УДК 004.8:004.056
DOI: 10.20535/SRIT.2308-8893.2023.2.06

НЕЙРОННІ МЕРЕЖІ: ДОСЛІДЖЕННЯ ПРАВИЛ ПРИЙНЯТТЯ НИМИ РІШЕНЬ

А.І. ПЕТРЕНКО, І.А. ВОХРАНОВ

Анотація. Питання отримання більшої зрозумілості поведінки нейронних мереж є досить актуальним, особливо у галузях із високим рівнем ризиків. Для вирішення цієї задачі досліджено можливості нового алгоритму декомпозиції DeepRED, здатного витягувати правила прийняття рішень глибинними нейронними мережами з декількома прихованими шарами DNN (Deep Neural Networks). Дослідження алгоритму DeepRED проводилося на прикладі вилучення правил експериментальної нейронної мережі за виконання класифікації зображень бази даних MNIST рукописних цифр, що дозволило виявити ряд обмежень алгоритму DeepRED.

Ключові слова: вилучення правил, нейронні мережі, DeepRED, машинне навчання, дерева рішень, графи рішень.

ОПИС ЗАДАЧІ І СТАНУ ЇЇ РІШЕННЯ

Натепер, нейронні мережі (NN, Neural Networks) мають дуже широкий спектр застосувань. Вони здатні вирішувати такі задачі, як задачі класифікації, з дуже високою ефективністю. Проте, не зважаючи на всі їх переваги, все ще існує проблема з можливістю використання нейронних мереж у галузях із високим рівнем ризиків. У таких сферах, як медицина, фінанси та енергетика, розуміння та передбачуваність поведінки систем є дуже критичними. Будь-який неочікуваний сценарій може привести до ризиків для життя чи здоров'я людей, або до ризику великих фінансових втрат [1, 2]. Саме тому, питання отримання більшої зрозумілості поведінки нейронних мереж є досить актуальним.

Нейронні мережі формують свою логіку завдяки використання шарів нейронів. Структуру всіх нейронних мереж, в загальному, можна звести до одного спільного вигляду: вхідний шар, вихідний шар та набір прихованих навчальних шарів (рис. 1).

Щоб подолати вказану незручність нейронних мереж, протягом останніх трьох десятиліть, детально розробляються різні способи пояснення "логіки" прийняття рішень нейронними мережами. Першим запровадженим та найбільш перспективним на сьогоднішній день підходом у цій галузі є ***вилучення правил*** *(RE, Rule Extraction*) із штучних нейронних мереж. *Вилучення правил — це підхід, що зосереджується на розкритті прихованих в мережі правил, з метою допомогти пояснити, як саме нейронна мережа*

приходить до остаточного рішення. Такі вилучені правила можуть використовуватись для зменшення небезпеки, відстеження стабільності та відмовостійкості системи, перевірки та підтвердження можливості використання нейронної мережі для конкретної задачі, тощо. Більшість дослідників зосереджуються на тому, що вилученні правила мають бути максимально зрозумілими, але в той же час повинні якомога точніше імітувати поведінку нейронної мережі.

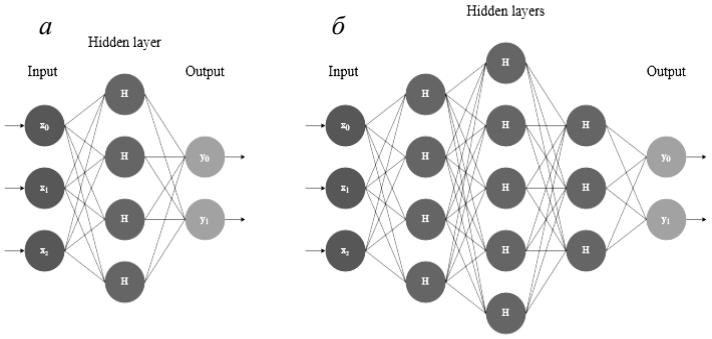


Рис. 1. Структура нейронних мереж: *а* — звичайної нейронної мережі, *б* — глибинної нейронної мережі

Існуючі методи вилучення правил поділяються на три групи: ***декомпозиційні, педагогічні та еклектичні*** [2]. Декомпозиційні методи повністю спираються на архітектуру мережі і використовують активації та ваги всіх нейронів, включаючи приховані шари. Зазвичай ці методи аналізують кожен нейрон, після чого, отримані описи поведінки цих нейронів певним чином об'єднуються та формують правила, що імітують поведінку всієї моделі [7].

Основними представниками декомпозиційних підходів є: метод КТ [10], метод на основі правил нечіткої логіки [6], поліноміальний алгоритм Цукімото [4], CRED [12], DeepRED [7].

Педагогічні методи, на відміну від декомпозиційних, не враховують внутрішню структуру нейронної мережі, а розглядають NN як єдину сутність (чорну скриньку). Їх принцип полягає в тому, щоб витягувати правила шляхом прямого відображення вхідних даних у вихідні. Іншим чином, це можна розглядати, як добре відому задачу апроксимації, де нейронна мережа виступає в ролі цільової функції, що приймає визначений набір вхідних параметрів та повертає певний результат класифікації. Маючи цю функцію, алгоритми намагаються знайти узгодженість між вхідними варіаціями та результатами. Найбільш популярними представниками педагогічних методів є: VIA [17] [18], TREPAN [13], BIO-RE [5], ANN-DT [3], STARE [20], KDRuleEx [8][9], RxREN [11].

Еклектичні методи представляють собою поєднання інших підходів і розглядають лише частину NN як чорний ящик. Оскільки визначення еклектичних підходів досить розпливчасте, іноді дослідники мають різні погляди на те, які методи правильно відносити до еклектичних, а які ні. Двома, досить відомими методами, які можна віднести до еклектичних, є MofN [2] та FERNN [15].

Більшість алгоритмів вилучення правил запропоновані лише для невеликих нейронних мережах з *одним прихованим шаром*. Проте, нещодавно було запропоновано новий алгоритм — ***DeepRED***, який здатен працювати з *глибинними нейронними мережами*. Цей алгоритм декомпозиції витягує проміжні правила для кожного шару NN, на основі чого, формує представлення того, як конкретна нейронна мережа приймає рішення [7]. Він вважа-

ється найбільш досконалим з точки зору зрозумілості вилучаємих правил і їх максимального наближення до поведінки НМ. Слід визнати, що існують модифікації DeepRED, які поліпшують його властивості [14].

Стаття організована таким чином: розділ I містить опис задачі і огляд стану її рішення; розділ II — огляд алгоритму; розділ III — опис покращення алгоритму; розділ IУ — експериментальні результати дослідження алгоритмів, а розділ V містить висновки щодо задачі, яка розглядається.

ОПИС РОБОТИ АЛГОРИТМУ DEEPRED

Основною сутністю будь-якої нейронної мережі є модель нейрона (рис. 2). Ідея нейронної моделі полягає в тому, що вхід x разом із зміщенням b зважуються вагами w, а потім підсумовуються разом. Зміщення b (*bias*) є скалярним значенням, тоді як вхідні дані x та ваги w мають векторне значення, тобто $x \in \mathbb{R}^n$ та $w \in \mathbb{R}^n$, де $n \in \mathbb{N}$, що відповідає розмірності вхідних даних. Їх сума $z = x^T x + b$ виступає в якості аргумента ***функції активації*** φ, в результаті чого, формується вихід нейронної моделі:

$$y = \varphi(z) = \varphi(w^T x + b) \tag{1}$$

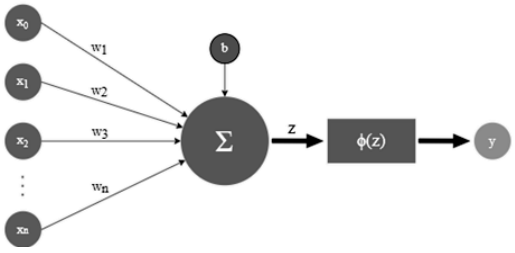
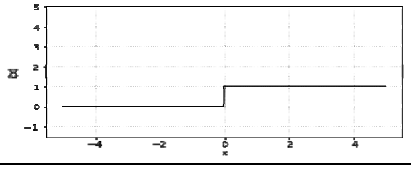
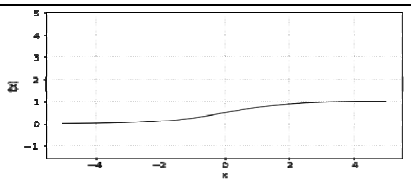


Рис. 2. Модель штучного нейрона

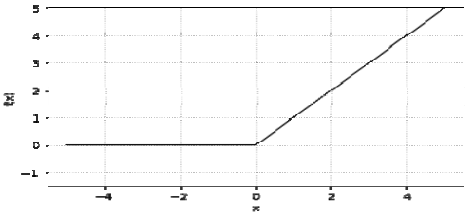
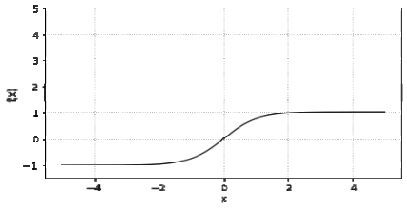
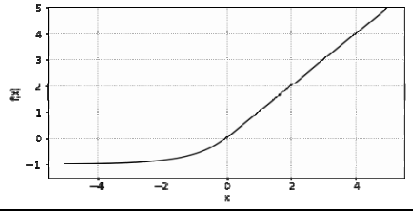
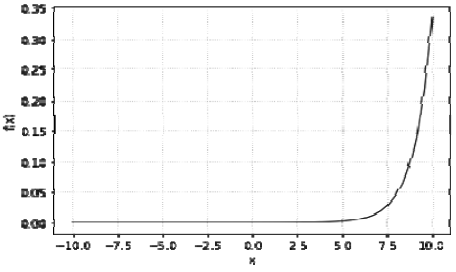
Деякі поширені функції активацій та їх графічні зображення приведені в табл. 1.

Вхідними даними алгоритму є навчальна вибірка даних і задані активації всіх шарів мережі. Найпопулярнішими функціями активації для глибинних нейронних мереж є функція активації ReLU і функція Softmax.

Т а б л и ц я 1 . Деякі функції активацій та їх графічні зображення

Назва	Функція (φ)	Графічне зображення
Step (східчаста)	$\varphi(x) = \begin{cases} 0 \text{ if } x < 0 \\ 1 \text{ if } x \geq 0 \end{cases}$	
Sigmoid (Logistic)	$\varphi(x) = \dfrac{1}{1+e^{-x}}$	

Продовження табл.1

Назва	Функція (φ)	Графічне зображення
Rectified Linear Unit (ReLU)	$\varphi(x) = \begin{cases} 0 & \text{if } x \le 0 \\ x & \text{if } x > 0 \end{cases}$	
Hyperbolic Tangent (Tanh)	$\varphi(x) = \dfrac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$	
Exponential Linear Unit (ELU)	$\varphi(x) = \begin{cases} \alpha(e^{x} - 1) & \text{if } x \le 0 \\ x & \text{if } x > 0 \end{cases}$	
Softmax	$\varphi(x) = \dfrac{e^{x_i}}{\sum_{j=1}^{J} e^{x_j}}$ for $i = 1, \ldots J$	

Алгоритм DeepRED було обрано в роботі, оскільки це метод декомпозиції, пристосований до аналізу глибинних нейронних мереж DNN (Deep Neural Networks), точніше до MLP (Multi-Layer Perceptron) з будь-якою глибиною, але без зворотних в'язків (feedback loop). Багатошаровий персептрон (MLP) відноситься до багаторівневої архітектури з усіма можливими зв'язками між шарами, які також позначаються як повністю зв'язані шари. Шари, у яких немає вхідних або вихідних нейронів, називають прихованими шарами (рис. 1.). Алгоритм використовує апроксимаційні моделі ієрархічно з глибиною, пропорційною загальному числу шарів NN. Апроксимуючі моделі правил прийняття рішень мають *структуру дерева*, вузли якого визначаються нейронами окремих шарів, а гілки якого мають значення, обчислені за формулою (1) для активацій нейронів, при цьому набір даних поділяється на менші підмножини. Для довільної мережі з k прихованими шарами заради зручності введемо позначення для кожного прихованого шару: $h_1, \ldots, h_k$. Таким чином, в результаті першого кроку, отримується *дерево рішень (DT),* що описує вихідний шар мережі через активації шару h_k. Наступним кроком алгоритму є обробка шару h_{k-1}. Для кожної умови із набору правил, отриманого на попередньому етапі, застосовується алгоритм C4.5 чи C5

[21, 22], щоб побудувати дерева рішень, які тепер будуть описувати шар h_k за допомогою h_{k-1}. Аналогічним чином, даний процес продовжується для кожного наступного шару, до тих пір, поки не буде отримано дерева рішень, що описують шар h_1 через входи нейронної мережі. При цьому, під час проходу алгоритму, застосовується механізм, що запобігає виконанню зайвих запусків C4.5. Якщо дерево рішень для опису певного випадку вже було побудоване раніше, наявні результати будуть просто скопійовані. Після отримання DT, що описують кожен шар мережі, виконується об'єднання отриманих дерев разом. Усі невідповідні та зайві правила в процесі викидаються. В результаті формується одне дерево рішень, що описує виходи NN на основі її входів, тобто описує поведінку самої нейронної мережі. На рис. 3 зображено псевдокод реалізації DeepRED, представлений автором алгоритму в його роботі [7].

ПОКРАЩЕННЯ РЕАЛІЗАЦІЇ АЛГОРИТМУ

Із псевдокоду, представленого на рис. 3, видно, що для ініціалізації дерев на першому кроці, в своїй первинній формі, алгоритм використовував набли-

```
Input: Neural network NN, training examples x
Output: Set of rules rules

activationValues = getHiddenActivationValues(NN,x)
activationValues(outputLayer) = NN(x)                          // one-hot encoded
foreach currentOutput ∈ outputNeurons do
    // intermediateTerms stores terms that describe higher-level terms
    intermediateTerms(outputLayer, 0) = currentOutput > 0.5
                          // currentOutput is the class of interest, 0 used as dummy
    foreach currentLayer ∈ hiddenLayersDescending(NN) do
        foreach term ∈ intermediateTerms(currentLayer+1) do
            if treeAlreadyExtractedFor(term, currentLayer+1) then
                intermediateTerms(currentLayer, term) = copyTermsFor(term, currentLayer+1)
            else
                intermediateTerms(currentLayer, term) = C4.5(activationValues(currentLayer),
                activationValues(currentLayer+1), term)
                        // Describe term in next deeper layer by terms of current layer
            end
        end
    end
    while getNumberOfLayers(intermediateTerms) > 2 do
        intermediateTerms = mergeIntermediateTerms(getNumberOfLayers(intermediateTerms),
        getNumberOfLayers(intermediateTerms)-1)
        intermediateTerms = deleteUnsatisfiableTerms(intermediateTerms)
        intermediateTerms = deleteRedundantTerms(intermediateTerms)
    end
    rules[] = intermediateTerms2Rules(intermediateTerms)
                // Describes currentOutput by rules consisting of input neuron splits
    delete(intermediateTerms)
end
```

Рис. 3. Псевдокод оригінальної реалізації алгоритму DeepRED [7]

ження типу: ***"IF output_i > 0,5 THEN class = i"***. Основною причиною цього є те, що оригінальна робота була зосереджена лише на проблемі бінарної класифікації, де використання таких правил доцільне. Проте, якщо потрібно застосовувати DeepRED у задачах з багатокласовою класифікацією, таке наближення частіше всього буде занадто грубим. Альтернативою в даному випадку є побудова DT на основі активацій останнього прихованого шару та результатів класифікації мережі в якості вхідних та вихідних векторів відповідно[14].

Оскільки взаємодії між шарами NN швидше мають форму графа, а не дерева, набагато точніший та більш виразний опис того, що відбувається в процесі злиття правил, забезпечує використання DDAG (Decision directed acyclic graph, або орієнтований ациклічний граф рішень) отриманих з DT, замість самих DT. Це, можна сказати, розширення до привичного дерева рішень. Єдиною відмінністю DDAG від DT є лише те, що він може мати структуру, відмінну від деревоподібної, тобто вузли можуть мати більше одного вхідного ребра (рис. 4). Завдяки такій структурі, DDAG має ряд пе-

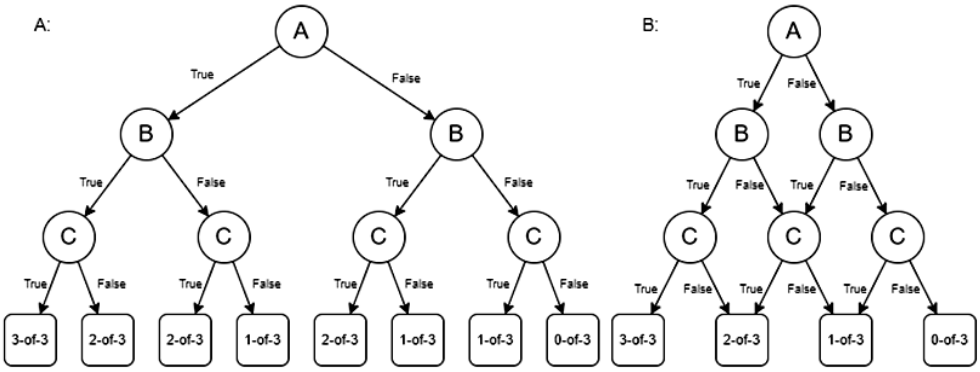


Рис. 4. Представлення правил M-of-{A,B,C} у вигляді: *а* — DT, *б* — DDAG

реваг для вирішення даного завдання, основними з яких є економніше використання пам'яті (що грає чималу роль при роботі в середовищі з обмеженими ресурсами) та складніша оцінка представлення правил. Також, за рахунок меншої кількості вузлів, що потрібні для представлення одного і того ж набору правил, порівняно з DT, DDAG набагато зручніші при їх аналізі, оскільки мають більш зручну для читання форму.

Algorithm 3.1 DeepRED pseudocode using DDAG

function DEEPRED($(h_l(X))_{l=0}^{L}$)
  $g_L \leftarrow$ INIT_DT($h_L(X)$)
  for $l = L-1, L-2, \ldots, 0$ **do**
    Set T as all unique splits in $ddag_{h_{l+1}}$
    for all $t \in T$ **do**
      $\tilde{X} \leftarrow h_l$
      $\tilde{Y} \leftarrow t(h_l(X))$
      $DT_t \leftarrow$ BUILD_DT($\tilde{X}, \tilde{Y}$)
    end for
    $g_l \leftarrow$ SUBSTITUTE($dag_{l+1}, (DT_t)_{t \in T}$)
    $g_l \leftarrow$ REMOVE_UNSATISFIABLE(g_l)
    $g_l \leftarrow$ REMOVE_REDUNDANT(g_l)
  end for
  return g_0
end function

Рис. 5. Псевдокод модифікованої реалізації алгоритму DeepRED

Псевдокод модифікованої реалізації алгоритму DeepRED приведений на рис. 5. Спочатку будується DT для останньої активації hL, тобто обчислюються аргументи максимуму $hL(x)$ для кожного значення $x \in X$.

Після ініціалізації цикл *for* продовжує розглядати шари у зворотному напрямку. На кроці *l* створюються DT для кожного розділеного вузла в DDAG gl+1. Потім DTs замінюють вузли всередині DDAG, створюючи таким чином новий DDAG gl із вхідними даними нижнього рівня. Нарешті, виконується виключення незадовільних та надлишкових вузлів з умов відповідності всіх вхідних даних вузлів значеням їх функцій активації. Заміна вузла за допомогою DT виконується шляхом переключення вхідних ребір *кореневого вузла (root*) DT і підключення усіх ребер, що ведуть до *листових (leaf) вузлів True і False* (істинних та хибних), до істинних та хибних ребер початкового вузла відповідно.

РЕЗУЛЬТАТИ ДОСЛІДЖЕНЬ

Як нейронну мережу, на якій досліджувалася задача класифікації рукописних символів, взято MLP з двома прихованими шарами, розмірами в 100 та 30 нейронів. На всіх проміжних шарах мережі використовувалася функція активації ReLU, а як функція активації вихідного шару застосовувалася Softmax. Таким чином, враховуючи розмірності вхідних та вихідних векторів, використовувалася нейронна мережа з наступними розмірами всіх шарів:

- 784 — вхідний шар (відповідає вектору вхідних значень);
- 100 — перший прихований шар;
- 30 — другий прихований шар;
- 2 — вихідний шар (відповідає вектору результатів класифікації).

Для навчання цієї експериментальної NN, з якої в ході експериментів вилучались правила прийняття рішень, використовувалась база даних рукописних цифр MNIST (Modified National Institute of Standards and Technology dataset) [23]. Вона містить навчальну вибірку розміром 60000 зображень та тестову вибірку розміром 10000 зображень. Всі зображення бази монотонні (у відтінках сірого різної інтенсивності) розміром $28 \times 28 = 784$. Для того, щоб сформувати вхідну вибірку для бінарної класифікації, притаманної алгоритму DeepRED, із бази даних MNIST було відібрано лише зображення, що містять цифри 0 та 1 (рис. 6). Кожен елемент вектора вхідних значень приймає значення в діапазоні [0, 1], де 0 — означає порожній піксель, а 1 — повністю зафарбований піксель.

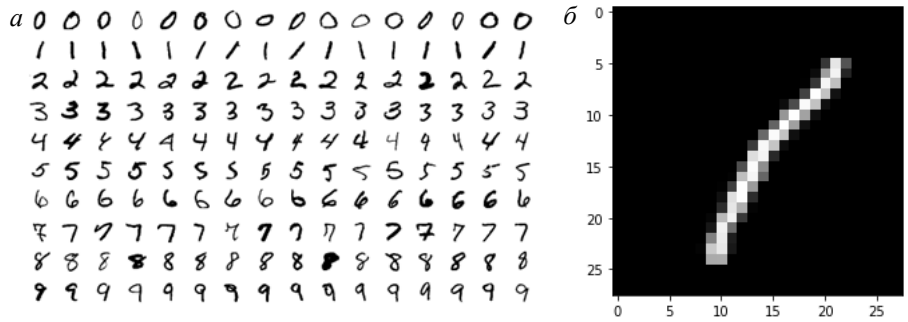


Рис. 6. Зображення бази даних MNIST: *а* — приклади зображень вибірки, *б* — перетворення зображення у вектор вхідних значень

Процес прийняття рішення щодо класифікації зображення з цифрою "1" шляхом проходження вузлами отриманого графа при вилученні правил ілюструється графом, який із-за своїх розмірів наданий окремо у форматі PDF-файла за посиланням: https://drive.google.com/file/d/1-dzEd8GsNpLpAlHl-zP-sQZjafxFgbs5/view?usp=sharing

На цьому графі в даному випадку, вузол {0} відповідає результату класифікації, який стверджує, що вхідне зображення містить цифру 0, а вузол {1}, відповідно, що вхідне зображення містить цифру 1. Інші вузли графа, що відповідають за відображення правил прийняття рішень нейронної мережі мають вигляд правил у форматі "$x_i >$ intensity", де x_i — це значення *i*-го елемента вектора вхідних значень, що відповідає інтенсивності відповідного пікселя зображення (відповідно до рис 6, *б*), а intensity — це значення інтенсивності, вилучене з "логіки" нейронної мережі, з яким ведеться порівняння інтенсивності пікселя зображення.

Порівняльні дослідження алгоритму DeepRED проводилися для випадків: бінарної класифікації, класифікації з 3-ма класами та класифікації з 4 ма класами. Відповідні результати перших двох запусків алгоритму наведені у табл. 3 та табл. 4. У них використано такі показники для порівняльного аналізу:

- DDAG fidelity — точність класифікації DDAG відносно класифікації NN на тестовій вибірці;
- DDAG accuracy — точність класифікації DDAG відносно еталонних значень тестової вибірки;
- DDAG recall — точність класифікації DDAG відносно класифікації NN на навчальній вибірці;
- final DDAG size — розміри фінального графа вилучених правил;
- DDAG size before pruning — розмірами графа рішень, перед видаленням зайвих правил.

З таблиць видно, що складність алгоритму значно зростає зі збільшенням кількості класів, а також зі збільшенням кількості шарів NN та їх розмірів завдяки зростанню розмірів побудованих DDAG, що призводить до росту обчислювальної складності. Окрім того, кінцеві результати сильно залежать від того, як буде ініціалізовано дерево на першому кроці.

Таблиця 3. Результати запусків Deep RED у випадку бінарної класифікації (зображення з цифрами 1 та 3)

Конфігурація MLP	NN accuracy	DDAG fidelity	DDAG accuracy	DDAG recall	Final DDAG size	DDAG size before pruning
MLP(20)	0.9995	0.9674	0.9748	0.9669	26	27
MLP(50)	0.9995	0.9646	0.9753	0.9732	16	16
MLP(100)	0.9991	0.9655	0.9711	0.9698	20	20
MLP(300)	0.9995	0.9734	0.9739	0.9713	18	18
MLP(50,25)	0.9986	0.9347	0.9608	0.9546	46	51
MLP(100,30)	0.9995	0.9669	0.9776	0.9722	28	43
MLP(300, 150)	0.9995	0.9757	0.9795	0.9693	19	19
MLP(50, 25, 9)	0.9986	0.9618	0.9734	0.9650	11	14
MLP(100,50,30)	0.9936	0.9417	0.9548	0.9522	65	69

Таблиця 4. Результати запусків Deep RED у випадку класифікації з 3 класами (зображення з цифрами 0, 1, 2)

Конфігурація MLP	NN accuracy	DDAG fidelity	DDAG accuracy	DDAG recall	Final DDAG size	DDAG size before pruning
MLP(20)	0.9930	0.9005	0.8967	0.8888	88	113
MLP(50)	0.9946	0.8599	0.8590	0.8579	91	109
MLP(100)	0.9939	0.8815	0.8770	0.8778	109	132
MLP(300)	0.9936	0.8561	0.8513	0.8557	87	163
MLP(100,30)	0.9940	0.7353	0.7426	0.7252	214	451
MLP(300,150)	0.9952	0.5329	0.3943	0.3942	147	328
MLP(50, 25, 9)	0.9942	0.8780	0.6222	0.6143	123	332
MLP(100,50,30)	0.9959	0.7925	0.4559	0.4499	134	357

ВИСНОВКИ

У процесі дослідження алгоритму DeepRED виявлено, що один із найбільш перспективних способів його використання — це застосування в поєднанні з процесом навчання нейронної мережі. За рахунок того, що вилучення правил проводиться безпосередньо під час навчання мережі, з'являється краще розуміння того, як та чи інша зміна в архітектурі NN впливає на отримані результати. По-суті, при правильному використанні результати вилучення правил можуть бути застосовані у вигляді додаткового критерія оцінки якості навчання.

На сьогоднішній день, вимірювання точності результатів класифікації на тестовій вибірці — це ледь не єдиний критерій, яким керуються під час навчання нейронних мереж. Але насправді, ця точність не завжди в повній мірі відображає те, наскільки добре пройшов навчальний етап. Якщо в однакових умовах з однієї мережі вдається вилучати чіткі та лаконічні правила, а з іншої — дещо ускладнені та заплутані, і, при цьому, точність у них однакова, логічніше буде віддати перевагу першій, оскільки, в загальному випадку, вона буде більш передбачувана.

Опираючись на результати, що були отримані в рамках проведених практичних експериментів, можна сказати, що вилучення правил з нейронних мереж за допомогою декомпозиційного алгоритму DeepRED виглядає досить перспективно. Вилучені за допомогою алгоритму правила мають досить зрозумілу форму та відносно не складні при їх аналізі. Особливо вдалим рішенням, хотілося б відмітити ідею покращення алгоритму, за рахунок розширення дерев рішень до графів рішень (DDAG). Це дозволило сильно полегшити можливості читання отримуваних правил. Звісно, варто також зазначити, що проблеми зі зручністю аналізу вилучених правил все ще будуть актуальними при досить великих розмірах DDAG, але як уже відмічалось, такі алгоритми постійно балансують в рамках компромісу точності та зрозумілості результатів.

Підсумовуючи все, можна зробити висновок, що вирішувати повноцінно задачу вилучення правил з нейронних мереж, у тій формі, у якій DeepRED представлений на даний момент, все ще досить проблематично. Основною проблемою, яка сильно ускладнює можливість його масового використання, є проблема універсальності. За рахунок того, що алгоритм здатний працювати лише з MLP, відсіюється дуже велика частка можливостей, щодо застосування нейронних мереж у критичних сферах. Адже, на сьогоднішній день, найбільш перспективними нейронними мережами, для вирішення складних задач є мережі, що мають архітектуру, складнішу за MLP. Проте, в той же час, варто розуміти, що DeepRED — це найкращий декомпозиційний алгоритм для вилучення правил, серед тих, що існують на даний момент. При цьому, сама ідея, на якій засновано даний алгоритм, виглядає перспективною, і, схоже на те, що можливостей щодо її покращень ще досить багато. Зокрема чекає вирішення проблеми з сильною залежністю від етапу початкової ініціалізації дерева і ообота над подальшої оптимізацією розмірів графів рішень, наприклад, за рахунок використання вузлів M-of-N.

ЛІТЕРАТУРА

1. Frank Emmert-Streib, Zhen Yang, Han Feng, Shailesh Tripathi, and Matthias Dehmer, “An Introductory Review of Deep Learning for Prediction Models With Big Data,” *Front. Artif. Intell.*, 2020. Available: https://www.frontiersin.org/articles/10.3389/frai.2020.00004/full
2. H. Jacobsson, “Rule Extraction from Recurrent Neural Networks: A Taxonomy and Review,” *Neural Computation*, vol. 17, no. 6, pp. 1223–1263, 2005. Available: https://doi.org/10.1162/0899766053630350
3. A. Bondarenko, L. Aleksejeva, V. Jumutcs, and A. Borisovs, “Classification Tree Extraction from Trained Artificial Neural Networks,” *Procedia Computer Science*, vol. 104, pp. 556–563, 2017. Available: https://doi.org/10.1016/j.procs.2017.01.172
4. Hiroshi Tsukimoto, “Extracting rules from trained neural networks,” *IEEE Transactions on Neural networks*, 2000. Available: https://ieeexplore.ieee.org/document/839008
5. Ismail Taha and Joydeep Ghosh, “Symbolic interpretation of artificial neural networks,” *Knowledge and Data Engineering, IEEE, 1999.* Available: https://ieeexplore.ieee.org/document/774103
6. J.M. Benitez, J.L. Castro, and I. Requena, “Are artificial neural networks black boxes?” *IEEE Transactions on Neural Networks*, 1997. Available: https://ieeexplore.ieee.org/document/623216
7. J.R. Zilke, E.L. Mencia, and F. Janssen, “DeepRED–Rule extraction from deep neural networks,” in *T. Calders, M. Ceci, D. Malerba (Eds.): Discovery Science 19th International Conference, DS 2016, Bari, Italy, Oct. 19–21, 2016, Proceedings, LNAI 9956*, pp. 457−473, 2016. Available: https://doi.org/10.1007/978-3-319-46307-0_29
8. Kamal Kumar Sethi, Durgesh Kumar Mishra, and Bharat Mishra, “Extended Taxonomy of Rule Extraction Techniques and Assessment of KDRuleEx,” *International Journal of Computer Applications*, 2012. Available: https://www.researchgate.net/publication/258652014_Extended_Taxonomy_of_Rule_Extraction_Techniques_and_Assessment_of_KDRuleEx

9. Kamal Kumar Sethi, Durgesh Kumar Mishra, and Bharat Mishra, "KDRuleEx: A Novel Approach for Enhancing User Comprehensibility Using Rule Extraction," *Third International Conference on Intelligent Systems Modelling and Simulation, IEEE*, 2012. Available: https://ieeexplore.ieee.org/document/6169675
10. LiMin Fu, "Rule generation from neural networks," *IEEE Transactions on Systems, Man, and Cy-bernetics*, 1994. Available: https://ieeexplore.ieee.org/document/299696
11. M. Gethsiyal Augasta and Thangairulappan Kathirvalavakumar, "Reverse engineering the neural networks for rule extraction in classification problems," *Neural Processing Letters*, 2012. Available: https://www.researchgate.net/publication/216628594_Reverse_Engineering_the_Neural_Networks_for_Rule_Extraction_in_Classification_Problems
12. Makoto Sato and Hiroshi Tsukimoto, "Rule extraction from neural networks via decision tree induction," *International Joint Conference on Neural Networks, IEEE*, 2001. Available: https://ieeexplore.ieee.org/document/938448
13. Mark W. Craven and Jude W. Shavlik, "Extracting tree-structured representations of trained networks," *Advances in Neural Information Processing Systems*, 1996. Available: https://proceedings.neurips.cc/paper/1995/file/45f31d16b1058d586fc3be7207b58053-Paper.pdf
14. Matej Fanta, Petr Pulc, and Martin Holena, *Rules Extraction from Neural Networks Trained on Multimedia Data*. 2019. Available: http://ceur-ws.org/Vol-2473/paper4.pdf
15. Rudy Setiono and Wee Kheng Leow, "FERNN: An algorithm for fast extraction of rules from neural networks," *Applied Intelligence*, 2000. Available: https://link.springer.com/article/10.1023/A:1008307919726
16. Rudy Setiono, *Extracting M-of-N Rules from Trained Neural Networks*. National University of Singapore. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.37.4104&rep=rep1&type=pdf
17. Sebastian Thrun, "Extracting provably correct rules from artificial neural networks," *Technical Report*. University of Bonn, Institut für Informatik III, 1993. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.31.9441
18. Sebastian Thrun, "Extracting rules from artificial neural networks with distributed representations," *Advances in Neural Information Processing Systems*, 1995. Available: https://proceedings.neurips.cc/paper/1994/file/bea5955b308361a1b07bc55042e25e54-Paper.pdf
19. Tameru Hailesilassie, "Rule Extraction Algorithm for Deep Neural Networks: A Review," *International Journal of Computer Science and Information Security*, vol. 14, no. 7, 2016. Available: https://arxiv.org/ftp/arxiv/papers/1610/1610.05267.pdf
20. Zhi-Hua Zhou, Shi-Fu Chen, and Zhao-Qian Chen, "A statistics based approach for extracting priority rules from trained neural networks," *International Joint Conference on Neural Networks, IEEE*, 2000. Available: https://ieeexplore.ieee.org/ document/861337
21. S.L. Salzberg, *C4.5: Programs for Machine Learning by Quinlan J. Ross Quinlan*. Morgan Kaufmann Publishers, Inc., 1993, pp. 38–48. Available: https://doi.org/10.1007/BF00993309
22. Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, 2006.
23. Daniel Etzold, *MNIST — Dataset of Handwritten Digits*. 2015. Available: https://medium.com/mlearning-ai/mnist-dataset-of-handwritten-digits-f8cf28edafe

Надійшла 01.06.2023

INFORMATION ON THE ARTICLE

Anatolii I. Petrenko, ORCID: 0000-0001-6712-7792, Educational and Research Institute for Applied System Analysis of the National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Ukraine, e-mail: tolja.petrenko@gmail.com

Ilya A. Vokhranov, Educational and Research Institute for Applied System Analysis of the National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Ukraine.

NEURAL NETWORKS: STUDYING THEIR DECISION-MAKING RULES / A.I. Petrenko, I.A. Vokhranov

Abstract. The question of a better understanding of the behavior of neural networks is quite relevant, especially in industries with a high level of risks. To solve this problem, the possibilities of the new DeepRED decomposition algorithm, capable of extracting decision-making rules by deep neural networks with several hidden layers, are explored in the paper. The study of the DeepRED algorithm was carried out on the example of extracting the rules of an experimental neural network during the classification of images of the MNIST database of handwritten digits, which made it possible to reveal a number of limitations of the DeepRED algorithm.

Keywords: rule extraction, neural networks, DeepRED, machine learning, decision trees, decision graphs.