УДК 683.519

ИССЛЕДОВАНИЕ НЕЧЕТКИХ НЕЙРОННЫХ СЕТЕЙ В ЗАДАЧАХ РАСПОЗНАВАНИЯ ОБЪЕКТОВ ЭЛЕКТРООПТИЧЕСКИХ ИЗОБРАЖЕНИЙ

Ю.П. ЗАЙЧЕНКО, И.М. ПЕТРОСЮК, М.С. ЯРОШЕНКО

Рассмотрена проблема распознавания объектов электрооптических изображений, полученных с помощью мультиспектральных систем. Для ее решения предлагается нечеткая нейронная сеть (ННС). Описаны различные алгоритмы обучения ННС в задаче классификации объектов. Приведены результаты экспериментальных исследований эффективности этих алгоритмов на реальных данных.

ВВЕДЕНИЕ

В последние годы получили широкое применение мультиспектральные и гиперспектральные системы для получения электрооптических (ЭО) изображений земной поверхности с целью последующего ее зондирования и распознавания на ней различных объектов, поиска полезных ископаемых и т.п. [1].

В частности, эти методы применяются для распознавания объектов на водной поверхности и в прибрежной зоне океана. Задача обнаружения и распознавания объектов затрудняется из-за высокого уровня помех вследствие влияния пены в прибрежной зоне. Кроме того, имеются другие факторы случайной и неопределенной природы.

Наличие неполной нечеткой информации и высокий уровень помех на полученных снимках мультиспектральных систем обусловливают применение систем с нечеткой логикой и нечетких нейросетей для распознавания объектов на ЭО-изображениях земной и морской поверхностях [2, 3].

Цель настоящей статьи — исследование эффективности различных алгоритмов обучения нечеткой нейросети в задаче распознавания объектов на ЭО-изображениях земной и морской поверхностях.

МОДЕЛЬ НЕЙРОНЕЧЕТКОЙ КЛАССИФИКАЦИИ

Модель нейронечеткой классификации впервые предложена Д. Науком и Р. Крузе в 1994 г. [6] и рассматривалась ими как один из вариантов реализации нечеткого перцептрона NEFClass (Neuro Fuzzy Classification).

NEFClass используется для определения класса или категории полученных входных образцов (паттернов). Паттерны — это векторы признаков $x=(x_1,\ldots,x_n)\in\mathbb{R}^n$, а класс C — подмножество $\mathbb{R}^n$. Мы принимаем, что пересечение двух разных классов является пустым. Величина признака паттерна представляется нечетким множеством, и классификация определяется множеством лингвистических правил. Для каждого входного признака x_i существует q_i нечетких множеств $\mu_1^i,\ldots,\mu_q^i$. Также есть база правил, которая вмещает k нечетких лингвистических правил $R_1,\ldots,R_k$.

База правил представляет собой аппроксимацию функции (неизвестной) φ: $\mathbb{R}^n\to[0,1]^m$, которая представляет задачу классификации, где $\varphi(x)=(c_1,\ldots,c_m)$ такая, что $c_i=1$ и $c_j=0$ ($j\in\{1,\ldots,m\}$, $j\neq i$). Значит, x принадлежит классу $C_i\to[0,1]^m$.

Обозначим наибольший компонент каждого вектора c единицей ($c=1$), а все другие компоненты положим равными нулю.

Нечеткие множества и лингвистические правила, обеспечивающие выполнение такой аппроксимации и определяющие результирующую систему NEFClass, будут получены из множества примеров для обучения. На рис. 1 приведена система NEFClass, которая классифицирует входные образцы с двумя признаками и двумя отдельными классами, используя пять лингвистических правил.

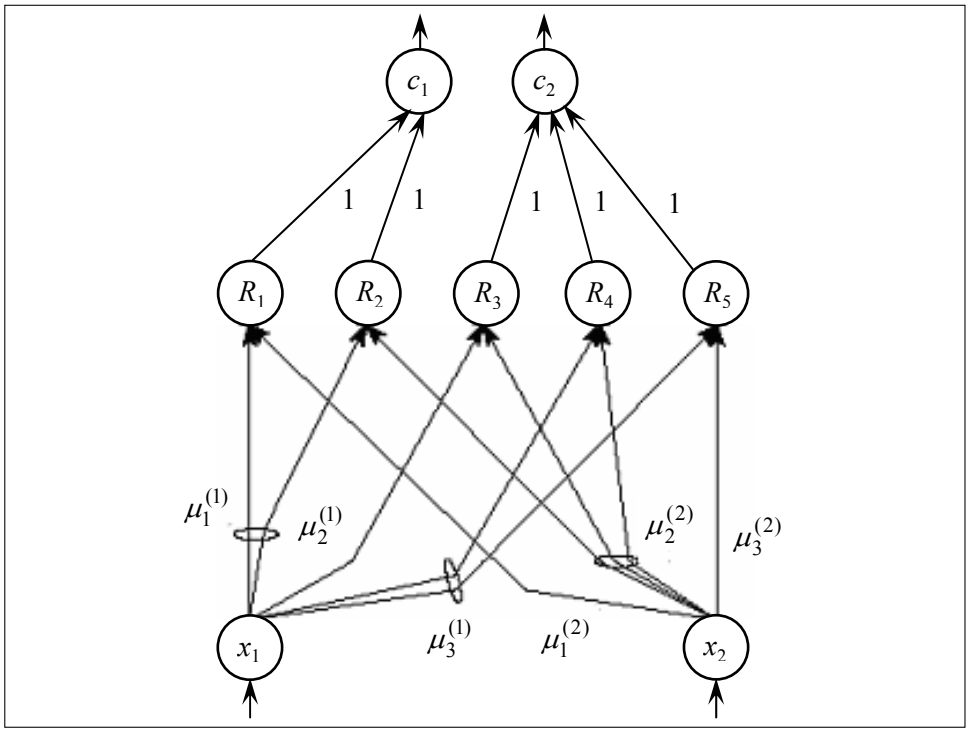

Рис. 1. Модель NEFClass с двумя признаками, двумя классами и пятью правилами

NEFClass — это трехслойный нечеткий персептрон со следующими спецификациями:

1. $U_1 = \{x_1, \ldots, x_n\}$ — входной слой, $U_2 = \{R_1, \ldots, R_k\}$ — слой нейронов правил, $U_3 = \{C_1, \ldots, C_m\}$ — выходной слой классифицирующих нейронов.

2. Каждая связь между узлами $x_i \in U_1$ и $R_r \in U_2$ помечена лингвистическим термом $A_{jr}^{(i)}$ ($j_r \in \{1, \ldots, q_i\}$).

3. $W(R, c) \in \{0, 1\}$ определяется для всех $R \in U_2$, $C \in U_3$.

4. Соединения, которые происходят из одного и того же входного узла x и имеют идентичные метки, одинаковый вес в любой момент времени. Такие соединения называются *связанными соединениями,* а их вес называется взвешенным.

5. Обозначим $L_{x,R}$ метку соединения между узлами $x \in U_1$ и $R \in U_2$, тогда для всех $R, R' \in U_2$ выполняется

$$(\forall (x \in U_1)\, L_{x,R} = L_{x,R'}) \Rightarrow R = R'. \tag{1}$$

6. Для всех узлов правил $R \in U_2$ и всех узлов $C, C' \in U_3$ имеем

$$(W(R, C) = 1) \wedge (W(R, C') = 1) \Rightarrow C = C'. \tag{2}$$

7. Для всех выходных узлов $C \in U_3$ выполняется

$$O_c = a_c = \mathrm{NET}_c. \tag{3}$$

8. Для всех выходных узлов $C \in U_3$ NET_c рассчитывается по формуле

$$\mathrm{NET}_C = \frac{\sum_{R \in U_2} W(R, C) O_R}{\sum_{R \in U_2} W(R, C)}. \tag{4}$$

Система NEFClass может быть построена по неполному знанию об образцах, а затем откалибрована путем обучения или создана с самого начала посредством обучения. Пользователь должен определить количество начальных нечетких множеств, разбивающих интервал значений, в которых лежит признак, а также наибольшее количество узлов правил, созданных в скрытом слое.

Каждое нечеткое множество μ_j^i маркируется лингвистическим термом $A_j^{(i)}$. Это могут быть такие термы, как «малый», «средний», «большой» и т.д. Нечеткие множества этих связей, направленные на один и тот же узел правил R, называются *антецедентами* узла правила R.

ОБУЧЕНИЕ В СИСТЕМЕ NEFCLASS

Система NEFClass может быть построена по частичным знаниям об образцах. Пользователь должен определить количество начальных нечетких множеств и задать значение k — максимальное число узлов правил, которые могут быть созданы в скрытом слое. Для обучения будем использовать треугольную функцию принадлежности. Рассмотрим алгоритм обучения.

Алгоритм обучения системы NEFClass состоит из двух этапов [6–9].

На *первом этапе* генерируется или задается база лингвистических правил вида ЕСЛИ – ТО. На *втором* — проводится настройка весов персептрона (нейронной сети) с помощью эмпирического алгоритма или других теоретически обоснованных алгоритмов обучения нейронных сетей, таких как градиентный и генетический.

Рассмотрим систему NEFClass из n входных нейронов $x_1,\ldots,x_n$, $k \le k_{\max}$ нейронами правил и m выходными нейронами $C_1,\ldots,C_m$. Пусть задано обучающее множество образцов $L=\{(p_1,t_1),\ldots,(p_s,t_s)\}$, каждый из которых состоит из входного $p \in \mathbb{R}^n$ и желаемого $t \in \{0,1\}^m$ образцов.

Алгоритм обучения базы правил подробно рассмотрен в работах [1–3] и поэтому здесь не приводится.

Рассмотрим различные алгоритмы обучения функций принадлежности нечетких правил с целью оценки их эффективности.

КЛАССИЧЕСКИЙ АЛГОРИТМ ОБУЧЕНИЯ

Этот алгоритм, предложенный Д. Науком и Р. Крузе, носит эмпирический характер и не имеет теоретического обоснования [6–9].

Шаги

1. Выбираем следующий образец (p,t) из L, распространяем его через систему NEFClass и определяем выходной вектор C.

2. Для каждого выходного нейрона C_i вычисляем

$$\delta_{C_i} = t_i - O_{C_i}. \tag{5}$$

3. Для каждого нейрона правил R такого, что его выход $O_R > 0$

а) вычисляем значение δ_R, равное

$$\delta_R = O_R(1-O_R)\sum_{C\in U_3} W(R,C)\,\delta_C; \tag{6}$$

б) находим такой вход x', что

$$W(x',R)(O_{x'}) = \min_{x\in U_1}\{W(x,R)(O_x)\}; \tag{7}$$

в) для нечетких множеств $W(x',R)$ определяем величину сдвига для параметров функции принадлежности (ФП), используя скорость обучения $\sigma > 0$

$$\delta_b = \sigma\delta_R(c-a)\,\mathrm{sgn}\,(O_{x'}-b), \tag{8}$$

$$\delta_a = -\sigma\delta_R(c-a)+\delta_b, \tag{9}$$

$$\delta_c = \sigma\delta_R(c-a)+\delta_b \tag{10}$$

и применяем изменения к $W(x',R)$;

г) вычисляем критерий остановки, в качестве которого можно взять, например, такой:

- величина СКО ($\Sigma\delta_{C_i}^2$) в течение n итераций не уменьшается;
- значение СКО или средняя относительная ошибка классификации достигла заданного (желательно близкого к нулю) значения;
- достигнуто минимальное заданное значение процента ошибочной классификации.

Критерий окончания, завершающий процесс обучения, сформулировать не просто, потому что ошибка обычно может и не равняться нулю в соответствии с множеством определений NET_c. Решением в данном случае будет определение максимального количества разрешенных ошибок классификации [6].

ГРАДИЕНТНЫЙ АЛГОРИТМ ОБУЧЕНИЯ В СИСТЕМЕ NEFCLASS

Пусть критерий обучения сети, которая имеет три слоя (один скрытый), такой:

$$e(W)=\sum_{i=1}^{M}\left(t_i-\mathrm{NET}_i(W)\right)^2\to\min, \tag{11}$$

где t_i — желаемое значение i-го выхода нейросети; $\mathrm{NET}_i(w)$ — фактическое значение i-го выхода нейросети, для весовой матрицы $W=[W^I, W^0]$, $W^I=W(x,R)=\mu_j(x)$, $W^0=W(R,C)$.

Т.е., критерий $e(w)$ является средним квадратом ошибки аппроксимации.

Пусть функции активации для нейронов скрытого слоя

$$O_R=\prod_{i=1}^{N}\mu_{j_i}^{(i)}(x_i),\quad j=1,\ldots,q_i, \tag{12}$$

где $\mu_{ji}(x)$ — функция принадлежности, которая имеет вид

$$\mu_{j_i}^{(i)}(x)=e^{-\frac{(x-a_{j_i})^2}{b_{j_i}^{\ 2}}}, \tag{13}$$

и функция активации нейронов выходного слоя (взвешенная сумма)

$$O_C=\frac{\sum_{R\in U_2}W(R,C)O_R}{\sum_{R\in U_2}W(R,C)} \tag{14}$$

или (функция максимума)

$$O_C=\max W(R,C)O_R. \tag{15}$$

Рассмотрим градиентный алгоритм обучения нечеткого персептрона.

1. Пусть $W(n)$ — текущее значение матрицы весов. Алгоритм имеет вид

$$W(n+1)=W(n)-\gamma_{n+1}\nabla_w e(W(n)), \tag{16}$$

где γ_n — размер шага на n-й итерации; $\nabla_w e(W(n))$ — градиент (направление), который уменьшает критерий (11).

2. На каждой итерации сначала обучаем (корректируем) входные веса W, зависящие от параметров a и b (13),

$$a_{ji}(n+1)=a_{ji}(n)-\gamma_{n+1}\frac{\partial e(W)}{\partial a_{ji}}, \tag{17}$$

$$b_{ji}(n+1)=b_{ji}(n)-\gamma'_{n+1}\frac{\partial eV(W)}{\partial b_{ji}}, \tag{18}$$

где γ'_{n+1} — размер шага для параметра b,

$$\frac{\partial e(W)}{\partial a_{ji}}=-2\sum_{k=1}^{M}\left((t_k-\mathrm{NET}_k(w))W(R,C_k)\right)O_R\frac{(x-a_{ji})}{b_{ji}^{\,2}}, \tag{19}$$

$$\frac{\partial e(W)}{\partial b_{ji}}=-2\sum_{k=1}^{M}\left((t_k-\mathrm{NET}_k(w))W(R,C_k)\right)O_R\frac{(x-a_{ji})^2}{b_{ji}^{\,3}}. \tag{20}$$

3. Находим (обучаем) выходные веса

$$\frac{\partial e(\mathrm{W}^0)}{\partial W(R,C_k)}=-\left(t_k-\mathrm{NET}_k(W^0)\right)O_R, \tag{21}$$

$$W_k^0(n+1)=W_k^0(n)-\gamma''_{n+1}\frac{\partial e(W^0)}{\partial W(R,C_k)}. \tag{22}$$

4. $n:=n+1$ и переходим на следующую итерацию.

Градиентный метод является первым предложенным алгоритмом обучения. Он простой в реализации, но имеет недостатки [2]: медленно сходится, находит только локальный экстремум.

МЕТОД СОПРЯЖЕННЫХ ГРАДИЕНТОВ ДЛЯ СИСТЕМЫ NEFCLASS

Алгоритм сопряженных градиентов, как и более общий алгоритм сопряженных направлений, получил применение в области оптимизации потому, что для широкого класса проблем он обеспечивает сходимость к оптимальному решению за конечное число шагов [2].

Название *сопряженные направления* происходит от использования сопряженных векторов. В векторном пространстве размерности N множество векторов $\{P_1,P_2,...,P_D\}$ образует множество сопряженнных направлений относительно матрицы A, если

$$P_iAP_j=0 \quad \text{для} \quad i\neq j, \tag{23}$$

где A — положительно определенная матрица размерности $N\times N$.

Векторы, которые удовлетворяют (23), называют *A-сопряженными.*

В *N-мерном* пространстве есть ровно $N-1$ независимых векторов, которые образуют *A-сопряженную* пару с вектором p_1.

Таким образом, чтобы найти оптимальное решение, нам необходимо только конечное число направлений.

Алгоритм сопряженных направлений систематически конструирует множество *A*-сопряженных векторов. Через максимум N шагов алгоритм найдет оптимальное направление (для квадратичной функции) и сходимость будет обеспечена.

Описание алгоритма

0. Предположим, что $K=0$. Инициализировать весовой вектор W и вычислить градиент $G=\text{grad}\,E(W)$. Предположим, что вектор начального направления $p_K=-\dfrac{G}{\|G\|}$.

1. Найти скаляр α^*, который минимизирует $E(W+\alpha p)$, для чего можно использовать метод Фибоначчи или «золотого сечения».

$$W(K+1)=W(K)+\alpha^* p(K). \tag{24}$$

2. Если $E(W(K+1))<\varepsilon_{\text{доп}}$, где $\varepsilon_{\text{доп}}$ — допустимая точность достижения минимума, то СТОП. Иначе — вычислить новое направление

$$G(k+1)=\text{grad}\,E(W(k+1)). \tag{25}$$

3. Если $\text{mod}\,\text{K}=0$, то новый вектор направления

$$P(k+1)=-\frac{G(k+1)}{\|G(k+1)\|}. \tag{26}$$

Иначе

$$\beta=\frac{G(K+1)^T G(K+1)}{G(K)^T G(K)} \tag{27}$$

и вычислить новый вектор направления

$$P_{k+1}=\frac{-G(k+1)+\beta p(k)}{\|-G(k+1)+\beta p(k)\|}. \tag{28}$$

4. Заменить $p(k)$ на $p(k+1)$ и $G(k)$ на $G(k+1)$. Переходим на шаг 1 следующей итерации.

В нашем алгоритме $G(k)$ рассчитывается для двух параметров (a и b) последовательно и отдельно, как приведено в градиентном алгоритме. Скорость обучения для этих двух параметров настраивается также отдельно.

ГЕНЕТИЧЕСКИЙ МЕТОД ОБУЧЕНИЯ ДЛЯ СИСТЕМЫ NEFCLASS

Это алгоритм глобальной оптимизации. В нем используются следующие механизмы [2]:

1) скрещивание родительских пар, генерация потомков;

2) мутация (действие случайных влияний);

3) естественный отбор лучших (селекция).

Цель обучения — минимизация среднеквадратической ошибки

$$E(W) = \frac{1}{M}\sum_{k=1}^{M}\left(t_k - \mathrm{NET}_k(W)\right)^2, \qquad (29)$$

где M — количество классов; t_k — желаемый класс; $\mathrm{NET}_k(W)$ — результат классификации; $W = [W_1, W_0]$, $W_1 = \left\| w_{ij}^I \right\|$, $W = \left\| w_{ij}^0 \right\|$ — веса, причем w_{ij}^I — это терм.

Любая особь представляется соответствующим вектором весов W.

Задается начальная популяция из N особей $\left[W_1(0),...,W_i(0),...,W_N(0)\right]$.

Вычисляем индекс пригодности (FI) и оцениваем качество прогнозирования

$$FI(W_i) = C - E(W_i) \to \max, \qquad (30)$$

где C — константа.

Дальше происходит скрещивание родительских пар. При выборе родителей используется вероятностный механизм. Обозначим P_i — вероятность выбора *i*-го отца.

$$P_i = \frac{FI\left(W_i(0)\right)}{\sum_{i=1}^{N} FI\left(W_i(0)\right)}. \qquad (31)$$

Потом осуществляется скрещивание выбранных пар.

Можно применять разные механизмы скрещивания. Например: для первого потомка берутся нечетные компоненты из вектора первого родителя, а четные — из второго:

$$W_i(0) \oplus W_k(0) \Rightarrow W_i(1) + W_k(1). \qquad (32)$$

Берется $\frac{N}{2}$ родительских пар (N — четное) и генерируются N потомков.

После того как сгенерированы потомки, на популяцию действует мутация

$$w'_{ij}(n) = w_{ij}(n) + \xi(n), \qquad (33)$$

где $\xi(n) = ae^{-\alpha n}$, $a = \mathrm{const} \in [-1;+1]$; α — показатель угасания мутации выбирается случайно из интервала [0, 1].

Дальше, после действия мутации, происходит селекция из популяции, позволяющей выбрать наиболее «приспособленные» особи. Можно использовать разные механизмы селекции.

1. Полная замена старой популяции новой.

2. Выбор N лучших из всех существующих особей $N_{\text{род}} + N_{\text{потомк}} = 2N$ по критерию максимума $\max(FI)$.

После выполнения скрещивания, мутации и селекции текущая итерация заканчивается. Итерации повторяются до тех пор, пока не будет выполняться один из критериев останова.

ЭКСПЕРИМЕНТЫ ПО РАСПОЗНАВАНИЮ ОБЪЕКТОВ ДАННЫХ

Для выбора данных из электрооптических изображений используется система ENVI и возможность картографировать, т.е. совмещать изображения по контрольным точкам. Изображения получены из разных спектральных камер [4], что позволяет иметь мультиспектральные изображения (рис. 2).

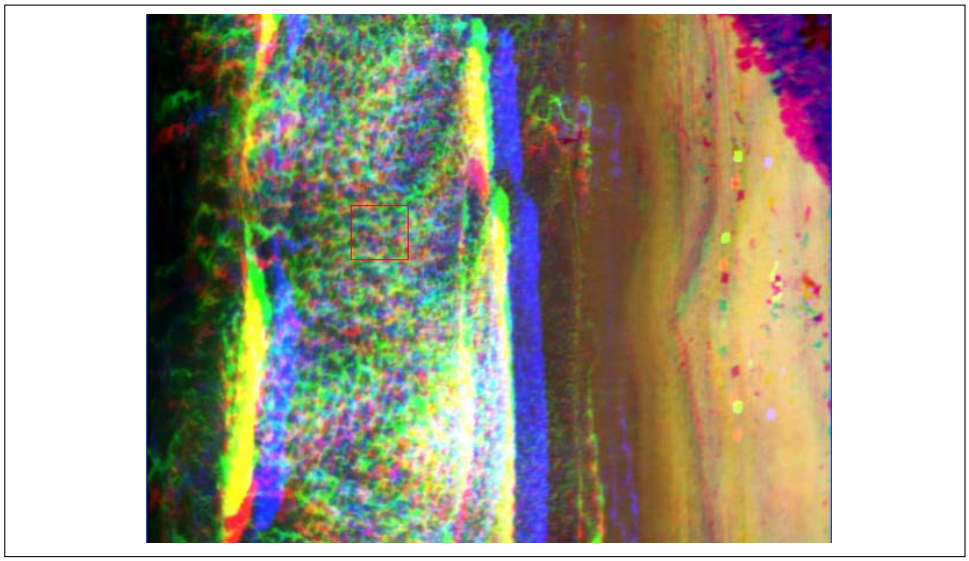

Рис. 2. Результаты картографирования

После выбора 15 контрольных точек из разных спектров (эта функция не автоматизирована) изображения совмещаются, и мы получаем так называемый мультиспектральный куб (рис. 3).

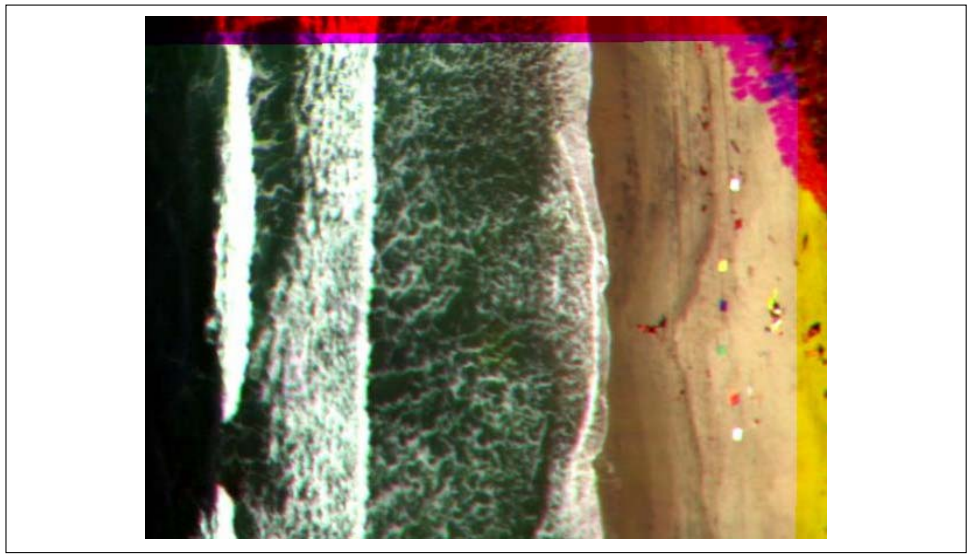

Рис. 3. Мультиспектральные изображения

Установлено девять типов разных поверхностей, которые необходимо классифицировать, для чего использовались так называемые ROI (Region of

Interest). На изображении определялась однородная область, например, песок, вода, пена, цели красного и белого цветов и т. д. (рис. 4).

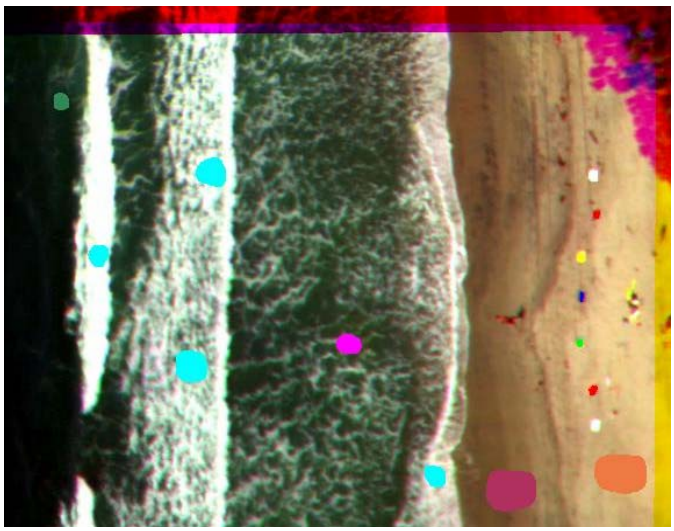

Рис. 4. Изображение из ROI

Используя систему обработки, определяют среднее значение и дисперсию для выбранного региона. Полученные таким образом данные сводятся в таблицу. Они характеризуют девять классов областей поверхности.

Цели: красная, зеленая, синяя, желтая. Поверхности: пена, вода, сухой и мокрый песок.

Т.е., эти виды поверхностей отвечают девяти выходным узлам в системе NEFClass.

Общее количество признаков, которые используются для классификации видов поверхностей, —четыре: яркость в красном спектре (КС), голубом (ГС), зеленом (ЗС), инфракрасном (ИС).

Общее количество данных составляет 99, по 11 на каждый класс.

Приведем основные статистические характеристики для набора данных, полученных с помощью мультиспектральной «Mantis» (табл. 1).

Таблица 1. Статистические характеристики для данных системы «Mantis»

Признаки	**Минимум**	**Максимум**	**Среднее**	**Стандартное отклонение**	**Корреляция между признаками и классом**
Яркость в КС	28,81	255,00	165,40	76,14	−0,46
— // — ГС	72,93	255,00	165,43	68,62	−0,32
— // — ЗС	44,34	254,89	121,57	57,64	−0,52
— // — ИС	17,03	255,00	140,84	81,58	−0,49

Далее проводим эксперименты по обучению распознавания образов, используя разработанную авторами программу моделирования NEFClass–BGCGG. Согласно основному принципу исследования моделей, будем проводить эксперименты, последовательно изменяя только один параметр.

Загружаем данные. Из имеющихся 99 образцов загружаем 54 в качестве обучающей выборки. Другие 45 будут использованы для тестирования.

Устанавливаем параметры в исходное положение (табл. 2).

Таблица 2. Значение параметров для работы программы

Параметр	Значение
Алгоритм генерации правил	Лучший для класса
Алгоритм обучения	Классический
Количество генерирующих правил	Максимальное
Функция агрегации	Взвешенная сумма
Обучение взвешенных коэффициентов между слоем правил и выходным слоем	Свободное
Количество термов для каждого признака	5 для всех
Скорость обучения для взвешенных коэффициентов между признаками и правилами	$\sigma_a = 0{,}1$ $\sigma_b = 0{,}1$ $\sigma_c = 0{,}1$
Скорость обучения для взвешенных коэффициентов между правилами и выходным слоем	$\sigma = 0{,}1$
Максимальное количество эпох	50

Во время процесса обучение было сгенерировано 15 правил (табл. 3).

Таблица 3. База правил нечеткого классификатора

Номер правила	Значение ФП для признака				Класс
	1	2	3	4	
1	4	4	4	4	0
2	4	0	1	4	1
3	4	0	0	4	1
4	4	1	0	4	1
5	2	3	1	1	2
6	1	0	1	0	3
7	4	4	1	4	4
8	3	4	3	3	5
9	3	3	2	3	5
10	4	4	3	3	5
11	0	0	0	0	6
12	3	2	1	2	7
13	1	0	0	1	8
14	1	1	0	1	8
15	1	0	0	0	8

Исследуем зависимость качества обучения от количества правил, которые генерируются на первом этапе. В качестве проверки проведем тестирование на проверочной выборке. Для этого зададим количество правил, начиная от 9 до 14 (табл. 4).

Результат закономерен. Чем больше правил, тем лучше результат тестовой классификации, т.е. в правилах сохраняется информация о каждом классе.

Таблица 4. Зависимость качества обучения от количества правил

Количество правил	СКО	Ошибочная классификация (ОК), %
9	13,071009	24
10	9,545608	15
11	9,910701	15
12	9,705482	15
13	4,769655	4
14	4,739224	4
15	4,751657	4

Исследуем влияние количества термов в признаках на качество обучения (табл. 5).

Таблица 5. Зависимость качества обучения от количества термов

Количество термов	СКО	ОК, %
4	5,928639	4
5	4,626252	4
6	4,957257	4
7	5,228448	4
8	5,633563	4
9	6,797175	4
10	7,897521	7

Очень интересный результат мы получили во время серии опытов. Из табл. 4 видно, что существует оптимальное количество термов, которые могут быть использованы для описания набора данных во время обучения. При увеличении количества термов растет количество ошибочно классифицированных образцов, т.е. при увеличении сложности модели увеличивается ошибка. Такая же ситуация наблюдается и в методе группового учета аргументов.

Проведем обучение системы классическим алгоритмом с оптимальным количеством термов на признаках. Построенные ФП для лингвистических значений признаков приведены на рис. 5.

Итоговая сумма квадратов ошибок составила 2,852081, количество ошибочных классификаций — 0, при тесте, как мы уже видели, количество ошибочных классификаций — 4%, СКО равняется 4,626252, что является хорошим результатом.

Проведем эксперименты для градиентного алгоритма (рис. 6).

Ошибка в конце обучения — 2,042015, что немножко лучше классического метода. При тестировании СКО равнялась 3,786005, а доля ошибочных классов — 4%. Далее была включена автоматическая настройка скорости адаптации параметров ФП, для чего использовался алгоритм «золотого сечения».

Проведем такие же эксперименты с алгоритмом сопряженного градиента (рис. 7).

И в завершение были проведены эксперименты по обучению генетическим алгоритмом с разными ФП — треугольной и гауссовской.

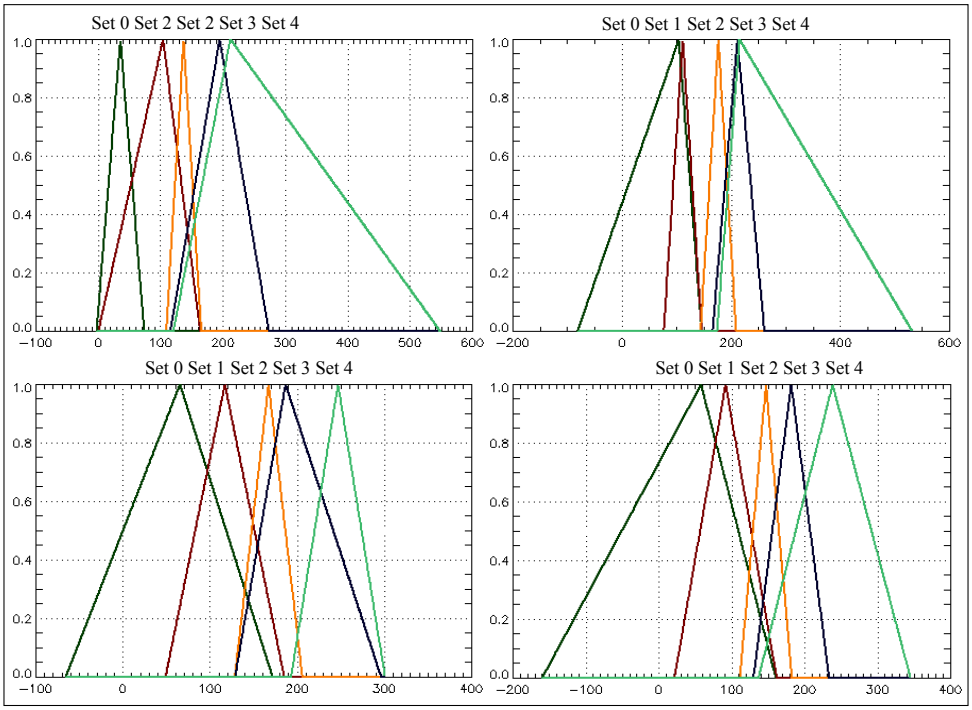

Рис. 5. Результат обучения классическим алгоритмом

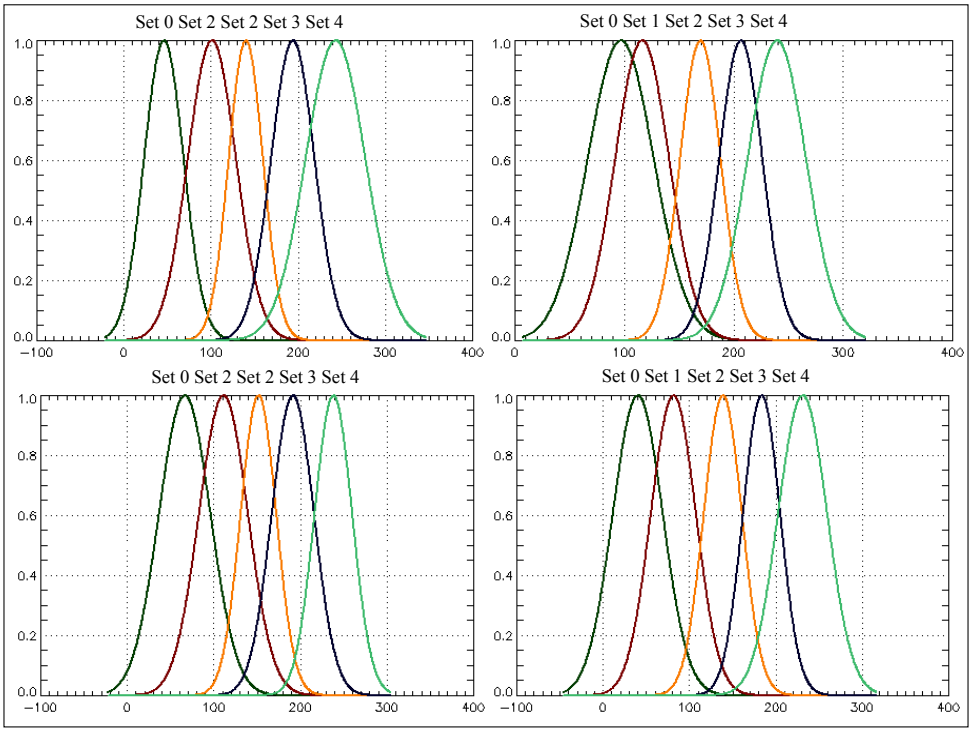

Рис. 6. Результат обучения градиентным методом

Результаты обучения разными алгоритмами показаны в табл. 6. Кстати, во время обучения для всех алгоритмов получен отличный результат по ошибочной классификации.

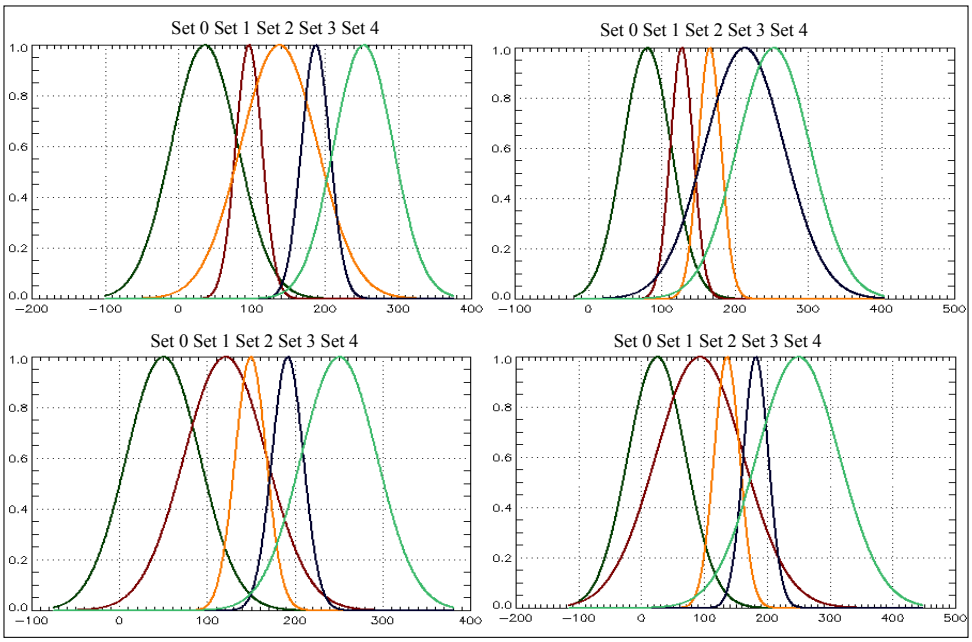

Рис. 7. Результат обучения методом сопряженного градиента

Таблица 6. Сравнительная таблица для разных алгоритмов обучения весов связей

Алгоритм обучения взвешенных коэффициентов	Обучение		Тестирование	
	СКО	ОК, %	СКО	ОК, %
Классический	6,650668	0	7,285827	4
Градиентный	5,9893	0	6,829068	4
Сопряженного градиента	1,132871	0	3,314763	4
Генетический с треугольной ФП	11,110936	0	13,677424	4
Генетический с гауссовской ФП	3,204446	0	4,568338	4

Для всех этот показатель равнялся нулю. Однако во время тестирования результаты были хуже: как минимум два образца классифицировались ошибочно. Росла также сумма квадратов ошибок для всех без исключения алгоритмов обучения. Для удобства количество итераций ограничивалось до 50.

Как видим, результаты удовлетворительны, уровень правильной классификации на проверочной части выборки составляет 96%. Но результаты можно было бы улучшить, сформировав более объемную выборку (рис. 8).

На графиках хорошо видно, что наилучшим методом по скорости сходимости является метод сопряженных градиентов. Затем — генетический с гауссовской функцией принадлежности. Менее эффективный — классический метод. Наименее эффективный в применении по скорости сходимости к минимуму — генетический метод с треугольной функцией принадлежности.

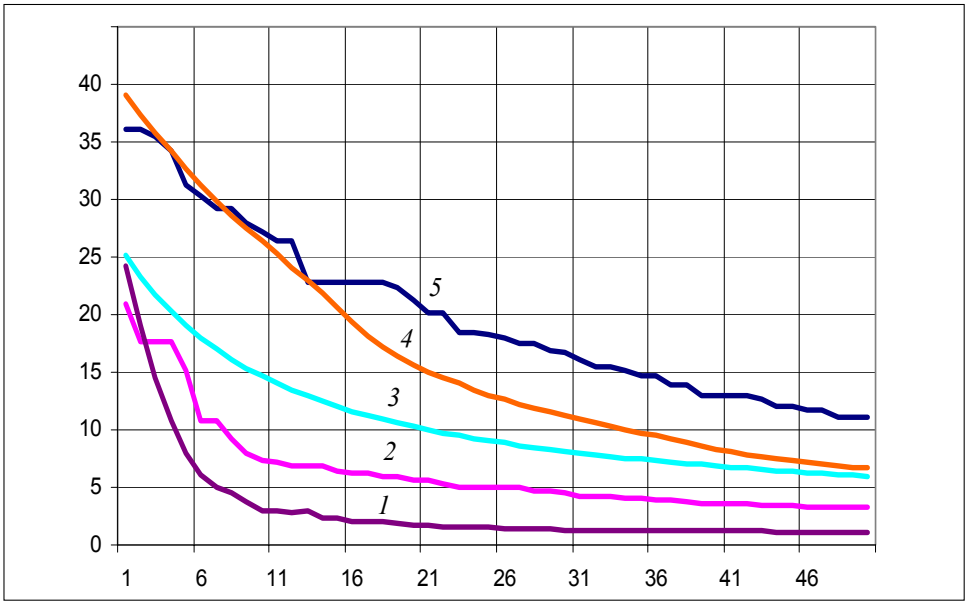

Рис. 8. Сравнительный график скорости сходимости к оптимальной классификации различных алгоритмов обучения: *1* — СГ; *2* — генетический с гаусс. ФП; 3 — градиентный; *4* — классический; *5* — генетический с треуг. ФП

Однако критерий, по которому построен график, неоднозначно отображает качество классификации. Важным критерием оценки методов является минимальное количество ошибочно классифицированных образцов. Все алгоритмы дали одни и те же итоговые результаты по данному критерию (табл. 6). Следовательно, мы убедились, что все алгоритмы работают адекватно и дают почти одинаковые результаты.

ВЫВОДЫ

1. На тестовых данных исследовано влияние различных параметров на процесс обучения в нечеткой нейросети. Установлено, что качество обучения зависит от скорости обучения. Мы увеличили параметр скорости обучения σ для весовых коэффициентов, которые лежат между признаками и правилами и значительно приблизились к минимальной точке функции ошибки. Но во время эксперимента было замечено, что задавать большую скорость нецелесообразно, так как возникает явление «осцилляции».

2. Наилучший алгоритм по скорости обучения — алгоритм сопряженных градиентов. Затем — генетический с гауссовской функцией принадлежности, за ними следуют градиентный и классический методы. Наиболее медленный по скорости сходимости — генетический метод с треугольной функцией принадлежности.

3. В целом на основе проведенных экспериментов на реальных данных подтверждена целесообразность применения нейронечеткого подхода к классификации объектов на электрооптических изображениях. Полученные нечеткие правила дают возможность эксперту или автоматизированной системе быстро проводить достоверную классификацию.

ЛИТЕРАТУРА

1. *Абламейко С.В., Лагуновский Д.М.* Обработка изображений: технология, методы, применение. — Минск: Ин-т технической кибернетики НАН Беларуси, 1999. — 300 с.
2. *Зайченко Ю.П.* Основи проектування інтелектуальних систем. Навчальний посібник. — Київ: Видавничий дім «Слово», 2004. — 352 с.
3. *Петросюк І.М., Зайченко Ю.П.* Порівняльний аналіз нейро-нечітких систем класифікації в умовах інформаційної невизначеності // Системні дослідження та інформаційні технології. — 2006. — № 3. — С. 110–124.
4. *ENVI* Tutorials. September, 2003, Edition, 2003. — http://www.RSIinc.com.
5. *Fuller R.* Introduction to Neuro-Fuzzy Systems. Advances in Soft Computing Series. — Berlin: Springer Verlag, 1999. — 240 p.
6. *Nauck D.* A fuzzy perceptron as a generic model for neuro-fuzzy approaches / Proc. Fuzzy-Systeme'94, Munich, October 1994. — P. 180–190.
7. *Nauck D., Kruse R.* NEFCLASS – A Neuro-Fuzzy Approach For The Classification of Data. — Applied Computing, 1995. — http://www.cs.tu-bs.de/~nauck/.
8. *Nauck D., Kruse R.* What are Neuro-Fuzzy Classifiers / Proc. Seventh International Fuzzy Systems Association World Congress IFSA'97. — Academia Prague, 1997. — **IV**. — P. 228–233.

Поступила 20.11.2007