Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Кемеровский государственный университет»
Новокузнецкий институт (филиал)

Факультет информатики, математики и экономики
Кафедра математики, физики и математического моделирования

Е.В. Решетникова, А.Е. Паульзен

МАТЕМАТИЧЕСКИЕ МОДЕЛИ И МЕТОДЫ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

Методические указания к выполнению лабораторных работ
для обучающихся по направлениям подготовки
01.03.02 Прикладная математика и информатика,
профиль «Математическое моделирование и информационные технологии»
02.03.03 Математическое обеспечение и администрирование информационных систем,
профиль «Программное и математическое обеспечение информационных технологий»

Новокузнецк
2020

УДК [378.147.88:004.89] (072)
ББК 74.484(2Рос-4Кем)я73+32.813.3я73
Р47

Е.В. Решетникова

Р47 Математические модели и методы искусственного интеллекта: методические указания к выполнению лабораторных работ для студентов факультета информатики, математики и экономики, обучающихся по направлениям подготовки 01.03.02 Прикладная математика и информатика, профиль «Математическое моделирование и информационные технологии»; 02.03.03 Математическое обеспечение и администрирование информационных систем, профиль «Программное и математическое обеспечение информационных технологий»: / Е.В. Решетникова, А.Е. Паульзен; Новокузнецкий ин-т (фил.) Кемеров. гос. ун-та. – Новокузнецк : НФИ КемГУ, 2020 – 56 с.

Приводятся методические указания по выполнению лабораторных работ по разделам: «Интеллектуальные задачи. Эвристическое программирование», «Разработка экспертных систем», «Разработка систем распознавания образов», «Проектирование нейронных сетей».

Методические указания предназначены для студентов всех форм обучения направлений 01.03.02 «Прикладная математика и информатика» и 02.03.03 «Математическое обеспечение и администрирование информационных систем».

Рекомендовано на заседании
кафедры математики, физики и
математического моделирования
Протокол № 3 от 22.10.2020

Заведующий каф. МФММ
_____ / Е.В.Решетникова

Утверждено методической комиссией
факультета информатики, математики и
экономики
Протокол № 4 от 12.11.2020

Председатель методической комиссии
ФИМЭ
_____/Г.Н.Бойченко

УДК [378.147.88:004.89] (072)
ББК 74.484(2Рос-4Кем)я73+32.813.3я73
Р47

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Примерно в 70-е годы прошлого столетия – начале фазы компьютерной революции был совершен концептуальный прорыв в новой области информатики и вычислительной техники, названной искусственным интеллектом. В эти годы была принята новая концепция, которая утверждала, что эффективность программы при решении задачи зависит от знаний, которыми она обладает, а не только от формализмов и методов вывода, которые она использует.

Наиболее значительными работами в области искусственного интеллекта являются разработки мощных компьютерных систем или экспертных систем, т.е. систем основанных на знаниях. Такие программы решения задач с представлением и применением фактических и эвристических знаний, совместной работой экспертов и инженеров по знаниям, разработчиков систем и логическим выводом позволяют переходить к новым информационным технологиям, к новой технологии программирования.

В настоящее время идет бурное развитие интеллектуальных систем, интеллектуальных концепций и технологий. Дисциплины, связанные с системами искусственного интеллекта, появились в связи с тенденциями образовательного процесса в сферах практической деятельности, связанных с решением задач интерпретации, диагностики, мониторинга, прогнозирования, планирования, проектирования, обучения, управления для плохо формализуемых проблем и зашумленных данных (знаний) при ограниченных ресурсах. Современный подход к решению таких проблем базируется на методах искусственного интеллекта.

Настоящие методические указания «Математические модели и методы искусственного интеллекта» направлены на формирование у обучающихся способностей к пониманию, совершенствованию и применению современного математического аппарата, используемого в теории искусственного интеллекта.

1 ИНТЕЛЛЕКТУАЛЬНЫЕ ЗАДАЧИ. ЭВРИСТИЧЕСКОЕ ПРОГРАММИРОВАНИЕ

1.1 Лабораторная работа №1 «Метод градиентного спуска»

1.1.1 Задания к лабораторной работе №1

1. Изучить теоретическую часть работы.

2. Реализовать метод градиентного спуска

3. Для функций двух видов: вогнутой и с вторичными минимумами применить реализованный метод, оценить скорость сходимости и возможность нахождения глобального минимума.

1.1.2 Методические указания к лабораторной работе №1

Цель работы - ознакомиться с методами поиска в непрерывном пространстве состояний в случаях отсутствия и наличия вторичных минимумов. Данная работа может выполняться группой студентов (максимально из 3 человек).

Теоретическая часть

Метод градиентного спуска.

Метод градиентного спуска - это классический метод поиска минимума дифференцируемой функции с аргументами, принимающими вещественные значения. Данный метод, как правило, применяется для многомерных функций, поскольку в одномерном случае существуют более эффективные методы поиска.

Как известно, градиент некоторый функции $f(x_1, x_2, \ldots, x_n)$ в некоторой точке показывает направление локального наискорейшего увеличения функции. Этот факт используется в методах градиентного спуска (подъема).

Эти методы описываются следующей последовательностью действий:

1. Установить номер итерации: i=0. Выбрать начальную точку $\left(x_1^{(0)}, x_2^{(0)}, ..., x_n^{(0)}\right)$.

2. Для текущей точки определить значение градиента:

$$\nabla f\left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right) = \left(\frac{\partial f}{\partial x_1}\left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right), \frac{\partial f}{\partial x_2}\left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right), ..., \frac{\partial f}{\partial x_n}\left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right)\right)$$

В случае если градиент не может быть вычислен аналитически, его компоненты могут быть оценены:

$$\frac{\partial f}{\partial x_j}\left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right) \approx \frac{f\left(x_1^{(i)}, ..., x_{j-1}^{(i)}, x_j^{(i)} + \Delta x, x_{j+1}^{(i)}, ..., x_n^{(i)}\right) - f\left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right)}{\Delta x}$$

3. Определить положение следующей точки:

$$\left(x_1^{(i+1)}, x_2^{(i+1)}, ..., x_n^{(i+1)}\right) = \left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right) - d\frac{\nabla f\left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right)}{\left|\nabla f\left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right)\right|}.$$

где d - параметр, определяющий скорость спуска, и положить i=i+1.

4. Перейти к шагу 2. если не выполнен критерий останова.

Существует несколько способов ввода критерия останова.

1) Наложить ограничение на количество итераций.

2) Проверять, что текущая точка или значение функции f меняются мало.

2.1.) При фиксированном шаге d изменение положения текущей точки происходит всегда на одну и ту же величину. В этом случае можно проверять изменение за несколько итераций и сравнивать с d:

$$\frac{\left|\left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right) - \left(x_1^{(i-k)}, x_2^{(i-k)}, ..., x_n^{(i-k)}\right)\right|}{kd}.$$

3) Адаптивный выбор шага d. Для этого на каждой итерации осуществляется выбор такого значения из $d_0 = d, d_1 = \frac{d}{w}, d_2 = d \cdot w$ (где w - некоторый параметр, как правило $w \in (1,2]$), что значение функции в точке

$$\left(x_1^{(i+1)}, x_2^{(i+1)}, ..., x_n^{(i+1)}\right) = \left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right) - d\frac{\nabla f\left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right)}{\left|\nabla f\left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right)\right|}$$ минимально.

Таким образом, если при большом d метод градиентного спуска «проскакивает» минимум, то d будет уменьшаться. Уменьшение d ниже заданного

порога также служит критерием остановки. Напротив, на пологих участках значение *d* будет увеличиваться.

При условии существования глобального минимума функции метод градиентного спуска обычно сходится (за исключением случаев, когда вдоль некоторого направления функция, монотонно убывая, стремится к некоторому конечному пределу при $|x| \to \infty$). Сходимость метода обеспечивается тем, что на каждой итерации выбирается такая точка $\left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right)$, что $f\left(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}\right) < f\left(x_1^{(i-1)}, x_2^{(i-1)}, ..., x_n^{(i-1)}\right)$. Метод, однако, не гарантирует нахождения глобального минимума, поскольку при достижении любого локального минимума метод не в состоянии определить направление на более глубокий минимум (и вообще обнаружить его существование) и останавливается в соответствии с выбранным критерием останова.

В связи с этим, выбор начальной точки может существенным образом сказываться на получаемом результате.

Экспериментальная часть

1. Провести анализ метода градиентного спуска. Для чего построить графики зависимости точности решения от числа итераций и сопоставить эти графики для функций различных типов. Для этого необходимо последовательно выполнить следующие действия.

1.1. Реализовать метод градиентного спуска и описать детали выбранной реализации.

1.2. Для нескольких вогнутых (обладающих единственным минимумом) функций построить графики зависимости $\left|x_i - x^*\right|$ (где x^* - истинное положение глобального минимума) от номера итерации i. Выбранные для исследования функции должны различаться средним значением модуля градиента (крутизной). Следует обратить внимание на выбор начальной точки которая должна отличаться от искомого минимума.

1.3. Для нескольких функций с многими локальными минимумами определить условия сходимости к глобальному минимуму. Определить, с какой вероятностью метод сходится к глобальному минимуму при случайном выборе (из некоторого фиксированного диапазона) начальной точки. Для исследования следует брать функции с разным числом минимумов.

1.4. Проанализировать полученные результаты. Определить предпочтительность использования метода при поиске минимума функции с одним и многими минимумами, опираясь на скорость сходимости и на легкость нахождения глобального минимума.

2. Сделать выводы по работе. Оформить отчет

Примеры функций для тестирования работы программы

№	Целевая функция	Глобальный минимум
1	$f(x)=\sin(x)+\sin\left(x^2\right)$	
2	$f(x)=\dfrac{6\cdot\sqrt[3]{6(x-3)^2}}{(x-1)^2+8}$	
3	$f(x)=-2x-8+3\cdot\sqrt[3]{6(x+4)^2}$	
4	$f(x)=\sqrt{3+x^2}+3\cos x$	
5	$f(x,y)=100\left(y-x^2\right)^2+(1+x)^2$ - функция Розенброка	$f(1,1)=0$
6	$f(x,y)=\left(y-x^2\right)\left(\dfrac{5,1}{4\pi^2}+x\dfrac{5}{\pi}\right)+10\left(1-\dfrac{1}{8\pi}\right)\cos x+10$	0,397887
7	$f(x,y)=20+e-20\exp\left(-0,2\sqrt{0,5\left(x^2+y^2\right)}\right)-\exp\dfrac{1}{2\left(\cos\left(2\pi x+\cos 2\pi y\right)\right)}$	0
8	$f(x,y)=\lvert x\rvert^2+\lvert y\rvert^3$	0
9	$f(x,y)=102+\left(x^2-10\cos(2\pi x)\right)+\left(y^2-10\cos(2\pi y)\right)$	0

1.2 Лабораторная работа №2 «Метод моделирования отжига»

1.2.1 Задания к лабораторной работе №2

1. Изучить теоретическую часть работы.

2. Реализовать метод моделирования отжига.

3. Для функций двух видов: вогнутой и с вторичными минимумами применить реализованный метод, оценить скорость сходимости и возможность нахождения глобального минимума.

1.2.2 Методические указания к лабораторной работе №2

Цель работы - ознакомиться с методами поиска в непрерывном пространстве состояний в случаях отсутствия и наличия вторичных минимумов. Данная работа может выполняться группой студентов (максимально из 3 человек).

Теоретическая часть

Метод моделирования отжига

Метод моделирования отжига предназначен для поиска глобального минимума некоторой функции $f : S \to R$, где S - некоторое пространство (необязательно непрерывное), элементы которого интерпретируются как состояния некоторой воображаемой физической системы, а значения самой функции - как энергия этой системы $E=f(x_1, x_2, \ldots, x_n)$ в состоянии $(x_1, x_2, \ldots, x_n) \in S$.

В методе моделирования отжига система в каждый момент времени находится в некотором состоянии $\left(x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)}\right)$ а также обладает некоторой температурой T, которая является управляемым параметром.

На каждой итерации система случайным образом переходит в новое состояние $\left(x_1^{(i+1)}, x_2^{(i+1)}, \ldots, x_n^{(i+1)}\right)$. Механизм выбора нового состояния состоит из двух частей:

1. Выбирается $\left(x_1^{(i+1)}, x_2^{(i+1)}, \ldots, x_n^{(i+1)}\right)$ в соответствии с некоторой функцией распределения $g\left(x_1^{(i+1)}, x_2^{(i+1)}, \ldots, x_n^{(i+1)}; x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)}; T\right)$. Как правило, эта функция

зависит только от расстояния $\left|\left(x_1^{(i+1)}, x_2^{(i+1)}, \ldots, x_n^{(i+1)}\right) - \left(x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)}\right)\right|$, причем с увеличением этого расстояния вероятность перехода понижается.

2. После случайного выбора $\left(x_1^{(i+1)}, x_2^{(i+1)}, \ldots, x_n^{(i+1)}\right)$ проверяется вероятность перехода в это новое состояние, исходя из разности энергий и текущей температуры: $h(\Delta E, T)$,

$$\Delta E = f\left(x_1^{(i+1)}, x_2^{(i+1)}, \ldots, x_n^{(i+1)}\right) - f\left(x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)}\right).$$

Здесь $h(\Delta E, T)$ показывает вероятность перехода в состояние с другой энергией. Проверка производится следующим образом: выбрасывается случайное число из диапазона [0, 1]. Если это число оказывается меньше, чем значение вероятности $h(\Delta E, T)$, то новое состояние $\left(x_1^{(i+1)}, x_2^{(i+1)}, \ldots, x_n^{(i+1)}\right)$ принимается, в противном случае шаг 1 повторяется. Функция $h(\Delta E, T)$, как правило, стремится к 1 при $\Delta E \to -\infty$ и стремится к 0 при $\Delta E \to +\infty$ (то есть предпочтение в среднем отдается состояниям с меньшей энергией).

Поскольку метод моделирования отжига базируется на физических принципах, то и функции распределения вероятностей $g\left(x_1^{(i+1)}, x_2^{(i+1)}, \ldots, x_n^{(i+1)}; x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)}; T\right)$ и $h(\Delta E, T)$ также часто заимствуются из физики. В частности, достаточно популярен больцмановский отжиг, в котором:

$$g\left(x_1^{(i+1)}, x_2^{(i+1)}, \ldots, x_n^{(i+1)}; x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)}; T\right) =$$

$$= (2\pi T)^{-\frac{n}{2}} \exp\left(\frac{-\left|\left(x_1^{(i+1)}, x_2^{(i+1)}, \ldots, x_n^{(i+1)}\right) - \left(x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)}\right)\right|^2}{2T}\right)$$

$$h(\Delta E, T) = \frac{1}{1 + \exp\left(\frac{\Delta E}{T}\right)} \approx \frac{-\Delta E}{T}.$$

Таким образом, температура T определяет, насколько в среднем может меняться текущее состояние $\left(x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)}\right)$, а также то, насколько в среднем может меняться энергия системы при переходе в новое состояние.

Поскольку переход в состояния с меньшей энергией более вероятен чем переход в состояния с более высокой энергией, то система будет больше времени проводить именно в низкоэнергетических состояниях.

Чтобы обеспечить сходимость системы к некоторому состоянию с наименьшей энергией, температуру системы понижают с переходом к следующей итерации. В больцмановском отжиге применяется следующий закон понижения температуры:

$$T_i = \frac{T_0}{\ln(1+i)},$$

где номер итерации $i>0$. Такой закон может, однако, потребовать большое число итераций, особенно при больших значениях начальной температуры T_i - в связи с чем используется более быстрое понижение температуры:

$$T_i = \frac{T_0}{1+i}.$$

Начальная температура неявно задает область, в которой будет осуществляться поиск глобального минимума, а также определяет необходимое для сходимости число итераций.

Экспериментальная часть

1. Провести анализ метода моделирования отжига. Для чего построить графики зависимости точности решения от числа итераций и сопоставить эти графики для функций различных типов. Для этого необходимо последовательно выполнить следующие действия.

1.1. Реализовать метод моделирования отжига и описать детали выбранной реализации.

1.2. Для нескольких вогнутых (обладающих единственным минимумом) функций построить графики зависимости $\left|x_i - x^*\right|$ (где x^* - истинное положение глобального минимума) от номера итерации i. Выбранные для исследования функции должны различаться средним значением модуля градиента (крутизной). Следует обратить внимание на выбор начальной точки которая должна отличаться от искомого минимума. В методе моделирования отжига следует также обратить внимание на задание начальной температуры и исследовать ее влияние на скорость сходимости.

1.3. Для нескольких функций с многими локальными минимумами определить условия сходимости к глобальному минимуму. Определить, какая начальная температура обеспечивает сходимость к глобальному минимуму. Для исследования следует брать функции с разным числом минимумов.

1.4. Проанализировать полученные результаты. Определить предпочтительность использования метода при поиске минимума функции с одним и многими минимумами, опираясь на скорость сходимости и на легкость нахождения глобального минимума.

2. Провести сравнительный анализ методов градиентного спуска (индивидуальное задание №1) и моделирования отжига. Определить предпочтительность использования каждого метода при поиске минимума функции с одним и многими минимумами, опираясь на скорость сходимости и на легкость нахождения глобального минимума.

3. Сделать выводы по работе. Оформить отчет

Примеры функций для тестирования работы программы приведены в лабораторной работе №1

1.3 Лабораторная работа №3 «Генетический алгоритм поиска минимума функции»

1.3.1 Задания к лабораторной работе №3

1. Изучить теоретическую часть работы.

2. Реализовать генетический алгоритм поиска минимума.

3. Для функций двух видов: вогнутой и с вторичными минимумами применить реализованный метод, оценить скорость сходимости и возможность нахождения глобального минимума. Провести оценку сходимости в зависимости от размера начальной популяции и скорости мутаций.

1.3.2 Методические указания к лабораторной работе №3

Цель работы - ознакомиться с методами поиска в непрерывном пространстве состояний в случаях отсутствия и наличия вторичных минимумов. Данная работа может выполняться группой студентов (максимально из 3 человек).

Теоретическая часть

Генетические алгоритмы и эволюционные стратегии поиска

Генетические алгоритмы (ГА) предназначены для нахождения экстремумов функций от произвольных объектов, используя при этом приемы, заимствованные у естественной эволюции. Сами объекты трактуются как некоторые организмы, а оптимизируемая функция - как приспособленность организмов, или фитнесс-функция. Множество возможных объектов взаимно однозначно отображается на некоторое подмножество множества битовых строк (обычно фиксированной длины). Эти строки трактуются как хромосомы (геномы).

Особенность генетических алгоритмов заключается в том, что они работают с битовыми строками, не опираясь на структуру исходных объектов, что позволяет применять ГА без модификации для любых объектов. Единственно, что требуется для такого применения. - это перекодирование объектов в геномы.

Эволюционные стратегии отличаются от генетических алгоритмов лишь в том, что в них используются структурированные описания объектов, то есть такие

описания, элементы которых имеют вполне четкий смысл в той предметной области, к которой относятся данные объекты.

Двумя основными механизмами эволюции, наиболее часто моделируемыми в генетических алгоритмах и эволюционных стратегиях, являются скрещивание и мутации. При этом схема эволюционных методов поиска выглядит следующим образом.

1. Сгенерировать начальную популяцию (случайную совокупность объектов).

2. Выбрать родительские пары.

3. Для каждой родительской пары с использованием оператора скрещивания породить потомство.

4. В хромосомы порожденного потомства внести случайные искажения оператором мутации.

5. Произвести отбор особей из популяции по значению их фитнесс-функции.

6. Повторять шаги 2-4, пока не выполнится критерий остановки.

Рассмотрим каждый из шагов чуть подробнее.

1. Генерация начальной популяции обычно производится равномерно по пространству генов (или по пространству описаний объектов). Размер популяции - установочный параметр.

2. Выбор родительских пар может осуществляться различными способами. Выбор родителей осуществляется в два этапа: выбор первого родителя и формирование пары. При выборе одного родителя обычно используется один из следующих способов:

- с равной вероятностью выбирается любая особь из имеющейся популяции:

- особь выбирается случайно с вероятностью, пропорциональной значению фитнесс-функции: то есть в этом случае значение фитнесс-функции сказывается не только на том, какие особи останутся в популяции в результате отбора, но и на то, сколько потомства они произведут.

Выбор второго родителя осуществляется по одному из следующих критериев:

- независимо от уже выбранного родителя (то есть второй родитель выбирается абсолютно так же, как и первый),

- на основе ближнего родства,

- на основе дальнего родства.

В последних двух случаях выбор одного родителя влияет на выбор другого родителя: с большей вероятностью формируются пары, состоящие из особей, которые больше похожи друг на друга (то есть ближе находятся в пространстве геномов или описаний объектов) в случае использования ближнего родства и меньше похожи - в случае дальнего родства. В генетических алгоритмах в качестве меры близости обычно используется *расстояние Хемминга*.

3. Оператор скрещивания - это оператор, который определяет, как из хромосом родителей формировать хромосомы их потомства. Часто применяется следующий оператор скрещивания: хромосомы делятся в некоторой случайной точке и обмениваются этими участками (то есть. все. что идет до этой точки, берется от одного родителя, а все, что после, - от другого). Это одноточечный кроссинговер. В многоточечном кроссинговере таких участков обмена больше.

При равномерном скрещивании каждый бит хромосомы берется от случайного родителя.

4. Мутации обычно осуществляются как случайная замена одного бита хромосомы. Скорость мутаций выражается в том, как часто они осуществляются. Это управляемый параметр, который влияет на скорость сходимости и вероятность попадания в локальный экстремум.

5. Отбор особей в новую популяцию чаще всего осуществляется одной из двух стратегий:

- пропорциональный отбор, при котором вероятность того, что некая особь останется в следующей популяции, пропорциональна значению фитнесс-функции этой особи:

- элитный отбор, при котором из популяции отбираются лучшие по значению фитнесс-функции особи, и они детерминированным образом переходят в следующую популяцию.

Формирование новой популяции может осуществляться как на основе потомков и родителей, так и на основе только потомков в зависимости от конкретной реализации.

6. Основные критерии останова базируются либо на числе сменившихся поколений (количестве выполненных итераций), либо на некотором условии стабильности популяции. Число поколений не является адаптивным по отношению к виду фитнесс-функции. поэтому используется обычно в качестве не основного критерия. Проверка стабильности популяции в общем виде, как правило, требует значительных вычислений, поэтому чаше используется проверка того, что максимальное по популяции значение фитнесс-функции перестает заметно расти от поколения к поколению. Все эти критерии останова соответствуют критериям останова в методе градиентного спуска, но учитывают ту специфику генетических алгоритмов, что в них на каждой итерации одновременно рассматривается не одно, а несколько решений.

Особенности реализации генетических операторов в эволюционных стратегиях.

Рассмотрим особенности реализации генетических операторов в эволюционных стратегиях на примере объектов, описаниями которых являются двухкомпонентные векторы: (*x, y*).

1. Генерация начальной популяции может осуществляться путем выбора случайных векторов из области $[x_{\min}, x_{\max}] \times [y_{\min}, y_{\max}]$, где величины $x_{\min}, x_{\max}, y_{\min}, y_{\max}$ задают ожидаемые минимальные и максимальные значения переменных *x* и *y* искомого положения экстремума фитнесс-функции.

2. При выборе родителей особенность эволюционных стратегий выражается в способе задания меры родства. В данном случае, мерой родства двух особей (x_1, y_1) и (x_2, y_2) может служить евклидово расстояние: $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$.

3. Результатом скрещивания двух особей в рассматриваемом случае будет являться особь $(\varepsilon x_1 + (1-\varepsilon) x_2, \varepsilon y_1 + (1-\varepsilon) y_2)$, где $\varepsilon \in [0,1]$ - случайная величина.

4. Результатом мутации для особи (*x*, *y*) будет являться особь $\left(x+\delta_x, y+\delta_y\right)$, где δ_x, δ_y - случайные величины. Их распределение вероятностей может быть выбрано гауссовым или, для простоты программной реализации, равномерным в некотором интервале. Дисперсия этих величин определяет скорость мутаций.

5 и 6. Операторы отбора и критерии останова в эволюционных стратегиях не имеют особых отличий от тех, которые используются в генетических алгоритмах.

Экспериментальная часть

1. Проанализировать зависимость характеристик работы эволюционных стратегий (скорости их сходимости и возможности обнаружения глобального экстремума фитнесс-функции) от установочных параметров - размера начальной популяции, скорости и типа мутаций, типа скрещивания и способов отбора родительских особей, а также от вида оптимизируемой функции. Для этого необходимо выполнить следующую последовательность действий.

1.1. Выполнить реализацию эволюционной стратегии и описать ее детали.

1.2. Для нескольких выпуклых (обладающих единственным максимумом) функций построить графики зависимости $\left|x_i - x^*\right|$ (где x^* - истинное положение глобального минимума, x_i – максимальное по текущей популяции значение фитнесс-функции) от номера популяции i при различных значениях скорости мутации и размера начальной популяции. Следует обратить внимание на выбор диапазона $\left[x_{\min}, x_{\max}\right]\times\left[y_{\min}, y_{\max}\right]$. Графики необходимо построить для двух случаев: когда точка глобального экстремума попадает в этот диапазон и когда она находится вне него.

1.3. Для нескольких функций с многими локальными экстремумами провести те же исследования, что и для функций с единственным экстремумом.

1.4. По результатам проведенных экспериментов, построить таблицы и графики зависимостей погрешности результата от размера начальной популяции, скорости и типа мутаций, при различных установочных параметрах: типа

скрещивания, способов отбора родительских особей. Проанализировать полученные результаты.

1.5. Определить, какое именно влияние оказывает скорость мутаций на функционирование алгоритма поиска. Определить, может ли большая скорость мутаций быть заменена большим размером популяции.

1.6. Установить различия в характере работы алгоритма поиска для функций с единственным и многими экстремумами.

2. Сделать выводы по работе. Оформить отчет

Примеры функций для тестирования работы программы приведены в лабораторной работе №1

2 РАЗРАБОТКА ЭКСПЕРТНЫХ СИСТЕМ

2.1 Лабораторная работа №4 «Методы построения ассоциативных сетей»

2.1.1 Задания к лабораторной работе №4

Реализовать программу проведения тестирования в некоторой предметной области для выявления ассоциативных связей между понятиями. Провести тестирование группы людей. Усреднить и проанализировать интервалы времени, необходимые испытуемым для установления наличия или отсутствия ошибок в утверждениях, связывающих два понятия, на основе чего определить длину пути в ассоциативной сети между соответствующими узлами. Восстановить ассоциативную сеть.

2.1.2 Методические указания к лабораторной работе №4

Цель: ознакомиться с методами извлечения ассоциативных связей между понятиями на основе психофизических экспериментов

Теоретическая часть

Описание методики построения ассоциативных сетей.

Очень давно было замечено, что человек имеет склонность к ассоциированию. Это послужило отправной точкой для создания ассоционистских теорий, в которых смысл некоторого понятия определяется через его ассоциативные связи с другими понятиями, которые в совокупности образуют своего рода сеть. Понятия являются основой нашего знания о мире, поэтому такую ассоциативную сеть можно рассматривать в качестве представления знаний. Ассоциативные связи возникают на основе опыта и выражают эмпирические отношения между признаками или поведением объектов.

Например, на основании опыта мы ассоциируем понятие «снег» с другими понятиями, такими как «зима», «холод», «белый», «лед» и т.д. Наши знания о снеге и истинность утверждений типа «снег белый» выражаются в виде сети ассоциаций. Если такая сеть реально используется человеком, то можно ли узнать, какова она?

Для ответа на этот вопрос психологами проводится следующий эксперимент. Людям задают вопросы из некоторой области, например, об особенностях и поведении птиц. Эти вопросы выбираются очень простыми: «Может ли голубь летать?». «Может ли канарейка петь?». «Является ли ворона птицей?».

При этом проверяется, конечно же. не правильность ответов на вопросы. Вместо этого оценивается время, требуемое для ответа. Все вопросы составляются так. что в них фигурируют два понятия. Наличие непосредственной ассоциации между этими понятиями должно было сказываться на времени ответа.

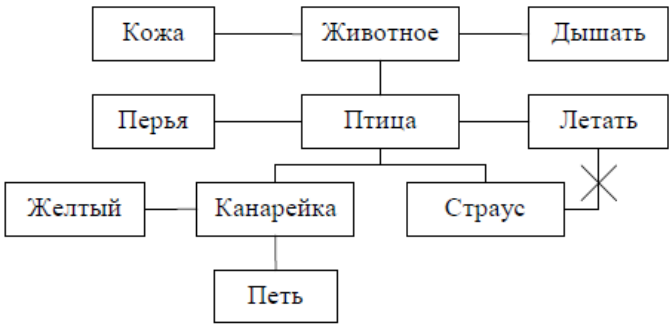
Как оказывается, времена ответов действительно различаются. Так, например, для ответа на вопрос «Может ли канарейка летать?» требуется больше времени, чем для ответа на вопрос: «Может ли канарейка петь?».

Коллинс и Квиллиан, ставившие этот эксперимент впервые, объясняли большее время ответа на некоторый вопрос тем. что участвующие в вопросе понятия, хотя и расположены достаточно близко в ассоциативной сети, не являются непосредственно связанными. При этом оказалось, что свойства объектов запоминаются людьми на наиболее абстрактном уровне.

Вместо того чтобы запоминать каждое свойство для каждой птицы (канарейки летают, вороны летают, голуби летают), люди хранят информацию о том. что

канарейки - птицы, а птицы, как правило, способны летать. Поскольку поют не все птицы, то способность к пению - более частное свойство, которое запоминается на менее абстрактном уровне, чем способность летать, поэтому и ответ на вопрос «Может ли канарейка петь?» требует меньше времени, так же. как и ответ на вопрос «Является ли канарейка желтой?». Таким образом, быстрее всего вспоминаются самые конкретные свойства объектов. Как оказалось, и исключения из правил также запоминаются на самом нижнем, детальном уровне. Примером может служить вопрос «Может ли страус летать?», который для ответа требует меньше времени, чем тот же вопрос о канарейке или другой «нормальной» птице.

На рисунке представлен пример фрагмента ассоциативной сети, которая могла бы быть построена в результате такого эксперимента.



В рамках данной работы предполагается, что в областях, относящихся к здравому смыслу, ассоциативные связи у разных людей устанавливаются сходным образом, поэтому для получения надежных значений времен ответов можно и нужно проводить усреднение по некоторой группе людей. При этом сферу понятий для тестирования нужно выбирать таким образом, чтобы ответы на составленные вопросы были очевидными для всех испытуемых. При малом числе испытуемых может использоваться повторное тестирование одного и того же человека, но необходимо, чтобы были выполнены следующие условия: база вопросов является достаточно большой (порядка 100 вопросов) и время перед повторным тестированием должно быть значительным (не менее суток).

Экспериментальная часть

В данной работе производят построение ассоциативной сети для некоторой ограниченной системы общеизвестных понятий на основе времени ответа человека

на вопросы, связывающие пары понятий. Для этого необходимо выполнить следующую последовательность действий.

1. Реализовать программу тестирования, которая выбирает из базы случайные, но неповторяющиеся вопросы и определяет с высокой точностью (порядка 0.1 секунды) время ответа. Ввод ответа рекомендуется осуществлять по факту нажатия одной из двух заранее определенных клавиш. Не рекомендуется использовать способы ввода, требующие перемещения мышки или ввода слов с нажатием Enter, поскольку это существенно снижает точность оценки времени. Показ очередного вопроса рекомендуется осуществлять только при готовности человека (о чем он может сообщать программе путем нажатия некоторой третьей клавиши). Рекомендуется, чтобы программа автоматически записывала результаты ответов в лог-файл.

2. Составить базу вопросов. Для этого следует выбрать область со сложными (иерархическими) отношениями между понятиями, которые предположительно описываются ассоциативной сетью. Однако сами понятия должны быть общеизвестными, а ответы на вопросы - не требующими привлечения дополнительных знаний.

3. Провести тестирование группы людей. При этом требуется установить предпочтительную продолжительность тестирования одного человека. Для этого необходимо определить, спустя какой промежуток времени начинает систематически увеличиваться время ответов или число ошибок. При тестировании каждого человека требуется объяснить ему условия эксперимента и продемонстрировать их на нескольких вопросах, которые затем не учитываются при анализе результатов.

4. Произвести усреднение времен ответов на вопросы по группе людей и вычислить дисперсии времен для каждого вопроса. Установить достоверность различия средних времен ответов на разные вопросы.

5. Построить ассоциативную сеть, если времена ответов на вопросы достоверно различаются. Построение сети рекомендуется начинать с пар понятий,

которым соответствуют наименьшие времена ответов. Эти понятия должны быть непосредственно связаны в сети.

6. Проанализировать полученные результаты. Если построение ассоциативной сети осуществлено удачно, то проанализировать ее структуру, в противном случае определить возможные причины неудачи. Сделать выводы по работе.

Содержание отчета.

1. Цель работы.

2 Задание.

3. Списки вопросов и вариантов ответов.

4. Протоколы проверки работоспособности на примерах.

5. Выводы по работе.

2.2 Лабораторная работа №5 «Продукционный метод построения базы знаний»

2.2.1 Задания к лабораторной работе №5

Выбрать систему понятий из некоторой предметной области и описать взаимосвязи между этими понятиями. Реализовать программу на языке Пролог, в которой установленные взаимосвязи описываются логическими формулами в исчислении предикатов в виде общих правил и частных фактов. Определить возможности интерпретатора Пролога по ответу на вопросы из данной области.

2.2.2 Методические указания к лабораторной работе №5

Цель. Ознакомиться с методами построения баз знаний, включающих понятия и взаимосвязи между ними, которые описаны в рамках логических представлений.

Теоретическая часть

Продукционные системы

База знаний - наиболее важная компонента экспертной системы, на которой основаны ее «интеллектуальные способности». Существует несколько способов представления знаний в ЭС, однако общим для всех них является то, что знания

представлены в символьной форме (элементарными компонентами представления знаний являются тексты, списки и другие символьные структуры). Тем самым в ЭС реализуется принцип символьной природы рассуждений, который заключается в том, что процесс рассуждения представляется как последовательность символьных преобразований.

Наиболее простым с точки зрения построения и широко используемым типом моделей принятия решений являются **продукционные системы** (ПС). Они представляют собой структурированные наборы **продукционных правил (ПП)** вида:

PR = <S, N, F, A => C, W>,

где S - сфера применения данного правила;

X - номер или имя правила;

F - условие активизации данного правила;

A => *C* - ядро;

W - постусловие.

В состав правил могут входить условия активизации F, которые представляют собой либо переменную, либо логическое выражение (предикат). Когда F принимает значение «истина», ядро продукции может быть активизировано. Если F «ложно», то ядро не активизируется.

Постусловие W описывает, какие изменения следует внести в ПС, и актуализируется только после того, как ядро продукции реализовалось.

Интерпретация ядра может быть различной в зависимости от вида A и C, находящихся по разные стороны знака секвенции «=>». Наиболее часто в ПС используют ПП вида

«если A, то C»,

где A и C - логические выражения, которые могут включать в себя другие выражения;

C может выполняться с определенной вероятностью: «если A, то возможно C».

Возможность может определяться некоторыми оценками реализации ядра. Например, если задана вероятность выполнения С при актуализации А, то ПП может быть таким: «если А, то с вероятностью Р выполнить С».

Оценка реализации ядра может быть лингвистической, связанной с лингвистической переменной: «если А, то с большей долей уверенности возможно С».

Прогнозирующие ПП, в которых описываются, например, последствия, ожидаемые при актуализации А: «если А, то с вероятностью Р можно ожидать С».

Достоинствами ПС являются:

- удобство описания процесса принятия решения экспертом (формализация его интуиции и опыта);

- простота редактирования модели;

- прозрачность структуры.

ПС в качестве моделей применимы в следующих случаях:

- не могут быть построены строгие алгоритмы или процедуры принятия решений, но существуют эвристические методы решения;

- существует, по крайней мере, один эксперт, который способен явно сформулировать свои

знания и объяснить свои методы применения этих знаний при принятии решения;

- пространство возможных решений относительно невелико (число решений счетно);

- задачи решаются методом формальных рассуждений:

- данные и знания надежны и не изменяются со временем.

Пролог

Пролог - это язык программирования, предназначенный для обработки символьной информации. Особенно хорошо он приспособлен для решения задач, в которых фигурируют объекты и отношения между ними.

Одним из замечательных свойств Пролога является то, что это достаточно простой язык, н студенты могли бы использовать его непосредственно в процессе

изучения вводного курса по искусственному интеллекту. Рассмотрим некоторые особенности языка Пролог.

- Программирование на Прологе состоит в определении отношений и в постановке вопросов, касающихся этих отношений

- Пролог-программа состоит из предложений. Каждое предложение заканчивается точкой.

- Прологовские предложения бывают трех типов: факты, правила и вопросы.

Факты содержат утверждения, которые являются всегда, безусловно верными.

Правила содержат утверждения, истинность которых зависит от некоторых условий.

С помощью **вопросов** пользователь может спрашивать систему о том, какие утверждения являются истинными. Чтобы получить вопрос необходимо перед предложением поставить символы: ?-.

- Предложения Пролога состоят из головы и тела. **Тело** - это список целей, разделенных запятыми или точкой запятой.

- Последовательность целей, перечисляемая через:

, означает конъюнкцию этих целевых утверждений (логическое И):

; означает дизъюнкцию этих целевых утверждений (логическое ИЛИ).

Факты - это предложения, имеющие пустое тело. **Вопросы** имеют только тело. **Правила** имеют голову и (непустое) тело. В правилах голова и тело разделены символами :-.

Аргументы отношения могут быть: конкретными объектами (**атомами**) или абстрактными объектами (**переменными**). **Атомы** могут представлять собой цепочки следующих символов:

- прописные буквы A, B,.... Z

- строчные буквы a, b,.... z

- цифры 0, 1, 2..... 9

- специальные символы, такие как *+ - / = : . & _ ~

Атомы можно создавать тремя способами:

- из цепочки букв, цифр и символа подчеркивания _. начиная такую цепочку со строчной буквы:

aHCa

s25

x_

- из специальных символов:

<--->

====>

…

.

::=

Следует соблюдать некоторую осторожность, поскольку часть цепочек специальных символов имеют в Прологе заранее определенный смысл. Примером может служить :-.

- из цепочки символов, заключенной в одинарные кавычки. Это удобно, если мы хотим, например, иметь атом, начинающийся с прописной буквы. Заключая его в кавычки, мы делаем его отличным от переменной:

'AHCa'

Переменные - это цепочки, состоящие из букв, цифр и символов подчеркивания. Они начинаются с прописной буквы или с символа подчеркивания:

_X

X

Result

Если переменная встречается в предложения только один раз, то нет необходимости изобретать ей имя. Можно использовать так называемую "анонимную" переменную, которая записывается в виде одного символа подчеркивания.

Пролог-система рассматривает вопросы как цели, к достижению которых нужно стремиться. Ответ на вопрос может оказаться или положительным, или отрицательным в зависимости от того, может ли быть соответствующая цель

достигнута или нет. Если на вопрос существует несколько ответов, пролог-система найдет столько из них, сколько пожелает пользователь.

По ходу вычислений вместо переменной может быть подставлен другой объект. Мы говорим в этом случае, что переменная **конкретизирована**.

Краткое описание основ языка Пролог

Программа на языке Пролог преимущественно состоит из списка логических утверждений, которые можно разделить на факты и общие правила. Эти утверждения составляются из имен предикатов, переменных и их значений, которые представляют собой символьную строку, начинающуюся с буквы, а также совокупности логических операций. В конце каждого утверждения ставится точка ".". Переменные обязательно начинаются с прописной буквы, поэтому все значения должны начинаться со строчной буквы. Использующиеся в Прологе обозначения логических операций представлены в таблице.

Имя операции	Логическая операция	Обозначение в Прологе
и	∧	,
или	∨	;
если	<=	:-
не	¬	not

Рассмотрим некоторые примеры утверждений на языке Пролог. Ниже приведен пример списка фактов, не содержащих переменных.

man(serg).

man(alex).

father(4erg. alex).

mother(kat. serg).

Эти факты говорят, что предикаты "man", "father", "mother" истинны при указанных значениях аргументов. Пролог работает в рамках предположения о замкнутости базы знаний. Это означает то, что все предикаты будут ложны для всех значений переменных, для которых не было указано обратное.

Общие правила в Прологе записываются так же, как и факты, но в них присутствуют переменные. При этом подразумевается, что общее правило верно при всех значениях всех входящих в него переменных. Иными словами, правило

pl(X):-p2(X,Y). означает логическое выражение (∀X,Y) p2(X,Y)=>p1(X).

Рассмотрим некоторые примеры общих фактов, записанных на языке Пролог, и приведем для них соответствующие формулы в логике предикатов.

father(X, Y) :- man(X), parent(X, Y).

(∀X, Y) man(X) ∧ parent(X, Y) => father(X, Y)

Если некто X является мужчиной и родителем Y, то X — отец Y.

parent(X, Y) :-mother(X, Y); father(X, Y).

(∀X, Y) mother(X, Y) ∨ father(X, Y) => parent(X, Y)

X является родителем Y, если X — отец или мать Y.

grandfather(X, Y) :- father(X, Z), parent(Z, Y).

(∀X, Y) (grandfather(X, Y) <= (∃Z) father(X, Z) ∧ parent(Z,Y))

X является дедушкой Y только тогда, когда X является отцом (некоторого) родителя Y.

За исполнение программы на языке Пролог отвечает интерпретатор, который интерактивно взаимодействует с пользователем, отвечая на его запросы. Запрос к интерпретатору также представляется в виде логического выражения, истинность которого требуется установить. Рассмотрим следующую простую программу.

mother(X, Y) :- woman(X), parent(X, Y).

father(X, Y) :- man(X), parent(X, Y).

grandfather(X, Y) :- father(X, Z). parent(Z, Y).

man(serg).

man(alex).

woman(kat).

parent(serg, victor).

parent(kat, victor).

parent(victor, alex).

И посмотрим на следующие запросы, начинающиеся с приглашения "? -", на которые интерпретатор возвращает некоторый ответ.

?- father(serg, victor).

Yes

?- grandfather(serg, alex).

Yes

?- mother(kat. alex).

No

?- father(victor. alex).

No

Как видно, для обработки этих запросов интерпретатору необходимо выполнить логический вывод, а не просто обратиться к базе правил. Также видно, как действует предположение о замкнутости базы знаний: поскольку в базе нет фактов об истинности parent(kat, alex) и man(victor), то последние два запроса возвращают "Нет".

В Прологе существуют также и запросы в виде выражений с переменными. Посмотрим, как работают эти запросы.

?-mother(kat, X).

X = victor

Yes

?- grandfather(X, alex).

X = serg

Yes

?- grandfather(X, Y).

X = serg, Y = alex

Yes

?- grandfather(X, kat).

No

?-man(X).

X = serg

Yes

Здесь можно отметить следующее. В подобного рода запросах предполагается квантор существования для входящих в него переменных. Выражение истинно, если найдется хотя бы одно подходящее значение X (не имеет значения, как именно обозначать переменные в запросе). В действительности, при нахождении в базе первого значения, для которого введенное выражение истинно, интерпретатор спрашивает пользователя, продолжать ли поиск. Поиск продолжается при нажатии команды ":" (или) до тех пор. пока не будет дан ответ "Нет" в связи с исчерпанием всех возможностей, например.

?- man(X).

X = serg ;

X = alex ;

No

Более сложные выражения в запросах также допустимы, например,

?- mother(kat, X), father(Y, X).

X = victor, Y = serg

Yes

Этот запрос позволяет определить, кто является отцом сына kat.

Следует обратить внимание на некорректность работы программы, содержащей противоречия вида

something(X) :- not(something(X)).

а также циклические определения вида

man(X) :- not(woman(X)).

woman(X) :- not(man(X)).

Экспериментальная часть

В данной работе производят построение простейшей базы знаний в рамках логического представления с использованием языка Пролог и устанавливают возможности и ограничения: 1) логических представлений при описании понятий и взаимосвязей между ними; 2) интерпретатора отвечать на запросы. Для этого необходимо выполнить следующую последовательность действий.

1. Выбрать систему взаимосвязанных понятий, подобную системе родственных отношений, но отличную от нее.

2. Формализовать отношения между понятиями в форме логических выражений в рамках исчисления предикатов первого порядка.

3. Реализовать логические выражения как общие правила в программе на языке Пролог; дополнить программу совокупностью частных фактов. Полученная в результате программа является основой отчета по лабораторной работе.

4. Определить возможности созданной базы знаний по ответу на запросы, требующие логического вывода (т.е. ответы на которые не содержатся в базе в явном виде). Установить запросы, на которые ответ интерпретатора не соответствует ожидаемому (уделить внимание неявному предположению замкнутости базы знаний).

5. Проанализировать полученные результаты. Сделать выводы по работе: сформулировать эмпирически обоснованные преимущества и недостатки логических представлений для систем взаимосвязанных понятий.

Содержание отчета.

1. Цель работы.

2 Задание.

3. Списки вопросов и вариантов ответов.

4. Протоколы проверки работоспособности на примерах.

5. Выводы по работе.

3 РАЗРАБОТКА СИСТЕМ РАСПОЗНАВАНИЯ ОБРАЗОВ

3.1 Лабораторная работа №6 «Представление изображений n-мерном векторном пространстве»

Цель работы – приобретение и закрепление знаний и получение практических навыков работы с простейшими алгоритмами распознавания на основе представления изображений в виде точек в n-мерном векторном пространстве.

3.1.1 Задания к лабораторной работе №6

1. Реализовать методы эталонных образов и ближайшего соседа для распознавания различных объектов в *n*-мерном векторном пространстве с помощью разных видов расстояний, в том числе и для качественных признаков.

2. Путем варьирования обучающей выборки (*n* должно быть не менее 5) определить влияние следующих факторов на вероятности правильного распознавания: наличие в обучающей выборке выбросов, размер обучающей выборки, форма областей, занимаемых классами в пространстве признаков.

3. Разработайте алгоритм и программу, моделирующую распознавание различных объектов по их принадлежности к шарообразным или конусообразным областям в n-мерном векторном пространстве. Задайтесь размерностью n-мерного векторного пространства, числом k классов и несколькими распознаваемыми объектами. Определите принадлежность предъявленных объектов к тому или иному классу при шарообразных и конусообразных областях.

4. Реализовать метод обобщенных решающих функций. Для поиска оптимальных параметров разрешающей функции реализовать методы наименьших квадратов и опорных векторов.

5. Путем варьирования обучающей выборки определить влияние следующих факторов на вероятности правильного распознавания: наличие в обучающей выборке выбросов, размер обучающей выборки, форма областей, занимаемых классами в пространстве признаков.

3.1.2 Методические указания к лабораторной работе №6

Теоретическая часть

Представление изображений в векторной форме

Каждая ось n-мерного пространства естественным образом соотносится с одним из n входов или с одним из n рецепторов распознающей системы. Каждый из рецепторов может находиться в одном из m состояний, если они дискретны, или иметь бесконечно большое число состояний, если рецепторы непрерывны. В зависимости от вида используемых рецепторов может порождаться непрерывное, дискретное или непрерывно-дискретное n-мерное пространство.

Как правило, в пространстве изображений вводится метрика - функция, которая каждой упорядоченной паре n-мерных векторов x и y пространства ставит в соответствие действительное число d(x, y), обладающее свойствами:

1) d(x, y) > 0, d(x, y)=0 тогда и только тогда, когда x = y;

2) d(x, y) = d(y, x);

3) d(x, y)<d(x, z)+d(z, y).

Введение метрики d(x, y) позволяет говорить о близости или удаленности точек в этом пространстве или о мере сходства анализируемых изображений.

Рассмотрим общие требования к мере сходства изображений.

Пусть задано некоторое конечное множество *S* = {S1 S2, ..., Sm) входных изображений, каждое из которых является точкой в n-мерном пространстве изображений. Мера сходства изображений L(Si, Sj) должна удовлетворять общим требованиям

1) свойству симметрии L(Si, Sj) = L(Sj, Si);

2) область значений функции L(Si, Sj) - множество неотрицательных чисел;

3) мера сходства изображения с самим собой должна принимать экстремальное значение но сравнению с любым другим изображением, т.е. в зависимости от способа введения меры сходства должно выполняться одно из двух соотношений:

L(Sk, Sk) = max L(Sk, Si)

L(Sk, Sk) = min L(Sk, Si)

4) в случае непрерывного т-мерного пространства и компактных образов функция L(Si, Sj) должна быть монотонной функцией удаления точек Si и Sj друг от друга в этом пространстве.

Анализ свойств метрики и меры сходства изображений показывает, что требования к функции L(Si, Sj) нетрудно выполнить в метрических пространствах. В частности, если в метрическом пространстве введено расстояние, то оно может быть использовано в виде меры сходства изображений.

Виды метрик:

1) расстояние по Евклиду $L(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{\sum_{k=1}^{n} \left(x_{ik} - x_{jk} \right)^2}$

2) расстояние по Минковскому $L(\mathbf{x}_i, \mathbf{x}_j) = \sqrt[\lambda]{\sum_{k=1}^{n} \left(x_{ik} - x_{jk} \right)^{\lambda}}$, $\lambda>2$ – целое число

3) $L(\mathbf{x}_i, \mathbf{x}_j) = \sum_{k=1}^{n} \left| x_{ik} - x_{jk} \right|$

В этих метриках все компоненты векторов входят с одинаковыми единичными весами. В тех случаях, когда компоненты векторов, отличаются на порядки, например, одни компоненты векторов измеряются метрами, а другие - миллиметрами, то для более адекватного учета подобных компонент в метрику вводятся весовые коэффициенты η, учитывающие практическую ценность различных компонент вектора:

4) $L(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{\sum_{k=1}^{n} \eta_k \left(x_{ik} - x_{jk} \right)^2}$

5) $L(\mathbf{x}_i, \mathbf{x}_j) = \sqrt[\lambda]{\sum_{k=1}^{n} \eta_k \left(x_{ik} - x_{jk} \right)^{\lambda}}$, $\lambda>2$ – целое число

6) $L(\mathbf{x}_i, \mathbf{x}_j) = \sum_{k=1}^{n} \eta_k \left| x_{ik} - x_{jk} \right|$

Задание весовых коэффициентов в формулах, определяющих расстояния, требует наличия определенной априорной информации и не всегда может быть

сделано оптимальных образом. Поэтому особый интерес представляют расстояния, в которых заложена идея выравнивания весов слагаемых от различных компонент, если они существенно отличаются по своим абсолютным значениям:

7) расстояние по Камберру: $L(\mathbf{x}_i, \mathbf{x}_j) = \sum_{k=1}^{n} \left| \frac{x_{ik} - x_{jk}}{x_{ik} + x_{jk}} \right|$

При решении задач распознавания сигналов часто возникают ситуации, когда сигналы, отличающиеся только амплитудой или смещением по оси ординат, или небольшими нелинейными искажениями, необходимо относить к одному классу. В этом случае может использоваться

8) расстояние по Кендалу: $L(\mathbf{x}_i, \mathbf{x}_j) = 1 - \frac{2}{n(n-1)} \sum_{q=1}^{n-1} \sum_{\substack{k=2, \\ k>q}}^{n} \Delta_{qk}^{i} \Delta_{qk}^{j}$,

где $\Delta_{qk}^{i} = \begin{cases} 1, & \text{если } x_{iq} > x_{ik}, \\ -1, & \text{если } x_{iq} < x_{ik}, q < k, \\ 0, & \text{если } x_{iq} = x_{ik}, \end{cases}$

Расстояние по Кендалу это расстояние для оценки близости в некотором смысле двух функций, заданных в n точках. Для оценки близости двух функций F(x) и G(x) заданных векторами своих значений в n точках часто применяется и

9) расстояние по Чебышёву: $L\left(F\left(\mathbf{x}_i\right), G\left(\mathbf{x}_j\right)\right) = \max_i \left| F\left(\mathbf{x}_i\right) - G\left(\mathbf{x}_j\right) \right|$

Мера близости между двумя векторами в n-мерном векторном пространстве может быть задана в виде угла.

10) $L(\mathbf{x}_i, \mathbf{x}_j) = \arccos \frac{\left(\mathbf{x}_i, \mathbf{x}_j\right)}{|\mathbf{x}_i| \cdot |\mathbf{x}_i|}$

Некоторые системы распознавания используют непосредственно скалярное произведение в качестве меры сходства изображений:

11) $L(\mathbf{x}_i, \mathbf{x}_j) = \left(\mathbf{x}_i, \mathbf{x}_j\right)$

Работа с качественными характеристиками объектов

Образы могут характеризоваться не только количественными, но и качественными характеристиками (цвет, вкус, ощущение, запах). Эти признаки изначально не описаны количественно. Образы либо обладают какими-то качественными характеристиками, либо не обладают.

Однако, каждому качественному атрибуту присущи и определенные интервалы изменения количественных характеристик, за пределами которых меняется и качественный атрибут. Например, определенному цвету изображения соответствует конкретный диапазон длин электромагнитных волн, за пределами которого цвет изменится.

Существуют различные подходы к распознаванию изображений с качественными характеристиками. Рассмотрим один из них, основанный па двоичном кодировании наличия или отсутствия какого-либо качественного признака. Изображение некоторого образа с качественными характеристиками представляется в виде двоичного вектора

$x_k=(x_{kl}, x_{k2}, \ldots, x_{kj}, ..., x_{kn})$,

Если изображение x_k обладает j-м признаком, то $x_{kj}=1$, а если нет, то $x_{kj}=0$.

Например, четыре объекта - вишня, апельсин, яблоко, дыня. Каждый из них имеет три признака: цвет, наличие косточки или семечек:

	Вектор признаков	Желтый цвет	Оранжевый цвет	Красный цвет	Есть косточка	Есть семечки
Вишня	x_1	нет	нет	да	да	нет
Апельсин	x_2	нет	да	нет	нет	да
Яблоко	x_3	да	нет	да	нет	да
Дыня	x_4	да	нет	нет	нет	да

Числовые значения признаков для рассматриваемого примера после их двоичного кодирования:

	Вектор признаков	Желтый цвет	Оранжевый цвет	Красный цвет	Есть косточка	Есть семечки
Вишня	x_1	$x_{11}=0$	$x_{12}=0$	$x_{13}=1$	$x_{14}=1$	$x_{15}=0$
Апельсин	x_2	$x_{21}=0$	$x_{22}=1$	$x_{23}=0$	$x_{24}=0$	$x_{25}=1$

Яблоко	x_3	x_{31}=1	x_{32}=0	x_{33}=1	x_{34}=0	x_{35}=1
Дыня	x_4	x_{41}=1	x_{42}=0	x_{43}=0	x_{44}=0	x_{45}=1

В результате получается многомерное двоичное пространство признаков, где можно использовать различные расстояния, применяемые для распознавания объектов с количественными характеристиками (1-6).

При двоичном кодировании качественных признаков может применяться и расстояние по Хеммингу, равное числу несовпадающих компонент векторов.

Более тонкая классификация объектов с качественными признаками получается при введении для каждой пары объектов, переменных, характеризующих их общность или различие:

x_j	x_i	
	1	0
1	a	h
0	g	b

Переменная a предназначена для подсчета числа общих признаков объектов x_j, и x_i: $a = \sum_{k=1}^{n} \mathbf{x}_{jk}\mathbf{x}_{ik}$.

С помощью переменной b подсчитывается число случаев, когда объекты x_j, и x_i не обладают одним и тем же признаком: $b = \sum_{k=1}^{n} \left(1-\mathbf{x}_{jk}\right)\left(1-\mathbf{x}_{ik}\right)$

Переменные g и h предназначены соответственно для подсчета числа признаков, присутствующих у объекта x_i и отсутствующих у объекта x_j и, присутствующих у объекта x_j и отсутствующими у объекта x_i:

$$g = \sum_{k=1}^{n} \mathbf{x}_{ik}\left(1-\mathbf{x}_{jk}\right),\ h = \sum_{k=1}^{n} \left(1-\mathbf{x}_{ik}\right)\mathbf{x}_{jk}$$

Из анализа переменных a, b, g, h следует, что,

- чем больше сходство между объектами тем больше должна быть переменная a, т.е. мера близости объектов или функция сходства должна быть возрастающей функцией от a,

- функция сходства должна быть симметричной относительно переменных g и h.

- относительно переменной b однозначный вывод сделать не удастся, поскольку, с одной стороны, отсутствие одинаковых признаков у объектов может свидетельствовать об их сходстве, однако, с другой стороны, если у объектов общим является только отсутствие одинаковых признаков, то они не могут относиться к одному классу.

Наиболее часто применяются следующие функции сходства:

12) Рассела и Рао $S\left(\mathbf{x}_i,\mathbf{x}_j\right)=\dfrac{a}{a+b+g+h}=\dfrac{a}{n}$

13) Жокара и Нидмена: $S\left(\mathbf{x}_i,\mathbf{x}_j\right)=\dfrac{a}{n-b}$

14) Дайтса: $S\left(\mathbf{x}_i,\mathbf{x}_j\right)=\dfrac{a}{2a+g+h}$

15) Сокаля и Снифа: $S\left(\mathbf{x}_i,\mathbf{x}_j\right)=\dfrac{a}{a+2(g+h)}$

16) Сокаля и Мишнера: $S\left(\mathbf{x}_i,\mathbf{x}_j\right)=\dfrac{a+b}{n}$

17) Кульжинского: $S\left(\mathbf{x}_i,\mathbf{x}_j\right)=\dfrac{a}{g+h}$

18) Юла: $S\left(\mathbf{x}_i,\mathbf{x}_j\right)=\dfrac{ab-gh}{ab+gh}$

Распознавание по принадлежности к заданной области пространства

При этом способе распознавания всё пространство изображений V разбивается на непересекающиеся области V_1, V_2, ..., V_{k+1}, где V_1, V_2, ..., V_k - области, содержащие изображения только одного соответствующего класса; V_{k+1} - область, не содержащая изображений, относящихся к указанным классам.

Если области заданы в виде шаров с центрами в точках $\mathbf{x}_1^*,\mathbf{x}_2^*,...,\mathbf{x}_k^*$ и радиусами $R_1, R_2, ..., R_k$, то решающее правило приобретает вид:

$\mathbf{x}_i \in V_j$, если $\sqrt{\sum_{l=1}^{n}\left(x_{jl}^{*} - x_{il}\right)^2} \le R_j$

При использовании для распознавания угла между векторами непересекающиеся области V_j задаются в виде конусов, а решающее правило имеет вид: $\mathbf{x}_i \in V_j$, если $\arccos\frac{\left(\mathbf{x}_i, \mathbf{x}_j^{*}\right)}{\left|\mathbf{x}_i\right| \cdot \left|\mathbf{x}_j^{*}\right|} \le \varphi_{j_{\max}}$, где $\mathbf{x}_1^{*}, \mathbf{x}_2^{*}, ..., \mathbf{x}_k^{*}$ - эталонные образы для областей V_1, V_2, ..., V_k

Экспериментальная часть

Необходимо провести сравнительный анализ метода эталонных образов и метода ближайшего соседа. При этом основными характеристиками являются вероятность распознавания и скорость работы в зависимости от параметров обучающей выборки: ее размеров, наличия в ней выбросов, формы областей, занимаемых классами. Для этого необходимо выполнить следующую последовательность действий.

1.1 Выполнить реализацию методов эталонных образов и ближайшего соседа.

1.2. Сформировать выборку образов, которую разделить на обучающую и тестовую часть. Образы из тестовой выборки не участвуют в обучении, а используются затем для определения процента правильных ответов, даваемых тем или иным методом. Разделение на обучающую и тестовую выборку желательно производить случайным образом. Размер тестовой выборки должен быть достаточно большим (по крайней мере, несколько десятков элементов).

1.3. Варьируя обучающую выборку, определить проценты правильного распознавания каждого из методов. Обучающую выборку следует изменять следующим образом: добавлять и исключать из нее элементы, чтобы менялся размер выборки, вносить в обучающую выборку ошибки (для нескольких элементов указывать неправильный класс).

1.4. Для определения скорости работы каждого из методов следует варьировать размер обучающей выборки. Для формирования обучающих выборок

больших размеров допустимо многократно дублировать содержимое исходной обучающей выборки. Для определения времени классификации нового образа следует многократно (в цикле) вызывать классифицирующую процедуру и оценивать время для такого многократного вызова, после чего делить полученное общее время на число вызовов.

1.5. Проанализировать полученные результаты. Определить, как влияют ошибки в обучающей выборке на каждый из методов, при каких размерах обучающей выборки у какого из методов больше процент правильного распознавания (и при какой форме областей, занимаемых классами), как влияет размер обучающей выборки на время классификации нового образа в каждом из методов. Сделать выводы по работе.

Для анализа метода обобщенных решающих функций при задании дополнительных признаков вручную. При этом основной характеристикой является вероятность распознавания в зависимости от параметров обучающей выборки: ее размеров, наличия в ней выбросов, формы областей, занимаемых классами, и в зависимости от привлекаемых дополнительных признаков. Для этого необходимо выполнить следующую последовательность действий.

2.1. Выполнить реализацию метода обобщенных решающих функций.

2.2. Сформировать выборку образов, которую разделить на обучающую и тестовую часть. Образы из тестовой выборки не участвуют в обучении, а используются затем для определения процента правильных ответов, даваемых при тех или иных дополнительных признаках. Разделение на обучающую и тестовую выборку желательно производить случайным образом. Размер тестовой выборки должен быть достаточно большим (по крайней мере, несколько десятков элементов).

2.3. Варьируя обучающую выборку, определить проценты правильного распознавания для различных дополнительных признаков (рекомендуется использовать линейные и квадратичные решающие функции). Обучающую выборку следует изменять следующим образом: добавлять и исключать из нее элементы, чтобы менялся размер выборки, вносить в обучающую выборку ошибки (для нескольких элементов указывать неправильный класс).

2.4. Для нелинейно разделимых классов образов осуществить перебор дополнительных признаков и найти минимальное количество признаков, при которых корректное разделение находится методом обобщенных решающих функций.

2.5. Проанализировать полученные результаты. Определить, как влияют ошибки в обучающей выборке на метод обобщенных решающих функций, при каких размерах обучающей выборки и при каких дополнительных признаках больше процент правильного распознавания (и при какой форме областей, занимаемых классами). Сделать выводы по работе.

3.2 Лабораторная работа №7 «Методы кластеризации»

Цель работы **–** ознакомиться с методами анализа пространства признаков в рамках задач кластеризации и выбора признаков, а также освоить их применение в различных условиях, определяемых характером распределения образов обучающей выборки.

3.2.1 Задания к лабораторной работе №7

1. Реализовать метод k внутригрупповых средних. Путем варьирования взаимного расположения и формы кластеров. образуемых образами обучающей выборки, определить ограничения метода кластеризации.

2. Реализовать метод оценивания плотности вероятностей на основе смесей. Путем варьирования компонент смеси и их количества, определить ограничения метода на основе смесей.

3.2.2 Методические указания к лабораторной работе №7

Экспериментальная часть

Для анализа метода k внутригрупповых средних, используемого для распознавания без учителя (кластеризации), требуется установить влияние формы и взаимного расположения кластеров на возможность их обнаружения данным методом, а также устойчивость результатов метода при выборе различных

начальных центров кластеров. Для этого необходимо выполнить следующую последовательность действий.

1.1. Выполнить реализацию метода k внутригрупповых средних.

1.2. Сформировать различные обучающие выборки образов, варьирующиеся по форме кластеров (круглые, сильно вытянутые, неправильной формы типа «Г»), их относительным размерам (одинаковые или разные размеры кластеров) и близости расположения кластеров.

1.3. Для нескольких обучающих выборок определить различия в конечных результатах при использовании разных способов задания начальных центров кластеров: начальные центры формируются из близко расположенных образов, случайно выбранных образов, наиболее удаленно расположенных образов. Оценить количество итераций, требуемых методу для схождения, при разных способах задания начальных центров.

1.4. Установить различия в результатах кластеризации для нескольких обучающих выборов в зависимости от того, используется ли евклидово расстояние, или оно нормируется на размеры кластеров.

1.5. Определить характер формируемых кластеров в случаях, когда заданное значение k отличается (как в большую, так и в меньшую сторону) от действительного числа кластеров в обучающей выборке.

1.6. Проанализировать полученные результаты. Определить ограничения метода k средних. Сделать выводы по работе.

Для исследования метода оценивания плотности вероятностей на основе смесей, используемого для оценки неизвестной плотности вероятностей, требуется установить влияние типа входящих в смесь распределений и их количества на точность оценки. Для этого необходимо выполнить следующую последовательность действий.

2.1. Сгенерировать обучающую выборку на основе заданной функции распределения плотности вероятностей.

2.2. Реализовать метод, основанный на представлении искомого распределения в виде конечных смесей, с целью оценки плотности вероятности в произвольной точке.

2.3. Оценить при каких входящих типах распределений в смесь, обнаружение искомого распределения будет наилучшим.

2.4. Установить, как количество входящих в смесь компонентов влияет на обнаружение искомого распределения, и на быстродействие программы

2.5. Проанализировать полученные результаты. Определить ограничения метода конечных смесей. Сделать выводы по работе.

4 ПРОЕКТИРОВАНИЕ НЕЙРОННЫХ СЕТЕЙ

4.1 Лабораторная работа №8 «Нейрон Хебба и правило его обучения»

Цель работы – изучить особенности нейрона Хебба и правило его обучения.

4.1.1 Задания к лабораторной работе №8

1. Разработайте структуру сети Хебба, которая способна распознавать четыре различные буквы вашего имени или фамилии.

2. Обучите нейронную сеть Хебба распознаванию четырех заданных букв вашего имени или фамилии.

3. Продемонстрируйте работоспособность сети при предъявлении обучающих изображений и изображений, содержащих ошибки.

4. Оформите отчет по лабораторной работе.

4.1.2 Методические указания к лабораторной работе №8

Теоретическая часть

При моделировании нейронных сетей в качестве искусственных нейронов обычно используется простой процессорный элемент. На его входы поступает вектор $X = (x_1, \ldots, x_n)$ входных сигналов, являющихся выходными сигналами других нейронов, а также единичный сигнал смещения. Все входные сигналы, включая и

сигнал смещения, умножаются на весовые коэффициенты своих связей и суммируются:

$$S = \sum_{i=1} x_i w_i + w_0 ,$$

где S – суммарный входной сигнал, который поступает на вход блока, реализующего функцию f активации нейрона;

w_i – весовые коэффициенты связей входных сигналов x_i, ..., x_n ,

w_0 – весовой коэффициент связи сигнала смещения.

Приведенная модель искусственного нейрона игнорируют многие известные свойства биологического прототипа. Например, она не учитывает временные задержки нейронов, эффекты частотной модуляции, локального возбуждения и связанные с ними явления подпороговой временной и пространственной суммации, когда клетка возбуждается не одновременно пришедшими импульсами, а последовательностями возбуждающих сигналов, поступающих через короткие промежутки времени. Не учитываются также периоды абсолютной рефрактерности, во время которых нервные клетки не могут быть возбуждены, т.е. как бы обладают бесконечно большим порогом возбуждения, который затем за несколько миллисекунд после прохождения сигнала снижается до нормального уровня. Этот список отличий, которые многие биологи считают решающими, легко продолжить, однако искусственные нейронные сети все же обнаруживают ряд интересных свойств, характерных для биологических прототипов.

Искусственные нейронные сети, предназначенные для решения разнообразных конкретных задач, могут содержать от нескольких нейронов до тысяч и даже миллионов элементов. Однако уже отдельный нейрон с биполярной функцией активации (на выходе имеется сигнал "1" или "–1") может быть использован для решения простых задач распознавания и классификации изображений.

Если выходной сигнал у нейрона принимает только два значения, то нейрон можно использовать для классификации предъявляемых изображений на два класса.

Пусть имеется множество M изображений, для которых известна корректная классификация на два класса $X^1 = \{X^{11}, X^{12}, ..., X^{1q}\}$, $X^2 = \{X^{21}, X^{22}, ..., X^{2p}\}$, $X^1 \cup X^2 = M$, $X^1 \cap X^2 = 0$, и пусть первому классу X^1 соответствует выходной сигнал $y = 1$, а классу X^2 – сигнал $y = -1$. Если, например, предъявлено некоторое изображение $X^\alpha = \left(X_1^\alpha, ..., X_n^\alpha\right), X^\alpha \in M$ и его взвешенная сумма входных сигналов превышает нулевое значение:

$$S = \sum_{i=1} x_i^\alpha w_i + w_0 > 0$$

то выходной сигнал $y = 1$ и, следовательно, входное изображение X^α принадлежит классу X^1. Если $S < 0$, то y = –1 и предъявленное изображение принадлежит второму классу.

Возможно использование отдельного нейрона и для выделения из множества классов $M = \{X^1 = \{X^1, ..., X^{1\kappa}\}, ..., X^i = \{X^i, ..., X^{iq}\}, ..., X^p = \{X^{p1}, ..., X^{pm}\}\}$ изображений единственного класса X^i. В этом случае полагают, что один из двух возможных выходных сигналов нейрона (например, 1) соответствует классу X^i а второй – всем остальным классам. Поэтому, если входное изображение X^α приводит к появлению сигнала y = 1, то $X^\alpha \in X^i$, если $y = -1$, (или $y = 0$, если используется бинарное кодирование) то это означает, что предъявленное изображение не принадлежит выделяемому классу.

Система распознавания на основе единственного нейрона делит все пространство возможных решений на две области с помощью гиперплоскости

$$x_1 w_1 + x_2 w_2 + ... + x_n w_n + w_0 = 0$$

Для двумерных входных векторов границей между двумя классами изображений является прямая линия: входные вектора, расположенные выше этой прямой, принадлежат к одному классу, а ниже – к другому.

Для адаптации, настройки или обучения весов связей нейрона может использоваться несколько методов. Рассмотрим один из них, получивший название «правило Хебба». Хебб, исследуя механизмы функционирования центральной нервной системы, предположил, что обучение происходит путем

усиления связей между нейронами, активность которых совпадает по времени. Хотя в биологических системах это предположение выполняется далеко не всегда и не исчерпывает всех видов обучения, однако при обучении однослойных нейросетей с биполярными сигналами оно весьма эффективно.

В соответствии с правилом Хебба, если предъявленному биполярному изображению $X=(x_1, \ldots, x_n)$ соответствует неправильный выходной сигнал, то веса w_i $(i=\overline{1,n})$ связей нейрона адаптируются по формуле

$$w_i(t+1)=w_i(t)+x_i y,\ (i=\overline{0,n})$$

где $w_i(t)$, $w_i(t+1)$ соответственно вес *i-й* связи нейрона до и после адаптации;

x_i $(i=\overline{1,n})$ –компоненты входного изображения;

$x_0 \equiv 1$ – сигнал смещения;

y – выходной сигнал нейрона.

В более полной и строгой форме алгоритм настройки весов связей нейрона с использованием правила Хебба выглядит следующим образом:

Шаг 1. Задается множество $M = \{(X^1, t^l),.., (X^m, t^m)\}$ состоящее из пар {входное изображение $X^k=(x_1^k,...,x_n^k)$, необходимый выходной сигнал нейрона t^k, $(k=\overline{1,m})$}. Инициируются веса связей нейрона:

$w_i = 0,\ i=\overline{0,n}$

Шаг 2. Для каждой пары (X^k, t^k), $k=\overline{0,m}$ пока не соблюдаются условия останова, выполняются шаги 3-5.

Шаг 3. Инициируется множество входов нейрона: $x_0=1,\ x_i=x_i^k,\ i=\overline{1,n}$

Шаг 4. Инициируется выходной сигнал нейрона: $y = t^k$.

Шаг 5. Корректируются веса связей нейрона по правилу:

$$w_i(new)=w_i(old)+x_i y,\ i=\overline{0,n}$$

Шаг 6. Проверка условий останова. Для каждого входного изображения X^k рассчитывается соответствующий ему выходной сигнал y^k:

$$y^k = \begin{cases} 1, \textit{если } S^k > 0 \\ -1, \textit{если } S^k \le 0, \ k = \overline{1,m} \end{cases}$$

где

$$S^k = \sum_{i=1}^{n} x_i^k w_i + w_0$$

Если вектор $(y^1, ..., y^m)$ рассчитанных выходных сигналов равен вектору $(t^1, ..., t^m)$ заданных сигналов нейрона, т. е. каждому входному изображению соответствует заданный выходной сигнал, то вычисления прекращаются (переход к шагу 7), если же $(y^1, ..., y^m) \neq (t^1, ..., t^m)$, то переход к шагу 2 алгоритма.

Шаг 7. Останов.

Однослойная нейронная сеть с двоичными нейронами может быть обучена с помощью алгоритма на основе правила Хебба. В этом случае она называется сетью Хебба. Использование других алгоритмов обучения этой же сети приводит и к изменению названия нейронной сети. Использование в названии сетей их алгоритмов обучения характерно для теории нейронных сетей. Для биполярного представления сигналов возможно обучение нейросети с помощью следующего алгоритма:

Шаг 1. Задается множество $M = \{(X^1, t^l), ..., (X^m, t^m)\}$ состоящее из пар {входное изображение $X^k = (x^k, ..., x_n^k)$, необходимый выходной сигнал нейрона t^k, (k =1, m)}. Инициируются веса связей нейрона: $w_{ji} = 0, j = \overline{1,n}, \ i = \overline{1,m}$

Шаг 2. Каждая пара (X^k, t^k), проверяется на правильность реакции нейронной сети на входное изображение. Если полученный выходной вектор сети ($y^k, ..., y^k$), отличается от заданного $t^1 = (t_1^k, ..., t_n^k)$, то выполняют шаги 3 -5.

Шаг 3. Инициируется множество входов нейронов: $x_0 = 1, x_j = x_j^k, j = \overline{1,n}$

Шаг 4. Инициируются выходные сигналы нейронов: $y_i = t_i^k, i = \overline{0,m}$

Шаг 5. Корректируются веса связей нейронов по правилу:

$$w_{ji}(new) = w_{ji}(old) + x_i y, \ j = \overline{0,n}, \ \ i = \overline{0,m}$$

Шаг 6. Проверяются условия останова, т. е. правильности функционирования сети при предъявлении каждого входного изображения. Если условия не выполняются, то переход к шагу 2 алгоритма, иначе – прекращение вычислений (переход к шагу 7).

Шаг 7. Останов.

Практическая часть

Пусть требуется обучить биполярный нейрон распознаванию изображений X^1 и X^2, приведенных на рис. 1.

Рисунок 1 – Входные изображения

При этом потребуем, чтобы изображению X^1 соответствовал выходной сигнал нейрона "+1", а изображению X^2 сигнал "–1".

Применение алгоритма Хебба дает следующие результаты:

Шаг 1. Задается множество

$M = \{(X^1 = (1, -1, 1, 1, 1, 1, -1, -1, 1), 1),$

$(X^2 = (1, 1, 1, 1, -1, 1, 1, -1, 1), -1)\};$

инициируются веса связей нейрона: $w_i = 0,\ i = \overline{0,9}$

Шаг 2. Для каждой из двух пар (X^1, 1), (X^2, –1) выполняются шаги 3 – 5 алгоритма.

Шаг 3. Инициируется множество входов нейрона для изображения первой пары: $x_0 = 1,\ x_i = x_i^1,\ i = \overline{0,9}$.

Шаг 4. Инициируется выходной сигнал нейрона для изображения первой пары: $y = t^1 = 1$.

Шаг 5. Корректируются веса связей нейрона по правилу Хебба

$w_i = w_i + x_i^1 y\ \ (i = \overline{0,n})$:

$$w_0 = w_0 + x_0 y = 0 + 1 \cdot 1 = 1;$$
$$w_1 = w_1 + x_1^1 y = 0 + 1 \cdot 1 = 1;$$
$$w_1 = w_3 = w_4 = w_5 = w_6 = w_9 = 1;$$
$$w_2 = w_2 + x_2^1 y = 0 + (-1) \cdot 1 = -1;$$
$$w_2 = w_7 = w_8 = -1.$$

Шаг 3. Инициируется множество входов нейрона для изображения X^2 второй пары:

$$x_0 = 1,\ x_i = x_i^2,\ i = \overline{0,9}.$$

Шаг 4. Инициируется выходной сигнал нейрона для изображения второй пары (X^2, t^2):

$$y = t^2 = -1.$$

Шаг 5. Корректируются веса связей нейрона:

$$w_0 = w_0 + x_0 y = 1 + 1 \cdot (-1) = 0;$$
$$w_1 = w_1 + x_1^2 y = 1 + 1 \cdot (-1) = 0;$$
$$w_1 = w_3 = w_4 = w_6 = w_9 = 0;$$
$$w_2 = w_2 + x_2^2 y = -1 + 1 \cdot (-1) = -2;$$
$$w_2 = w_7 = -2;$$
$$w_5 = w_5 + x_5^2 y = 1 + (-1) \cdot (-1) = 2;$$
$$w_8 = w_8 + x_8^2 y = -1 + (-1) \cdot (-1) = 0.$$

Шаг 6. Проверяются условия останова.

Рассчитываются входные и выходной сигналы нейрона при предъявлении изображения X^1:

$$S^1 = \sum_{i=1}^{9} x_i^1 w_i + w_0 = 1 \cdot 0 + (-1) \cdot (-2) + 1 \cdot 0 + 1 \cdot 0 + 1 \cdot 2 + 1 \cdot 0 + (-1) \cdot (-2) + (-1) \cdot 0 + 1 \cdot 0 + 0 = 6$$

$$y^1 = 1,\ так\ как\ S^1 > 0.$$

Рассчитываются входной и выходной сигналы нейрона при предъявлении изображения X^2:

$$S^2 = \sum_{i=1}^{9} x_i^2 w_i + w_0 = 1 \cdot 0 + 1 \cdot (-2) + 1 \cdot 0 + 1 \cdot 0 + (-1) \cdot 2 + 1 \cdot 0 + 1 \cdot (-2) + (-1) \cdot 0 + 1 \cdot 0 + 0 = -6$$

$y^2 = -1$, *так как* $S^2 < 0$.

Поскольку вектор (y^1, y^2) = (1, –1) равен вектору (t^1, t^2), то вычисления прекращаются, так как цель достигнута – нейрон правильно распознает заданные изображения.

Шаг 7. Останов.

4.2 Лабораторная работа №9 «Элементарный персептрон Розенблатта»

Цель работы – приобретение и закрепление навыков работы с перцептроном Розенблатта.

4.2.1 Задания к лабораторной работе №9

1. Разработайте структуру однослойного перцептрона, способного распознавать первые буквы Вашего имени и Вашей фамилии.

2. Выполнить программную реализацию персептрона и алгоритма его обучения распознаванию.

4.2.2 Методические указания к лабораторной работе №9

Теоретическая часть

Персептрон Розенблатта является исторически первой обучаемой нейронной сетью. Существует несколько версий персептрона. Рассмотрим классический персептрон – сеть с пороговыми нейронами и входными сигналами, равными нулю или единице.

Элементарный перцептрон состоит из элементов трёх типов: S-элементов, A-элементов и *одного* R-элемента (рис.2).

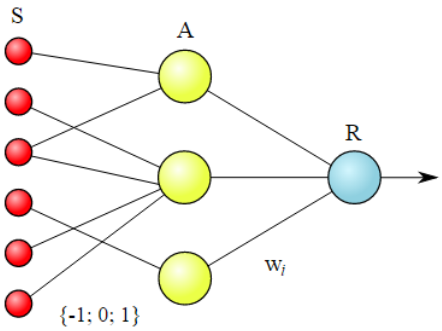


Рисунок 2 – Архитектура перцептрона

S-элементы – это слой сенсоров или рецепторов. В физическом воплощении - это, например, светочувствительные клетки сетчатки глаза. Каждый рецептор может находиться в одном из двух состояний – *покоя* или *возбуждения*, и только в *последнем* случае он передаёт единичный сигнал в следующий слой, ассоциативным элементам.

A-элементы называются ассоциативными, потому что каждому такому элементу соответствует целый набор (*ассоциация*) S-элементов. A-элемент *активизируется,* как только количество сигналов от S-элементов на его входе превысило некоторую величину θ.

Сигналы от возбудившихся A-элементов, в свою очередь, передаются в *сумматор R,* причём сигнал от *i*-го ассоциативного элемента передаётся с коэффициентом ω_i.

Этот коэффициент называется *весом* A–R связи.

R-элемент подсчитывает сумму значений входных сигналов, помноженных на веса (линейная форма). R-элемент и элементарный перцептрон, выдаёт «1», если линейная форма превышает порог θ, иначе на выходе - «−1».

Математически, функцию, реализуемую R-элементом, можно записать так:

$$f(x) = sign\left(\sum_{i=1}^{n} w_i x_i - \theta\right)$$

Обучение элементарного перцептрона состоит в изменении весовых коэффициентов связей A–R.

Веса связей S–A (которые могут принимать значения {−1; 0; +1}) и значения порогов A-элементов выбираются случайным образом в самом начале и затем не изменяются.

Практическая часть

На рис. 3 изображена схема однослойного персептрона, предназначенного для распознавания цифр.

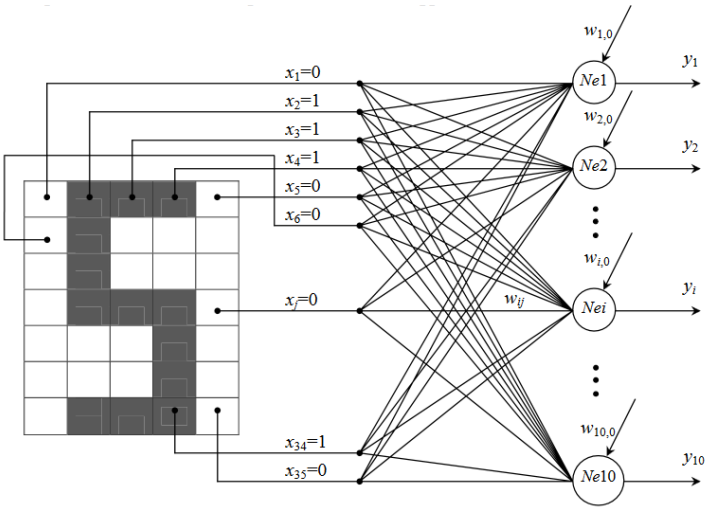


Рисунок 3 - Однослойный персептрон для распознавания цифр

Данная нейронная сеть имеет 10 нейронов, организованных таким образом, чтобы каждой цифре соответствовал свой нейрон. Выход первого нейрона y_1 должен быть равен единице, если персептрону предъявляется цифра «1» и нулю для выходов всех остальных нейронов. Выход y_2 должен быть равен единице, если персептрону показывается цифра «2», при этом остальные выходы нейронов должны быть равны нулю. И так далее до цифры «0». Алгоритм обучения однослойного персептрона, представленного на рис. 3, с помощью дельта-правила выглядит следующим образом:

Шаг 1. Инициализация. Всем весам персептрона w_{ij} и $w_{i,0}$ $(i=\overline{1,10},\ j=\overline{1,35})$ присваиваются небольшие случайные значения из диапазона [-0,1; +0,1].

Шаг 2. На вход персептрона подается очередной входной вектор $X[t]=\{x_1[t], x_2[t],..., x_{35}[t]\}$,

где t – номер итерации.

Каждый из 10 нейронов выполняет взвешенное суммирование входных сигналов $\sigma_i[t] = \sum_{j=1}^{35} w_{ij}[t] + w_{i,0}[t]$

и вырабатывает выходной сигнал

$$y_i[t] = \begin{cases} 1, \text{ если } \sigma_i[t] \geq 0; \\ 0, \text{ если } \sigma_i[t] < 0. \end{cases}$$

Шаг 3. Для каждого нейрона определяется ошибка $\beta_i[t] = (d_i[t] - y_i[t])$, где $d_i[t]$ – требуемое значение выхода *i*-го нейрона, а $y_i[t]$ – полученное на шаге 2 значение *i*-го выхода.

Шаг 4. Дельта-правило. Производится модификация весовых коэффициентов персептрона в соответствии с формулами:

$$w_{ij}[t+1] = w_{ij}[t] + \Delta w_{ij}[t];$$
$$\Delta w_{ij}[t] = \alpha \cdot \beta_i[t] \cdot x_j[t];$$
$$w_{i,0}[t+1] = w_{i,0}[t] + \Delta w_{i,0}[t];$$
$$\Delta w_{i,0}[t] = \alpha \cdot \beta_i[t],$$

где α – коэффициент скорости обучения с помощью которого можно управлять величиной коррекции весов Δ ($0<\alpha\leq 1$).

РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА

Основная учебная литература

1. Боровская, Е.В. Основы искусственного интеллекта [Электронный ресурс] : учебное пособие / Е.В. Боровская, Н.А. Давыдова. — Электрон. дан. — Москва : Издательство "Лаборатория знаний", 2016. — 130 с. — Режим доступа: https://e.lanbook.com/book/84083

2. Системы искусственного интеллекта. Часть 1 [Электронный ресурс]: Учебное пособие / Сергеев Н.Е. - Таганрог: Южный федеральный университет, 2016. - 118 с. Режим доступа: http://znanium.com/catalog/product/991954

Дополнительная учебная литература

1. Методы искусственного интеллекта [Электронный ресурс]: Монография/ Осипов Г.С. - М.:Физматлит, 2011. - 296 с. Режим доступа: http://znanium.com/catalog/product/544787

2. Джонс, М.Т. Программирование искусственного интеллекта в приложениях [Электронный ресурс] / М.Т. Джонс. — Электрон. дан. — Москва : ДМК Пресс, 2011. — 312 с. — Режим доступа: https://e.lanbook.com/book/1244

3. Матвеев, М.Г. Модели и методы искусственного интеллекта. Применение в экономике [Электронный ресурс] : учебное пособие / М.Г. Матвеев, А.С. Свиридов, Н.А. Алейникова. — Электрон. дан. — Москва : Финансы и статистика, 2008. — 448 с. — Режим доступа: https://e.lanbook.com/book/5343

4. Рутковская, Д. Нейронные сети, генетические алгоритмы и нечеткие системы: Пер. с польск. И.Д.Рудинского [Электронный ресурс] : учебное пособие / Д. Рутковская, М. Пилиньский, Л. Рутковский. — Электрон. дан. — Москва : Горячая линия-Телеком, 2013. — 384 с. — Режим доступа: https://e.lanbook.com/book/11843

5. Васильев, В.Н. Оптические технологии искусственного интеллекта. В 2-х т. Том 1. Основы оптических информационных технологий и искусственных нейронных сетей [Электронный ресурс] : учебное пособие / В.Н. Васильев, А.В.

Павлов. — Электрон. дан. — Санкт-Петербург : НИУ ИТМО, 2017. — 80 с. — Режим доступа: https://e.lanbook.com/book/110516

6. Галушкин, А.И. Нейронные сети: основы теории [Электронный ресурс] / А.И. Галушкин. — Электрон. дан. — Москва : Горячая линия-Телеком, 2017. — 496 с. — Режим доступа: https://e.lanbook.com/book/111043