Міністерство освіти і науки України

Івано-Франківський національний технічний університет нафти і газу

Кафедра програмного забезпечення автоматизованих систем

***Лабораторна робота №6***

*на тему:*

***«Метод групового врахування аргументів»***

Виконав ст. гр. ПІ-13-2

Федишин А.Г.

Перевірив:

Дитко Т.В.

м.Івано-Франківськ

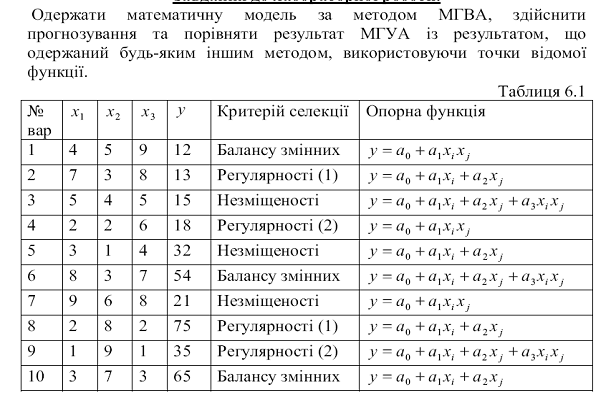
2015

**Тема:** Метод групового врахування аргументів.

**Мета** Навчитись визначати глобальні екстремуми складних функцій за допомогою генетичних алгоритмів.

**Завдання:**

Одержати математичну модель за методом МГВА, здйіснити прогнозування та порівняти результат МГВА з результатом, що одержати будь-яким іншим методом, використовуючи точки відомої функції.



**Короткі теоретичні відомості:**

Згідно принципу евристичної самоорганізації, математична модель оптимальної складності відповідає мінімуму де- якого критерію, що називають критерієм селекції. В якості такого критерію використовуються критерії регулярності і незміщеної моделі, або балансу змінних. Встановлено, що при поступовому підвищенні складності моделі критерії проходять через мінімальні значення. Комп'ютер знаходить глобальний мінімум і вказує на єдину модель оптимальної складності при дуже малому об’ємі апріорної інформації. Людина повинна лише вибрати критерій у відповідності до задачі. Об’єм повного перебору зменшується за допомогою алгоритмів багаторядної селекції, подібно то- му, як це практикують при селекції рослин чи тварин. Метод групового врахування аргументів (МГВА) реалізує ряд алгоритмів поступового ускладнення моделі за правилами багаторядної селекції. Автором методу е відомий український вчений Олексій Григорович Іваненко. МГВА застосовується для вирішення задач зменшення інформаційної невизначеності, а саме прогнозування, розпізнавання образів, автоматичної класифікації, ідентифікації характеристик, а також для автоматичного оптимального управління. Алгoритми МГВА відтворюють схему масової селекції. В них е генератори комбінацій, що ускладняються із ряду в ряд і порогові самовідбори найкращих з них. Якщо об'єкт описуються функцією:

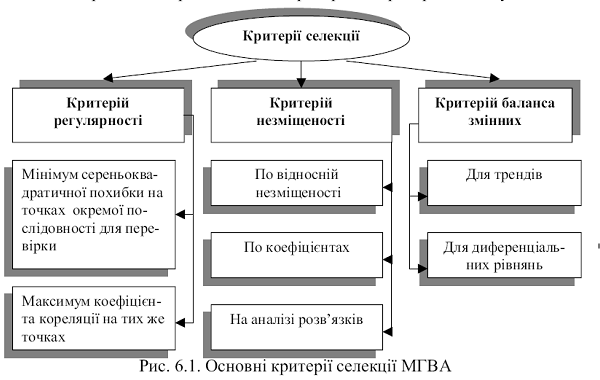
φ=f(х1, х2,....хm), яка, наприклад, є степеневим поліномом, то вона замінюється декількома рядами "частинних" описань:

1-й ряд селекції:

у1 = f(х1, х2), у2 = f(х1х3).....,yѕ = f(xm-1, xm)

2-й ряд селекції:

z1 = f(y1, y2), z2 = f(y1, y3).....,zp = f(ys-1, ys)



**Виконання роботи:**

**Код програми мовою С#:**

***Клас GMDH:***

Реалізація алгоритму МГВА

class GMDH {

Random rand = new Random();

private int inputsNum;

private int outputsNum;

private double learningRate = 0.96;

private double minAlpha = 0.01;

private double A = 0.023;

private double alpha = 0.6;

private double[] d;

private double[][] w;

public GMDH (int inputsNum, int outputsNum) {

this.inputsNum = inputsNum;

this.outputsNum = outputsNum;

w = new double[inputsNum][];

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

w[i] = new double[outputsNum];

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

w[i][j] = rand.NextDouble();

}

}

}

public void ComputeOutput (int[] vectorArray) {

d = new double[inputsNum];

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

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

d[i] += Math.Pow((w[i][j] - vectorArray[j]), 2);

}

}

}

public void Train (int[][] input) {

int dMin = 0;

while (alpha > minAlpha) {

for (int vecNum = 0; vecNum < inputsNum; vecNum++) {

ComputeOutput(input[vecNum]);

dMin = FindMin(d);

UpdateWeights(input[vecNum], dMin);

}

alpha = learningRate \* alpha;

}

}

private void UpdateWeights (int[] inputs, int dMin) {

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

w[dMin][i] = w[dMin][i] + (alpha \* (inputs[i] - w[dMin][i]));

if (alpha > A) {

if (dMin == 0) {

w[dMin + 1][i] = w[dMin + 1][i] + (alpha \* (inputs[i] - w[dMin + 1][i]));

} else if (dMin == inputsNum - 1) {

w[dMin - 1][i] = w[dMin - 1][i] + (alpha \* (inputs[i] - w[dMin - 1][i]));

} else {

w[dMin - 1][i] = w[dMin - 1][i] + (alpha \* (inputs[i] - w[dMin - 1][i]));

w[dMin + 1][i] = w[dMin + 1][i] + (alpha \* (inputs[i] - w[dMin + 1][i]));

}

}

}

}

private int FindMin (double[] array) {

int minIndex = 0;

for (int i = 0; i < array.Length; i++) {

if (array[i] < array[minIndex])

minIndex = i;

}

return minIndex;

}

public void Print (int[][] input) {

int dMin = 0;

Console.WriteLine("Testing:\n");

for (int vecNum = 0; vecNum < inputsNum; vecNum++) {

ComputeOutput(input[vecNum]);

dMin = FindMin(d);

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

dMin = FindMin(d);

}

}

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

Console.WriteLine("iteration#" + (i + 1));

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

Console.Write(w[i][j].ToString("F4") + ", ");

}

Console.WriteLine();

}

Console.WriteLine(Environment.NewLine + "Testing Result:");

for (int vecNum = 0; vecNum < (input[0].Length - 1); vecNum++) {

ComputeOutput(input[vecNum]);

dMin = FindMin(d);

Console.WriteLine("Expected coef А(" + vecNum + "): " + dMin);

}

}

}

***Клас Program:***

Точка входу в програму

static class Program {

[STAThread]

static void Main () {

int[][] input = new int[][] {

new int[] {0, 1, 1, 1, 0, 0, 0},

new int[] {0, 0, 1, 0, 1, 0, 1},

new int[] {1, 0, 1, 0, 1, 0, 0},

new int[] {0, 0, 0, 1, 0, 1, 1},

new int[] {1, 1, 0, 0, 1, 0, 0},

new int[] {0, 1, 1, 0, 1, 1, 0},

new int[] {1, 0, 0, 1, 0, 0, 1}

};

GMDH gmdh = new GMDH(input.Length, input[0].Length);

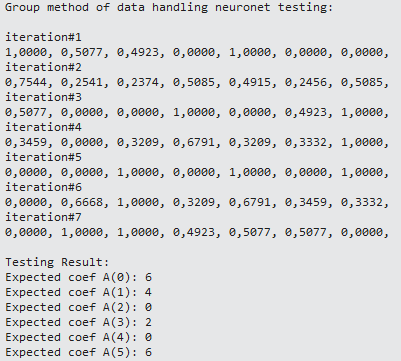
gmdh.Train(input);

gmdh.Print(input);

}

}

**Результат виконання програми:**



**Висновок:** На цій лабораторній роботі було розглянуто нейронні мережі на основі методу групового врахування аргументів та було розроблено програмний засіб, який реалізує даний метод.