УДК 683.519

ИССЛЕДОВАНИЕ НЕЧЕТКИХ НЕЙРОННЫХ СЕТЕЙ В ЗАДАЧАХ МАКРОЭКОНОМИЧЕСКОГО ПРОГНОЗИРОВАНИЯ

Ю.П. ЗАЙЧЕНКО, СЕВАЕЕ ФАТМА, К.М. ТИТАРЕНКО, Н.В. ТИТАРЕНКО

Рассмотрены нечеткие нейронные сети, использующие логический вывод Мамдани и Цукамото. Описаны алгоритмы обучения сетей и приведены результаты их исследования в задачах макроэкономического прогнозирования.

ВВЕДЕНИЕ

Нестабильная экономическая ситуация в Украине вызывает оправданное недоверие к прогнозам вообще и к прогнозам, выполненным классическими методами, в частности. Это обусловлено сложными нелинейными зависимостями между показателями и быстрой их динамикой. Возникает вопрос, применимы ли к рыночной экономике, особенно в переходный период, классические методы прогнозирования.

Для моделирования подобных зависимостей широко применяются нейронные сети (НС), способные обучаться на зашумленных и неполных данных и осуществлять прогноз (как правило, кратковременный). Однако в обычных НС невозможно использовать априорную информацию о процессе, которая традиционно задается в форме правил «ЕСЛИ-ТО» в пространстве лингвистических переменных. Именно такую информацию можно задать в НС, основанных на нечеткой логике (нечетких нейронных сетях — ННС), сокращая число необходимых циклов обучения и повышая точность прогноза.

Широко распространены такие классы ННС, как нечеткие контроллеры Мамдани и Цукамото. Цель настоящей статьи — исследование возможностей и эффективности применения этих контроллеров в задачах прогнозирования в макроэкономике.

Контроллер Мамдани исследуется при треугольных и гауссовых функциях принадлежности, а Цукамото — при нелинейных функциях принадлежности (ФП).

СИСТЕМЫ НЕЧЕТКОГО ЛОГИЧЕСКОГО ВЫВОДА

Аппарат нечеткой логики и нечетких множеств

Наверное, наиболее поразительным свойством человеческого интеллекта является способность принимать правильные решения, имея неполную и

нечеткую информацию. Построение моделей приближенных рассуждений человека и использование их в интеллектуальных компьютерных системах — одно из самых перспективных направлений развития современной теории принятия решений.

Значительный вклад в это направление внес Л. Заде (L. Zadeh). Введя понятие лингвистической переменной и допустив, что в качестве ее значений (термов) выступают нечеткие множества, Заде предложил аппарат для описания процессов интеллектуальной деятельности, содержащий нечеткость и неопределенность выражений. Это разрешило создать фундамент теории нечетких множеств и нечеткой логики, а также предпосылки для внедрения методов нечеткого управления в инженерную практику.

Математическая теория нечетких множеств позволяет описывать нечеткие понятия и знания, оперировать ими и делать нечеткие заключения. Нечеткое управление оказывается особенно полезным, когда исследуемые процессы слишком сложны для анализа с помощью общепринятых методов или доступные источники информации интерпретируются лишь качественно, неточно или неопределенно. Нечеткая логика, предоставляющая эффективные средства отображения неопределенностей и неточностей реального мира и на которой основано нечеткое управление, ближе к человеческому мышлению и естественным языкам, чем традиционные логические системы.

НЕЧЕТКИЙ ЛОГИЧЕСКИЙ ВЫВОД

Используемый в разных экспертных и управляющих системах механизм нечетких выводов в своей основе имеет базу знаний, формируемую специалистами предметной области в виде совокупности нечетких предикатных правил вида

$$\Pi_1 : \text{если } x \text{ есть } A_1, \text{ то } y \text{ есть } B_1,$$

$$\Pi_2 : \text{если } x \text{ есть } A_2, \text{ то } y \text{ есть } B_2,$$

$$\cdots$$

$$\Pi_n : \text{если } x \text{ есть } A_n, \text{ то } y \text{ есть } B_n,$$

где x — входная переменная (имя для известных значений данных); y — переменная вывода (имя для значения данных, которое будет вычислено); A и B — функции принадлежности, определенные соответственно на x и y.

Поясним более детально. Знание эксперта $A \to B$ отражает нечеткое причинное отношение предпосылки и заключения, поэтому его можно назвать нечетким отношением и обозначить как

$$R = A \to B,$$

где «$\to$» называют нечеткой импликацией.

Отношение R можно рассматривать как нечеткое подмножество прямого произведения $X \times Y$ полного множества предпосылок X и выводов Y. Таким образом, процесс получения (нечеткого) результата вывода B' с

использованием данного наблюдения A' и знания $A \to B$ можно представить в виде композиционного правила нечеткий «modus ponens»

$$B' = A' \bullet R = A' \bullet (A \to B),$$

где «$\bullet$» — операция свертки.

Как операцию композиции, так и операцию импликации в алгебре нечетких множеств можно реализовывать по-разному (при этом будет отличаться и получаемый результат), но в любом случае общий логический вывод осуществляется за следующие четыре этапа.

1. *Введение нечеткости (фаззификация, fuzzification).* ФП, определенные на входных переменных, применяются к их фактическим значениям для установления степени истинности каждой предпосылки каждого правила.

2. *Логический вывод.* Вычисленное значение истинности для предпосылок каждого правила применяется к заключениям каждого правила. Это приводит к одному нечеткому подмножеству, которое будет назначено каждой переменной вывода для каждого правила. В качестве правил логического вывода обычно используются только операции min (МИНИМУМ) или prod (УМНОЖЕНИЕ). В логическом выводе МИНИМУМА функция принадлежности вывода «отсекается» по высоте, соответствующей вычисленной степени истинности предпосылки правила (нечеткая логика «И»). В логическом выводе УМНОЖЕНИЯ функция принадлежности вывода масштабируется с помощью вычисленной степени истинности предпосылки правила.

3. *Композиция.* Все нечеткие подмножества, назначенные к каждой переменной вывода (во всех правилах), объединяются вместе, чтобы сформировать одно нечеткое подмножество для всех переменных вывода. При подобном объединении обычно используются операции max (МАКСИМУМ) или sum (СУММА). При композиции МАКСИМУМА комбинированный вывод нечеткого подмножества конструируется как поточечний максимум по всем нечетким подмножествам (нечеткая логика «ИЛИ»). При композиции СУММЫ комбинированный вывод нечеткого подмножества формируется как поточечная сумма по всем нечетким подмножествам, назначенным переменной вывода правилами логического вывода.

4. *Приведение к четкости (дефаззификация, defuzzification).* Используется, если нужно преобразовать нечеткий набор выводов в четкое число. Существует значительное количество методов приведения к четкости, некоторые из которых рассмотрим ниже.

Пример. Пусть система описывается следующими нечеткими правилами:

Π_1 : если x есть A, то w есть D,

Π_2 : если y есть B, то w есть E,

Π_3 : если z есть C, то w есть F,

где x, y и z — имена входных переменных; w — имя переменной вывода, а A, B, C, D, E, F — заданные функции принадлежности (треугольной формы).

Рассмотрим процедуру получения логического вывода (рис. 1). Предполагается, что заданы конкретные (четкие) значения входных переменных x_0, y_0, z_0.

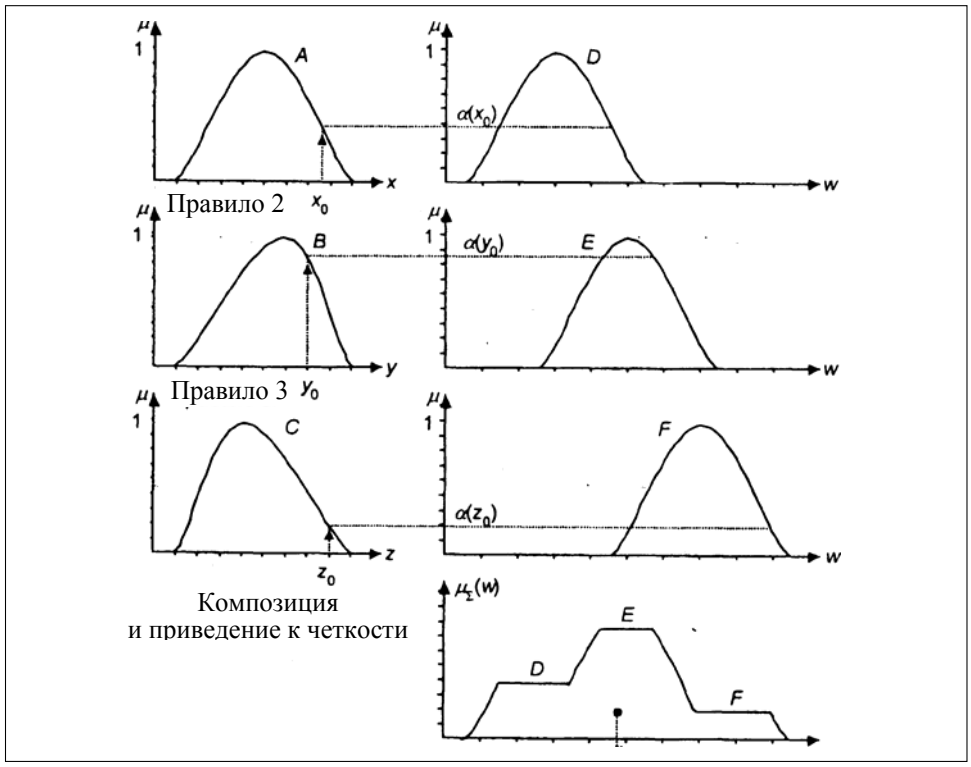

Рис. 1. Процедура получения логического вывода

На первом этапе по данным значениям, исходя из функций принадлежности A, B, C, находятся степени истинности $\alpha(x_0), \alpha(y_0)$ и $\alpha(z_0)$ для предпосылок каждого из трех приведенных правил.

На втором — происходит «отсечение» функций принадлежности выводов правил (D, E, F) на уровнях $\alpha(x_0)$, $\alpha(y_0)$ и $\alpha(z_0)$.

На третьем — рассматриваются функции принадлежности, усеченные на предыдущем этапе, и производится их объединение с использованием операции max, в результате чего получается комбинированное нечеткое подмножество, описываемое функцией принадлежности $\mu_\Sigma(w)$ и соответствующее логическому выводу для выходной переменной w.

На четвертом — находится, при необходимости, четкое значение выходной переменной, например, с применением центроидного метода: четкое значение выходной переменной определяется как центр тяжести для кривой $\mu_\Sigma(w)$:

$$w_0 = \frac{\int\limits_\Omega w\,\mu_\Sigma(w)dw}{\int\limits_\Omega \mu_\Sigma(w)dw}.$$

Рассмотрим следующие наиболее употребительные модификации алгоритма нечеткого вывода, считая, для простоты, что базу знаний организуют два нечетких правила вида

Π_1 : если x есть A_1 и y есть B_1, то z есть C_1,

Π_2 : если x есть A_2 и y есть B_2, то z есть C_2,

где x и y — имена входных переменных; z — имя переменной вывода; A_1, B_1, C_1, A_2, B_2, C_2 — некоторые заданные функции принадлежности. При этом четкое значение z_0 необходимо определить на основе приведенной информации и четких значений x_0 и y_0.

АЛГОРИТМ МАМДАНИ

Данный алгоритм соответствует рассмотренному примеру и рис. 1. В этой ситуации математически он может быть описан следующим образом.

1. Введение нечеткости. Находятся степени истинности для предпосылок каждого правила

$$A_1(x_0), A_2(x_0), B_1(y_0), B_2(y_0).$$

2. Логический вывод. Находятся уровни «отсечения» для предпосылок каждого из правил (с использованием операции МИНИМУМ)

$$\alpha_1 = A_1(x_0) \wedge B_1(y_0),$$
$$\alpha_2 = A_2(x_0) \wedge B_2(y_0),$$

где «$\wedge$» обозначена операция логического минимума (min). Затем находятся «усеченные» функции принадлежности

$$C_1' = (\alpha_1 \wedge C_1(z)),$$
$$C_2' = (\alpha_2 \wedge C_2(z)).$$

3. Композиция. Производится объединение найденных усеченных функций с использованием операции МАКСИМУМ (max, обозначенное далее как «$\vee$»), что приводит к получению итогового нечеткого подмножества для переменной выхода с функцией принадлежности

$$\mu_\Sigma(z) = C(z) = C_1'(z) \vee C_2'(z) = (\alpha_1 \wedge C_1(z)) \vee (\alpha_2 \wedge C_2(z)).$$

4. Приведение к четкости. Проводится для нахождения z_0, например, центроидным методом.

АЛГОРИТМ ЦУКАМОТО

Исходные посылки, как у предыдущего алгоритма, но здесь предполагается, что функции $C_1(z)$, $C_2(z)$ монотонны (рис. 2).

1. Введение нечеткости (как в алгоритме Мамдани).

2. Нечеткий вывод. Сначала находятся уровни «отсечения» α_1 и α_2 (как в алгоритме Мамдани), а затем решениями уравнений

$$\alpha_1 = C_1(z_1) \quad \text{и} \quad \alpha_2 = C_2(z_2)$$

определяются четкие значения (z_1 и z_2) для каждого исходного правила.

3. Определение четкого значения переменной вывода (как взвешенное среднее z_1 и z_2)

$$z_0 = \frac{\alpha_1 z_1 + \alpha_2 z_2}{\alpha_1 + \alpha_2}.$$

В общем случае (дискретный вариант центроидного метода)

$$z_0 = \frac{\sum_{i=1}^{n} \alpha_i z_i}{\sum_{i=1}^{n} \alpha_i}.$$

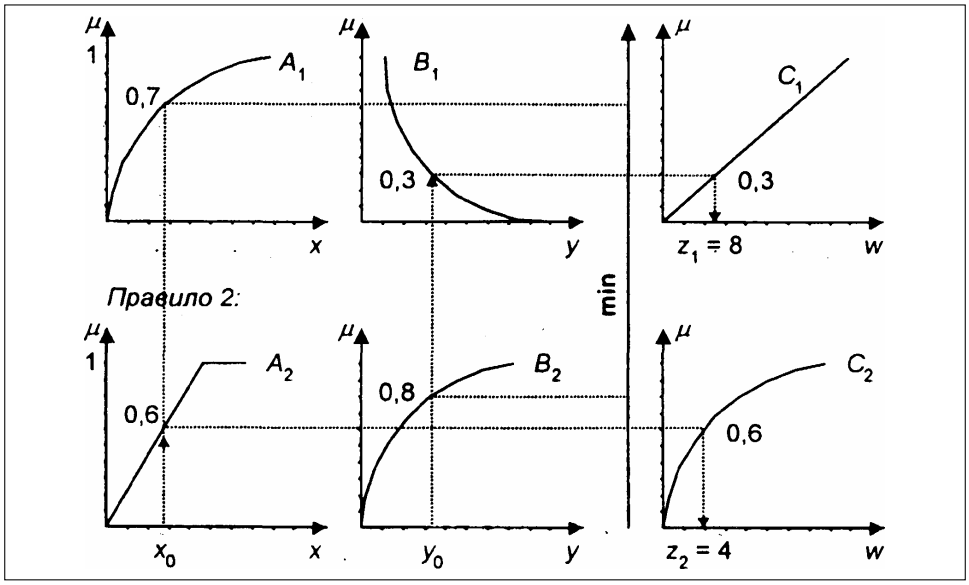

Рис. 2. Иллюстрация к алгоритму Цукамото

МЕТОДЫ ПРИВЕДЕНИЯ К ЧЕТКОСТИ

1. Выше рассмотрен один из методов — центроидный. Приведем соответствующие формулы еще раз. В общем случае

$$z_0 = \frac{\int_\Omega z\, C(z) dz}{\int_\Omega C(z) dz},$$

для дискретного варианта

$$z_0 = \frac{\sum_{i=1}^{n} \alpha_i z_i}{\sum_{i=1}^{n} \alpha_i}.$$

2. Первый максимум (First-of-Maxima). Четкая величина вывода находится как наименьшее значение, при котором достигается максимум итогового нечеткого множества (рис. 3, *а*) $z_0 = \min\left\{ z \middle| C(z) = \max_U C(U) \right\}$.

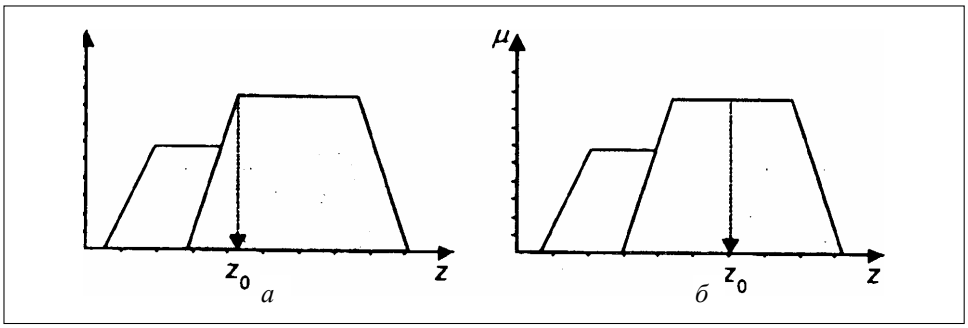

Рис. 3. Иллюстрация к методам приведения к четкости: *а* — первый и *б* — средний максимумы

3. Средний максимум (Middle-of-Maxima). Четкое значение находится по формуле

$$z_0 = \frac{\int_G z dz}{\int_G dz},$$

где G — подмножество элементов, максимизирующих C (рис. 3, *б*). Для дискретного варианта (C дискретное)

$$z_0 = \frac{1}{n} \sum_{i=1}^{n} z_i .$$

4. Критерий максимума (Max-Criterion). Четкое значение выбирается произвольно из множества элементов, для которых C достигает максимума

$$z_0 = \min\left\{ z \middle| C(z) = \max_U C(U) \right\}.$$

5. Высотная дефаззификация (Height defuzzification). Элементы области определения Ω, для которых значения функции принадлежности меньше, чем некоторый уровень α, в расчет не принимаются, и четкое значение рассчитывается соответственно выражению

$$z_0 = \frac{\int_{C_\alpha} z\, C(z) dz}{\int_{C_\alpha} C(z) dz},$$

где C_α — нечеткое множество α-уровня (см. выше).

ЭФФЕКТИВНОСТЬ НЕЧЕТКИХ СИСТЕМ ПРИНЯТИЯ РЕШЕНИЙ

Возможность использования аппарата нечеткой логики базируется на таких результатах:

1. В 1992 г. Ванг показал, что нечеткая система является универсальным аппроксиматором, т. е. может аппроксимировать любую непрерывную функцию на компакте U с произвольной точностью, если использует набор $n\ (n \to \infty)$ правил

Π_i: если x есть A_i и y есть B_i, то z есть C_i, $i = 1 \ldots n$ при следующих условиях:

- гауссовых функциях принадлежности

$$A_i(x) = \exp\left[-\frac{1}{2}\left(\frac{x-\alpha_{i1}}{\beta_{i1}}\right)^2\right], \quad B_i(y) = \exp\left[-\frac{1}{2}\left(\frac{y-\alpha_{i2}}{\beta_{i2}}\right)^2\right],$$

$$C_i(z) = \exp\left[-\frac{1}{2}\left(\frac{z-\alpha_{i3}}{\beta_{i3}}\right)^2\right];$$

- композиции в виде произведения $[A_i(x)\ \&\ B_i(y)] = A_i(x)B_i(y)$;
- импликации в форме (Larsen)

$$[A_i(x)\ \&\ B_i(y)] \to C_i(z) = A_i(x)B_i(y)C_i(z);$$

- центроидном методе приведения к четкости

$$z_0 = \frac{\sum_{i=1}^{n} \alpha_{i3} A_i B_i}{\sum_{i=1}^{n} A_i B_i},$$

где α_{i3} — центры C_{ii}.

Иначе говоря, Ванг доказал теорему: для каждой вещественной непрерывной функции g, заданной на компакте U, и для произвольного $\varepsilon > 0$ существует нечеткая система, которая формирует исходную функцию $f(x)$ такую, что

$$\sup_{x \in U} \| g(x) - f(x) \| \le \varepsilon,$$

где $\| \bullet \|$ — символ принятого расстояния между функциями.

2. В 1995 г. Кастро показал, что логический контроллер Мамдани также является универсальным аппроксиматором при следующих условиях:

- симметричных треугольных функциях принадлежности

$$A_i(x) = \begin{cases} 1 - |a_i - x| \big/ \alpha_i, & \text{если} \quad |a_i - x| \le \alpha_i, \\ 0, & \text{если} \quad |a_i - x| > \alpha_i, \end{cases}$$

$$B_i(y) = \begin{cases} 1 - \left|b_i - x\right| \Big/ \beta_i, & \text{если} \quad \left|b_i - y\right| \le \beta_i, \\ 0, & \text{если} \quad \left|b_i - y\right| > \beta_i; \end{cases}$$

- композиции с использованием операции min

$$\left[A_i(x) \;\&\; B_i(y)\right] = \min\left\{A_i(x) B_i(y)\right\};$$

- импликации в форме Мамдани и центроидного метода приведения к четкости

$$z_0 = \frac{\sum_{i=1}^{n} c_i \min\left\{A_i(x),\, B_i(y)\right\}}{\sum_{i=1}^{n} \min\left\{A_i(x),\, B_i(y)\right\}},$$

где c_i — центры C_{ii}.

В целом, системы с нечеткой логикой целесообразно применять:

- для сложных процессов, когда нет простой математической модели;
- если экспертные знания об объекте или о процессе можно сформулировать только в лингвистической форме.

Системы, базирующиеся на нечеткой логике, нецелесообразно применять:

- если необходимый результат может быть получен каким-нибудь другим (стандартным) путем;
- когда для объекта или процесса уже найдена адекватная и легко исследуемая математическая модель.

Основные недостатки систем с нечеткой логикой:

- Исходный набор нечетких правил формулируется экспертом-человеком и может оказаться неполным или противоречивым.
- Вид и параметры функций принадлежности, описывающие входные и выходные переменные системы, выбираются субъективно и могут не полностью отражать реальность.

ОБУЧЕНИЕ НЕЧЕТКОГО КОНТРОЛЛЕРА НА БАЗЕ ННС

Процесс обучения определяется нечеткой ошибкой и работает параллельно для каждого нечеткого правила. После того как C выработано контроллером и новое состояние объекта становится известным, вычисляется ошибка E, которая распространяется в обратном направлении. Каждое правило анализирует свой вклад в выход управления и оценивает свое заключение (т.е. привело оно к увеличению или уменьшению ошибки). При этом, если правило действовало в нужном направлении, то оно должно стать более чувствительным и выработать заключение, которое увеличивает управление. И наоборот, если действовало в неправильном направлении, то оно должно стать менее чувствительным. Учитывая, что ФП описываются двумя

параметрами (b_i, a_i), то один из них фиксируется, а второй меняется. В данном случае меняется a_i.

Алгоритм обучения сводится к настройке функций принадлежности условий и следствий правил.

1. Для каждого правила вычисляется ошибка

$$R_i : e_{R_i} = \begin{cases} -r_i E, & \text{если} \quad \text{sign } C_i = \text{sign } C_{\text{opt}}, \\ r_i E, & \text{если} \quad \text{sign } C_i \neq \text{sign } C_{\text{opt}}. \end{cases}$$

2. e_{R_i} передается в v и μ модули. Происходит изменение ФП v-модулей

$$a_k^{\text{new}} = \begin{cases} a_k - \sigma e_{R_i} |a_k - b_k|, & \text{если} \quad a_k < b_k, \\ a_k + \sigma e_{R_i} |a_k - b_k|, & \text{если} \quad a_k > b_k, \end{cases}$$

где σ — скорость обучения.

Если модуль v_k используется несколькими R-модулями, то его ФП меняется столько раз, сколько R-модулей связано с ним.

3. Корректируются ФП условия:

$$a_{jk}^{\text{new}} = \begin{cases} a_{jk} - \sigma e_{R_i} |a_{jk} - b_{jk}|, & \text{если} \quad a_{jk} < b_{jk}, \\ a_{jk} + \sigma e_{R_i} |a_{jk} - b_{jk}|, & \text{если} \quad a_{jk} > b_{jk}. \end{cases}$$

Модуль x_j связан с модулем R_k через μ_{jk}. Этот алгоритм называется алгоритмом с подкреплением.

Если процесс обучения оказался успешным, то база правил сформирована верно. Можно организовать процесс обучения таким образом, чтобы не только настраивать ФП правил, но и корректировать сами правила. Если выход правила соответствует семантике управления, то оно сохраняется в базе. Если же выход правила противоположен исходному, то его заключение меняется на противоположное или же правило совсем удаляется.

РАСПРОСТРАНЕНИЕ ОШИБКИ В НЕЧЕТКОМ НЕЙРОННОМ КОНТРОЛЛЕРЕ

Одна из проблем конструирования нечеткого контроллера — это выбор подходящей функции принадлежности. Данную проблему можно решить с помощью интеграции методики обучения НС и архитектуры нечеткого контроллера.

Стандартный метод состоит в добавлении еще одного модуля в архитектуру, учитывая необходимость коррекции ошибок. Мы рассматриваем вычисления управляющей переменной по данным измерения входных переменных как последовательную процедуру в многослойных НС, где входные сигналы распространяются в прямом направлении (feed forward), но если действительные значения выходов отличаются от желаемых, то ошибка рас-

пространяется в обратном направлении с учетом величин, рассчитанных во время прямого хода.

Рассмотрим теперь полную архитектуру нечеткого контроллера (рис. 4).

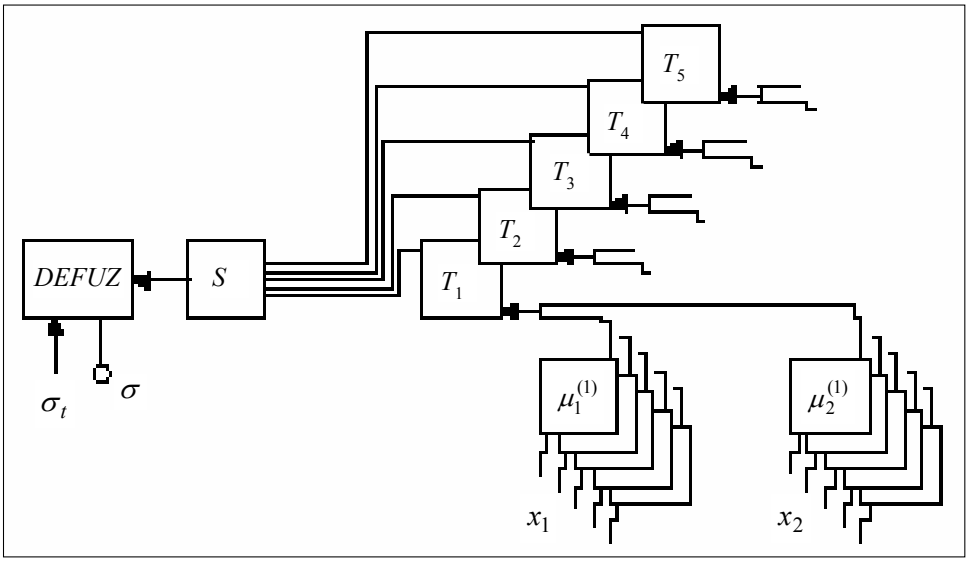

Рис. 4. Архитектура системы

Она состоит из следующих компонентов:

MSF — модуль, реализующий процедуру фаззификации.

T — модуль для агрегации входных данных.

S — модуль для агрегации выходных данных.

DEFUZ — модуль, выполняющий процесс дефаззификации.

Модуль *T* выбирается операцией минимизации (пересечения), *S* — максимизации (объединения).

На рис. 4 имеется два входа x_1 и x_2 для пяти правил, где *MSF* заданы как $\mu_i^1, \ldots, \mu_i^5$, $i = 1,2$.

Значения, полученные *MSF*-модулями, передаются сначала *T*-модулям, а потом через *S*-модули — процессу дефаззификации. Выходной сигнал σ сравнивается с желаемым выходным сигналом σ_t и определяется ошибка δ. Проблема заключается в настройках входных и выходных *MSF*-модулей.

Процедура обучения. Ошибка $\delta_\sigma^{(i)}$ является комбинацией бокового смещения доменов $\delta_\sigma^{(S,i)}$ и зависит от формы функций $\delta_\sigma^{(T,i)}$. Такие ошибки распространяются в обратном направлении соответственно в *S*- и *T*-модулях. Теперь мы можем записать

$$\delta_{\sigma_I}^{(i)} = \left(\delta_\sigma^{(S,i)}, \delta_\sigma^{(T,i)}\right).$$

Величины ошибок соотносятся с позицией выходной величины и ее желаемым значением. В методе прямого распространения следует уделять

особое внимание ошибкам в случае, если выходное и желаемое значения находятся близко к границам измеряемого интервала.

Будем рассматривать следующие два метода дефаззификации: центроидный (COA) и метод среднего значения максимума (MOM). После того как получено четкое выходное значение $\sigma = DEFUZ(S)$, мы должны определить сигнал ошибки δ_σ как разницу $\sigma - \sigma_t$ между выходным значением и желаемым выходным значением σ_t.

Рассмотрим MOM-процедуру дефаззификации.

Существует четыре ситуации, в которых выходная величина принимает неверное значение.

1. Если желаемое значение расположено под вершиной S (рис. 5, *а*), но сдвинуто немного влево или вправо.

2. Если желаемое значение принадлежит области определения нечеткого множества T_i, которое генерируется вершиной S_i, но σ_t не располагается под вершиной S.

3. Если желаемое значение не принадлежит области определения нечеткого множества T_i, которое генерируется вершиной S_i, но оно по-прежнему находится внутри области принадлежности S (рис. 5, *б*).

4. Если желаемое значение σ_t не принадлежит области определения S.

В двух первых случаях вершина S сгенерирована правильным выходным нечетким множеством T, но она смещена. Мы предполагаем, что только T_i влияет на неверное значение выхода σ. Эту ошибку можно преодолеть, изменяя форму T_i таким образом, чтобы положение середины вершины T_i было σ_t (прерывистые линии на рис. 5, *б*).

В третьем случае вершина S сгенерирована неверным выходным нечетким множеством. Эта ошибка происходит из-за входных нечетких множеств, которые мы сначала должны изменить, чтобы проверить, что вершина S сгенерирована правильно T_i.

Предполагаем, что вершина S сгенерирована T_j и σ_t принадлежит области принадлежности T_j. Следовательно, мы должны увеличить значения $\min\limits_k \left\{\mu_k^{(j)}\right\}$ на выходе T_j и уменьшить величину $\min\limits_k \left\{\mu_k^{(i)}\right\}$. Затем применяем T_i таким же образом, как и в двух первых случаях.

В четвертом случае, когда σ_t не принадлежит области определения S_i, делаем вывод, что в нашей базе правил присутствует ошибка.

Это может быть либо в случае, когда не существует правила, покрывающего область нахождения σ_t, либо правило не соответствует действительности. Здесь следует использовать правило, соответствующее данной ситуации. Такая процедура выполняется не автоматически, а задается пользователем.

Алгоритм нечеткого распространения ошибки. Опишем алгоритм нечеткого распространения ошибки в нечеткой нейронной сети (рис. 5).

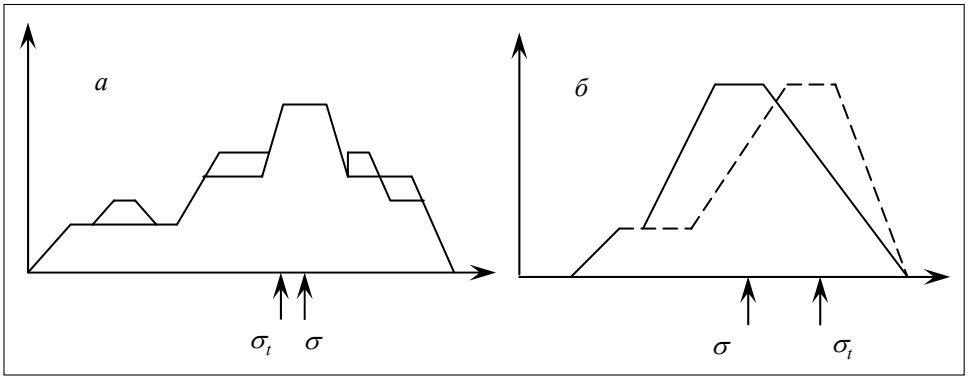

Рис. 5. Регулирование σ увеличением/уменьшением формы всех или одного из T нечетких множеств

1. После того как устройство *DEFUZ* сгенерировало σ, оно передает на обратном ходе правильное значение σ_t и фактическое значение выхода σ в *S*-модуль.

2. Проверяется условие принадлежности σ_t области определения S. Если оно не выполняется (т.е. $\sigma_t \notin$ области определения S), то эксперт должен добавить новое правило в базу правил и пересчитать текущее выходное значение перед началом процедуры обучения. В противном случае переходим на следующий шаг.

3. Если *S*-модуль передает в обратном направлении σ и σ_t, то высота S $(h_{\max})$ и максимальная высота каждого нечеткого множества, не соответствующего вершине $h_{\max}$, принадлежат *T*-модулям.

4. Каждый *S*-модуль проверяет выполнение условий:

- Если высота $T_i = S$: $h_i = h_{\max}$ и σ_t не принадлежит области определения T_i, то посылаем $h_{\max}$, $h(T_i)$ и понижающий сигнал всем подключенным к T_i μ-модулям.
- Если $h(T_i) = h(S) = h_{\max}$ и σ_t принадлежит области определения T_i, но $\sigma_t \neq \sigma$, то изменяем форму T_i таким образом, чтобы σ_t определяло положение максимума вместо σ.
- Если $h(T_i) \neq h(S) = h_{\max}$ и σ_t принадлежит области определения T_i, то посылаем значения высоты $h(S)$, $h(T_i)$ и повышающий сигнал всем μ-модулям, подключенным к T_i, принимаем $h(S)$ как $\min\limits_{k=I,K} \left\{\mu_k^{(i)}\right\} = \min\limits_{k} \left\{\mu_k(T_i)\right\}$ и изменяем форму T_i таким образом, чтобы $\max \mu(S)$ достигался в точке σ_t вместо точки σ.
- Если $h(T_i) = h(S)$ и σ_t не принадлежит области определения T_i, то прерываем распространение ошибки в этом модуле.

5. Аналогично п.4, каждый μ-модуль проверяет выполнение условий:

- Если поступает понижающий сигнал и $\mu_i^{(j)}(x_j) = h(T_j)$, то изменяем форму $\mu_i^{(j)}$ так, чтобы $\mu_i^{(j)}(x_j) = h_{\max}$.

- Если поступает повышающий сигнал и $\mu_i^{(j)}(x_j) < h(T_j)$, то изменяем форму $\mu_i^{(j)}$ так, чтобы $\mu_i^{(j)}(x_j) = h(S)$.

Итак, с применением технологии обучения НС был представлен алгоритм Back propagation для ошибки в нечетком контроллере. Рассмотрен нечеткий контроллер как НС, где нечеткие множества хранятся в узлах системы. Алгоритм обучения определяет модуль, отвечающий за ошибку в сигнале на выходе и распространяющий информацию в обратном направлении через НС, что позволяет ему изменять нечеткие множества (параметры их ФП).

ПОСТАНОВКА ЗАДАЧИ МАКРОЭКОНОМИЧЕСКОГО ПРОГНОЗИРОВАНИЯ И ЭКСПЕРИМЕНТАЛЬНЫЕ ИССЛЕДОВАНИЯ

В качестве входных данных используются показатели: индекс потребительских цен — ИПЦ (0), а также денежные агрегаты — M0 и M2 и эти же данные с шагом 1 и 2 — M0(-1); M2(-1); M0(-2) и M2(-2). Выходной прогнозируемой переменной является значение ИПЦ в следующий момент времени (1 месяц). Размер скользящего окна — 10 точек, 11-я — прогнозная.

Эксперименты проводились с НК Мамдани с треугольными ФП и с гауссовыми ФП, а также НК Цукамото с линейными ФП. Результаты приведены в табл. 1, где указана СКО на обучающей и проверочной выборках (10 окон). Как видно из приведенных результатов, все три нечетких контроллера отлично справились с задачей прогнозирования на проверочных точках, что свидетельствует как об эффективности алгоритмов обучения контроллеров, так и о целесообразности применения систем с нечетким логическим выводом для прогнозирования в макроэкономике.

Таблица 1. Выбор лучшего метода (окно — 14 первых точек)

Номер измерения	Мамдани, треугольные ФП, пересечение в форме минимума		Мамдани, гауссовские ФП, пересечение в форме произведения		Цукамото, линейные ФП	
	СКО (обучающая выборка)	СКО (проверочная выборка)	СКО (обучающая выборка)	СКО (проверочная выборка)	СКО (обучающая выборка)	СКО (проверочная выборка)
1	0,02661	0,12793	0,0483	0,15609	0,0292	0,12881
2	0,03107	0,1348	0,04113	0,05968	0,05605	0,10135
3	0,06203	0,19753	0,0537	0,11517	0,04518	0,03932
4	0,04227	0,06942	0,06067	0,03355	0,04793	0,09438
5	0,00688	0,03328	0,02823	0,05945	0,03698	0,11423
6	0,01299	0,11387	0,03337	0,09743	0,05396	0,19284
7	0,00813	0,01397	0,08836	0,0748	0,03536	0,08257
8	0,00705	0,0722	0,08238	0,04632	0,02827	0,13531
9	0,02111	0,17541	0,08936	0,16248	0,06276	0,11651
10	0,00847	0,0778	0,09111	0,11834	0,03905	0,11877
Среднее	**0,022661**	**0,101621**	**0,061661**	**0,092331**	**0,043474**	**0,112409**

Как показал первый эксперимент, лучшим, хотя и с небольшим отрывом, оказался контроллер Мамдани с гауссовыми ФП и нечетким пересечением в виде произведения. (СКО на проверочной выборке из 10 точек составляет всего 0,092, относительная средняя ошибка прогноза — 9,29.)

Таблица 2. Влияние положения окна на качество прогноза

Окно	СКО (обучающая выборка)	СКО (проверочная выборка)
0 – 10	0,06311	0,03695
	0,08957	0,04265
	0,05682	0,04388
	0,06983333	**0,04116**
2 – 12	0,03384	0,03214
	0,09313	0,04958
	0,05452	0,048
	0,06049667	**0,04324**
4 – 14	0,066	0,04377
	0,04141	0,05052
	0,09749	0,05053
	0,0683	**0,048273333**
6 – 16	0,06285	0,07032
	0,07409	0,06459
	0,0729	0,05625
	0,06994667	**0,06372**
8 – 18	0,05526	0,07767
	0,07218	0,10083
	0,0834	0,09887
	0,07028	**0,092456667**
10 – 20	0,00985	0,0993
	0,07122	0,08457
	0,08482	0,09522
	0,05529667	**0,09303**
12 – 22	0,02691	0,09358
	0,05971	0,09803
	0,09022	0,07577
	0,05894667	**0,089126667**
Итого СКО	**0,06472857**	**0,067286667**

В следующих экспериментах использовался именно контроллер Мамдани с такими ФП.

Во втором эксперименте мы исследовали влияние положения окна обучения на качество обучения. Результаты приведены в табл. 2. Как следует из приведенных данных, присутствует явная тенденция к росту ошибки прогнозирования при сдвиге окна вправо.

В третьем эксперименте исследовалось влияние количества правил на выход сети. В нем варьировалась вероятность вывода правил для включения в базу правил ННК. Результаты приведены в табл. 3. Наилучший результат был достигнут при всех правилах ($p = 1$). В дальнейшем до точки 0,5 наблюдался устойчивый рост СКО, а потом — спад.

Это может свидетельствовать об эффекте переучивания сети (overfitting), который затем пропадает с уменьшением числа правил.

АНАЛИЗ ПОЛУЧЕННЫХ РЕЗУЛЬТАТОВ

Как видно из приведенных таблиц, все три контроллера отлично справились с задачей аппроксимации контрольной выборки. Это свидетельствует о работоспособности алгоритмов обучения контроллеров, а также о высокой репрезентативности самой выборки (это очевидно — она выбиралась случай-

ным образом). Т.е. все контроллеры можно использовать для решения основной задачи на реальных данных.

Таблица 3. Влияние наполнения базы правил на качество прогноза

P	СКО (обучающая)	СКО (проверочная)	Количество правил
1,0	0,06311 0,08957 0,05682 **0,06983333**	0,03695 0,04265 0,04388 **0,04116**	95
0,9	0,09615 0,06763 0,08725 0,06944 0,09394 **0,082882**	0,03923 0,09407 0,05471 0,03458 0,07877 **0,060272**	85
0,8	0,0758 0.0408 0,05468 0,09833 0,1008 **0,074082**	0,08626 0,07011 0,06611 0,08782 0,03536 **0,069132**	78
0,7	0,07452 0,08887 0,13067 0,04775 0,08338 **0,085038**	0,03255 0,06717 0,06815 0,11539 0,08021 **0,072694**	64
0,6	0,0454 0,11291 0,04537 0,06939 0,05771 **0,066156**	0,0938 0,04815 0,06632 0,09047 0,09908 **0,079564**	59
0,5	0,05924 0,10042 0,09767 0,04777 0,05276 **0,071572**	0,11491 0,08577 0,12388 0,07836 0,08818 **0,09822**	47
0,4	0,06259 0,05311 0,09642 0,09894 **0,0765264**	0,05529 0,09281 0,07437 0,03017 **0,070172**	39
0,3	0,1185 0,05224 0,08783 0,06995 **0,08100928**	0,03378 0,09027 0,06682 0,06527 **0,0652624**	25
0,2	0,07312 0,1065 0,07512 0,05972 **0,07909386**	0,03359 0,0367 0,05475 0,02985 **0,04403048**	20

Как показал первый тест, лучшим (хотя и с небольшим отрывом) оказался контроллер Мамдани, работающий с гауссовыми ФП и нечетким пере-

сечением в виде произведения (СКО на проверочной выборке из 10 точек = 0,09). Поэтому для дальнейшего рассмотрения выбираем именно его.

Во втором тесте мы исследовали влияние положения окна обучения на качество обучения. Как ни странно, присутствует явная тенденция к росту ошибки распознавания при сдвиге окна вправо, т. е. начало выборки несет больше информации о процессе, чем конец. Поэтому для следующего теста выбираем первое окно.

Так как чем меньше правил, тем меньше сложность сети и вероятность перенастройки, необходимо провести влияние количества правил на выход сети. Это было сделано в тесте 3. Лучший результат получен при всех правилах (вероятность попадания в базу = 1), в дальнейшем до точки 0,5 наблюдался устойчивый рост СКО, а потом — спад. Это может свидетельствовать о пропадании эффекта переучивания сети с уменьшением количества правил.

ВЫВОДЫ

В настоящей работе проведены исследования нечетких нейронных контроллеров с логическим выводом Мамдани и Цукамото в задаче прогнозирования экономических показателей.

1. Все нечеткие контроллеры способны решать задачи прогнозирования сложных нелинейных динамических процессов, причем наиболее эффективным оказался ННК Мамдани.

2. Исследование влияния числа правил на точность прогноза показали, что наиболее эффективным оказывается ННК при всех правилах ($p = 1$), а также при 20 правилах.

3. Проведенные эксперименты показали большие потенциальные возможности ННС и подтвердили их эффективность в задачах макроэкономического прогнозирования.

ЛИТЕРАТУРА

1. *Зайченко Ю.П., Мухаммед М., Шаповаленко Н.В.* Нечіткі нейронні мережі і генетичні алгоритми в задачах макроекономічного прогнозування // Наук. вісті НТУУ «КПІ». — 2002. — № 4. — С. 20–30.
2. *Круглов В.В., Борисов В.В.* Искусственные нейронные сети. Теория и практика. — М.: Горячая линия. Телеком. — 2001. — 260 с.
3. *Оссовский С.* Нейронные сети для обработки информации / Пер. с польского И.Д. Рудинского. – М.: Финансы и статистика, 2002. — 344 с.

Поступила 14. 01. 2004