О программной реализации нейро-нечеткой сети на основе байесовской логико-вероятностной модели

Г. А. Хамчичев[1], Г. И. Кожомбердиева[2], Д. П. Бураков[3]

[1,2]*Петербургский государственный университет путей сообщения Императора Александра I*
[3]*Независимый исследователь, Санкт-Петербург*

[1]gyqn@yandex.ru, [2]kgi-liizht@yandex.ru, [3]burakovdmitry8@gmail.com

Аннотация. **Доклад посвящен программной реализации нейро-нечеткой сети, структура которой, основанная на байесовской логико-вероятностной модели (БЛВ-модели) нечеткого вывода, ранее предложена и опубликована авторами. В программе на языке Java, помимо БЛВ-модели в нейросетевом контексте, реализованы адаптированные алгоритмы, используемые при построении (алгоритм сеточного разбиения для генерации нечетких правил) и обучении (алгоритм обратного распространения ошибки и гибридный алгоритм) сети, известные по использованию в сети ANFIS из пакета MATLAB. Программа является инструментом для решения практических задач, исследования эффективности и возможностей использования нейро-нечеткой сети на основе БЛВ-модели.**

Ключевые слова: нейро-нечеткая сеть; байесовская логико-вероятностная модель нечеткого вывода; машинное обучение; обучение нейро-нечеткой сети; алгоритмы обучения; язык программирования Java

I. Введение

Нейро-нечеткие сети (Neuro-Fuzzy Networks, NFN) представляют собой многослойные нейросетевые нечеткие системы, обладающие способностью формировать выходные сигналы на основе фаззификации значений входных сигналов, использования базы нечетких правил и благодаря адаптивной настройке значений параметров, подаваемых на параметрические слои нейронов. Нейроны непараметрических и параметрических слоев нейро-нечеткой сети выполняют расчетные операции в соответствии с применяемой моделью нечеткого вывода. Нейросетевая структура таких систем делает их способными к обучению и в этой связи к использованию в качестве универсальных функциональных аппроксиматоров. Как и в классических нейронных сетях, в нейро-нечетких сетях в процессе обучения происходит изменение параметров сети, благодаря чему сеть настраивается таким образом, чтобы ее результирующие выходные значения соответствовали ожидаемым.

Структура многослойной нейро-нечеткой сети (ННС) на основе байесовской логико-вероятностной модели (БЛВ-модели) нечеткого вывода опубликована авторами в [1] и прошла апробацию на Международном симпозиуме по автоматизации, информатизации и вычислительной технике (ISAIC 2022, Пекин, КНР) и Международной конференции по мягким вычислениям и измерениям (SCM'2023, Санкт-Петербург, Россия) [2–4]. В ранее опубликованных работах авторы ссылались на многочисленные примеры использования нейро-нечетких сетей, свидетельствующие об актуальности и интенсивности современных исследований и разработок в этой области искусственного интеллекта. В последние годы сети этого типа зарекомендовали себя как мощный инструмент, обеспечивающий эффективное решение задач автоматического управления и регулирования, диагностики, прогнозирования временных рядов, а также иных задач, связанных с анализом и обработкой данных и сводимых к аппроксимации функций нескольких переменных.

Программная реализация ННС на основе БЛВ-модели нечеткого вывода, которой посвящен настоящий доклад, укрепила авторов во мнении, что предложенная ими сетевая структура сопоставима с известными нейро-нечеткими сетями Такаги–Сугено–Канга и Ванга–Менделя [5]. Сеть ANFIS (Adaptive Neuro-Fuzzy Inference System), доступная в пакете MATLAB [6] и основанная на модели нечеткого вывода Такаги–Сугено–Канга, послужила авторам как ориентиром при разработке графического интерфейса пользователя, так и подходящим аналогом для сравнения функциональных возможностей ННС на основе БЛВ-модели.

II. Нейро-нечеткая сеть на основе байесовской логико-вероятностной модели

БЛВ-модель нечеткого вывода предложена Г. И. Кожомбердиевой на Международной конференции по мягким вычислениям и измерениям (SCM'2019, Санкт-Петербург, Россия) [7], и позднее исследована и программно реализована авторами настоящего доклада [8]. Опыт этой программной реализации использован при разработке программы для создания и обучения ННС на основе БЛВ-модели.

Нейросетевая интерпретация БЛВ-модели нечеткого вывода и структурная схема ННС на ее основе представлены в [1–3]. Здесь только кратко остановимся на функциональности нейронов, распределенных по слоям сети. Следует отметить, что на начальном этапе разработки программы, с учетом полученных предварительных результатов обучения ННС, Г. А. Хамчичевым внесены полезные изменения в функциональное описание некоторых слоев сети.

Сеть состоит из семи слоев, три из которых (первый, третий и шестой) являются параметрическими. Первый слой нейронов выполняет фаззификацию каждой входной переменной x_n, $n = \overline{1,N}$, то есть расчет значения функции принадлежности (ФП) для каждого j-го терма, $j = \overline{1,M_n}$, каждой n-й входной лингвистической переменной (ЛП), причем, в отличие от представления структуры ННС в [1–3], в программной реализации в качестве ФП используется частный случай ($b = 1$) обобщенной гауссовой функции

$$g_{j,n}(x_n; c_{j,n}, \sigma_{j,n}) = e^{-\frac{(x_n - c_{j,n})^{2b}}{\sigma_{j,n}^2}}, \quad (1)$$

которая при определенных значениях показателя степени b может определять треугольную и трапецеидальную ФП. Центры $\boldsymbol{c}$ и коэффициенты ширины $\boldsymbol{\sigma}$ ФП являются параметрами первого слоя ННС.

Второй слой нейронов рассчитывает условные вероятности $P(e|H_k)$, $k = 1, \dots, K$ на основе базы нечетких правил, трансформированной в набор функций вероятностной логики (ФВЛ). ФВЛ принимают в качестве аргументов значения ФП $g_{j,n}$, рассчитанные нейронами первого слоя сети. Отметим, что в качестве алгоритма генерации базы нечетких правил для программной реализации ННС выбран алгоритм сеточного разбиения, обсуждаемый в следующем разделе доклада. Этим обусловлен простой вид формулы для расчета условной вероятности k-м нейроном второго слоя:

$$P(e|H_k) = \prod_{n=1}^{N} g_{j,n}(x_n; c_{j,n}, \sigma_{j,n}). \quad (2)$$

Нейроны третьего слоя выполняют “взвешивание” значений условных вероятностей параметрами-весами нечетких правил $\boldsymbol{w}$. Нейрон четвертого слоя суммирует взвешенные условные вероятности. Нейроны пятого слоя по формуле, основанной на формуле Байеса, формируют апостериорное распределение вероятностей:

$$P(H_k|e) = \frac{w_k \cdot P(e|H_k)}{\sum_{l=1}^{K} w_l \cdot P(e|H_l)}, \quad (3)$$

где K – число термов выходной ЛП (байесовских гипотез H_k), соответствующих числу ФВЛ (оценивающих степень истинности свидетельств в пользу каждой гипотезы), w_k – вес -го правила, $w_k \in [0; 1]$.

Нейроны шестого слоя и нейрон-сумматор седьмого слоя отвечают за дефаззификацию выходной переменной. Итоговое значение выходной переменной y вычисляется по следующей формуле:

$$\hat{y} = \sum_{k=1}^{K} \tilde{y}_k \cdot P(H_k|e), \quad (4)$$

где $P(H_k|e)$ – элемент апостериорного распределения вероятностей (3), а $\tilde{y}_k$ – характерное значение соответствующего терма выходной ЛП. При этом, в отличие от структуры ННС в [1–3], в качестве параметров шестого слоя в программной реализации сети выступают не коэффициенты α_k, предназначавшиеся для вычисления значений $\tilde{y}_k$ как выпуклых комбинаций граничных точек соответствующих термам интервалов на шкале выходной переменной, а непосредственно характерные значения $\tilde{\boldsymbol{y}}$ термов выходной ЛП. Результатом работы сети является число $\hat{y}$, аппроксимирующее значение некоторой функции $y = f(x_1, \dots, x_N)$.

Итак, параметрами ННС на основе БЛВ-модели нечеткого вывода, которые уточняются и обновляются в процессе обучения сети, являются:

- центры $\boldsymbol{c}$ и коэффициенты ширины $\boldsymbol{\sigma}$ гауссовых ФП (первый слой нейронов);
- веса нечетких правил $\boldsymbol{w}$ (третий слой нейронов);
- характерные значения термов выходной ЛП $\tilde{\boldsymbol{y}}$ (шестой слой нейронов).

Параметры $\boldsymbol{c}$, $\boldsymbol{\sigma}$ и $\boldsymbol{w}$ отнесем к нелинейным параметрам, поскольку они используются в нелинейных функциях (1) и (3), а параметры $\tilde{\boldsymbol{y}}$ – к линейным, так как они используются в линейной функции (4).

III. АЛГОРИТМЫ, ИСПОЛЬЗУЕМЫЕ ПРИ ПОСТРОЕНИИ И ОБУЧЕНИИ НЕЙРО-НЕЧЕТКОЙ СЕТИ

Для полноценной работы с ННС необходимы алгоритмы двух типов. Во-первых, это алгоритмы начального формирования ее структуры, которые на основе имеющегося набора данных, выбранного вида ФП и указанного количества термов входных ЛП выполняют генерацию нечетких правил. Во-вторых, это алгоритмы обучения ННС, выполняющие обновление параметров выражений (1–4) таким образом, чтобы после каждой эпохи обучения сети (одного цикла полного прохода по обучающему набору данных), уменьшалось бы значение функции потерь

$$L(y_i, \hat{y}_i(\boldsymbol{c}, \boldsymbol{\sigma}, \boldsymbol{w}, \tilde{\boldsymbol{y}})) = \frac{1}{2}(y_i - \hat{y}_i)^2, \quad (5)$$

где y_i и $\hat{y}_i$ соответственно представляют целевое и предсказанное значения для i-й строки данных $x_1, \dots, x_N$ из обучающего набора, $i = \overline{1,Z}$. Обучение завершается либо по завершении определенного числа эпох, либо по достижении заданной величины ошибки.

В программе для формирования структуры ННС используется алгоритм сеточного разбиения (Grid Partitioning), поскольку он доступен в MATLAB и несложен в реализации, а для обучения выбраны алгоритм обратного распространения ошибки (АОРО), основанный на градиентном спуске (ГС), и гибридный алгоритм (ГА) [4]. Как показывает практика работы с ANFIS в MATLAB, ГА обеспечивает более быструю сходимость обучения, чем АОРО.

A. Алгоритм сеточного разбиения

Суть алгоритма сеточного разбиения состоит в равномерном распределении термов каждой n-й входной ЛП по шкале значений соответствующей входной переменной. Так, если ФП для термов входной ЛП определяются как гауссовы ФП (1), то шкала делится на $s_n + 1$ равных интервалов, где s_n – выбранное количество термов, и далее в качестве центров $c_{j,n}$ выбираются значения точек соприкосновений этих интервалов, а в качестве коэффициентов ширины $\sigma_{j,n}$ – длины интервалов. С помощью аналогичного разбиения шкалы выходной переменной на $K + 1$ равных интервалов выбираются и $\tilde{y}_k$ – характерные значения термов выходной ЛП. Количество характерных значений

K соответствует количеству правил, каждое из которых формируется как конъюнкция утверждений вида «ЛП$_n$ = Терм$_j$», где n – индекс входной ЛП, j – индекс ее терма.

B. Адаптация алгоритма обратного распространения ошибки

Процесс обучения направлен на минимизацию функции потерь путем итеративной коррекции параметров ННС. В адаптации АОРО для его применения при обучении ННС на основе БЛВ-модели ключевым элементом является выработка правила обновления параметров сети $\boldsymbol{c}, \boldsymbol{\sigma}, \boldsymbol{w}, \tilde{\boldsymbol{y}}$ на основе ГС при минимизации функции (5). Для обновления каждого параметра ННС используется следующее выражение:

$$\pi^* = \pi - \eta_\pi \frac{\partial L}{\partial \pi}, \tag{6}$$

где π^* – новое значение параметра, π – его текущее значение, η_π – темп изменения параметра, а $\frac{\partial L}{\partial \pi}$ – частная производная функции (5) по этому параметру, найденная с использованием цепного правила.

В табл. I представлены конкретные варианты правила (6), используемые для обновления параметров ННС в ходе обучения.

ТАБЛИЦА I. ПРАВИЛА ОБНОВЛЕНИЯ ПАРАМЕТРОВ ННС

Параметр	Темп изменения параметра	Ограничение на новое значение параметра
$\tilde{y}_k$	$\eta_{\tilde{y}} = \frac{\lambda_{\tilde{y}}}{\sqrt{\sum_{k=1}^{K}\left(\frac{\partial L}{\partial \tilde{y}_k}\right)^2}}$	–
w_k	$\eta_w = \frac{\lambda_w}{\sqrt{\sum_{k=1}^{K}\left(\frac{\partial L}{\partial w_k}\right)^2}}$	$0 \le w_k^* \le 1$
$c_{j,n}$	$\eta_{c_{j,n}} = \frac{\lambda_c}{\sqrt{\sum_{n=1}^{N}\sum_{j=1}^{M_n}\left(\frac{\partial L}{\partial c_{j,n}}\right)^2}}$	–
$\sigma_{j,n}$	$\eta_{\sigma_{j,n}} = \frac{\lambda_\sigma}{\sqrt{\sum_{n=1}^{N}\sum_{j=1}^{M_n}\left(\frac{\partial L}{\partial \sigma_{j,n}}\right)^2}}$	${\sigma_{j,n}}^* \ge 0$

Здесь $\lambda_{\tilde{y}}, \lambda_w, \lambda_c, \lambda_\sigma$ – произвольно подбираемые размеры шага изменения соответствующего параметра сети в ходе обучения ННС. Наряду с числом используемых эпох обучения, пороговым значением функции потерь (5) и направлением ГС, они рассматриваются как *параметры обучения* ННС.

Для контроля за тем, чтобы значения весов правил в процессе обучения не выходили за установленный диапазон $[0; 1]$, для их изменения используется метод спроецированного градиентного спуска (Projected Gradient Descent, PGD) [9]: $w_k^* = \max(\min(w_k^*, 1), 0)$.

Порядок, в котором обновляются линейные параметры и веса, не критичен, так как производные функции ошибки по этим параметрам не взаимосвязаны, что позволяет осуществлять их обновление параллельно и независимо. Это обеспечивает гибкость в оптимизации и упрощает реализацию алгоритма.

C. Адаптация гибридного алгоритма обучения

Этот алгоритм сочетает в себе как ГС, так и метод наименьших квадратов (МНК). Алгоритм ГС применяется для тонкой настройки нелинейных параметров $\boldsymbol{c}$, $\boldsymbol{\sigma}$ и $\boldsymbol{w}$, обеспечивая эффективное снижение общей ошибки (5). В то же время, МНК идеально подходит для быстрой и точной корректировки линейных параметров $\tilde{\boldsymbol{y}}$ благодаря его способности к минимизации ошибок в линейно зависимых данных [5].

Для адаптации ГА часть уже реализованного алгоритма АОРО изменена таким образом, что при обратном проходе не выполняется изменение линейных параметров $\tilde{\boldsymbol{y}}$ с помощью ГС. Вместо этого в начале каждой эпохи обучения выполняется прямой проход по всему обучающему набору данных, и с помощью МНК линейные параметры $\tilde{\boldsymbol{y}}$ обновляются путем решения системы линейных алгебраических уравнений

$$\tilde{\boldsymbol{y}} = (P^T P)^{-1} P^T Y,$$

где вектор целевых значений Y содержит реальные выходные данные обучающего набора, а матрица P формируется на основе значений выходов пятого слоя сети – апостериорных вероятностей (3), рассчитанных для всех обучающих примеров.

Таким образом, МНК направлен на нахождение оптимального вектора характерных значений $\tilde{\boldsymbol{y}}$, минимизируя функцию потерь, определенную как сумма квадратов разностей между реальными и предсказанными значениями.

После завершения этого шага происходит обновление нелинейных параметров $\boldsymbol{c}$, $\boldsymbol{\sigma}$ и $\boldsymbol{w}$ в результате обратных проходов по каждой строке набора данных, как в АОРО.

D. Метрики оценки качества обученной сети

В программе используются известные метрики качества аппроксимации, такие как MSE (среднеквадратическая ошибка, Mean Squared Error), RMSE (корень из среднеквадратической ошибки, Root Mean Squared Error), MAE (средняя абсолютная ошибка, Mean Absolute Error), MAPE (средняя абсолютная процентная ошибка, Mean Absolute Percentage Error) и R^2 (коэффициент детерминации) [9–11]. Эти метрики позволяют оценить точность и эффективность нейро-нечетких сетей в задачах аппроксимации. **MSE** измеряет среднюю квадратичную разницу между предсказанными и реальными значениями, оценивая общую точность модели. Она особенно полезна для выявления больших ошибок, так как квадратичная функция штрафует их более сильно. **RMSE** представляет собой квадратный корень из MSE, обеспечивая оценку ошибки в тех же единицах, что и исходные данные. Является аналогом среднеквадратического отклонения (СКО). В дальнейшем мы будем ориентироваться на эту ошибку для сравнения сети с ANFIS из пакета MATLAB, поскольку там по умолчанию используется именно эта метрика. **MAE** рассчитывает среднее абсолютное отклонение предсказанных значений от фактических, являясь мерой линейной точности. **MAPE** выражает ошибку в процентах, что полезно для оценки относительной точности без привязки к масштабу данных. **R^2** указывает на долю дисперсии зависимой

переменной, объясненную моделью, оценивая ее предсказательную силу.

Включение данных метрик в программу обусловлено их способностью к комплексной оценке точности, стабильности и адаптивности ННС к новым данным.

IV. Функциональные возможности программы, реализующей ННС на основе БЛВ-модели

Разработанная программа обеспечивает построение и обучение ННС, позволяющей эффективно решать задачи аппроксимации функций нескольких переменных. Функциональные возможности программы и поддерживающий их реализацию графический интерфейс пользователя сопоставимы с известным аналогом ANFIS из пакета MATLAB.

A. Построение сети

Пользователь указывает обучающий набор данных, количество термов входных ЛП и начальное значение весов $\boldsymbol{w}$. На этой основе программа автоматически создает нечеткие правила, необходимые для формирования структуры ННС, и определяет начальные значения параметров сети $\boldsymbol{c}$, $\boldsymbol{\sigma}$, $\tilde{\boldsymbol{y}}$.

В программе не поддерживается возможность редактирования правил, сгенерированных на этапе построения сети, причем правила содержат только просто реализуемую операцию конъюнкции. Однако на практике использование (в сетях типа ANFIS) только конъюнкции в нечетких правилах достаточно для получения точного результата в решаемых задачах.

B. Обучение сети

Работу с программой на этапе обучения сети иллюстрируют рис. 1 и рис. 2. Пользователь выбирает алгоритм обучения (АОРО или ГА), устанавливает пороговое значение функции потерь, задает количество эпох обучения. Настраивать параметры обучения ННС можно в окне, представленном на рис. 1.

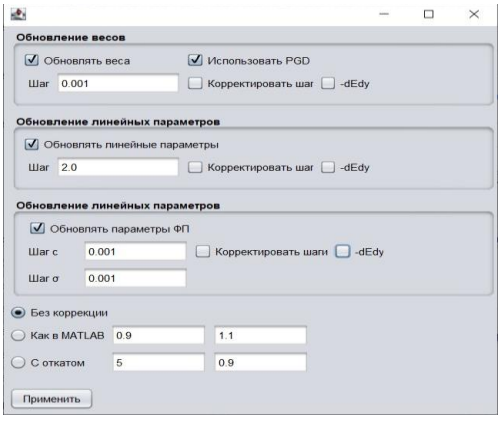

Рис. 1. Вид окна настройки параметров обучения

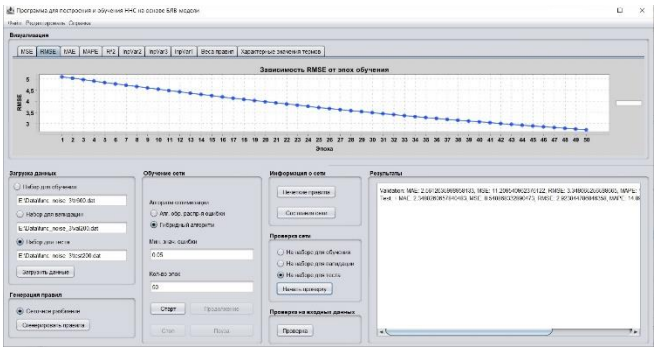

Рис. 2. Вид окна программы на этапе обучения сети

Во время обучения программа отображает динамические графики изменения ключевых метрик качества аппроксимации (как это показано для RMSE на рис. 2), а также графики изменения параметров сети.

V. Заключение

По мнению авторов, разработанная программа является гибким инструментом для решения задач аппроксимации функциональных зависимостей, а также исследования эффективности и возможностей использования ННС на основе БЛВ-модели. По завершении отладки можно планировать использование программы в образовательном процессе и регистрацию в Роспатенте. Кроме того, в дальнейшем планируется расширить набор доступных функций принадлежности, базу алгоритмов генерация правил и обучения сети, а также дополнить список стратегий обучения.

Опыт использования программы в качестве исследовательского инструмента представлен в докладе Г. А. Хамчичева и Г. И. Кожомбердиевой «Опыт построения и обучения нейро-нечеткой сети на основе байесовской логико-вероятностной модели» в настоящем сборнике.

Благодарность

Авторы искренне признательны д-ру физ.-мат. наук, профессору М. М. Луценко за ценные консультации.

Список литературы

[1] Кожомбердиева Г. И., Бураков Д. П., Хамчичев Г. А. Структура нейро-нечеткой сети на основе байесовской логико-вероятностной модели // Мягкие измерения и вычисления. 2022. Т. 61. № 12. С. 52–64.

[2] Kozhomberdieva, G.; Burakov, D. and Khamchichev, G. Neural Network Interpretation of Bayesian Logical-Probabilistic Fuzzy Inference Model. In Proceedings of the 3rd International Symposium on Automation, Information and Computing – ISAIC (Beijing, China, 2022), SciTePress, Vol. 1, pp. 50–56.

[3] Кожомбердиева Г. И., Бураков Д. П., Хамчичев Г. А. О нейросетевой интерпретации байесовской логико-вероятностной модели нечеткого вывода // Международная конференция по мягким вычислениям и измерениям. 2023. Т. 1. С. 25–28.

[4] Хамчичев Г. А., Кожомбердиева Г. И. О настройке и возможностях обучения нейро-нечеткой сети на основе байесовской логико-вероятностной модели // Международная конференция по мягким вычислениям и измерениям. 2023. Т. 1. С. 29–33.

[5] Осовский С. Нейронные сети для обработки информации; [пер. с польск. И. Д. Рудинского]. 2-е изд., перераб. и доп. М.: Горячая линия – Телеком, 2018. 448 с.

[6] Neuro-Adaptive Learning and ANFIS [Электронный ресурс]. URL: https://www.mathworks.com/help/fuzzy/neuro-adaptive-learning-and-anfis.html (дата обращения 07.04.2024).

[7] Кожомбердиева Г. И. Байесовская логико-вероятностная модель нечеткого вывода // Международная конференция по мягким вычислениям и измерениям. 2019. Т. 1. С. 35–38.

[8] Кожомбердиева Г. И., Хамчичев Г. А., Бураков Д. П. Программа для решения задач нечеткого вывода на основе байесовской логико-вероятностной модели. Свидетельство о государственной регистрации программы для ЭВМ RU 2021662943 от 09.08.2021.

[9] Мэрфи К. П. Вероятностное машинное обучение: введение / пер. с англ. А. А. Слинкина. М.: ДМК Пресс, 2022. 940 с.

[10] Бишоп К. М. Распознавание образов и машинное обучение.: Пер. с англ. // СПб.: ООО Диалектика. 2020. 960 с.

[11] Hyndman R. J., Koehler A. B. Another look at measures of forecast accuracy // International Journal of Forecasting. 2006. Vol. 22. No. 4. pp. 679–688.