УДК 004.81

МОДЕЛИРОВАНИЕ ГИБРИДНОЙ НЕЧЕТКОЙ СЕТИ НА БАЗЕ SCIKIT-FUZZY

Долженко А.И.

доктор экономических наук, доцент

alexdoljenko@mail.ru

Ермолов И.А.

магистрант

ermolov962104@gmail.com

Аннотация: Дана краткая характеристика гибридной нечеткой сети, а также ее преимущества перед классическими нейронными сетями и системами нечеткого вывода. Представлена структура модели нечеткой гибридной сети и приведен пример её реализации на базе библиотеки scikit-fuzzy.
Ключевые слова: гибридная нечеткая сеть, нейронные сети, scikit-fuzzy, системами нечеткого вывода.

SIMULATION OF A HYBRID FUZZY NETWORK ON THE SCIKIT-FUZZY BASIS

Dolzhenko A.I.

Ermolov I.A.

Abstract: A brief description of the hybrid fuzzy network is given, as well as its advantages over classical neural networks and fuzzy inference systems. The model structure of a fuzzy hybrid network is presented and an example of its implementation using the scikit-fuzzy library is given.
Keywords: hybrid fuzzy network, neural networks, scikit-fuzzy, fuzzy inference systems.

На сегодняшний день существует множество подходов к исследованию сложных систем на базе моделей искусственного интеллекта. Одни из самых популярных подходов являются нейронные сети. Нейронные сети хороши для задач прогнозирования, классификации, кластеризации и распознавания образов, но весьма неудобны для выяснения вопроса, как они это осуществляют. С другой стороны, существуют системы с нечеткой логикой,

которые объясняют полученные ими выводы, но они не могут автоматически приобретать знания для использования их в механизмах выводов. Для использования преимуществ обоих систем был предложен подход гибридных сетей [1]. Для построения нейронных сетей многие исследователи используют язык Python и различные библиотеки, например, Keras. Библиотека Scikit-fuzzy предназначена для моделирования нечетких моделей, но она не предполагает возможности обучения [2]. В связи с этим актуальной является задача разработки на языке Python модели для гибридных или нейро-нечетких сетей.

Гибридная нечеткая сеть представляет собой нейроны различных слоев, связанные между собой определенным образом (рисунок 1).

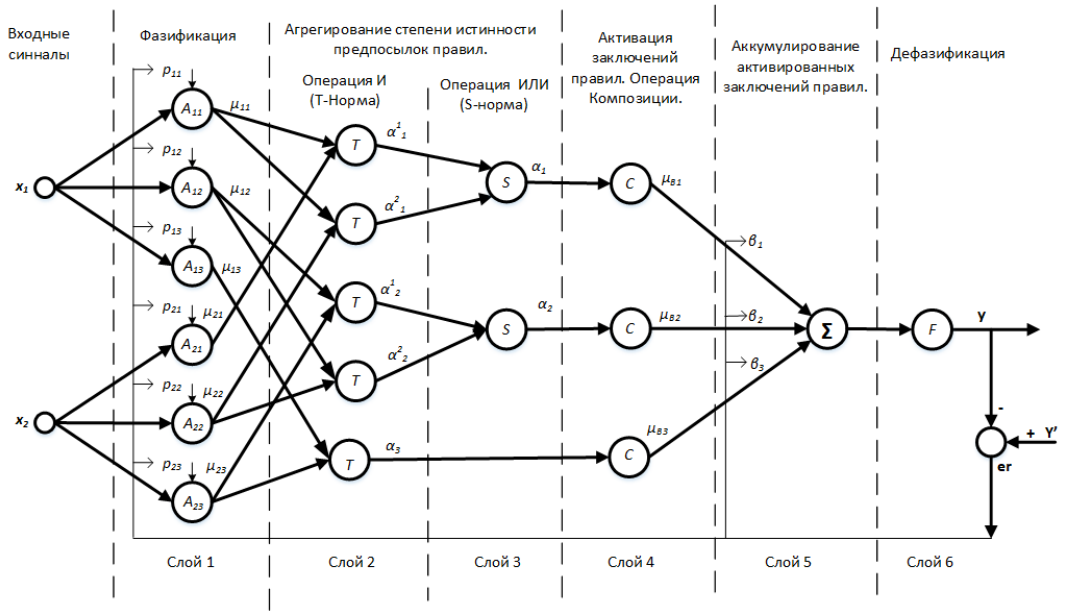

Рисунок 1 – Гибридная нечеткая сеть

Слой 1 фазификации – преобразовывает четкие входные значения в значения функции принадлежности соответствующего ей терма лингвистической переменной [3]. Количество нейронов соответствует суммарному количеству термов лингвистических переменных. На выходе слоя формируется значения степени принадлежности входных переменных нечетким множествам, описывающим термы лингвистических переменных.

Слой 2 агрегации (Т-норма) - Реализует агрегирование степени истинности предпосылок правил базы знаний (Конъюнкция). Количество нейронов соответствует количеству правил.

Слой 3 агрегации (S-норма) - Реализует агрегирование степени истинности предпосылок правил базы знаний (Дизъюнкция). Количество нейронов соответствует количеству термов консеквента.

Слой 4 активации – Реализует активацию (композицию) заключений правил. Количество нейронов соответствует количеству термов консеквента.

Слой 5 аккумулирования – Реализует аккумулирование активированных заключений правил. Количество нейронов – один.

Слой 6 дефазификации – Реализует преобразование нечеткого вывода в четкое значение выхода модели. Количество нейронов – один.

При разработке программного инструментария для гибридной нечеткой модели ставилась задача построить модель типа MISO (Multi Inputs – Single Output), которая имеет несколько входов и один выход. Модель должна реализовывать алгоритм нечеткого вывода Мамдани и обучаться градиентным методом, используя алгоритм обратного распространения ошибки [4].

Разработанная программная структура классов гибридной модели на языке Python соответствует структуре слоев нечеткой сети. Так же существует класс самой сети «FuzzyNetwork» и служебные классы для описания сущностей: антецедент – «Antecedent», консеквент – «Consequent», нечеткие правила – «Rule» и т. д.

Для первичной настройки сети необходимо создать входные антецеденты, выходной консеквент. Инициализировать их термы начальными параметрами. Количество параметров зависит от используемой функции принадлежности (треугольная, трапецеидальная, гауссова, сигмоидальная и другие). Затем необходима создать базу правил. После этого создается экземпляр гибридной сети, в которую при инициализации передается база правил. На следующем шаге вызывается метод активации

сети, в который передается вектор входных параметров, каждый из которых связан с консевентом.

Код инициализации гибридной сети представлен на рисунке 2.

```
quality = Antecedent("Качество еды", np.arange(0, 11, 1))
service = Antecedent("Качество обслуживания", np.arange(0, 11, 1))
tip = Consequent("Чаевые", np.arange(0, 26, 1))

quality["Плохо"] = [0, 0, 5]
quality["Средне"] = [0, 5, 10]
quality["Хорошо"] = [5, 10, 10]
service["Плохо"] = [0, 0, 5]
service["Средне"] = [0, 5, 10]
service["Хорошо"] = [5, 10, 10]
tip["Плохо"] = [0, 0, 13]
tip["Средне"] = [0, 13, 25]
tip["Хорошо"] = [13, 25, 25]

rules = [
    Rule(quality["Плохо"] & service["Плохо"], tip["Плохо"]),
    Rule(quality["Средне"] & service["Средне"], tip["Средне"]),
    Rule(quality["Средне"] & service["Хорошо"], tip["Хорошо"]),
    Rule(quality["Хорошо"] & service["Хорошо"], tip["Хорошо"]),
]

net = FuzzyNetwork(rules)
net.activate([quality.connect(6.5), service.connect(9.8)])
```

Рисунок 2 – Код инициализации гибридной сети

Класс «FuzzyNetwork» принимает так же и необязательные параметры которые соответствуют слоям сети. Конструктор класса «FuzzyNetwork» представлен на рисунке 3.

```
def __init__(
        self,
        rules: List[Rule],
        fuzzification_class: Type[FuzzificationLayer] = TrimFuzzificationLayer,
        t_norm_aggregation_class: Type[AggregationLayer] = MinAggregationLayer,
        s_norm_aggregation_class: Type[AggregationLayer] = MaxAggregationLayer,
        activation_class: Type[ActivationLayer] = MinActivationLayer,
        accumulation_class: Type[AccumulationLayer] = MaxAccumulationLayer,
        defuzzification_class: Type[
            DefuzzificationLayer
        ] = CentroidDefuzzificationLayer,
    ):
```

Рисунок 3 – Конструктор класса «FuzzyNetwork»

В конструкторе происходит извлечение антицендентов и консеквентов и переданных правил. После этого в зависимости от извлеченных данных происходит инициализация нейронов всех слоев.

Обучение происходит с помощью метода обратного распространения ошибки. При обучении системы корректируются параметры функции

активации для слоя фазификации (ρ_{ij}), она же функция принадлежности, а также коэффициенты доверия слоя активации заключений правил (b_i) [5].

В результате проведенной работы была разработана программная реализация гибридной нечеткой сети, на языке Python и библиотеки scikit-fuzzy. Модель с прямым распространением сигнала реализует вывод по алгоритмы Мамдани [6]. Для обучения использован градиентный метод.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Нурматова Е.В., Ровинец Г.О., Сидельников А.В. Программная реализация системы нечеткого логического вывода по алгоритму Мамдани // Вестник компьютерных и информационных технологий. – 2012. – №5(95). – С. 31 – 36.
2. skfuzzy 0.2 docs // https://pythonhosted.org/scikit-fuzzy/.
3. Нечеткое моделирование и управление / А. Пегат; пер с англ. – 2-е изд. – М.: БИНОМ. Лаборатория знаний, 2013. – 789 с. – С 180-182.
4. Храмов В.В. Нечетко-множественный подход в маркетинговых исследованиях // Интеллектуальные ресурсы – региональному развитию. – 2019. – Т. 5, №1. – С. 110 – 115.
5. Борисов В.В., Круглов В.В., Федулов А.С. Нечеткие модели и сети. – М.: Горячая линия – Телеком, 2007. – 284 с.
6. Латыпов Р.Р. Модели нечеткого сопоставления при решении задач маркетинга // Интеллектуальные ресурсы – региональному развитию. 2019. – Т. 5, №1. – С. 75 – 80.