МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТРАНСПОРТА»

Н.В. РЯЗАНЦЕВА

НЕЙРОННЫЕ СЕТИ

Методические указания

к выполнению лабораторных работ по дисциплине
“Техническая кибернетика на транспорте”

Гомель 2003

УДК 004.93:007.52:681.32

Рецензент – В.Я. Негрей, доктор технических наук, профессор, первый проректор БелГУТа- проректор по обучающей работе

Н.В.Рязанцева

Нейронные сети: Методические указания к выполнению лабораторных работ по дисциплине “Техническая кибернетика на транспорте” – Гомель: БелГУТ, 2003.-70 с.

ВВЕДЕНИЕ

Методические указания предназначены для изучения и практического освоения студентами всех форм обучение основ нейроинформатики как современной технологии построения систем поддержки принятия решений в рамках изучения курса “Техническая кибернетика на транспорте”. Методические указания также рекомендуются для использования при выполнении курсовых и исследовательских дипломных проектов.

Порядок выполнения и сдачи лабораторных работ

Соответственно графику студенты перед выполнением лабораторной работы должны ознакомиться с методическими указаниями по ее выполнению и рекомендованной литературой. Во время занятий каждый студент получает индивидуальный вариант задачи.

Для получения зачета по каждой работе студент сдает преподавателю оформленный отчет с выводами и рекомендациями, текстами разработанных программ, файлами программ, которые выполняются, файлами данных и текстом отчета.

Отчет выполняют на белой бумаге формата А4. Текст размещают только с одной стороны листа. Поля страницы со всех сторон – 20 мм. Листы скрепляют с помощью канцелярских скрепок. Для набора текста отчета используют редактора MS Word : шрифт Times New Roman, 12 пунктов. Междустрочный интервал: полуторный – для текста отчета, одинарный – для листингов программ, таблиц и распечаток данных.

Во время собеседования студент должный показать знания о цели работы, по теоретическому материалу, о методах выполнения каждого этапа работы, по содержанию основных разделов разработанного отчета с демонстрацией результатов на конкретных примерах.

Студент должный уметь правильно анализировать полученные результаты и объяснить физическую сущность полученных зависимостей и характеристик. Для самопроверки при подготовке к выполнению и сдачи лабораторной работы студент должный ответить

на контрольные вопросы, приведенные в конце описания соответствующей лабораторной работы.

Общий зачет студент получает после выполнения и сдачи последней лабораторной работы.

Лабораторная работа № 1

ОДНОСЛОЙНЫЙ ПЕРСЕПТРОН. АЛГОРИТМ УИДРОУ-ХОФФА

Цель работы – изучить различные модели формального нейрона и методы обучения однослойного персептрона; исследовать влияние вида функции активации нейрона, а также шага обучения на время и точность обучения и классификации; ознакомиться со стандартными программными средствами для моделирования однослойного персептрона.

1 Краткие сведения из теории

Нервная система человека состоит из клеток, которые называются нейронами, и очень сложна: порядка 10^{11} нейронов принимают участие в 10^{15} передающих связях, которые имеют длину метр и больше. Каждый нейрон имеет много качеств, общих с другими клетками, но его уникальной способностью является прием, обработка и передача электрохимических сигналов по нервным путям, которые образовывают коммуникационную систему мозга.

В биологическом нейроне дендриты идут от тела нервной клетки к другим нейронам, где они принимают сигналы в точках соединения, которые называются **синапсами**.

Принятые синапсом входные сигналы идут к телу нейрона. Здесь они суммируются, причем одни входы стремятся возбудить нейрон, другие – воспрепятствовать его возбуждению. Когда суммарное возбуждение в теле нейрона превышает некоторый порог, нейрон возбуждается, посылая по аксону сигнал другим нейронам. У этой основной функциональной схемы много исключений, тем не менее большинство искусственных нейронных сетей (НС) моделируют лишь эти простые свойства.

Итак, нейрон (формальный нейрон, нейроподобный элемент) представляет собой примитивное вычислительное устройство, которое имеет несколько входов и один выход, и есть основным вычислительным элементом НС.

Однослойный персептрон является одним из простейших вариантов НС (рис. 1) и содержит лишь один нейрон. Будучи

самостоятельной моделью НС с одной стороны, однослойный персептрон (формальный нейрон) является основным конструкционным элементом для большинства моделей НС, с другой стороны.

На вход однослойного персептрона (формального нейрона) поступает набор входных сигналов x_1, x_2, ..., x или входной вектор x. Каждый входной сигнал умножается на соответствующий вес связи w_1, w_2,..., w – аналог эффективности синапса (межнейронного контакта).

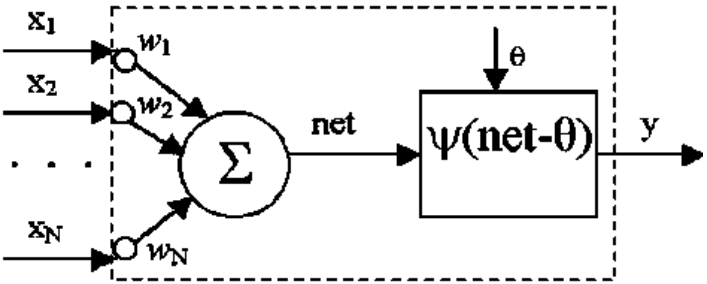


Рисунок 1 – Однослойный персептрон (формальный нейрон)

Вес связи является скалярной величиной, положительной для возбудимых и отрицательной для тормозящих связей. Взвешенные весами связей входные сигналы поступают на блок суммирования, который аналогичен телу клетки, где осуществляется алгебраическое суммирование и определяется уровень возбуждения нейроподобного элемента *net*:

$$net = \sum_{i=1}^{N} w_i x_i .$$

Исходный сигнал нейрона y определяется путем пропускания уровня возбуждения *net* через нелинейную функцию активации ψ:

$$y = \psi\,(net - \theta),$$

где θ – некоторый постоянный сдвиг (аналог порога нейрона). Обычно используются простейшие нелинейные функции:

бинарная (пороговая): $u(x) = \begin{cases} 1, при\ \ x > 0, \\ 0, при\ x < 0, \end{cases}$

или сигмоидная: $\psi(x) = 1 / (1 + e^{-x})$.

В зависимости от типа функции активации различают дискретные персептроны, используют пороговую функцию активации, и действительные – которые используют действительные функции активации, например, сигмоидную функцию.

Каждый нейрон имеет небольшую память, которая реализуется весовыми коэффициентами и порогом нейрона. Поэтому нейроны можно рассматривать как запоминающие устройства. В то же время нейроны могут рассматриваться как примитивные процессоры, которые осуществляют вычисление значения функции активации на основе различия взвешенной суммы входных сигналов и порога.

Алгоритм обучения однослойного дискретного персептрона имеет вид.

Шаг 1. Весам $w_i(0)$ ($u = 1, ..., N$) и порогу $\theta(0)$ присваиваются случайные значения (через $w_i(t)$ обозначен весовой коэффициент i-го входа персептрона в момент времени t, через $\theta(t)$ обозначена величина сдвига (порога) нейрона в момент времени t).

Шаг 2. Предъявляются очередной входной вектор $x = \{x_1, ..., x\}^T$ из учебного множества и желательный выход $y^*(t)$ ($y^*(t) = 1$, если $x(t)$ относится к классу A, $y^*(t) = 0$, если $x(t)$ относится к классу B).

Шаг 3. Вычисляется значения на выходе персептрона по формулам:

$$net = \sum_{i=1}^{N} w_i(t)x_i(t),$$

$$y(t) = ш(net - u(t)).$$

Шаг 4. Корректируются веса персептрона соответственно равенствам:

$w_i(t + 1) = w_i(t) + \eta(y^*(t) - y(t))x_i(t)$, $i = 1, 2, ..., N$,

$\theta(t + 1) = \theta(t) + \eta(y^*(t) - y(t))$,

где η – положительный корректирующий прирост.

Шаг 5. Если достигнута сходимость, то процедура обучения заканчивается; в противоположном случае – переход к шагу 2.

Соответственно данному алгоритму сначала вырабатывается инициализация параметров персептрона случайными значениями. Потом поочередно предъявляются экземпляры с известной классификацией, выбранные из учебного множества, и корректируются веса соответственно формулам шагов 3 и 4. Величина коррекции определяется положительным корректирующим

приростом η, конкретное значение которого выбирается большим, чтобы быстрее вырабатывалась коррекция весов, и в то же время настолько малым, чтобы не допустить чрезмерного возрастания значений весов.

Процедура обучения продолжается до тех пор, пока не будет достигнута сходимость, то есть пока не будут получены веса, которые обеспечивают правильную классификацию для всех образов из обучающего множества.

В том случае, когда обучающие выборки разделить гиперплоскостью невозможно для обучения персептрона можно использовать **алгоритм Уидроу-Хоффа**, который минимизирует среднеквадратичную ошибку между желательными и реальными выходами сети для учебных данных. Этот алгоритм также можно применять для обучения однослойного действительного персептрона. Алгоритм Уидроу-Хоффа можно записать в том же виде, как и вышеописанный алгоритм, предполагая, что в узлах персептрона нелинейные элементы отсутствуют, а корректирующий прирост η в процессе итераций постепенно уменьшается к нулю.

Если для решения задачи распознавания образов используется дискретный персептрон, то решающее правило относит входной образ к классу A, если выход персептрона равняется 1, и к классу B – в противоположном случае.

В случае, если для решения задач распознавания образов используется действительный персептрон, то решающее правило относит входной образ к классу A, если выход сети больше 0.5, и к классу B – в противоположном случае.

2 Индивидуальное задание

Используя конспект лекций и рекомендованную литературу, изучить модель формального нейрона и алгоритмы обучения однослойного персептрона.

Используя документацию рекомендованного преподавателем программного средства, выучить его архитектуру и компоненты, предназначенные для моделирования и обучение однослойного персептрона, ввода и вывода данных, подготовки отчетов ,вывода и отображения результатов работы.

Ознакомиться с составом и порядком выполнения работы.

3 Порядок выполнения работы

3.1 Получить у преподавателя варианты задач для обучения и моделирование однослойного персептрона.

Входными данными являются: вид функции активации нейрона; количество входов однослойного персептрона; обучающая и контрольная выборки (наборы пар значений входов и желательных выходов персептрона); значения максимальной допустимой ошибки обучения; значения параметра сходимости обучения.

3.2 Написать и отладить программу , которая моделирует формальный нейрон (однослойный персептрон) и реализует алгоритмы обучения дискретного и действительного персептронов. Предусмотреть в программе отображения на экран и запись в файл на диске текущего состояния параметров персептрона и результатов его работы: матрицы весов, ошибки классификации , значений на входах и выходе персептрона, а также времени обучения и классификации на основе персептрона.

3.3 Для соответствующих варианту входных данных осуществить обучение персептрона и сохранить в файле на диске результаты его работы для обучающей и контрольной выборок.

3.4 Несколько раз изменить вид функции активации нейрона и выполнить п.3.3.

3.6. Результаты выполнения п.п. 3.3 - 3.4 занести в таблицу, столбцы которой должны иметь названия: вид функции активации, метод обучения, время обучения, время классификации для наученного персептрона (для обучающей и контрольной выборок), ошибка классификации для обучающей и контрольной выборок. Проанализировать полученные результаты и сделать выводы о том, как влияет вид функции активации формального нейрона на время обучения и классификации , а также величину ошибки обучения и классификации однослойного персептрона.

3.7 Для тех же входных данных, что и в п.3.3 выполнить моделирование и обучения однослойного персептрона, а также выполнить п.п.3.3 - 3.6 на основе программного продукта, предложенного преподавателем. Результаты занести в таблицу. Проанализировать полученные результаты.

3.8 На основе таблиц п.3.6 и п.3.7 дать сравнительную характеристику программы, разработанной студентом, и программного средства, предложенного преподавателем.

3.9 На основе разработанной студентом программы, изменяя значение шага обучения, для персептрона с заданной функцией активации исследовать, как влияет величина шага обучения на время обучения. Построить график зависимости времени обучения персептрона от величины шага обучения.

Содержание отчета

4. 1 Цель работы.

4.2 Краткое описание модели формального нейрона (однослойного персептрона).

4.3 Схема и краткое описание (основная идея) алгоритмов обучения дискретного и действительного персептронов.

4.4 Краткое описание и текст программы, которая реализует алгоритмы 4.3.

4.5 Входные данные и результаты работы программы 4.4, таблица 3.6.

4.6 Краткое описание, входные данные и результаты работы с программным средством, предложенным преподавателем, таблица 3.7.

4.7 Результаты выполнения п.3.9, график зависимости времени обучения от величины шагу обучения.

4.8 Анализ полученных результатов и выводы.

Контрольные вопросы

1. Дайте определение и объясните взаимосвязь понятий: формальный нейрон, вес (весовой коэффициент), порог, функция активации, персептрон, желательный и реальный выход персептрона, обучения персептрона, классификация, аппроксимация, оценивания, ошибка обучения / классификации, время обучения / классификации.
2. В чем сходство и отличие биологических и формальных нейронов?

3. Влияет ли вид функции активации нейрона на продолжительность обучения и работы однослойного персептрона, величину ошибки обучения и классификации (оценивания) и, если влияет, то как?
4. Какие функции активации формальных нейронов чаще всего используют и почему?
5. Какие задачи можно решать на основе однослойных персептронов, а какие нельзя? Обоснуйте и докажите ответ. Приведите примеры.
6. Можно ли научить дискретный однослойный персептрон вычислять значение функций: $y = (x_1$ ***and*** $x_2)$, $y = (x_1$ ***xor*** $x_2)$, $y =$ ***not*** $((***not***\ x_1)$ ***and*** $x_2)$, а действительный однослойный персептрон – вычислять значение функций: $y = 3x_{1\text{-}0},05x_2$, $y = sin(x_1) + 0,3x_2$, $y = 0,5x_1 + 2x_{2\text{-}2},5(x_1 - x_2) + 5,5$? Ответа обоснуйте и, если возможно, объясните рисунками.
7. Объясните, что означают линейная раздельность и линейная неразделенность классов.
8. Целесообразно ли применять однослойный персептрон для классификации нелинейно разделенных образов?
9. Всегда ли совпадают алгоритмы обучения однослойного персептрона?
10. Влияет ли величина шага обучения на время обучения однослойного персептрона? Влияет ли объем обучающей выборки на скорость обучения однослойного персептрона? Влияет ли количество использованных признаков на скорость обучения однослойного персептрона?
11. Что такое репрезентативная выборка данных? Должна ли обучающая выборка быть репрезентативной? Должна ли тестовая выборка быть репрезентативной? Влияет ли репрезентативность обучающей выборки на точность классификации экземпляров тестовой выборки?
12. Влияет ли репрезентативность тестовой выборки на точность классификации экземпляров тестовой выборки, на точность обучения персептрона по обучающей выборке?

Лабораторная работа № 2

ПОЛНОСВЯЗНЫЕ НЕЙРОННЫЕ СЕТИ ХОПФИЛДА. ПСЕВДОИНВЕРСНОЕ УЧЕБНОЕ ПРАВИЛО.

Цель работы – изучить модель полносвязнной бинарной НС Хопфилда и методы ее обучения; сравнить методы обучения НС Хопфилда и рассмотреть примеры ее практического использования; сравнить возможности сети Хопфилда с возможностями дискретного однослойного персептрона; ознакомиться с стандартными программными средствами для моделирования НС Хопфилда.

1 Краткие сведения из теории

НС Хопфилда (псевдоинверсная НС) задается четверкой (N, w, θ, x), где N – число нейронов в сети, $\theta=\{\theta_1, \theta_2, ..., \theta_N\}$ – вектор внешних влияний. Нейроны связаны по принципу “все со всеми”, это значит, что в сети Nx связей. Связь между i-м и j-м нейронами - w_{ij}. Величина w_{ij} называется весом связи и может быть нулем, положительным или отрицательным числом. Веса связей задаются матрицей $w = \{w_{ij}\}$, $u, j = 1, .., N$. В модели Хопфилда связи симметричные, то есть $w_{ij} = w_{ji}$. Состояние сети определяется вектором состояний нейронов $x=\{x_1, ..., x\}$.

Нейрон рассматривается как двухстабильный пороговый элемент (модель Мак-Каллока - Питтса). Состояние x_i нейрона u может иметь два значения 0 и 1 или –1 и 1. Нейрон u имеет внешний вход θ, входы от других нейронов x_j и один выход, равный x_i. Вход в нейрон u (постсинаптический потенциал) определяется суммой взвешенных состояний, связанных с ним нейронов:

$$net_i = \sum_{j=1}^{N} w_{ij} x_j + \theta_i .$$

В зависимости от величины входа net_i нейрон u изменяет свое состояние или остается в предыдущем состоянии соответственно порогового правилу $net_i^{k+1} = \psi(net_i^k)$, где k, k+1 – номера старого и нового состояний нейрона u, а $\psi(x)$ – функция активации нейрона:

$\psi(x) = \begin{cases} 0, x \geq 0 \\ 1, x < 0 \end{cases}$ (пороговая) или $\psi(x) = \dfrac{1}{1+e^{-x}}$ (сигмоидная).

Сеть может изменять свое состояние синхронным или асинхронным способом.

В **синхронном случае** все нейроны одновременно изменяют свои состояния. Аналитическое выражение перехода сети с состояния x_k в x_{k+1} записывается в матричной форме: $net_k = wx_k + \theta_k$, $x_{k+1} = \psi(net_k)$, где $x_k = \{x_1^k, x_2^k, ..., x_N^k\}$, $net_k = \{net_1^k, net_2^k, ..., net_N^k\}$. Функция ψ применяется к вектору net_k поэлементно.

В **асинхронном случае** каждый нейрон может изменять свое состояние случайно, при этом он использует информацию о возобновленных состояниях других нейронов. Аналитическая запись перехода сети с состояния x_k в x_{k+1} в асинхронном случае, когда нейрон m изменяет свое состояние, имеет вид: $net_m^k = w_m x_k + \theta_k$, $x_{k+1} = \{x_1^k, ..., \psi(net_m^k), ..., x_N^k\}$, где w_m – строка матрицы w с номером m.

Начиная с начального состояния x_0 и работая синхронно или асинхронно, сеть генерирует последовательность состояний $x_0, x_1, ..., x$, что в благоприятных случаях заканчивается стабильным состоянием, в неблагоприятных случаях могут возникнуть колебания.

Основной операцией, которая выполняется нейронною сетью, является умножения матрицы на вектор (в синхронном случае) или вектора на вектор (в асинхронном случае) с последующим вычислением нелинейной функции. Однако благодаря массовости связей большого числа нейронов при такой довольно простой операции сеть способна решать сложные задачи.

Одной из задач, которая решается с помощью НС, является задача распознавания образов. Сеть с N нейронов может восстанавливать образы размера N, что запомнены в сети, то есть на неполной или неточной информацией об этих образах. Это разрешает создавать на основе НС блоки ассоциативной памяти в вычислительных системах. Эталонные образы кодируются словами длиной N, которые состоят из 1 и –1 (0 и 1). Переход нейрона с состояния 1 в -1 (и наоборот) будем считать ступенчатым.

Работу НС, которая содержит нейроны, состояние которых определяется действием внешних стимулов, выходы которого являются реакцией на стимул, можно рассматривать как пофрагментную классификацию действующих стимулов. Множество

передсинаптических потенциалов, которые отвечают положительной реакции нейрона $x_i(t) = 1$, создает компактную область вокруг вектора w_{ij}.

Эта область не пустая лишь при выполнении условия:

$$\theta_i(t) \le \left(N \sum_{j=1}^{N} w_{ij}^2 \right)^{0.5}.$$

Комбинируя значение весовых коэффициентов w_{ij} и порогов θ_u можно создавать НС, которые способны к классификации образов без любых ограничений. Нахождения значений величин w_{ij} и $\theta_u(t)$, которые обеспечивают необходимые реакции на заданном множестве стимулов, составляет задачу обучения НС.

Распознавание образа достигается сетью путем эволюции из начального состояния, которое отвечает введенному образу, в конечное состояние, которым является ассоциация образа с запомненным раньше образом. Основная задача построения сети состоит в наделении ее распознающими свойствами, которые состоят в том, что при подаче на вход сети версии образа, сеть на выходе способна восстанавливать оригинал из шума. То есть, имея M образов-эталонов необходимо найти такую матрицу связей w, что заставляла бы сеть проявлять распознающие свойства относительно этих эталонов. Нахождения такой матрицы составляет процесс обучения НС, а правило вычисления матрицы w есть учебное правило. Производительность НС определяется количеством эталонов M, которые могут быть запомнены и распознаны сетью, которая состоит из N нейронов, и тем, насколько хорошо происходит распознавание, то есть отделения эталонов от шума при данном количестве эталонов M.

Заметным шагом на пути к увеличению производительности псевдоинверсных НС было предложение Хопфилда рассматривать их с энергетической точки зрения. Процесс распознавания можно представить как "сдвиг" сети в минимумы некоторой энергетической функции E в пространстве состояний. Предложенное Хопфилдом определение этой функции имеет вид:

$$E(t) = -0.5\, x(t)(wx(t) - \theta(t)),$$

где $x(t)$ – вектор состояния системы; $\theta(t)$ – внешнее поле, которое определяет порог чувствительности; w – оператор, который

учитывает расстояние между состояниями системы.

Проекционный алгоритм обучения (псевдоинверсный алгоритм) НС Хопфилда состоит в том, что мы осуществляем проекцию учебного множества на множество весов НС, которое разрешает обучить сеть один раз перед использованием, и не требует выполнения итеративного процесса коррекции весов, как, например, в алгоритмах обучения персептронов.

Пусть есть M эталонных образов x_k, $k = 1, ..., M$. Положив $\theta_u = 0$, $u = 1, ..., N$, можем записать выражение для нахождения весов сети:

$$w_{ij} = \sum_{k=1}^{M} x_i^k x_j^k .$$

Начиная работу в состоянии x_0, сеть может попасть в одно из следующих трех положений:

– прийти в стабильное состояние, которое отвечает эталонному образу, который по хемминговому расстоянию (равному числу компонентов, в которых различаются два вектора) является наиболее близким к x_0;
– прийти в стойкое состояние, которое не отвечает никакому эталону, то есть к ошибочному образу;
– оказаться втянутой в колебательный процесс.

Результаты моделирования сети показывают, что сеть работает хорошо, то есть без ошибок восстанавливает эталонные образы из случайных, если в нее записывается не более, чем $0,15N$ эталонных образов.

Для большинства задач возможности одноразового внесения информации в ассоциативную память оказывается недостаточно, так как нужно в процессе работы прибавлять в память новую информацию – проекционный алгоритм (см. выше) оказывается непригодным. Для решения таких задач используют **итеративный проекционный алгоритм** обучения. Пусть мы имеем наученную для M наборов значений НС Хопфилда, тогда для внесения в память $M+1$ набора значений, значения весов можно установить:

$$w_{ij}^{M+1} = w_{ij}^{M} + \frac{(x_i^M - \sigma_i)(x_j^M - \sigma_j)}{\sum_{j=1}^{N} x_j^{M+1}(x_j^{M+1} - \sigma_j)},$$

где $\sigma_i = \sum_{j=1}^{N} w_{ij}^{M} x_j^{M+1}$.

Для упрощения вычислительной процедуры предложены и прочие правила установки весов, среди которых в особенности следует выделить:

$$w_{ij}^{M+1} = w_{ij}^{M} + \frac{(x_i^{M+1} - \sigma_i) x_j^{M+1}}{N}.$$

При уменьшении веса связей, которые замыкают выход нейрона на его вход, НС способна вспоминать образы, которые казались безнадежно утерянными. Этот эффект был положен в основу разработки нового метода повышение объема ассоциативной памяти НС Хопфилда, что разрешило почти в два раза увеличить теоретическую границу для объема памяти и сделало возможным регулировать количество образов, которые запоминаются.

Теоретическая граница объема памяти для псевдоинверсного учебного правила составляет 50% от количества нейронов сети, но практически она очень велика. При обучении на основе псевдоинверсного учебного правила поведение сети существенным образом зависит от уровня обратной связи нейронов. Уменьшение обратной связи будет оказывать содействие улучшению характеристики учебного правила НС.

Теоретический анализ этой методики и ее экспериментальная проверка путем программного моделирования, показывают, что объем памяти модифицированной псевдоинверсной НС может не только достигать теоретической границы 50% от количества нейронов, но и значительно превышать ее. Предложенную методику назвали **разносечением сети**, а сеть, построенную по этой методике, – **разноскечиной псевдоинверсной НС**.

Для псевдоиверсного учебного правила при увеличении заполнения памяти НС (M/N) диагональные элементы синаптической матрицы начинают доминировать над другими ее элементами, а с увеличением веса диагональных элементов уменьшается вероятность попадания сети в локальные минимумы. Эти два факта разрешили предложить модификацию псевдоинверсного учебного правила.

После того, как значения синаптической матрицы будут найдены, все диагональные элементы этой матрицы необходимо частично уменьшить по правилу:

$$w_{ii}^{*} = Dw_{ii}, \qquad 0 < D < 1.$$

Такое сокращение ослабляет уровень отрицательной обратной связи нейронов, который приводит к определенной дестабилизации поведения НС. Уменьшая диагональные элементы матрицы, мы уменьшаем величину соотношения w_{ii} / w_{ij}, что дает эффект, похожий на сокращение количества запомненных эталонов, то есть сокращения уменьшения насыщения памяти сети. Полную НС, построенную по этому правилу назвали **разносечениой сетью**. Синаптическая матрица w^{*} может быть определена как:

$$w^{*} = w - (1 - D)I = xx - (1 - D)I,$$

где D – коэффициент разносечения, $0 < D < 1$, оптимальное значение коэффициента D лежит в границах $0.1 - 0.2$; I – единичная матрица.

После обучения НС диагональные элементы синаптической матрицы увеличиваются на положительный коэффициент $D < 1$. При испытании такой сети, постсинаптичный потенциал каждого нейрона получает прирост: $d_i = (D-1)w_{ii}x_i$.

Поскольку $D < 1$, а $w_{ii} > 0$, увеличения имеет знак, противоположный выходу нейрона, и действует как дестабилизирующий фактор. Критический объем памяти сети при деформации увеличивается, и удваивается при $D = 0$.

Экспериментально подтвержденная возможность значительного (в 2-3 разы) увеличения объема ассоциативной памяти НС при ослаблении диагонали в 3-5 раз (D = 0,2-0,3). Дальнейшее уменьшение веса диагональных элементов синаптической матрицы при параллельной организации нейровычислений увеличивает риск потери стабильности сети.

3 ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Используя конспект лекций и рекомендованную литературу, выучить модель НС Хопфилда и алгоритмы ее обучения.

Используя документацию рекомендованного преподавателем программного средства, выучить его архитектуру и компоненты, предназначенные для моделирования и обучение НС Хопфилда, ввода и вывода данных, подготовки отчетов (вывода и отображения

результатов работы).Ознакомиться с составом и порядком выполнения работы.

3.1 Получить у преподавателя варианты задач для моделирования и обучение НС Хопфилда. Входными данными будут: количество входов нейронов и размер сети (число нейронов); набор пар значений входов x_i и желательных выходов y_i НС Хопфилда.

3.2 Написать и отладить программу , которая моделирует НС Хопфилда и реализует алгоритмы ее обучения: проекционный (псевдоинверсный) алгоритм и итерационный проекционный алгоритм, а также разносечение псевдоинверсное правило. Предусмотреть в программе отображения на экран и запись в файл на диске текущего состояния параметров НС и результатов его работы: матрицу весов, число ошибочных решений распознавания, значений на входах и выходах НС Хопфилда, продолжительность обучения и работы НС.

3.3 Для соответствующих варианту входных данных сделать обучение НС каждым из методов и сохранить в файле на диске результаты ее работы.

3.4 Результаты занести в таблицу, столбцы которой должны иметь названия: метод обучения, время обучения, время классификации, число ошибочных решений при распознавании.

3.5 Проанализировать полученные результаты и дать сравнительную характеристику методов обучения НС Хопфилда.

3.6 Выполнить п.3.3-3.5 для программных продуктов, предложенных преподавателем. Сделать анализ и дать сравнительную характеристику разработанной Вами программы и программных продуктов, предложенных преподавателем.

3.7 Для той же задачи, которая и в гг. 3.3, 3.6 осуществить моделирование на основе однослойного персептрона. Результаты расчетов занести в таблицу.

3.8 Дать сравнительную характеристику НС Хопфилда и однослойного персептрона.

4 СОДЕРЖАНИЕ ОТЧЕТА

4.1 Сформулированная цель работы.

4.2 Короткое описание и схема модели НС Хопфилда.

4.3 Схема и короткое описание алгоритмов обучения НС Хопфилда.

4.4 Короткое описание и текст программы, которая реализует алгоритмы 4.3.

4.5 Входные данные и результаты работы программы 4.4.

4.6 Короткое описание, входные данные и результаты работы с программным средством, предложенным преподавателем.

4.7 Результаты выполнения п. 3.7.

4.8 Анализ полученных результатов и выводы. Лаконичные ответы на контрольные вопросы.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Дайте определение понятий: полносвязная НС Хопфилда, функция вычислительной энергии, ассоциативная память, псевдоинверсное обучающее правило, проективный алгоритм настраивания весов, эффект разносечения (эффект Городничего), эталон.
2. Разрешает ли модель и рассмотренные алгоритмы обучения НС Хопфилда построить на ее основе ассоциативное запоминающее устройство, способное запоминать столько образов, сколько нейронов в сети? Ответ объясните.
3. Опишите эффект Городничего и рассмотрите перспективы и методы его использования.
4. Какие задачи можно решать на основе бинарных НС Хопфилда, а которые нельзя? Обоснуйте и докажите ответ. Приведите примеры. Сравните возможности НС Хопфилда и дискретного персептрона.
5. Можно ли научить бинарную НС Хопфилда восстанавливать по ключу 1101**** эталонные наборы кодовых слов: а) 11011001, 01010010, 11001111, б) 11011001, 11011010, 01011111, в) 11011001, 11010010, 11011111 ? Ответ объясните.
6. Целесообразно ли применять бинарные сети Хопфилда для классификации нелинейно раздельных образов?
7. Всегда ли совпадают проекционные алгоритмы обучение сети Хопфилда?
8. Целесообразно ли применять НС Хопфилда при решении задач, для решения которых может использоваться однослойный персептрон?

Лабораторная работа № 3

САМООРГАНИЗУЮЩИЕСЯ НЕЙРОННЫЕ СЕТИ КОХОНЕНА : SOM И LVQ

Цель работы – выучить модели и свойства нейронных сетей Кохонена и методы их обучения; сравнить возможности НС Кохонена с возможностями дискретного однослойного персептрона и НС Хопфилда; ознакомиться с программными средствами, которые моделируют НС Кохонена.

1 КРАТКИЕ СВЕДЕНИЯ ИЗ ТЕОРИИ

Нейронная сеть SOM

Карта признаков самоорганизации Кохонена (Kohonen Self-Organizing Map – SOM) относится к классификаторам, для обучения которых используются выборки образов с заранее не заданной классификацией.

Задачей сети есть определения принадлежности входного вектора признаков *s*-го экземпляра выборки $x^s = \{x^s_1, x^s_2, ..., xs\}^T$ до одного из L возможных кластеров, представленных векторными центрами $w_j = \{w_{j1}, w_{j2}, ..., wj\}^T$, $j = 1, 2, ..., L$, где ^{T} – символ транспонирования.

Обозначим *i-в* компоненту входного вектора x^s в момент времени t как $x^s_i(t)$, а вес *i*-го входа *j*-го узла в момент времени t как $w_{ij}(t)$.

Если узлы SOM являются линейными, а вес *i*-го входа *j*-го узла равняется w_{ij}, $i = 1, 2, ..., N$, $j = 1, 2, ..., L$, тогда при соответствующих значениях порогов каждый *i-ma* выход сети с точностью до несущественных постоянных будет равняется евклидовому расстоянию d_j между предъявленным входным вектором x^s_i и *j*-м центром кластера.

Считается, что вектор x^s належит к *j*-го кластеру, если отстань d_j для *j*-го центра кластера w_j минимальная, то есть если $d_j \le d_k$ для каждого $k \neq j$.

При обучении НС предъявляются входные векторы без указания

желательных выходов и корректируются веса соответственно алгоритму, который предложил выдающийся финский ученый, академик Тейво Кохонен. Алгоритм Кохонена, что формирует SOM, требует, чтобы возле каждого узла было определенное поле *NE*, размер которого с течением времени постоянно уменьшается.

Шаг 1. Инициируются веса входов узлов маленькими случайными значениями. Устанавливается начальный размер поля *NE*.

Шаг 2. Предъявляется новый входной вектор x^s.

Шаг 3. Вычисляется расстояние (метрическое свидетельство) d_j между входным вектором и каждым исходным узлом j:

$$d_j = \sum_{i=1}^{N} (x_i^s(t) - w_{ji}(t))^2$$

Шаг 4. Определяется узел j^* с минимальным расстоянием d_j.

Шаг 5. Корректируются веса входов узлов, которые находятся в поле $NE_j(t)$ узла j^*, таким образом, чтобы новые значения весов были уровни:

$$w_{ji}(t+1) = w_{ji}(t) + з(t)(x_i^s - w_{ji}(t)), \quad j \in NE_j(t), \quad i = 1,2,...,N.$$

При этом корректирующий прирост $\eta(t)$ $(0 < \eta(t) < 1)$ должный спадать с ростом t.

Шаг 6. Если сходимость не достигнута, то перейти к шагу 2.

Сходимость считается достигнутой, если веса стабилизировались и корректирующий прирост η в шаге 5 снизился к нулю.

Если число входных векторов в учебном множестве есть большим относительно избранного числа кластеров, то после обучение веса сети будут определять центры кластеров, распределенные в просторные входов таким образом, что функция плотности этих центров будет аппроксимировать функцию плотности вероятности входных векторов. Кроме того, веса будут организованы таким образом, который топологично близкие узлы будут отвечать физически близким (в смысле евклидового расстояния) входным векторам.

Интерпретация результатов классификации SOM

Важно отметить, что при классификации с помощью SOM, номер узла, к которого отнесенный экземпляр, и фактический номер

его класса в общем случае не совпадают – разделяя экземпляры, SOM делает субъективную классификацию, которая не имеет того реального фактического содержания, которым мы наделяем классы.

Результаты классификации SOM могут быть наделены фактическим смыслом путем постановки в соответствие номеру каждого узла SOM номера того фактического класса, к которого относится большая часть экземпляров обучающей выборки, отнесенных SOM к данному узлу. Для осуществления такой постановки можно предложить использовать простой способ, основанный на использовании ассоциативного запоминающего устройства (АЗП).

Алгоритм обучение системы SOM-АЗП имеет вид:

Шаг 1. Реализуется учебный эксперимент и определяются фактические классы экземпляров. Вырабатывается обучения SOM для всех экземпляров обучающей выборки.

Шаг 2. Для каждого узла SOM подсчитывается число экземпляров, которые относятся к каждому из фактических классов.

Шаг 3. Каждому узлу SOM относится в соответствие тот фактический класс, к которого относится большая часть экземпляров, отнесенных SOM к данному узлу. Постановка соответствия вырабатывается путем записи пара (кортежа) <номер узла SOM, номер класса> в АЗП. В качестве АЗП может быть использован как блок линейной или динамической памяти, которая обслуживается соответствующей процедурой, так и нейромережна ассоциативная память:

а) для системы с двумя классами – однослойный дискретный персептрон;

б) для системы с большим числом классов – многослойная нейронна сеть или комбинация ассоциативной памяти на основе НС Хопфілда с нейромережним селектором максимума. При этом на соответствующие входы НС Хопфілда подаются сигналы от каждого из узлов SOM, а на выходе получают 0, если номер узла SOM не сопоставленный данному классу и 1 – если сопоставленный. Нейромережний селектор максимума определяет номер узла НС Хопфілда (то есть номер фактического класса), для которого выход равняется 1, для всех других узлов SOM выход НС Хопфілда будет равняться 0.

Нейронная сеть LVQ

Одной из моделей НС, перспективных для диагностики и прогнозирование, есть квантования учебных векторов (Learning Vector Quantization – LVQ).

В основе алгоритмов LVQ лежит механизм обучение пласта конкурирующих нейронов (конкурирующего пласта), контролируемый учителем. Конкурирующий пласт может автоматически учиться классифицировать входные векторы. В сущности, он представляет собою карту признаков самоорганизации Кохонена. Подол классов, который определяет карта признаков самоорганизации Кохонена, основанный только на расстоянии между входными векторами. Если два входных вектора очень близкие, то конкурирующий пласт, очень вероятно, отнесет их до одного класса. Однако, деление на классы, выработанное конкурирующим пластом, как правило, не совпадает с тем, что определяет учитель. Возникает задача разработки механизма, который бы разрешал карте признаков самоорганизации Кохонена осуществлять классификацию, близкую к заданной учителем. Методом, который позволяет решать поставленную задачу, есть LVQ.

Нейронная сеть LVQ (рис. 3) состоит из двух последовательно соединенных пластов нейронов: конкурирующего пласта и линейного пласта. Оба пласта НС LVQ содержат по одном конкурирующему и одном линейном нейроне на каждый подкласс / целевой класс. Обозначим S^1 – количество подклассов, S^2– количество целевых классов (S^1 всегда будет больше, чем S^2).

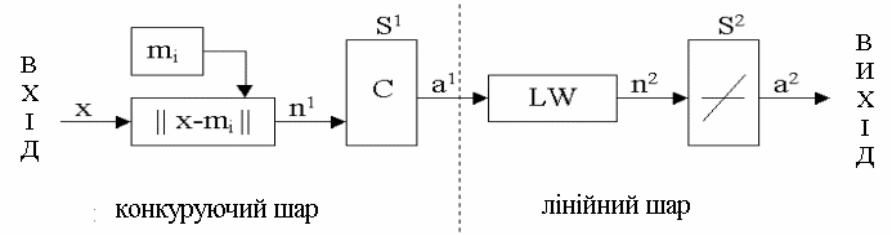


Рисунок 3 – Схема LVQ – нейронной сети

Конкурирующий пласт осуществляет деление входных векторов

x на классы, выделяя центры сосредоточения входных векторов m_i. Для этого определяются расстояния $n^1=||x - m_i||$ между входными векторами x и начальными значениями центров сосредоточения векторов m_i, $u = 1, 2, .., q$, где q – количество входных векторов.

Линейный пласт превращает класс входного вектора, определенный конкурирующим пластом – подкласс a^1, в класс, определенный пользователем – целевой класс a^2, путем умножения a^1 на значение весов LW линейных нейронов, которые устанавливаются равными 1, если целевой класс и подкласс совпадают и 0 – в противоположном случае. Соответствующие произведения $n^2=a^1LW$ подаются на выходы всех линейных нейронов, образовывая бинарный вектор a^2, все элементы которого равные 0, за исключением элемента, который отвечает целевому классу (этот элемент равняется 1).

Пусть некоторое количество векторов с свободными параметрами m_i помещенное в входное пространство для аппроксимации разных областей входного вектора x их квантованными значениями. Каждому класса значений x назначается несколько векторов с свободными параметрами, и потом принимается решения об отнесение x к тому классу, к которого належит ближайший вектор m_i. Пусть индекс c определяет ближайший к x вектор m_i, обозначенный дальше как m_c:

$$c = arg\ \min_i \{\|x - m_i\|\}\ .$$

Значения для m_i, что минимизируют ошибку классификации, могут быть найденные как асимптотичні значения в следующем процессе обучения. Пусть $x(t)$ – входная выборка, $m_i(t)$ – представления последовательности m_i, дискретизованой за временем. Начиная из правильно определенных начальных значений, основной процесс **алгоритма LVQ1** определяют следующие выражения:

$m_c(t + 1) = m_c(t) + \alpha(t)[x(t) - m_c(t)]$, если x и m_c належат один и одном и тому же классу;

$m_c(t + 1) = m_c(t) - \alpha(t)[x(t) - m_c(t)]$, если x и m_c належат разным классам;

$m_i(t + 1) = m_i(t)$, $\forall u \neq c$.

Здесь $0 < \alpha(t) < 1$, $\alpha(t)$ может быть константой или монотонно уменьшаться с течением времени.

Для алгоритма LVQ1 рекомендуется, чтобы первоначальное значение α было меньше 0.1.

Решения задачи классификации в **алгоритме LVQ2** идентично алгоритма LVQ1. Однако в процессе обучения LVQ2 два вектора с свободными параметрами m_i и m_j, что есть ближайшими соседями x, модифицируются одновременно. Один из них должен принадлежать к классу 1, а другой – к классу 2. Кроме того, x должен находиться в зоне значений, которое называется "окном", какое определенное вокруг середины плоскости, образованной векторами m_i и m_j. Пусть d_i и d_j – евклидовые расстояния x от m_i и m_j, тогда x определено попадет в "окно" относительной ширины w, если

$$min\left(\frac{d_i}{d_j}, \frac{d_j}{d_i}\right) > s, \qquad \text{где } s = \frac{1-w}{1+w}.$$

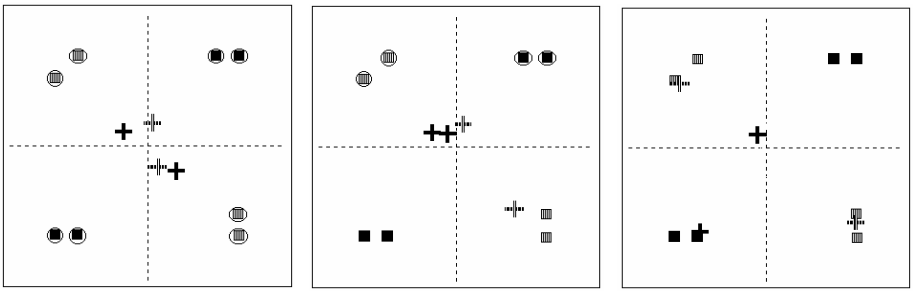
Рекомендуется, чтобы значения относительной ширины "окна" w находились в границах от 0.2 до 0.3.

Алгоритм обучения LVQ2 имеет вид:

$$m_i(t+1) = m_i(t) - \alpha(t)[x(t) - m_i(t)],$$
$$m_j(t+1) = m_j(t) + \alpha(t)[x(t) - m_j(t)],$$

где m_i и m_j – два ближайших к x вектору с свободными параметрами, причем x и m_j належат до одного и одного и того же класса, в то время как x и m_i належат разным классам, кроме того, x должный попадать в "окно".

На рис. 4 квадратами обозначенные значения признаков экземпляров, а крестами – значения весов НС LVQ. Штриховка фигур обозначает их принадлежность к классу (сплошная – класс 1, пунктирная – класс 2). По ошибке классифицированные экземпляры выделены кругами.



а) б) в)

Рисунок 4 – Процесс квантования учебных векторов

На рис. 4а) показанное начальное расположение весов НС LVQ – все экземпляры классифицированы неверно. На рис. 4б) для тех же экземпляров показанное положение весов НС LVQ после одной итерации обучения с помощью алгоритма LVQ2 – одновременно модифицированные значения двух векторов, количество неправильно классифицированных экземпляров уменьшилось. На рис. 4в) показанное положение весов НС LVQ после 10 итераций обучения – все экземпляры классифицированы верно.

2 ДОМАШНЯЯ ЗАДАЧА

Используя конспект лекций и рекомендованную литературу, выучить модели НС Кохонена и алгоритмы их обучения.

Используя документацию рекомендованного преподавателем программного средства, выучить его архитектуру и компоненты, предназначенные для моделирования и обучение НС Кохонена, ввода и вывода данных, подготовки отчетов (вывода и отображения результатов работы).

Ознакомиться с составом и порядком выполнения работы.

3 ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

3.1 Получить у преподавателя варианты задач для моделирования и обучение SOM и LVQ.

Входными данными есть: число классов, количество входов нейронов и размер сети (число нейронов); набор пар значений входов x_i и желательных выходов y_i; вид метрического свидетельства SOM.

3.2 Написать и отладить программу на Паскале, Си, что моделирует SOM и реализует алгоритмы ее обучения. Предусмотреть в программе отображения на экран и запись в файл на диске текущего состояния параметров НС и результатов их работы: матрицы весов, значения на входах и выходах, продолжительность их обучения и работы.

3.3 Для соответствующих варианту входных данных сделать обучение SOM и сохранить в файле на диске результаты ее работы.

3.4 Выполнить п.3.3, варьируя поочередно вид / параметры метрического свидетельства SOM. Результаты занести в таблицу,

столбцы которой должны иметь названия: вид метрического свидетельства, ошибка деления на классы (число ошибочных решений об отнесение экземпляров к узлам, к которым они не имеют фактической близости). Проанализировать полученные результаты и сделать выводы о том, как влияет вид метрического свидетельства на число ошибочных решений.

3.5 Выполнить п.п. 3.3-3.4 для программного средства, предложенного преподавателем. Сделать анализ и дать сравнительную характеристику разработанной Вами программы и программного продукта, предложенного преподавателем.

3.6 Используя программное средство, предложенное преподавателем, осуществить моделирование задачи, которая отвечает варианту, для НС LVQ. Результаты занести в таблицу, столбцы которой должны иметь названия: алгоритм обучения, время обучения, ошибка обучения, ошибка классификации.

3.7 Для тех же данных, что и в п. 3.6 осуществить моделирование на основе однослойного персептрона и НС Хопфилда. Результаты занести в таблицу.

3.8 Проанализировать результаты выполнения п.3.6-3.7 и сравнить возможности НС LVQ с НС Хопфилда и однослойным персептроном.

4 СОДЕРЖАНИЕ ОТЧЕТА

4.1 Сформулированная цель работы.

4.2 Короткие теоретические сведения и описание моделей НС Кохонена.

4.3 Схемы и описание алгоритмов работы и обучения НС Кохонена.

4.4 Описание и текст программы, которая реализует алгоритмы 4.3.

4.5 Входные данные и результаты работы программы 4.4.

4.6 Входные данные и результаты работы с программным средством, предложенным преподавателем.

4.7 Анализ полученных результатов и выводы.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Дайте определение понятий: нейронная сеть, метрическое свидетельство, ассоциативная память, SOM, LVQ.
2. Какие задачи можно решать на основе SOM и LVQ, а которые нельзя? Обоснуйте и докажите ответ. Приведите примеры.
3. Сравните возможности SOM, НС Хопфилда и однослойного дискретного персептрона.
4. Как влияют вид и параметры метрического свидетельства на точность определения центров сосредоточения экземпляров. Ответ объясните и проиллюстрируйте.
5. Предложите способы учета априорной информации о значимости признаков при обучении SOM.
6. Целесообразно ли использовать SOM для предыдущего анализа данных в задачах диагностики и прогнозирования?
7. Предложите стратегию (алгоритм) определения необходимого количества нейронов конкурирующего пласта LVQ.

Лабораторная работа № 4

МНОГОСЛОЙНЫЙ ПЕРСЕПТРОН. ОБОБЩЕННЫЙ ГРАДИЕНТНЫЙ АЛГОРИТМ ОБУЧЕНИЯ

Цель работы – выучить модель многослойного персептрона и методы его обучения; исследовать влияние шага обучения и вида корректирующего правила весов на продолжительность, точность обучения и классификации; ознакомиться с программными продуктами, которые моделируют многослойный персептрон.

1 КРАТКИЕ СВЕДЕНИЯ ИЗ ТЕОРИИ

Основным вычислительным элементом многослойного персептрона или многослойной нейронной сети (БНС) есть формальный нейрон. Он выполняет параметрическое нелинейное преобразование входного вектора x в скалярную величину y. Нейроны образовывают сеть, которая характеризуется следующими параметрами и свойствами: M – число пластов сети, N_μ – число нейронов μ-го пласта, связи между нейронами в пласте отсутствуют.

Выходы нейронов μ-го пласта, $\mu = 1, 2, ..., M-1$ поступают на входы нейронов только следующего μ+1-го пласта. Внешний векторный сигнал x поступает на входы нейронов только первого пласта, выходы нейронов последнего M-го пласта образовывают вектор выходов сети $y^{(M)}$. Структура сети показанная на рис. 5.

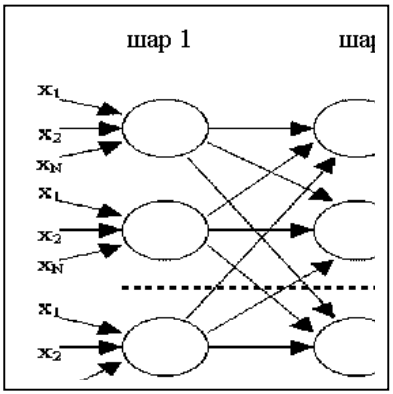


Рисунок 5 – Структура многослойной нейронной сети

Каждый i-й нейрон μ-го пласта (μi-й нейрон) превращает входной вектор $x^{(\mu,и)}$ в исходную скалярную величину $y^{(\mu,и)}$. Это преобразование состоит из двух этапов: сначала вычисляется дискриминантная функции $net^{(\mu,и)}$, которая дальше

превращается в исходную величину $y^{(\mu,и)}$.

Дискриминантная функция представляет собой отрезок многомерного ряда Тейлора. Коэффициенты разложения отрезка многомерного ряда Тейлора образовывают вектор весовых коэффициентов $w^{(\mu,и)}$, или память нейрона. Дискриминантная функция нейрона имеет вид:

$$net^{(м,i)} = w_0^{(м,i)} + \sum_{j=1}^{N} w_j^{(м,i)} x_j^{(м,i)},$$

где $w^{(\mu,и)} = (w_0^{(\mu,и)}, w_1^{(\mu,и)}, ..., w^{(\mu,и)})^T$ – вектор весовых коэффициентов нейрона; $x_j^{(\mu,и)}$ – *j-ая* компонента *N-измеримого* входного вектора $x^{(\mu,и)}$.

Нелинейное преобразование $y^{(\mu,и)} = \psi(net^{(\mu,и)})$ задается функцией активации, которая есть монотонной и ограниченной. В частности, при положительных или нулевых выходах нейрона такой функцией может быть сигмоидная функция $\psi(x) = 1/(1+e^{-x})$.

Обозначим через $y^{(\mu)} = (y^{(\mu,1)}, y^{(\mu,2)}, ..., y^{(\mu,N\mu)})^T$ вектор выхода нейронов μ-го пласта.

Процесс обучение сети, осуществляется в результате минимизации целевой функции – некоторого критерия качества $F(w)$, что характеризует интегральную меру близости выходов сети $y^{(M)}(k)$ и указаний учителя $y^*(k)$:

$$F(w) = \frac{1}{k}\sum_{m=1}^{k} Q(e(w,m)),$$

где k – номер текущего цикла обучения НС; m = 1, 2, ..., k – номера предыдущих циклов обучения НС; w – составленный вектор-столбец весовых коэффициентов сети, которые составляют векторы-столбцы $w^{(\mu)} = (w^{(\mu,1)T}, w^{(\mu,2)T}, ..., w^{(N\mu)T})^T$, $\mu = M, M-1, ..., 1$ каждого пласта. Мгновенный критерий качества $Q(\varepsilon(w, k))$, что входит в интегральный критерий качества $F(w)$, зависит от вектора ошибки сети $Q(\varepsilon(w, m))$: $\varepsilon(w, m) = y^{(M)}(m) - y^*(m)$.

Для каждого входного вектора x из учебного множества должное быть определенный вектор желательных выходов сети y^*. Если БНС, которая обучается, используется как классификатор, то обычно желательные выходы имеют низкий уровень (0 ли меньше 0,1), кроме выхода узла, который отвечает классу, к которого относится x; этот выход в данном случае имеет высокий уровень (1 ли

больше 0,9).

Градиентные методы обучения БНС основанные на использовании градиента целевой функции $F(w)$. Эти методы носят итеративный характер, так как компоненты градиента оказываются нелинейными функциями. Все дальше рассмотренные методы основаны на итерационной процедуре, которая реализуется соответственно формуле:

$$w_{k+1} = w_k + \alpha_k s(w_k),$$

где w_k , w_{k+1} – текущее и новое приближения значений весов и порогов НС к оптимальному решению, соответственно; α_k – шаг сходимости; $s(w_k)$ – направление поиска в *N-измеримому* просторные весов. Способ определения $s(w_k)$ и α_k на каждой итерации зависит от особенностей конкретного метода.

Обобщенный градиентный алгоритм относительно задачи **обучения БНС** имеет следующий вид.

Шаг 1. Инициализация: Задаются параметры БНС: N – число входов, M – число пластов, начальные веса и пороги w. Задаются параметры алгоритма обучения: максимально допустимое число циклов обучения *Epochs*, параметр сходимости алгоритма ε_1 – цель обучения (в качестве ее обычно выступает максимально допустимая среднеквадратичная ошибка), ε_2 – параметр сходимости вдоль прямой (для простоты можно считать $\varepsilon_2 = \varepsilon_1$).

Шаг 2. Положить счетчик итераций $k = 0$.

Шаг 3. Вычислить компоненты $\dfrac{\partial Q(e(w_{k-1},k))}{\partial w}$.

Шаг 4. Выполняется ли равенство $\left\| \dfrac{\partial Q(e(w_{k-1},k))}{\partial w} \right\| \le e_1$?

Так: сходимость достигнутая. Перейти на шаг 13.

Ни: перейти на шаг 5.

Шаг 5. Выполняется ли неравенство $k > Epochs$?

Так: достигнутое максимальное число циклов обучения, сходимость не достигнутая. Перейти на шаг 13.

Ни: перейти на шаг 6.

Шаг 6. Вычислить $s(w_k)$.

Шаг 7. Выполняется ли неравенство $\dfrac{\partial Q(e(w_{k-1},k))^T}{\partial w} s(w_k) < 0$?

Так: перейти на шаг 9.

Ни: положить: $s(w_k) = -\frac{\partial Q(e(w_{k-1}, k))}{\partial w}$. Перейти на шаг 9.

Шаг 8. Найти такое значение α_k, при котором $F(w_k + \alpha_k s(w_k)) \to min$, используя параметр ε_2.

Шаг 9. Положить $w_{k+1} = w_k + \alpha_k s(w_k)$.

Шаг 10. Выполняется ли неравенство $F(w_{k+1}) < F(w_k)$?

Так: перейти на шаг 11.

Ни: перейти на шаг 13.

Шаг 11. Выполняется ли неравенство $\frac{\|Дw\|}{\|w_k\|} \le e_1$?

Так: окончания поиска: нет продвижения к решению. Перейти на шаг 13.

Ни: перейти на шаг 12.

Шаг 12. Положить $k = k + 1$. Перейти на шаг 3.

Шаг 13. Останов.

Следует отметить, что в процессе одномерного поиска необходимо, по возможности, избегать точных вычислений, так как эксперименты показывают, что на выполнение операций поиска вдоль прямой расходуется очень значительная часть общего времени вычислений.

В вышеописанном обобщенном градиентном алгоритме можно использовать разные градиентные методы путем определения соответствующих направлений поиска на шаге 6. Определяя соответствующим образом $s(w_k)$, можно трансформировать рассмотренный обобщенный градиентный алгоритм практически в любой градиентный алгоритм обучения БНС, которое разрешает существенным образом упростить разработку нейросетевых систем без жесткой привязки к определенного градиентного алгоритму.

Рассмотрим некоторые частные случаи обобщенного градиентного алгоритма.

Метод Коши (метод быстрейшего спуска, в обучении нейронных мереж известный, как алгоритм обратного распространения ошибки первого порядка или Backpropagation) состоит в реализации правила (шаг 6 обобщенного градиентного алгоритма):

$$s(w_k) = -\varepsilon \nabla_w Q(e(w_k, k)) = -\varepsilon \frac{\partial Q(e(w_k, k))}{\partial w}.$$

Обозначив текущий градиент $g = \frac{\partial Q}{\partial w}$,

получим: $s(w_k) = -\gamma_k g_k$, где γ_k – скорость обучения.

Величина γ или полагается постоянной, и тогда обычно последовательность w_k совпадает у окружающей среды оптимального значения w, или она есть нисходящей функцией времени так, как это делается в стохастических алгоритмах оптимизации и адаптации.

Данная процедура может выполняться до тех пор, пока значения управляемых сменных не стабилизируются или пока ошибка не уменьшится к приемлемому значению. Следует отметить, что данную процедуру характеризует медленная скорость сходимости и возможность попадания в локальные минимумы функционала.

Для **метода Ньютона** или алгоритма обратного распространения ошибки второго порядка шаг 6 обобщенного градиентного алгоритма имеет вид:

$$s(w_k) = -\left(\sum_{m=1}^{k} \frac{\partial}{\partial w} \left(\frac{\partial Q(m)}{\partial w} \right)^{\Phi} \right)^{-1} g_k.$$

Алгоритмы сопряженных градиентов представляют собою подкласс методов, которые квадратично совпадают. Для алгоритмов сопряженных градиентов шаг 6 обобщенного градиентного алгоритма имеет вид:

$$s(w_k) = -g_k + \beta_k s(w_{k-1}).$$

В большинства алгоритмов сопряженных градиентов размер шагу корректируется при каждой итерации, в отличие от других алгоритмов, где скорость, которая учится, используется для определения размера шагу. Разные версии алгоритмов сопряженных градиентов отличаются способом, за которым вычисляется константа β.

Для **алгоритма сопряженных градиентов Флетчера-Ривса** правило вычисления константы β имеет вид:

$$в_k = \frac{g_k^T g_k}{g_{k-1}^T g_{k-1}},$$

где β – отношения квадрата нормы текущего градиента к квадрату нормы предыдущего градиента.

В обобщенном градиентном алгоритме отсутствует процедура возвращения к начальной итерации для метода Флетчера-Ривса, но вместе с тем тесты, включенные в алгоритм, обеспечивают выявление любой трудности, ассоциированных с необходимостью возвращения при расчетах по метода соединенных градиентов.

Для **алгоритма спряжених градиентов Полака - Рибьера** правило вычисления константы β имеет вид:

$$в_k = \frac{\Delta g_{k-1}^T g_k}{g_{k-1}^T g_{k-1}},$$

где β – внутреннее произведение предыдущего изменения в градиенте и текущего градиента, разделенное на квадрат нормы предыдущего градиента.

Алгоритм Левенберга - Марквардта требует наличия информации о значение производных целевой функции. В алгоритме Левенберга- Марквардта используется метод обратного распространения ошибки первого порядка, чтобы вычислить якобиан J целевой функции относительно весов и порогов сети.

Для алгоритма Левенберга- Марквардта шаг 6 обобщенного градиентного алгоритма имеет вид:

$$s(w_k) = -[H_k + \eta I]^{-1} g_k,$$

где $H = J^T J$; J – якобиан; $g_k = J^T e$ – текущий градиент; e – вектор ошибок; η – скаляр; I – единичная матрица.

Адаптивное значение η увеличивается в η^+ раз до тех пор, пока значения целевой функции не уменьшится. После чего изменения вносятся в сеть и η уменьшается в η^- раз.

2 ДОМАШНЯЯ ЗАДАЧА

Используя конспект лекций и рекомендованную литературу, выучить модель БНС и алгоритмы ее обучения.

Используя документацию рекомендованного преподавателем программного средства, выучить его архитектуру и компоненты, предназначенные для моделирования и обучение БНС, ввода и вывода данных, подготовки отчетов (вывода и отображения результатов работы).

Ознакомиться с составом и порядком выполнения работы.

3 ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

3.1 Получить у преподавателя варианты задач для моделирования многослойного персептрона. Входными данными есть: число пластов и число нейронов в каждом пласте; вид функции активации нейронов; количество входов и выходов персептрона; набор пар значений входов x_i и желательных выходов y_i персептрона; максимально допустимая ошибка обучения; шаг обучения.

3.2 Используя предложенное преподавателем программное средство, осуществить обучение и моделирование многослойного персептрона для соответствующих варианту данных.

3.3 Несколько раз изменить вид функции активации нейрона и выполнить п.3.2

3.4 Несколько раз изменить алгоритм обучения и выполнить п.3.2-3.3.

3.5 Результаты моделирования занести в таблицу, столбцы которой должны иметь названия: алгоритм обучения, вид функции активации нейронов, ошибка обучения, ошибка распознавания, время обучения, время классификации персептрона.

3.6. Проанализировать полученные результаты и сделать выводы о том, как влияют вид функции активации и вид корректирующего правила весов (алгоритм обучения) на время обучения и классификации , а также величину ошибки обучения / классификации многослойного персептрона.

4 СОДЕРЖАНИЕ ОТЧЕТА

4.1 Сформулированная цель работы.

4.2 Короткие теоретические сведения, ответа на контрольные вопросы и описание модели многослойного персептрона.

4.3 Схема и описание обобщенного градиентного алгоритма.

4.4 Входные данные и результаты моделирования.

4.5 Анализ полученных результатов и выводы. Лаконичные ответы на контрольные вопросы.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Дайте определение понятий: формальный нейрон, вес, функция активации, персептрон, желательный и реальный выход персептрона, обучения и работа персептрона, классификация, аппроксимация, ошибка обучения / классификации, целевая функция обучения.
2. Влияет ли вид функции активации на время обучения и классификации персептрона, величину ошибки обучения и, если так, то как?
3. Какие функции активации чаще всего используют и чему?
4. Какие задачи можно решать на основе многослойных персептронов, а какие нельзя? Обоснуйте ответ. Приведите примеры.
5. Можно ли научить дискретный многослойный персептрон вычислять значение функций: 1) $y = x_1$ ***and*** x_2, 2) $y = x_1$ ***xor*** x_2, 3) $y =$ ***not*** ((x_1) ***and*** x_2), а действительный многослойный персептрон – вычислять значение функций: 1) $y = 3x_{1\text{-}0}.05x_2$, 2) $y = sin(x_1) + 0.3x_2$, 3) $y = 0.5x_1 + 2x_{2\text{-}2}.5(x_1 - x_2) + 5.5$, 4) $y = x_1 / (sin(\pi) * x_2)$?
6. Может ли действительный многослойный персептрон моделировать функцию $y = x_1^2 + 0.5x_1 + x_2$, если число его пластов равняется: а) 1, б) 2, в) 3? Ответа обоснуйте и, если возможно, объясните рисунками.
7. Опишите суть обобщенного градиентного алгоритма.
8. Какие методы обучения БНС Вам известны? Дайте сравнительную характеристику известных Вам методов обучения БНС.
9. Всегда ли совпадает метод Коши (алгоритм обратного распространения ошибки) для многослойного персептрона?
10. Что из методов обратного распространения ошибки Вы рекомендовали бы для применения, если критерием оптимальности обучения НС есть: а) скорость обучения, б) простота вычислений, в) универсальность?
11. Возможно ли использования неградиентных методов многомерной безусловной оптимизации для настраивания весов БНС, и, если возможно, то насколько это целесообразно делать?
12. Влияет ли объем обучающей выборки на скорость обучения БНС? Влияет ли количество используемых признаков на скорость обучения однослойного персептрона?

13. Что такое репрезентативная выборка данных?
14. Должна ли обучающая выборка быть репрезентативной?
15. Должна ли тестовая выборка быть репрезентативной?
16. Влияет ли репрезентативность обучающей выборки на точность классификации экземпляров тестовой выборки?
17. Влияет ли репрезентативность тестовой выборки на точность классификации экземпляров тестовой выборки?
18. Влияет ли репрезентативность тестовой выборки на точность обучения персептрона по обучающей выборке?

Лабораторная работа № 5

РАДИАЛЬНО-БАЗИСНЫЕ НЕЙРОННЫЕ СЕТИ

Цель работы – выучить модель радиально-базисной сети и методы ее обучения; сравнить радиально-базисную сеть с многослойным персептроном по скорости обучения и точности классификации; ознакомиться с программными продуктами, которые моделируют радиально-базисные сети.

1 КРАТКИЕ СВЕДЕНИЯ ИЗ ТЕОРИИ

Радиально-базисная НС (РБНС) состоит из двух пластов. Соединительные весовые векторы пластов будем обозначать $w^{(\mu,j)}$, где μ – номер пласта (μ = 1, 2), j – номер нейрона (узла) в пласте.

Базисные (или ядерные) функции в первом пласте вырабатывают локализованную реакцию на входной стимул. Исходные узлы сети формируют взвешенную линейную комбинацию из базисных функций, вычисленных узлами первого пласта.

Исходные узлы отвечают исходным классам, в то время, как узлы первого пласта представляют собою кластеры (количество кластеров m задается пользователем), на которые разбивается входное пространство. Обозначим $x = (x_1, ..., x_i, ..., x)$ и $y = (y_1, ..., y_i, ..., y)$ – вход и выход сети, соответственно. Здесь N – количество признаков, а K – число классов.

Выход u_j j-го узла первого пласта, используя ядерную функцию Гауссіан как базисную, определяется по формуле:

$$u_j = exp\left[-\frac{\left(x - w^{(1,j)}\right)^T \left(x - w^{(1,j)}\right)}{2\sigma_j^2}\right], \qquad j=1, 2, ..., m,$$

где x – входной образ (экземпляр);

$w^{(1,j)}$ – его входной весовой вектор (то есть центр Гауссіана для узла j);

σ^2_j – параметр нормализации j-го узла, такой что $0 < u_j < 1$ (чем ближе вход к центру Гауссіана, та сильнее реакция узла).

Выход y_j j-го узла второго пласта определяется из выражения:

$$y_j = w^{(2,j)T} u, j = 1, 2, ..., K;$$

где $w^{(2,j)}$ – весовой вектор для j-го узла второго пласта и u –

вектор выходов первого пласта.

Сеть выполняет линейную комбинацию нелинейных базисных функций.

Задача обучение сети складывается в минимизации ошибки:

$$E = \frac{1}{2}\sum_{s=1}^{S}\sum_{j=1}^{K}\left(y_j^s - y_j^{s*}\right)^2,$$

где y^{s*}_{j} и y^{s}_{j} – желательное и расчетное значения выхода j-го узла исходного пласта для s-го экземпляра; S – размер набора данных (количество экземпляров); K – число исходных узлов (число классов). Дале для наглядности верхний индекс s опущен.

Обучения РБНС может выполняться двумя разными способами.

Первый способ состоит в том, что алгоритмом кластеризации формируется фиксированное множество центров кластеров. Потом минимизацией квадратичной ошибки, то есть минимизацией E, получают ассоциации центров кластеров с выходом.

Второй способ состоит в том, что центры кластеров могут быть также научены рядом с весами от первого пласта к исходному пласту методом градиентного спуска. Однако, обучение центров рядом с весами может привести к попаданию сети в локальные минимумы.

Пусть центры кластеров будут обозначены $w^{(1,\text{j})}$, j = 1, ..., m. Параметр нормализации σ_{j} представляет меру распределения данных, которые ассоциируются с каждым узлом.

Обучения в исходном пласте выполняется после того, как определенные параметры базисных функций. Веса обычно учат, используя алгоритм среднеквадратичных отклонений:

$$\Delta w^{(\mu,j)} = -\eta e_j u,$$

где $e_{\text{j}} = y_{\text{j}} - y^*_{\text{j}}$; η – коэффициент скорости (шаг) обучения.

2 ДОМАШНЯЯ ЗАДАЧА

Используя конспект лекций и рекомендованную литературу, выучить модель радиально-базисной сети и алгоритмы ее обучения.

Используя документацию рекомендованного преподавателем программного средства, выучить его архитектуру и компоненты, предназначенные для моделирования и обучение РБНС, ввода и

вывода данных, подготовки отчетов (вывода и отображения результатов работы).

Ознакомиться с составом и порядком выполнения работы.

3 ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

3.1 Получить у преподавателя варианты задач для обучения и моделирование РБНС. Входными данными есть обучающая и контрольная выборки (наборы пар значений входов и желательных выходов персептрона).

3.2 Используя предложенное преподавателем программное средство, осуществить обучение и моделирование РБНС и БНС для соответствующих варианту данных.

3.3 Результаты моделирования занести в таблицу, столбцы которой должны иметь названия: модель НС, алгоритм обучения, ошибка обучения, ошибка распознавания, время обучения, время классификации.

3.4. Проанализировать полученные результаты и сделать выводы о том, какая модель НС лучше подходит для решения задачи, которая отвечает варианту.

4 СОДЕРЖАНИЕ ОТЧЕТА

4.1 Сформулированная цель работы.

4.2 Короткое описание модели РБНС.

4.3 Схема и описание алгоритмов обучения РБНС.

4.4 Короткое описание, входные данные и результаты работы с программным средством, предложенным преподавателем, таблица 3.3.

4.5 Анализ полученных результатов и выводы. Лаконичные ответы на контрольные вопросы.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Дайте определение и объясните взаимосвязь понятий: нейрон, вес, порог, радиально-базисная сеть, классификация, аппроксимация, оценивания, ошибка обучения / классификации, время обучения / классификации.
2. В чем сходство и отличие РБНС и БНС?

3. Какие задачи можно решать на основе РБНС, а которые нельзя? Обоснуйте и докажите ответ. Приведите примеры. Целесообразно ли применять РБНС для классификации сложно (нелинейно) раздельных образов?
4. Всегда ли совпадают алгоритмы обучения РБНС?
5. Можно ли научить РБНС вычислять значение функций: 1) $y = x_1$ ***and*** x_2, 2) $y = x_1$ ***xor*** x_2, 3) $y =$ ***not*** $(($***not*** $x_1)$ ***and*** $x_2)$, 4) $y = 3x_{1\text{-}0},05x_2$, 5) $y = sin(x_1) + 0,3x_2$, 6) $y = 0,5x_1 + 2x_{2\text{-}2},5(x_1 - x_2) + 5,5$, 7) $y = x_1 / (sin(\pi) * x_2)$?

Лабораторная работа № 6

ЭВРИСТИЧЕСКИЕ МОДЕЛИ И МЕТОДЫ ОБУЧЕНИЯ НЕЙРОННИХ СЕТЕЙ

Цель работы – изучить эвристические модели и методы обучения НС; сравнить эвристические методы обучения НС между собой, а также методы обучения БНС; рассмотреть примеры практического использования изученных методов и моделей НС.

1 КРАТКИЕ СВЕДЕНИЯ ИЗ ТЕОРИИ

Эвристический алгоритм обучение классификации двухслойного персептрона

Пусть мы имеем учебную выборку $x = \{x^1, x^2, ..., x\}$, что состоит из S экземпляров, которые характеризуются N признаками x^q_i, где q – номер экземпляра обучающей выборки, $и$ – номер признака.

Каждому экземпляру обучающей выборки сопоставленный номер класса y^q, $\forall y^q \in \{K_0, K_1\}$, где K_0 и K_1 – условные пометки разных классов экземпляров. Условимся, что экземпляр, который проверяется, належит к классу K_0, если значения параметра, который прогнозируется, к моменту времени прогнозирования будет больше некоторого предельного значения. В противном случае экземпляр належит к классу K_1.

Одним из методов настраивания весов для двухслойного персептрона может служить эвристический алгоритм, модель НС для которого представленная на рис. 6.

Параметры и функции активации НС необходимо определить по следующим правилам.

Функция активации $\psi^{(М,и)}$ i-го нейрона $м$-го пласта:

$$ш^{(1,i)}(x) = \frac{1}{1+e^{-x}}, \quad \forall i = \overline{1,N}; \qquad ш^{(2,1)}(x) = \begin{cases} 0, x \le 0, \\ 1, x > 0. \end{cases}$$

Весовой коэффициент $w_j^{(мi)}$ j-го входа i-го нейрона $м$-го пласта:

$$w_j^{(мi)} = \begin{cases} 0, & i \ne j,\ j > 0,\ м = 1, \\ в_i, & i = j,\ j > 0,\ м = 1, \end{cases} \quad w_j^{(мi)} = \begin{cases} -u_i в_i, & j = 0,\ м = 1, \\ б_i, & м = 2, \end{cases}$$

$i = 1, 2, ..., N, \quad j = 0, 1, 2, ..., N.$

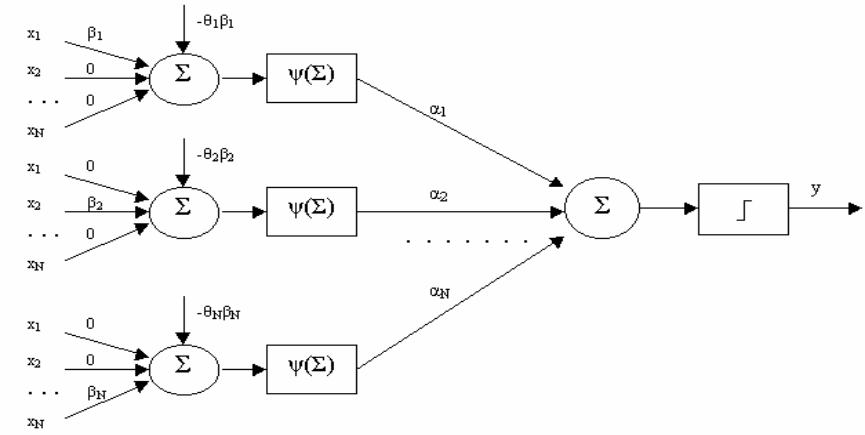


Рисунок 6 – Двухслойный персептрон

Для нахождения значений параметров $\alpha_и$, $\beta_и$ и $\theta_и$ предлагается использовать следующий алгоритм.

Шаг 1. Вычислить:

M_{x_i} – математическое ожидание *i*-ї признака x_i:

$$M_{x_i} = \frac{1}{S}\sum_{q=1}^{S} x_i^q \ , \ u = 1, 2, ..., N,$$

где S – количество экземпляров обучающей выборки, x^q_i – значения *i*-ї признака *q*-го экземпляра обучающей выборки.

$M_{x_i}^{K_1}$ – математическое ожидание *i*-ї признака для экземпляров обучающей выборки, которые належат к классу K_1:

$$M_{x_i}^{K_1} = \frac{1}{S^{K_1}}\sum_{q=1}^{S^{K_1}} x_i^q, x_i^q \in K_1, \ u = 1, 2, ..., N,$$

где S^{K_1} – количество экземпляров обучающей выборки, которые належат к классу K_1.

$M_{x_i}^{K_0}$ – математическое ожидание *i*-ї признака для экземпляров обучающей выборки, которые належат к классу K_0:

$$M_{x_i}^{K_0} = \frac{1}{S^{K_0}} \sum_{q=1}^{S^{K_0}} x_i^q, x_i^q \in K_0, u = 1, 2, ..., N,$$

где S^{K_0} – количество экземпляров обучающей выборки, которые належат к классу K_0.

M_y – математическое ожидание номера класса: $M_y = \frac{1}{S} \sum_{q=1}^{S} y^q$,

где y^q – номер класса q-го экземпляра обучающей выборки.

Если необходимо, для нахождения порога $\theta_и$, вычислить:

Дисперсии признаков: $D_{x_i} = \frac{1}{S} \sum_{q=1}^{S} \left(x_i^q - M_{x_i}\right)^2$, $u = 1, 2, ..., N$.

Дисперсии признаков экземпляров, которые относятся к классу K_1:

$$D_{x_i}^{K_1} = \frac{1}{S^{K_1}} \sum_{q=1}^{S^{K_1}} \left(x_i^q - M_{x_i}^{K_1}\right)^2, x_i^q \in K_1, u = 1, 2, ..., N.$$

Дисперсии признаков экземпляров, которые относятся к классу K_0:

$$D_{x_i}^{K_0} = \frac{1}{S^{K_0}} \sum_{q=1}^{S^{K_0}} \left(x_i^q - M_{x_i}^{K_0}\right)^2, x_i^q \in K_0, u = 1, 2, ..., N.$$

Дисперсию номера класса: $D_y = \frac{1}{S} \sum_{q=1}^{S} \left(y^q - M_y\right)^2$.

Шаг 2. Вычислить коэффициенты корреляции каждой i-ї признака и номера класса: $r_{x_i y} = \frac{\sum_{q=1}^{S} \left(x_i^q - M_{x_i}\right)\left(y^q - M_y\right)}{\sqrt{\sum_{q=1}^{S} \left(x_i^q - M_{x_i}\right)^2 \sum_{q=1}^{S} \left(y^q - M_y\right)^2}}$.

Шаг 3. Вычислить степень (частицу) влияния *i*-ї признака на номер класса экземпляра: $б_i = \dfrac{|\, r_{x_i y}\,|}{\sum_{j=1}^{N} |\, r_{x_j y}\,|}$, $i = 1, 2, ..., N$,

где j – номер текущего признака.

Вычислить коэффициент $\beta_и$, что учитывает наиболее вероятное размещение полюсов (центров сосредоточения экземпляров) классов при одномерной классификации за *i*-ю признаком: $в_i = sign\left(M_{x_i}^{K_1} - M_{x_i}^{K_0}\right)$.

Этот коэффициент будет равняться: +1, если полюс класса K_0 расположенный по левую сторону полюса класса K_1 по вехе значений *i*-ї признака; –1, если полюс класса K_0 расположенный по правую сторону полюса класса K_1 по вехе значений *i*-ї признака; 0, если полюса классов совпадают.

Вычислить значение порога, относительно которого будем осуществлять одномерную классификацию экземпляров за *i*-ю признаком. Для нахождения значения порога можно предложить довольно много разных способов:

$$u_i = \frac{|\, M_{x_i}^{K_1} - M_{x_i}^{K_0}\,|}{2} + min\left(M_{x_i}^{K_1}, M_{x_i}^{K_0}\right),\ i = 1, 2, ..., N;$$

$$u_i = M_{x_i} + \frac{D_{x_i(0.5-M_y)}}{r_{x_i y} D_y},\ i = 1, 2, ..., N,$$

$$u_i = max\left(M_{x_i}^{K_1}, M_{x_i}^{K_0}\right) - |\, M_{x_i}^{K_1} - M_{x_i}^{K_0}\,| \Big/ \left(1 + D_{x_i}^{K_0} \Big/ D_{x_i}^{K_1}\right),\ u = 1, 2, ..., N.$$

Шаг 4. Оценить вероятности ошибки переименования классов для экземпляров обучающей и (или) контрольной выборок и сделать вывод о применимости данного алгоритма для решения поставленной задачи.

Для этого следует найти значение номеров классов для экземпляров обучающей и (или) контрольной выборок по правилу классификации (3), после чего определить количество ошибочных решений $S_{пом}$ и оценить вероятность принятия ошибочных решений $P_{пом} = S_{пом} / (S + S_к)$, где S и $S_к$ – размер обучающей и контрольной выборок, соответственно.

Эвристический алгоритм обучение классификации трехслойного персептрона

Пусть заданная обучающая выборка, которая состоит из S экземпляров x^q, $q = 1, 2, ..., S$, которые характеризуются N признаками x^q_i, и = 1, 2, ..., N, и каждому x^q сопоставленный класс A ли B.

Исходя из предположение, что экземпляры одного класса вероятнее всего будут расположены ближе в просторные признаков, определим для каждого класса и каждого признака координаты центров сосредоточения (центров веса) экземпляров.

Координата центра сосредоточения экземпляров, которые належат к классу A, за i-ю признаком $C^A_{x_i}$ будет определяться из выражения:

$$C^A_{x_i} = \frac{1}{N^A}\sum_{q=1}^{N^A} x_i^q,\ x_i^q \in A,$$

где N^A – количество экземпляров, которые належат к классу A. Аналогичным чином может быть определенная координата центра сосредоточения экземпляров, которые належат к классу B, за i-ю признаком $C^B_{x_i}$.

Зная координаты центров сосредоточения экземпляров обучающей выборки для обеих классов, можно осуществлять классификацию по расстоянию нового экземпляра от этих центров в *N-измеримой* системе координат.

Для этого для каждого нового экземпляра x^q последовательно находятся расстояния этого экземпляра от центров сосредоточения экземпляров для каждого из классов:

$$R_A^2 = \sum_{i=1}^{N}\left(C^A_{x_i} - x_i^q\right)^2, \quad R_B^2 = \sum_{i=1}^{N}\left(C^B_{x_i} - x_i^q\right)^2.$$

Новый экземпляр относят к классу A, если $R^2_A < R^2_B$, в противоположном случае – к классу B.

Такой способ классификации – сравнения расстояний нового экземпляра от центров сосредоточения экземпляров обучающей выборки по всем координатам (признаках) одновременно. При этом не учитывается информация о значимости признаков. Для устранения

этого недостатка представим класс экземпляра K как функцию суммы результатов частичной классификации за i-ю и j-ю признаками K_{ij} с учетом их значимостей $б_{ij}$: $K = u\left(\sum_{i,j}^{N} б_{ij} K_{ij}\right)$, где $u(x) = \begin{cases} 1, x > 0, \\ 0, x \le 0. \end{cases}$

Условимся, что класс A будет кодироваться значением 1, а класс B – значением 0.

Для определения результатов частичной классификации за i-ю и j-ю признаками K_{ij} для экземпляра x^q найдем расстояния этого экземпляра от центров сосредоточения экземпляров, которые належат классам A и B на плоскости, образованной i-ю и j-ю признаками:

$$R^2_{C^A_{ij}}\left(x^q\right) = \left(C^A_{x_i} - x^q_i\right)^2 + \left(C^A_{x_j} - x^q_j\right)^2,\ R^2_{C^B_{ij}}\left(x^q\right) = \left(C^B_{x_i} - x^q_i\right)^2 + \left(C^B_{x_j} - x^q_j\right)^2.$$

Если $R^2_{C^A_{ij}}\left(x^q\right) < R^2_{C^B_{ij}}\left(x^q\right)$, то будем считать, что на плоскости (u,j) $x^q \in A$, иначе $x^q \in B$. То есть, если $R^2_{C^A_{ij}}\left(x^q\right) < R^2_{C^B_{ij}}\left(x^q\right)$, то $K_{ij} = 1$, иначе – $K_{ij} = 0$.

Для нахождения значимостей $б_{ij}$ результатов частичной классификации за i-ю и j-ю признаками для всех экземпляров обучающей выборки определим количество ошибочных решений при двумерной классификации за i-ю и j-ю признаками:

$$N_{пом\,K_{ij}} = \sum_{q=1}^{s}\left(\left|K^q_{ij} - K^q\right|\right),$$

где K^q – значения, сопоставленное класса q-го экземпляра; K^q_{ij} – результат двумерной классификации q-го экземпляра за i-ю и j-ю признаками.

Потом найдем значимости $б_{ij}$ результатов частичной классификации за i-ю и j-ю признаками:

$$б_{ij} = \left(1 - \frac{N_{помK_{ij}}}{s}\right) \Big/ \sum_{i,j}^{N}\left(1 - \frac{N_{помK_{ij}}}{s}\right).$$

Для упрощения вычислений можно предложить альтернативный вариант установки значений $б_{ij}$: $б_{ij} = 1 \Big/ \sum_{i,j} 1$.

Такой вариант немного ускорит работу алгоритма, но при этом значимости частичных результатов классификации учитываться не будут.

Поскольку результаты классификации $K_{ij} = K_{ji}$, то для оптимизации вычислительного процесса при обучении и распознавании зададим области определения для u и j: $u, j \in [1, 2, ..., N]$, $\forall\ u \le j$.

В этом случае при вычислении результата классификации будут использоваться частичные результаты двумерных классификаций за i-ю и j-ю признаками ($i \ne j$) и результаты одномерной классификации за i-ю (j-ю) признаком ($i = j$).

Очевидно, если положить: $u, j \in [1, 2, ..., N]$, $\forall\ u < j$, то будут учитываться только частичные результаты двумерной классификации за i-ю и j-ю признаками ($i \ne j$).

В свою очередь, если положить: $u, j \in [1, 2, ..., N]$, $\forall\ u = j$, то будут учитываться только частичные результаты одномерной классификации за i-ю (j-ю) признаком ($i = j$).

Для настраивания весов трехслойного персептрона может служить следующий алгоритм.

Для нейросетевой реализации сравнения расстояний и определения значения K_{ij} можно использовать следующее выражение:

$$K_{ij} = u\left(R^2_{C^B_{ij}}\left(x^q\right) - R^2_{C^A_{ij}}\left(x^q\right)\right),$$

где $u(x)$ – логистическая функция.

Если функция $u(x)$ будет дискретной, например, пороговою: $u(x) = \begin{cases} 1, x > 0, \\ 0, x \le 0. \end{cases}$, то K_{ij} будет принимать значение 0 или 1. Если функция $u(x)$ будет действительной, например, сигмоидной: $u(x) = \frac{1}{1 + e^{-x}}$, то K_{ij} будет принимать значение на интервале [0,1]: чем более близкое значение этой функции будет до 0, тот ближе экземпляр будет к классу, какому сопоставленное значение 0, и, соответственно, наоборот, чем более близкое значение этой функции будет до 1, тот ближе экземпляр будет к классу, какому сопоставленное значение 1. Использования сигмоидной функции может быть более лучшей на

практике, поскольку она разрешает не только определить к какому классу ближе экземпляр, но и на сколько ближе.

Для вычисления различия расстояний $R^2_{C^B_{ij}}\left(x^q\right) - R^2_{C^A_{ij}}\left(x^q\right)$ подставим соответствующие выражения:

$$\mathrm{R}^2_{C^{\mathrm{B}}_{ij}}\left(\mathrm{x}^{\mathrm{q}}\right) - \mathrm{R}^2_{C^{\mathrm{A}}_{ij}}\left(\mathrm{x}^{\mathrm{q}}\right) = \left(\mathrm{C}^{\mathrm{B}}_{\mathrm{x_i}} - \mathrm{x}^{\mathrm{q}}_{\mathrm{i}}\right)^2 + \left(\mathrm{C}^{\mathrm{B}}_{\mathrm{x_j}} - \mathrm{x}^{\mathrm{q}}_{\mathrm{j}}\right)^2 - \left(\mathrm{C}^{\mathrm{A}}_{\mathrm{x_i}} - \mathrm{x}^{\mathrm{q}}_{\mathrm{i}}\right)^2 - \left(\mathrm{C}^{\mathrm{A}}_{\mathrm{x_j}} - \mathrm{x}^{\mathrm{q}}_{\mathrm{j}}\right)^2 ,$$

раскроем дужки, сгруппируем члены по *u* и *j* и приведем подобные. После нескладных математических преобразований получим:

$$R^2_{C^B_{ij}}\left(x^q\right) - R^2_{C^A_{ij}}\left(x^q\right) = \tilde{i} + \tilde{j} ,$$

где $\tilde{i} = \left(C^B_{x_i}\right)^2 - \left(C^A_{x_i}\right)^2 + 2x^q_i\left(C^A_{x_i} - C^B_{x_i}\right)$, $\tilde{j} = \left(C^B_{x_j}\right)^2 - \left(C^A_{x_j}\right)^2 + 2x^q_j\left(C^A_{x_j} - C^B_{x_j}\right)$.

Легко видеть, что выражения для $\tilde{i}$ и $\tilde{j}$ могут быть вычислены на основе формального нейрона, который имеет один вход, на который подается значения x_{i} или x_{j}, вес которого равняется $2\left(C^A_{x_i} - C^B_{x_i}\right)$ или $2\left(C^A_{x_j} - C^B_{x_j}\right)$, соответственно. Порог нейрона (нулевой вес) в этом случае будет равняться $\left(C^B_{x_i}\right)^2 - \left(C^A_{x_i}\right)^2$ или $\left(C^B_{x_j}\right)^2 - \left(C^A_{x_j}\right)^2$, соответственно.

Трехслойный персептрон, веса которого вычисленные на основе рассмотренного алгоритма, изображенный на рис. 7.

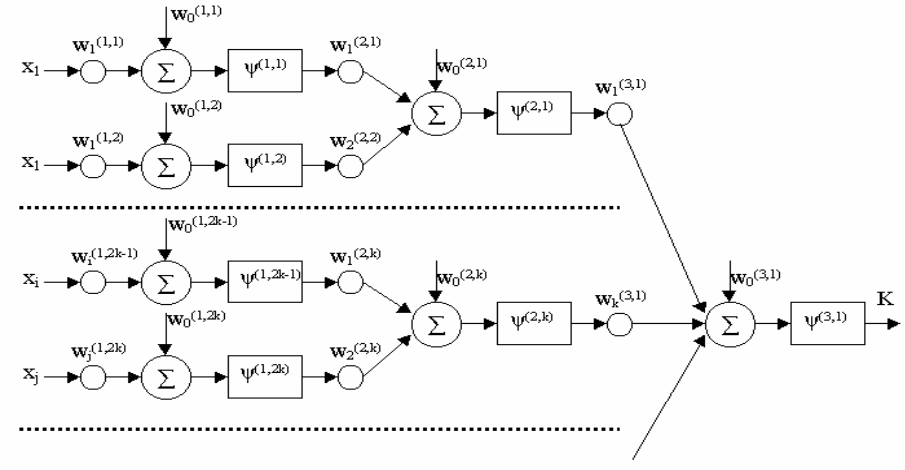


Рисунок 7 – Трехслойный персептрон

Параметры и функции активации НС определяют по следующим правилам.

Функция активации $\psi^{(M,k)}$ k-го нейрона м-го пласта:

$$u^{(мk)}(x) = \frac{1}{1+e^{-x}}, \quad м = 1,2; \forall k;$$

$$u^{(3,1)}(x) = \begin{cases} 0, x \le 0, \\ 1, x > 0, \end{cases}$$

Весовой коэффициент $w_p^{(мk)}$ p-го входа k-го нейрона $м$-го пласта:

$$w_p^{(мk)} = \begin{cases} б_{ij}, м = 3, p = j + (i-1)(N - 0.5i), \\ 0, м = 3, k = 1, p = 0, \\ 0, м = 2, \forall k, p = 0, \\ 1, м = 2, \forall k, p = 1,2, \\ 2\left(C_{x_i}^A - C_{x_i}^B\right), м = 1, k = 2\left(j + (i-1)(N - 0.5i)\right) - 1,\ p = i, \\ 2\left(C_{x_j}^A - C_{x_j}^B\right), м = 1, k = 2\left(j + (i-1)(N - 0.5i)\right),\ p = j, \\ \left(C_{x_i}^B\right)^2 - \left(C_{x_i}^A\right)^2, м = 1, k = 2\left(j + (i-1)(N - 0.5i)\right) - 1,\ p = 0, \\ \left(C_{x_j}^B\right)^2 - \left(C_{x_j}^A\right)^2, м = 1, k = 2\left(j + (i-1)(N - 0.5i)\right),\ p = 0, \\ 0, м = 1, k = 2\left(j + (i-1)(N - 0.5i)\right) - 1,\ p > 0, p \neq i, \\ 0, м = 1, k = 2\left(j + (i-1)(N - 0.5i)\right),\ p > 0, p \neq j, \end{cases}$$

$$\forall i, j: \ \ i = 1,2,....,N; \ \ j = i,(i+1),...,N .$$

Алгоритм обучения шестислойного персептрона

Признаки, которые характеризуют объект (процесс), что моделируется, будем условно разделять на значащие и незначащие. КР значащих будем относить те признаки, которые тесно связанны с исходным параметром и пренебрежение которыми может существенным образом ухудшить модель, к незначащих – те признаки, которые слабо связанны (или вообще не связанны) с исходным параметром и пренебрежение которыми не ухудшает модель, или ухудшает, но не намного.

Традиционно для оценки степени связи параметров используют разные критерии, наиболее известным представителем которых есть коэффициент корреляции. Однако коэффициент корреляции в основном применяется лишь для оценки взаимосвязи действительных параметров, в то время как в многих задачах распознавания необходимо получать оценку взаимосвязи действительного (признак) и дискретного (номер класса) параметров. С другой стороны, при построении распознающих моделей желательно заранее знать, сколько потребуется разделяющих плоскостей для осуществления

классификации, что нельзя оценить, исходя из коэффициента корреляции.

Объединяя вышеизложенные понимания, в качестве меры связи признака и исходного параметра (меры влияния признака на исходный параметр) будем использовать количество интервалов, на которые разбивается диапазон значений признака, таких, что экземпляры, с значением признака, что пожгли в один интервал, относятся до одного и того же класса, а экземпляры сопредельных интервалов относятся к разным классам. Очевидно, что такая мера разрешит не только оценить значимость признака (чем меньше количество интервалов, тем более значимость и наоборот), но и оценить необходимое количество розділяючих прямых для классификации за данным признаком. Одновременно с оценкой значимости признаков есть возможным для каждого интервала найти предельные значения признака для экземпляров обучающей выборки, которые можно будет использовать при классификации.

Обобщая вышеизложенное, сформулируем **алгоритм оценки значимости признаков и расчета параметров рішаючого правила.**

Шаг 1. Инициализация. Задать учебную выборку экземпляров, представленную в виде массива данных *p*, в котором признака лініаризовані по строкам, а экземпляры – по столбцам, а также соответствующий массив *t*, что содержит номера классов, сопоставленные экземплярам обучающей выборки (0 или 1). Создать массив *nx*, равный за размером количества признаков *N*, элементы которого будут содержать число интервалов для каждого признака. Установить $nx(u) = 0$, $u = 1, ..., N$, где *u* – номер текущего признака. Занести количество экземпляров обучающей выборки в сменную *S*. Установить номер текущего признака $u = 1$.

Шаг 2. Если $u \le N$, тогда перейти на шаг 3, в противоположном случае – перейти на шаг 12.

Шаг 3. Занести в буфер признака *x* вектор значений *i*-ї признака из обучающей выборки: $x(j) = p(i, j), j = 1, ..., S$; занести в буфер класса у копию массива *t*: $y(j) = t(j), j = 1, ..., S$.

Шаг 4. Отсортировать массивы *x* и *y* в порядке возрастания массива *x* (шаги 4.1 - 4.7 реализуют простейший алгоритм бульбашкового сортировка, которая можно заменить на практике более быстродействующим алгоритмом).

Шаг 4.1 Установить номер текущего экземпляра обучающей выборки $j = 1$.

Шаг 4.2 Если $j \le S$, тогда перейти на шаг 4.3, в противоположном случае – перейти на шаг 5.

Шаг 4.3 Установить номер текущего экземпляра: $k = j + 1$.

Шаг 4.4 Если $k \le S$, тогда перейти на шаг 4.5, в противоположном случае – перейти на шаг 4.7.

Шаг 4.5 Если $x(j) > x(k)$, тогда установить: $tmpx = x(j)$, $x(j) = x(k)$, $x(k) = tmpx$, $tmpy = y(j)$, $y(j) = y(k)$, $y(k) = tmpy$, где $tmpx$ и $tmpy$ – буферные сменные.

Шаг 4.6 Установить: $k = k + 1$. Перейти на шаг 4.4.

Шаг 4.7 Установить: $j = j + 1$. Перейти на шаг 4.2.

Шаг 5. Установить: $s = 1$, $k = 1$.

Шаг 6. Если $s \le S$, тогда установить $tempa = x(s)$, где $tempa$ – буфер для сохранности левой границы k-го интервала i-ї признака, и перейти на шаг 7, в противоположном случае – перейти на шаг 11.

Шаг 7. Пока ($s < S$) и ($y(s) = y(s + 1)$) выполнять: $s = s + 1$.

Шаг 8. Если ($s = S$) и ($y(s) = y(s – 1)$, тогда установить: $Kx(u, k) = y(s)$, $Ax(u, k) = tempa$, $Bx(u, k) = x(s)$, $k = k + 1$, $s = s + 1$, перейти на шаг 10. Здесь $Kx(u, k)$ – номер класса сопоставленный экземплярам обучающей выборки, значения i-ї признака которых попадает в середину k-го интервала; $Ax(i, k)$ и $Bx(i, k)$ – левая и правая границы k-го интервала i-ї признака, соответственно.

Шаг 9. Если ($s < S$) и ($y(s) \ne y(s + 1)$), тогда установить: $Kx(u, k) = y(s)$, $Ax(u, k) = tempa$, $Bx(u, k) = x(s)$, $k = k + 1$, $s = s + 1$, $nx(i) = nx(u) + 1$, в противоположном случае – установить: $Kx(u, k) = y(s)$, $Ax(u, k) = x(s)$, $Bx(u, k) = x(s)$, $k = k + 1$, $s = s + 1$.

Шаг 10 Перейти на шаг 6.

Шаг 11 Установить: $u = u + 1$, перейти на шаг 2.

Шаг 12 Останов.

В результате выполнения шагов 1-12 для учебного пара $\{p, t\}$ мы получим массив nx, что содержит для каждого признака количество интервалов на который он разбивается (для оценки информативности признаков необходимо принять $NNx(u) = min(nx) / nx(u)$, $u = 1, ..., N$), а также массивы Ax, Bx и Kx, что содержат информацию о границах интервалов и номера классов, сопоставленных им для всех признаков. На основе этих массивов будем осуществлять классификацию.

Одноизмеримую классификацию за і-ю признаком будем осуществлять следующим способом. Найдем интервал, в который попадает значения признака и отнесем экземпляр за данным признаком к классу, номер которого сопоставленный интервала, в который попало значения признака. Если значения признака не попадает в ни один интервал из определенных в массивах *Ax* и *Bx*, тогда отнесем экземпляр за данным признаком к классу, сопоставленному экземплярам ближайшего интервала.

Сделав одноизмеримые классификации экземпляра за всеми признаками, отобразим результаты классификаций из интервала [0, 1] на интервал [-1, 1] и найдем их сумму, взвешенную с помощью коэффициентов $NNx(u)$. Очевидно, что результаты одноизмеримой классификации для значимых признаков в этом случае будут вносить больший вклад в сумму, чем результаты классификации за малозначимыми признаками. Рассчитав взвешенную сумму, отобразим ее на интервал [0, 1], что и будет итоговым результатом классификации экземпляра за всеми признаками.

Для оценки относительной надежности классификации для нескольких экземпляров разделим модуль взвешенной суммы результатов классификации (без отображения на интервал [0, 1]) на максимальное за модулем значения взвешенной суммы для данных экземпляров.

Обобщая вышесказанное, запишем **алгоритм классификации**.

Шаг 1. Инициализация. Задать массивы *p*, *nx*, *Ax*, *Bx* и *Kx*.

Шаг 2. Найти оценки значимости для признаков: $NNx(u) = min(nx(u)) / nx(u)$, $u = 1, ..., N$.

Шаг 3. Установить номер текущего экземпляра $j = 1$.

Шаг 4. Если $j \le S$, тогда перейти на шаг 5, в противоположном случае – перейти на шаг 15

Шаг 5. Установить значение взвешенной суммы результатов одноизмеримых классификаций *j*-го экземпляра $rj = 0$, номер текущего признака *j*-го экземпляра: $i = 1$.

Шаг 6. Если $u \le N$, тогда перейти на шаг 7, в противоположном случае – перейти на шаг 12

Шаг 7. Установить результат классификации для *j*-го экземпляра за *i*-ю признаком $r(i) = 0$;

Шаг 8. Определить интервал, к которого относится *j*-ий экземпляр за *i*-ю признаком и номер класса, сопоставленный данному интервалу.

Шаг 8.1 Если $p(u, j) < Ax(u, 1)$, тогда $r(u) = Kx(u, 1)$, перейти на шаг 9, в противоположном случае – перейти на шаг 8.2.

Шаг 8.2 Если $p(u, j) > Bx(u, nx(u) + 1)$, тогда установить $r(u) = Kx(u, nx(u) + 1)$ и перейти на шаг 9, в противоположном случае – перейти на шаг 8.3.

Шаг 8.3 Установить: $k = 1$.

Шаг 8.4 Если $k \le nx(u) + 1$, тогда перейти на шаг 8.5, в противоположном случае – перейти на шаг 9.

Шаг 8.5 Если $(p(u, j) \ge Ax(u, k))$ и $(p(u, j) \le Bx(u, k))$, тогда установить $r(u) = Kx(u, k)$ и перейти на шаг 9, в противоположном случае – перейти на шаг 8.6.

Шаг 8.6 Если $(k < nx(u) + 1)$ и $(p(u, j) > Bx(u, k))$ и $(p(u, j) < Ax(u, k + 1))$, тогда перейти на шаг 8.7, в противоположном случае – перейти на шаг 8.8.

Шаг 8.7 Если $(Ax(u, k + 1) – p(u, j)) < (p(u, j) – Ax(u, k + 1))$, тогда установить $r(u) = Kx(u, k + 1)$, в противоположном случае – установить $r(u) = Kx(u, k)$.

Шаг 8.8 Установить: $k = k + 1$, перейти на шаг 8.4

Шаг 9. Если $r(u) > 0$, тогда установить r(и) = 1, в противоположном случае установить $r(u) = –1$.

Шаг 10. Установить: $rj = rj + (r(u)NNx(u))$.

Шаг 11. $u = u + 1$, перейти на шаг 6.

Шаг 12 Установить $pr(j) = rj$, где $pr(j)$ – массив, который содержит оценки относительной уверенности (надежности) классификации.

Шаг 13. Если $rj > 0$, тогда установить $t(j) = 1$, в противоположном случае – установить $t(j) = 0$. Здесь t – массив результатов классификации.

Шаг 14. Установить: $j = j + 1$, перейти на шаг 4.

Шаг 15. Установить: $pr = |pr| / max(|pr|)$.

Рассмотренный алгоритм классификации есть неитеративным и может быть использован для построения и настраивание весов нейронної сети прямого распространения – шестишарового персептрона.

Для этого функции активации для всех нейронов сети следует задать как $\psi(x)=\begin{cases}0, x\le 0;\\ 1, x>0.\end{cases}$

Весовой коэффициент q-го входа ρ-го нейрона μ-го пласта установить соответственно формулам:

$$w_q^{(\mu,\rho)}=\begin{cases}1, \mu=3, \forall\rho, q=1;\\ Kx(i,k), \mu=3, \forall\rho, q=0;\\ -1, \mu=2, \forall\rho, q=0;\\ 1, \mu=2, \forall\rho, q>0;\\ 1, \mu=1, \rho=2i-1, q=1;\\ -1, \mu=1, \rho=2i, q=1;\\ -Ax(i,k), \mu=1, \rho=2i-1, q=0;\\ Bx(i,k), \mu=1, \rho=2i, q=0;\end{cases} \quad w_q^{(\mu,\rho)}=\begin{cases}NNx(i), \mu=6, \rho=1, q=2i-1;\\ -NNx(i), \mu=6, \rho=1, q=2i;\\ 0, \mu=6, \rho=1, q=0;\\ 0, \mu=5, \forall\rho, q=0;\\ 1, \mu=5, \rho=2i-1, q=1;\\ -1, \mu=5, \rho=2i, q=1;\\ 0, \mu=4, \forall\rho, q=0;\\ 1, \mu=4, \forall\rho, q>0;\end{cases}$$

$k = 1, .., nx(i)$; $i = 1, ..., N$.

Схема персептрона, веса которого настроенные по предложенной формуле представленная на рисунке 8.

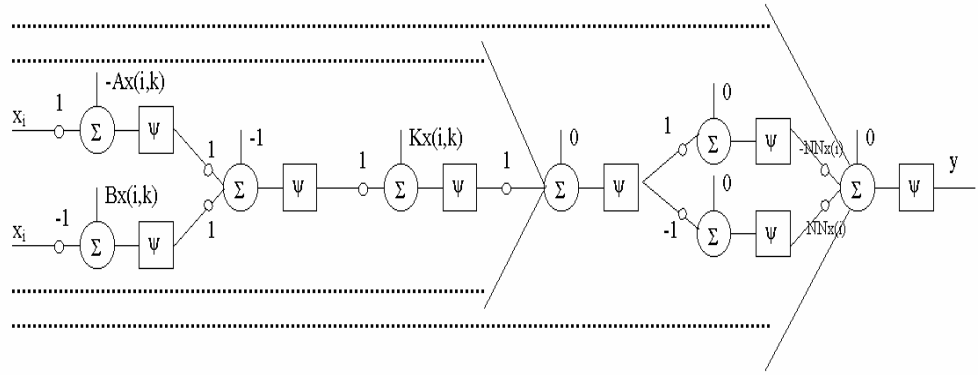


Рисунок 8 – Схема персептрона

2 ДОМАШНЯЯ ЗАДАЧА

Используя конспект лекций и рекомендованную литературу, выучить эвристические модели и методы обучения НС.

Ознакомиться с составом и порядком выполнения работы.

3 ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

3.1 Получить у преподавателя варианты задач для обучения и моделирование НС.

Входными данными есть обучающая и контрольная выборки (наборы пар значений входов и желательных выходов персептрона).

3.2 Написать и отладить программу на Паскале, Си, что моделирует эвристические модели НС и реализует алгоритмы их обучения. Предусмотреть в программе отображения на экран и запись в файл на диске текущего состояния параметров НС и результатов их работы: матрицы весов, ошибки классификации, значения на входах и выходе НС, а также время обучения и классификации на основе НС.

3.3 Для соответствующих варианту входных данных сделать обучение НС и сохранить в файле на диске результаты их работы для обучающей и контрольной выборок.

3.4. Результаты выполнения п.п. 3.3 занести в таблицу, столбцы которой должны иметь названия: модель и метод обучения НС, время обучения, время классификации, ошибка классификации.

3.5 На основе полученной таблицы дать сравнительную характеристику моделей и методов обучения НС.

4 СОДЕРЖАНИЕ ОТЧЕТА

4.1 Сформулированная цель работы.

4.2 Описание эвристических моделей НС и алгоритмов их обучения.

4.4 Описание и текст программы, которая реализует алгоритмы 4.2.

4.5 Входные данные и результаты работы программы 4.4, таблица 3.4.

4.6 Анализ полученных результатов и выводы. Лаконичные ответы на контрольные вопросы.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Дайте определение и объясните взаимосвязь понятий: НС, персептрон, классификация, эвристика.

2. Какие задачи можно решать на основе рассмотренных эвристических моделей НС, а которые нельзя? Обоснуйте ответ. Приведите примеры.
3. Объясните, что означают линейная раздельность и линейная нераздельность классов.
4. Всегда ли совпадают рассмотренные эвристические алгоритмы обучения НС?
5. Сравните рассмотренные эвристические модели и алгоритмы обучения НС и известные вам методы классификации, а также стандартные модели и методы обучения НС.

ЛИТЕРАТУРА

1. Neural Network Toolbox User Guide / Beale M., Demuth H. - Natick: Mathworks, 1997. - 700 p.
2. Аведьян Э.Д. Алгоритмы настройки многослойных нейронных сетей // Автоматика и телемеханика. – 1995. - № 4. - С. 106-118
3. Адаменко В.А., Басов Ю.Ф., Дубровин В.И., Субботин С.А. Нейросетевая обработка сигналов в задачах диагностики газотурбинных авиадвигателей // Цифровая обработка сигналов и ее применение: 3-я Международная конференция и выставка.-Г.:РНТОРЭС им. А.С. Попова, 2000.-С. 40-45.
4. Адаменко В.А., Дубровин В.И., Жеманюк П.Д., Субботин С.А. Диагностика лопаток авиадвигателей по спектрам свободных затухающих колебаний после ударного возбуждения // Автоматика-2000. Международная конференция из автоматического управления, Львов, 11-15 сентября 2000: Работы в 7-ми томах.-Т. 5.-Львов: Государственный НДІ информационной инфраструктуры, 2000.- С. 7-13.
5. Адаменко В.А., Дубровин В.И., Субботин С.А. Диагностика лопаток авиадвигателей по спектрам затухающих колебаний после ударного возбуждения на основе нейронных сетей прямого распространения // Новые материалы и технологии в металлургии и машиностроении, 2000, № 1, С. 91-96.
6. Ачасова С.М. Вычисления на нейронных сетях // Программирование. – 1991. - № 2. - С. 40-52
7. Бовель Э.И., Паршин В.В. Нейронные сеты в системах автоматического распознавания речи // Зарубежная радиоэлектроника. - 1998. - №4. - С. 50-57
8. Богуслаев А.В., Дубровин В.И., Субботин С.А., Яценко В.К. Модель коэффициента упрочнения деталей ГТД // Технологические системы, 2001, № 3.-С.42-45.
9. Резник А.М. Итеративный проекционный алгоритм обучения нейронных сетей // Кибернетика и системный анализ. – 1993. - №6. - С.131-141.
10. Резник А.М., Городничий Д.О., Сычев А.С. Регулирование локальной обратной связи в нейронных сетях с проекционным алгоритмом обучения // Кибернетика и системный анализ. – 1996. - №6. - С. 153-162.
11. Дубровин В.И., Колесник Н.В., Субботин С.А. Нейросетевое прогнозирование состояния функционально-метаболических систем в спортсменов высшей квалификации // Моделирование неравновесных систем-2000: Материалы III Всероссийского семинара / Под .ред. А.Н. Горбуна.-Красноярск: ИПЦ КГТУ, 2000.-С. 82-83.
12. Дубровин В.И., Морщавка С.В., Пиза Д. М., Субботин С.А. Применение радиально-базисных нейронных сетей для обработки данных

дистанционного зондирования растений // Цифровая обработка сигналов и ее применение: 3-я Международная конференция и выставка.-Г.:РНТОРЭС им. А.С. Попова, 2000.-С.48-53.
13. Дубровин В.И., Морщавка С.В., Пиза Д. М., Субботин С.А. Распознавание растений по результатам дистанционного зондирования на основе многослойных нейронных сетей // Математические машины и системы, 2000, № 2-3, С. 113-119.
14. Дубровин В.И., Субботин С.А. Алгоритм классификации с оценкой значимости признаков // Радиоэлектроника. Информатика. Управления, 2001, № 2, С. 145-150.
15. Дубровин В.И., Субботин С.А. Алгоритм многомерной классификации и его нейросетевая интерпретация // Радиоэлектроника. Информатика. Управления, 2000, № 2, С. 49-54.
16. Дубровин В.И., Субботин С.А. Диагностика состояния технических процессов и объектов на основе нейросетевого квантования обучающих векторов // Проектирование и технология электронных средств, 2001, № 4, С. 20-27.
17. Дубровин В.И., Субботин С.А. Индивидуальное прогнозирование надежности изделий электронной техники на основе нейронных сетей // Труды VII Всероссийской конференции “Нейрокомпьютеры и их применение” НКП-2001 с международным участием , Москва, 14-16 февраля, 2001 г.- Г.: ИПУ РАН, С. 228-231.
18. Дубровин В.И., Субботин С.А. Интегрированные многоклассификаторные нейросетевые системы диагностики // Электротехника и электроэнергетика, 2001, № 1, С. 38-43.
19. Дубровин В.И., Субботин С.А. Комбинированный метод классификации // Электротехника и электроэнергетика, 2000, № 2, С. 55-59.
20. Дубровин В.И., Субботин С.А. Моделирование акустических характеристик магнитных головок // Труды III Всероссийской научно-технической конференции "Нейроинформатика-2001".-Г.: МИФИ, 2001.-Ч.2, С.89-96.
21. Дубровин В.И., Субботин С.А. Моделирование лентопротяжного механизма на основе многослойной нейронной сеты прямого распространения // Современные проблемы радиоэлектроники / Сборник научных трудов / Под ред. А.В. Сарафанова.-Красноярск: ИПЦ КГТУ, 2000.-С. 257-258.
22. Дубровин В.И., Субботин С.А. Нейросетевая подсистема диагностического программного комплекса // Нейрокомпьютеры: разработка и применение, 2001, №2, С. 55-62.
23. Дубровин В.И., Субботин С.А. Нейросетевое моделирование лентопротяжного механизма // Труды VII Всероссийской конференции

“Нейрокомпьютеры и их применение” НКП-2001 с международным участием , Москва, 14-16 февраля, 2001 г.- Г.: ИПУ РАН, С. 153-156.
24. Дубровин В.И., Субботин С.А. Обобщенный градиентный алгоритм обучения многослойных нейронных сетей //Электротехника и электроэнергетика, 2000, № 1, С. 17-22.
25. Дубровин В.И., Субботин С.А. Онлайновые методы управления качеством: гибридная диагностика на основе нейронных сетей // Радиоэлектроника. Информатика. Управления, 2001, № 1, С. 158 –163.
26. Дубровин В.И., Субботин С.А. Эвристический алгоритм классификации и его нейросетевая интерпретация //Радиоэлектроника. Информатика. Управления, 2000, № 1, С. 72-76.
27. Дубровин В.И., Субботин С.А., Яценко В.К. Методика оценки коэффициента упрочнения деталей газотурбинных авиадвигателей // Техническая диагностика и неразрушающий контроль, 2001, № 3.-С. 42-45.
28. Дубровин В.И., Субботин С.А., Яценко В.К. Построение нейросетевой модели коэффициента упрочнения при обкатке деталей энергетических установок // Электротехника и электроэнергетика, 2001, № 2, С. 38-42.
29. Дубровин В.И., Субботин С.А.. Построение адаптивных систем классификации на основе нейронных сетей с латеральным торможением. // Радиоэлектроника. Информатика. Управления, 1999, №2 ,С. 110-114.
30. Дубровін В.И., Субботін С.О. Выбор функций активации формального нейрона и исследования их влияния на качество обучения нейронних мереж // Вестник Национального университета “Львовская политехника” “Компьютерные системы проектирования. Теория и практика”, № 398, 2000.-С.12-17.
31. Кривенко В.И., Евченко Л.Н, Субботин С.А. Нейросетевое моделирование суммарного показателя качества жизни больных хроническим обструктивным бронхитом в ассоциации с клиническими особенностями течения заболевания // Вестник новых медицинских технологий, 2001, Т. VIII, № 4, С. 7-10.
32. Круглов В.В., Борисов В.В. Искусственные нейронные сеты: теория и практика.-Г.: Горячая линия – Телезапятых, 2001.-382 с.
33. Нейрокомпьютеры и интеллектуальные роботы / Амосов Н.М., Байдык Т.Н., Гольцев А.Д. и др.; под ред. Амосова Н.М. - Киев: Научная мысль, 1991. - 272 с.
34. Уоссермен Ф. Нейрокомпьютерная техника.- Г.: Мир, 1992.
35. http://csit.narod.ru/people/Subbotin.htm, http://www.neuropower.de http://neurnews.iu.bmstu.ru – статьи с нейроінформатики
36. http://www.matlab.ru, http://www.mathworks.com, http://www.statsoft.ru – математические пакеты MATLAB и Statistica.

ПРИБАВЛЕНИЕ А

МОДЕЛИРОВАНИЯ НЕЙРОННИХ МЕРЕЖ В СРЕДЕ MATLAB

MATLAB представляет собою математический пакет, предназначенный для решения задач вычислительной математики, математической физики и построения численных моделей сложных объектов и процессов.

Пакет MATLAB состоит из интерпретатора – модельной среды, которая имеет терминальный интерфейс, ядра (набора простейших стандартных операций, функций и процедур для вычислений), а также библиотек функций (Toolbox).

Преимуществами пакета есть: богатые графические возможности, большой набор математических функций, простота встроенного языка MATLAB, возможность автоматического преобразования текстов программ языком MATLAB в тексты программ на языке Сии, а также то, что тексты библиотечных функций поставляются в исходном виде. К недостаткам пакета следует отнести отсутствие дружеского интерфейса пользователя и низкую скорость работы.

Для моделирования НС с помощью пакета MATLAB необходимо установить и использовать библиотеку Neural Network Toolbox. Рассмотрим примеры построения моделей и обучения НС языком пакета MATLAB (табл. А.1 – А.5).

Таблица А.1 – Моделирования и обучения базисный^-базисных-радиально-базисных НС

x = [1 2 3];	Задаем значение признаков экземпляров обучающей выборки: 3 экземпляры (столбце), 1 признак (строки).
y = [2.0 4.1 5.9];	Задаем значение параметра, который прогнозируется, для 3 экземпляров обучающей выборки.
net = *newrb*(*x*,*y*);	Создаем и учим радиально-базисную НС
a = *sim*(*net*,*x*)	Вычисляем по наученной сети *net* значения

	параметра, который прогнозируется, для экземпляров, которые характеризуются набором значений признаков *x*.

Таблица А.2 – Моделирования и обучения НС LVQ

x=[1-2 2 0 4-5 3];	Задаем значение признаков экземпляров обучающей выборки: 7 экземпляров (столбце), 1 признак (строки).
y=[1 2 1 2 1 2 1];	Задаем номера классов для 7 экземпляров обучающей выборки.
yc=*ind*2*vec*(*y*);	Превращаем номера классов в внутренний формат.
net=*newlvq*(*minmax*(*x*),4, [0.6 0.4], '*learnlv*1');	Создаем нейронну сеть *net* и определяем ее топологию: диапазон изменения значений признаков определяется функцией *minmax*, количество скрытых нейронов (кластеров) – 4, априорная вероятность отнесения экземпляров до одного класса – 0.6, к другому – 0.4, в качестве метода обучение сети используем метод LVQ1.
net.trainParam.epochs= 1000;	Задаем максимально допустимое количество циклов обучения (эпох).
net.trainParam.show= 100;	Задаем период отображения информации о процессе обучения на экране в циклах (эпохах) обучения.
net.trainParam.lr=0.05;	Задаем шаг обучения.
net=*train*(*net*,*x*,*yc*);	Учим нейронну сеть *net* на основе обучающей выборки, представленной набором значений признаков экземпляров *x* и набором значений соответствующих им номеров классов *y*.
a=*sim*(*net*,*x*);	Вычисляем по наученной сети *net*

	номера классов для экземпляров, которые характеризуются набором значений признаков *x*.
ac=vec2ind(*a*)	Превращаем номера кластеров в удобный для восприятия формат и выдаем на экран.

Таблица А.3 – Моделирования и обучения персептрона

x = [0.1 0.5 0.2 0.4 0.3 0.9; 0.9 0.5 0.8 0.6 0.7 0.1; 0.3 0.0 0.6 0.1 0.2 0.9];	Задаем значение признаков экземпляров обучающей выборки: 6 экземпляров (столбце), 3 признака (строки).
y = [1 0 1 0 1 0];	Задаем номера классов для 6 экземпляров обучающей выборки.
net=newff(*repmat*([0 1], 3, 1), [2,1], {'*logsig*', '*logsig*'}, '*trainlm*');	Создаем нейронну сеть net и определяем ее топологию: диапазон изменения значений признаков [0 1], количество признаков – 3, количество исходных сменных – 1, на первом пласте – 2 нейроны, на втором пласте 1 – нейрон, нейроны 1 и 2 пластов имеют сигмоїдні функции активации (*logsig*), для обучение сети используется метод Левенберга-Марквардта (*trainlm*).
net.trainparam. show=25;	Задаем период отображения информации о процессе обучения на экране в циклах (эпохах) обучения.
net.trainparam.lr= 0.01;	Задаем шаг обучения.
net.trainparam. epochs=500;	Задаем максимально допустимое количество циклов обучения (эпох).
net.trainparam. goal=0.01;	Задаем максимально допустимое значение критерия обучения (ошибки обучения).
ct=cputime;	Определяем и запоминаем текущее значение счетчика времени в

	сменной *ct*.
net=train(*net*, *x*, *y*);	Учим нейронну сеть *net* на основе обучающей выборки, представленной набором значений признаков экземпляров *x* и набором значений соответствующих им номеров классов *y*.
ct=cputime-ct	Определяем текущее значение счетчика времени, отнимаем от него значения сменной *ct* – определяем время обучения НС, что заносим в сменную *ct* и выдаем на экран (признак печати на экран – отсутствие символа “;” в конце оператора).
a=round(*sim*(*net*, *x*));	Вычисляем по наученной сети *net* номера классов для экземпляров, которые характеризуются набором значений признаков *x*.

Таблица А.4 – Моделирования и обучения НС SOM

x = rand(1,400);	Задаем набор значений для одной признака 400 экземпляров с помощью генератора случайных чисел
net=newsom(*minmax*(*x*), [2 5]);	Создаем нейронну сеть *net* и определяем ее топологию: диапазон изменения значений признаков определяется функцией *minmax,* в первом пласте массив нейронов – 2x5.
net=train(*net*, *x*);	Учим нейронну сеть net (формируем кластеры) на основе обучающей выборки, представленной набором значений признаков экземпляров *x*.
y=sim(*net*,*x*);	Вычисляем по наученной сети *net* номера кластеров, сопоставленных нейронам исходного пласта НС для экземпляров, которые характеризуются набором значений признаков *x*.
yc=vec2ind(*y*)	Превращаем номера кластеров в удобный для

	восприятия формат и выдаем на экран.

Таблица А.5 – Моделирования и обучения НС Хопфілда

x = [–1 1; –1 –1; 1 1];	Задаем значение признаков экземпляров обучающей выборки: 2 экземпляры (столбце), 3 признака (строки).
net = *newhop*(*x*);	Создаем и учим НС Хопфілда.

СОДЕРЖАНИЕ

Общие положения..... 3

1. **Лабораторная работа № 1.** Однослойный персептрон. Алгоритм Уідроу-Хоффа..... 4

2. **Лабораторная работа № 2.** Повнозв'язні нейронні сети Хопфілда. Псевдоінверсне учебное правило. Эффект рознасичення..... 11

3. **Лабораторная работа № 3.** Нейронні сети Кохонена, которые самоорганизуются: SOM и LVQ..... 19

4. **Лабораторная работа № 4.** Многослойный персептрон. Обобщенный градієнтний алгоритм обучения..... 27

5. **Лабораторная работа № 5.** Базисный^-базисные-радиально-базисные нейронні сети..... 35

6. **Лабораторная работа № 6.** Эвристические модели и методы обучения нейронних мереж..... 38

Литература..... 53

Прибавление А. Моделирования нейронних мереж в среде MATLAB..... 56