**Содержание**

Введение

## 1 Модели обучения

## 2 Введение в нейронные сети

## 3 Краткие сведения о нейроне

## 4 Искусственный нейрон

## 5 Искусственные нейронные сети. Персептрон

## 6 Проблема XOR

## 7 Решение проблемы XOR

## 8 Нейронные сети обратного распространения

## 9 Повышение эффективности обучения

## 10 Подготовка входных и выходных данных

## 11 Методы обучения

## 12 Нейронные сети Хопфилда и Хэмминга

Выводы

Литература

**Введение**

Тема контрольной работы «Самообучающиеся системы» по дисциплине «Проектированиеинтеллектуальных систем».

В современном мире прогресс производительности программиста достигается в тех случаях, когда часть интеллектуальной нагрузки берут на себя компьютеры. Одним из способов достигнуть максимального прогресса в этой области, является "искусственный интеллект", когда компьютер берет на себя не только однотипные, многократно повторяющиеся операции, но и сам сможет обучаться.

Целью изучения дисциплины является подготовка специалистов в области автоматизации сложноформализуемых задач. Задачей изучения дисциплины является приобретение знаний о фундаментальных алгоритмах, применяемых при построении систем искусственного интеллекта, а также методов разработки программных приложений, реализующих эти системы.

Принципиальное отличие интеллектуальных систем от любых других систем автоматизации заключается в наличии базы знаний о предметной среде, в которой решается задача. Неинтеллектуальная система при отсутствии каких-либо входных данных прекращает решение задачи, интеллектуальная же система недостающие данные извлекает из базы знаний и решение выполняет.

Основное внимание в контрольной работе уделяется исследованиям в области искусственного интеллекта - самообучающимся системам.

**1 Модели обучения**

В психологии под обучением понимают способность к приобретению ранее неизвестных умений и навыков. В интелектуальных системах (ИС) неформальное понимание обучения трактуется аналогично. Предполагается, что в процессе деятельности ИС анализирует имеющуюся информацию и на основе анализа извлекает из нее полезные закономерности. Обучение как математическая задача может быть отнесена к классу оптимизационных проблем поиска описаний. Индивидуальная оптимизационная задача L есть пятерка

*<XL, YL, L, FL, JL>,*

где *XL, YL* - множества входных и выходных записей;

*L  XL YL* - отношение (или функция *: XL YL*);

*FL* - множество отношений (*fL  XL YL* для всех *fL  FL*), называемых описаниями;

*JL* - оператор качества для *FL*, показывающий для каждого *fL  FL* степень его близости к *L*.

Задача состоит в отыскании оптимального по *JL* описания *f\*L* на *FL*. Для задач, относимых к обучению, характерна неполнота спецификации. Например, оператор качества может быть плохо формализуемым, либо информация об отношении.

Система (человек или машина) может получать новые знания многими способами. Можно, например, вывести нужную информацию как логическое следствие имеющихся знаний, получить ее модификацией имеющихся знаний, рассчитывая на «аналогичность» ситуаций, попытаться вывести общий закон из имеющихся примеров. Следующие задачи традиционно относятся к задачам обучения по примерам:

* прогнозирование: дана последовательность чисел: 2, 8, 14, … Чему равен следующий член последовательности?
* Идентификация (синтез) функций: имеется «черный ящик», о внутреннем устройстве которого можно судить по его поведению, подавая на вход сигналы и получая в ответ выходные. Требуется по этой информации сформулировать описание работы анализируемого устройства. К задачам идентификации относятся расшифровка структуры конечных автоматов, индуктивный синтез программ на языке Lisp;
* Расшифровка языков – поиск правил синтеза текстов некоторого языка на основе анализа конкретных текстов на этом языке ( расшифровка кодов, систем письменности и т.д.). Задачей такого же типа является обучение распознаванию образов;
* Индуктивный вывод – в широком смысле в это направление вписываются все рассмотренные выше задачи, а в узком смысле совпадает с проблемой расшифровки языков;
* Синтез с дополнительной информацией. В качестве дополнительной информации может использоваться структура примеров, их родовидовая принадлежность и т.д. К дополнительной информации относятся также контрпримеры. Например, возможность синтеза программ упрощается, если с каждой парой вход – выход задается «траектория» (последовательность состояний программы без учета их тождественности) ее вычисления.

Все рассмотренные задачи в зависимости от предмета исследования относятся к одной из двух категорий: синтезу языков или синтезу функций. Для задач обучения по примерам характерны следующие два свойства:

* все они являются задачами нахождения описаний;
* задаваемая в виде примеров информация является недостаточной для однозначного формирования требуемого описания.

Возможны ситуации, когда новые примеры ничего не изменяют в уже полученном решении. Не существует алгоритма, способного определять момент стабилизации решения и, соответственно, невозможно алгоритмически определить количество примеров для достижения стабильного обучения. Предположение о предельной стабилизации гипотез является основой гипотетико-дедуктивного подхода, согласно которому решение задачи обучения по примерам включает 4 этапа:

* наблюдение: сбор и накопление исходных данных (примеров);
* обобщение: нахождение «разумной» гипотезы н об искомом описании;
* дедукцию: выдвижение различных следствий из н или прогнозов на основе ее;
* подтверждение: проверка гипотезы Н; если гипотеза Н подтверждается, то Н остается в качестве текущей гипотезы и весь процесс повторяется сначала, а в противном случае гипотеза Н заменяется новой.

Для большинства задач обучения по примерам не существует универсальных способов их решения. В теории первый результат такого рода был получен в 1967 году Гольдом, из которого следовал, что по множеству позитивных примеров нельзя синтезировать множество регулярных языков.

##

## **2 Введение в нейронные сети**

Постоянно возрастает необходимость в системах, которые способны не только выполнять однажды запрограммированную последовательность действий над заранее определенными данными, но и способны сами анализировать вновь поступающую информацию, находить в ней закономерности, производить прогнозирование и т.д. В этой области приложений самым лучшим образом зарекомендовали себя так называемые нейронные сети – самообучающиеся системы, имитирующие деятельность человеческого мозга.

Область науки, занимающаяся построением и исследованием нейронных сетей, находится на стыке нейробиологии, математики, электроники и программирования и называется нейрокибернетикой или нейроматематикой (neurocomputing). Способность нейронной сети к обучению была впервые исследована Дж. Маккаллоком и У. Питсом. В 1943 г. вышла их работа «Логическое исчисление идей, относящихся к нервной деятельности», в которой была построена модель нейрона и сформулированы принципы построения искусственных нейронных сетей. В 1962 г. Ф. Розенблат (Корнельский университет) предложил модель нейронной сети, названную персептроном. В 70 – х годах японским ученым К. Фукушима была предложена другая модель – когнитрон, способная хорошо распознавать сложные образы (иероглифы и т.п.) независимо от поворота и изменения масштаба изображения. В 1982 г. американский биофизик Дж. Хопфилд предложил модель нейронной сети, названную его именем.

Нейронные сети наиболее эффективны в системах распознавания образов, ассоциативной памяти, прогнозирования и адаптивного управления и неэффективны в областях, требующих точных вычислений. На рынке появляются реализации нейроподобных систем в виде пакетов программ, нейроплат и нейрочипов. Исследованиями в области нейрокомпьютеров занимаются в Японии: фирмы NBC, Nihon Denki, Mitec, Fujitsu, Matsusita, Mitsubishi, Sony, Toshiba, Hitachi; в США: AT&T, IBM, Texas Instruments, Xerox.

##

## **3 Краткие сведения о нейроне**

Нейрон – это нервная клетка, состоящая из тела и отростков, соединяющих ее с внешним миром.

**Рисунок** **1 - Биологический нейрон**

Биологический нейрон содержит следующие структурные единицы:

Тело клетки (т) — сома: содержит ядро (я), митохондрии (обеспечивают клетку энергией), другие органеллы, поддерживающие жизнедеятельность клетки.

Дендриты (д) – входные волокна, собирают информацию от других нейронов. Активность в дендритах меняется плавно. Длина их обычно не больше 1 мм.

Мембрана – поддерживает постоянный состав цитоплазмы внутри клетки, обеспечивает проведение нервных импульсов.

Цитоплазма — внутренняя среда клетки. Отличается концентрацией ионов K+, Na+, Ca++ и других веществ по сравнению с внеклеточной средой.

Аксон (а), один или ни одного у каждой клетки, – длинное, иногда больше метра, выходное нервное волокно клетки. Импульс генерируется в аксонном холмике (а.х.). Аксон обеспечивает проведение импульса и передачу воздействия на другие нейроны или мышечные волокна (мв). Ближе к концу аксон часто ветвится.

Синапс (с) – место контакта нервных волокон — передает возбуждение от клетки к клетке. Передача через синапс почти всегда однонаправленная. Различают пресинаптические и постсинаптические клетки — по направлению передачи импульса.

Шванновские клетки (шв.кл). Специфические клетки, почти целиком состоящие из миелина, органического изолирующего вещества. Плотно "обматывают" нервное волокно 250 слоями миелина. Неизолированные места нервного волокна между шванновскими клетками называются перехватами Ранвье (пР). За счет миелиновой изоляции скорость распространения нервных импульсов возрастает в 5 - 10 раз и уменьшаются затраты энергии на проведение импульсов. Миелинизированные волокна встречаются только у высших животных.

В центральной нервной системе человека насчитывается от 100 до 1000 типов нервных клеток, в зависимости выбранной степени детализации. Они отличаются картиной дендритов, наличием и длиной аксона и распределением синапсов около клетки.

Клетки сильно связаны между собой. У нейрона может быть больше 1000 синапсов. Близкие по функциям клетки образуют скопления, шаровидные или параллельные слоистые. В мозгу выделены сотни скоплений. Кора головного мозга – тоже скопление. Толщина коры — 2 мм, площадь — около квадратного фута.

Нервный импульс (спайк) – процесс распространения возбуждения по аксону от тела клетки (аксонного холмика) до окончания аксона. Это основная единица информации, передаваемая по волокну, поэтому модель генерации и распространения нервных импульсов (НИ) — одна из важнейших в теории НС.

Импульсы по волокну передаются в виде скачков потенциала внутриклеточной среды по отношению к внешней среде, окружающей клетку. Скорость передачи – от 1 до 100 м/с. Для миелинизированных волокон скорость передачи примерно в 5 – 10 раз выше, чем для немиелинизированных.

При распространении форма спайка не меняется. Импульсы не затухают. Форма спайка фиксирована, определяется свойствами волокна и не зависит от того, каким способом создан импульс.

При воздействии вспышек света постоянной длительности и различной интенсивности вырабатывались импульсы в соответствующем зрительном волокне. Было определено, что от интенсивности света зависит не амплитуда импульсов и их форма, а плотность и общее количество.

##

## **4 Искусственный нейрон**

Несмотря на большое разнообразие вариантов нейронных сетей, все они имеют общие черты. Так, все они, так же, как и мозг человека, состоят из большого числа связанных между собой однотипных элементов – нейронов, которые имитируют нейроны головного мозга. На рисунке показана схема нейрона.

**Рисунок** **2 - Схема нейрона**

Из рисунка видно, что искусственный нейрон, так же, как и живой, состоит из синапсов, связывающих входы нейрона с ядром; ядра нейрона, которое осуществляет обработку входных сигналов и аксона, который связывает нейрон с нейронами следующего слоя. Каждый синапс имеет вес, который определяет, насколько соответствующий вход нейрона влияет на его состояние. Состояние нейрона определяется по формуле:

*S = ∑nxiwi*,

где

*n* – число входов нейрона;

*xi* – значение ***i***-го входа нейрона;

*wi* – вес *i*-го синапса.

Затем определяется значение аксона нейрона по формуле

*Y = f(S)*,

где *f* – некоторая функция, которая называется активационной.

Самая простая интерпретация выработки сигнала в аксон – сравнение суммарного возбуждения с некоторым пороговым значением. Исходя из этой интерпретации, искусственный нейрон будет иметь схему, показанную на рисунке 3.

**Рисунок** **3 - Элементы схемы нейрона**

Наиболее часто в качестве активационной функции используется так называемый сигмоид, который имеет следующий вид:

.

Основное достоинство этой функции в том, что она дифференцируема на всей оси абсцисс и имеет очень простую производную:

*f '(s) = α f(s)(1 – f(s)).*

При уменьшении параметра ***a*** сигмоид становится более пологим, вырождаясь в горизонтальную линию на уровне 0,5 при*a*=0. При увеличении ***a*** сигмоид все больше приближается к функции единичного скачка.

##

## **5 Искусственные нейронные сети. Персептрон**

Нейроны в сети могут соединяться регулярным или случайным образом. Работа сети разделяется на обучение и тестирование.

Под обучением понимается процесс адаптации сети к предъявляемым эталонным образам путем модификации (в соответствии с тем или иным алгоритмом) весовых коэффициентов связей между нейронами.

Тестирование – это работа сети при подаче входного возбуждения (образа для распознавания) и неизменных весовых коэффициентах. Если сеть работает правильно, то подаваемое на вход возбуждение должно вызывать появление на выходе той реакции, на которую сеть была обучена, и при этом сеть должна стабилизироваться.

Свойства нейронной сети зависят от следующих факторов:

* модели нейронов;
* структуры сети;
* алгоритма обучения;
* порядка предъявления сети эталонных образов;
* характера самих эталонов.

Искусственные нейронные сети могут иметь слоистую структуру.

Слой, воспринимающий внешнее раздражение, называют рецепторным. Как правило, он служит для распределения сигналов на другие слои и никакой вычислительной работы не выполняет.

Слой, с которого снимается реакция сети, называется эффекторным.

Слои, находящиеся между этими двумя, называются скрытыми.

Связи между слоями в направлении от рецепторного слоя к эффекторному, называются прямыми, а с противоположным направлением – обратными.

Самым распространенным видом сети стал многослойный персептрон.В многослойном персептроне (МСП) нет обратных связей. Такие модели называются сетями прямого распространения. Они не обладают внутренним состоянием и не позволяют без дополнительных приемов моделировать развитие динамических систем.

Функция активации «жесткая ступенька» используется в классическом формальном нейроне:

┌ 0, x < θ;

f(x) = ┤

└ 1, x ≥ θ.

Функция вычисляется двумя - тремя машинными инструкциями, поэтому нейроны с такой нелинейностью требуют малых вычислительных затрат. Эта функция чрезмерно упрощена и не позволяет моделировать схемы с непрерывными сигналами.

Сеть состоит из произвольного количества слоев нейронов. Нейроны каждого слоя соединяются с нейронами предыдущего и последующего слоев по принципу "каждый с каждым". Количество нейронов в слоях может быть произвольным. Обычно во всех скрытых слоях одинаковое количество нейронов. Существует путаница с подсчетом количества слоев в сети. Входной слой не выполняет никаких вычислений, а лишь распределяет входные сигналы, поэтому иногда его считают, иногда — нет.

Каждый слой рассчитывает нелинейное преобразование от линейной комбинации сигналов предыдущего слоя. Отсюда видно, что линейная функция активации может применяться только для тех моделей сетей, где не требуется последовательное соединение слоев нейронов друг за другом. Для многослойных сетей функция активации должна быть нелинейной, иначе можно построить эквивалентную однослойную сеть, и многослойность оказывается ненужной. Если применена линейная функция активации, то каждый слой будет давать на выходе линейную комбинацию входов. Следующий слой даст линейную комбинацию выходов предыдущего, а это эквивалентно одной линейной комбинации с другими коэффициентами, и может быть реализовано в виде одного слоя нейронов.

Многослойная сеть может формировать на выходе произвольную многомерную функцию при соответствующем выборе количества слоев, диапазона изменения сигналов и параметров нейронов. Как и ряды, многослойные сети оказываются универсальным инструментом аппроксимации функций. Видно отличие работы нейронной сети от разложения функции в ряд:

*f(x) = ∑cifi(x).*

За счет поочередного расчета линейных комбинаций и нелинейных преобразований достигается аппроксимация произвольной многомерной функции при соответствующем выборе параметров сети.

В многослойном персептроне нет обратных связей. Такие модели называются сетями прямого распространения. Они не обладают внутренним состоянием и не позволяют без дополнительных приемов моделировать развитие динамических систем.

##

## **6 Проблема XOR**

Наглядной и уже ставшей классической иллюстрацией ограниченности для однослойного персептрона является функция "исключающее ИЛИ". Эта булева функция от двух переменных принимает значение "истина", когда значения входных переменных различны, и "ложь" – в противном случае. Попробуем применить однослойную сеть, состоящую из одного нейрона , для построения этой функции.

Для сохранения симметрии будем сопоставим значению "ложь" сигнал нейросети, равный -1, а значению "истина" — равный 1. Значения входов и выходов дискретны, и есть смысл использовать жесткую ступеньку в качестве функции активации. Требуемая таблица истинности и соответствующие уравнения для весов и порогов для функции "исключающее или" имеет вид:

**Таблица 1 - Система неравенств для построения функции XOR**

|  |  |  |  |
| --- | --- | --- | --- |
| *x1* | *x2* | *y* | Условие |
| 1 | 1 | -1 |  w1 + w2 - Θ < 0 |
| 1 | -1 | 1 |  w1 - w2 - Θ > 0 |
| -1 | 1 | 1 | -w1 + w2 - Θ > 0 |
| -1 | -1 | -1 | -w1 - w2 - Θ < 0 |

Вычтем из первого уравнения второе, а из третьего — четвертое. Получим несовместную систему:

2w2 < 0;

2w2 > 0.

Хотя система условий в таблице - лишь система неравенств, пусть содержащая три переменных и четыре уравнения, она оказывается несовместной. Следовательно, функция XOR не реализуется на однослойном персептроне. Тот же результат можно получить графически. Возможные значения входов сети — на рисунке.

**Рисунок** **4 - Графическое представление функции XOR**

Один нейрон с двумя входами может сформировать решающую поверхность в виде прямой. Требуется провести ее так, чтобы отделить значения y = 1 от значений y = -1. Очевидно, что это невозможно.

Задачи, для которых не существует решающей поверхности в виде гиперплоскости, называются линейно неразделимыми.

Хотя данный пример нагляден, он не является серьезным ограничением нейросетей. Функция XOR легко формируется уже двухслойной сетью, причем многими способами.

##

## **7 Решение проблемы XOR**

Для построения булевых функций c помощью нейросетей есть завершенные математические методы [Мкртчян71]. Рассмотрим простейший пример и построим нейронную сеть без предварительного обучения.

Запишем функцию XOR в виде: *y = x1 XOR x2 = (x1 AND NOT x2) OR (NOT x1 AND x2)*. Алгоритмические обозначения операторов использованы для наглядности. Легко построить двухслойную сеть для реализации такой функции. Инверсию можно осуществить отрицательным весовым коэффициентом, а операции AND и OR — каждую отдельным нейроном с различными значениями порогов. Схема сети представлена на рисунке 6.

**Рисунок** **5 - Двухслойная сеть, реализующая функцию XOR.**

**Таблица 2 -Таблица истинности для нейронной сети**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ***x1*** | ***x2*** | ***d1*** | ***d2*** | ***y1*** | ***y2*** | ***d*** | ***y*** |
| -1 | -1 | 0 | 0 | -1 | -1 | -2w | -1 |
| -1 | 1 | -w | w | -1 | 1 | w | 1 |
| 1 | -1 | w | -w | 1 | -1 | w | 1 |
| 1 | 1 | 0 | 0 | 1 | 1 | -2w | -1 |

## **8 Нейронные сети обратного распространения**

Нейронные сети обратного распространения – это инструмент поиска закономерностей, прогнозирования, качественного анализа. Такое название – сети обратного распространения (back propagation), - они получили из-за используемого алгоритма обучения, в котором ошибка распространяется от выходного слоя к входному, т. е. в направлении, противоположном направлению распространения сигнала при нормальном функционировании сети.

Нейронная сеть обратного распространения состоит из нескольких слоев нейронов, причем каждый нейрон слоя *i* связан с каждым нейроном слоя *i+1*, т. е. речь идет о полносвязной НС.

В принципе, для создания нейронных сетей количество слоев определяется исходя из поставленной задачи. Чем больше слоев в сети, тем более сложную функцию можно представить с помощью ИНС. При решении практических задач обычно выбираются один или два скрытых слоя, так как при большем количестве слоев трудно обучать ИНС.

В общем случае задача обучения НС сводится к нахождению некой функциональной зависимости *Y=F(X)* где *X* – входной, а *Y* – выходной векторы. В общем случае такая задача, при ограниченном наборе входных данных, имеет бесконечное множество решений. Для ограничения пространства поиска при обучении ставится задача минимизации целевой функции ошибки НС, которая находится по методу наименьших квадратов:

где

*yj* – значение *j*-го выхода нейросети;

*dj* – целевое значение *j*-го выхода;

*k*– число нейронов в выходном слое.

Обучение нейросети производится методом градиентного спуска, т. е. на каждой итерации изменение веса производится по формуле

где *ή* – параметр, определяющий скорость обучения.

Для выходного слоя можно записать компоненты градиента по весам функции ошибки:

где

yj– значение выхода *j*-го нейрона;

Sj– взвешенная сумма входных сигналов *j*-го нейрона;

wij– *i*-тый вход *j* – того нейрона.

При этом производная взвешенной суммы по весам будет равна:

где *xi* – значение ***i***-го входа нейрона.

Для выходного слоя производная функции ошибки будет равна:

,

где *p* – номер обучающего образца.

Если в качестве активационной функции используется сигмоид, то будет определяться формулой:

,

где , определяет текущее значение выходного сигнала.

Расчеты по вышеприведенным формулам позволяют выполнить коррекцию весов для нейронов последнего слоя. Для получения формул для других слоев можно воспользоваться формулой производной функции ошибки по входному сигналу (n – номер слоя):

.

Последний сомножитель тождественно совпадает с весовым коэффициентом *i* – того входа *j* – того нейрона:

.

Но производная по входному значению для последнего слоя тождественно совпадает с производной по соответствующему выходу для предыдущего слоя:

.

Первый множитель формулы (7) определяется формулой:

,

где k – число нейронов в слое n.

Вводится вспомогательная переменная

Тогда можно задать рекурсивную формулу для определения (n-1) - го слоя, если известны следующего n - го слоя.

 .

Нахождение же для последнего слоя нейронной сети не представляет трудности, так как известен целевой вектор, т. е. вектор тех значений, которые должна выдавать нейронная сеть при данном наборе входных значений.

.

Можно записать формулу для приращений весовых коэффициентов в раскрытом виде

.

Полный алгоритм обучения нейросети складывается из следующих шагов:

1) подать на вход НС один из обучающих образов и определить значения выходов нейронов;

2) рассчитать функцию ошибки *Е.* Если значение ошибки не превышает допустимое, то выбрать следующий обучающий образец и перейти на шаг 1. Целесообразно организовать случайный выбор образцов из обучающей последовательности. Иначе – к следующему пункту;

3) рассчитать для выходного слоя нейронной сети по формуле (12) и рассчитать изменения весов выходного слоя *n* по формуле (13);

4) рассчитать по формулам (11) и (13) соответственно и для остальных слоев, n = N-1..1;

5) скорректировать все веса нейронной сети:

6) перейти на шаг 1.

##

## **9 Повышение эффективности обучения**

Простейший метод градиентного спуска, рассмотренный выше, очень неэффективен в случае, когда производные по различным весам сильно отличаются. Это соответствует ситуации, когда значение функции S для некоторых нейронов близка по модулю к 1 или когда модуль некоторых весов много больше 1. В этом случае для плавного уменьшения ошибки надо выбирать очень маленькую скорость обучения, но при этом обучение может занять непозволительно много времени.

Простейшим методом усовершенствования градиентного спуска является введение момента µ, когда влияние градиента на изменение весов изменяется со временем. Тогда формула (13) примет вид

Дополнительным преимуществом от введения момента является способность алгоритма преодолевать мелкие локальные минимумы.

##

## **10 Подготовка входных и выходных данных**

Данные, подаваемые на вход сети и снимаемые с выхода, должны быть правильно подготовлены.

Один из распространенных способов — масштабирование:

*x = ( x’ – m)\*c*,

где

*x’* - исходный вектор,

*x* - масштабированный;

*m* – вектор усредненных значений совокупности входных данных;

*с* - масштабный коэффициент.

Масштабирование желательно, чтобы привести данные в допустимый диапазон. Если этого не сделать, то возможно несколько проблем:

1) нейроны входного слоя или окажутся в постоянном насыщении (*|m|* велик, дисперсия входных данных мала) или будут все время заторможены (*|m|* мал, дисперсия мала);

2) весовые коэффициенты примут очень большие или очень малые значения при обучении (в зависимости от дисперсии), и, как следствие, растянется процесс обучения и снизится точность.

Рассмотрим набор входных данных для сети с одним входом:

*{x’}* = {10 10.5 10.2 10.3 10.1 10.4}.

Если функция активации — гиперболический тангенс с множеством значений, то при весовых коэффициентах около единицы нейроны входного слоя окажутся в насыщении для всех ***x′.*** Применим масштабирование с *m*=10,2 и *c*=4. Это даст значения в допустимом диапазоне (-1; 1).

Выходы сети масштабируются так же. Т.к. пользователь сам выбирает смысл выходного вектора при создании сети, то данные подготавливаются так, чтобы диапазон изменения выходных сигналов лежал на рабочем участке функции активации.

Основное отличие НС в том, что в них все входные и выходные параметры представлены в виде чисел с плавающей точкой обычно в диапазоне [0..1]. В то же время данные предметной области часто имеют другое кодирование. Это могут быть числа в произвольном диапазоне, даты, символьные строки. Таким образом, данные о проблеме могут быть как количественными, так и качественными.

Качественные данные можно разделить на две группы: упорядоченные (ординальные) и неупорядоченные. Способы кодирования этих данных рассматриваются на примере задачи о прогнозировании успешности лечения какого-либо заболевания. Например, примером упорядоченных данных могут быть данные о дополнительных факторах риска при каком - либо заболевании.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  **нет** | **ожирение** | **алкоголь** | **курение** | **гипертония** |

Опасность каждого фактора возрастает при движении слева направо.

В первом случае у больного может быть несколько факторов риска одновременно. В таком случае необходимо использовать такое кодирование, при котором отсутствует ситуация, когда разным комбинациям факторов соответствует одно и то же значение. Наиболее распространен способ кодирования, когда каждому фактору ставится в соответствие разряд двоичного числа. 1 в этом разряде говорит о наличии фактора, а 0 о его отсутствии. Параметру *нет* можно поставить в соответствии число 0. Таким образом, для представления всех факторов достаточно 4-х разрядного двоичного числа. Таким образом, число 01012 означает наличие у больного гипертонии и употребления алкоголя, а числу 00002 соответствует отсутствие у больного факторов риска. Таким образом факторы риска будут представлены числами в диапазоне [0..15].

Другим возможным примером может быть возраст больного:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  до 25 лет | 25-39 лет | 40-49 лет | 50-59 лет | 60 и старше |

Во этом случае также можно кодировать все значения двоичными весами, но это будет нецелесообразно, т.к. набор возможных значений будет слишком неравномерным. Более правильным будет установка в соответствие каждому значению своего веса, отличающегося на 1 от веса соседнего значения. Так, число 3 будет соответствовать возрасту 50--59лет. Таким образом, возраст будет закодирован числами в диапазоне [0..4].

В принципе аналогично можно поступать и для неупорядоченных данных, поставив в соответствие каждому значению какое-либо число. Однако это вводит нежелательную упорядоченность, которая может исказить данные, и сильно затруднить процесс обучения. В качестве одного из способов решения этой проблемы можно предложить поставить в соответствие каждому значению одного из входов НС. В этом случае при наличии этого значения соответствующий ему вход устанавливается в 1 или в 0 при противном случае. К сожалению, данный способ не является панацеей, ибо при большом количестве вариантов входного значения число входов НС разрастается до огромного количества. Это резко увеличит затраты времени на обучение. В качестве варианта обхода этой проблемы можно использовать несколько другое решение. В соответствие каждому значению входного параметра ставится бинарный вектор, каждый разряд которого соответствует отдельному входу НС. Например, если число возможных значений параметра 128, то можно использовать 7 разрядов.

## **11 Методы обучения**

Алгоритмы обучения бывают с учителем и без. Алгоритм называется алгоритмом с учителем, если:

1) при обучении известны и входные, и выходные вектора сети;

2) имеются пары вход + выход - известные условия задачи и решение.

В процессе обучения сеть меняет свои параметры и учится давать нужное отображение *X → Y*. Сеть учится давать результаты, которые нам уже известны. За счет способности к обобщению сетью могут быть получены новые результаты, если подать на вход вектор, который не встречался при обучении.

Алгоритм относится к обучению без учителя, если известны только входные вектора, и на их основе сеть учится давать наилучшие значения выходов. Что понимается под “наилучшими” — определяется алгоритмом обучения.

Персептрон обучается с учителем. Это означает, что должно быть задано множество пар векторов *{xs, ds}*, где *{ xs }* - формализованное условие задачи, *{ ds }* - известное решение для этого условия. Совокупность пар *{xs, ds}* составляет обучающее множество. *S* - количество элементов в обучающем множестве - должно быть достаточным для обучения сети, чтобы под управлением алгоритма сформировать набор параметров сети, дающий нужное отображение *X → Y*. Количество пар в обучающем множестве не регламентируется. Если элементов слишком много или мало, сеть не обучится и не решит поставленную задачу.

##

## **12 Нейронные сети Хопфилда и Хэмминга**

Среди различных конфигураций искусственных нейронных сетей (НС) встречаются такие, при классификации которых по принципу обучения, строго говоря, не подходят ни обучение с учителем, ни обучение без учителя. В таких сетях весовые коэффициенты синапсов рассчитываются только однажды перед началом функционирования сети на основе информации об обрабатываемых данных, и все обучение сети сводится именно к этому расчету. С одной стороны, предъявление априорной информации можно расценивать, как помощь учителя, но с другой – сеть фактически просто запоминает образцы до того, как на ее вход поступают реальные данные, и не может изменять свое поведение, поэтому говорить о звене обратной связи с "миром" (учителем) не приходится. Из сетей с подобной логикой работы наиболее известны сеть Хопфилда и сеть Хэмминга, которые обычно используются для организации ассоциативной памяти. Далее речь пойдет именно о них.

Структурная схема сети Хопфилда приведена на рисунке. Она состоит из единственного слоя нейронов, число которых является одновременно числом входов и выходов сети. Каждый нейрон связан синапсами со всеми остальными нейронами, а также имеет один входной синапс, через который осуществляется ввод сигнала. Выходные сигналы, как обычно, образуются на аксонах Задача, решаемая данной сетью в качестве ассоциативной памяти, как правило, формулируется следующим образом. Известен некоторый набор двоичных сигналов (изображений, звуковых оцифровок, прочих данных, описывающих некие объекты или характеристики процессов), которые считаются образцовыми. Сеть должна уметь из произвольного неидеального сигнала, поданного на ее вход, выделить ("вспомнить" по частичной информации) соответствующий образец (если такой есть) или "дать заключение" о том, что входные данные не соответствуют ни одному из образцов. В общем случае, любой сигнал может быть описан вектором *X = { xi: i=0...n-1}, n* – число нейронов в сети и размерность входных и выходных векторов. Каждый элемент *xi* равен либо +1, либо -1. Обозначим вектор, описывающий *k*-ый образец, через *Xk*, а его компоненты, соответственно, – *xik, k=0...m-1, m* – число образцов. Когда сеть распознает (или "вспомнит") какой-либо образец на основе предъявленных ей данных, ее выходы будут содержать именно его, то есть *Y = Xk*, где *Y* – вектор выходных значений сети: *Y = { yi: i=0,...n-1}.* В противном случае, выходной вектор не совпадет ни с одним образцовым.

**Рисунок** **6 - Структурная схема сети Хопфилда**

Если, например, сигналы представляют собой некие изображения, то, отобразив в графическом виде данные с выхода сети, можно будет увидеть картинку, полностью совпадающую с одной из образцовых (в случае успеха) или же "вольную импровизацию" сети (в случае неудачи).

На стадии инициализации сети весовые коэффициенты синапсов устанавливаются следующим образом:

Здесь *i* и *j* – индексы, соответственно, предсинаптического и постсинаптического нейронов; *xik, xjk* – *i*-ый и *j*-ый элементы вектора *k*-ого образца.

Алгоритм функционирования сети следующий (***p*** – номер итерации):

1) На входы сети подается неизвестный сигнал. Фактически его ввод осуществляется непосредственной установкой значений аксонов:

*yi(0) = xi , i = 0...n-1,*

поэтому обозначение на схеме сети входных синапсов в явном виде носит чисто условный характер. Ноль в скобке справа от ***yi*** означает нулевую итерацию в цикле работы сети.

2) Рассчитывается новое состояние нейронов

и новые значения аксонов

где ***f*** – активационная функция в виде скачка, приведенная на рисунке 7, а.

**Рисунок** **7 - Активационная функция**

3) Проверка, изменились ли выходные значения аксонов за последнюю итерацию. Если да – переход к пункту 2, иначе (если выходы застабилизировались) – конец. При этом выходной вектор представляет собой образец, наилучшим образом сочетающийся с входными данными.

Как говорилось выше, иногда сеть не может провести распознавание и выдает на выходе несуществующий образ. Это связано с проблемой ограниченности возможностей сети. Для сети Хопфилда число запоминаемых образов ***m*** не должно превышать величины, примерно равной *0.15 • n*. Кроме того, если два образа А и Б сильно похожи, они, возможно, будут вызывать у сети перекрестные ассоциации, то есть предъявление на входы сети вектора А приведет к появлению на ее выходах вектора Б и наоборот.

**Рисунок 8 - Структурная схема сети Хэмминга**

Когда нет необходимости, чтобы сеть в явном виде выдавала образец, то есть достаточно, скажем, получать номер образца, ассоциативную память успешно реализует сеть Хэмминга. Данная сеть характеризуется, по сравнению с сетью Хопфилда, меньшими затратами на память и объемом вычислений, что становится очевидным из ее структуры (см. рисунок).

Сеть состоит из двух слоев. Первый и второй слои имеют по *m* нейронов, где *m* – число образцов. Нейроны первого слоя имеют по *n* синапсов, соединенных с входами сети (образующими фиктивный нулевой слой). Нейроны второго слоя связаны между собой ингибиторными (отрицательными обратными) синаптическими связями. Единственный синапс с положительной обратной связью для каждого нейрона соединен с его же аксоном.

Идея работы сети состоит в нахождении расстояния Хэмминга от тестируемого образа до всех образцов. Расстоянием Хэмминга называется число отличающихся битов в двух бинарных векторах. Сеть должна выбрать образец с минимальным расстоянием Хэмминга до неизвестного входного сигнала, в результате чего будет активизирован только один выход сети, соответствующий этому образцу.

1) На стадии инициализации весовым коэффициентам первого слоя и порогу Tk активационной функции присваиваются следующие значения

, *i=0...n-1*

*Tk = n / 2, для k = 0, . . . , m – 1;*

*0 < ε < 1/m,*

где *n* – количество входов;

*m* – количество нейронов в слое.

После этого полученными значениями инициализируются значения аксонов второго слоя:

*yj(2) = yj(1), j = 0...m-1*

2) Вычислить новые состояния нейронов второго слоя:

и значения их аксонов:

Активационная функция ***f*** имеет вид порога (рисунок 3.2, б), причем величина ***F*** должна быть достаточно большой, чтобы любые возможные значения аргумента не приводили к насыщению.

3) Проверить, изменились ли выходы нейронов второго слоя за последнюю итерацию. Если да – перейди к шагу 2, иначе – конец.

Из оценки алгоритма видно, что роль первого слоя весьма условна: воспользовавшись один раз на шаге 1 значениями его весовых коэффициентов, сеть больше не обращается к нему, поэтому первый слой может быть вообще исключен из сети (заменен на матрицу весовых коэффициентов).

**Выводы**

В процессе изучения темы «Самообучающиеся автоматы» мы ознакомились со следующими вопросами:

## - моделями обучения;

## - введением в нейронные сети;

## - краткими сведениями о нейроне;

## - искусственным нейроном;

## - проблемой XOR;

## - решением проблем XOR;

## - нейронными сетями обратного распространения;

## - повышением эффективности обучения;

## - методами обучения;

## - нейронными сетями Хопфилда и Хэмминга.

## **Литература**

1. Уоссермен Ф., Нейрокомпьютерная техника, - М.,Мир, 1992.

2. Горбань А.Н. Обучение нейронных сетей. - М.: ПараГраф, 1990

3. Горбань А.Н., Россиев Д.А. Нейронные сети на персональном компьютере. - Новосибирск: Наука, 1996

4. Gilev S.E., Gorban A.N., Mirkes E.M. Several methods for accelerating the training process of neural networks in pattern recognition // Adv. Modelling & Analysis, A. AMSE Press. – 1992. – Vol.12, N4. – P.29-53

5. С. Короткий. Нейронные сети: алгоритм обратного распространения.

6. С. Короткий, Нейронные сети: обучение без учителя. Artificial Neural Networks: Concepts and Theory, IEEE Computer Society Press, 1992.

7. Заенцев И. В. Нейронные сети: основные модели./Учебное пособие к курсу "Нейронные сети" для студентов 5 курса магистратуры к. электроники физического ф-та Воронежского Государственного университета – e-mail: ivz@ivz.vrn.ru

8. Лорьер Ж.Л. Системы искусственного интеллекта. – М.: Мир, 1991. – 568 с.

9. Искусственный интеллект. – В 3-х кн. Кн. 2. Модели и методы: Справочник/ Под ред. Поспелова Д. А. – М.: Радио и связь, 1990. – 304 с.

10. Бек Л. Введение в системное программирование.- М.: Мир, 1988.

11. Шлеер С., Меллор С. Объектно-ориентированный анализ: моделирование мира в состояниях. – К.: Диалектика, 1993. – 240 с.

12. Буч Г. Объектно-ориентированный анализ и проектирование с примерами приложений на С++. - http://www.nexus.odessa.ua/files/books/booch.

13. Аджиев В. MS: корпоративная культура разработки ПО – http:// www.osp.ru

14. Трофимов С.А. Case-технологии. Практическая работа в Rational Rose. – М.: ЗАО «Издательство БИНОМ», 2001.

15. Новичков А. Эффективная разработка программного обеспечения с использованием технологий и инструментов компании RATIONAL. – http://www.interface.ru

16. Selic B., Rumbaugh J. Использование UML при моделировании сложных систем реального времени. - http://www.interface.ru.