Лабораторная работа №4.

Построение нейронной сети Кохонена средствами языка программирования высокого уровня

Цель работы: *Научиться программировать и обучать нейронную сеть.*

Задание: *Создать нейронную сеть со структурой "слой Кохонена", и обучить ее распознаванию нижеприведенных массивов (приложение А).*

Ход работы:

1. Разработать структурную схему нейронной сети, способной выполнить классификацию данных согласно заданию на лабораторную работу.
2. Запрограммировать нейронную сеть, обучаемую по методу выпуклой комбинации (см. приложение Б).
3. Обучить нейронную сеть на приведенных в приложении А примерах.
4. Опробовать работоспособность нейронной сети на произвольно измененном примере.
5. Оформить отчет в электронном виде (топология сети; входные данные; полученные выходные данные; измененный пример; выходные данные с измененным примером).

Приложение А.

Класс 1.
- 1,2,3,4,5,6,7,8,9
- 1,3,3,4,5,6,7,8,9
- 1,2,3,4,5,6,7,9,9
- 2,2,3,4,5,6,7,8,9

Класс 2.
- 9,8,7,6,5,4,3,2,1
- 9,8,7,6,5,4,3,2,2
- 8,8,7,6,5,4,3,2,1
- 9,8,7,5,5,4,3,2,1

Класс 3.
- 1,2,1,2,1,2,1,2,1
- 1,3,1,2,1,2,1,2,1
- 1,2,2,2,1,2,1,2,1
- 1,2,1,2,1,2,2,2,2

Класс 4.
- 4,4,4,4,2,2,2,2,1
- 4,5,4,4,2,3,2,2,1
- 4,4,4,3,2,2,2,1,1
- 4,4,4,4,2,3,3,2,1

Класс 5.
- 1,2,3,1,2,3,1,2,3
- 1,2,4,1,2,3,1,2,3
- 1,2,3,2,2,3,1,2,3
- 1,2,3,1,2,3,3,2,3

Приложение Б.

Теоретические сведения:

Структура сети.

Сеть Кохонена содержит единственный слой нейронов, имеющих линейную функцию активации. Каждый из нейронов связан отдельными весами с каждым из входов сети. Таким образом, структура слоя Кохонена аналогична структуре однослойного перцептрона; все различия обусловлены алгоритмами функционирования и обучения сети.
В отличие от перцептрона, выходы нейронов слоя Кохонена влияют на выходы других нейронов. В своей простейшей форме слой Кохонена функционирует в духе "победитель забирает все", т. е. для данного входного вектора один и только один нейрон Кохонена выдает на выходе логическую единицу, все остальные выдают ноль. Такой режим работы сети называется режимом аккредитации. Если требуется только распознавание входных образов, т.е. отнесение их к одному из известных образов, сеть Кохонена, работающая в режиме аккредитации, может быть построена следующим образом:

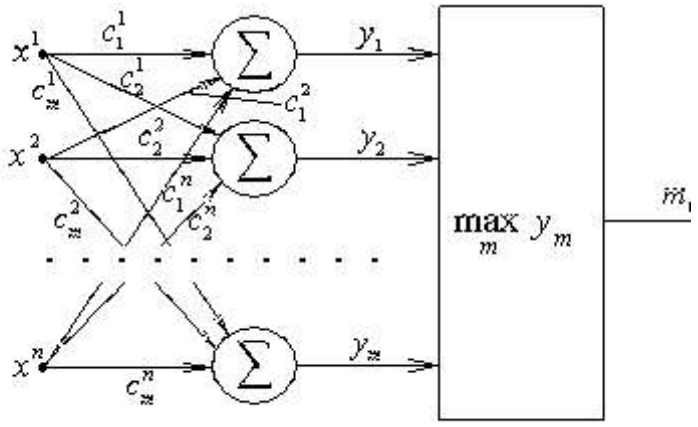

Рис. 1 Структура сети Кохонена

На рис. 1 выходной блок - интерпретатор - устройство или программа, формирующая на выходе номер нейрона, имеющего максимальное выходное значение, а нейроны - обычные сумматоры (функция активации не используется).
Как в режиме функционирования, так и в режиме обучения входные векторы должны быть нормированы к одинаковым значениям длины, во избежание нарушения работы сети.
Функционирование сети.
Каждый нейрон принимает все значения входного вектора и формирует на выходе их взвешенную сумму:

$$y_i = \sum_{j=1}^{N} c_i^j x^j, \; i = \overline{1,\mathrm{M}}, \qquad (1),$$

где *N,M* - количество входов и нейронов соответственно, c_i^j - вес от *j*-го входа к *i*-му нейрону.
Нейрон, имеющий максимальное выходное значение y_i, считается "победителем". Его выходное значение устанавливается равным единице. Выходные значения всех остальных нейронов устанавливаются в ноль. В реализации, показанной на рис. 1, на выход выдается только номер выигравшего нейрона m_0.

Обучение сети.

Задача обучения - научить сеть активировать один и тот же нейрон для похожих векторов X^k на входе. Не важно, какой конкретно нейрон будет активирован.

Присвоение начальных значений. Обычно начальные значения в нейронных сетях выбираются малыми случайными числами. Для слоя Кохонена такой выбор возможен, но имеет недостатки. Разумеется, если ядра классов нормированы, то и начальные значения нужно нормировать.
Если веса инициализируются случайными значениями с равномерным распределением, то возникает

проблема. Когда ядра распределяются равномерно, то в областях пространства *X*, где входных векторов мало, ядра будут использоваться редко, т.к. будет мало похожих векторов. В тех областях, где входных векторов много, плотность ядер окажется недостаточной, и непохожие объекты будут активировать один и тот же нейрон, т.к. более похожего ядра не найдется. Для устранения проблемы можно выделять ядра в соответствии с плотностью входных векторов. Но распределение входных векторов часто бывает заранее неизвестно. В этом случае помогает метод выпуклой комбинации, рассмотренный далее.

Обучение сети. Веса сети настраиваются при помощи следующего итеративного алгоритма.

1.Присвоить начальные значения весовым коэффициентам.

2.Подать на вход один из векторов X^k .

3.Рассчитать вектор выходов слоя Кохонена $\{y^{m,k}\}$, и определить номер выигравшего нейрона , выход которого максимален:

$$m_0 = \max_m y^{m,k} \quad (2)$$

4.Скорректировать веса только выигравшего нейрона :

$$c^i_{m_0} = c^i_{m_0} + \alpha\left(x^{i,k} - c^i_{m_0}\right),\ i = \overline{1,N}. \quad (3)$$

где α - скорость обучения, малая положительная величина. Веса корректируются так, что вектор весов приближается к текущему входному вектору. Скорость обучения управляет быстротой приближения ядра класса (вектора весов) к входному вектору.

5.Перейти к пункту 2.

Алгоритм выполняется до тех пор, пока веса не перестанут меняться.

Метод выпуклой комбинации. Этот метод позволяет правильно распределить плотность ядер классов в соответствии с плотностью входных векторов в пространстве X.
На шаге 1, в отличие от описанного выше алгоритма, всем весам сети присваиваются одинаковые начальные значения:

$$c^j_i = \frac{1}{\sqrt{N}},\ i = \overline{1,M},\ j = \overline{1,N} \quad (4)$$

Значение $1/\sqrt{N}$ используется для приведения длин всех весовых векторов сети к единице.

Обучение сети множеству векторов $\{X^k\}$ далее проводится последовательным выполнением шагов 2-5, но используются не векторы X^k , а специально сформированные векторы:

$$x^{k,j\,\prime} = x^{k,j}\beta(t) + \frac{1-\beta(t)}{\sqrt{N}},\ j = \overline{1,N} \quad (5)$$

где $\beta(t)$ - коэффициент, изменяющийся от 0 до 1 по мере обучения. В начале обучения $\beta(t)=0$, и все обучающие векторы совпадают с начальным значением вестовых векторов сети. По мере обучения $\beta(t)$ увеличивается, и обучающие векторы расходятся из начальной точки на гиперсфере к своим конечным значениям X^k . Каждый нейрон захватывает один или несколько входных векторов. Таким образом, в процессе обучения все нейроны распределяются правильно, и в сети не остается "ненужных" необученных нейронов.
Обычная сеть Кохонена работает в режиме аккредитации, когда ненулевое значение на выходе формирует только один нейрон. Если остальные нейроны не затормаживать, а нормировать вектор

выходов сети к единице, например, при помощи функции активации *softmax*:

$$OUT_j = \frac{e^{NET_j}}{\sum_{i=1}^{M} e^{NET_i}} = \frac{e^{y_j}}{\sum_{i=1}^{M} e^{y_i}}, \quad (6)$$

то выходы сети можно интерпретировать, как вероятность отнесения входного вектора к каждому из классов. Такой режим работы сети называется режимом интерполяции. Название режима говорит о том, что если входной вектор X плавно изменяется от одного весового вектора C_i к другому вектору C_j, то выход сети в режиме интерполяции будет плавно меняться от решения i к решению j.