УДК 519.71

О.Г. РУДЕНКО, А.А. БЕССОНОВ

РОБАСТНАЯ НЕЙРОЭВОЛЮЦИОННАЯ ИДЕНТИФИКАЦИЯ НЕЛИНЕЙНЫХ НЕСТАЦИОНАРНЫХ ОБЪЕКТОВ

Аннотация. Предложено использование нейроэволюционного подхода к построению математических моделей нелинейных нестационарных объектов при наличии негауссовских помех измерений. Рассмотрена общая структура эволюционной нейросети прямого распространения. Проведено имитационное моделирование различных случаев нестационарности, подтвердившее эффективность развиваемого подхода.

Ключевые слова: идентификация, робастность, нелинейный нестационарный объект, искусственная нейронная сеть, эволюционные вычисления, генетический алгоритм.

ВВЕДЕНИЕ

Задача идентификации нелинейного нестационарного динамического объекта состоит в получении оценки описывающей его нелинейной функции по измерениям входных и выходных переменных.

Классические методы построения математической модели объекта используют аппроксимацию нелинейности рядами или полиномами и требуют решения задач структурной (определение вида и степени аппроксимирующего полинома) и параметрической (определение коэффициентов разложения) идентификации. Сложность получения адекватного математического описания существенно возрастает, если параметры исследуемого объекта или условия его функционирования являются нестационарными.

Для решения данной задачи в последнее время все более широко применяется нейроэволюционный подход, сочетающий эволюционные вычисления и искусственные нейронные сети (ИНС).

ИНС В ЗАДАЧЕ ИДЕНТИФИКАЦИИ

Рассмотрим задачу идентификации нелинейного нестационарного динамического объекта, описываемого моделью

$$y(k) = f(x(k), k) + \xi(k), \tag{1}$$

где $x(k) = [y(k-1), \ldots, y(k-l), u(k-1), \ldots, u(k-n)]^{\mathrm{T}}$ — $N \times 1$-вектор обобщенного входного сигнала ($N = l + n$); $y(i)$, $u(i)$ — выходной и входной сигналы объекта в момент времени i соответственно; l и n — порядки запаздывания по выходному и входному каналам соответственно; $f(\cdot)$ — неизвестная нелинейная функция; $\xi(x)$ — помеха.

Среди ИНС наибольшее распространение при решении задачи идентификации нелинейных динамических объектов (1) в настоящее время получили многослойный персептрон (МП) и радиально-базисные сети (РБС), использующие соответственно аппроксимации нелинейного оператора $f(\cdot)$ вида

$$\hat{y}(k) = \hat{f}(k) = f^q[(W^q)^{\mathrm{T}} f^{q-1}[(W^{q-1})^{\mathrm{T}} f^{q-2}[\ldots f^1[(W^1 x(k) + b_1)^{\mathrm{T}}]\ldots]]] + b_q \tag{2}$$

и

$$\hat{y}(k) = \hat{f}(k) = w_0 + \sum_{i=1}^{N} w_i \Phi_i(x) = w_0 + W^{\mathrm{T}} \Phi(k), \tag{3}$$

где W^i — вектор весовых параметров нейронов i-го слоя сети, $f^i[\bullet]$ — активационная функция (АФ) i-го слоя, b_i — смещение i-го нейрона, $\Phi(k)$ — вектор выбранных базисных функций (БФ).

При нейросетевом подходе задача идентификации заключается в обучении сети, состоящем в определении вектора ее параметров θ (весов, параметров активационных и базисных функций и т.д.), обеспечивающего минимум функционала

$$F(e) = \frac{1}{k}\sum_{i=1}^{k}\rho(e(i,\theta)), \tag{4}$$

т.е. являющегося решением системы уравнений

$$\frac{\partial F(e)}{\partial \theta_j} = \sum_{i=1}^{k}\rho'(e(i,\theta))\frac{\partial e(i,\theta)}{\partial \theta_j} = 0. \tag{5}$$

Здесь $\rho(e(i,\theta))$ — некоторая функция потерь, зависящая от вида закона распределения помехи ξ; $e(i) = y(i) - \hat{y}(i)$, $\hat{y}(i)$ — выходной сигнал модели; $\rho'(e(i,\theta)) = \dfrac{\partial\rho(e(i,\theta))}{\partial e(i,\theta)}$ — функция влияния. Введение весовой функции $\omega(i,\theta) = \dfrac{\rho'(e(i,\theta))}{e(i,\theta)}$ позволяет записать систему уравнений (5) следующим образом:

$$\sum_{i=1}^{k}\omega(e(i,\theta))e(i,\theta)\frac{\partial e(i,\theta)}{\partial \theta_j} = 0. \tag{6}$$

При этом минимизация функционала (4) эквивалентна минимизации взвешенного квадратичного функционала

$$F(e) = \frac{1}{k}\sum_{i=1}^{k}\omega(i,\theta)e^2(i,\theta). \tag{7}$$

При использовании ИНС также возникают задачи структурной и параметрической оптимизации, соответствующие выбору оптимальной топологии сети и ее обучению (настройке параметров). Если задача определения структуры является дискретной оптимизационной (комбинаторной), то поиск оптимальных параметров осуществляется в непрерывном пространстве с помощью классических методов оптимизации.

Традиционные методы определения структуры сети заключаются либо в последовательном ее усложнении путем ввода новых нейронов и новых связей между ними, либо в последовательном ее упрощении, начиная с некоторой достаточно сложной топологии. Наиболее известный метод выбора топологии сети с одновременным определением весов — каскадная корреляция, использующий обучение с учителем и основанный на максимизации значения корреляции между выходами вновь вводимого узла (нейрона) и ошибкой ИНС, был предложен С. Фалманом [1, 2]. Структура каскадно-корреляционной сети, начальная топология которой включает только один нейрон, путем автоматического обучения становится многослойной. При этом число скрытых нейронов, определяющих сложность сети, пошагово увеличивается, уменьшая ошибку обучения. Основной недостаток данного метода — сложность распараллеливания вычислений вследствие наличия связей между всеми скрытыми узлами — устраняется различными модификациями данного метода (использование упрощенной каскадной корреляции, случайного выбора слоя и т.д.) [3].

Для обучения (оценивания параметров) сети применяются, как правило, методы, требующие вычисления градиента используемого функционала $\rho'(e(i,\theta))$: алгоритм обратного распространения ошибки (ОР), метод сопряженных градиентов, алгоритм Гаусса–Ньютона, алгоритм Левенберга–Марквардта и т.д. Несмотря на популярность этих методов не только при обучении ИНС, но и при решении других задач оптимизации, они имеют ряд существенных недостатков. Так, получаемое решение зависит от формы минимизируемой функции, вектора начальных условий, они «застревают» в локальных экстремумах, а при наличии нескольких экстремумов не обеспечивают нахождения глобального, их применение требует дифференцируемости минимизируемых функционалов. Кроме того, сложным является определение оптимальных параметров алгоритмов, обеспечивающих их максимальную скорость сходимости, точность, робастность и т.д.

Попытки устранить недостатки традиционных методов синтеза и функционирования ИНС привели к появлению нового класса сетей — эволюционирующих ИНС (ЭИНС), в которых в дополнение к традиционному обучению используется иная фундаментальная форма адаптации — эволюция, реализуемая применением эволюционных вычислений [4–10].

Эволюционный подход использует в качестве основных структурных элементов вычислительные модели механизмов естественной эволюции в отличие от обучения, базирующегося на коннективистском подходе к моделированию деятельности мозга с помощью ИНС. Если эволюция — достаточно медленный процесс, определяющий законы изменения вида, то обучение является процессом более быстрым, улучшающим адаптацию особи к изменяющимся условиям путем изменения ее характеристик.

С использованием в ЭИНС этих двух форм адаптации — эволюции и обучения, позволяющих изменять структуру сети, ее параметры и алгоритмы обучения без внешнего вмешательства, данные сети становятся наиболее приспособленными для работы в нестационарных условиях и при наличии неопределенности относительно свойств исследуемого объекта и условий его функционирования.

Основным преимуществом использования эволюционных алгоритмов (ЭА) в качестве алгоритмов обучения является то, что многие параметры ИНС могут быть закодированы в геноме и определяться параллельно. Более того, в отличие от большинства алгоритмов оптимизации, предназначенных для потактового решения задачи, ЭА оперируют множеством решений — популяцией, что позволяет достичь глобального экстремума, не «застревая» в локальных. При этом информация о каждой особи популяции кодируется в хромосоме (генотипе), а получение решения (фенотипа) осуществляется после эволюции (отбора, скрещивания, мутации) путем декодирования.

Среди ЭА, являющихся стохастическими и включающих эволюционное программирование, эволюционные стратегии, генетические алгоритмы, генетическое программирование, в частности программирование с экспрессией генов, большое распространение получили генетические алгоритмы (ГА) [11].

Классический ГА содержит следующие шаги.

1. Создание начальной популяции.
   1.1. Инициализация хромосомы каждой особи.
   1.2. Оценивание начальной популяции.
2. Этап эволюции — построение нового поколения.
   2.1. Отбор кандидатов на скрещивание (селекция).
   2.2. Скрещивание, т.е. порождение каждой парой отобранных кандидатов новых индивидов.
   2.3. Мутация.
   2.4. Оценивание новой популяции.

Критерием останова алгоритма обычно является либо прохождение максимально допустимого количества поколений, либо достижение функцией приспособленности какой-либо особи некоторого заранее заданного порогового значения.

ГА абстрагируют фундаментальные процессы дарвиновской эволюции: естественного отбора и генетических изменений вследствие рекомбинации и мутации. Однако помимо дарвиновских ГА могут быть реализованы механизмы эволюции Ламарка, изменяющие (улучшающие) хромосомы, и Болдуина, улучшающие приспособляемость хромосомы без ее изменения [12, 13].

Эволюция Ламарка основана на наследовании приобретенных характеристик, полученных в процессе обучения. Так как ламаркизм не может быть обнаружен в биологических системах, был разработан биологически более правдоподобный механизм, основанный на эффекте Болдуина. В отличие от обучения Ламарка данный тип эволюции не позволяет родителям передавать полученную ими в процессе обучения информацию своим потомкам, а сохраняет после обучения только значения фитнесс-функции, определяя тем самым расстояние до глобального оптимума.

ЭВОЛЮЦИОННАЯ ОПТИМИЗАЦИЯ ИНС

Как отмечалось выше, эволюция в ЭИНС может касаться архитектуры сети, ее весов, вида и параметров АФ (БФ), алгоритма обучения (рис. 1). При использовании ГА для построения ЭИНС одной из основных проблем является выбор метода кодирования возможного решения и генетических операторов. В различных задачах используются различные методы кодирования и генетические операторы.

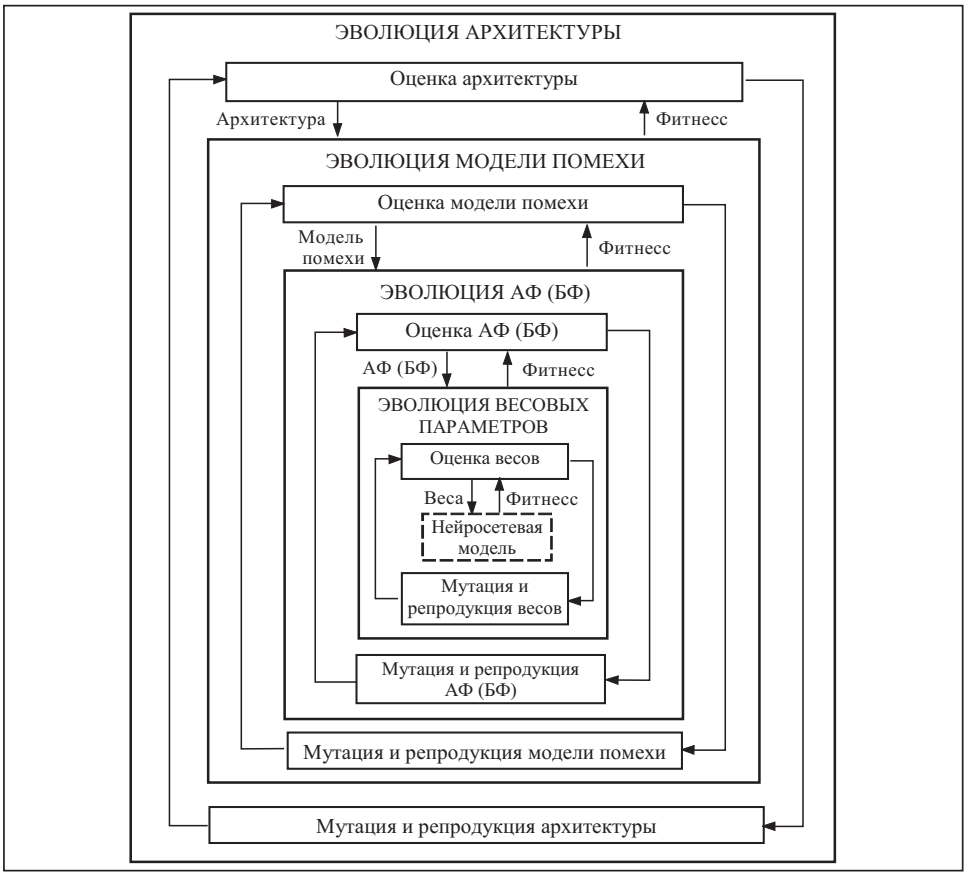

Рис. 1

Так, эволюционная оптимизация топологии сети состоит из двух фаз: кодирование (прямое, когда хромосома содержит всю информацию о топологии сети (слои, нейроны, связи), либо непрямое, когда в хромосоме кодируется только число слоев и число нейронов в каждом слое, что существенно уменьшает длину хромосомы) и реализация ЭА.

Обычно настройка параметров осуществляется после получения приемлемой топологии сети, что характеризуется допустимым значением приспособленности [14, 15]. Эволюционная настройка весовых параметров сети также требует задания способа их кодирования (двоичное или вещественное), влияющего на выбор операторов скрещивания и мутации, а также определяющего исследуемое пространство поиска и отношение между генотипом и фенотипом (отношение «один-к-одному», «один-ко-многим» и «многие-к-одному»). При этом, однако, необходимо учитывать, что вследствие случайной инициализации весов и применения конкретного алгоритма обучения возникают ошибки в оценивании приспособленности (фитнесс-функции). Так, изменение начальных значений приводит к тому, что один и тот же генотип после обучения может иметь разную приспособленность. Попытка получения усредненных результатов путем обучения с раз-

ными начальными условиями приводит к резкому возрастанию вычислительных затрат. Кроме того, при одних и тех же начальных условиях различные алгоритмы обучения могут приводить к различным результатам, особенно если используемый критерий обучения имеет несколько экстремумов. В результате имеем отображение типа «один-ко-многим» генотипов в фенотипы [4, 5]. Поэтому, как показано в [4, 5, 16, 17], получить более точные результаты можно путем одновременной эволюции архитектуры и весов.

Ввиду того, что ИНС образуют нейроны, осуществляющие нелинейные преобразования сигналов в соответствии с видом их активационных функций, выбор этих функций также играет важную роль в нейросетевом представлении, так как существенно влияет на сложность вычислений, и для упрощения обычно предполагается, что активационные функции нейронов одинаковы и выбраны экспертом. Информация об активационной функции также может быть закодирована в хромосоме.

Эффективность алгоритма обучения сети существенно зависит от ее архитектуры, поэтому для полученной определенным образом и неизменяющейся топологии сети может быть найден оптимальный алгоритм настройки ее параметров. Однако построение такого алгоритма в общем случае невозможно, поэтому возникает задача автоматической коррекции (адаптации) правила обучения в зависимости от изменения условий функционирования, архитектуры сети. При использовании эволюционного подхода для этих целей адаптация правила обучения может рассматриваться как динамический процесс «обучения обучению» сети [4, 5, 18–20]. Следует отметить, что в этом случае может осуществляться как адаптация отдельных параметров алгоритма, например коэффициента усиления, параметра взвешивания, так и изменение его структуры, например переход от алгоритма Гаусса–Ньютона к алгоритму Левенберга–Марквардта.

Поскольку вследствие своей стохастической природы ЭА всегда пытаются найти глобальный экстремум, в некоторых задачах эволюционное обучение происходит медленнее, чем традиционное, использующее, например, ускоренное обратное распространение, а в некоторых задачах — быстрее [21–23]. Объединить преимущества обоих подходов удается при гибридном обучении, когда ЭА применяются для приближенного определения области глобального экстремума в пространстве параметров, а традиционные алгоритмы — для тонкой настройки параметров, т.е. результаты эволюционной (глобальной) оптимизации используются в качестве начальных условий для процедуры локальной оптимизации [24–26]. Блок-схема такого обучения приведена на рис. 2.

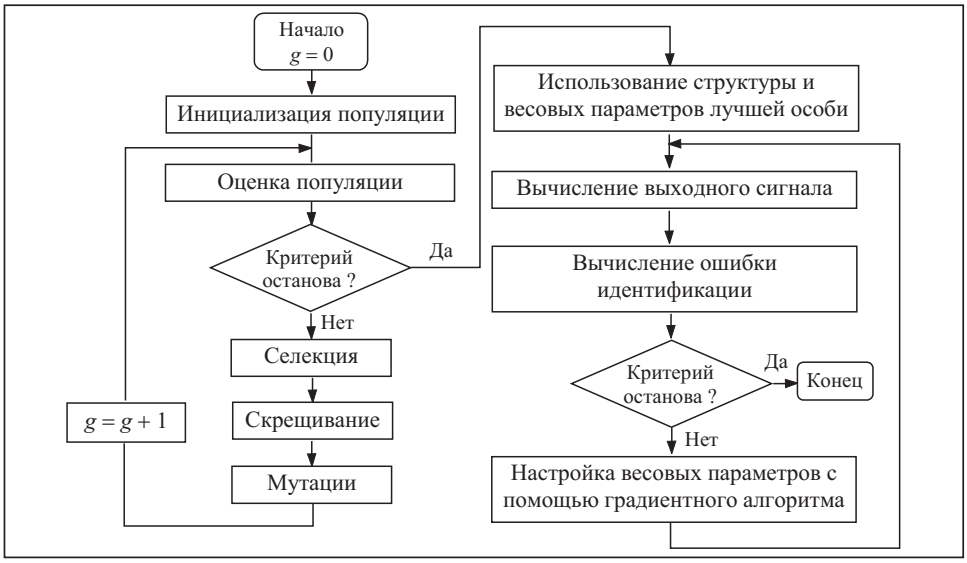

Рис. 2

Подобный подход был использован в гибридном алгоритме, предложенном в [27] для синтеза эволюционирующих сетей прямого распространения. На каждой итерации в процессе эволюции гибридный алгоритм выбирает сеть из популяции и обучает ее с помощью алгоритма ОР определенное количество эпох. Если обучение повышает производительность сети, обученная сеть совместно со значением фитнесс-функции будет возвращена в популяцию для дальнейшей эволюции. Этот механизм сохраняет приобретенные характеристики, полученные на основе обучения, что соответствует эволюции Ламарка. Такой подход использовался в работе [24] при построении сети обратного распространения для решения задач классификации. Показано, что тонкая настройка каждой сети с помощью алгоритма ускоренного ОР позволяет уменьшить пространство поиска до набора локальных оптимальных (седловых) точек, в результате чего поиск глобального оптимума становится более эффективным.

Впервые использование эффекта Болдуина для обучения эволюционирующих нейронных сетей было предложено в [28]. В этой работе случайный поиск применялся к каждой нейронной сети, полученной с помощью эволюционного поиска. Экспериментальные результаты показывают, что данный гибридный алгоритм может найти глобальный оптимум, который является недостижимым при использовании лишь эволюционного поиска.

Как отмечается в [29], эффект Болдуина в некоторых случаях может приводить к неэффективным гибридным алгоритмам, в то время как использование эволюции Ламарка является весьма эффективным в нестационарных условиях функционирования сети. ГА совместно с алгоритмом ОР использовался в [30] для одновременной настройки топологии и весов МСП. В [31] предложен нейроэволюционный метод растущей топологии, использующий генетическое кодирование топологии сети и ее весов. При этом генетические операторы применяются как для введения нового нейрона, так и для удаления старого. В работе [30] улучшенный ГА используется для обучения трехслойной полносвязной сети прямого распространения, которая после обучения может стать частично связанной. В [31] ГА совместно с алгоритмом ОР применяется для настройки как архитектуры, так и весов многослойного персептрона, а в работе [32] алгоритм ускоренного ОР используется для улучшения решения и достижения ближайшего локального минимума от найденного с помощью ГА решения. При этом популяция инициализируется особями, представляющими собой ИНС с различным числом скрытых слоев. Мутация, многоточечный кроссовер, добавление/удаление нейрона и ускоренное обучение ОР используются в качестве генетических операторов. ГА осуществляет поиск и оптимизацию архитектуры, производит первоначальную настройку весов для этой архитектуры, а также определяет параметры алгоритма обучения. Операторы ускоренного обучения ОР и процедуру удаления нейрона можно рассматривать как элементы стратегии эволюции Ламарка. В генетическом методе обратного распространения [30, 33] ГА выбирает начальные веса и изменяет количество нейронов в скрытом слое с помощью пяти генетических операторов: мутации, многоточечного кроссовера, добавления, удаления и замены нейронов скрытого слоя. Алгоритм ОР используется затем для настройки этих весов, т.е. четко разделяется глобальный и локальный поиск. Таким образом, эта стратегия не требует использования ламаркизма. Данный метод модифицирован в [34] путем интеграции ускоренного ОР как оператора обучения и реализует эволюцию Ламарка.

В [35] ЭА применялся для одновременной оптимизации структуры и весов РБС при использовании двоичного кодирования.

Эволюционирующие ИНС. При переходе от ИНС к ЭИНС для всех типов сетей используются общие эволюционные процедуры (инициализация популяции, оценка популяции, селекция, скрещивание, мутации), различие заключается лишь в способе кодирования структуры и параметров той или иной ИНС в виде хромосомы. Схема работы такого эволюционного алгоритма настройки ИНС приведена на рис. 2.

В начале работы нейроэволюционного алгоритма (НА) случайным образом инициализируется популяция P_0, состоящая из N особей (ИНС): $P_0 = \{H_1, H_2, \ldots, H_N\}$. При этом каждая особь в популяции получает свое уникальное описание, закодированное в хромосоме $H_j = \{h_{1j}, h_{2j}, \ldots, h_{Lj}\}$, которая состоит из L генов, где $h_{ij} \in [w_{\min}, w_{\max}]$ — значение i-го гена j-й хромосомы ($w_{\min}$ — минимальное и $w_{\max}$ — максимальное допустимые значения соответственно). На рис. 3 и 4 показаны примеры МП и РБС, форматы хромосом и соответствие между параметрами сетей и хромосомами (представление их параметров в хромосоме). Следует отметить, что длина хромосомы зависит от размерности идентифицируемого объекта и максимально допустимого количества нейронов. Каждая хромосома состоит из генов, в которых хранится информация о соответствующих параметрах сети. В начале хромосомы расположены гены s_1, s_2, c_1, c_2, которые содержат информацию о параметрах помехи и являются активными лишь в случае идентификации зашумленных объектов. Следующий ген кодирует информацию о смещении нейрона выходного слоя сети (b_5 для МП и w_0 для РБС). Затем следуют блоки генов, кодирующие параметры соответствующих нейронов скрытого слоя. Первый ген каждого такого блока (1/0) определяет, присутствует ли соответствующий нейрон в структуре сети, т.е. участвует ли он в вычислении выходной реакции сети на поступивший входной сигнал. Так как в качестве $f(\cdot)$ в МП обычно используют сигмоидальные АФ, например вида

$$f(x) = (1 + e^{-\alpha x})^{-1}, \tag{8}$$

то следующий ген в его хромосоме (α) определяет форму АФ данного нейрона.

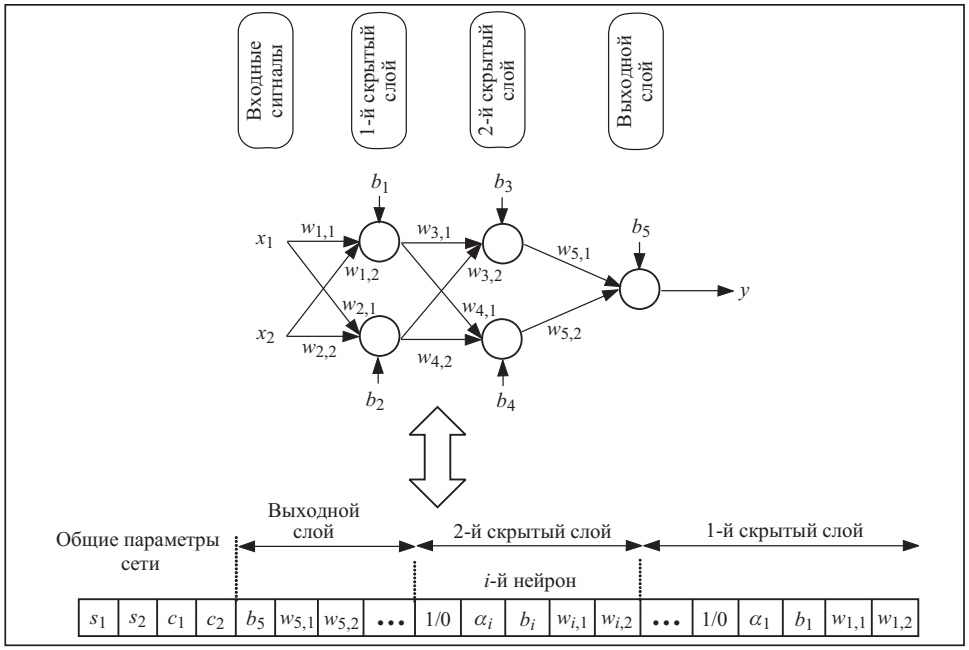

Рис. 3

В РБС в качестве БФ могут быть использованы, например, следующие функции:

$$\Phi_i(x) = \exp\left\{-\frac{\| x - \mu_i \|^2}{\sigma_i^2}\right\}, \tag{9}$$

$$\Phi_i(x) = \left(1 - \frac{(x-\mu_i)^2}{\sigma_i^2}\right)\exp\left\{-\frac{(x-\mu_i)^2}{\sigma_i^2}\right\}, \tag{10}$$

$$\Phi_i(x) = \exp\left\{-\frac{|x-\mu_i|}{\sigma_i}\right\}, \tag{11}$$

$$\Phi_i(x) = \frac{2(x-\mu_i)}{\sigma_i}\exp\left\{-\frac{(x-\mu_i)^2}{\sigma_i^2}\right\}, \tag{12}$$

где μ_i, σ_i — центры и радиусы базисных функций соответственно; $||\cdot||$ — евклидова норма.

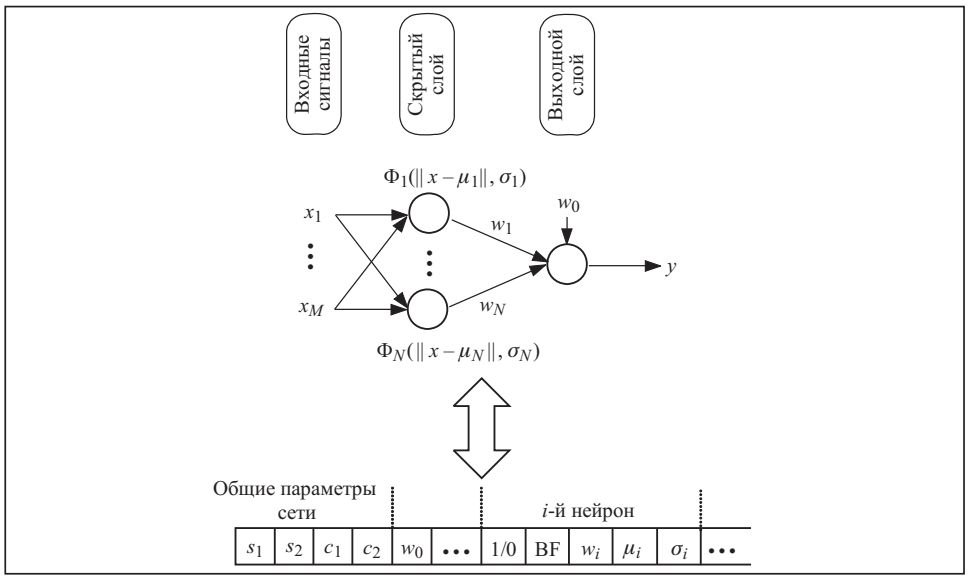

Рис. 4

В связи с большим количеством БФ, которые могут применяться в РБС, в ее хромосому вводится специальный ген BF, отвечающий за кодирование вида используемой функции. Поскольку в МП и РБС применяются нелинейные функции преобразования с различным количеством параметров, то и кодирующие их хромосомы несколько отличаются [36–39].

Затем в хромосоме следует группа генов, кодирующая непосредственно весовые параметры соответствующего нейрона. На этапе инициализации всем этим параметрам с помощью датчика случайных чисел присваивают начальные значения, находящиеся в некотором допустимом диапазоне.

После формирования начальной популяции производится оценка приспособленности каждой входящей в нее особи, определяемой некоторой функцией приспособленности (фитнесс-функцией). При идентификации параметров нелинейных систем с помощью ГА в них традиционно используется в качестве фитнесс-функции квадратичный функционал

$$\rho(e(i,\theta)) = 0.5e^2(i,\theta). \tag{13}$$

При наличии выбросов в обучающей выборке применение данной функции приспособленности приводит к тому, что решения, получаемые как алгоритмами обучения, которые используют вычисления производных, так и с помощью ЭА, являются неудовлетворительными [40].

Выбор функции потерь, отличной от квадратичной, позволяет обеспечить робастность оценок, т.е. их работоспособность практически для всех распределений помех. Так, для ε-засоренных вероятностных распределений, описываемых моделью Тьюки–Хьюбера [41]

$$p(x) = (1-\varepsilon)p_0(x) + \varepsilon q(x), \tag{14}$$

где $p_0(x)$ — плотность соответствующего основного распределения $N(0, \sigma_1^2)$; $q(x)$ — плотность засоряющего (произвольного) распределения $N(0, \sigma_2^2)$, $\sigma_1^2 << \sigma_2^2$; $\varepsilon \in [0,1]$ — параметр, характеризующий степень засорения основного распределения, на основе минимизации функционала

$$\rho(e(k)) = \begin{cases} \dfrac{e^2(k)}{\tau}, & |e(k)| \le \tau, \\ \tau |e(k)| - \dfrac{e^2}{2}, & |e(k)| > \tau, \end{cases} \tag{15}$$

были получены нелинейные оценки огрубленного или робастного метода максимального правдоподобия [41–43].

В настоящее время существует достаточно много функций $\rho(e(i))$, обеспечивающих получение робастных оценок, например [44–51]

$$\rho[e(k)] = |e(k)|^{\lambda}, \tag{16}$$

$$\rho[e(k)] = \ln \cos h(e(k)), \tag{17}$$

$$\rho[e(k)] = \sqrt{1 + e^2(k)} - 1, \tag{18}$$

$$\rho[e(k)] = \begin{cases} 1 - \cos\left(\dfrac{\pi e(k)}{\tau}\right), & |e(k)| \le \tau, \\ 2, & |e(k)| > \tau, \end{cases} \tag{19}$$

$$\rho[e(k)] = \tau^2 \left(\frac{|e(k)|}{\tau} - \ln\left(1 + \frac{|e(k)|}{\tau}\right)\right). \tag{20}$$

Необходимо отметить, что при использовании функционалов типа (15), (19), (20) возникает проблема выбора (оценивания) параметра τ, который также может быть внесен в качестве дополнительного гена в хромосому и оценен с помощью ГА.

При использовании модели засорения (14) можно с помощью ГА оценить величины s_1^2 и s_2^2 (как дополнительные параметры сети, см. рис. 3, 4), являющиеся оценками σ_1^2 и σ_2^2, которые необходимо учитывать при вычислении значения следующей фитнесс-функции:

$$f_i(x_j) = \frac{1}{M} \sum_{j=1}^{M} \hat{e}_j^2(k), \tag{21}$$

где $\hat{e}_j^2(k) = \begin{cases} \dfrac{(y_j^*(x_j) - \hat{y}_j(x_j))^2}{s_1^2(k)}, & \text{если } |y_j^*(x_j) - \hat{y}_j(x_j)| \le 3s_1(k), \\ \dfrac{(y_j^*(x_j) - \hat{y}_j(x_j))^2}{s_2^2(k)} & \text{в противном случае.} \end{cases}$

Следующим шагом является отбор особей, хромосомы которых будут принимать участие в формировании нового поколения, и последующее скрещивание этих особей. Задача оператора скрещивания (кроссовера) заключается в передаче генетической информации от родительских особей к их потомкам. По завершении работы данного оператора любой ген любой особи в новой популяции может мутировать, т.е. изменить свое значение.

Поскольку в хромосоме используется гибридная кодировка, при мутации необходимо осуществлять различные операции для различных методов кодирования. Двоичное кодирование гена активизации нейрона должно использовать инверсную мутацию, при которой некоторые биты хромосомы могут изменить свое значение с единицы на нуль или наоборот. При вещественном кодировании параметров БФ и весовых параметров возможно использование различных типов мутации.

Так, адаптивная мутация Гаусса, рассмотренная в работе [14], вначале осуществляет коррекцию шага мутации, а затем производит непосредственно изменение значения мутирующего в хромосоме гена. Эту процедуру можно описать следующим образом:

$$\nu_j(k) = \nu_j(k-1)\exp[lN(0,1) + l'N(0,1)], \tag{22}$$

$$h'_{ij} = h_{ij} + \nu_j(k)N(0,1), \tag{23}$$

где $N(0,1)$ — случайная величина, распределенная по нормальному закону с нулевым математическим ожиданием и единичной дисперсией; l и l' — параметры, для которых в работе [15] были получены следующие оптимальные значения: $l = (2N)^{-0.5}$ и $l' = (2\sqrt{N})^{-0.5}$ соответственно.

В качестве альтернативы гауссовской мутации в настоящее время все чаще используется адаптивная мутация Лапласа, ставшая особенно популярной в задачах минимизации многоэкстремальных функций. Настройка шага мутации осуществляется по правилу (22), т.е. как и при использовании адаптивной мутации Гаусса, а для мутации гена используется соотношение

$$h'_{ij} = h_{ij} + \nu_j(k)L(\alpha), \tag{24}$$

где L — случайная величина, распределенная по закону Лапласа с параметром α, плотность вероятности которой имеет вид

$$f(x) = \begin{cases} \dfrac{1}{2}e^{\alpha x}, & \text{если } x \le 0, \\ 1 - \dfrac{1}{2}e^{-\alpha x}, & \text{если } x > 0. \end{cases} \tag{25}$$

Наконец, при использовании уменьшающейся мутации Коши изменение шага мутации происходит следующим образом:

$$\nu_j(k) = \gamma\nu_j(k-1), \tag{26}$$

где $\gamma \in [0,1]$ — некоторая константа (в работе [14] предлагается использовать $\gamma = 0.95$).

Изменение значения гена происходит по правилу

$$h'_{ij} = h_{ij} + \nu_j(k)C(k,t), \tag{27}$$

где $C(k,t)$ — случайная величина, распределенная по закону Коши

$$f(x) = \frac{1}{\pi}\left(\frac{\tau}{x^2 + t^2}\right) \tag{28}$$

с параметром t.

По окончании формирования нового поколения осуществляется его оценка. В случае выполнения критерия останова ГА начинает работу градиентный алгоритм настройки весовых параметров, который осуществляет «тонкую» настройку наилучшей сети, отобранной с помощью ГА. В качестве такого алгоритма могут быть использованы модификация алгоритма ОР, алгоритм Гаусса–Ньютона, алгоритм Левенберга–Марквардта и т.д.

МОДЕЛИРОВАНИЕ

Эксперимент 1. Решается задача идентификации нестационарного динамического объекта, описываемого уравнением

$$y(k) = 0.725\beta(k)\sin\left(\frac{16u(k-1) + 8y(k-1)}{\beta(k)(3 + 4u^2(k-1) + 4y^2(k-1))}\right) + 0.2u(k-1) + 0.2y(k-1), \tag{29}$$

где $\beta(k) = \begin{cases} 1.0, & 1 \le k \le 500, \\ 0.9, & 501 \le k \le 1000, \\ 0.8, & 1001 \le k \le 1500, \\ 0.7, & 1501 \le k \le 2000, \\ 0.6, & 2001 \le k \le 2500, \\ 0.5, & 2501 \le k \le 3000, \\ 0.4, & 3001 \le k \le 3500, \\ 0.3, & 3500 \le k \le 4000. \end{cases}$

Начальная популяция состоит из 128 особей (РБС). Максимальное количество нейронов в скрытом слое ограничено до 15, на каждой итерации алгоритма каждой особи предъявлялось $M = 2500$ обучающих пар. На рис. 5, *а* приведен график изменения значения фитнесс-функции особи-победителя, вычисляемой по формуле

$$f_i(x_j) = \frac{1}{M} \sum_{j=1}^{M} e_j^2(x, k). \tag{30}$$

Пунктирной линией на графике обозначено целевое значение фитнесс-функции, которое в данном эксперименте принято равным единице. В промежутках между пиковыми значениями фитнесс-функции, являющимися следствием изменения структуры объекта, наблюдаются резкие ее падения, возникающие после применения алгоритма Левенберга–Марквардта для уточнения весовых параметров РБС. С уменьшением параметра $\beta(k)$ возрастает сложность объекта (29), приведшая к невозможности достижения заданного желаемого значения фитнесс-функции за 500 шагов.

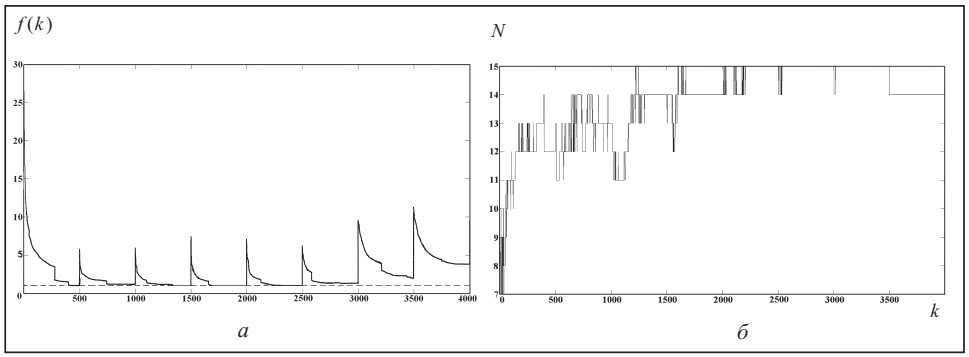

Рис. 5

На рис. 5, *б* показан график изменения количества нейронов в скрытом слое особи-победителя, т.е. имеющей наименьшее значение фитнесс-функции (30) в популяции. Здесь изменение коэффициента $\beta(k)$ на каждых 500-х шагах приводит также к необходимости изменения структуры сети. В табл. 1 представлены данные о количестве нейронов с БФ вида (9) и (10) на итерации, предшествующей изменению структуры объекта.

Таблица 1

БФ	Количество нейронов в сети при k							
	500	1000	1500	2000	2500	3000	3500	4000
Гаусса (9)	8	5	6	10	12	7	9	10
Вейвлет (10)	4	8	8	4	3	8	6	4

Эксперимент 2. С использованием эволюционирующих РБС и МП решается задача идентификации сильно зашумленного объекта (29)

$$y(k)=0.725\beta(k)\sin\left(\frac{16u(k-1)+8y(k-1)}{\beta(k)(3+4u^2(k-1)+4y^2(k-1))}\right)+0.2u(k-1)+0.2y(k-1)+\xi(k),$$

где $\beta(k)=1$, $\xi(k)=(1-\varepsilon)q_1(k)+\varepsilon q_2(k)$ — засоренная помеха ($q_1(k)$, $q_2(k)$ — нормально распределенные помехи с дисперсиями σ_1^2 и σ_2^2 соответственно).

В данном эксперименте менялись во времени параметры помехи. Так, с первого по 500-й шаг $\sigma_2=6$, а начиная с шага 501 и заканчивая 1000-м имеем $\sigma_2=12$. Значение σ_1 не изменялось во времени и оставалось равным 0.6. Результаты идентификации для РБС представлены на рис. 6, а для МП — на рис. 7. Значение фитнесс-функции в данном эксперименте вычислялось с помощью уравнения (21), в котором учитывались оценки параметров помехи σ_1 и σ_2.

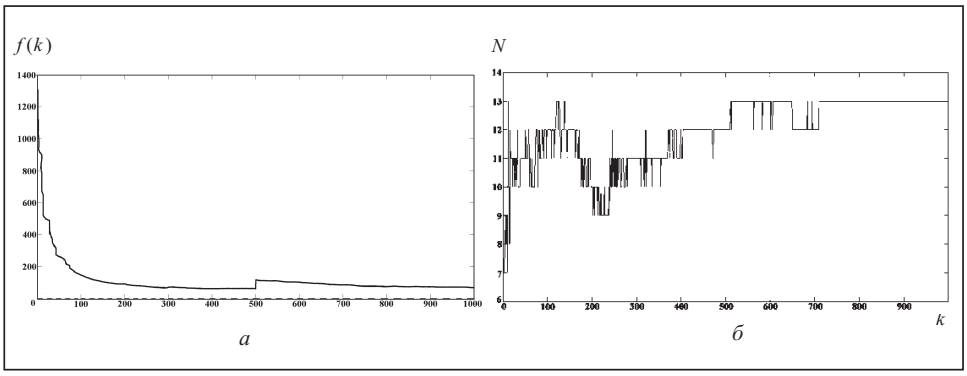

Рис. 6

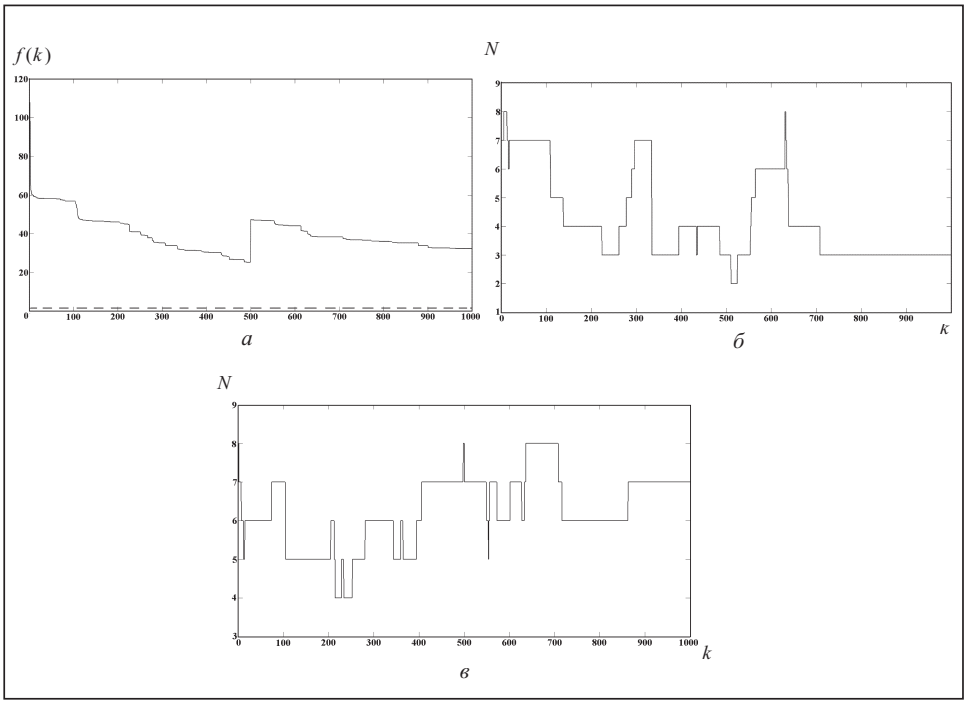

Рис. 7

Эксперимент 3. Проводится идентификация объекта с изменяющейся во времени структурой, описываемого следующими уравнениями:

$$y(k)=\begin{cases}\dfrac{15u(k-1)\,y(k-1)}{2+50[u(k-1)]^2}+0.5u(k-1)-0.25\,y(k-1)+0.1 & \text{при } k=\overline{1,500},\\ \dfrac{\sin(\pi u(k-1)\,y(k-1))+2u(k-1)}{3} & \text{при } k=\overline{501,1000}.\end{cases}$$

Результаты исследования РБС и МП показаны на рис. 8 и рис. 9 соответственно.

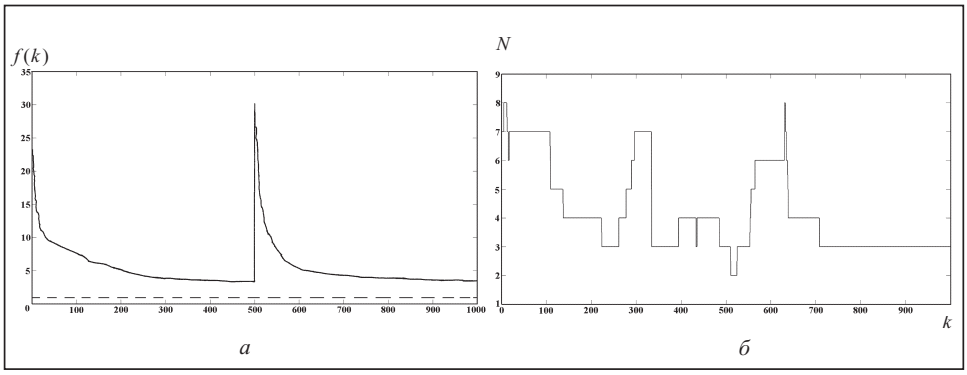

Рис. 8

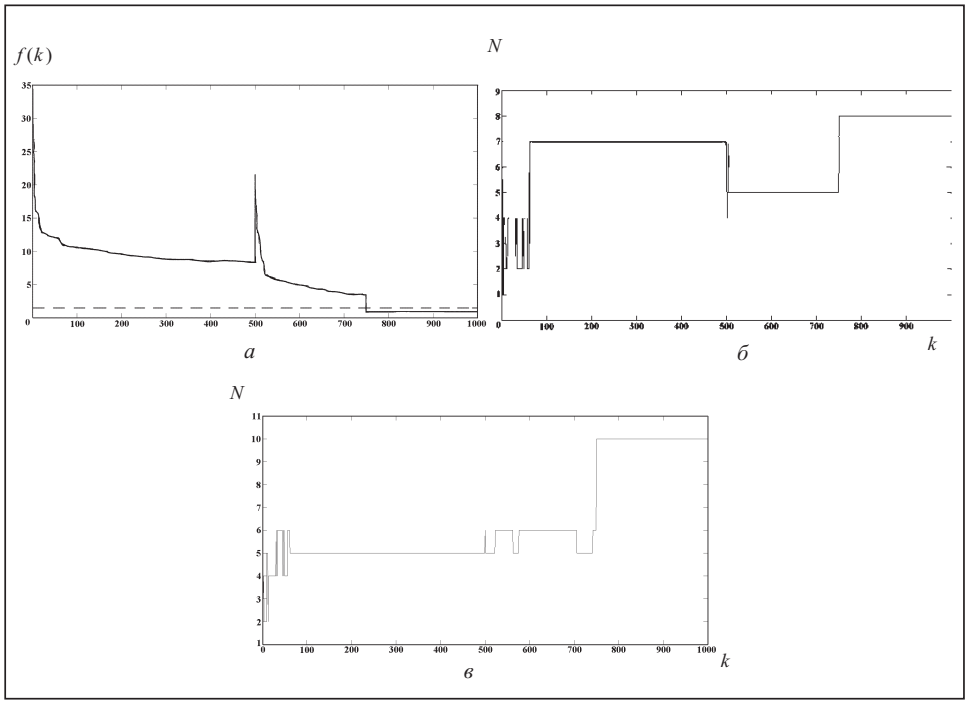

Рис. 9

Как следует из результатов моделирования, применение развиваемого подхода является достаточно эффективным при идентификации объектов, структура которых существенно изменяется во времени.

ЗАКЛЮЧЕНИЕ

Использование нейроэволюционного подхода, сочетающего ИНС и эволюционные вычисления, для решения задачи идентификации нелинейных динамических объектов является достаточно универсальным и оказывается весьма эффективным в нестационарных условиях. Такой подход позволяет оперативно корректировать структуру и параметры сетей, а также алгоритмы их обучения

в зависимости от изменения свойств как исследуемого объекта, так и окружающей среды, а выбор подходящей фитнесс-функции обеспечивает робастность получаемых моделей.

Результаты моделирования свидетельствуют о том, что существенного повышения точности синтезируемых нейросетевых моделей в ряде случаев можно достичь гибридным обучением, использующим эволюционные вычисления на начальном этапе при выборе структуры сетей и «грубой» настройке их параметров и алгоритмы градиентного типа — для «тонкой» настройки параметров отобранных сетей.

Хотя при применении эволюционных алгоритмов возникает ряд проблем, связанных с выбором способов кодирования и генетических операторов, развитие нейроэволюционного подхода представляется целесообразным и перспективным в связи с возможностью его реализации на параллельных и распределенных вычислительных средствах, получающих в настоящее время все более широкое распространение.

СПИСОК ЛИТЕРАТУРЫ

1. Fahlman S. Fast learning variations on back-propagation: an empirical study // Proc. of the 1988 Connectionist Models Summers School / D.S. Touretzky, G.E. Hinton, T. Sejnowski (Eds.). — Pittsburgh: Morgan Kaufmann, 1988. — P. 38–51.
2. Fahlman S., Lebiere C. The cascade-correlation learning architecture // Adv. Neur. Inform. Proces. Syst. / D.S. Touretzky (ed.). — Los Altos (CA): Morgan Kaufmann, 1990. — **2**. — P. 524–532.
3. Riley M.J., Thompson Ch.P., Jenkins K.W. Improving the performance of cascade correlation neural networks on multimodal functions // Proc. of the World Congress of Engineering. — London, 2010. — **3**. — 7 p.
4. Yao X. Evolving artificial neural networks // Proc. of the IEEE. — 1999. — **87**, N 9. — P. 1423–1447.
5. Yao X., Lin Y. A new evolutionary system for evolving artificial neural networks // IEEE Trans. on Neural Networks. — 1997. — **3**, N 3. — P. 694–713.
6. Floreano D., Durr P., Mattiussi C. Neuroevolution: from architecture to learning // Evol. Intel. — 2008. — **1**. — P. 47–62.
7. Fogel D.B. An introduction to simulated evolutionary optimization // IEEE Trans. on Neural Networks. — 1994. — **5**, N 1. — P. 3–14.
8. Back T., Hammel V., Schwefel H.-D. Evolutionary computation: Comments on the history and current state // IEEE Trans. on Evol. Comput. — 1997. — **1**, N 4. — P. 3–17.
9. Zhang B., Wu Y., Lu J., Du K.-L. Evolutionary computation and its applications in neural and fuzzy systems // Appl. Computational Intelligence and Soft Comput., 2011. — 20 p.
10. Whitley D. An overview of evolutionary algorithm: practical issues and common pitfalls // Inform. and Software Technology. — 2001. — **43**. — P. 817–831.
11. Holland J. Adaptation in natural and artificial systems. — 2nd edn. — Cambridge: MIT Press, 1992. — 228 p.
12. Whitley D., Gordon Y.S., Mathias K. Lamarkian evolution, The Baldwin effect and function optimization // Proc. of the Parallel Problem Solving from Nature. — Berlin: Springer-Verlag, 1994. — P. 6–15.
13. Girand-Carrier Ch. Unifying learning with evolution through Baldwinian evolution and Lamarkism: A case study // Proc. Symp. on Comput. Intel. and Learning (CoIL-2000). — Chios (Greece): MIT GmbH, 2000. — P. 36–41.
14. Schwefel H.P. Numerical optimization of computer models. — Chichister; N.Y.: John Wiley & Sons, 1981. — 389 p.
15. Bäck T., Schwefel H.-P. An overview of evolution algorithms for parameter optimization // Evol. Comput. — 1993. — **1**, N 1. — P. 1–23.
16. Koza J.R., Rice J.P. Genetic generation of both the weights and architecture for neural networks // Proc. IEEE Int. Joint Conf. on Neural Networks. — Seattle (WA): IEEE Press, 1991. — **2**. — P. 71–76.
17. Lefort V., Knibbe K., Beslon G., Fevrel J. Simultaneous optimization of weights and structure of an RBF neural network // Artif. Evol. — Berlin: Springer-Verlag, 2006. — P. 49–60.

18. Ninton G.E., Nowlan S.J. How learning can guide evolution // Complex Systems. — 1987. — **1**. — P. 495–502.
19. Chalmers D.J. The evolution of learning: an experiment in genetic connectionism // Proc. of the 1990 Connectionist Models Summer School / D.S. Touretzky, J.L. Elman, G.E. Hinton (Eds.). — San Mateo (CA): Morgan Kaufmann, 1990. — P. 81–90.
20. Baxter J. The evolution of learning algorithms for artificial neural networks // Complex Systems / D. Green, T. Bosso-Majer (Eds.). — Amsterdam: IOS Press, 1992. — P. 313–326.
21. Kinnebrock W. Accelerating the standart backpropagation method using a genetic approach // Neurocomputation. — 1994. — **6**, N 5–6. — P. 583–588.
22. Ku K.W.C., Mak M.W., Siu W.C. Approaches of combining local and evolutionary search for training neural networks: A review and some new results // Adv. Evol. Comput.: Theory and Appl. — Berlin: Springer-Verlag, 2003. — P. 615–641.
23. Ding S., Xu X., Zhu H. Studies of optimization algorithms for some artificial neural networks based on genetic algorithm (GA) // J. Comput. — 2011. — **6**, N 5. — P. 939–946.
24. Braun H., Zagorski P. ENZO-M — a hybrid approach for optimazing neural networks by evolution and learning // Proc. of the Parallel Problem Solving from Nature. — Berlin: Springer-Verlag, 1994. — P. 440–451.
25. Guo L., Huang D.-S., Zhao W. Combining genetic optimization with hybrid learning algorithm for radial basis function neural networks // Electron. Lett. — 2003. — **39**, N 22. — P. 1600–1601.
26. Harpham C., Dawson C.W., Brown M.R. A review of genetic algorithms applied to training radial basis function networks // Neural Comput. and Appl. — 2004. — **13**, N 3. — P. 193–201.
27. Chun Lu, Bingxue Shi, Lu Chen. Hybrid BP-GA for multilayer feedforward neural networks // Electronics, Circuits and Systems: The 7th IEEE Intern. Conf. — 2000. — **2**. — P. 958–961.
28. Hinton G.E., Nowlan S.J. How learning can guide evolution // Complex Systems. — 1987. — **1**. — P. 495–502.
29. French R.M., Messinger A. Genes, phenes and the Baldwin effect: learning and evolution in a simulated population // Artif. Life. — 1994. — **4**. — P. 277–282.
30. G-Prop: global optimization of multilayer perceptrons using GAs / P.A. Castillo, J.J. Merelo, V. Rivas et al. // Neurocomputing. — 2000. — **35**. — P. 149–163.
31. Stanley K.O., Miikkulainen R. Evolving neural networks through augmenting topologies // Evol. Comput. — 2002. — **10**, N 2. — P. 99–127.
32. Leung F.H.F., Lam H.K., Ling S.H., Tam P.K.S. Tuning of the structure and parameters of a neural network using an improved genetic algorithm // IEEE Trans. on Neural Networks. — 2003. — **14**, N 1. — P. 79–88.
33. Montana D.J., Davis L. Training feedforward networks using genetic algorithms // Proc. of the 11th Int. Joint Conf. on Artificial Intelligence. — Detroit (Mich.): Morgan Kaufmann, 1989. — P. 762–767.
34. Neruda R., Slusnu S. Parameter genetic learning of perceptron networks // Proc. of the 10th WSEAS Int. Conf. of SYSTEMS. — Athens (Greece): Stevens Points (Wis., USA), 2006. — P. 92–97.
35. Multiobjective evolutionary optimization of the size, shape, and position parameters of radial basis function networks for the function approximation / J. Gonzalez, I. Rojas, J. Ortega et al. // IEEE Trans. on Neural Networks. — 2003. — **14**, N 6. — P. 1478–1495.
36. Evolving multilayer perceptrons / P.A. Castillo, J. Carpio, J.J. Merelo et al. // Neur. Proces. Letters. — 2000. — **12**. — P. 115–127.
37. Buchtala O., Klimek M., Sick B. Evolutionary optimization of radial basis function classifiers for data mining applications // IEEE Trans. on Systems, Man, and Cybernetics, Part B. — 2005. — P. 928–947.
38. Maillard E.P., Gueriot D. RBF neural network, basis functions and genetic algorithm // Proc. Int. Conf. Neural Networks. — Houston (Tex), 1997. — **4**. — P. 2187–2192.
39. Chen S., Wu Y., Luk B.L. Combined genetic algorithm optimization and regularized orthogonal least squares learning for radial basis function networks // IEEE Trans. on Neural Networks. — 1999. — **10**, N 5. — P. 1239–1243.
40. Jin Y., Branke J. Evolutionary optimization in uncertain environments — A survey // IEEE Trans. Evol. Comput. — 2005. — **5**, N 3. — P. 303–317.

41. Хьюбер П. Робастность в статистике. — М.: Мир, 1984. — 304 с.
42. Цыпкин Я.З. Основы информационной теории идентификации. — М.: Наука, 1984. — 320 с.
43. Цыпкин Я.З., Поляк Б.Т. Огрубленный метод максимального правдоподобия // Динамика систем. — Горький: Изд-во Горьк. ун-та, 1977. — Вып. 12. — С. 22–46.
44. Lee C.C., Chung P.C., Tsai J.R., Chang C.I. Robust radial basis function networks // IEEE Trans. on Syst., Man, and Cybernetics. — 1999. — **29**, N 6. – P. 674–684.
45. Руденко О.Г., Бессонов А.А. Робастное обучение радиально-базисных сетей // Кибернетика и системный анализ. — 2011. — № 6. — С. 38–46.
46. Руденко О.Г., Бессонов А.А. Робастное обучение вейвлет-нейросетей // Междунар. науч.-техн. журн. «Проблемы управления и информатики». — 2010. — № 5. — С. 66–79.
47. Руденко О.Г., Бессонов А.А. М-обучение радиально-базисных сетей с использованием асимметричных функций влияния // Там же. — 2012. — № 1. — С. 79–93.
48. Rudenko O., Bezsonov O. Function approximation using robust radial basis function networks // J. Intel. Learn. Syst. and Appl. — 2011. — N 3. — P. 17–25.
49. Бессонов А.А., Руденко С.О. Идентификация нелинейных нестационарных объектов с помощью эволюционного многослойного персептрона // Вестн. ХНТУ. — 2012. — № 1 (44). — С. 130–135.
50. Бессонов А.А. Обучение радиально-базисных сетей с помощью генетических алгоритмов с адаптивной мутацией // Системи обробки інформації. — 2012. — № 3(101). — С. 177–180.
51. Hsieh J.-G., Lin Y.-L., Jeng J.-H. Preliminary study on Wilcoxon learning machines // IEEE Trans. on Neural Networks. — 2008. — **19**, N 2. — P. 201–211.

Поступила 05.09.2012