ИНФОРМАТИКА И ИНФОРМАЦИОННЫЕ ПРОЦЕССЫ

УДК 004.83 Научная статья
DOI: 10.35330/1991-6639-2023-6-116-109-115
EDN: FWBYMM

Разработка алгоритма на основе логических операций для обнаружения закономерностей в неполных данных

Л. А. Лютикова

Институт прикладной математики и автоматизации –
филиал Кабардино-Балкарского научного центра Российской академии наук
360000, Россия, г. Нальчик, ул. Шортанова, 89 А

Аннотация. В данной работе представлен метод локальной интерпретации решений обученной нейронной сети функциями многозначной логики предикатов. Локальная интерпретация нейронной сети относится к процессу объяснения решений, принимаемых моделью на конкретном примере или в окрестности конкретного входа. В основе предлагаемого подхода лежит множество функций многозначной логики, которые представляют собой обобщенные операции, отвечающие определенным требованиям. Комбинируя эти функции, можно обнаружить внутренние закономерности в данных и даже корректировать результаты, полученные с помощью нейронных сетей. Предложенный метод был исследован в контексте задач классификации с использованием многомерных дискретных признаков. В таких случаях каждый признак может принимать одно из k возможных значений и иметь равную важность для идентификации класса. Этот подход открывает новые возможности для понимания и объяснения правил, лежащих в основе данных, которые не всегда очевидны при использовании обычных нейронных сетей.

Ключевые слова: интерпретация, многозначная логика, нейронная сеть, обобщенное сложение, данные

Поступила 30.10.2023, одобрена после рецензирования 01.11.2023, принята к публикации 10.11.2023

Для цитирования. Лютикова Л. А. Разработка алгоритма на основе логических операций для обнаружения закономерностей в неполных данных // Известия Кабардино-Балкарского научного центра РАН. 2023. № 6(116). С. 109–115. DOI: 10.35330/1991-6639-2023-6-116-109-115

MSC: 68T27 Original article

Development of an algorithm based on logical operations to detect patterns in data with missing values

L.A. Lyutikova

Institute of Applied Mathematics and Automation –
branch of Kabardino-Balkarian Scientific Center of the Russian Academy of Sciences
360000, Russia, Nalchik, 89 A Shortanov street

Abstract. This paper presents a method of local interpretation of solutions of a trained neural network by functions of multivalued predicate logic. The local interpretation of a neural network refers to the process of explaining the decisions made by the model on a specific example or in the vicinity of a specific input. The proposed approach is based on a set of functions of multivalued logic, which are generalized operations that meet certain requirements. By combining these functions, it is possible to detect internal

patterns in the data and even correct the results obtained with the help of neural networks. The proposed method was investigated in the context of classification problems using multidimensional discrete features. In such cases, each attribute can take one of k possible values and have equal importance for class identification. This approach opens up new possibilities for understanding and explaining the rules underlying the data, which are not always obvious when using conventional neural networks.

***Keywords**:* interpretation, multivalued logic, neural network, generalized addition, data

Submitted 30.10.2023, approved after reviewing 01.11.2023, accepted for publication 10.11.2023

For citation. Lyutikova L.A._Development of an algorithm based on logical operations to detect patterns in data with missing values. *News of the Kabardino-Balkarian Scientific Center of RAS.* 2023. No. 6(116). Pp. 109–115. DOI: 10.35330/1991-6639-2023-6-116-109-115

Введение

Методы интерпретации работы нейронной сети позволяют анализировать и понимать, как и почему сеть принимает определенные решения или делает прогнозы.

Логические методы интерпретации линейной нейронной сети обычно основаны на анализе весов и пороговых значений модели. Поскольку линейная нейронная сеть представляет собой комбинацию линейных операций (умножение на веса) и нелинейных функций активации, интерпретация может быть связана с определением важности каждого признака или переменной в модели. Локальные методы фокусируются на объяснении решений модели на конкретных примерах или входах, в то время как глобальные методы интерпретации стремятся предоставить более общую и глобальную информацию о модели. Оба типа методов имеют свои преимущества и могут быть полезны в разных ситуациях в зависимости от поставленной задачи интерпретации.

В работе предлагается генерация интерпретируемых функций. Предлагается подобрать множество логических функций многозначной логики предикатов или на основе построения деревьев, которые воспроизводят поведение нейронной сети и позволяют объяснить принимаемые решения.

Методы интерпретации работы нейронной сети на основе построения деревьев позволяют создать понятную и интерпретируемую модель, которая объясняет принимаемые решения сети [1–4].

Постановка задачи

Допустим, у нас есть объекты, которые характеризуются набором свойств или признаков. Каждый объект представлен в виде многомерного вектора, где каждая координата соответствует значению одной из характеристик объекта. Однако информация о некоторых характеристиках может отсутствовать [5].

Задачей линейной нейронной сети является способность принимать входные данные (набор признаков объекта) и выдавать соответствующий выходной результат.

$X = \{x_1, x_2, ..., x_n\}$, где $x_i \in \{0,1,...,k_r - 1\}$, $k_r \in [2,...,N], N \in Z$ – входные данные $X_i = \{x_1(y_i), x_2(y_i), ..., x_n(y_i)\}$, $i = 1,...,n$, $y_i \in Y, Y = \{y_1, y_2, ..., y_m\}$ – выходные данные:

$$\begin{pmatrix} x_1(y_1) & x_2(y_1) & ... & x_n(y_1) \\ x_1(y_2) & x_2(y_2) & ... & x_n(y_2) \\ ... & ... & ... & ... \\ x_1(y_m) & x_2(y_m) & ... & x_n(y_m) \end{pmatrix} \to \begin{pmatrix} y_1 \\ y_2 \\ ... \\ y_m \end{pmatrix}.$$

Каждый нейрон в сети имеет свой набор весов, который позволяет ему распознавать элементы заданной предметной области, основываясь на наборе признаков. Важным преимуществом является использование прямых сумм для формирования архитектуры и настройки параметров сети без необходимости решения сложных оптимизационных задач.

Это позволяет аппроксимировать зависимость между входными и выходными данными и использовать нейронную сеть для обработки и анализа объектов в заданной предметной области [7].

Задача логической интерпретации прямых сумм нейронной сети заключается в нахождении функций многозначной логики, которые при применении к тем же входным данным, что и нейронная сеть, дают те же самые выходные значения.

Другими словами, мы стремимся найти логические выражения или правила, которые могут точно описать и объяснить решения, принимаемые нейронной сетью. Это позволяет нам иметь более понятное и интерпретируемое представление работы сети, используя понятные логические операции и правила.

Таким образом, логическая интерпретация прямых сумм нейронной сети помогает нам перевести сложные вычисления и принятие решений сети в понятные логические формулы или правила, что может быть полезным для понимания и объяснения работы сети, а также проверки и верификации ее действий [6].

Алгоритм нахождения функций, соответствующих прямой сумме

Рассмотрим систему многозначной логики, на которой определены операции $L = \left\langle L, \bar{x}, \&, \vee, \bar{}, \to \right\rangle$, $L = \{0,1,...,k-1\}$, множество функций обобщенного сложения, обозначаемое как Σ. Цель состоит в том, чтобы найти это множество функций, удовлетворяющих условию $Y = f(X)$.

Все искомые функции $\sigma(x, y)$ должны обладать следующим свойством: $\sigma(x,0) = \sigma(0, x) = x$, будем называть их функциями обобщенного сложения.

Требуется найти множество таких функций Σ, которые удовлетворяют еще одному условию – $\Sigma(a_1,...,a_{n-1}, a_n) = y$.

Алгоритм строим в виде дерева решений.

Предположим, что у нас есть *k* возможных значений для переменной, где *k* может быть любым числом. Хотя конкретное значение неизвестно, оно должно принимать одно из *k* возможных значений. Следовательно, у нас есть *k* разветвлений в дереве, соответствующих этим возможным значениям.

Мы можем представить это в виде условий, где каждое условие проверяет значение и перенаправляет нас по соответствующей ветке дерева. Условия могут быть записаны следующим образом:

Будем считать, что $\Sigma(a_1,...,a_{n-1}, a_n) = \Sigma(a_1,...,a_{n-1}) + a_n = \sigma(\Sigma(a_1,...,a_{n-1}), a_n)$. Величина $\Sigma(a_1,...,a_{n-1})$ неизвестна, но она должна принимать одно из значений $L = \{0,1,...,k-1\}$. Будет выполнено одно из *k* условий:

$$\Sigma(a_1,...,a_{n-1}) = 0,\ \Sigma(a_1,...,a_{n-1}) = 1\ ,\ldots,\ \Sigma(a_1,...,a_{n-1}) = k-1. \quad (1)$$

Значит,

$$\Sigma(0, a_n) = y,\ \Sigma(1, a_n) = y,\ \ldots,\ \Sigma(k-1, a_n) = y, \quad (2)$$

т.е. получаем *k* разветвлений дерева.

На следующем шаге рассмотрим следующее соотношение $\Sigma(a_1,...,a_{n-1}) = \Sigma(a_1,...,a_{n-2}) + a_{n-1} = \sigma(\Sigma(a_1,...,a_{n-2}), a_{n-1})$ с учетом сформулированных ранее предположений относительно значений $\Sigma(a_1,...,a_{n-1})$, т.е. каждая из ветвей (2) разобьется в свою очередь еще на *k*.

Для построения этих функций мы можем создать дерево решений, которое будет иметь *k* разветвлений, где *k* – количество возможных значений для переменной. Каждое разветвление в дереве будет соответствовать одному из возможных значений переменной. Мы проверяем значение переменной и переходим по соответствующей ветви дерева, чтобы определить, какую функцию из множества Σ использовать.

Таким образом, каждое условие в дереве будет проверять значение переменной и направлять нас по соответствующей ветви в дереве, где мы применяем соответствующую функцию обобщенного сложения из множества Σ [8].

Это позволяет нам построить множество функций, которые учитывают различные значения переменной и обеспечивают гибкость и адаптивность в принятии решений на основе входных данных.

Пример 1. Рассмотрим систему трехзначной логики, где мы хотим построить решающую функцию на основе множества функций обобщенного сложения Σ $L = \left\langle L, \bar{x}, \&, \vee, \bar{}, \to \right\rangle$, $L = \{0,1,2\}$. Для достижения этой цели мы создаем дерево решений, которое имеет несколько ветвлений, соответствующих возможным значениям переменной. Каждое ветвление в дереве представляет определенное значение переменной, и на каждом уровне дерева мы применяем соответствующую функцию обобщенного сложения из множества Σ.

Пусть задано: $\Sigma\left(1,0,\frac{1}{2},1\right) = \frac{1}{2}$.

Ветвь 1. $\Sigma\left(1,0,\frac{1}{2}\right) = 0 \Rightarrow$ необходимо, чтобы $\Sigma(0,1) = \frac{1}{2}$, но это противоречит условию $\sigma(x,0) = \sigma(0,x) = x$. Значит, путь в дереве не существует.

Ветвь 2. $\Sigma\left(1,0,\frac{1}{2}\right) = \frac{1}{2} \Rightarrow \Sigma\left(\frac{1}{2},1\right) = \frac{1}{2}$. Путь существует. $\Sigma(1,0) = 0$ – невозможно, путь в дереве не существует.

2. $\Sigma(1,0) = \frac{1}{2}$ – невозможно, путь в дереве не существует.

3. $\Sigma(1,0) = 1$ – возможно.

Продолжаем цепочку: $\Sigma(1,0) = 1$, $\Sigma\left(1,\frac{1}{2}\right) = \frac{1}{2}$, $\Sigma\left(\frac{1}{2},1\right) = \frac{1}{2}$. Рассмотрим 3-ю ветвь:

$\Sigma\left(1,0,\frac{1}{2}\right) = 1 \Rightarrow \Sigma(1,1) = \frac{1}{2}$.

Такое возможно, поэтому строим ветви далее:

1. $\Sigma(1,0) = 0$ – невозможно, путь в дереве не существует.

2. $\Sigma(1,0) = \frac{1}{2}$ – невозможно, путь в дереве не существует.

3. $\Sigma(1,0) = 1$ – возможно. Восстановим цепочку действий и получим:

$$\Sigma(1,0)=1,\ \Sigma\left(1,\frac{1}{2}\right)=1,\ \Sigma(1,1)=\frac{1}{2}.$$

Таким образом, если мы рассмотрим таблицу истинности без учета свойства коммутативности (табл. 2), то она будет отражать значения функции в соответствии с заданными входами независимо от порядка аргументов.

Однако если мы учтем свойство коммутативности, то получим таблицу истинности, которая отражает значения функции, учитывая возможность изменения порядка аргументов (табл. 3). Это означает, что результаты функции будут одинаковыми независимо от того, какой из аргументов является первым, а какой вторым.

Учет коммутативности позволяет упростить анализ и понимание функции, поскольку мы можем рассматривать только уникальные комбинации аргументов, а не все возможные перестановки. Это также позволяет нам обобщить и применять функцию к различным наборам аргументов, сохраняя при этом одинаковые значения функции [9–11].

Таким образом, учет коммутативности в таблице истинности помогает нам упростить анализ функции и рассматривать ее более обобщенным образом, не зависящим от конкретных порядков аргументов.

Табл. 1. / Table 1.

	0	$\frac{1}{2}$	1
0	0	$\frac{1}{2}$	1
$\frac{1}{2}$	$\frac{1}{2}$		$\frac{1}{2}$
1	1	$\frac{1}{2}$	

Табл. 2. / Table 2.

	0	$\frac{1}{2}$	1
0	0	$\frac{1}{2}$	1
$\frac{1}{2}$	$\frac{1}{2}$		1
1	1		$\frac{1}{2}$

Табл. 3. / Table 3.

	0	$\frac{1}{2}$	1
0	0	$\frac{1}{2}$	1
$\frac{1}{2}$	$\frac{1}{2}$		1
1	1	1	$\frac{1}{2}$

Для данного примера мы имеем таблицу истинности, где занятые ячейки соответствуют необходимым условиям для выполнения заданного тождества, а пустые ячейки соответствуют несущественным условиям. Каждая свободная ячейка может принимать три возможных значения: 0, пусто и 1. Это означает, что каждая свободная ячейка порождает три варианта, которые могут быть заполнены.

Исходя из этого, мы можем определить точное количество функций в классе решений, то есть мощность этого класса.

В случае таблицы 1 количество функций в классе решений составляет 9, поскольку мы не предполагаем свойство коммутативности заранее и исследуем все возможные комбинации значений аргументов.

В случае таблицы 2 количество функций в классе решений также составляет 9. Однако если мы считаем свойство коммутативности наперед заданным, то количество функций в классе решений будет равно 3 (табл. 3).

Таким образом, утверждается, что коммутативность уменьшает мощность множества допустимых решений, поскольку количество свободных ячеек в таблице истинности уменьшается.

Можно утверждать, что существует алгоритм, который позволяет определить, можно ли выразить заданную функцию в виде формулы, используя операции обобщенного сложения. Алгоритм применяется построчно к заданной функции, представленной в виде таблицы истинности, а затем полученные результаты пересекаются. Если результирующее

множество пересечений не пусто, это означает, что может быть найдена группа решающих функций. Если же пересечение пусто, это означает, что заданную зависимость нельзя представить одной функцией, но можно подобрать минимальное количество функций, которые удовлетворяют решениям нейронной сети.

Заключение

Исследования в области логических функций и их применение в анализе данных предоставляют нам инструменты для выявления скрытых закономерностей и решения различных задач. Предложенные алгоритмы, основанные на построении таблиц истинности и учете коммутативности, позволяют получить логические функции, которые могут быть рассмотрены как адекватные локальные решатели поставленных задач.

Логические функции, полученные в результате этих алгоритмов, представляют собой формулы, состоящие из операций обобщенного сложения. Обобщенное сложение позволяет нам объединять и комбинировать значения переменных, учитывая их взаимодействие и влияние друг на друга. Такие функции обладают свойствами полноты и замкнутости относительно логических операций, что делает их мощным инструментом для анализа и обработки данных.

Применение этих логических функций в анализе данных позволяет нам выявить скрытые закономерности и зависимости между переменными. Они позволяют нам моделировать сложные взаимосвязи и отношения, которые сложно выразить другими методами. Путем применения этих функций к различным наборам данных мы можем обнаруживать не только явные закономерности, но и более скрытые и нетривиальные зависимости.

REFERENCES

1. *Журавлев Ю. И.* Об алгебраическом подходе к решению задач распознавания или классификации // Проблемы кибернетики. 1978. Т. 33. С. 5–68.

Zhuravlev Yu.I. On the algebraic approach to solving problems of recognition or classification. *Problemy kibernetiki* [Problems of cybernetics]. 1978. Vol. 33. Pp. 5–68. (In Russian)

2. *Шибзухов З. М.* Корректные алгоритмы агрегирования операций // Распознавание образов и анализ изображений. 2014. № 3–24. С. 377–382.

Shibzukhov Z.M. Correct algorithms for aggregating operations. *Raspoznavaniye obrazov i analiz izobrazheniy* [Pattern recognition and image analysis]. 2014. No. 3–24. Pp. 377–382. (In Russian)

3. Naimi A.I., Balzer L.B. Multilevel generalization: an introduction to super learning. *European Journal of Epidemiology*. 2018. Vol. 33. Pp. 459–464.

4. Wang H., Smith S. Big data analysis and perturbation using a data mining algorithm. *Journal of Soft Computing Paradigm*. 2021. No. 3–01. Pp. 19–28.

5. Joe MrK Vijesh, Jennifer S. Raj User Recommendation System Dependent on Location-Based Orientation Context. *Journal of Trends in Computer Science and Smart Technology*. 2021. No. 3–01. Pp. 14–23.

6. Grabisch M., Marichal J-L., Pap E. Aggregation functions. Cambridge University Press. 2009. Vol. 127. P. 16.

7. Calvo T., Belyakov G. Aggregating functions based on penalties. *Fuzzy sets and systems*. 2010. No. 10–161. Pp. 1420–1436.

8. Mesiar R., Komornikova M., Kolesarova A., Calvo T. Fuzzy aggregation functions: a revision. Sets and their extensions: representation, aggregation and models. Springer-Verlag. Berlin, 2008.

9. Yang F., Yang Zh., Cohen W.W. Differentiable learning of logical rules for reasoning in the knowledge base. *Advances in the field of neural information processing systems*. 2017. Pp. 2320–2329.

10. *Флах П.* Машинное обучение: наука и искусство построения алгоритмов, которые извлекают знания из данных. Москва: ДМК Пресс, 2015. 400 с.

Flach P. *Mashinnoye obucheniye: nauka i iskusstvo postroyeniya algoritmov, kotoryye izvlekayut znaniya iz dannykh* [Machine learning: the science and art of building algorithms that extract knowledge from data]. Moscow, DMK Press, 2015. 400 p. (In Russian)

11. Lyutikova L.A., Shmatova E.V. Algorithm for constructing logical operations to identify patterns in data. *E3S Web of Conferences*, Moscow, November 25–27, 2020. Vol. 224. P. 01009.

Информация об авторе

Лютикова Лариса Адольфовна, канд. физ.-мат. наук, зав. отделом нейроинформатики и машинного обучения, Институт прикладной математики и автоматизации – филиал Кабардино-Балкарского научного центра Российской академии наук;

360000, Россия, г. Нальчик, ул. Шортанова, 89 А;

lylarisa@yandex**,** ORCID: https://orcid.org/0000-0003-4941-7854

Information about the author

Lyutikova Larisa Adolfovna, Candidate of Physical and Mathematics Sciences, Head of the Department of Neuroinformatics and Machine Learning, Institute of Applied Mathematics and Automation – branch of Kabardino-Balkarian Scientific Center of the Russian Academy of Sciences;

360000, Russia, Nalchik, 89 A Shortanov street;

lylarisa@yandex**,** ORCID: https://orcid.org/0000-0003-4941-7854