1. Исследование НС с пороговой функцией активации (персептронов)

Цель работы: изучение архитектуры персептрона и специальных функций для создания персептрона, настройки его весов и смещений и адаптации, ознакомление с демонстрационными примерами, а также приобретение навыков построения и обучения персептронов для различных областей применения.

Теоретические сведения

Персептрон – это однослойная нейронная сеть с **S** нейронами и **R** входами, каждый из которых может состоять из нескольких элементов. Передаточной функцией каждого нейрона является ступенчатая функция типа **hardlim** или **hardlims**. Помимо основных входов, нейроны персептрона имеют вход для постоянного смещения, равного единице. Элементы входов и смещения взвешиваются с помощью функции скалярного произведения **dotprod** и суммируются с помощью функции накопления **netsum.**

Создание персептрона

Создание персептрона производится следующей функцией:

net = newp(P, T, tf, lf),

где **net** – объект класса **network**;

P – массив размера **RxQ1,** содержащий Q1 репрезентативных входных векторов размерности R;

T - массив размера S**xQ2,** содержащий Q2 репрезентативных выходных векторов размерности S;

tf – передаточная функция из списка **{ hardlim, hardlims },** причем по умолчанию задается **hardlim;**

lf – обучающая функция из списка **{learnp, learnpn},** причем по умолчанию – **learnp**.

При создании персептрона, матрица весов и вектор смещений инициализируются нулями с помощью функций **initzero**.

Количество входов и выходов автоматически определяется по размерности матриц P и T.

Обучение персептрона

Персептрон обучается по дельта-правилу Розенблатта, которое является инкрементальным алгоритмом обучения (изменение весовых коэффициентов производится после каждой подачи образа). Только в этом случае обеспечивается сходимость алгоритма для линейно-разделимых классов. В NN Toolbox инкрементальный алгоритм обучения может быть реализован как с помощью функции adapt, так и с помощью функции train. Далее будут рассмотрены оба варианта, но следует учитывать, что с течением времени возможности функции adapt разработчиками уменьшаются, и более предпочтительным (дальновидным) является использование функции train.

Обучение персептрона с помощью функции adapt

Функция adapt предназначена только для инкрементального обучения. Она вызывает функцию, определенную в net.adaptFcn. Единственная из таких функций в последней версии Matlab (R2011a) – adaptwb. Функция adaptwb в свою очередь использует функции обучения (learn) весовых коэффициентов и смещений.

Отметим, что функции adaptwb невозможно передать никакие параметры, такие как скорость обучения или количество проходов. В предыдущих версиях Matlab такая возможность существовала (через свойство net.adaptParam).

Обучение персептрона с помощью функции train

Функция train предназначена как для инкрементального, так и для пакетного обучения. Она вызывает функцию, определенную в net.trainFcn. Для инкрементального обучения возможны следующие варианты функций net.trainFcn:

- trainc, trains – последовательное циклическое инкрементальное обучение
- trainr - последовательное инкрементальное обучение со случайным порядком

Функциям train можно передать различные параметры с помощью структуры net.trainParam.

Параметр	Описание	По умолчанию
net.trainParam.epochs	Макс количество эпох обучения	10
net.trainParam.goal	Цель обучения (ошибка)	0
net.trainParam.showCommandLine	Выводить ли в командную строку информацию по обучению	1
net.trainParam.showWindow	Показывать графически ход обучения	1
net.trainParam.lr	Скорость обучения	0.01
net.trainParam.max_fail	Максимальное число эпох для завершения обучения (early stopping)	5
net.trainParam.min_grad	Минимальный градиент ошибки – по его достижению окончание обучения	1e-10
net.trainParam.show	Через сколько эпох обновлять вывод информации	25
net.trainParam.time	Максимальное время обучения	inf

Эпоха обучения означает обучение по всей обучающей выборке.

Цель обучения – значение ошибки, при котором обучение завершается. Для расчета этой ошибки используется свойство net. performFcn:

- mae – средняя абсолютная ошибка
- mse – средняя относительная ошибка
- sse – суммарная квадратичная ошибка

Необходимо учитывать, что эти функции ошибки используются в основном для оценки качества аппроксимации функций. Для оценки качества классификации эти функции подходят не всегда. Также следует учитывать то, что окончательное значение ошибки является усредненным по всей выборке и выходам.

Скорость обучения является множителем функции приращения весового коэффициента. Очень маленькая скорость "грозит" медленной сходимостью, очень большая – сильными колебаниями и как следствие также долгой сходимостью.

Задание локальных функций обучения

При использовании функций adapt или train для обучения персептрона необходимо задать т.н. локальные функции обучения – для весовых коэффициентов и смещений. Это осуществляется путем задания следующих свойств объекта network:

- net.inputWeights{i,j}.learnFcn – связи от входов к слоям сети
- net.layerWeights{i,j}.learnFcn – связи между слоями
- net.biases{i}.learnFcn – смещения.

Зам. При вызове функции newp перечисленные выше свойства автоматически устанавливаются либо в значение 'learnp' по умолчанию, либо в значение, которое задается пользователем.

Зам. При использовании локальных функций обучения обучающую выборку необходимо задавать в формате cell array, т.е. внутри фигурных скобок.

Для персептрона возможно задание двух типов локальных функций обучения - learnp и learnpn. Функция learnp осуществляет обучение по классическому дельта-правилу Розенблатта:

dw = e*p',

где e – ошибка между реальным и желаемым выходными сигналами, p' – транспонированный вектор входа.

Функция learnpn реализует дельта правило с нормализацией входного сигнала:

dw = e*p' / sqrt(1 + p(1)^2 + p(2)^2) + ... + p(R)^2),

Это позволяет сделать время обучения нечувствительным к большим или малым выбросам векторов входа

Примеры

Пример 1

Создать персептрон с одним нейроном и одноэлементным входом, диапазон значений которого от **0** до **1**, и проанализировать значения параметров его вычислительной модели, выполнив следующие действия:

1. Создать и инициализировать персептрон:

```
net = newp([0 1], 1)      % –создаем персептрон;
net.inputWeights{1,1} % – веса входа объекта;
net.biases{1}         % – смещение нейрона;
net.IW{1,1}           % – значение весов;
net.b{1}    % – значение смещения;
net.IW{1,1}=[3]     % – задание веса;
net.b{1}=[4]          % – задание смещения;
net = init(net);        % – инициализация нулями;
net.IW{1,1}           % – нулевое значение веса;
net.b{1}    % –нулевое значение смещения;
net.inputWeights{1,1}.initFcn='rands';
net.biases{1}.initFcn='rands';
net = init(net);        % – инициализация случайными значениями;
net.IW{1,1}, net.b{1}    % – новые значения;
p = {[0] [1] [0.5]};        % – последовательность входов;
a = sim(net, p)       % – моделирование сети;
```

2. Сгенерировать схему командой gensim, проанализировать ее и привести в отчет.

```
gensim(net);
```

Пример 2

Создать персептрон с одним нейроном и одним двухэлементным вектором входа, значения элементов которого изменяются в диапазоне от **–2** до **2**, настроить веса и смещение для реализации разделяющей линии

$$-p_1 + p_2 + 1 = 0,$$

а затем с помощью моделирования определить классы значений входного вектора, выполнив следующие действия:

1. Создаем персептрон.

```
net = newp([-2 2; -2 2], 1).
```

2. Производим инициализацию:

```
net.IW{1,1} = [-1 1] ; net.b{1} = [1].
```

3. Выполняем проверку персептрона:

```
p = [1;1];
a = sim(net, p)       % a = 1;
p = [1;-1];
a = sim(net, p)       % a = 0.
```

4. Определяем классы значений для векторов:

```
p = {[-2;-2] [-2;-1] [-2;0] [-2;1] [-2;2]…
[-1;-2] [-1;-1] [-1;0] [-1;1] [-1;2]…
[0;-2] [0;-1] [0;0] [0;1] [0;2]…
[1;-2] [1;-1] [1;0] [1;1] [1;2]…
[2;-2] [2;-1] [2;0] [2;1] [2;2];
a = sim(net, p)        % [0]-0-й класс; [1]-1-й класс.
```

Практические задания

Задание 1 (dataset 1) Классификация, 2 входа, 2 линейно-неразделимых класса, 9 примеров

1. Сформируйте набор входных и желаемых выходных образов (“0” – класс 1, “X” – класс 2. Постройте график и нанесите на него эти точки.

2. Для заданного варианта сформируйте собственный вариант решения задачи без ошибок и реализуйте его аналитически вначале математически с помощью формул, а затем с помощью 2-х или 3-х слойного персептрона (для этого вначале с помощью кусочно-линейной аппроксимации задайте функции, определяющие принадлежность к классу 1 и 2).

3. Сгенерируйте схему командой gensim (net), проанализируйте ее и приведите в отчет.

4. Проверьте полученные функции в заданных точках, а также в близлежащих точках.

5. Попытайтесь решить задачу распознавания путем обучения 1-слойного персептрона на множестве входных примеров. Проанализируйте полученный результат.

Задание 2 (dataset 2) – ЛФ, несколько входов, 1 выход

1. Запишите выражение для вашей ЛФ в форме СДНФ.

2. Реализуйте полученную СДНФ в форме 2-слойного персептрона (net) в Matlab.

3. Сгенерируйте схему командой gensim (net), проанализируйте ее и приведите в отчет.

4. Проверьте правильность работы полученной НС по таблице истинности.

5. Попробуйте обучить однослойный персептрон на Вашей функции, используя в качестве обучающей выборки фрагменты таблицы истинности. Проанализируйте результаты и посчитайте среднюю ошибку.

Задание 3 (dataset 3) – классификация, 2 классса

1. Запишите аналитическое выражение для функции, реализующей Ваше разбиение плоскости на 2 класса. Постройте график функции и разбиение плоскости на классы.

2. Реализуйте полученную функцию в форме 2-х или 3-х слойного персептрона net в Matlab.

3. Сгенерируйте схему командой gensim (net), проанализируйте ее и приведите в отчет.

4. Проверьте правильность работы полученной НС путем построения разбиения плоскости на классы, которое реализует персептрон.

5. Попытайтесь решить задачу распознавания путем обучения 1-слойного персептрона на множестве входных примеров. Для этого вначале сформируйте обучающую выборку необходимого объема.

6. После обучения посчитайте среднюю ошибку, проанализируйте результаты – какую функцию реализует обученная сеть.

Задание 4 (dataset 4) – классификация, С класссов

1. Запишите аналитическое выражение для функции, реализующей Ваше разбиение плоскости на m классов. Постройте график с разбиением плоскости на m классов.

2. Реализуйте полученную функцию в форме 2-х или 3-х слойного персептрона net в Matlab.

3. Сгенерируйте схему командой gensim (net), проанализируйте ее и приведите в отчет.

4. Проверьте правильность работы полученной НС путем построения разбиения плоскости на классы, которое реализует персептрон.

Задание 5

Задание 5 Классификация данных на плоскости персептроном – линейно разделимые и неразделимые классы

1. Сгенерировать две выборки X-Y1, X-Y2, представляющих собой множество двух линейно разделимых и линейно-неразделимых классов на плоскости.

a. Сгенерировать множество равномерно случайно распределенных точек X в квадрате с углами (0,0) и (1,1).

б. Провести через квадрат прямую линию y = *k* x + *b* так, чтобы с обеих сторон линии было примерно одинаковое количество точек.

Зам. Значения *k* и *b* должны быть уникальными.

Промаркировать точки с одной стороны линии как относящиеся к первому классу, а с другой стороны – ко второму классу. Запомнить разбиение точек как X-Y1.

в. Провести через квадрат кривую линию, разделяющую его на две части такую, чтобы:

- в каждой части было примерно одинаковое число точек
- невозможно было провести прямую линию, разделяющую квадрат на такие же два множества точек.

Например, $y=kx^2+b$ или y = k sin(x+b)

Промаркировать точки с одной стороны линии как относящиеся к 1 классу, а с другой стороны – ко второму классу. Запомнить разбиение точек как X-Y2.

2. Создать персептрон с двойным входом.

Задать его весовые коэффициенты и смещение равными случайными числами.

Обучить персептрон на выборке X-Y1 с помощью набора функций train, trainc, learnp.

Определить количество эпох, затраченное для обучения (чтобы ошибка была равна 0) – для различных начальных значений весовых коэффициентов и смещения. Затем усреднить результат.

Визуализировать процесс обучения:

- постройте динамику изменения весовых коэффициентов [w1,w2], [w1,b], [w2,b] с эпохами обучения;
- постройте динамику (эволюцию) прямых, реализуемых персептроном (по одной прямой за некоторое количество эпох, так. чтобы общее число прямых было не более 10).
- постройте график зависимости ошибки от номера эпохи.

Сравнить полученную прямую с идеальной разделяющей прямой.

3. Обучить персептрон на выборке X-Y1 с помощью набора функций train, trainc, learnpn (обучение с нормализацией). Определить количество эпох обучения, требуемое для достижения нулевой ошибки и сравнить с результатами предыдущего пункта.

4. Обучить персептрон на выборке X-Y1 с применением последовательного случайного алгоритма обучения trainr. персептрон на выборке X-Y1 с применением пакетного алгоритма обучения trainb. Определить количество эпох обучения, требуемое для достижения нулевой ошибки с использованием функций обучения learnp, learnpn. Сравнить результаты с последовательным обучением с использованием функции trainc.

5. Обучить персептрон на выборке X-Y1 с применением пакетного алгоритма обучения trainb. Определить количество эпох обучения, требуемое для достижения нулевой ошибки с использованием функций обучения learnp, learnpn. Сравнить результаты с последовательным обучением с использованием функции trainc, trainr.

6. Для выборки X-Y2 произвести обучение персептрона с использованием последовательных (trainc, trainr) и пакетного (trainb) алгоритмов обучения.

Зам. В данном случае невозможно выполнить обучение без ошибки. Поэтому при обучении необходимо либо задавать ненулевую ошибку, при достижении которой обучение необходимо прекращать (net.trainparam.goal), либо задавать максимальное количество эпох обучения (net.trainparam.epochs).

Задать реально достижимое значение ошибки и определить количество эпох обучения, после которого это значение ошибки достигается.

Сравнить результаты.

Определить среднюю ошибку классификации. Представить графически результаты обучения (разбиение на классы, реализуемое персептроном с учетом реальной принадлежности данных к классам).