Math-Net.Ru

Общероссийский математический портал

С. П. Дударов, Н. Д. Кириллов, Модель нейросетевого дефаззификатора заключений в процедурах нечетко-логического вывода и её программная реализация, *Матем. моделирование*, 2020, том 32, номер 8, 91–105

DOI: <https://doi.org/10.20948/mm-2020-08-06>

Использование Общероссийского математического портала Math-Net.Ru подразумевает, что вы прочитали и согласны с пользовательским соглашением

<http://www.mathnet.ru/rus/agreement>

Параметры загрузки:

IP: 54.198.67.100

26 апреля 2023 г., 15:33:03

МОДЕЛЬ НЕЙРОСЕТЕВОГО ДЕФАЗЗИФИКАТОРА ЗАКЛЮЧЕНИЙ В ПРОЦЕДУРАХ НЕЧЕТКО-ЛОГИЧЕСКОГО ВЫВОДА И ЕЁ ПРОГРАММНАЯ РЕАЛИЗАЦИЯ

© 2020 г. *С.П. Дударов, Н.Д. Кириллов**

Российский химико-технологический университет им. Д.И. Менделеева

*nik2010spartak@yandex.ru

DOI: 10.20948/mm-2020-08-06

Представлена математическая модель нейросетевого дефаззификатора. Она представляет собой двухслойный перцептрон и служит для преобразования нечеткого решения в числовую форму в процедурах нечетко-логического вывода. Модель позволяет оптимизировать вычислительную нагрузку, которая возникает при использовании стандартного метода центра тяжести, за счет использования нейронной сети. Было проведено обучение и тестирование с различными настройками нейросетевой модели. Также была доказана эффективность такого подхода с замером времени выполнения вычислительных операций.

Ключевые слова: нейросетевой дефаззификатор, нейросетевая модель, нейронная сеть, дефаззификация, нечетко-логический вывод, математическая модель дефаззификатора.

MATHEMATICAL MODEL OF THE NEURAL NETWORK CONCLUSION DEFUZZIFICATOR IN FUZZY-LOGIC OUTPUT PROCEDURES AND ITS SOFTWARE IMPLEMENTATION

S.P. Dudarov, N.D. Kirillov

Mendeleev University of Chemical Technology of Russia, Moscow

This paper presents a mathematical model of a neural network defuzzificator. It is a two-layer perceptron and serves to convert a fuzzy solution to a numerical form in fuzzy logic derivation procedures. The model allows to optimize the computational load that occurs when using the standard center of gravity method, through the use of a neural network. Training and testing was conducted with various settings of the neural network model. The effectiveness of this approach with measuring the time of computing operations was also proved.

Key words: neural network defuzzificator, neural network model, neural network, defuzzification, fuzzy-logical derivation, mathematical model of a defuzzificator.

1. Введение

В современном информационном обществе постоянное увеличение объемов информации приводит к существенным изменениям в методах ее переработки и анализа, что, в свою очередь, способствует созданию и внедрению эффективных интеллектуальных систем и технологий управления и поддержки принятия решений [1]. Среди них все большую популярность приобретают системы управления и поддержки принятия решений на основе нечеткой логики и теории нечетких множеств, где оперируют данными в виде лингвистических переменных и нечетких чисел, а также осуществляют различные преобразования с использованием специальных операций и процедур нечетко-логического вывода.

Сегодня нечётко-логические системы применяются в различных отраслях: химической, нефтехимической и нефтеперерабатывающей промышленности, автомобилестроении, пищевой сфере, при производстве бытовой техники и цифровых продуктов, на транспорте, в медицине, безопасности и других. Тенденция такова, что такие системы в дальнейшем будут только увеличивать свое распространение, что связано с рядом их важных преимуществ [2, 3]:

- модели нечеткой логики формируются на основе сравнительно простого описания объекта экспертами предметной области и при этом не требуют от экспертов знания глубоких основ точных и естественных наук: физики, химии, математики;

- модели нечеткой логики могут быть более простыми с точки зрения программной и аппаратной реализации, чем классические модели математической физики и химии [2, 3], хотя есть и примеры, показывающие спорность этого утверждения для ряда областей применения и классов решаемых задач [4];

- возможность решения таких задач, которые не могут быть решены классическими методами по причине высокой сложности или полного отсутствия адекватных моделей, которые описывают исследуемые процессы и объекты.

Перспективность использования систем управления и поддержки принятия решений на основе нечеткой логики подтверждается автоконцернами (Nissan, Mitsubishi), активно внедряющими их в производства автомобилей, а именно в системах оптимизации расхода топлива [5] и противоскользких тормозных системах [6]. NASA, например, применяет нечеткие системы для управления процессами при стыковке космических аппаратов [7].

Один из наиболее важных этапов нечетко-логического вывода – дефаз-

зификация нечеткого решения. Она представляет собой процедуру получения числового значения элемента универсального множества из нечеткого значения лингвистической переменной. Дефаззификация может быть более или менее время- и ресурсоемкой процедурой, что, в свою очередь, зависит от выбранного метода [8,9]. При этом самые востребованные методы дефаззификации (методы центра тяжести) являются наиболее сложными и, как следствие, наиболее вычислительно-затратными. К тому же в ряде задач эту процедуру приходится выполнять многократно, что накладывает существенные ограничения для использования процедур нечетко-логического вывода [10,11].

В широко используемых алгоритмах вывода Мамдани или Ларсена процедура дефаззификации применяется к композиции нечётких подзаключений с учётом степеней принадлежности, соответствующих этим подзаключениям. Если каждое подзаключение j описывается функцией принадлежности $\mu_j(x)$, ограниченной соответствующим значением степени принадлежности m_j , то композиция нечётких подзаключений в алгоритме вывода Мамдани может быть описана так [8]:

$$\mu(x) = \bigcup_{j=1}^n (\mu_j(x) \cap m_j). \quad (1)$$

С учетом вида функций $\mu_j(x)$ композиция нечетких подзаключений $\mu(x)$ в общем случае представляет собой кусочно-линейную или кусочно-нелинейную зависимость. Любой из группы методов центра тяжести требует в процессе своего применения численного интегрирования данной зависимости, составленной из фрагментов функций, среди которых могут быть гауссианы и логистические сигмоидные функции. Именно это обуславливает их ресурсоемкость и, как следствие, необходимость сокращения объемов производимых вычислений при сохранении высокой точности дефаззификации.

Решение данной проблемы, представленное в настоящей работе, нацелено на поиск альтернативных методов дефаззификации, эквивалентных методам центра тяжести. В качестве такого решения предлагается применение специальным образом обученных искусственных нейронных сетей – двухслойных перцептронов.

В рамках поставленной цели решаются следующие задачи:

- разработка нейросетевой модели, учитывающей особенности представления входных данных процедуры дефаззификации;
- реализация универсального программного модуля для дефаззификации нечёткой композиции произвольной структуры с целью последующей генерации обучающей и тестовой выборок и обучения перцептрона;

– реализация в виде программно-алгоритмического обеспечения и динамически подключаемой библиотеки модуля расчёта и алгоритма обучения двухслойного перцептрона для дефаззификации нечёткой композиции заданной структуры;

– сравнительный анализ эффективности независимого использования двух модулей дефаззификации и оценка целесообразности практического использования нейросетевой дефаззификации взамен стандартной процедуры.

2. Постановка задачи дефаззификации

2.1. Описание и формализация метода центра тяжести. Метод центра тяжести является одним из наиболее распространённых и, как уже было сказано, часто применяется в алгоритмах Мамдани, Ларсена и Тсукамото для преобразования нечеткого множества в четкое числовое значение. Дефаззификация этим методом осуществляется с помощью соотношения:

$$X^* = \frac{\int_{x_{\min}}^{x_{\max}} x\mu(x)dx}{\int_{x_{\min}}^{x_{\max}} \mu(x)dx}, \quad (2)$$

где X^* – четкое числовое значение результата дефаззификации; x – переменная универсального множества, может принимать значения от $x_{\min}$ до $x_{\max}$; $\mu(x)$ – функция принадлежности.

Значения $x_{\min}$ и $x_{\max}$ ограничивают все нечеткое множество справа и слева соответственно, то есть в этих пределах осуществляется интегрирование. Стоит отметить, что на практике интегрирование осуществляется в пределах носителя объединения подзаключений – элементов нечеткого множества с положительными значениями функции принадлежности [8].

Модификации метода центра тяжести связаны с ограничениями на область определения независимой переменной универсального множества и область допустимых значений подынтегральной функции [8].

Рис.1. Пример композиции трех подзаключений с отображением результата дефаззификации методом центра тяжести.

Графически метод центра тяжести можно представить с помощью рис.1. Интерпретацией данного метода является нахождение горизонтального центра тяжести двумерной фигуры, которая определена границами носителя нечеткого множества, функцией принадлежности [12] и осью независимых переменных универсального множества.

2.2. Выбор входных и выходных переменных нейронной сети. Перед разработкой нейросетевой модели необходимо правильно выбрать входные и выходные переменные. Если с выходным результатом все довольно очевидно – это одна переменная, которая будет хранить в себе четкое числовое значение дефаззификации, то при выборе входных переменных имеются некоторые требования, которые были приняты во внимание:

1. Следует рассмотреть n нечетких подзаключений, и для каждого подзаключения высота равна 1;
2. Следует использовать унимодальные функции принадлежности для каждого нечеткого подзаключения, то есть у каждого подзаключения будет только один максимум;
3. Все функции принадлежности должны относиться к одному математическому классу;
4. Следует использовать непрерывное универсальное множество, которое ограничено слева и справа, как на рис.1;
5. Следует использовать взаимное дополнение функций принадлежности каждого подзаключения нечеткого множества в точках максимумов:

$$\sum_{j=1}^n m_j(x) = 1. \quad (3)$$

Последнее условие введено в целях минимизации количества входных переменных в нейросетевой модели дефаззификатора. Благодаря ему все границы носителей отдельных подзаключений совпадают с координатами высот соседних подзаключений. В этой связи нет необходимости задавать координаты границ носителей и высот через различные входы нейронной сети.

Тем не менее в практике нечётко-логического вывода нередки случаи невыполнения ограничения (3) как в меньшую, так и в большую сторону относительно единицы. Это особенно актуально, когда функции принадлежности нечётких подзаключений получены в результате обобщения множества экспертных оценок. В этом случае проблема снимается введением в нейронную сеть дополнительных входных переменных, соответствующих границам носителей нечётких подзаключений. Это вдвое увеличивает размер входного вектора, как следствие, усложняет структуру нейронной сети

и увеличивает время её обучения, однако при однократном вычислении результата дефаззификации по уже обученной сети замедление вычисления выхода будет практически незаметным.

В настоящей работе рассматривается пример с ограничением (3) с тем, чтобы оценить корректность нейросетевой дефаззификации в случае минимального количества факторов для функций принадлежности, заданных единственным экспертом.

Таким образом, учитывая все требования и формализацию метода центра тяжести, для получения результата дефаззификации с учётом ограничения (3) достаточно иметь набор координат максимумов каждого подзаключения $x_1, x_2, \dots, x_n$ и соответствующее значение функции принадлежности для каждого подзаключения $m_1, m_2, \dots, m_n$. Получается, что для n подзаключений требуется $2n$ входных переменных, которые будут подаваться на вход нейросетевой модели. Выходная переменная, как уже было отмечено, будет соответствовать результату дефаззификации X^* .

2.3. Выбор структуры и настроек нейронной сети. По причине того, что основная цель задачи – снижение объемов вычислений в ходе процедуры нечетко-логического вывода, было принято решение использовать двухслойный перцептрон, который имеет один скрытый слой и один выходной слой. Использование более одного скрытого слоя может нейтрализовать преимущества выбранного подхода и привести к повышению вычислительной нагрузки. С учетом всего сказанного выше, нейросетевая модель имеет вид, который представлен на рис.2.

Рис.2. Нейросетевая модель.

Таким образом, в выходном слое представленной модели имеется только один нейрон, потому что только одно выходное значение получается в результате дефаззификации. В скрытом слое имеется k нейронов. Это число будет варьироваться для достижения оптимальных показателей модели: удовлетворительной вычислительной нагрузки и достаточной точности. С учетом представленной структуры количество весовых коэффициентов (Q) рассчитывается следующим образом:

$$Q = 2k(n + 1) + 1. \quad (4)$$

Кроме основных настроек нейросетевой модели, стоит отметить и настройки сети при обучении и тестировании. Обучение перцептрона осуществляется с помощью метода обратного распространения ошибки, наиболее распространенного для такого типа сетей [13]. Входные данные как при обучении, так и при тестировании сети подавались в нормализованном виде, что делает модель более универсальной.

В качестве активационной функции (f), зависящей от состояния отдельного нейрона s , используется сигмоидная логистическая функция с параметром насыщения $\alpha=1$

$$f(s) = 1 / (\exp(-\alpha s) + 1). \quad (5)$$

В процессе обучения в большинстве случаев коэффициент скорости обучения уменьшается линейно. Для достижения более точных результатов варьировалась, в том числе величина скорости обучения.

2.4. Формирование обучающей и тестовой выборки. При обучении использовалась обучающая выборка, объем которой регулировался по эмпирическому принципу. Сначала значение объема получалось по определенному правилу («правилу 2-5»), согласно которому объем обучающей выборки в примерах рекомендуется выбирать в 2-5 раз большим, чем число весовых коэффициентов [13]. Затем для достижения более точных результатов объем обучающей выборки увеличился и превысил число весовых коэффициентов примерно в 9-16 раз. Таким образом, от количества нейронов в скрытом слое зависит число весовых коэффициентов, что, в свою очередь, влияет на число примеров обучающей выборки.

Величина тестовой выборки была определена равной 20% от величины обучающей выборки. В соответствии с [10,13] такое число тестовых примеров является оптимальным для проверки результатов после обучения сети.

3. Описание программного комплекса

3.1. Организационная и функциональная структуры. При разработке программного комплекса применен объектно-ориентированный подход, который является наиболее распространённым при создании программных средств такого класса, что подтверждается имеющимися исследованиями и разработкой аналогичных программных продуктов [14]. Это позволяет выделить простые элементы, образующие цельную структуру системы.

В нашей работе таким простым элементом является нейрон, т.е. нейрон представляет экземпляр класса, который обладает свойствами и методами, необходимыми для обучения и расчета результата сети. Соответственно, набор экземпляров нейронов образуют слой, а набор слоев, в свою очередь, представляет нейросетевую модель на программном уровне. Поскольку в скрытом слое количество нейронов может варьироваться, это делает использование объектно-ориентированного подхода еще более целесообразным в данной работе.

В ходе решения поставленных задач были разработаны две основные подсистемы программного комплекса:

1. Подсистема расчета результата дефаззификации композиции нечетких подзаключений методом центра тяжести, использующим традиционный подход с интегрированием функции. Здесь рассчитывается точный результат дефаззификации в процессе перебора элементов универсального множества для всех активных нечетких подзаключений. Подсистема служит для последующей генерации обучающей и тестовой выборок.

2. Подсистема расчета результата дефаззификации, использующая нейросетевую модель, которая осуществляет обучение и тестирование сети, а также реализует основную расчетную функцию нейросетевой дефаззификации.

Рис.3. Функциональная структура программного комплекса.

На рис.3 представлена функциональная структура – блоки логического взаимодействия подсистем программного комплекса.

Вторая подсистема также имеет сложную структуру и состоит из нескольких основных блоков, которые и определяют работу всей подсистемы:

1. Блок генерации обучающей и тестовой выборки. В этом блоке случайным образом генерируются пары элементов универсального множества и соответствующих им максимумов степени принадлежности отдельных нечетких подзаключений. Затем набор этих пар подается на вход подсистеме расчета результата дефаззификации, где получается точное решение. Таким образом, формируется пример, который далее используется либо для обучения, либо для тестирования.

2. Блок обучения. После генерации заданный набор примеров подается на обучение. С помощью метода обратного распространения ошибки рассчитываются весовые коэффициенты каждого нейрона. В блоке реализовано случайное перемешивание примеров для каждой эпохи обучения, также имеется возможность изменения параметров скорости обучения сети. По результатам обучения рассчитывается ошибка. Полученные значения весовых коэффициентов сохраняются в системе.

3. Блок тестирования. В нем выполняется генерация примеров для тестирования. В ходе его работы происходит считывание весовых коэффициентов ранее обученной сети и выполняется расчет результата нейросетевой дефаззификации для каждого тестового примера. Также рассчитывается тестовая ошибка нейросетевой модели.

4. Блок дефаззификации. Служит для расчета результата дефаззификации стандартным методом центра тяжести. Этот блок используется как на этапе обучения, так и на этапе тестирования. Использование этого блока необходимо для расчета ошибок обучения и тестирования.

3.2. Обоснование выбора средств разработки. В качестве основного языка разработки был выбран язык C++. Его выбор обусловлен тем, что это очень многофункциональный язык, поддерживаемый огромным сообществом разработчиков. Он предоставляет большие возможности для объектно-ориентированного программирования, а также имеет возможность подключения сторонних библиотек.

Разработка программного комплекса осуществлялась в интегрированной среде разработки Microsoft Visual Studio 2017. Эта среда позволяет эффективно работать с динамической памятью за счет оптимизации ресурсов компьютера.

3.3. Программные и технические характеристики и требования для работы модулей комплекса. В ходе разработки программного комплекса учтена возможность запуска всего модуля в различных операционных системах: в Windows с использованием консоли и в Unix-системах с использованием терминала программа будет работать идентично.

Комплекс может работать на компьютере со следующими минимальными характеристиками и требованиями:

- ОЗУ 4Гб;
- Процессор Intel Core 2 Duo.

Для запуска не требуются специальное программное обеспечение, так как стандартный компилятор установлен по умолчанию практически на всех операционных системах. Тем не менее для более гибкого пользования и удобной настройки программного комплекса рекомендуется установить Visual Studio от компании Microsoft.

4. Работа программного комплекса

Рассмотрим несколько примеров работы программного комплекса. Все примеры были выполнены с использованием стандартного компилятора Visual Studio с вызовом командной строки Windows.

4.1. Начальное меню. При запуске программы в первую очередь появляется меню, в котором пользователь может выбрать, какой этап использования нейросетевой модели выполнить. Пример меню изображен на рис.4.

```

-----Нейронная сеть < Обратное распространение ошибки >-----
Выберите режим:
1 - Обучение сети
2 - Проверка сети <тестовая выборка/тестовый пример>
3 - Справка
>

```

Рис.4. Главное меню программного комплекса.

Для пользователя также предусмотрено справочное описание, которым он может воспользоваться в режиме реального времени.

4.2. Пример генерации и работы модуля без использования нейросетевой модели. На рис.5 продемонстрирован результат работы функции генерации примеров. Для демонстрации использовался набор входных данных, состоящий из пяти пар значений элемента универсального множества, соответствующий вершине Λ -функции и степеней принадлежности нечеткого подзаключения. Результатом дефаззификации, рассчитанным традиционно и принимаемым за эталон, выступает последнее значение в каждой строке.

```

-8 : 0.7 : -4 : 0.4 : 0 : 0.2 : 5 : 0 : 10 : 0 : -3.85 :
-10 : 0 : -3.2 : 1 : -0.5 : 0.8 : 2.2 : 0.2 : 9.4 : 0 : -2.501 :
-9.5 : 0 : -2.4 : 0 : 1.8 : 0 : 4.6 : 0.1 : 7.7 : 0 : 4.7426 :
-8.8 : 0 : -5.6 : 0.7 : -1.7 : 0.2 : 2.9 : 0.7 : 9.3 : 0 : 1.0167 :
-7.8 : 0.2 : -5.3 : 0.2 : 1.5 : 0 : 5.9 : 0 : 6.4 : 0 : -3.49 :
-6 : 0 : -4.5 : 0 : -1.9 : 0.8 : 5.5 : 0.2 : 7 : 0 : -0.1944 :
-9.3 : 0 : -4.2 : 0 : -0.9 : 0 : 2.7 : 0.9 : 8.3 : 0 : 3.2501 :
-8.4 : 0 : -4.5 : 0 : -0.9 : 0.4 : 4.1 : 0 : 8.8 : 0 : -0.34 :

```

Рис.5. Фрагмент обучающей или тестовой выборки.

На рис.6 приведен пример графического представления композиции нечетких подзаключений для первой строки данных из фрагмента выборки.

Рис.6. Пример графического представления для одного набора входных данных.

4.3. Пример обучения сети. При выборе пункта обучения нейронной сети появляется следующее меню (рис.7), в котором программа позволяет задать количество нейронов в скрытом слое, количество обучающих эпох, количество примеров, которые следует сгенерировать для обучения.

Остальные редко меняющиеся параметры, такие как скорость обучения или параметр насыщения сигмоидной логистической функции, задаются программно через конфигурационный файл.

```

-----Нейронная сеть < Обратное распространение ошибки >-----
Выберите режим:
1 - Обучение сети
2 - Проверка сети <тестовая выборка/тестовый пример>
3 - Справка
>1
Обучение сети, параметры по умолчанию:
-----
Количество нейронов < 'neuron' > = 20
Количество обучающих эпох < 'count' > = 40000
Количество примеров для обучения < 'N' > = 950
Режим: обучение нейронной сети
-----
>
    
```

Рис.7. Окно настройки обучения сети.

Ошибка = 0.0199955	-----	607
Ошибка = 0.0205508	-----	608
Ошибка = 0.0199288	-----	609
Ошибка = 0.0198196	-----	610
Ошибка = 0.019946	-----	611
Ошибка = 0.0199945	-----	612
Ошибка = 0.0195974	-----	613

Рис.8. Процесс обучения с выводом информации каждой эпохи.

Чтобы начать обучение, нужно ввести команду «start» в командном окне и нажать клавишу Enter. В процессе обучения происходит вывод номера

эпохи и соответствующей ей ошибки обучения. Пример такого вывода изображен на рис.8. Ошибка рассчитывается для нормализованных примеров. Таким образом, можно легко делать выводы и управлять процессом настройки параметров и структуры нейронной сети вне зависимости от происхождения и масштаба переменных.

4.4. Пример тестирования сети. В разработанном программном комплексе тестирование запускается, с точки зрения пользователя, практически идентично предыдущему случаю. Однако вывод результатов при тестировании несколько отличается. В командное окно выводятся результаты работы модулей как с использованием нейросетевой модели, так и без нее, а также в нормализованном и денормализованном видах для каждого примера. Затем выводится усредненная по всем примерам ошибка. Пример работы изображен на рис.9.

```

Нормализованные выходы сети - реальный: 0.652979 --- 0.65871
Денормализованные выходы сети - реальный: 3.05957 --- 3.1742

Нормализованные выходы сети - реальный: 0.131141 --- 0.1365
Денормализованные выходы сети - реальный: -7.37719 --- -7.27

Нормализованные выходы сети - реальный: 0.702169 --- 0.687315
Денормализованные выходы сети - реальный: 4.04337 --- 3.7463

Нормализованные выходы сети - реальный: 0.504731 --- 0.47695
Денормализованные выходы сети - реальный: 0.0946262 --- -0.461

Нормализованные выходы сети - реальный: 0.867393 --- 0.875455
Денормализованные выходы сети - реальный: 7.34786 --- 7.5091

Нормализованная ошибка = 0.0182324
Денормализованная ошибка = 0.364647

```

Рис.9. Процесс тестирования с выводом информации для каждого примера, а также усредненных значений ошибки в нормализованном и денормализованном виде.

5. Оценка эффективности модуля нейросетевой дефаззификации

5.1. Оценка ошибки сети. В первую очередь необходимо оценить ошибку, которая была получена при использовании нейросетевой модели. Очевидно, что без достижения удовлетворительного значения ошибки не имеет смысла сравнивать модули по времени, затраченному на расчёты.

Итак, в качестве тестовых данных был выбран диапазон элементов универсального множества $[-10; 10]$, на котором выделено пять нечетких подзаключений. Ошибка представлена в нормализованном виде в табл.1 для нескольких вариантов количества нейронов в скрытом слое.

Как видно из этой таблицы, ошибка обучения варьируется от 0.39% до 0.71%, а ошибка тестирования – от 1.2% до 2.3%. Такие результаты можно считать удовлетворительными, и сделать вывод о том, что нейросетевая модель может быть использована для получения результата дефаззификации.

Стоит добавить, что такие результаты были получены при заданных, но специально не подбиравшихся настройках сети: скорость обучения, ко-

личество обучающих примеров и т.д. Вероятно, выходные значения, варьируя эти настройки, можно подобрать более близкими к оптимальным и получить наименьшую ошибку.

Таблица 1. Значения ошибок обучения и тестирования нейросетевой модели при различных значениях количества нейронов в скрытом слое.

Количество нейронов	Средняя ошибка обучения, %	Средняя ошибка теста, %	Отношение числа примеров обучения (теста) к числу связей
10	0.71	2.31	16.53 (6.61)
15	0.68	1.91	11.36 (4.55)
18	0.71	1.88	9.57 (3.83)
20	0.52	1.71	11.26 (3.46)
22	0.51	1.67	10.27 (3.16)
25	0.45	1.62	13.99 (2.8)
30	0.44	1.33	15.54 (2.35)
35	0.39	1.246	15.15 (2.02)

5.2. Оценка временных затрат как характеристика вычислительной нагрузки. После получения удовлетворительной величины ошибки стоит сравнить разработанные модули по вычислительной нагрузке. Однако оценить объем вычислений проще всего через затраченное на это время.

Для сравнения были рассмотрены две нейросетевые модели, состоящие из 20 и 35 нейронов в скрытом слое. Такой выбор был сделан по причине того, что первая модель соответствует средней величине ошибки, а вторая модель – минимальной ошибке из табл.1. Были использованы несколько примеров с расчетом по нейросетевому модулю и с помощью стандартных методов численного интегрирования функций. Время измерялось с помощью встроенных инструментов Visual Studio и языка C++. Полученные и представленные в табл.2 результаты совершенно очевидно отображают пре-

Таблица 2. Время выполнения программного модуля с применением нейросетевой модели и методом центра тяжести.

Модуль	Время на 1000 примеров, с	Время на 10000 примеров, с	Время на 100000 примеров, с
С нейросетевой моделью – 20 нейронов	0.015	0.127	0.993
С нейросетевой моделью – 35 нейронов	0.023	0.187	1.423
Стандартный расчет методом центра тяжести	0.391	4.131	40.949

имущество использования нейросетевой дефаззификации. В зависимости от количества нейронов и примеров нейросетевая модель выигрывает по времени в несколько десятков раз. Наибольший эффект от такого выигрыша, разумеется, достигается не при разовых дефаззификациях, а в ходе их многократного циклического повторения, например, для вывода решения в нечетких клеточных автоматах [11].

6. Заключение

Разработан программный комплекс для выполнения дефаззификации в процедурах нечетко-логического вывода традиционным методом центра тяжести и с использованием двухслойного перцептрона.

Проведено сравнение двух методов дефаззификации: традиционного и нейросетевого. Полученные результаты свидетельствуют о существенном выигрыше нейросетевой дефаззификации по времени, а также о хорошем уровне ошибки расчета для нее.

Таким образом, нейросетевой модуль является эффективным инструментом, который целесообразно применять в многократно повторяющихся процедурах нечетко-логического вывода. Вместе с тем, его использование требует предварительного подбора настроек и структуры нейронной сети и ее продолжительного обучения. Представленный в работе подход можно развивать далее за счет подбора оптимальных настроек нейросетевой модели, которые могут еще значительно уменьшить ошибку при существенном выигрыше в затраченном времени.

Разработанный программный модуль нейросетевой дефаззификации может быть эффективно использован в качестве подключаемой библиотеки при разработке информационно-аналитических, информационно-моделирующих, экспертных систем, систем поддержки принятия решений, математический аппарат которых базируется на процедурах нечетко-логического вывода решений.

СПИСОК ЛИТЕРАТУРЫ

1. *Е.В. Суслова.* Интеллектуальные системы поддержки принятия решений // Молодой ученый, 2017, №3, с.171-174;
E.V. Suslova. Intellektualnye sistemy podderzhki priniatiia reshenii // Molodoi uchenyi, 2017, №3, s.171-174.
2. *А.А. Байченко, Л.А. Байченко, В.А. Арет.* Применение нечеткой логики в управлении предприятием пищевой промышл. // Науч. журнал НИУ ИТМО, 2014, №3, с.35-69;
A.A. Baichenko, L.A. Baichenko, V.A. Aret. Primenenie nechetkoi logiki v upravlenii predpriatiem pishchevoi promyshlennosti // Nauch. zhurnal NIU ITMO, 2014, №3, s.35-69.
3. *В.Б. Успенский.* Современная теория управления. Методы синтеза и оптимизации систем управления. Конспект лекций для студентов специальностей 7.080202 и

- 7.080201 / В.Б. Успенский, Л.В. Шипулина. – Х.: НТУ «ХПИ», 2013, 136 с.
V.B. Uspenskii. Sovremennaiia teoriia upravleniia. Metody sinteza i optimizatsii sistem upravleniia. Konspekt lektsii dlia studentov specialnostei 7.080202 i 7.080201 / V.B. Uspenskii, L.V. Shipulina. – KH.: NTU «KHPI», 2013, 136 s.
4. *И.С. Коберси, А.В. Кияшко, Е.А. Македонов, Е.Р. Крамаренко, В.И. Финаев. Система управления напряжением генератора на базе нечёткой логики // Инженерный вестник Дона, 2015, №2-2, с.1-14;*
I.S. Kobersi, A.V. Kiiashko, E.A. Makedonov, E.R. Kramarenko, V.I. Finaev. Sistema upravleniia napriazheniem generatora na baze nechetkoi logiki // Inzhenernyi vestnik Dona, 2015, №2-2, s.1-14.
5. *R. Syahputra. Application of neuro-fuzzy method for prediction of vehicle fuel consumption // J. of Theoretical & Applied Information Technology, 2016, v.86, №1, p.138-150.*
6. *R.C. David, R.B. Grad, R.E. Precup, M.B. Rădac, C.A. Dragoş, E.M. Petriu. An approach to fuzzy modeling of anti-lock braking systems // Soft Computing in Industrial Applications, Springer, Cham, 2014, v. 223, p.83-93.*
7. *L. Sun, W. Huo. Adaptive fuzzy control of spacecraft proximity operations using hierarchical fuzzy systems // IEEE/ASME Trans. Mechatronics, 2016, v.21, № 3, p.1629-1640.*
8. *С.П. Дударов, П.Л. Панаев. Алгебра нечёткой логики и анализ нечётких множеств: учеб. пособие. – М.: РХТУ им. Д.И. Менделеева, 2019, 84 с;*
S.P. Dudarov, P.L. Papaev. Algebra nechetkoi logiki i analiz nechetkikh mnozhestv: ucheb. posobie. – М.: RKHTU im. D.I. Mendeleeva, 2019, 84 s.
9. *W.T. Dobrosielski, J. Szczepański, H. Zarzycki. A proposal for a method of defuzzification based on the golden Ratio–GR // Novel Developments in Uncertainty Representation and Processing, Springer, Cham, 2016, p.75-84.*
10. *Н.С. Ермолаев, Н.Д. Кириллов, С.П. Дударов. Модель универсального нейросетевого дефаззификатора // Успехи в химии и хим. технологии, 2017, т.31, №8 (189), с.66-68;*
N.S. Ermolaev, N.D. Kirillov, S.P. Dudarov. Model universalnogo neirosetevogo defazifikatora // Uspekhi v khimii i khimicheskoi tekhnologii, 2017, t.31, №8 (189), s.66-68.
11. *А.Н. Кудряшов. Моделирование рассеяния примеси в атмосфере с использованием нечеткого клеточного автомата / А.Н. Кудряшов, С.П. Дударов. Успехи в химии и хим. технологии, т. XXVI, №1 (130). – М.: РХТУ им. Д.И. Менделеева, 2012, с.24–28;*
A.N. Kudriashov. Modelirovanie rasseianiia primesi v atmosfere s ispolzovaniem nechetkogo kletochnogo avtomata / A.N. Kudriashov, S.P. Dudarov. Uspekhi v khimii i khim. Tekhnologii, t.XXVI, № 1 (130). – М.: RKHTU im. D.I. Mendeleeva, 2012, s.24–28.
12. *M.M. Adnan, A. Sarkheyli, A.M. Zain, H. Haron. Fuzzy logic for modeling machining process: a review // Artificial Intelligence Review, 2015, v.43, № 3, p.345-379.*
13. *С.П. Дударов, П.Л. Панаев. Теоретические основы и практическое применение искусственных нейронных сетей. Уч. пособие. – М.: РХТУ им. Д.И. Менделеева, 2014, 104с;*
S.P. Dudarov, P.L. Papaev. Teoreticheskie osnovy i prakticheskoe primenenie iskusstvennykh neironnykh setei. Uch. posobie. – М.: RKHTU im. D.I. Mendeleeva, 2014, 104 s.
14. *С.О. Янчи, С.П. Дударов. Использование объектно-ориентированного подхода при разработке программных средств нейросетевого моделирования / Успехи в химии и хим. технологии, т. XXII, № 1 (81). – М.: РХТУ им. Д.И. Менделеева, 2008, с.55–59;*
S.O. Ianchi, S.P. Dudarov. Ispolzovanie obieektno-orientirovannogo podkhoda pri razrabotke programmnykh sredstv neirosetevogo modelirovaniia / Uspekhi v khimii i khim. tekhnologii, t. XXII, № 1 (81). – М.: RKHTU im. D.I. Mendeleeva, 2008, s.55–59.

Поступила в редакцию 14.10.2019
 После доработки 14.10.2019
 Принята к публикации 25.11.2019