CC BY

http://dx.doi.org/10.35596/1729-7648-2024-22-6-62-69

Оригинальная статья
Original paper

УДК 681.142.2

ИСПОЛЬЗОВАНИЕ НЕЙРОСЕТЕЙ ДЛЯ РЕШЕНИЯ ЗАДАЧ ПРИКЛАДНОЙ ЛОГИКИ

Ю. О. ГЕРМАН, О. В. ГЕРМАН

Белорусский государственный университет информатики и радиоэлектроники (г. Минск, Республика Беларусь)

Поступила в редакцию 21.07.2024

© Белорусский государственный университет информатики и радиоэлектроники, 2024
Belarusian State University of Informatics and Radioelectronics, 2024

Аннотация. Рассматривается использование эвристического нейросетевого решателя для решения NP-трудных задач (определения (не)противоречивости системы логических уравнений). Эта проблема актуальна и важна, например, при выполнении экспресс-анализа непротиворечивости базы знаний экспертной системы, принятии решений на основе нечетких логических моделей, распознавании многомерных объектов и др. Обученная нейросеть выполняет роль высокоэффективного эвристического решателя, причем количество уравнений и переменных, используемых в логической модели, мало влияет на скорость принятия решений нейросетью, одновременно вероятность точного решения для параметрически определенного класса задач близка к единице. Под параметрически определенным классом задач понимается множество задач, описываемых многомерными векторами параметров, удовлетворяющих некоторому общему закону распределения вероятностей. Одно такое семейство параметров, предложенное и использованное для обучения нейросети, приведено в статье. Показано, как генерировать противоречивые и непротиворечивые экземпляры индивидуальных систем логических уравнений. Проведена серия более чем из 200 экспериментов по апробации модели, получены границы доверительного интервала вероятности правильного решения, что позволяет судить об эффективности модели. Показано, как применить нейросеть для проверки (не)противоречивости логической модели знаний. Построенная модель может быть эффективно дополнена новыми векторами параметров и применена в различных областях прикладных исследований.

Ключевые слова: нейросеть, система логических уравнений, эвристический поиск решения, проверка непротиворечивости.

Конфликт интересов. Авторы заявляют об отсутствии конфликта интересов.

Для цитирования. Герман, Ю. О. Использование нейросетей для решения задач прикладной логики / Ю. О. Герман, О. В. Герман // Доклады БГУИР. 2024. Т. 22, № 6. С. 62–69. http://dx.doi.org/10.35596/1729-7648-2024-22-6-62-69.

USAGE OF NEURAL NETWORKS FOR SOLVING APPLIED LOGIC PROBLEMS

JULIA O. GERMAN, OLEG V. GERMAN

Belarusian State University of Informatics and Radioelectronics (Minsk, Republic of Belarus)

Submitted 21.07.2024

Abstract. The article deals with heuristic neural network-based solver for NP-hard problems (determining the (in)consistency of a system of logical equations). This problem is relevant and important, for example, when performing express analysis of the consistency of the knowledge base of an expert system, decision-making based on fuzzy logic models, recognition of multidimensional objects, etc. The trained neural network plays the role of a highly efficient heuristic solver, and the number of equations and variables used in the logical model has little

effect on the speed of decision-making by the neural network, while the probability of an exact solution for a parametrically defined class of problems is close to one. A parametrically defined class of problems is understood as a set of problems described by multidimensional vectors of parameters that satisfy some general law of probability distribution. One such family of parameters, proposed and used for training a neural network, is given in the article. It is shown how to generate inconsistent and consistent instances of individual systems of logical equations. A series of more than 200 experiments to test the model was carried out, the limits of the confidence interval of the probability of a correct decision were obtained, which allows us to evaluate the effectiveness of the model. It is shown how to implement a neural network to check the (in)consistency of a logical knowledge model. The constructed model can be effectively supplemented with new parameter vectors and applied in various fields of applied research.

Keywords: neural network, system of logical equations, heuristic search for a solution, consistency check.

Conflict of interests: The authors declare no conflict of interests.

For citation: German Ju. O., German O. V. (2024) Usage of Neural Networks for Solving Applied Logic Problems. *Doklady BGUIR*. 22 (6), 62–69. http://dx.doi.org/10.35596/1729-7648-2024-22-6-62-69 (in Russian).

Введение

Известно [1, 2], что задачи, связанные с решением систем логических уравнений (СЛУ), в частности задачи выполнимости булевых формул (SAT – satisfyability), в общем случае трудноразрешаемы. Использование нейросетей для решения СЛУ активно исследовалось в [3–7]. В [2–4] отмечены основные недостатки нейросетей для точного решения SAT: необходимость огромного обучающего множества с миллионами экземпляров, критическая зависимость от обучающей выборки; время поиска решения может быть даже больше, чем у классических несетевых решателей, и др. Обучение нейросетевых решателей выполняется как на основе уже собранных известных решений (benchmarks), так и с использованием точных решателей типа MiniSat и др. В первом случае разработчик «привязан» к параметрам (способам генерации) SAT в используемом конкретном наборе, причем доступные наборы определяют в общем разные параметризованные классы SAT. Второй вариант требует значительных временных затрат на обучение.

Нейросетевые решатели (типа NeuroSAT, WALKSAT-Net и др.) для приближенного поиска решения связаны со следующими проблемами: возможность зацикливания, остановка на локальном минимуме с потерей ответа, чувствительность к параметрам задачи. Такие решатели обладают серьезным недостатком – не всегда имеется возможность оценить истинность найденного решения (в случае невыполнимости). В [7] отмечается, что ни в классе точных, ни в классе приближенных решателей для SAT нет наилучшего. Согласно теореме Boppana и Hastad, нет эффективного статистически оптимального алгоритма для SAT. Поэтому требуется, например, выбор самого решателя под индивидуальные SAT либо использование группы приближенных решателей с выбором решения из найденного множества и т. п. Интерес представляет построение приближенного нейросетевого решателя, обученного для определенного параметризованного класса SAT как для самостоятельного использования, так и в коллективном контексте. Таких параметризованных классов необозримо много, поэтому выбор параметров определяет параметризованный класс, а не наоборот. Например, в [5] отмечено, что число параметров для обучения сети достигало 48, но не указано, что это предел.

Учитывая экспоненциальную вычислительную сложность SAT (на данный момент), параметризованный нейросетевой классификатор с фиксированным числом параметров не способен обеспечить (даже статистически) оптимальное решение для теоретически неограниченного числа задач SAT. Пусть построен точный классификатор на основе обученной нейросети с фиксированным числом параметров F. Тогда он будет решать задачу SAT за полиномиальное от размера описания SAT время, что невозможно, если SAT экспоненциально сложна. Размер входного описания вектора параметров зависит только от формата числового представления параметров и растет как логарифмическая функция $O(F \cdot \mathrm{Log}_{10}\ \mathrm{MaxSize})$, где MaxSize – максимальное по абсолютной величине возможное значение параметра (на 64-разрядной ЭВМ это 2^{64}). В данной статье число параметров равно восьми при статистически приемлемых результатах распознавания параметризованного класса, что в целом отрицает идею существенного наращивания их числа. Рассмотрим некоторый параметризованный класс SAT и построим для него приближенный нейросетевой решатель. Предлагается (это – первая цель исследований авторов) новая техника

генерации выполнимых и невыполнимых SAT, которой можно «охватить» весьма широкий диапазон параметризованных классов SAT. При этом время решения SAT даже сравнительно больших размеров (несколько сот переменных и несколько тысяч уравнений) оказывалось в пределах одной секунды на ноутбуке с частотой 2,5 ГГц.

Оценка эффективности приближенных SAT-решателей строится не так, как для точных, для которых критическими являются факторы потребляемого времени и памяти (в SATContest время решения ограничено, статистический анализ решателя не производится, полученные решения сравниваются с решениями, доставляемыми точным решателем). Общепризнанные подходы типа SATContest не отменяют стандартный статистический анализ, основанный на сборе и обработке экспериментальных данных для исследуемого параметризованного класса SAT (вторая цель исследований авторов).

Предварительные определения и данные

Дизъюнкт – это логическая формула, связывающая булевские переменные или их отрицания посредством операции логического ИЛИ (дизъюнкции) [1, 8]. Задача SAT заключается в установлении того, имеет решение данная система дизъюнктов или нет (т. е. является противоречивой).

Для обучения нейросети решению SAT нужно сгенерировать обучающее множество индивидуальных задач SAT, для которых заранее известно, имеют они решение или нет. Задача SAT должна быть представлена в обучающей таблице вектором параметров. Использовались следующие параметры (атрибуты): **A1** – отношение числа дизъюнктов к числу переменных; **A2** – среднее число переменных в дизъюнкте; **A3** – отношение числа вхождений переменных с отрицанием к общему числу вхождений переменных в дизъюнктах; **A4** – отношение числа дизъюнктов, содержащих только негативные или только позитивные переменные, к общему числу дизъюнктов; **A5** – отношение числа двухлитерных дизъюнктов к общему числу дизъюнктов; **A6** – отношение сумм длин всех дизъюнктов к произведению числа дизъюнктов на число переменных (длина дизъюнкта равна числу указанных в нем переменных с отрицанием или без; тавтологические дизъюнкты не рассматривались); **A7** – отношение числа вхождений переменных без отрицания к числу вхождений переменных с отрицанием; **A8** – отношение атрибута **A1** к **A7** (**A1**/**A7**). Так, для СЛУ

$$\begin{aligned} \mathrm{D1} &= x_1 \vee {\sim}x_2 \vee {\sim}x_3; \\ \mathrm{D2} &= {\sim}x_1 \vee {\sim}x_4; \\ \mathrm{D3} &= x_2 \vee x_4; \\ \mathrm{D4} &= x_3 \vee x_4; \\ \mathrm{D5} &= x_1 \vee x_2; \\ \mathrm{D6} &= {\sim}x_3 \vee {\sim}x_4 \end{aligned} \qquad (1)$$

имеем вектор параметров

Параметр	**A1**	**A2**	**A3**	**A4**	**A5**	**A6**	**A7**	**A8**
Значение	1,50	2,16	0,46	0,83	0,83	0,54	1,17	1,28

СЛУ (1) непротиворечива и имеет одно из решений: x_1, x_2, x_3, ${\sim}x_4$.

Выбор параметров – это задача исследователя. Для приближенного нейросетевого решателя важно статистически удовлетворительное итоговое решение, при этом и число параметров, и их информативность смещаются на второй план. Иногда задачу отбора параметров возлагают на саму генеративную нейросеть. Для того чтобы автоматизировать процесс накопления данных, задачу SAT сведем к задаче о минимальном покрытии 0,1-матрицы [8]. Соответствующая матрица показана на рис. 1.

Каждый дизъюнкт представлен отдельным столбцом. В ячейке столбца пишем «1», если переменная в этой строке входит в данный дизъюнкт (с учетом знака). Присоединяем к матрице дополнительные столбцы-тавтологии вида $x_i \vee {\sim}x_i$. Покрытием является множество строк, таких, что в каждом столбце матрицы покрытия хотя бы одна из этих строк содержит «1». Покрытие минимально, если оно минимально возможного размера. Имеет место следующий результат: если система дизъюнктов выполнима, то минимальное покрытие соответствующей 0,1-матрицы содержит ровно n строк (n – число переменных) [8]. В рассматриваемом примере минимальное покрытие содержит четыре строки.

	D1	D2	D3	D4	D5	D6	DD1	DD2	DD3	DD4
x_1	1				1		1			
x_2			1		1			1		
x_3				1					1	
x_4			1	1						1
$\sim x_1$		1					1			
$\sim x_2$	1							1		
$\sim x_3$	1					1			1	
$\sim x_4$		1				1				1

Рис. 1. Матрица покрытия для системы логических уравнений (1)
Fig. 1. Covering matrix for system of logical equations (1)

Генерировались матрицы покрытия на основе Python-скрипта случайным образом на основе биномиального закона распределения вероятностей *pr* появления «1» в ячейках, чтобы определять числовые значения показателей **A1**–**A8**. Вероятности *pr* случайно выбирались из значений, равномерно распределенных на фиксированном интервале (см. ниже). При этом требовалось обеспечить наличие хотя бы одной «1» в столбце, что практически имеет место при $m \times pr \geq 1$ (m – число строк матрицы, $m = 2n$; n – число переменных).

Если решение есть, проверяем, сгенерирована ли хотя бы одна «1» в ячейках, соответствующих переменным из наперед заданного решения. Если ни одна из сгенерированных единиц не соответствует значению переменной в заданном решении, то «перебрасываем» любую случайно сгенерированную «1» в произвольно выбираемую ячейку, соответствующую элементу решения. Если решения для исходной системы дизъюнктов нет, то следует искусственно присоединить к матрице 0,1-подматрицу *SM* с $(n + 1)$-столбцами; *SM* заполняется так, чтобы в каждой строке было не более одной единицы, но не было нулевых столбцов. Очевидно, возможности заполнения такой матрицы ограничены, поскольку имеются столбцы с одной единственной единицей. Для устранения этого недостатка можно ввести дополнительную переменную (-ные), например z, и расширить матрицу (рис. 2), определив условие $\sim z = 1$. Для этого вводятся два фиктивных дизъюнкта DD7 и DD8:

$$x_1 \vee z \text{ и } \sim x_1 \vee z. \tag{2}$$

	DD1	DD2	DD3	DD4	DD5	DD6	DD7	DD8
x_1	1						1	
x_2	1							
x_3		1						
x_4			1					
x_5			1					
z							1	1
$\sim x_1$				1				1
$\sim x_2$				1				
$\sim x_3$					1			
$\sim x_4$					1			
$\sim x_5$						1		
$\sim z$		1				1		

Рис. 2. Минимальное покрытие с не менее чем шестью строками
Fig. 2. Minimum coverage with at least six lines

Минимальное покрытие представленной подматрицы *SM* не может содержать пять и менее строк. То есть исходная система дизъюнктов на переменных $x_1 \ldots \sim x_5$ не имеет допустимого решения даже с учетом фиктивной переменной z в (2). Описанная техника позволяет эффективно генерировать выполнимые и невыполнимые системы дизъюнктов случайным образом, что особенно важно для больших по размеру матриц СЛУ. Число фиктивных переменных можно увеличить, действуя по аналогии.

Результаты экспериментов и статистический анализ

Для построения нейросети сгенерировано случайным образом 47 0,1-матриц покрытия с заранее известными решениями Target = 0 (SAT невыполнима) и Target = 1 (SAT выполнима). Использовали csv-файл с обучающим набором и Python-код, приведенный ниже:

```
import pandas as pd
from sklearn.neural_network import MLPClassifier
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
import pickle
# Read the CSV file into a DataFrame
data = pd.read_csv('c:/neuralexp5.csv')
# Separate the features (X) and the target variable (y)
X = data.drop('Target', axis=1)
y = data['Target']
# Create an MLP classifier
mlp = MLPClassifier(hidden_layer_sizes=(50, 50), max_iter=1000, random_state=42)
# Train the classifier
mlp.fit(X, y)
X_test=[[1.5,2.16,0.61,0.83,0.83,0.54,1.17,1.28]]
y_pred = mlp.predict(X_test)
# Evaluate the classifier on the test set
print("y_pred:", y_pred)
accuracy = mlp.score(X, y)
print("accuracy=",accuracy)
# Save the model to a file
filename = 'c:/neuralnet_model.pkl'
with open(filename, 'wb') as file:
   pickle.dump(mlp, file)
   print('neuralnet model saved')
```

Модель созданной нейросети сохранена в файле `c:/neuralnet_model.pkl`. Было проведено 200 экспериментов. Размеры генерируемых матриц в каждом эксперименте выбирались случайно и варьировались от 10×10 до 600×2000 (строк × столбцов) со случайно определяемой плотностью единичных элементов в пределах [0,01–0,20] (табл. 1). В 200 экспериментах решение было потеряно в восьми случаях.

Таблица 1. Экспериментальный состав
Table 1. Experiment composition

Число строк	Число столбцов	Плотность единичных элементов	Число матриц
[10, 20]	[10, 30, 80]	[0,10–0,20]	50
[40, 60]	[80, 100, 150]	[0,015–0,060]	60
[100, 400, 600]	[1000, 2000]	[0,01–0,03]	90

Для каждого эксперимента значение SAT устанавливалось точно, как объяснено выше. Таким образом, и ложно отрицательные, и ложно положительные ответы трактовались как потеря решения нейросетью. Общее число экспериментов N определялось на основе формулы, используемой как для нормального, так и для биномиального (Бернулли) распределения:

$$N = (Z_{\alpha/2} + Z_{\beta})^2(k + 1)/f^2, \quad (3)$$

где $Z_{\alpha/2}$ – z-оценка для соответствующего уровня значимости (например, $Z_{\alpha/2} = 1{,}96$ для $\alpha = 0{,}05$); Z_{β} – z-оценка для желаемой статистической мощности (например, $Z_{\beta} = 0{,}84$ для 80%-ной мощности); k – число независимых критериев (предикторов), $k = 7$; f – величина эффекта, $f = R/\mathrm{sqrt}(1 - R^2)$; R – значение множественного коэффициента корреляции между предикторами и выходной переменной.

Так, для $R \in [0{,}3–0{,}4]$, что соответствует средней величине эффекта, получим $N \in [157–209]$. Использовали скрипт Python для оценки доверительного интервала (confidence interval) вероятности биномиального распределения неверного ответа построенной нейросети на основе результатов экспериментов и метод

```
# Calculate confidence interval
confidence_interval = sms.DescrStatsW(data).tconfint_mean()
```

класса `statsmodels.stats.api as sms` с выводом результата **`Confidence Interval: (0.012607200785124728, 0.067392799214875270)`**.

Доверительный интервал для вероятности ошибки заключения нейросети по данным экспериментов вычисляется по формуле

$$\text{mean} - t_{\text{alpha/2}} \cdot \text{std\_err}, \text{mean} + t_{\text{alpha/2}} \cdot \text{std\_err}, \quad (4)$$

где mean – среднее значение; std_err – стандартное отклонение; $t_{\text{alpha/2}}$ – критическое значение t-распределения для заданного уровня значимости (обычно 95 %, что соответствует ошибке не более 5 %).

Формула (4) может быть использована для биномиального распределения, которое приближенно аппроксимируется нормальным при числе испытаний не менее 30.

Массив data фиксирует случаи неверного заключения нейросети там, где стоит «1». Таким образом, если индивидуальная задача SAT попадает в параметризованный класс, определяемый обучающей таблицей, то вероятность правильного итогового решения достаточно высока. Гипотеза о принадлежности индивидуального вектора (**A1**, …, **A8**) данному классу векторов может быть проверена с помощью статистических критериев. Формула для расчета t-критерия Стьюдента в случае проверки гипотезы о принадлежности многомерного объекта выборке на основе евклидова расстояния выглядит следующим образом:

$$t = (d - \mu)/(s/n^{0,5}), \quad (5)$$

где t – t-статистика; d – евклидово расстояние между многомерным объектом и центром выборки (центроидом); μ – среднее значение расстояния до центроида в выборке; s – стандартное отклонение расстояний в выборке; n – размер выборки (количество объектов в выборке).

Используя таблицу критических значений статистики t-критерия Стьюдента, находим соответствующее значение t для заданного уровня значимости ($\alpha = 0{,}05$) и числа степеней свободы ($df = n - 1$). Для принятия объекта в выборку вычисленное по (5) значение не должно превосходить $t_{\text{кр}}$.

Пример практической задачи

Механизм принятия решений на основе нейросети можно реализовать в продукционной экспертной системе с базой знаний вида «если $f_1 \& f_2 \& \ldots f_n$, то g», где посылки f_i – суть значения (условия, ограничения) некоторых параметров, а g – заключение. Необходимо привести параметры и заключение к булевским переменным (в общем случае). Область изменения каждого параметра разбивается на интервалы I_{j1}, I_{j2}, …, I_{jk}; каждый интервал I_{jt} представляется одной и только одной булевской переменной x_{jt}. То же выполняется и для заключений при введении переменных g_1, g_2, …, g_r согласно классу (области) соответствующего заключения. В систему (базу) знаний включаются условия: $\Sigma_t\, x_{jt} = 1$, $\Sigma_p\, g_p \le 1$ (нельзя получить два разных заключения, но можно ни одного). Первое заменяется системой дизъюнктов: $x_{j1} \vee x_{j2} \vee \ldots \vee x_{jk}$; $\sim x_{j1} \vee \sim x_{j2}$; $\sim x_{j1} \vee \sim x_{j3}$; … ; $\sim x_{jk-1} \vee \sim x_{jk}$ ($j = 1, J$), второе – на $\sim g_1 \vee \sim g_2$; $\sim g_1 \vee \sim g_3$; … ; $\sim g_{r-1} \vee \sim g_r$.

Одна из важных задач – выполнить экспресс-анализ базы знаний на непротиворечивость. Здесь возможны разные постановочные варианты. В простейшем случае противоречивость иллюстрируется наличием, например, продукционных правил вида $x_p = a \& x_q = b \to y$ с входным набором $\text{IN}_\alpha = \{x_p = a, x_q = b\}$ и $x_p = a \to \sim y$ с входным набором $\text{IN}_\beta = \{x_p = a\}$, $\text{IN}_\beta \subseteq \text{IN}_\alpha$ и взаимоисключающими заключениями y и $\sim y$. Для проверки нужно (временно) исключить условие $\Sigma_p\, g_p \le 1$ и попеременно добавлять в систему знаний конъюнкции $g_i \& g_k$, проверяя выполнимость системы дизъюнктов. Если для какой-то пары $g_i \& g_k$ система выполнима, то легко проверить, имеет

место $IN_i \subseteq IN_k$ или нет. Более сложный случай предполагает, что посылки правил могут состоять из произвольных логических формул. Здесь проверка непротиворечивости представляет самостоятельную задачу.

Заключение

1. Нейронную сеть как эвристический решатель можно использовать для задач SAT весьма большого размера. При этом предложенный вектор параметров схватывает «усредненные» характеристики матрицы покрытия, т. е. не «чувствителен» к размерам задачи, конкретному виду SAT, размерам противоречивой части в случае невыполнимости и др. Речь идет о среднестатистической SAT в рамках параметризованного класса задач.

2. Предложен эффективный способ генерации выполнимых и невыполнимых SAT, который можно использовать для обучения нейросети на системах очень больших размеров, генерируемых для интересующих исследователя вероятностных распределений единичных элементов в матрицах покрытия.

3. Техника анализа статистического решателя реализуется в рамках стандартной статистической парадигмы, а не путем сравнения с точным решателем. Приближенные нейросетевые решатели можно интегрировать в точные, например, используя поиск на основе техники разрешения конфликтов – CDCL (conflict-driven clause learning) (одно из первых алгоритмических решений было предложено в [9]). Альтернативный вариант – коллективное применение приближенных решателей.

Список литературы

1. Garey, M. R. Computers and Intractability: A Guide to the Theory of NP-Completeness / M. R. Garey, D. S. Johnson. New York: W. H. Freeman and Co., 1979.
2. Handbook of Satisiability / A. Biere [et al.] // IOS Press. Amsterdam, 2021.
3. Selsam, D. Guiding High Performance SAT Solvers with Unsat-Core Predictions / D. Selsam, N. Bjorner // arXiv. 2019. P. 1–19.
4. Machine Learning for SAT: Restricted Heuristics and New Graph Representations / M. Shirokikh [et al.] // arXiv. 2023. P. 1–17.
5. Machine Learning Methods in Solving the Boolean Satisfiability Problem / W. Guo [et al.] // arXiv. 2022. P. 1–8.
6. Menai, M. B. Solving the Maximum Satisfiability Problem Using an Evolutionary Local Search Algorithm / M. B. Menai, M. Batouche // The International Arab Journal of Information Technology. 2005. Vol. 2, No 2. P. 154–161.
7. An Overview of Modern Learning Techniques in Constraint Solving / A. Popescu [et al.] // Journal of Intelligent Information Systems. 2022. Vol. 58. P. 91–118.
8. Герман, О. В. Введение в теорию экспертных систем и обработку знаний / О. В. Герман. Минск: ДизайнПро, 1995.
9. Птичкин, В. А. Об эвристическом усилении метода резолюций / В. А. Птичкин, О. В. Герман, В. Г. Найденко // Автоматика и вычислительная техника. 1993. Вып. 21. С. 88–93.

References

1. Garey M., Johnson D. S. (1979) *Computers and Intractability: A Guide to the Theory of NP-Completeness.* New York, W. H. Freeman and Co. Publ.
2. Biere A., Heule M., van Maaren H., Walsh T. (2021) Handbook of Satisiability. *IOS Press*. Amsterdam.
3. Selsam D., Bjorner N. (2019) Guiding High Performance SAT Solvers with Unsat-Core Predictions. *arXiv*. 1–19.
4. Shirokikh M., Shenbin M., Alekseev A., Nikolenko S. (2023) Machine Learning for SAT: Restricted Heuristics and New Graph Representations. *arXiv*. 1–17.
5. Guo W., Zhen H.-L., Li X., Yuan M., Jin Y. (2022) Machine Learning Methods in Solving the Boolean Satisfiability Problem. *arXiv*. 1–8.
6. Menai M. B., Batouche M. (2005) Solving the Maximum Satisfiability Problem Using an Evolutionary Local Search Algorithm. *The International Arab Journal of Information Technology*. 2 (2), 154–161.
7. Popescu A., Seda P.-E., Felfernig A., Uta M., Alas M., Le V-M. (2022) An Overview of Modern Learning Techniques in Constraint Solving. *Journal of Intelligent Information Systems*. 58, 91–118.

8. German O. V. (1995) *Introduction to the Theory of Expert Systems and Knowledge Processing*. Minsk, DesignPro Publ. (in Russian).
9. Ptichkin V. A., German O. V., Naydenko V. G. (1993) On Heuristic Strengthening the Resolution Method. *Automatics and Computing Technique*. (21), 88–93 (in Russian).

Вклад авторов

Герман Ю. О. выполнила тестирование модели нейросети и сбор экспериментальных данных. Написала введение, заключение, указала способ применения *t*-критерия Стьюдента для параметризованного класса задач выполнимости булевых формул.

Герман О. В. определил общую концепцию работы, указал способ генерации индивидуальных задач выполнимости булевых формул с наперед известным ответом об их выполнимости/невыполнимости.

Author's contribution

German Yu. O. performed testing of the neural network model and collection of experimental data. Wrote the introduction, conclusion, indicated the method of applying the Student's *t*-test for a parameterized class of satisfiability problems for Boolean formulas.

German O. V. defined the general concept of the work, indicated a method for generating individual SAT problems with apriori known answer about their satisfiability/unsatisfiability.

Сведения об авторах

Герман Ю. О., канд. техн. наук, доц. каф. информационных технологий автоматизированных систем, Белорусский государственный университет информатики и радиоэлектроники

Герман О. В., канд. техн. наук, доц. каф. информационных технологий автоматизированных систем, Белорусский государственный университет информатики и радиоэлектроники

Адрес для корреспонденции

220013, Республика Беларусь,
г. Минск, ул. П. Бровки, 6
Белорусский государственный университет
информатики и радиоэлектроники
Тел.: +375 29 612-42-32
E-mail: ovgerman@tut.by
Герман Олег Витольдович

Information about the authors

German Ju. O., Cand. of Sci., Associate Professor at the Information Technologies in Automatized System Department, Belarusian State University of Informatics and Radioelectronics

German O. V., Cand. of Sci., Associate Professor at the Information Technologies in Automatized System Department, Belarusian State University of Informatics and Radioelectronics

Address for correspondence

220013, Republic of Belarus,
Minsk, P. Brovki St., 6
Belarusian State University
of Informatics and Radioelectronics
Tel.: +375 29 612-42-32
E-mail: ovgerman@tut.by
German Oleg Vitoldovich