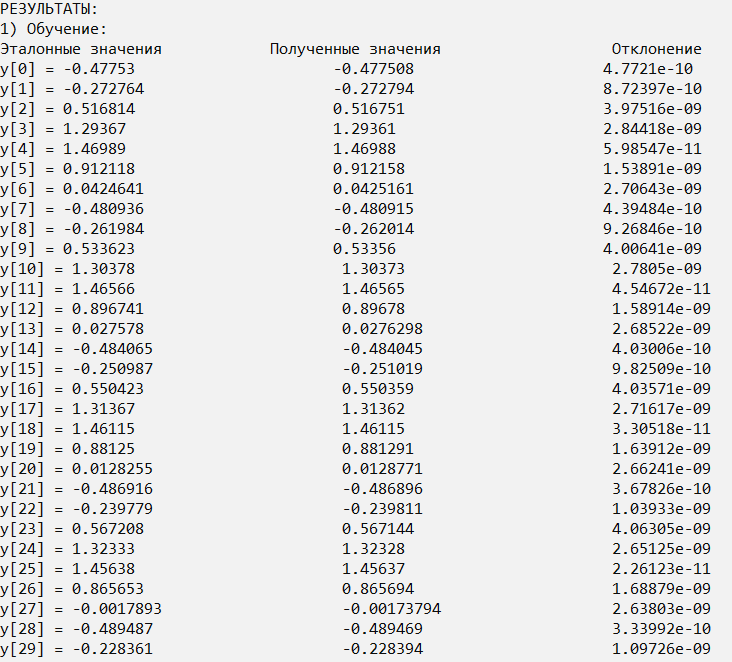
predict[i + Num\_Learning] -= T;

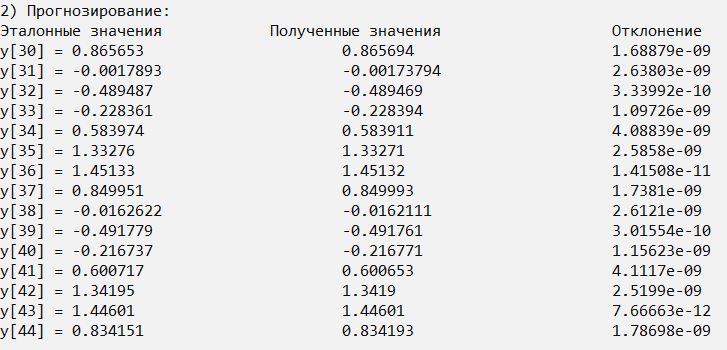
cout << "y[" << i + Num\_Learning << "] = " << setw(30) << left << Y[i + Num\_Learning] << setw(30) << left << predict[i + Num\_Learning] << pow(Y[i + Num\_Learning] - predict[i + Num\_Learning], 2) << endl;

}

delete[] predict;

}





Вывод: изучил обучение и функционирование линейной ИНС при решении задач прогнозирования