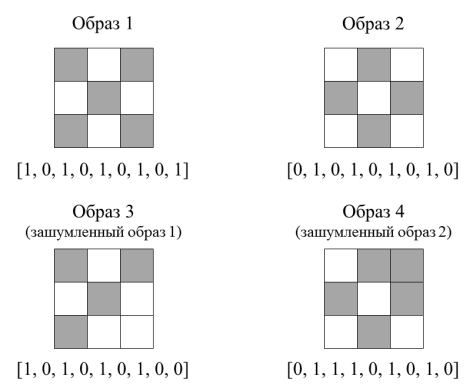
**Лущик Никита КС-44**

**Нейронная сеть адаптивного резонанса (АРТ-1)**

**Входные данные**



У нейронов по 9 входов, скорость адаптации – 0.6, критерий R – 0.7, лямбда – 2. В начале эпохи массив примеров перемешивается.

**Листинг Kohonen.py**

**import** Common

**from** random **import** shuffle

**def** ART1(inputArray, Rcr, lamda, V, epo):

*# region variables*

W = [] *# neuron*

T = [] *# neuron*

Y = []

R = []

X = inputArray[0]

WWinnerIndex = "No match"

*# endregion variables*

*# cluster init 1.0*

**def** init(X, W, T):

tmp0 = []

tmp1 = []

**for** i **in** range(len(X)):

tmp0.append((lamda\*X[i])/(lamda - 1 + sum(X))) *# 1.2*

tmp1.append(X[i])

W.append(tmp0.copy())

T.append(tmp1.copy())

**def** printInfo():

**print**("W:")

Common.FloatNumerizedPrint(W, 2)

**print**("**\n**T:")

Common.FloatNumerizedPrint(T, 2)

**print**("**\n**Y:")

Common.NumerizedPrint(Y)

**print**("**\n**R:")

Common.NumerizedPrint(R)

**print**("Match to cluster (WWinner):", WWinnerIndex)

init(X, W, T)

**print**("Iter 0: init")

printInfo()

*# cluster compare # 2.0*

**for** epoch **in** range(epo):

shuffle(inputArray)

**print**("**\n**Epoch", epoch, "**\n**----------------------")

**for** e **in** range(1, len(inputArray), 1):

X = inputArray[e] *# 2.1*

**print**("**\n**Iter", e, "**\n**X:", X)

temp = 0

**for** i **in** range(len(W)):

**for** j **in** range(len(W[0])):

temp += W[i][j]\*X[j]

Y.append(temp)

temp = 0

temp = 0

**for** i **in** range(len(T)): *# 3.1*

**for** j **in** range(len(T[0])):

temp += T[i][j]\*X[j] *# For all, its OK*

R.append(temp/sum(X))

temp = 0

winner = False

**if** (max(R) >= Rcr): *# 3.2*

WWinnerIndex = R.index(max(R))

**for** j **in** range(len(W[0])):

W[WWinnerIndex][j] = (1 - V)\*W[WWinnerIndex][j] + V\*(lamda\*X[j]) / (lamda - 1 + sum(X))

T[WWinnerIndex][j] = (1 - V)\*T[WWinnerIndex][j] + V\*X[j]

winner = True

**if**(winner == False):

init(X, W, T)

**print**("No winner, new cluster added")

printInfo()

Y.clear()

R.clear()

WWinnerIndex = "No match"

**def** main():

inputArray = [[1, 0, 1, 0, 1, 0, 1, 0, 1], *# 1*

[0, 1, 0, 1, 0, 1, 0, 1, 0], *# 2*

[1, 0, 1, 0, 1, 0, 1, 0, 0], *# 1 mod*

[0, 1, 1, 1, 0, 1, 0, 1, 0], *# 2 mod*

ART1(inputArray, 0.7, 2, 0.6, 10)

main()

**Листинг Common.py**

**def** NumerizedPrint(array, end = "**\n**"):

**for** i **in** range(len(array)):

**print**("%3d) %s" % (i, array[i]), end=end)

**def** FloatNumerizedPrint(array, roundsigns):

**for** i **in** range(len(array)):

**print**("%3d)" % i, end=" ")

**for** j **in** range(len(array[0])):

**print**(round(array[i][j], roundsigns), end=" ")

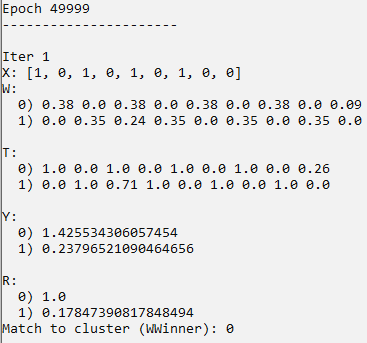
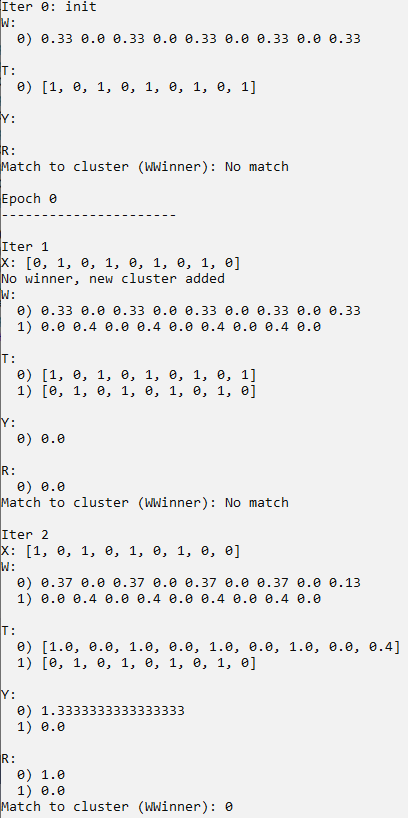
**print**("**\n**", end="")

**Значения коэффициентов с последней итерации**

****

****

**Фрагмент вывода программы**

**Вывод**

В результате перебора всех примеров обучающей выборки в нейронной сети самоорганизуются 2 кластера, различающиеся составами примеров. Первый кластер включает пример 1 и 3(зашумленный 1), второй – 2 и 4(зашумленный 2). Сеть точно классифицирует примеры, на которых обучена, и продолжает обучаться при подаче на входы новых примеров.