МИНИСТЕРСТВО ВЫСШЕГО И СРЕДНЕГО СПЕЦИАЛЬНОГО ОБРАЗОВАНИЯ РЕСПУБЛИКИ УЗБЕКИСТАН

Ташкентский институт текстильной и легкой промышленности

Кафедра
«Автоматизация и компьютеризация технологических процессов»

Методическое указание

к лабораторным работам по курсу
«Интеллектуальные системы управления»

Ташкент 2011 г.

Данное методическое указание предназначено для магистров специальностей 5А521802-«Автоматизация технологических и производственных процессов» и 5А523811-«Автоматизация процессов текстильной промышленности».

Вводятся основные понятия из теории искусственных нейронных сетей. Рассматриваются вопросы реализации и настройки нейронных сетей в системе MATLAB на примере персептронных нейронных сетей.

Рассмотрены линейные нейронные сети, их структурные схемы, правила обучения по методу наименьших квадратов. На задачах классификации векторов и фильтрации временных сигналов показано применение линейных сетей.

Методические указания подготовлены на кафедре "Автоматизация и компьютеризация технологических процессов"

По данному методическому указанию можно изучить предмет «Интеллектуальные системы управления».

Составители: к.т.н., доц. И.Х.Сиддиков,
асс. Ю. А. Жукова
Рецензенты: д.т.н., проф. Арипов Н. М. (ТИИТ), д.т.н., проф. Плахтиев А. М. (ТИТЛП)

Рассмотрено и утверждено на заседании учебно-методического совета ТИТЛП

Протокол №_____ от _____20___ года

Лабораторная работа № 4.
Процедуры настройки параметров персептронных нейронных сетей.
Процедура адаптации

Цель работы: изучение алгоритма настройки параметров персептронных нейронных сетей с помощью процедуры адаптации в системе MATLAB.

Общие сведения

Многократно используя функции sim и learnp для изменения весов и смещения персептрона, можно в конечном счете построить разделяющую линию, которая решит задачу классификации при условии, что персептрон может решать ее. Каждая реализация процесса настройки с использованием всего обучающего множества называется *проходом* или *циклом*. Такой цикл может быть выполнен с помощью специальной функции адаптации adapt. При каждом проходе функция adapt использует обучающее множество, вычисляет выход, погрешность и выполняет подстройку параметров персептрона.

Процедура адаптации не гарантирует, что синтезированная сеть выполнит классификацию нового вектора входа. Возможно, потребуется новая настройка матрицы весов **W** и вектора смещений **b** с использованием функции adapt.

Чтобы пояснить процедуру адаптации, рассмотрим простой пример. Выберем персептрон с одним нейроном и двухэлементным вектором входа (рис. 1.).

Эта сеть достаточно проста, так что все расчеты можно выполнить вручную.

Предположим, что можно с помощью персептрона решить задачу классификации векторов, если задано следующее обучающее множество:

$$\left\{\mathbf{p}_1=\begin{bmatrix}2\\2\end{bmatrix},t_1=0\right\}\left\{\mathbf{p}_2=\begin{bmatrix}1\\-2\end{bmatrix},t_2=1\right\}\left\{\mathbf{p}_3=\begin{bmatrix}-2\\2\end{bmatrix},t_3=0\right\}\left\{\mathbf{p}_4=\begin{bmatrix}-1\\1\end{bmatrix},t_4=1\right\}. \quad (1)$$

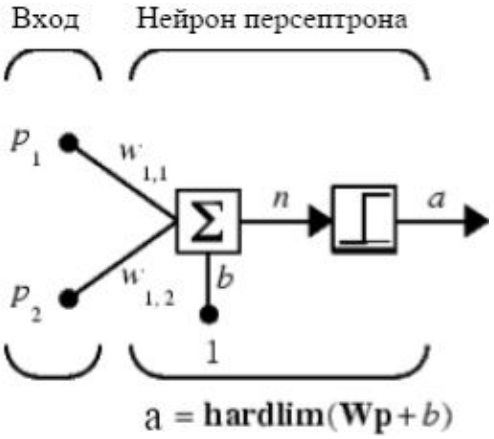

Рис. 1.

Используем нулевые начальные веса и смещения. Для обозначения переменных каждого шага используем круглые скобки. Таким образом, начальные значения вектора весов $w^T(0)$ и смещения $b(0)$ соответственно равны $w^T(0) = [0\ 0]$ и $b(0) = 0$.

1-й шаг процедуры адаптации

Вычислим выход персептрона для первого вектора входа *p1*, используя начальные веса и смещение:

$$a=\text{hardlim}\left(\mathbf{w}^{\mathrm{T}}(0)\mathbf{p}_1+b(0)\right)=\text{hardlim}\left(\begin{bmatrix}0&0\end{bmatrix}\begin{bmatrix}2\\2\end{bmatrix}+0\right)=\text{hardlim}(0)=1. \quad (2)$$

Выход не совпадает с целевым значением $t1$, поэтому необходимо применить правило настройки (обучения) персептрона, чтобы вычислить требуемые изменения весов и смещений:

$$\begin{cases} e = t_1 - a = 0 - 1 = -1; \\ \Delta \mathbf{w}^{\mathrm{T}} = e\mathbf{p}_1^{\mathrm{T}} = (-1)[2 \;\; 2] = [-2 \;\; -2]; \\ \Delta b = e = (-1) = -1; \end{cases} \quad (3)$$

Вычислим новые веса и смещение, используя введенные ранее правила обучения персептрона.

$$\begin{cases} \mathbf{w}^{\mathrm{T}}{}^{new} = \mathbf{w}^{\mathrm{T}}{}^{old} + \Delta \mathbf{w}^{\mathrm{T}} = [0 \;\; 0] + [-2 \;\; -2] = [-2 \;\; -2] = \mathbf{w}^{\mathrm{T}}(1); \\ b^{new} = b^{old} + \Delta b = 0 + (-1) = -1 = b(1). \end{cases} \quad (4)$$

2-й шаг процедуры адаптации

Обратимся к новому вектору входа **p**2, тогда

$$a = \text{hard lim}\left(\mathbf{w}^{\mathrm{T}}(1)\mathbf{p}_2 + b(1)\right) = \text{hard lim}\left(\left[[-2 \;\; -2]\begin{bmatrix} 1 \\ -2 \end{bmatrix}\right] + (-1)\right) = \text{hard lim}(1) = 1. \quad (5)$$

В этом случае выход персептрона совпадает с целевым выходом, так что погрешность равна 0 и не требуется изменений в весах или смещении. Таким образом,

$$\begin{cases} \mathbf{w}^{\mathrm{T}}(2) = \mathbf{w}^{\mathrm{T}}(1) = [-2 \;\; -2]; \\ b(2) = b(1) = -1. \end{cases} \quad (6)$$

3-й шаг процедуры адаптации

Продолжим этот процесс и убедимся, что после третьего шага настройки не изменились:

$$\begin{cases} \mathbf{w}^{\mathrm{T}}(3) = \mathbf{w}^{\mathrm{T}}(2) = [-2 \;\; -2]; \\ b(3) = b(2) = -1. \end{cases} \quad (7)$$

4-й шаг процедуры адаптации

После четвертого примем значение

$$\begin{cases} \mathbf{w}^{\mathrm{T}}(4) = [-3 \;\; -1]; \\ b(4) = 0. \end{cases} \quad (8)$$

Чтобы определить, получено ли удовлетворительное решение, требуется сделать один проход через все векторы входа с целью проверить, соответствуют ли решения обучающему множеству.

5-й шаг процедуры адаптации

Вновь используем первый член обучающей последовательности и получаем

$$\begin{cases} \mathbf{w}^{\mathrm{T}}(5) = \mathbf{w}^{\mathrm{T}}(4) = [-3 \;\; -1]; \\ b(5) = b(4) = 0. \end{cases} \quad (9)$$

6-й шаг процедуры адаптации

Переходя ко второму члену, получим следующий результат:

$$\begin{cases} \mathbf{w}^{\mathrm{T}}(6) = [-2 \;\; -3]; \\ b(6) = 1. \end{cases} \quad (10)$$

Этим заканчиваются ручные вычисления.

Расчеты с использованием функции adapt.

Вновь сформируем модель персептрона, изображенного на рис. 1:

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

Введем первый элемент обучающего множества:

```
p = {[2; 2]}; t = {0};
```

Установим параметр passes (число проходов), равным 1, и выполним один шаг настройки:

```
net.adaptParam.passes = 1;
[net,a,e] = adapt(net,p,t); a,e
a =
    [1]
e =
    [-1]
```

Скорректированные вектор весов и смещение определим следующим образом:

```
twts = net.IW{1,1}, tbiase = net.b{1}
twts =
    -2    -2
tbiase =
    -1
```

Это совпадает с результатами, полученными при ручном расчете.

Теперь можно ввести второй элемент обучающего множества и т. д., то есть повторить всю процедуру ручного счета и получить те же результаты.

Но можно эту работу выполнить автоматически, задав сразу все обучающее множество и выполнив один проход:

```
clear, net = newp([-2 2;-2 2],1);
net.trainParam.passes = 1;
p = {[2;2] [1;-2] [-2;2] [-1;1]};
t = {0 1 0 1};
```

Теперь обучим сеть.

```
[net,a,e] = adapt(net,p,t);
```

Возвращаются выход и ошибка

```
a, e
a =
    [1]    [1]    [0]    [0]
e =
    [-1]    [0]    [0]    [1]
```

Скорректированные вектор весов и смещение определяем следующим образом:

```
twts = net.IW{1,1}, tbiase = net.b{1}
twts =
    -3    -1
tbiase =
     0
```

Моделируя полученную сеть по каждому входу, получим

```
a1 = sim(net,p)

a1 =
    [0]    [0]    [1]    [1]
```

Можно убедиться, что не все выходы равны целевым значениям обучающего множества. Это означает, что следует продолжить настройку персептрона.

Выполним еще один цикл настройки:

```
[net,a,e] = adapt(net,p,t); a, e

a =
    [0]     [0]     [0]     [1]
e =
    [0]     [1]     [0]     [0]
twts = net.IW{1,1}, tbiase = net.b{1}
twts =
    -2    -3
tbiase =
     1
a1 = sim(net,p)

a1 =
    [0]     [1]     [0]     [1]
```

Теперь решение совпадает с целевыми выходами обучающего множества, и все входы классифицированы правильно.

Если бы рассчитанные выходы персептрона не совпали с целевыми значениями, то необходимо было бы выполнить еще несколько циклов настройки, применяя функцию adapt и проверяя правильность получаемых результатов.

Итак, для настройки (обучения) персептрона применяется процедура адаптации, которая корректирует параметры персептрона по результатам обработки каждого входного вектора. Применение функции adapt гарантирует, что любая задача классификации с линейно отделимыми векторами будет решена за конечное число циклов настройки.

Нейронные сети на основе персептрона имеют ряд ограничений. Во-первых, выход персептрона может принимать только одно из двух значений (0 или 1); во-вторых, персептроны могут решать задачи классификации только для линейно отделимых наборов векторов.

Если векторы входа линейно неотделимы, то процедура адаптации не в состоянии классифицировать все векторы должным образом Для решения более сложных задач можно использовать сети с несколькими персептронами. Например, для классификации четырех векторов на четыре группы можно построить сеть с двумя персептронами, чтобы сформировать две разделяющие линии и таким образом приписать каждому вектору свою область.

Итак, основное назначение персептронов – решать задачи классификации. Они великолепно справляются с задачей классификации линейно отделимых векторов, при этом сходимость гарантируется за конечное число шагов.

Количество циклов обучения зависит от длины отдельных векторов, но и в этом случае решение может быть построено. Демонстрационная программа demop4 поясняет, как влияет выброс длины вектора на продолжительность обучения.

Решение задачи классификации линейно неотделимых векторов возможно либо путем предварительной обработки входных векторов с целью сформировать линейное отделимое множество входных векторов, либо путем использования многослойных персептронов. Возможно также применить другие типы нейронных сетей, например, линейные сети или сети с обратным распространением, которые могут выполнять классификацию линейно неотделимых векторов входа._

Порядок выполнения работы

1. Для обучающего множества персептронной нейронной сети, разработанной в лабораторной работе № 3, при нулевых начальных значениях весов и смещения

выполнить процедуру адаптации ручным расчетом и моделированием с использованием функции adapt системы MATLAB.

2. Определить количество циклов настройки сети. Сравнить результаты расчетов с результатами, полученными в лабораторной работе № 3.

3. Осуществить моделирование настроенной нейронной сети для пяти новых наборов входных векторов и проверить правильность решения задачи классификации сетью.

4. Повторить процедуру настройки персептронной нейронной сети при нулевых начальных значениях весов и смещения с использованием функции adapt системы MATLAB, увеличив длину одного из векторов обучающего множества в 10–30 раз. Сравнить количество циклов обучения с результатами п. 1.

5. Повторить процедуру настройки персептронной нейронной сети при нулевых начальных значениях весов и смещения с использованием функции adapt системы MATLAB, уменьшив длину одного из векторов обучающего множества в 10–15 раз. Сравнить количество циклов обучения с результатами п. 1.

6. Распечатать текст программы.

7. Составить отчет, который должен содержать:

– цель лабораторной работы;

– структурную схему нейронной сети;

– ручной расчет настройки сети:

– текст программы и результаты моделирования;

– выводы.

Список литературы

1. Медведев В. С. Нейронные сети / В. С. Медведев, В. Г. Потемкин. – М.: Диалог МИФИ, 2002.

2. Дьяконов В. П. MATLAB 5.3.1 с пакетами расширений / В. П. Дьяконов, И. В. Абраменкова, В. В. Круглов. – М.: Нолидж, 2001.

3. Комашинский В. И. Нейронные сети и их применение в системах управления и связи / В. И. Комашинский, Д. А. Смирнов. – М.: Горячая линия – Телеком, 2002.