ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

ВЯТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ АВТОМАТИКИ И ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ
КАФЕДРА ЭЛЕКТРОННЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИН

Методические рекомендации по выполнению лабораторной работы №8

Тема: «РАЗРАБОТКА И ИССЛЕДОВАНИЕ НЕЙРОСЕТЕВЫХ ЭКСПЕРТНЫХ СИСТЕМ НА БАЗЕ СЕТИ ART2 И СЕТИ ВСТРЕЧНОГО РАСПРОСТРАНЕНИЯ»

Спец. 230101, 5 курс

Киров 2010

УДК 519. 322

Методические указания для студентов очного отделения специальности 230101.65 по выполнению лабораторных работ по курсу «Системы искусственного интеллекта» / Вятский государственный университет. Киров, 2009, 41 с./

Составитель к.т.н., доцент кафедры ЭВМ В.С. Ростовцев

Лабораторная работа №8

Целью выполнения лабораторной работы является разработка и исследование нейросетевой экспертной системы (ЭС) на базе нейронной сети АРТ2 и нейронной сети встречного распространения.

Лабораторная работа выполняется в два этапа:

1 этап. Создание нейросетевой ЭС на базе сети встречного распространения

2 этап. Создание нейросетевой ЭС на базе сети АРТ2.

1.1 Этап 1. Создание нейросетевой ЭС на базе сети встречного распространения

1.1.1 Вариант задания согласовать с преподавателем и ознакомиться с принципом работы нейронной сети встречного распространения. Ознакомиться с теорией нейронной сети встречного распространения, приведенной в приложении А.

В папке «Задания» выбрать номер варианта и исследовать многослойную нейронную сеть с использованием коммерческой программы NeuroPro 0.25, загрузив проект с расширением *.npp. Выбрать оптимальную архитектуру НС, проверить на наличие противоречивых примеров (Меню «Нейросеть» команда «Анализ обучающего множества».

1.1.2 Создать обучающую выборку ЭС в программе Excel и сохранить в формате *.csv. Подготовить не менее семи тестовых примеров, по одному на кластер.

1.1.3 Запустить программу cpn.exe, открыть созданную обучающую выборку, установить параметры сети и щелкнуть по кнопке ОК.

1.1.4 Выполнить обучение, а затем тестирование нейросетевой ЭС загрузив поочередно семь тестовых примеров.

1.1.5 Повторить п. 1.1.3 и 1.1.4 изменяя параметры сети встречного распространения. Выбрать оптимальные параметры сети встречного распространения.

1.2 Этап 2. Создание нейросетевой ЭС на базе сети АРТ2

1.2.1 Ознакомиться с теорией нейронной сети встречного распространения, приведенной в приложении Б.

Обучающую выборку для нейронной сети АРТ2 использовать с первого этапа. Подготовить не менее семи тестовых примеров, по одному на кластер.

1.2.2 Скопировать обучающую выборку ЭС в программу БЛОКНОТ и сохранить в формате *.TXT. Подготовить не менее семи тестовых примеров, по одному на кластер. Пример оформления приведён на рис.1.

15

9

0.85 0.85 0.20 0.65 0.80 0.75 0.70 0.80 0.75

0.40 0.40 0.90 0.85 0.95 0.50 0.80 0.95 0.50

0.90 0.85 0.40 0.90 0.65 0.50 0.85 0.80 0.50

0.30 0.60 0.60 0.80 0.95 0.45 0.85 0.85 0.90

0.80 0.95 0.35 0.65 0.40 0.90 0.20 0.95 0.95

0.15 0.45 0.30 0.80 0.50 0.35 0.85 0.85 0.35

0.70 0.80 0.30 0.80 0.85 0.35 0.55 0.80 0.50

0.65 0.20 0.25 0.55 0.60 0.50 0.85 0.85 0.40

0.50 0.25 0.60 0.65 0.30 0.70 0.65 0.50 0.20

0.10 0.15 0.70 0.90 0.85 0.30 0.80 0.25 0.30

0.20 0.25 0.60 0.85 0.55 0.30 0.25 0.95 0.35

0.70 0.75 0.30 0.50 0.45 0.35 0.65 0.40 0.25

0.85 0.45 0.35 0.60 0.40 0.35 0.10 0.15 0.35

0.90 0.60 0.40 0.60 0.70 0.50 0.95 0.55 0.45

0.40 0.35 0.60 0.65 0.80 0.40 0.40 0.45 0.30

Рисунок 1- Пример обучающей выборки из 9 полей и 15 записей

1.2.3. Запустить программу ArtNet. Для создания нового проекта ART-сети необходимо выбрать пункт меню *ПРОЕКТ/СОЗДАТЬ*. Далее необходимо ввести параметры нейросети и загрузить обучающую выборку:

- Степень схожести - параметр, определяющий, насколько похож должен быть входной вектор на образец класса, чтобы нейросеть его распознала. Число из промежутка (0,1). Предпочтительно использовать 0,65.
- Скорость обучения - параметр, определяющий, насколько быстро будет изменяться образец класса под влиянием входных векторов, отнесенных к этому классу. Число из промежутка от 0 до 1.
- Число нейронов сети - параметр, определяющий максимально допустимое количество распознаваемых классов. Целое число из промежутка (0; 50].

1.2.4 Для загрузки файл обучающей выборки для нейросети, нажать кнопку «Открыть ОВ». Файл с обучающей выборкой должен находиться в пределах той же папки, что и программа ArtNet.

1.2.5 Созданный проект может быть сохранен на диске при помощи команд меню *ПРОЕКТ/СОХРАНИТЬ, ПРОЕКТ/СОХРАНИТЬ КАК*.

В дальнейшем можно работать с сохраненными файлами проекта, выбрав пункт меню *ПРОЕКТ/ОТКРЫТЬ*.

1.2.6 После создания проекта или изменения его параметров программа сама предложит обучить ART-сети. Вручную эту операцию можно запустить,

выбрав пункт меню *НЕЙРОСЕТЬ/ОБУЧЕНИЕ*. При необходимости можно прервать операцию, выбрав пункт меню Выполнение/Прервать.

Образцы распознанных классов отобразятся в таблице «Классы».

1.2.7 Результаты обучения также можно посмотреть в протоколе обучения, содержащем параметры обучаемой нейросети, обучающую выборку и класс, к которому отнесен каждый вектор обучающей выборки. При желании, данный протокол (как и протоколы обучения и классификации) можно сохранить, воспользовавшись пунктом меню Нейросеть/Сохранить протокол.

1.2.8 **Тестирование сети.** Данная операция доступна только после того, как нейросеть обучена. Для выполнения операции необходимо выбрать пункт меню *НЕЙРОСЕТЬ/ТЕСТИРОВАНИЕ*. Тестирование предусматривает ввод вектора пользователем вручную либо выбор файла, содержащего вектора для тестирования. При этом вектора, не отнесенные ни к одному из классов можно сохранить в качестве образца нового класса, поставив флажок «Сохранять новый вектор». Чтобы ввести вектор вручную, необходимо выбрать в списке «Примеры для тестирования» пункт «вручную» и нажать кнопку «Задать». В появившейся таблице ввести вектор и нажать кнопку «Начать».

1.2.9 Чтобы провести тестирование на векторах из файла, необходимо выбрать в списке *«ПРИМЕРЫ ДЛЯ ТЕСТИРОВАНИЯ»* пункт *«ИЗ ФАЙЛА»* и нажать кнопку *«ВЫБРАТЬ».* После выбора файла появится таблица с векторами из файла. Для начала тестирования нажать кнопку *«НАЧАТЬ»*. При необходимости можно прервать операцию, выбрав пункт меню *ВЫПОЛНЕНИЕ/ПРЕРВАТЬ*. Результаты тестирования отображаются в протоколе тестирования.

1.2.10 **Настройка проекта.** Выбранные при создании проекта параметры можно изменить, выбрав пункт *НЕЙРОСЕТЬ/НАСТРОЙКА*. Изменить можно следующие параметры нейросети: степень схожести, скорость обучения, количество резервных нейронов. После выполнения данной операции обязательно переобучение нейросети. В противном случае при сохранении

проекта информация о классах сохранена не будет, тестирование также будут недоступно.

1.2.11 Повторить п. 1.2.6 -1.2.10 с изменёнными параметрами сети и выбрать оптимальные параметры для решения задачи нейросетевой ЭС.

1.2.12 Сравнить способы реализации нейросетевой ЭС на базе нейронной сети АРТ2, нейронной сети встречного распространения и многослойной нейронной сети (NeuroPro 0.25), используя семь заранее созданных тестовых наборов.

2 Результаты выполнения лабораторной работы оформить в отчёт.

Приложение А

Нейронная сеть встречного распространения

1.1 Описание нейронной сети встречного распространения

Многие алгоритмы обучения нейронных сетей (НС) характерны тем, что необходимым участником процесса обучения является «учитель», который заранее производит классификацию используемых при обучении объектов и результаты этой классификации вводит в машину. Однако в биологических системах такой механизм обучения трудно себе представить. Способность разных людей одинаково проводить классификацию «без учителя» заставляет думать, что различия, на которых основывается такая способность, имеют объективный и безусловный характер.

Для реализации этих алгоритмов на практике необходимо решить два вопроса. Первый связан с выбором способа разбиения объектов на классы без учителя, второй — с выработкой правила отнесения входного образа к определенному классу. Необходимо описать способ проведения границы между множествами объектов и сформулировать алгоритм ее построения.

Процесс разбиения некоторого множества объектов на классы называется *кластеризацией.*

Примером НС, использующей алгоритм обучения без учителя, является введенная Т. Кохоненом (1982) «самоорганизующаяся карта признаков» (Self-Organizing Feature Maps, SOM). Другое, часто используемое название сети Кохонена, — KCN (Kohonen Clustering Networks). KCN используют для отображения нелинейных взаимосвязей данных на достаточно легко интерпретируемые (чаще всего двумерные) сетки, представляющие метрические и топологические зависимости входных векторов, объединяемых в кластеры.

Однако большей популярностью пользуются гибридные НС, которые представляют собой объединение различного рода сетей и концепций их обучения. Одной из широко используемых комбинированных сетей, использующихся на практике, является двухслойная сеть встречного распространения (CPN — *Counterpropagation Network*), первым слоем которой является самоорганизующаяся сеть Кохонена, а вторым — выходная звезда С. Гроссберга. Сети данного вида успешно применяют в финансовых и экономических приложениях, таких, как рассмотрение заявок на предоставление займов, предсказание трендов цен акций, товаров и курсов обменов валют. Можно ожидать успешного применения CPN-сети в задачах интерполяции.

1.2.1 Нейронные сети Кохонена

Данные сети позволяют в результате обучения осуществлять топологически непрерывное отображение F входного n-мерного пространства в выходное m-мерное пространство; т.е. $F{:}R^n \to R^m$. При этом обучение здесь происходит «без учителя» на основе образов поступающих на сеть. Для обучения самоорганизующихся нейронных сетей используется *конкурентный метод*, который был предложен в 1976 году С. Гроссбергом (S. Grossberg) и затем развит в работах финского ученого Т. Кохонена (Т.

Kohonen) [1]. По мере поступления входных образов на такую сеть посредством обучения происходит разбиение *n*-мерного входного пространства на различные области решений, каждой из которых соответствует отдельный нейрон. Границы отдельной области перпендикулярны линиям, проведенным между центроидами соседних областей решений. Такое разделение пространства называется диаграммой Вороного (Voronoi) или картами Кохонена. Для двухмерного случая ($n = 2, m > n$) область решений представляет собой правильные шестиугольники (рисунок 1.1), в результате чего получается наименьшая ошибка. Для $n > 2$ наилучшая форма областей решений является неизвестной [2].

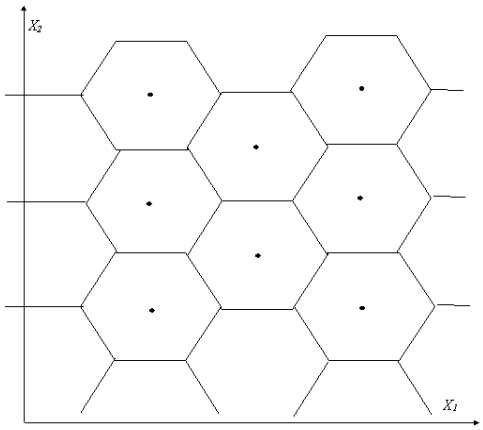


Рисунок 1.1 — Разбиение входного пространства образов

Таким образом, самоорганизация таких сетей происходит в результате топологического упорядочивания входной информации по различным кластерам, количество которых является *m*.

Топологическое упорядочивание информации напоминает процессы происходящие в головном мозге при его развитии (введение), когда осуществляется формирование топологически упорядоченных нейронных структур.

1.2.2 Конкурентное обучение

Конкурентное обучение (*competitive learning*) является основным

методом для обучения нейронных сетей Кохонена. Конкурентная нейронная сеть в общем случае представляет собой двухслойную нейронную сеть с прямыми связями (рисунок 1.2). Первый слой выполняет чисто распределительные функции, причем каждый нейрон его имеет соединения со всеми нейронными элементами выходного слоя. Во втором слое происходит конкуренция между нейронными элементами, в результате которой определяется нейрон-победитель.

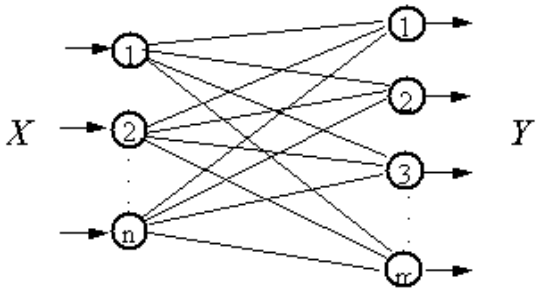


Рисунок 1.2 — Топология конкурентной нейронной сети

Для *нейрона-победителя* синаптические связи усиливаются, а для остальных нейронов не изменяются или могут уменьшаться. Победителем в конкуренции является нейрон, который в результате подачи на вход сети определенного образа имеет максимальную взвешенную активность

$$S_j = \sum_i w_{ij} x_i = W_j \, X^T \tag{1}$$

где $X = \{x_1, x_2, \ldots, x_n\}$ — входной образ, $W_j = \{w_{1j}, w_{2j}, \ldots, w_{nj}\}$ — вектор столбец весовых коэффициентов *j*-го выходного нейрона. Пусть

$$S_k = \max_j S_j .$$

Тогда активность выходных нейронов определяется по формуле (2).

$$y_j = F(S_j) = \begin{cases} 1, & \text{åñëè } j = k \\ 0, & \text{åñëè } j \neq k \end{cases} \tag{2}$$

где $j = \overline{1, m}.$

Таким образом, после обучения нейронной сети при подаче входного образа активность нейрона-победителя принимается равной единице, а остальных нейронов нулю. Это правило известно под названием *«победитель*

берет все» (winner take all). Выражение (1) эквивалентно скалярному произведению вектора весов соответствующего нейронного элемента на входной вектор нейронной сети

$$S_j = \left|W_j\right| \cdot \left|X\right| \cdot \cos\alpha, \tag{3}$$

где $\alpha = \hat{W_j X}$, $\left|W_j\right|$ и $\left|X\right|$ — модули векторов W_j и X.

Обозначим $P = \left|X\right| \cdot \cos\alpha$, где P — проекция вектора X на вектор W. Тогда

$$S_j = \left|W_j\right| \cdot P \tag{4}$$

Если векторы $\left|W_j\right|$ и $\left|X\right|$ не нормированы, то происходит неадекватное определение нейрона победителя (рисунок 1.3). Как следует из рисунка, нейрон, вектор весов которого W_2 больше отличается от входного образа X чем нейрон, имеющий весовой вектор W_1, становится победителем.

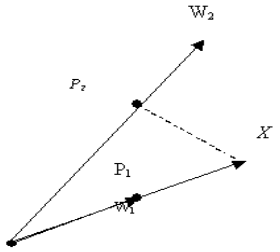


Рисунок 1.3 — Геометрическая интерпретация определения нейрона победителя при ненормированном весовом векторе и входном образе

Поэтому, при определении нейрона-победителя по взвешенной активности (3), необходимо нормализовать весовые и входные векторы для каждого нейрона. Нормализация осуществляется следующим образом:

$$\left|X\right| = \sqrt{\sum_i x_i^2} = 1, \tag{5}$$

$$\left|W_j\right| = \sqrt{\sum_i w_{ij}^2} = 1. \tag{6}$$

Тогда взвешенную активность *j*-го нейрона можно представить как

$$S_j = \left|W_j\right| \cdot \left|X\right| \cdot \cos\alpha = \cos\alpha\,. \qquad (7)$$

Из этого выражения следует, что максимальную активность будет иметь тот нейрон, весовой вектор которого коллинеарен входному вектору. Концы векторов при этом находятся на поверхности *n*-мерной сферы (гиперсферы), радиус которой равен 1.

В выражении (7) взвешенная сумма эквивалентна коэффициенту взаимной корреляции между входным и весовым вектором. Он будет равен 1, когда угол между векторами равен нулю. Отсюда следует, что правило настройки весовых коэффициентов нейрона-победителя должно соответствовать вращению вектора W_k в сторону вектора X .

В результате можно записать правило обучения для вектора весов нейрона-победителя (8).

$$W_k(t+1) = W_k(t) + \gamma(X - W_k(t))\,, \qquad (8)$$

где $0 < \gamma < 1$ характеризует скорость обучения.

В качестве нейрона победителя выбирается такой нейрон, весовой вектор которого наиболее близок к входному вектору. В обычной форме правило обучения для *k*-го нейрона-победителя можно представить, как

$$w_{ik}(t+1) = w_{ik}(t) + \gamma(x_i - w_{ik}(t)),$$

где $i = \overline{1,n}.$

При применении данного правила к *k*-му нейрону усиливается его выходная активность (рисунок 1.4).

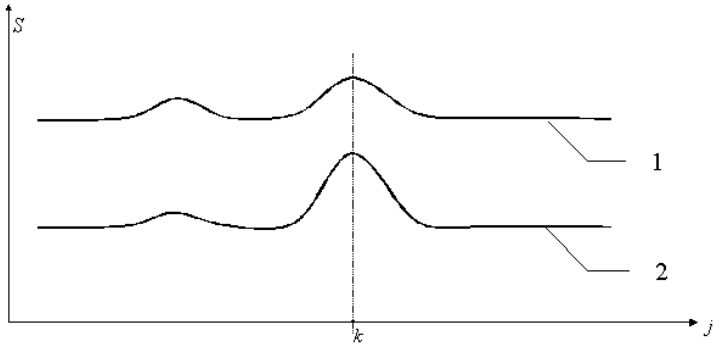


Рисунок 1.4 — Профили активности нейронов.

1 — профиль активности нейронов до обучения; 2 — после обучения.

Так как весовой вектор должен быть нормированным, то правило (8) изменения весового вектора модифицируется по выражению (9).

$$W_k(t+1) = \frac{W_k(t) + \gamma(X(t) - W_k(t))}{\left(W_k(t) + \gamma(X(t) - W_k(t)\right)}. \quad (9)$$

При применении этого правила для обучения нейронной сети весовые векторы нейронов будут вращаться в направлении кластеров входных образов, как показано на рисунке 1.5.

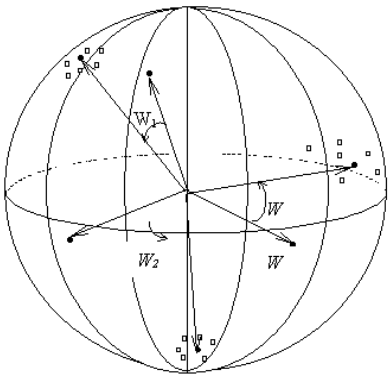


Рисунок 1.5 — Изменение весовых векторов в процессе обучения:
p — вектор образцов; w — вектор весов.

В случае использования ненормализованных векторов для определения нейрона победителя нужно оперировать вместо взвешенной активности (1) евклидовым расстоянием.

$$D_j = \left|X - W_j\right| = \sqrt{(x_1 - w_{1j})^2 + (x_2 - w_{2j})^2 + ... + (x_n - w_{nj})^2}. \quad (10)$$

При помощи (10) определяется нейрон победитель с номером *k*, который соответствует минимальному евклидовому расстоянию между входным и весовым вектором.

$$D_k = \min_j \left|X - W_j\right|. \quad (11)$$

Тогда настройка весового вектора нейрона-победителя происходит по формуле (12).

$$W_k(t+1) = W_k(t) + \gamma(X(t) - W_k(t)) \quad (12)$$

При использовании выражения (12) для обучения одного нейронного элемента на интервале входных значений [a,b] (x$\in$ [a,b]) вес нейрона с течением времени стремиться к середине интервала.

Недостатком описанного выше метода обучения является то, что при случайной инициализации весовых векторов, может получиться так, что некоторые нейроны никогда не будут победителями. Для нейтрализации этого недостатка можно расширить правило обучения по формуле (13).

$$\begin{aligned} W_j(t+1) &= W_j(t) + \gamma \cdot (X - W_j(t)), \forall j = k \\ W_j(t+1) &= W_j(t) + \gamma' \cdot (X - W_j(t)), \forall j \neq k \end{aligned}, \quad (13)$$

где $\gamma' << \gamma$. Данное правило позволяет отобразить весовые векторы побежденных нейронов в такую область, где увеличиваются их шансы в конкуренции. Другим вариантом является частотно-чувствительное конкурентное обучение (sensitive competitive learning). Здесь для каждого нейрона ведется статистика его побед. Пусть f_j — частота нахождения *j*-го нейрона в состоянии победителя. Тогда нейрон победитель определяется по формуле (14).

$$D_k = \min_j \left| X - W_j \right| f_j . \quad (14)$$

Чем чаще нейрон становится победителем, тем меньше шансов он имеет в конкуренции.

Итак, при конкурентном обучении все множество входных образов разбивается на кластеры, каждому из которых соответствует свой нейрон. При поступлении на вход НС неизвестного образа, она будет его относить к такому кластеру, на который он больше всего похож. В этом заключается обобщающая способность такого типа НС. В 1978 г. С. Гроссберг доказал сходимость конкурентных методов обучения [3].

1.2.3 Самоорганизующиеся карты Кохонена

Самоорганизующиеся карты Кохонена являются дальнейшим расширением нейронных сетей с конкурентным обучением. Они были разработаны Кохоненом в 1982 году. Топология НС Кохонена состоит из двух слоев. Первый слой выполняет распределительные функции, а нейроны второго слоя расположены на плоскости образуя матрицу (рисунок 1.6).

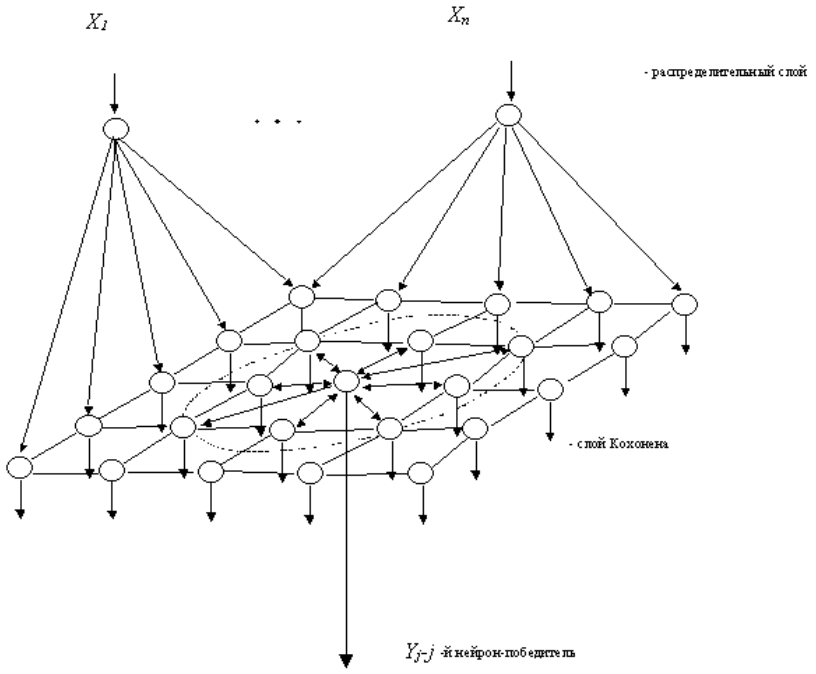


Рисунок 1.6 — Топология нейронной сети Кохонена

В своем простейшем виде сеть Кохонена функционирует по принципу «победитель берет все». При этом она должна выполнять топологически упорядоченное отображение входных векторов на матрицу нейронов второго слоя. Соседние наиболее похожие входные образы должны отображаться на соседние нейроны матрицы. Это достигается путем введения области притяжения G для нейрона победителя, в радиусе действия которой нейроны активно изменяют свои весовые векторы в сторону входного образа. Область притяжения можно описать функцией притяжения, которую обозначим $h(t,k,p)$. Здесь t — время, k — номер нейрона-победителя, p — номер искомого нейрона. В дискретном варианте функция притяжения определяется следующим образом:

$$h(t,k,p)=\begin{cases}1, & \text{åñëè } p\in G\\ 0, & \text{åñëè p}\notin \text{G}\end{cases} \qquad (15)$$

В область притяжения нейрона k входят все нейроны, находящиеся на определенном расстоянии от нейрона победителя. В непрерывном варианте часто используется функция Гаусса (рисунок 1.7).

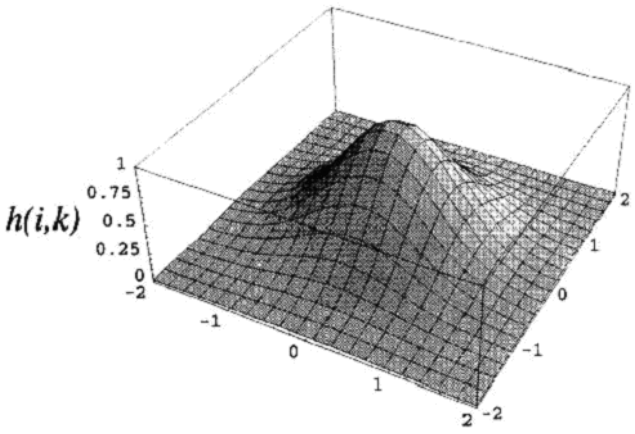


Рисунок 1.7 — Двухмерная функция Гаусса

Она определяется при помощи выражения (16).

$$h(p,k,t) = e^{\frac{-|u_k - u_p|^2}{2\sigma^2(t)}}, \tag{16}$$

где $(u_k - u_p)$ — расстояние между нейронами, $\sigma(t)$ — среднеквадратичное отклонение (радиус области притяжения). Положение каждого нейрона в матрице характеризуется его координатами.

$$u_k = (i_k, j_k),\ u_p = (i_p, j_p). \tag{17}$$

Тогда

$$|u_k - u_p|^2 = (i_k - i_p)^2 + (j_k - j_p)^2. \tag{18}$$

В процессе обучения нейронной сети Кохонена изменяются весовые коэффициенты не только нейрона победителя, но и всех нейронов внутри области притяжения. Так, для *p*-ого нейрона весовой вектор изменяется по закону (19).

$$W_p(t+1) = W_p(t) + \gamma(t)h(t,k,p)(X(t) - W_p(t)). \tag{19}$$

С увеличением времени обучения радиус области притяжения уменьшается. В результате нейронные элементы сжимаются около нейрона победителя, пока он не останется один. Это изображено на рисунке 1.8, где $G(t)$ — область притяжения в момент времени *t*. Введем декартову систему координат, так, что каждый нейрон имеет координаты (i, j), где $i = \overline{1,m};\ j = \overline{1,m}$.

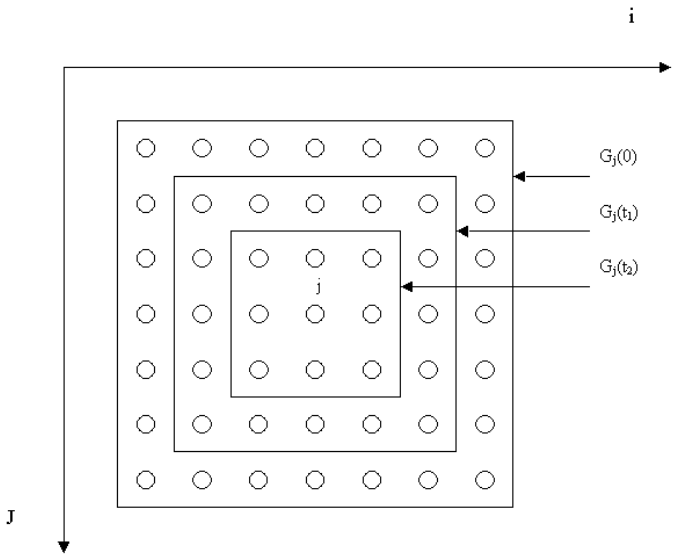


Рисунок 1.8 — Изменение области притяжения с течением времени

Поставим в соответствие нейрону с координатами (i, j) весовой вектор W_{ij}, который определяется по формуле (20).

$$W_{ij} = (w_{1ij}, w_{2ij}, ..., w_{nij}), \tag{20}$$

где n — количество нейронных элементов входного слоя.

Тогда процедуру обучения сети Кохонена для непрерывной функции притяжения можно представить в виде следующей последовательности действий [4]:

1. Случайно инициализируются весовые коэффициенты W нейронной сети.

2. Задается начальное значение радиуса притяжения s и момент времени $t = 1$.

3. Подается входной образ $X^l (l = 1)$ на нейронную сеть и вычисляется норма вектора: $D_{ij} = (X^l - W_{ij})$, где $i = \overline{1,m}, j = \overline{1,m}$; W_{ij} — весовой вектор нейрона с координатами (i, j).

4. Определяется нейрон победитель: $D(k_1, k_2) = \min\limits_{i,j} D_{ij}$, где $k = (k_1, k_2)$ — координаты нейрона победителя.

5. Для каждого нейрона производится вычисление функции притяжения.

$$h(p,k,t) = e^{\frac{-|u_k - u_p|^2}{2\sigma^2(t)}},$$

где $p = \overline{1,m^2}$.

6. В соответствии с функцией притяжения осуществляется модификация весовых коэффициентов: $W_{ij}(t+1) = W_{ij}(t) + \gamma(t)h(p,k,t)(X^l(t) - W_{ij}(t))$, где $\gamma(t) = \frac{1}{t}$.

7. Переходим к пункту 3 и повторяем процедуру для $l = 2,3,..,L,$ где L-общее количество входных образов.

8. Увеличиваем на единицу квант времени, уменьшаем радиус области притяжения s и повторяем процесс, начиная с пункта 3.

Обучение производится до получения желаемой степени согласования между весовыми и выходными векторами. Начальное значение области притяжения s может охватывать всю матрицу нейронов, а затем последовательно уменьшается, как показано на рисунке 1.8. Для дискретной функции притяжения процедура обучения является аналогичной. Функционирование такой сети происходит путем определения нейрона победителя и присвоения ему единичного значения, а остальным нейронам нулевого значения.

В процессе обучения происходит упорядочивание весовых коэффициентов таким образом, что уменьшается разница между весами соседних нейронов. Покажем это. Пусть весовые векторы соседних нейронов изменяется как

$$\begin{aligned} W_1(t+1) &= W_1(t) + \gamma(x(t) - W_1(t)) = (1-\gamma)W_1(t) + \gamma X(t) \\ W_2(t+1) &= W_2(t) + \gamma(x(t) - W_2(t)) = (1-\gamma)W_2(t) + \gamma X(t) \end{aligned}.$$

Расстояние между ними равно

$$\left|W_1(t+1) - W_2(t+1)\right| = \left|(1-\gamma)(W_1(t) - W_2(t))\right|.$$

Так как $0 < \gamma < 1$, то

$$\left|W_1(t+1)-W_2(t+1)\right| < \left|W_1(t)-W_2(t)\right|.$$

Рисунок 1.9 — Изменение весовых коэффициентов сети Кохонена

Таким образом, в процессе обучения разница между весовыми векторами топологически близких нейронов уменьшается. Это показано на рисунке 1.9 для сети с двумя входными нейронами и 8×8 выходными нейронами. Линии соединяют значения весов нейрона с координатами (i, j) с весами нейрона $(i+1, j)$ и $(i, j+1)$. В начале $(t=0)$ веса выбраны случайно и беспорядочно распределены на плоскости. Затем веса изменяются, так что их плотность приблизительно соответствует плотности вероятности входных векторов.

1.2.4 Сети встречного распространения

Сеть со встречным распространением *CPN* представляет собой соединение самоорганизующейся сети Кохонена и выходной звезды Гроссберга. Топология сети встречного распространения приведена на рисунке 1.10. В рамках этой архитектуры элементы сети Кохонена не являются выходом сети, а служат лишь входами для выходного слоя — выходной звезды Гроссберга. Создатель сети встречного распространения Р. Хехт-Нильсен *(R. HehtNilsen,* 1987) рекомендует использовать эти архитектуры для решения задач аппроксимации функций и заполнения пробелов в таблице данных.

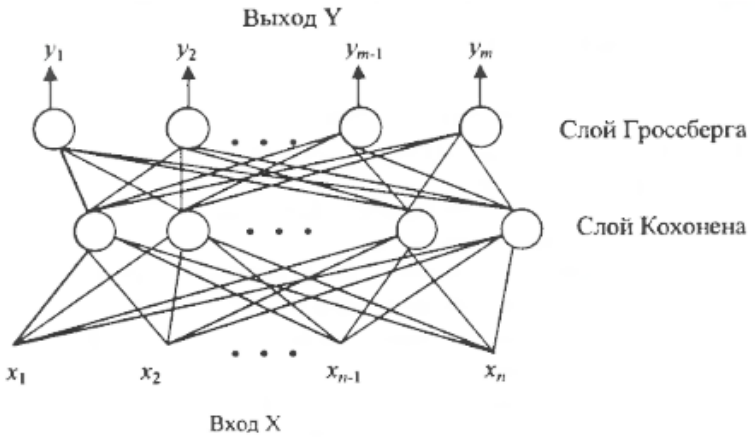


Рисунок 1.10 — Топология сети встречного распространения.

Идеи С. Гроссберга, связанные с исследованиями в области биологической кибернетики, реализуются во многих НС. В частности, конфигурации входных и выходных звезд Гроссберга используют для создания иерархических НС.

Нейрон в форме входной звезды имеет и входов, которым соответствуют весовые коэффициенты $\mathbf{W}=(w_1, w_2, ..., w_n)$ и один выход $\mathbf{Y}$, являющийся взвешенной суммой входов. Входная звезда обучается выдавать сигнал на выходе всякий раз, когда на входы поступает определенный вектор. Таким образом, входная звезда является своего рода детектором состояния входов и реагирует только на свой входной вектор. Подстройка весовых коэффициентов проводится по следующей формуле:

$$\mathbf{W}_i(t+1) = \mathbf{W}_i(t) + \mu\left(\mathbf{X}_i - \mathbf{W}_i(t)\right)$$

где $\mathbf{W}_i(t)$ — весовой вектор *i*-й входной звезды на *t*-м такте обучения; $\mathbf{X}_i$ — входной вектор; μ — скорость обучения. Скорость обучения μ имеет начальное значение в пределах 0,1…0,2 и постепенно уменьшается в процессе обучения.

Выходная звезда Гроссберга выполняет противоположную функцию — при поступлении сигнала на вход выдается определенный вектор. Нейрон этого типа имеет один вход и m выходов с весами $\mathbf{W}=(w_1, w_2, ..., w_m)$, которые подстраиваются в соответствии с формулой

$$\mathbf{W}_i(t+1) = \mathbf{W}_i(t) + \alpha\left(\mathbf{Y}_i - \mathbf{W}_i(t)\right) \qquad (21)$$

где $\mathbf{W}_i(t)$ — весовой вектор *i*-й выходной звезды на *t*-м такте обучения; $\mathbf{Y}_i$ — выходной вектор; α — скорость обучения. Рекомендуется начать обучение со значения α в пределах единицы, постепенно уменьшая до значений, близких к нулю. Итерационный процесс будет сходиться к некоторому усредненному образу, полученному из совокупности обучающих векторов.

Особенностью нейронов в форме звезд Гроссберга является избирательность памяти. Каждый нейрон в форме входной звезды помнит «свой» относящийся к нему образ и игнорирует остальные. Каждой выходной звезде соответствует некоторая конкретная командная функция. Образ памяти связывается с определенным нейроном, а не возникает вследствие взаимодействия между нейронами в сети.

Обучение *CPN* состоит из двух шагов. На первом шаге весовые векторы слоя Кохонена настраиваются таким образом, чтобы провести распределение входных векторов по классам, каждый из которых соответствует одному нейрону-победителю. Обучение проводится без учителя. Точность кластеризации в этом случае будет гарантирована только тогда, когда обучающая выборка является представительной.

На втором шаге осуществляется обучение с учителем. Проводится подстройка весовых коэффициентов выходного слоя Гроссберга на примерах с заданным выходом с использованием формулы (21). При этом настраиваются только веса, соответствующие связям с теми элементами слоя Кохонена, которые являются «победителями» в текущем такте обучения (выигравшие элементы посылают выходной сигнал, равный единице). Темпы обучения нейронов Кохонена и Гроссберга должны быть согласованы; кроме того, в слое Кохонена подстраиваются также веса всех нейронов в окрестности победителя, которая постепенно сужается до одного нейрона.

При функционировании сети в режиме распознавания нейроны слоя Гроссберга по сигналу нейрона-победителя в слое Кохонена воспроизводят на выходах сети образ в соответствии со значениями его весовых

коэффициентов. В случае, когда слой Гроссберга состоит из единственного элемента, получающийся скалярный выход равен одному из весов, соответствующих связям этого элемента (с выигравшим нейроном).

1.2.5 Алгоритм обучения CPN-сети

Окончательно алгоритм обучения сети встречного распространения можно сформулировать следующим образом:

1. Случайно инициализируются весовые коэффициенты W всех слоев.

2. Задается начальное значение радиуса притяжения s_0, скорость обучения выходной звезды Гроссберга $\alpha_0 \in (0{,}1; 0{,}2)$, максимальное число итераций N, минимальная ошибка E и момент времени $t = 1$.

3. Подается входной образ $X^l (l = 1)$ на вход нейронной сети и вычисляется норма вектора: $D_{ij} = (X^l - W_{ij})$, где $i = \overline{1,m},\ j = \overline{1,m};\ W_{ij}$ — весовой вектор нейрона с координатами (i, j).

4. Определяется нейрон победитель: $D(k_1, k_2) = \min\limits_{i,j} D_{ij}$, где $k = (k_1, k_2)$ — координаты нейрона победителя.

5. Для каждого нейрона производится вычисление функции притяжения $h(p,k,t) = e^{\frac{-|u_k - u_p|^2}{2\sigma^2(t)}}$, где $p = \overline{1,m^2}$.

6. В соответствии с функцией притяжения осуществляется модификация весовых коэффициентов слоя Кохонена по формуле $W_{ij}(t+1) = W_{ij}(t) + \beta h(p,k,t)(X^l(t) - W_{ij}(t))$, где β — скорость обучения сети Кохонена, изменяющаяся по закону $\beta = 1/t$.

7. Осуществляется модификация весовых коэффициентов слоя Гроссберга по формуле $W_{ij}(t+1) = W_{ij}(t) + \alpha\left(Y^l(t) - W_{ij}(t)\right)$

8. Переходим к пункту 3 и повторяем процедуру для $l = 2,3,..,L,$ где L — общее количество входных образов.

9. Рассчитываем усредненное расстояние между выходными и

весовыми векторами по формуле $d_t = \frac{1}{L}\sum_{l=1}^{L}\sqrt{\sum_{i=1}^{M}\left(Y_i - Y_i^*\right)^2}$, где Y — реальный выход НС, Y^* — желаемый выход НС, M — число выходов НС.

10. Рассчитать ошибку $e_t = \frac{d_{t-1} - d_t}{d_{t-1}}$.

11. Проверяем условия $t \geq N$, $e_t \leq E$. Если выполнено хотя бы одно из условий, переходим к шагу 13. Иначе — к шагу 12.

12. Увеличиваем на единицу квант времени, уменьшаем радиус области притяжения s по формуле $s = s - s_0 / N$, скорость обучения $\alpha = \alpha - \alpha_0 / N$ и повторяем процесс, начиная с пункта 3.

13. Конец.

1.2 Интерфейс главного окна

При запуске программы открывается главное окно, представленное на рисунке 3.1:

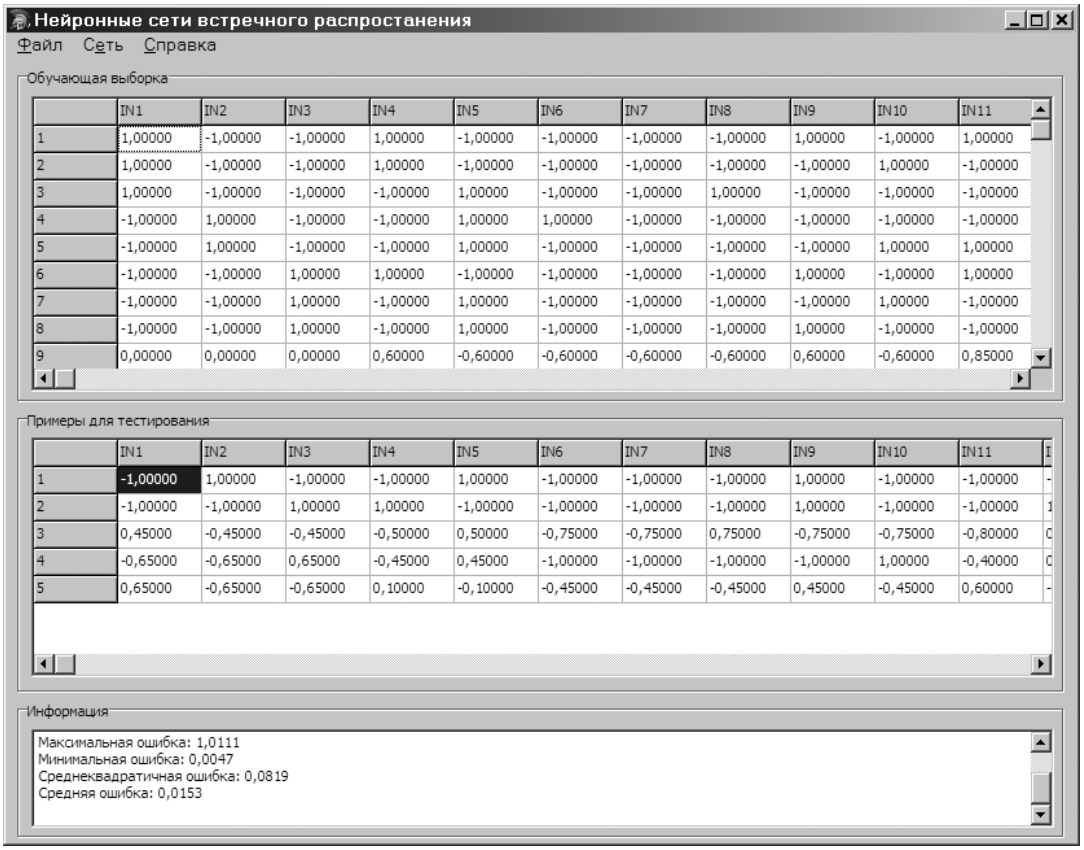


Рисунок 3.1 — Главное окно программы.

На главном окне программы располагаются:

- таблица с обучающей выборкой;
- таблица с примерами для тестирования (прогнозирования);
- поле с дополнительной информацией о сети. Эту информацию можно сохранить в текстовый файл, выбрав в контекстном меню пункт «Сохранить…»;
- главное меню.

Справка по работе с программой вызывается по пункту «Справка» меню «Справка». При выборе пункта «Новая сеть…» меню «Файл» пользователю будет предложено выбрать файл в формате CVS(вариант DOS) с данными. Такой файл данных можно создать средствами любого

табличного процессора, например, Microsoft Excel, OpenOffice.org Calc (рисунок 3.2). Чтобы указать программе, что данная строка предназначена для тестирования, а не для обучения, достаточно удалить значения всех выходов в соответствующей строке.

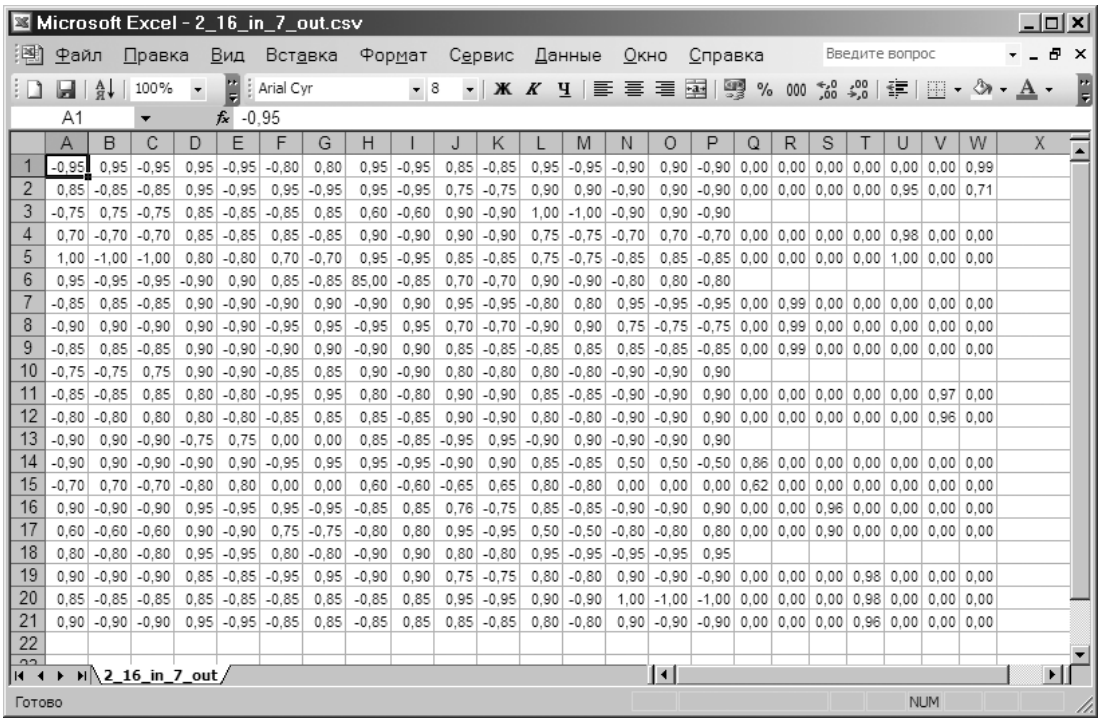


Рисунок 3.2 — Окно программы Microsoft Excel с данными.

1.3.2 Интерфейс вспомогательных окон

После выбора пользователем нужного файла, откроется окно «Свойства нейронной сети», представленное на рисунке 3.3:

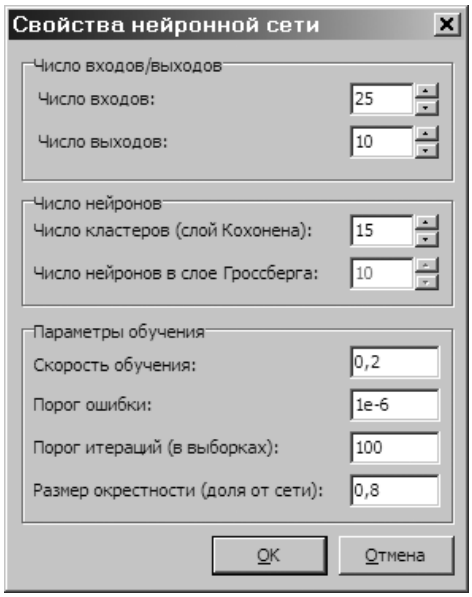


Рисунок 3.3 — Окно свойств НС.

Далее пользователь должен задать необходимые параметры будущей НС (число входов/выходов (оно должно совпадать с числом входов и выходов, содержащихся в файле с данными), число нейронов и параметры обучения). После нажатия на кнопку «ОК» будет создана НС встречного распространения выбранной конфигурации и считан файл с данными.

При выборе пункта «Обучить…» меню «Сеть» откроется следующее окно:

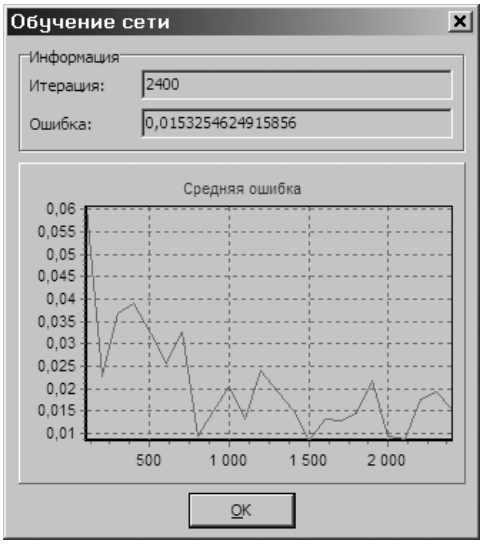
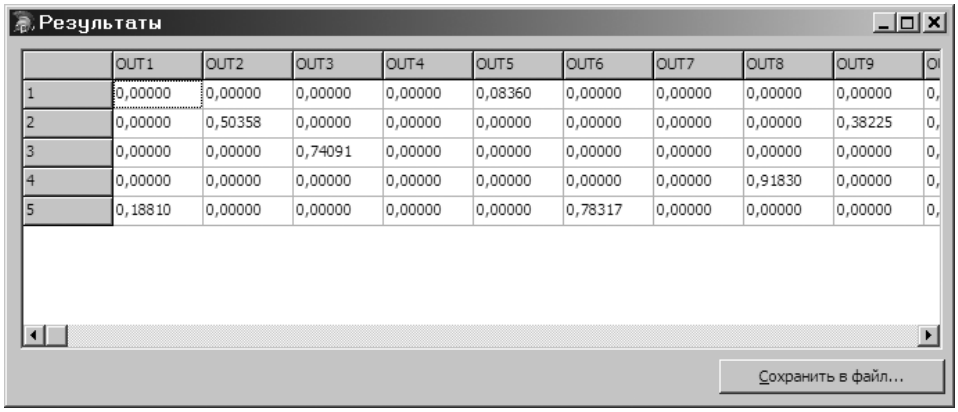


Рисунок 3.4 — Окно «Обучение сети».

При нажатии на кнопку «Обучить» начнется процесс обучения НС, при этом будет обновляться информация о номере текущей итерации, текущей средней ошибке и график ее изменения. Процесс обучения можно остановить, нажав на кнопку «Прервать».

При выборе пункта «Тестирование…» меню «Сеть» откроется окно, представленное на рисунке 3.5:



	OUT1	OUT2	OUT3	OUT4	OUT5	OUT6	OUT7	OUT8	OUT9	O
1	0,00000	0,00000	0,00000	0,00000	0,08360	0,00000	0,00000	0,00000	0,00000	0,
2	0,00000	0,50358	0,00000	0,00000	0,00000	0,00000	0,00000	0,00000	0,38225	0,
3	0,00000	0,00000	0,74091	0,00000	0,00000	0,00000	0,00000	0,00000	0,00000	0,
4	0,00000	0,00000	0,00000	0,00000	0,00000	0,00000	0,00000	0,91830	0,00000	0,
5	0,18810	0,00000	0,00000	0,00000	0,00000	0,78317	0,00000	0,00000	0,00000	0,

Рисунок 3.5 — Результаты тестирования.

В данном окне расположена таблица с данными, полученными программой. Их можно сохранить в CSV-файл, нажав на кнопку «Сохранить в файл…».

При выборе пункта «Структура сети…» меню «Сеть» откроется окно, представленное на рисунке 3.6:

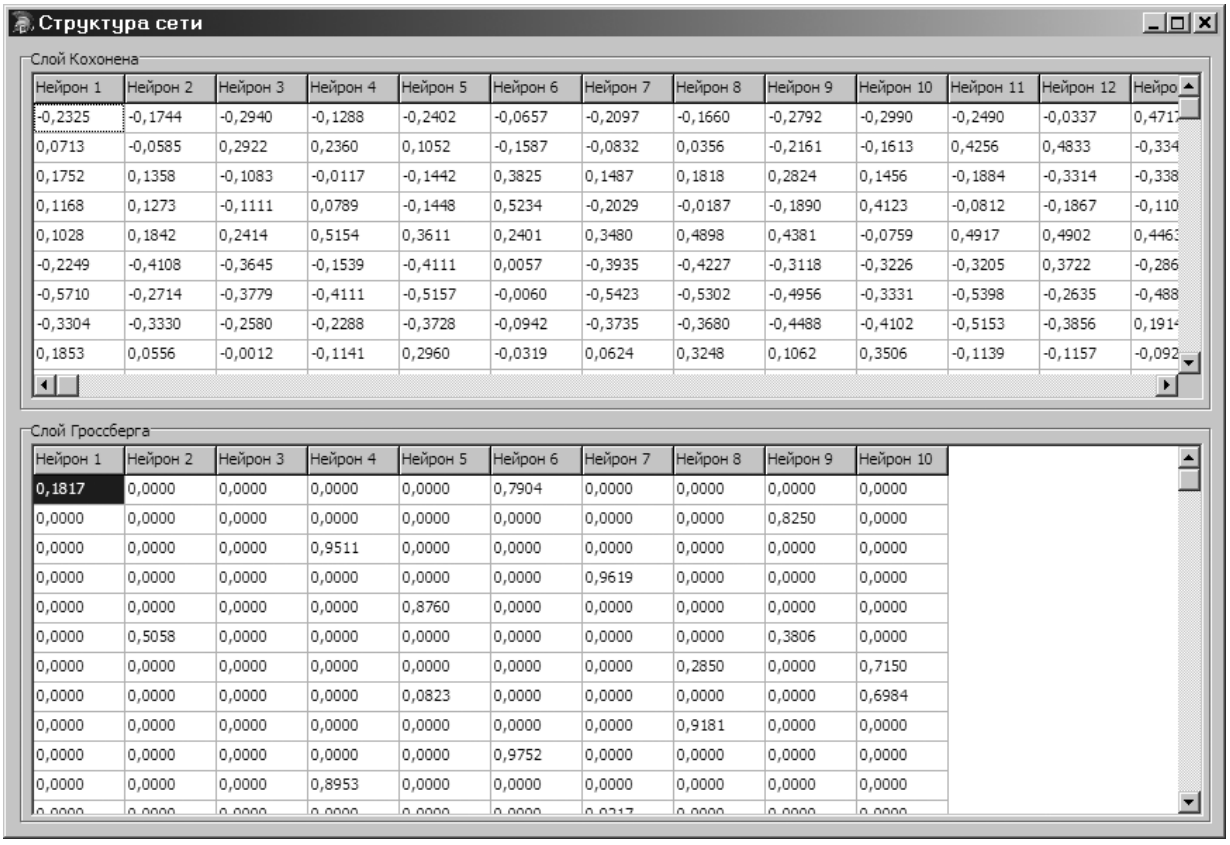


Рисунок 3.6 — Окно структуры НС.

В окне расположены две таблицы, содержащие веса синапсов всех нейронов со слоев Кохонена и Гроссберга.

При выборе пункта «Параметры сети…» меню «Сеть» откроется окно, показанное ранее на рисунке 3.3. Здесь пользователь может изменить параметры уже созданной НС, после чего ее необходимо переобучить.

2.4 Порядок работы с программой

1. В любом табличном процессоре (Microsoft Excel, OpenOffice.org Calc) подготовить файл с исходными данными. При этом должны быть

удалены те выходы, которые будут прогнозироваться с помощью программы.

2. Сохранить данный файл в формате CSV.
3. Запустить cpn.exe
4. В меню «Файл» выбрать пункт «Новая сеть…» и указать путь к созданному файлу с данными.
5. В появившемся окне «Свойства нейронной сети» задать необходимые параметры:
   а) число входов и выходов сети;
   б) число кластеров, на которые будет разбиваться входные образы;
   в) скорость обучения;
   г) порог ошибки (минимальное значение средней ошибки, при достижении которого прерывается процесс обучения сети);
   д) максимальное число итераций обучения;
   е) размер окрестности выигравшего нейрона (в долях от числа кластеров).

   Нажать на кнопку «ОК».
6. После этого, если входной файл не содержал ошибок, будут заполнены таблицы «Обучающая выборка» и «Примеры для тестирования».
7. В меню «Сеть» выбрать пункт «Обучить».
8. В появившемся окне «Обучение сети» нажать на кнопку «Начать». Процесс обучения можно остановить, нажав на кнопку «Прервать». В процессе обучения выводится номер текущей итерации, величина средней ошибки и строится график, показывающий изменение средней ошибки в процессе обучения. По завершении обучения нажать на кнопку «ОК».
9. После обучения в поле «Информация» главного окна будут отображены все данные о сети. Эти данные можно сохранить в файл, выбрав в контекстном меню поля пункт «Сохранить…».

10. Дополнительно структуру сети (веса синапсов всех нейронов всех слоев) можно посмотреть, выбрав пункт «Структура сети…» меню «Сеть».
11. Чтобы запустить протестировать сеть, необходимо выбрать пункт «Тестирование» меню «Сеть». В появившемся окне «Результаты» будет отображена таблицы со значениями, предсказанными сетью. Результаты можно сохранить в файл, нажав на соответствующую кнопку.
12. Параметры уже созданной НС можно изменить, выбрав пункт «Параметры…» меню «Сеть».

Контрольные вопросы

1. Принцип обучения нейронной сети Кохонена.
2. Принцип обучения нейронной сети выходная звезда Гроссберга.
3. Диаграмма Воронова.
4. Самоорганизующиеся карты Кохонена.
5. Структура сети встречного распространения.
6. Области применения Кохонена и сети встречного распространения.
7. Алгоритм конкурентного обучения.
8. Частотно-чувствительное конкурентное обучение (sensitive competitive learning).
9. Алгоритм обучения CPN-сети

Библиографический список

1. Осовский С. Нейронные сети для обработки информации/ пер. с польского И.Д. Рудинского.-М.: Финансы и статистика, 2002.-344с.
2. Головко В. А. Нейронные сети: обучение, организация и применение. Кн. 4: Учеб. Пособие для вузов М.: ИПРЖР, 2ОО1.-256с
3. Нейронные сети: история развития теории. Кн. 5: Учеб. пособие для вузов. / Под общей ред. А.И. Галушкина, ЯЗ. Цыпкина. - М.: ИПРЖР, 2001. - 840 с: ил. (Нейрокомпьютеры и их применение).

Приложение Б

Нейронная сеть адаптивной резонансной теории

1.1 Принципы функционирования сети APT-2

1.1.1 Алгоритм обучения/работы сети ART–2

Сети теории адаптивного резонанса ART2 по своим классификационным признакам, особенностям обучения, структуры и функционирования схожи с сетями ART–1. Отличие заключается лишь в том, что обрабатываемые многомерные вектора являются не бинарными, а вещественными. Вследствие этого входной слой сети представляет собой совокупность шести подслоев, каждый из которых включает по m нейронов, где m – размерность входного вектора (Рисунок 2). Назначение указанных подслоев заключается в нормализации входного вектора и приведения его к виду близкому к векторам, обрабатываемым сетью ART1.
Опишем алгоритм обучения/работы сети ART2.
Входными данными сети являются последовательно подаваемые вещественные вектора произвольной размерности m: X = (x1, x2, … , xm).
Параметрами сети являются:

- порог близости r – действительное число из интервала (0; 1), при этом чем ближе оно к 1, тем требование близости является более строгим;
- a, b, c – произвольные положительные числовые коэффициенты, выполняющие нормализующую функцию (возможные значения: 10, 10, 0.1, соответственно);
- e – малое действительное положительное число, применяемое для предотвращения деления на ноль (обычно 0.001);
- q – малое действительное положительное число (обычно 0.001), служащее критерием отсечения шума с использованием следующего соотношения:

$$h(x)=\begin{cases}0, 0 \le |x| \le \theta \\ x, |x| > \theta\end{cases}$$

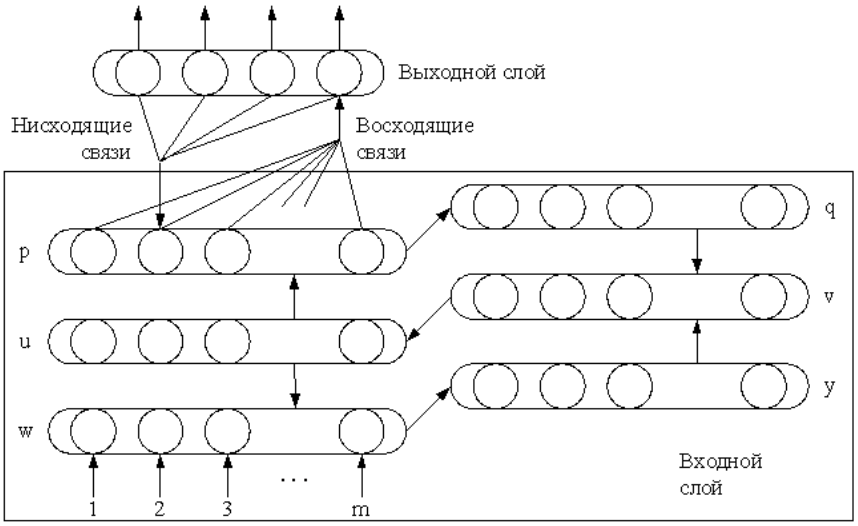


Рисунок 2 – Нейронная сеть ART2

Введем обозначения:

- tij – вес нисходящей связи от j-го нейрона в выходном слое к i-му нейрону во входном слое (подслой p);
- bij – вес восходящей связи от i-го нейрона во входном слое (подслой p) к j-му нейрону во выходном слое;
- pi, ui, wi, qi, vi, yi – состояние нейрона i в соответствующем подслое ($i = \overline{1,m}$).

В начале работы сеть включает по m нейронов в каждом подслое входного слоя и n=1 нейрон выходного слоя. Веса связей инициализируются следующими значениями:

$$t_{ij}(0) = 0; \quad b_{ij}(0) = \frac{1}{1+m}; \quad i = \overline{1,m}; \; j = \overline{1,n}$$

Состояния всех нейронов в подслоях входного слоя являются нулевыми.

При поступлении очередного вектора X = (x1, x2, … , xm) выполняются следующие операции:

1. Все нейроны выходного слоя делаются активными.
2. Последовательно вычисляются состояния нейронов в подслоях:

$$w_i = x_i + a \cdot u_i; \quad y_i = \frac{w_i}{\varepsilon + \|W\|};$$

$$v_i = h(y_i) + b \cdot h(q_i); \quad u_i = \frac{v_i}{\varepsilon + \|V\|};$$

$$p_i = u_i; \quad q_i = \frac{p_i}{\varepsilon + \|P\|}; \quad i = \overline{1,m},$$

здесь $\|Z\|$ означает длину вектора Z:

$$\|Z\| = \sum_{i=1}^{m} z_i^2 .$$

3. Для всех активных нейронов вычисляется значение функции активации:

$$f_j(X) = \sum_{i=1}^{m} b_{ij} \cdot p_i, \quad j = \overline{1,n}$$

4. Из всех активных нейронов выявляется нейрон-победитель:

$$\frac{\sum_{i=1}^{m} t_{iN} \cdot x_i}{\sum_{i=1}^{m} x_i} \geq \rho$$

Если активных нейронов больше нет, то генерируется новый выходной нейрон с номером n+1 (появляется новый кластер), со следующими связями:

$$t^{n+1} = b^{1,n+1} \quad b^{n+1} = \frac{1}{1 + \|b\|}, \quad i = \overline{1,m}; \quad n = n + 1$$

,

после чего переход на п. 9.

5. Пересчет состояний нейронов подслоя p:

$$p_i = u_i + f_N(X) \cdot t_{iN}, \quad i = \overline{1,m}$$

6. Нейрон-победитель проходит проверку на близость:

$$r_i = \frac{u_i + c \cdot p_i}{\varepsilon + \|U\| + c \cdot \|P\|}; \quad i = \overline{1,m}; \quad \frac{\rho}{\varepsilon + \|R\|} \leq 1$$

Если условие выполняется, то переход на п. 8, иначе – на п. 7.

7. Нейрон победитель делается неактивным, переход на п. 4.

8. Корректируются веса связей нейрона-победителя, прошедшего проверку на близость:

$$t_{iN} = b_{iN} = \frac{u_i}{1 - f_N(X)}; \quad i = \overline{1,m}$$

9. Работа с очередным вектором закончена.

Результатом работы сети, как и в случае с ART1, является либо номер нейрона-победителя, прошедшего проверку на близость, т.е. факт распознавания вектора и его отнесения к уже известному классу (кластеру) объектов, либо сообщение об отсутствии кластера для данного вектора и создании такого кластера.

1.3.2 APT-2A

Другой популярный APT алгоритм назван APT 2A . Он использует угол между вектором прототипа класса и входным вектором, чтобы найти подходящий класс. Рисунок 3 иллюстрирует соотношения для двумерных входных векторов.

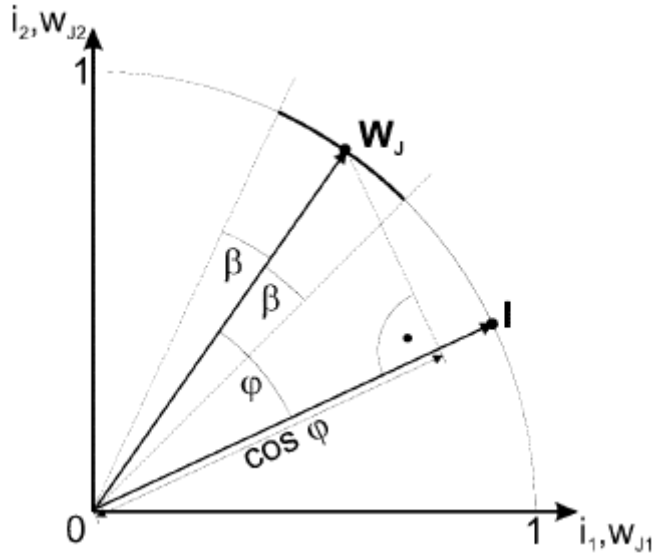


Рисунок 3 – Соотношения для двумерных входных векторов в АРТ-2А

Основные функции АРТ 2А-алгоритма:

1. Предобработка

Входные значения длжны быть неотрицательными и нормализованными к единице Евклидовой длины, обозначенной символом функции N

$$\mathbf{I} = \aleph(\mathbf{A}) = \frac{\mathbf{A}}{\sqrt{\sum_{i=1}^{m} a_i^2}} = \frac{\mathbf{A}}{\|\mathbf{A}\|} \quad a_i \geq 0 \quad \forall i \quad \|\mathbf{A}\| > 0.$$

2. Выбор

Восходящие связи, определяющие выбор вектора-победителя определяются так

$$t_j = \begin{cases} \mathbf{I} \cdot \mathbf{W_j} & \text{,если j – существующий класс} \\ \alpha \cdot \sum_{i=1}^{m} i_i & \text{, иначе} \end{cases}$$

$$0 \leq \alpha \leq \frac{1}{\sqrt{m}.}$$

Восходящие связи определены по-разному для существующих и незарегистрированных прототипов. Выбираемый параметр α>=0 определяет максимальную глубину поиска для подходящего класса. При α =0 все существующие прототипы будут проверены перед тем как новый прототип будет выбран как победитель.

3. • Состязание

Резонанс и адаптация происходят также, и если индекс победителя J -индекс нового прототипа или если J - существующий прототип и

$$\rho \leq \mathbf{I} \cdot \mathbf{W_J} = t_J.$$

4. • Адаптация

Адаптация заключительного вектора-победителя требует его изменения

$$\mathbf{W_J^{(new)}} = \aleph\left(\eta \cdot \mathbf{I} + (1-\eta) \cdot \mathbf{W_J^{(old)}}\right) \quad 0 \leq \eta \leq 1.$$

АРТ сети 2А-типа всегда используют быстро - передающий медленно-запоминающий способ. Поэтому норма изучения установлена η=1 если J –

новый прототип, и к более низким значениям при дальнейшей адаптации. Так как состязание и выбор не оценивают значения новых прототипов, нет никакой потребности инициализировать их определенными значениями. APT 2A- сети не должны использоваться при быстро-обучающемся способе с η=1, потому что прототипы начинают "скакать" между всеми векторами класса вместо того, чтобы сходиться.

1.3.3 APT 2A-C

Главное неудобство APT 2A - потеря всей информации, закодированной в длине входного вектора, потому что все векторы нормализованы к единице Евклидовой длины. Другими словами, APT 2A не может найти различие между двумя незакодированными входами A1 и A2, где A1=A2*c. При использовании кодирования дополнения, вся информация, хранившая в длине незакодированного вектора закодирована в результирующий вектор $\mathbf{I} = (\mathbf{A}, \mathbf{A}^c)$. Один из способов включить дополнительный код в APT-2A алгоритм состоит в его использовании в качестве дополнительного шага предварительной подготовки вектора перед входом в алгоритм. Однако при этом прототипы нормализованы к единице измерения и адаптированы к нормализованным входным векторам. Чтобы сохранить геометрическую интерпретацию прототипов как меру всех незакодированных входных векторов , принадлежащих классу, нормализацию перемещают из функции предобработки и адаптации в функцию выбора и состязания. Полный алгоритм:

1. Предобработка

$$\mathbf{I} = (\mathbf{A}, \mathbf{A}^c) \quad a_i \in [0,1] \quad \forall i.$$

2. Выбор

$$t_j = \begin{cases} \aleph(\mathbf{I}) \cdot \aleph(\mathbf{W_j}) & \text{,если j – существующий класс} \\ \alpha \cdot \sum_{i=1}^{m} i_i & \text{, иначе} \end{cases}$$

$$0 \le \alpha \le \frac{1}{\sqrt{m}}.$$

3. Состязание

   Как и в APT-2A, резонанс и адаптация происходит, когда J – индекс нового класса, или если J – индекс победившего прототипа и $\rho \le t_J$.

4. Адаптация

$$\mathbf{W_J^{(new)}} = \eta \cdot \mathbf{I} + (1-\eta) \cdot \mathbf{W_J^{(old)}} \quad 0 \le \eta \le 1.$$

1.3.4 ART-2A-E

Другой путь сохранить информацию о длине вектора при обработке в ART-2A – заменить меру ART-2A эвклидовым измерением сходства и пропускать нормализацию длины входа данных на этапе подготовки и адаптации.

1. Предобработка

Все элементы входного вектора должны быть приведены в интервал [0,1]

2. Выбор

Восходящие связи определены с использованием меры Эвклида, нормализуемой по размерности m входного вектора. Это сохраняет степень схожести независимо от числа элементов вектора. Меру вычитают из 1, чтобы получить $t_j=1$, если входной вектор и прототип w_j тождественны.

$$t_j = 1 - \sqrt{\frac{1}{m}\sum_{i=1}^{m}(i_i - w_{ji})^2}.$$

Новые прототипы должны быть инициализированы значениями $w_{ij} \geq 1$, чтобы достичь достаточно глубокого поиска существующего прототипа.

3. Состязание

Состязательная функция остаётся той же, что и в предыдущем алгоритме.

4. Адаптация

$$\mathbf{W_J^{(new)}} = \eta \cdot \mathbf{I} + (1-\eta) \cdot \mathbf{W_J^{(old)}} \quad 0 \leq \eta \leq 1.$$

1.3.5 Fuzzy ART

1. Предобработка

Все элементы входного вектора должны быть приведены в интервал [0,1]

2. Выбор

Восходящие связи, ведущие к предварительному выбору прототипа, определены с использованием нечеткой конъюнкции (^), определяемой как

$$x \wedge y = \min\{x, y\}$$

$$\mathbf{X} \wedge \mathbf{Y} = (x_1 \wedge y_1, \cdots, x_m \wedge y_m).$$

Отдельная сетевая активность t_j может быть рассмотрена как степень того, насколько прототип w_j, является нечетким подмножеством входного образца I.

$$t_j = \frac{|\mathbf{I} \wedge \mathbf{W_j}|}{\alpha + |\mathbf{W_j}|}$$

Здесь Y – нечеткое подмножество X, если X^У=У . Размер вектора |X| определен его L_1 нормой (суммой его компонентов).

3. Состязание

Схожесть входящего I и текущего победившего прототипа w_j измеряется степенью того, насколько I является нечетким подмножеством w_j. Резонанс наступает при

$$\rho \leq \frac{|\mathbf{I} \wedge \mathbf{W_J}|}{|\mathbf{I}|}.$$

4. Адаптация

Победивший прототип w_j адаптируется путем приведения его значений к общему min вектора I и Wj

$$\mathbf{W_J^{(new)} = \eta \cdot (I \wedge W_J^{(old)}) + (1 - \eta) \cdot W_J^{(old)}}.$$

Мера обучения $\eta \in [0,1]$ определяет как быстро прототипы сходятся к общему min всех входящих образцов, связанных с одинаковым классом. При $\eta \to 1$ сеть работает в режиме быстрого обучения, стабилизация состояния сети происходит после нескольких представлений всех тренировочных образцов.
Наоборот, низкая мера обучения ведет к медленному обучающему режиму. АРТ-сети могут быть запущены в чисто классифицирующем режиме при установке меры обучения предварительно обученной нейросети к 0, что мешает всем пртоипам быть модифицированными новыми входными векторами. Новые прототипы инициализируются константой $w_{ij} \geq 1$. Чем выше значение $w_{ij} \geq 1$, тем ниже восходящая активность t_j от нового прототипа. То есть $w_{ij} \to \infty$ гарантирует то, что все существующие прототипы будут проверены, перед тем как новый прототип будет признан победителем.
В таблице 1 все описанные алгоритмы сведены вместе.

Таблица А1 – Общий вид алгоритмов АРТ-2А и Fuzzy-ART

	Fuzzy ART	ART 2A-C	ART 2A-E	ART 2A
Preproc.	$\mathbf{I} = (\mathbf{A}, \mathbf{A}^c)$ $a_i \in [0,1] \quad \forall \quad i$		$\mathbf{I} = \mathbf{A}$ $a_i \in [0,1] \quad \forall \quad i$	$\mathbf{I} = \frac{\mathbf{A}}{\|\mathbf{A}\|} = \aleph(\mathbf{A})$ $a_i \geq 0 \quad \forall \quad i$ $\|\mathbf{A}\| > 0$
Choice	$t_j = \frac{\|\mathbf{I} \wedge \mathbf{W_j}\|}{\|\mathbf{W_j}\|}$	$t_j = \frac{\mathbf{I} \cdot \mathbf{W_j}}{\|\mathbf{I}\| \cdot \|\mathbf{W_j}\|}$	$t_j = 1 - \sqrt{\frac{\sum_i (i_i - w_{ji})^2}{m}}$	$t_j = \mathbf{I} \cdot \mathbf{W_j}$
Match	$\rho \leq \frac{\|\mathbf{I} \wedge \mathbf{W_J}\|}{\|\mathbf{I}\|}$	$\rho \leq t_J$		
Adapt.	$\mathbf{W_J}^{(new)} =$ $\eta \cdot (\mathbf{I} \wedge \mathbf{W_J}^{(old)})$ $+(1-\eta) \cdot \mathbf{W_J}^{(old)}$	$\mathbf{W_J}^{(new)} = \eta \cdot \mathbf{I} + (1-\eta) \cdot \mathbf{W_J}^{(old)}$		$\mathbf{W_J}^{(new)} = \aleph(\eta \cdot \mathbf{I}$ $+(1-\eta) \cdot \mathbf{W_J}^{(old)})$

1.4 Принципы функционирования сети АРТ–3

Следующим шагом в развитии АРТ явилась сеть АРТ-3. Особенности обучения нейронов сетей АРТ-1 и АРТ-2 не позволяют использовать эти сети, как элементы более крупных иерархических нейросистем, в частности, компоновать из них многослойные сети. Это затрудняет представление в АРТ иерархически организованной информации, что характерно для систем восприятия человека и животных.

Эти проблемы решены в сети АРТ-3, которая выступает как многослойная архитектура. При переходе от слоя к слою происходит контрастирование входных образов и запоминание их в виде все более общих категорий. При этом основной задачей каждого отдельного слоя является сжатие входящей информации.

Образ входит в адаптирующийся резонанс между некоторой парой слоев, в дальнейшем этот резонанс рапространяется на следующие слои иерархии. В АРТ-1 и АРТ-2 недостаточный уровень резонанса приводил к генерации сигнала сброса, что приводило к полному торможению слоя распознавания. В случае многослойной сети АРТ-3 это недопустимо, так как это разрывает поток информации. Поэтому в АРТ-3 введен специальный механизм зависимости активности синапсов обратных связей от времени, аналогичный рефрактерному торможению биологического нейрона после передачи возбуждения. Поэтому вместо полного сброса сигнала происходит торможение синаптических сигналов обратной связи, и слой сравнения получает исходное состояние возбуждения для выполнения фазы поиска нового резонанса.

Интересным предложением является также использование в многослойной иерархии слоев, которые не являются слоями АРТ, а принадлежат некоторой другой архитектуре. В этом случае система получается гибридной, что может привести к возникновению новых полезных свойств.

Развитие теории АРТ продолжается. По высказыванию авторов теории, АРТ представляет собой нечто существенно более конкретное, чем философское построение, но намного менее конкретное, чем законченная программа для компьютера. Однако уже в современном виде, опираясь на свою более чем 20-летнюю историю, сети АРТ демонстрируют свои успешные применения в различных областях. АРТ сделала также важный шаг в общей проблеме моделирования пластично-стабильного восприятия.

2 Описание интерфейса программы

2.1 Схема диалог

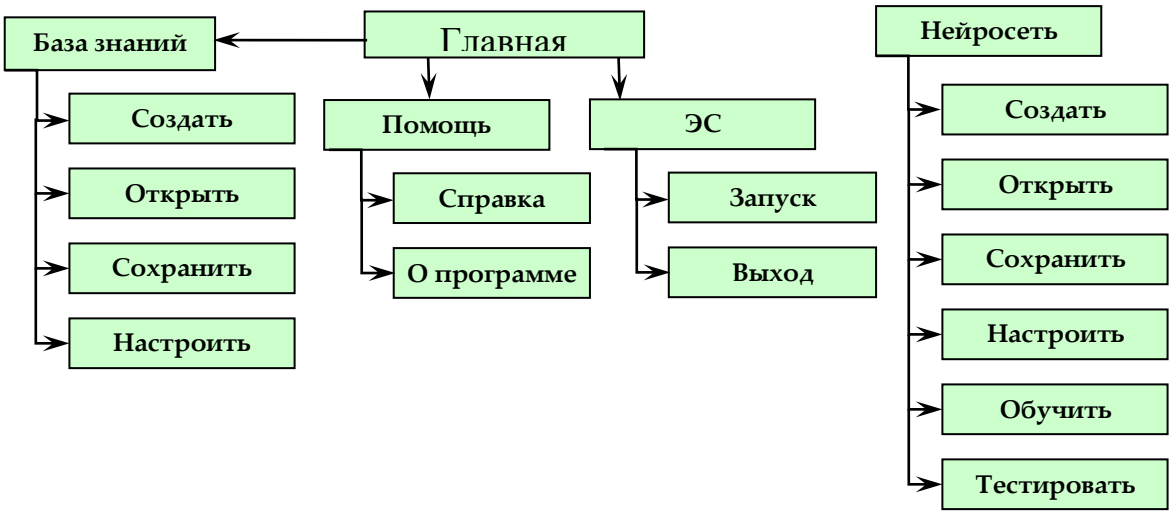


Рисунок А1 – Структура интерфейса программы

2.2 Меню программы

Так как основой интерфейса программы является меню, то рассмотрим его подробнее. Меню программы состоит из следующих основных пунктов:

- База знаний. Для работы с файлами баз знаний.
- Нейросеть. Для работы с нейронной сетью, задания параметров нейронной сети и параметров обучения.
- ЭС. Для запуска ЭС и завершения работы программы.
- Помощь. Для вызова справки по программе и отображения информации о программе.

2.3 Меню «База знаний»

Данный пункт меню состоит из следующих подпунктов: создать, открыть, редактировать. Рассмотрим работу каждого подпункта подробнее:

- Создать. При выборе этого подпункта происходит запрос для задания параметров базы знаний и последующего ее создания.
- Открыть. При выборе этого подпункта происходит открытие стандартного диалогового окна открытия файлов для открытия файла базы знаний.
- Редактировать. При выборе этого подпункта происходит отображение текущей базы знаний для ее редактирования.

2.4 Меню «Нейросеть»

Данный пункт меню состоит из следующих подпунктов: создать, открыть, сохранить, обучить, параметры. Рассмотрим работу каждого подпункта подробнее:

- Создать. При выборе этого подпункта происходит запрос для задания параметров нейронной сети и последующего ее создания.

- Открыть. При выборе этого подпункта происходит открытие стандартного диалогового окна открытия файлов для открытия файла нейронной сети.

- Сохранить. При выборе этого подпункта происходит открытие стандартного диалогового окна сохранения файлов для записи файла нейронной сети на диск.

- Обучить. При выборе этого подпункта происходит обучение нейронной сети с использованием данных из базы знаний.

- Параметры. При выборе этого подпункта происходит открытие окна параметров нейронной сети для возможности их изменения и последующего обучения. Это необходимо при исследовании влияния различных параметров нейронной сети на скорость и качество обучения.

2.5 Пункт «ЭС»

Данный пункт меню состоит из следующих подпунктов: запуск, выход.
Подпункт «Запуск» предназначен для запуска нейросетевой экспертной системы с предварительным вводом входных параметров нейронной сети и последующего процесса функционирования нейросетевой экспертной системы.
Подпункт «Выход» завершает приложение

2.6 Меню «Помощь»

Меню «Помощь» содержит в себе два подпункта: помощь и о программе. При выборе подпункта «О программе» происходит вызов диалогового окна, содержащего информацию о назначении и версии данной программы. При выборе подпункта «Помощь» происходит вызов справки, содержащей описание работы программы, а также некоторую необходимую теоретическую информацию о нейронных сетях, принципах их функционирования.

Библиографический список

1. Thomas Frank, Karl-Friedrich Kraiss, Torsten Kuhlen «Comparative Analysis of Fuzzy ART and ART-2A Network Clustering Performance»

2. http://www.gotai.net/documents/doc-nn-007-11.aspx