НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Навчально-науковий інститут прикладного системного аналізу
Кафедра системного проектування

До захисту допущено:
Завідувач кафедри
_____ Вадим МУХІН
«___»_____2022 р.

Магістерська дисертація

на здобуття ступеня магістра
за освітньо-професійною програмою
“Інтелектуальні сервіс-орієнтовані розподілені обчислювання”
зі спеціальності 122 “Комп'ютерні науки”
на тему: “Вилучення правил прийняття рішень нейронними мережами”

Виконав:
студент ІІ курсу, групи ДА-01мн
Вохранов Ілля Анатолійович _____

Науковий керівник:
Проф., доктор технічних наук
Петренко Анатолій Іванович _____

Рецензент:
Проф., доктор технічних наук
Глоба Лариса Сергіївна _____

Засвідчую, що у цій дипломній роботі
немає запозичень з праць інших авторів
без відповідних посилань.
Студент _____

Київ – 2022 року

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Навчально-науковий інститут прикладного системного аналізу

Кафедра системного проектування

Рівень вищої освіти – другий (магістерський)

Спеціальність – 122 "Комп'ютерні науки"

Освітньо-наукова програма – "Інтелектуальні сервіс-орієнтовані розподілені обчислювання"

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ Вадим МУХІН

«___» _____ 2022р.

ЗАВДАННЯ

на магістерську дисертацію студенту

Вохранову Іллі Анатолійовичу

1. Тема дисертації «Вилучення правил прийняття рішень нейронними мережами», науковий керівник дисертації: Петренко Анатолій Іванович, д.т.н., проф., затверджені наказом по університету від «___» _____ 2022 р. № _____.

2. Термін подання студентом дисертації: 11 червня 2022

3. Об’єкт дослідження: декомпозиційний алгоритм вилучення правил з глибинних нейронних мереж – DeepRED.

4. Предмет дослідження:

- Наукові статті до опрацювання
- Існуючі роботи з оглядом роботи алгоритму DeepRED
- Оригінальна робота автора алгоритму: «Jan Ruben Zilke. Extracting Rules from Deep Neural Networks. *Master's Thesis*, 2015»

5. Перелік завдань, які потрібно розробити

- Розглянути та проаналізувати предметну область: дослідити існуючі підходи до розшифровки автоматичних процедур прийняття нейронними мережами рішень щодо класифікації об’єктів.
- Дослідити математичну базу і особливості алгоритму DeepRED при його застосуванні для вилучення правил прийняття рішень нейронними мережами.
- Розгорнути експериментальний стенд в складі нейронної мережі і реалізації алгоритму DeepRED.
- Провести дослідження можливості щодо вилучення зрозумілих правил з нейронної мережі.
- Дослідити вплив на правила вилучення зміни архітектури нейронної мережі (наприклад, зміни кількості прихованих прошарків).
- Проведення підсумків за результатами виконаної роботи, оформлення звіту роботи.

6. Перелік графічного (ілюстративного) матеріалу:

- Презентація для захисту роботи.

7. Орієнтовний перелік публікацій

- Вилучення правил з нейронних мереж за допомогою алгоритму DeepRED

8. Дата видачі завдання: 31 січня 2022

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Строк виконання етапів магістерської дисертації	Примітка
1.	Отримання завдання	31 січня 2022	
2.	Збір інформації	7 лютого 2022	
3.	Ознайомлення з літературою і підготовка теоретичної частини роботи	18 лютого 2022	
4.	Аналіз вимог завдання, вибір методів і засобів розв’язання поставленої задачі	22 лютого 2022	
5.	Розробка архітектури експериментального стенду	4 березня 2022	
6.	Проектування макету стенду. Тестування стенду	18 березня 2022	
7.	Розробка програмної реалізації алгоритму DeepRED	2 травня 2022	

8.	Проведення експериментів на стенді під час функціонування нейронної мережі	15 травня 2022	
9.	Оформлення звіту практики	11 червня 2022	

Студент І.А. Вохранов

Науковий керівник А.І. Петренко

АНОТАЦІЯ

До магістерської дисертації Вохранова Іллі Анатолійовича на тему: «Вилучення правил прийняття рішень нейронними мережами»

Дана дисертаційна робота присвячена питанню вилучення правил прийняття рішень із нейронних мереж, що вирішують задачу класифікації, за допомогою декомпозиційного підходу DeepRED. Метою роботи є дослідження можливостей практичного використання алгоритму DeepRED для вилучення та аналізу правил.

В роботі розглядаються основні принципи процесу вилучення правил з нейронних мереж та проводиться дослідження алгоритму DeepRED. В ході дослідження сфери вилучення правил, проведено досить детальний розбір елементів архітектури нейронних мереж та принципу їх роботи (включаючи процес навчання). Задля кращого розуміння можливих підходів до вилучення правил в роботі також розглядаються існуючі на сьогоднішній день методи вилучення правил. В процесі виконання основної частини даної роботи, проводиться дослідження алгоритму DeepRED та можливості його практичного застосування. DeepRED – це найбільш перспективний декомпозиційний алгоритм вилучення правил на сьогоднішній день. Розгляд алгоритму і ряду його покращень, а також, аналіз результатів вилучення правил та порівняння серії його запусків за різних умов, дозволили отримати загальне уявлення щодо можливості його практичного використання та ряду обмежень, що присутні на даний момент.

Загальний обсяг роботи — 82 сторінки, 23 рисунки, 26 таблиць, 28 посилань.

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

ABSTRACT

a master’s degree thesis of Illia Vokhranov entitled “Extracting decision rules from trained neural networks”.

This work is devoted to the question of removing decision rules from neural networks that solve classification tasks, using the decomposition approach - DeepRED. The work aims to study the possibilities of practical use of the DeepRED algorithm to extract and analyze rules.

The paper considers the basic principles of the process of extracting rules from neural networks and conducts a study of the DeepRED algorithm. A detailed analysis of the architecture of neural network elements and principles of operation (including the learning process) was conducted while studying the area of rule extraction. To better understand the possibilities of rule extraction, we also looked through the available methods. In the process of performing the main part of this work, a study of the DeepRED algorithm and the possibility of its practical application was conducted. DeepRED is the most promising decomposition rule extraction algorithm. Consideration of the algorithm and a few of its improvements, as well as analysis of the results of removing rules and comparing a series of its launches under different conditions, gave us a general idea of its practical use and some limitations that are currently present.

The total volume of work is 82 pages, 23 figures, 26 tables, and 28 references.

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

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ І ТЕРМІНІВ

ML – machine learning – машинне навчання

NN – neural network – нейронна мережа

DNN – deep neural network – глибока нейронна мережа

DT – decision tree – дерево рішень

DDAG – decision directed acyclic graph – орієнтований ациклічний граф рішень

MLP – multilayer perceptron – багатошаровий перцептрон

SVM – support-vector machines – метод опорних векторів

RE – rule extraction – вилучення правил

ВСТУП

Нейронні мережі – це дуже потужні структури, принцип яких базується на функціональності людського мозку. Вони належать до галузі дослідження, яка називається машинним навчанням (ML). Головною характеристикою машинного навчання є здатність до навчання, тобто покращення продуктивності моделі в певній області на основі досвіду. Найбільш поширеними задачами машинного навчання є класифікація, регресія, кластеризація, оцінка густини та зниження розмірності.

У даній дисертаційній роботі основна увага буде приділятись задачі класифікації, а тому всі методи, підходи та рішення, відповідно, будуть розглядатись, в першу чергу, з перспективи придатності у вирішенні цього завдання. Усі алгоритми машинного навчання, що вирішують завдання класифікації, глобально можна розділити на дві категорії: ***описові***, тобто алгоритми, що створюють класифікатори, які читаються людиною (рішення класифікатора можна пояснити словами); і алгоритми типу ***чорний ящик***, які виконують класифікацію, але інформація про прийняття ними рішень прихована в параметрах конкретної обчислювальної моделі. Найбільш розповсюдженим прикладом описових класифікаторів є дерева рішень (DT), а найбільш популярним прикладом алгоритмів “чорного ящика”, в свою чергу, є нейронні мережі (NN). [18]

Достатньо великі нейронні мережі (глибинні нейронні мережі), здатні реалізовувати навіть дуже складні функції. Останнім часом було проведено багато роботи з покращення навчання цих структур, щоб вони могли набагато краще узагальнювати навчальні дані та формувати правильніші залежності. [9] Основною перевагою нейронних мереж є те, що вони можуть досягати набагато кращих результатів, ніж описові алгоритми. Моделі глибинного навчання проводять революцію у великій кількості галузей, таких як комп’ютерний зір, системи рекомендацій, обробка мови та багатьох інших. За ними закріпився

статус найсучасніших технік для вирішення багатьох задач машинного навчання. Завдяки гнучкості та потужності всіх видів нейронних мереж, деякі їх варіації досить добре підходять для багатьох складних завдань.

Опис проблеми

Незважаючи на те, що нейронні мережі досягають високої ефективності класифікації, у них є серйозний недолік: «люди не можуть ні будувати, ні розуміти нейронні мережі» [Artificial Intelligence: A Modern Approach, Stuart J. Russell, Peter Norvig, 1994]. Оскільки для їх побудови існують автоматизовані навчальні механізми, побудова не являється якоюсь критичною проблемою. Крім того, якщо в результаті побудови отримано не той результат, що очікувався, модель можна створити заново (з дещо зміненою архітектурою), і так доки не буде отримано бажаний результат. Хоч такий процес і може потребувати досить багато зусиль та часу, модель будується лише один раз, тому це допустимо. Проте, інша проблема – відсутність розуміння того, як нейронна мережа приймає рішення, іноді може бути дуже серйозною. [9] Нейронні мережі мають хороші перспективи застосування у багатьох системах, де безпека та надійність дуже критичні. Деякі з таких сфер: медична діагностика, фінансові ринки, енергосистеми, авіація, керування промисловими процесами і виявлення несправностей, моніторинг стану транспортних засобів та багато інших. Однак такі критично важливі системи потребують можливості перевірки надійності, інакше це стає загрозою для безпеки чи ризиком великих фінансових втрат. [25]

В той же час, описові алгоритми ML, на відміну від нейронних мереж, працюють у формі білого ящика. Підходи, засновані на правилах, такі як дерева рішень або прості правила “ЯКЩО-ТО”, набагато більш зрозумілі та не мають таких проблем з можливістю перевірки надійності. Але, при цьому, цими методами дуже складно досягти результатів, схожих до тих, що досягаються нейронними мережами. [18]

Перспективи вирішення проблеми

Потреби в розумінні процесів прийняття рішень NN постійно зростають з розширенням їх застосування у реальних задачах. Щоб подолати головну слабкість нейронних мереж, протягом останніх трьох десятиліть, детально вивчаються різні способи пояснення “логіки” прийняття рішень нейронними мережами. Першим запровадженим, та найбільш перспективним на сьогоднішній день, підходом у цій галузі є вилучення правил (RE) 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. [25] Більшість дослідників зосереджуються на тому, що вилученні правила мають бути максимально зрозумілими, але в той же час повинні якомога точніше імітувати поведінку нейронної мережі.

На сьогоднішній день, основним викликом в питанні вилучення правил є вилучення правил з глибинних мереж. Наявність великої кількості прихованих шарів у нейронних мережах сильно ускладнює цю задачу. На даний момент, робіт, в яких було б проведено детальний аналіз проблем вилучення правил із більш складних глибинних нейронних мереж досить мало. Більшість алгоритмів вилучення правил зосереджені лише на 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, після чого вони об’єднуються разом. Таким чином, формується набір правил, що відображає, як саме конкретна нейронна мережа приймає рішення, при генерації результату. [18]

1 НЕЙРОННІ МЕРЕЖІ

Перед тим, як поглиблюватись у питання вилучення правил з глибинних нейронних мереж, необхідно чітко розуміти, на основі яких внутрішніх залежностей вони функціонують та як саме нейронна мережа здатна приймати рішення щодо класифікації, чи будь-якої іншої задачі. Саме тому, в цьому розділі буде розглянуто основні елементи архітектури та принципи роботи нейронних мереж з перспективи вилучення правил прийняття рішень.

1.1 Основи NN

Як уже зазначалось раніше, принцип роботи нейронних мереж базується на роботі людського мозку, тобто, іншими словами, нейронна мережа - це сильно спрощена математична модель мозку людини. Це означає, що свої уявлення вона отримує завдяки використанню шарів нейронів. Загальну структуру всіх нейронних мереж можна звести до одного спільного вигляду: вхідний шар, вихідний шар та набір прихованих навчальних шарів (Рис. 1.1).

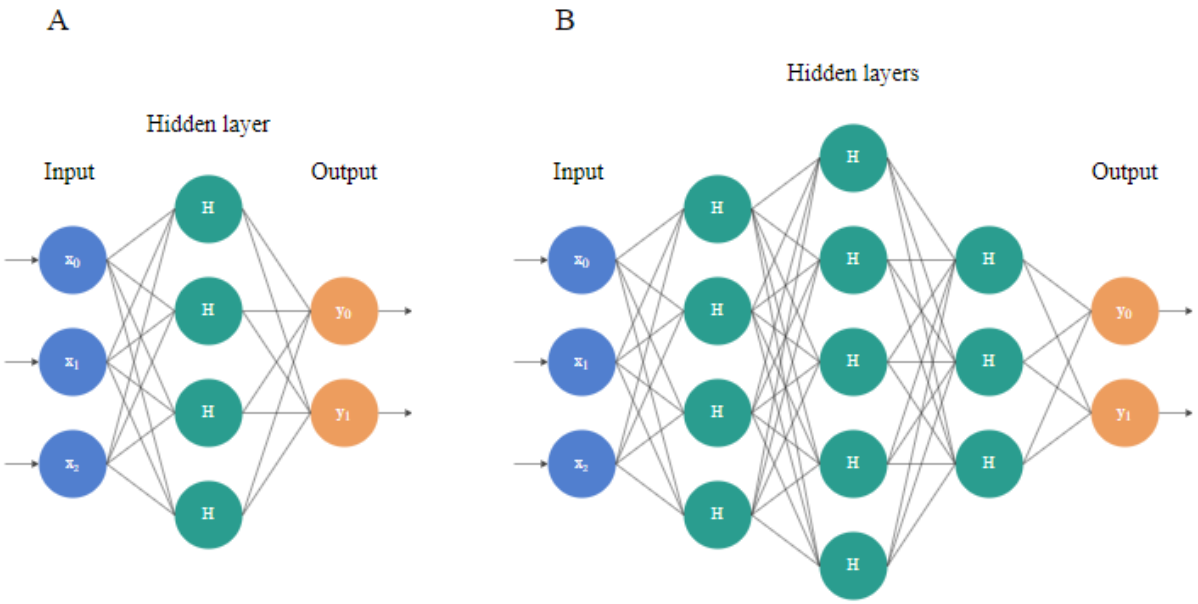


Рисунок 1.1 — Структура нейронних мереж; A – звичайної нейронної мережі, B – глибинної нейронної мережі

Склад навчальних шарів нейронної мережі та набір зв'язків між цими шарами, можуть дуже сильно варіюватись, в залежності від задач, які вона має вирішувати. Усі ці варіації і формують різні типи архітектур нейронних мереж. [2]

При розв'язуванні задачі класифікації, використання нейронної мережі, по-суті, виглядає просто як використання багатовимірної функції, тобто для заданих вхідних векторів повертається вихідний вектор, що складається зі значень без залежності від часу чи послідовності входів.

1.2 Модель штучного нейрона

Основною сутністю будь-якої нейронної мережі є модель нейрона (Рис. 1.2).

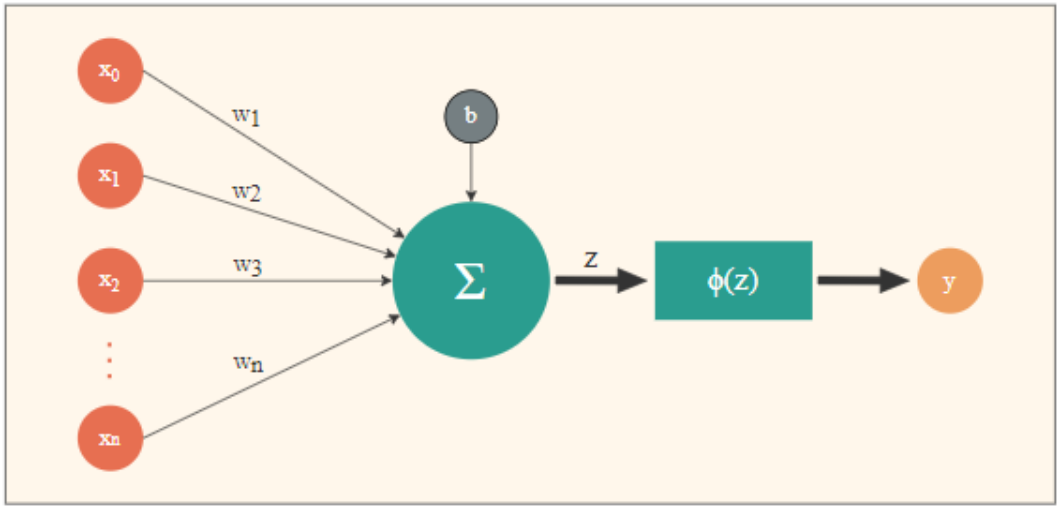


Рисунок 1.2 — Модель штучного нейрона

Основна 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 $z = w^T x + b$ виступає в якості аргумента функції активації ϕ, в результаті чого, формується вихід нейронної моделі:

$$y = \varphi(z) = \varphi(w^T x + b) \quad [2]$$

1.3 Функція активації

У більшості архітектур нейронних мереж, функція активації ϕ виконує нелінійне перетворення z, оскільки передбачається, що поділ між класами в більшості випадків також є нелінійним. Лінійний поділ можна отримати за допомогою перцептрона (найпростішої моделі нейронних мереж, що складається лише з одного нейрона), його функція активації – це східчаста функція (step function).

Однією з найпопулярніших функцій активації для глибинних нейронних мереж є функція активації ReLU (Rectified Linear Unit):

$$f(x) = x^+ = \max(0, x)$$

Іншою, часто використовуваною функцією активації є функція softmax:

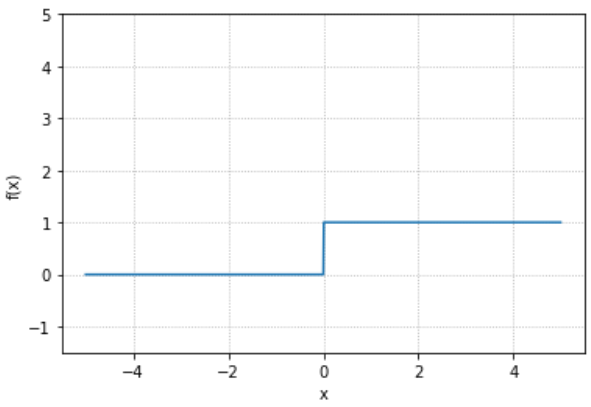
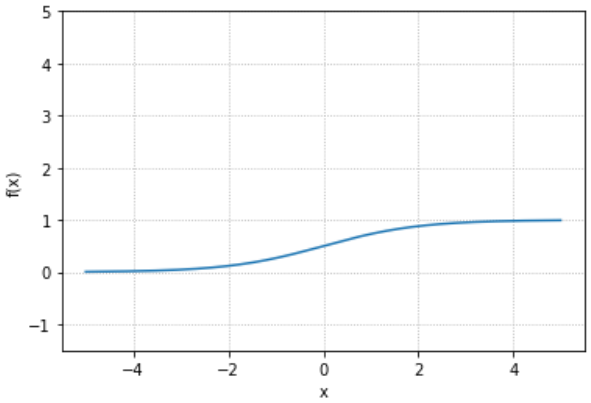
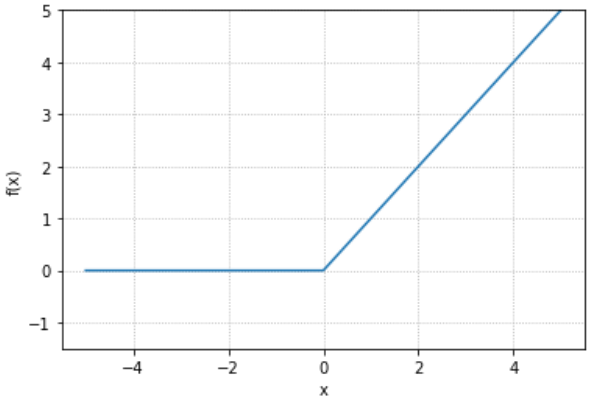
$$y_i = \frac{e^{x_i}}{\sum_{j=1}^{J} e^{x_j}}$$

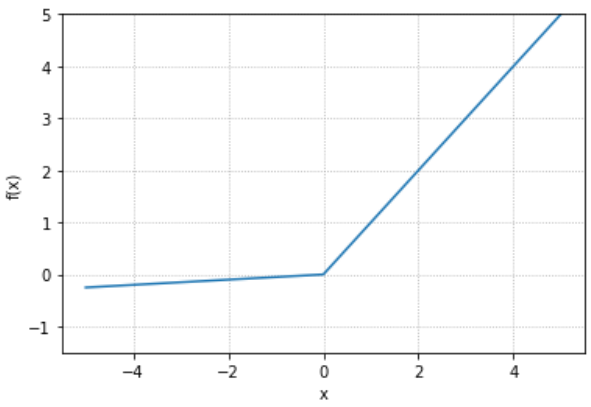
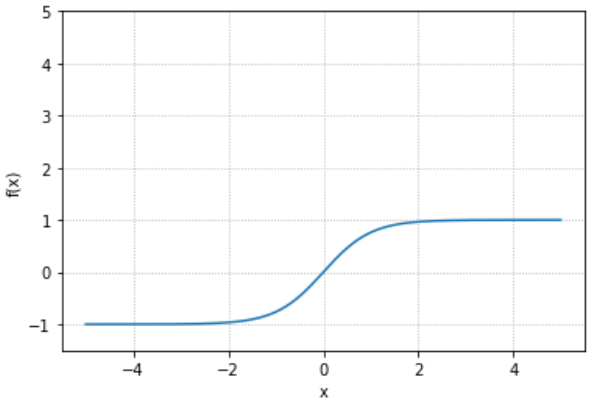
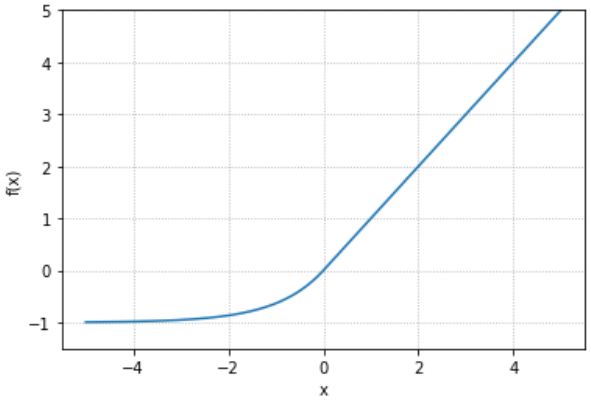
Вона відображає n-вимірний вектор x в n-вимірний вектор y, що має властивість $\sum_i y_i = 1$. Таким чином, компоненти у представляють ймовірності для кожного з n елементів. Softmax часто використовується в останньому шарі нейронної мережі.

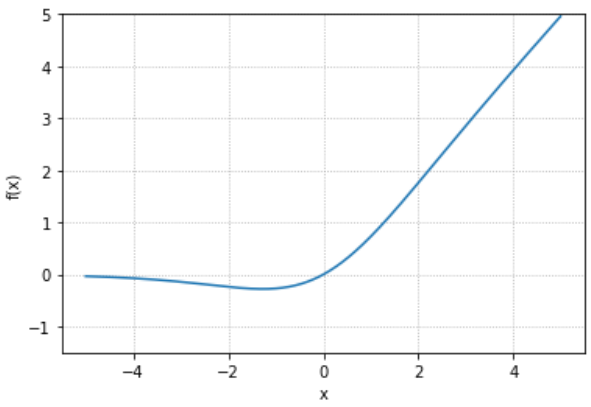
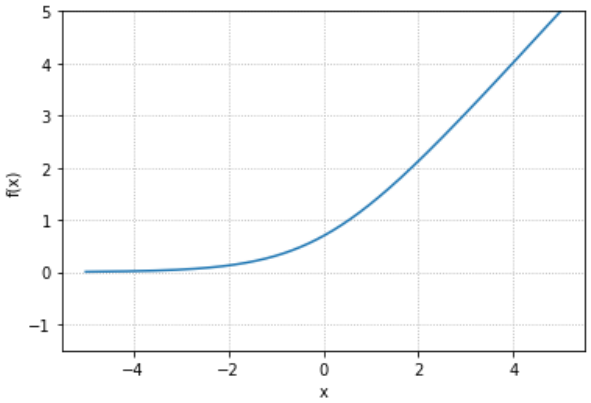
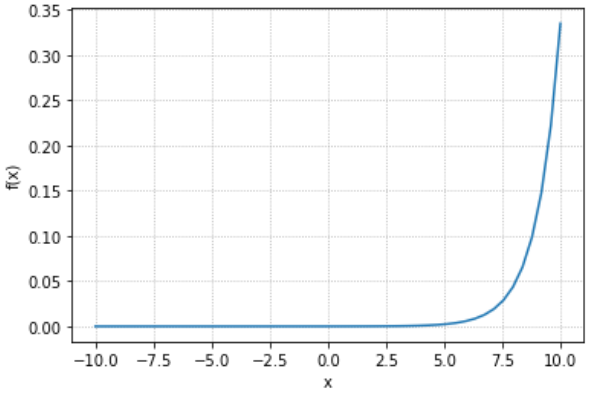
Короткий огляд функцій активацій, що зустрічаються в нейронних мережах найчастіше, з їх графічними зображеннями проведено в таблиці 1.1.

Таблиця 1.1 — Найбільш поширені функції активацій та їх графічні зображення

Назва	**Функція (φ)**	**Графічне зображення**

Step (східчаста)	$\varphi(x) = \begin{cases} 0 & if\ x < 0 \\ 1 & if\ x \geq 0 \end{cases}$	
Sigmoid (Logistic)	$\varphi(x) = \frac{1}{1 + e^{-x}}$	
Rectified Linear Unit (ReLU)	$\varphi(x) = \begin{cases} 0 & if\ x \leq 0 \\ x & if\ x > 0 \end{cases}$	

Leaky ReLU	$\varphi(x) = \begin{cases} 0.01x & if\ x < 0 \\ x & if\ x \geq 0 \end{cases}$	
Hyperbolic Tangent (Tanh)	$\varphi(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$	
Exponential Linear Unit (ELU)	$\varphi(x) = \begin{cases} \alpha(e^x - 1) & if\ x \leq 0 \\ x & if\ x > 0 \end{cases}$	

Swish	$\varphi(x) = \frac{x}{1 + e^{-\beta x}}$	
Softplus	$\varphi(x) = \ln(1 + e^{x})$	
Softmax	$\varphi(x) = \frac{e^{x_i}}{\sum_{j=1}^{J} e^{x_j}}$ *for i = 1, … J*	

1.4 Формування результату

Для всіх нейронів мережі, активації обчислюються рекурсивно, починаючи з вхідного шару. Виходи NN, відповідно, визначаються активаціями нейронів вихідного шару. Входи нейронів формуються або на основі зовнішнього набору

даних, або за рахунок виходів інших нейронів. Кожен нейрон мережі, обов’язково має зв’язок із хоча б одним нейроном іншого шару (а зазвичай з багатьма), і, при цьому, усі такі з'єднання зважуються певними вагами. Ваги визначають важливість кожного входу нейрона, тобто призначаються відповідно відносної важливості конкретного зв’язку, порівняно з іншими. Коли всі виходи нейронів з попереднього шару множаться на ваги та підсумовуються, утворюється значення, яке далі подається, як аргумент, до функції активації і, як результат формується вихід нейрона. Функції активації потрібні для внесення в можливості моделювання мережі нелінійності. Для нейронів кожного шару мережі, функція активації задана попередньо, і вона, по-суті, визначає, чи буде відповідний нейрон «активовано» та наскільки «активним» він буде у формуванні залежностей.

1.5 Навчання

По-суті, навчання нейронної мережі – це вирішення задачі оптимізації, ціллю якої є знаходження ваг, що мінімізують функцію втрат на навчальних даних. Основний підхід для знаходження локальних мінімумів було запроваджено ще кілька століть тому, і його принцип полягає в ітераційному переміщенні параметрів у напрямку градієнта (градієнтний спуск), в результаті чого, значення функції втрат падає на кожному кроці. Саме на цій ідеї основано більшість сучасних підходів (Adam, AdaGrad, AdaDelta та інші). Процес полегшення підбору параметрів всередині нейронної мережі, під час її навчання, відомий як оптимізація, а алгоритм, що використовується для вирішення цієї задачі, називається оптимізатором. Іноді оптимізатори можуть застосовувати більш складні підходи, порівняно з градієнтними спуском, наприклад, використовувати другу похідну для швидшої збіжності, але такі методи споживають багато пам’яті та обчислювальної потужності, тому вони не надто поширені при роботі з більшими чи глибшими NN. [18]

Якщо розглядати навчання мережі більш детально, то воно полягає в підборі якомога кращих значень ваг та зміщень (за принципом, описаним вище) для всіх внутрішніх зв’язків та нейронів. Цей процес ітераційний і під час кожної ітерації, відбувається пряме та зворотне поширення інформації мережею. Пряме поширення інформації відбувається, коли на вхід мережі подається набір вхідних даних із навчальної вибірки, і всі нейрони, шар за шаром, застосовують свої перетворення до значень, отриманих від нейронів попереднього шару, і передають результати до нейронів наступного шару. Після того, як усі нейрони виконають свої обчислення, буде досягнуто вихідного шару мережі, який сформує результат, що відповідає заданому вхідному набору значень. До отриманого результату, далі застосовується функція втрат, яка дозволяє оцінити помилку та зрозуміти, чи було отримано покращення внаслідок коригування вагів. Після розрахунку похибки, отримана інформація поширюється назад. Починаючи з вихідного шару, всі нейрони мережі отримують сигнал про втрати, з урахуванням відповідного відносного внеску, який кожен нейрон вніс в загальний результат. В ідеалі, ми хочемо, щоб втрати були нульовими, тобто очікувані та отримані результати не мали між собою розбіжності. Тому, під час навчання, ваги зв’язків нейронів будуть поступово коригуватися, доки не буде отримано хороші результати. [2]

2 ВИЛУЧЕННЯ ПРАВИЛ З NN

2.1 Постановка задачі

Як уже відмічалось у вступній частині даної роботи, основним недоліком штучних нейронних мереж є те, що вони працюють за принципом “чорного ящика”. Їх ефективність у вирішенні проблеми класифікації, дійсно перевершує інші підходи у переважній більшості випадків. Але, при цьому, відсутність пояснювальних можливостей, фактично, стає основним бар’єром для їх повноцінного використання в критичних сферах. Для того, щоб вирішити цю

проблему, та зробити нейронні мережі більш зрозумілими для людини, у 1990-х роках було проведено багато досліджень, які були зосереджені на вилученні правил прийняття рішень із NN, тобто описі їх поведінки (чи поведінки їх компонентів) у формі правил.

Підвищення прозорості нейронних мереж, шляхом вилучення правил, надає дві основні переваги. По-перше, це дає користувачеві певне уявлення про те, як нейронна мережа користується вхідними даними, у прийнятті своїх рішень. Таким чином, навіть, може розкриватись те, що в літературі часто називають прихованими функціями в NN, коли правила використовуються для пояснення окремих нейронів. А також, частиною розуміння може бути визначення деяких особливо важливих атрибутів, чи виявлення несправностей нейронної мережі. Глибина можливостей розкриття деталей функціонування нейронної мережі, звісно, сильно варіюється в залежності від підходу до вилучення правил, але так чи інакше, це призводить до ефекту кращого розуміння впливу вхідних даних та обраної структури NN на фінальний результат.

Другою перевагою підвищення прозорості NN, є те, що розуміння принципів та особливостей роботи конкретної нейронної мережі є обов’язковим, коли вона має використовуватись у сферах з критичною необхідністю передбачуваності та стабільності програмного забезпечення, таких як медична та фінансова сфери. У таких умовах, дуже важливо, щоб система забезпечувала можливість перевірки виходів штучної нейронної мережі за всіх можливих варіацій вхідних даних. Таким чином, використання методів вилучення правил може забезпечити можливість перевірки моделей на основі нейронних мереж, а це, в свою чергу, дозволить розширювати області застосування NN. [9]

Для того, щоб краще формалізувати завдання вилучення правил із нейронної мережі, можна звернутись до тверджень ряду дослідників, що займались цією проблемою раніше. Більшість авторів мають досить схожу думку в цьому плані: опис поведінки моделі, отриманий в результаті вилучення правил, має бути

зрозумілим, але при цьому дуже важливо, щоб він був максимально наближеним до поведінки мережі прогнозування. Приймати таке твердження за основу, дійсно має чималий сенс, оскільки, враховуючи те, що основною особливістю нейронних мереж є їх здатність вивчати досить складну поведінку, з урахуванням великої кількості факторів, будь-яке “віддалення” від такої поведінки при формуванні опису правил моделі, потенційно буде приводити до більшої кількості відхилень. В той же час, якщо основною метою досліджень є вирішення описаних раніше проблем, пов’язаних зі складністю застосування нейронних мереж у критичних сферах, де важлива максимальна передбачуваність моделей, будь-яке відхилення опису від справжньої поведінки моделі можна повноправно вважати похибкою отриманого результату. Таким чином, хоч основною ціллю вилучення правил є отримання якомога більш зрозумілих описів поведінки NN, також дуже важливо прагнути мінімізації похибки отриманих описів відносно оригінальної моделі. По-суті, всі підходи вилучення правил з нейронних мереж, постійно балансують між точністю та зрозумілістю отриманих результатів.

Для кращого орієнтування в предметній області та більш чіткого розуміння проблеми вилучення правил з нейронних мереж, у наступному підрозділі буде проведено загальний огляд основних методів вилучення правил, що вписуються у виділені рамки задачі.

2.2 Загальний огляд існуючих підходів до вилучення правил

Методи вилучення правил можна класифікувати за багатьма критеріями та особливостями. Найбільш традиційний поділ базується на тому, як різні підходи використовують архітектуру нейронної мережі. У такому форматі, методи поділяються на три групи – **декомпозиційні**, **педагогічні** та **еклектичні**. Педагогічні підходи розглядають NN як чорний ящик, тому їх можна використовувати навіть для інших класифікаторів, аніж NN, наприклад, SVM. З іншого боку, методи декомпозиції повністю спираються на архітектуру мережі і

використовують активації та ваги всіх нейронів, включаючи приховані шари. Еклектика – це змішування, поєднання різнорідних стилів, відповідно, в свою чергу, еклектичні підходи представляють собою поєднання інших підходів і розглядають лише частину NN як чорний ящик. [18]

Далі, в даному підрозділі, буде проведено поверхневий огляд основних представників з кожної групи підходів. Головною ціллю цього огляду є лише формування загальної картини того, які роботи були проведено дослідниками даної сфери, та як формувались і розвивались різні методи вилучення правил з NN, тому всі деталі функціонування цих підходів розглядатись не будуть.

2.2.1 Декомпозиційні підходи

Як уже зазначалось раніше, для вилучення правил із нейронних мереж, декомпозиційні підходи працюють на рівні нейронів. 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. [9]

Метод KT

Одним із перших декомпозиційних підходів для вилучення правил із нейронних мереж був метод KT (походить від слова knowlegetron), який представив Лімін Фу (Limin Fu) у 1994 році [13]. Даний алгоритм, для своєї роботи, потребує якомога простішої нейронної мережі з активаціями sigmoid. В процесі його виконання, відбувається прохід по всіх шарах мережі, в процесі чого, інформація про кожен нейрон і його ваги перетворюються в правила формату IF-THEN. Отримання цих правил відбувається шляхом пошуку комбінацій вхідних атрибутів, що перевищують поріг нейрона, тобто активують його. Результат створюється шляхом злиття всіх отриманих правил, шар за шаром, за принципом: вихідні дані одних нейронів є вхідними для інших.

Під час формування правил, алгоритм також використовує механізми, що дозволяють стримувати розростання дерева у ситуаціях, коли подальше покращення неможливе. Для цього, алгоритм шукає правила, що дублюються, і правила, що скасовуються іншими (надлишкові) та викидає їх. Цей процес має назву “pruning” (обрізання) і є досить розповсюдженою технікою стиснення даних в алгоритмах машинного навчання.

Перетворення у правила нечіткої логіки

У 1997 році в своїй роботі “*Are Artificial Neural Networks Black Boxes?*” Бенітез (Benítez) та ін. запропонували алгоритм опису нейронних мереж за допомогою правил нечіткої логіки [8]. Перевага запропонованого підходу над іншими полягає в тому, що представлений метод не просто наближає поведінку нейрона, а, по-суті, перетворює кожен нейрон у правило нечіткої логіки. Тобто, сформоване правило реалізує точно таку ж функцію, що й вихідний нейрон. Автори продемонстрували, що нейронні мережі та набір нечітких правил, сформованих певним чином є еквівалентними.

Звісно, очевидним недоліком такого представлення правил є більша складність інтерпретації поведінки нейронної мережі, тому автори додатково представляють ще процедуру переписування правил таким чином, щоб зробити їх більш зрозумілими.

Поліноміальний алгоритм Цукімото

У 2000 році була представлена робота Цукімото (Tsukimoto) – “Extracting rules from trained neural networks”, в якій автор представив свій метод вилучення правил із нейронної мережі [6]. Підхід Цукімото, за своїм принципом, дуже схожий на метод КТ, по-суті, його можна вважати покращенням методу КТ. Він так саме використовує алгоритм пошарової декомпозиції з формуванням правил IF-THEN для кожного окремого нейрона, та притримується стратегії пошуку вхідних конфігурацій, що перевищують поріг нейрона. Основною перевагою методу Цукімото є його обчислювальна складність, яка є поліноміальною, в той

час як у метода KT вона експоненційна. Алгоритм досягає поліноміальної часової складності шляхом пошуку відповідних термінів за допомогою багатолінійного функціонального простору. [6]

Також, у своїй роботі, окрім представлення поліноміального алгоритму, Цукімото побудував теорію навколо булевих функцій, яка дозволяє використання неперервних змінних в діапазоні [0, 1] замість використання лише дискретних {0, 1}. Іншими словами – ця теорія дозволяє вилучати правила нейронної мережі у формі нечіткої логіки. [6]

CRED

Іншим важливим етапом розвитку в сфері вилучення правил з нейронних мереж було представлення в 2001 році нового алгоритму під назвою CRED (Continuous/discrete Rule Extractor via Decision tree induction) авторами Сато та Цукімото (Makoto Sato, Hiroshi Tsukimoto) [15]. Даний алгоритм базується на ідеї опису поведінки мережі через дерева рішень (DT). Для побудови дерев рішень використовується алгоритм C4.5.

На першому кроці алгоритму, дерева рішень будуються на основі активацій прихованого шару та виходів NN. Тобто, таким чином, вузли дерева, є правилами, що описують поведінку нейронів прихованого шару, а листки представляють класи класифікації (Рис. 2.1). Кожен вузол, що представляє собою точку розходження гілок рішень, по-суті, відображає собою поріг точної активації нейрона. Як результат, на даному етапі, отримуються дерева, що описують вихідні класи через проміжні значення прихованого шару.

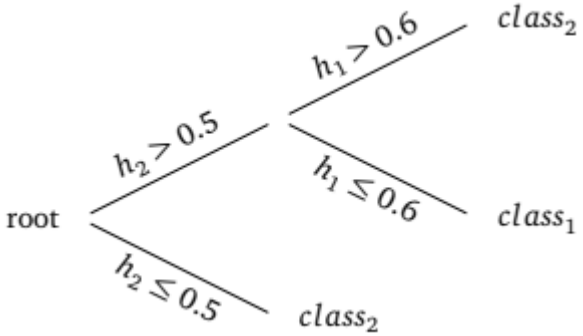


Рисунок 2.1 — Результат першого кроку CRED: дерево рішень, що описує виходи нейронної мережі на основі її прихованого шару [9]

На наступному кроці алгоритму, для кожного вузла поділу рішень будується інше дерево рішень, на основі вхідного шару нейронної мережі. У цих деревах рішень листи тепер відповідають не за класи класифікації, а за проміжні значення, що використовуються у вузлах попередніх дерев, як показано на рисунку 2.2. Таким чином, отримані на цій стадії дерева, описують стани нейронів прихованого шару мережі через вхідні змінні.

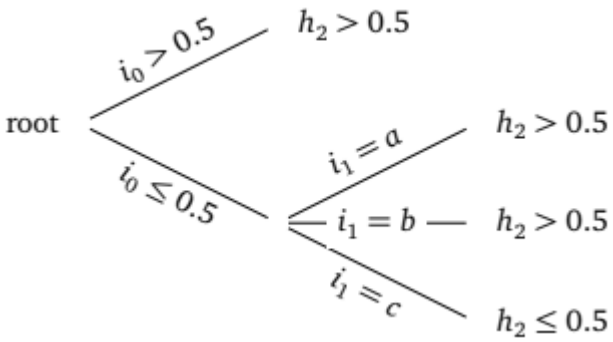


Рисунок 2.2 — Результат однієї ітерації другого кроку CRED: дерево рішень, що описує випадок “$h_2 > 0,5$” на основі входів нейронної мережі [9]

В результаті двох попередніх етапів алгоритму, тепер є два набори правил: проміжні правила, що описують виходи NN за допомогою прихованого шару, та правила, що описують прихований шар на основі вхідних даних NN. Відповідно, останнім кроком алгоритму буде заміна та об’єднання цих наборів правил таким чином, щоб в результаті було сформовано правила, що описують вихід нейронної мережі на основі її вхідних даних. В процесі цього злиття також застосовуються механізми спрощення правил, що викидають взаємовиключні та зайві правила.

Основним недоліком даного алгоритму є те, що він обмежений можливістю працювати лише з мережами MLP з одним прихованим шаром, тобто не може використовуватись для аналізу глибинних нейронних мереж.

DeepRED

Вирішення проблеми обмеженості CRED на роботі лише з одношаровими MLP було запропоновано Яном Зілке (Jan Ruben Zilke) у 2015 році. Автор представив алгоритм DeepRED (Deep Neural Network Rule Extraction via Decision Tree Induction) [9], який, по-суті, є розширенням CRED на загальний MLP. Основна ідея алгоритму полягає в представленні логіки роботи нейронної мережі за допомогою дерев рішень. Для цього, починаючи з останнього прихованого шару і проходячи рекурсивно до вхідного шару, DeepRED будує дерева рішень, для кожного шару. Після такого проходу, всі отримані дерева рішень об’єднуються разом, за принципом, подібним тому, що використовується в CRED.

Робота з алгоритмом DeepRED є основною ціллю даної роботи, тому більш детально його буде описано в наступному підрозділі.

2.2.2 Педагогічні підходи

Педагогічні підходи, 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, ця категорія алгоритмів намагається знайти узгодженість між вхідними варіаціями та результатами, при цьому, частина таких підходів додатково ще може використовувати навчальні дані.

VIA

У своїх роботах у 1993 та 1995 роках, Себастіан Тран (Sebastian Thrun) запропонував використовувати аналіз інтервалів валідності (Validity Interval Analysis, або VIA), для отримання правил, що імітують поведінку нейронної мережі [22][23]. Основна ідея методу полягає в тому, щоб знайти вхідні

інтервали, за яких вихід NN буде стабільним, тобто прогнозований клас буде залишатись тим самим при дещо змінених конфігураціях входів.

Алгоритм обирає певний приклад, або вхідну конфігурацію і випадковим чином збільшує інтервал випадкового вхідного параметра. Після цього проводиться перевірка чи результат збігається, чи отримується протиріччя. Якщо після збільшення інтервалу, VIA підтвердить свою дійсність (якщо результат збігається), його буде збережено, інакше інтервал буде скинуто до попереднього значення. Процес повторюється до тих пір, поки не будуть знайдені всі відповідні інтервали.

TREPAN

У 1996 році Кравен та Шавлік (Mark Craven, Jude Shavlik) представили алгоритм Trepan [16]. Стратегія вилучення правил, яку представили автори, полягає в тому, що нейронна мережа розглядається, як модель, що здатна приймати, так звані “запити”, що складаються з вхідних параметрів, та повертати відповіді у вигляді результату класифікації (тобто нейронна мережа виступає в ролі звичайної функції). Далі, в якості основи для вивчення правил, створюється великий штучний набір даних, який уже подається до алгоритму побудови дерева рішень, щоб згенерувати правила, які будуть імітувати поведінку нейронної мережі. Таким чином, в якості навчальної вибірки, на основі якої генеруються правила, використовують не лише дані, на яких безпосередньо відбувалось навчання моделі, а й інші вибіркові дані. Такий підхід, з використанням додаткових вибірок даних, отриманих із моделі, дуже поширений серед педагогічних алгоритмів, і алгоритм Trepan був одним із перших, де його було застосовано.

Алгоритм Trepan працює дуже подібно до звичайних алгоритмів індукції дерева рішень, як CART та C4.5. Він також будує дерево рішень шляхом рекурсивного розбиття простору екземплярів. Проте, на відміну від інших алгоритмів, Trepan будує дерево рішень за стратегією розширення найкращого

першого дерева (а не просто в глибину) та дозволяє дереву мати вузли у формі правил “M-of-N”. Крім цього, алгоритм також здатний відбирати додаткові навчальні приклади в більш глибоких точках дерева. В результаті, алгоритм будує дерево рішень, яке виступає в якості набору зрозумілих правил, що імітують роботу нейронної мережі. [17]

BIO-RE

У ситуації, коли має місце найпростіший випадок з двійковими входами, можна вважати, що нейронна мережа – це просто булева функція. Для цього випадку, автори Таха та Гош представили алгоритм BIO-RE (Binarized Input-Output Rule Extraction). Суть методу полягає в тому, що він створює всі можливі комбінації вхідних даних та пропускає їх через нейронну мережу. Використовуючи отримані результати, для кожного випадку будується таблиця істинності. Маючи цю велику штучну навчальну вибірку, далі можна використовувати будь-який зручний алгоритм побудови правил, що будуть моделювати поведінку нейронної мережі. В залежності від того, який алгоритм навчання далі буде використовуватись, уже буде залежати те, наскільки точно буде повторюватись поведінка мережі та наскільки зрозумілими будуть отримані правила. [9]

ANN-DT

У 1999 році Шміц (Gregor Schmitz) разом з кількома іншими дослідниками представили, дещо схожий за принципом на TREPAN, алгоритм ANN-DT [4]. Метод побудований на алгоритмі побудови дерев рішень CART, з деякими відмінностями в реалізації. ANN-DT використовує підхід розширення вибірки даних за допомогою мережі для отримання більшого навчального набору з умовою, що отриманий штучний навчальний набір все ще має бути репрезентативним. Для цього застосовується метод найближчого сусіда, у якому обчислюється відстань точки вибірки до найближчої точки в наборі навчальних даних і порівнюється з еталонним значенням.

STARE

Ідея з генерацією великих таблиць істинності для навчання, аналогічно до BIO-RE, також реалізована в алгоритмі STARE, запропонованому Чжоу (Zhi-Hua Zhou) та іншими в їх роботі в 2000 році. Основною перевагою STARE є його здатність обробляти не лише двійкові та дискретні атрибути, а й працювати з безперервними вхідними даними. Для формування таблиць істинності, алгоритм подає мережі різні вхідні параметри, при цьому, для кожного безперервного атрибута проводиться перебір в діапазоні його значень з високою частотою. Ще однією відмінністю від BIO-RE, також є те, що STARE має попередньо визначений алгоритм вилучення правил з отриманих таблиць істинності. В ньому, формування правил спершу проводиться на основі дискретних атрибутів, а уже тоді, коли нові правила з них витягти не виходить, далі використовуються безперервні атрибути.

KDRuleEx

За дуже схожим принципом до алгоритму Trepan працює також, представлений в 2012 році, алгоритм KDRuleEx [11][12]. Аналогічно до Trepan, він також генерує додаткові навчальні екземпляри, коли основа для побудови наступних точок розділення правил занадто мала. Для того, щоб створювати ці нові навчальні приклади, KDRuleEx використовує генетичний алгоритм. У результаті роботи цього методу, формується таблиця рішень, яка представляє собою опис правил, що симулюють поведінку нейронної мережі.

RxREN

У 2012 році було представлено досить нестандартний підхід до вилучення правил NN у роботі “Reverse engineering the neural networks for rule extraction in classification problems”. ЇЇ автори пропонують отримувати правила шляхом зворотної інженерії нейронних мереж і представляють, відповідно до концепції, свій алгоритм RxREN (Rule extraction by Reverse Engineering the Neural networks), який зосереджений на створенні максимально зрозумілих правил.

На першому етапі алгоритм видаляє незначні атрибути та знаходить обов’язкові діапазони входів мережі. Незначні атрибути визначаються шляхом аналізу продуктивності мережі, ігноруючи окремі вхідні атрибути. Якщо точність результатів прогнозування NN без відповідного атрибута не опускається нижче заздалегідь визначеного порогу, такий атрибут вважається незначним. Обов’язкові діапазони входів для пар клас-ознака визначаються вибірками значень, що змінюють результат класифікації мережі після скидання одного з атрибутів.

На другому етапі формуються правила, беручи за умову належність до всіх обов’язкових діапазонів для кожного класу. Тобто, для кожного класу та кожного значного атрибута, RxREN обчислює відповідні діапазони на основі неправильних результатів класифікації. Кожен такий діапазон має нижнє та верхнє порогове значення, в межах яких буде отримуватись потрібний клас. Далі, з отриманих інтервалів будуються правила. При цьому, застосовується ще процедура скорочення правил, під час якої, алгоритм визначає правила, які можна видалити без сильного впливу на результат.

2.2.3 Еклектичні підходи

Як уже зазначалось раніше, якщо методи виділення правил включають елементи як педагогічних, так і декомпозиційних підходів, тоді вони відносяться до еклектичних. Визначення еклектичних підходів дещо розпливчасте, але якщо намагатись описати загальні особливості їх роботи більш конкретно, то можна сказати, що еклектичні підходи використовують знання про внутрішню архітектуру нейронної мережі щоб доповнити високорівневий алгоритм навчання.

MofN

У 1993 році Товеллом і Шавліком (Geoffrey Towell, Jude Shavlik) було представлено алгоритм MofN [3], назва якого базується на його здатності витягувати правила M-of-N. MofN у більшій мірі поводить себе як

декомпозиційний алгоритм, намагаючись виділити правила, що будуть пояснювати окремі нейрони, але в той же час, він також застосовує техніку педагогічних підходів: розглядає важливість кожного окремого атрибута у загальному результаті класифікації мережі.

Основною ідеєю MofN є групування входів нейронів в категорії з приблизно однаковою вагою. Після цього, якщо якісь групи вхідних даних визначено як незначні, вони можуть бути видалені. Групи можуть вважатись незначними, якщо значення активації цілої такої групи не має істотного впливу на вихід нейрона, або, якщо результат NN не залежить від цих атрибутів. Останнім етапом роботи алгоритму є формування правил M-of-N за допомогою кластерів.

FERNN

Ще одним важливим підходом, який можна віднести до еклектичних є запропонований в 2000 році алгоритм FERNN (Fast Extraction of Rules from NN) [20]. Як і у випадку з попереднім підходом, він теж, в більшій мірі, працює як декомпозиційний, але при цьому розглядає важливість входів мережі та викидає не важливі.

Алгоритм використовує дерева рішень для вилучення правил між прихованим і вихідним шарами одношарових MLP. Побудова DT відбувається за допомогою алгоритму C4.5. Маючи набори правильно класифікованих навчальних прикладів, FERNN аналізує активації нейронів прихованого шару. Для кожного нейрона значення активації сортуються в порядку збільшення, після чого алгоритм C4.5 знаходить кращу точку поділу та формує дерево рішень. На наступному кроці видаляються нейронні входи, що не впливають на результат нейронної мережі. В кінці, з отриманого дерева рішень отримуються правила. Якщо це можливо, вузли поділу DT групуються та замінюються на правила у формі M-of-N, якщо ні, тоді з них формуються правила IF-THEN.

2.3 DeepRED

Як зазначається в оригінальній роботі [9], серед усіх представлених на момент дослідження підходів до вилучення правил, найбільш перспективно виглядав алгоритм CRED. Основними його перевагами виділяється те, що він повинен бути потенційно більш точним за педагогічні підходи, а також, повинен мати можливість вивчати такі складні проблеми, як проблему XOR, якщо вони закодовані у внутрішній структурі нейронної мережі. Саме тому, автором було вирішено розглянути можливість застосування CRED для вилучення правил з глибинних нейронних мереж.

В своїй оригінальній реалізації, CRED був розрахований лише на роботу з одним прихованим шаром, але після розширення його алгоритмічного підходу, шляхом отримання додаткових дерев рішень і проміжних правил для кожного додаткового прихованого шару, було отримано підхід, що дозволив працювати з багатьма шарами. Новий, модифікований алгоритм отримав назву DeepRED (Deep neural network Rule Extraction via Decision tree induction).

На сьогоднішній день, DeepRED представляє собою найбільш перспективний метод декомпозиції, що може використовуватись для вилучення правил з глибинних нейронних мереж. Але, при цьому, є важливе уточнення: алгоритм розрахований на роботу з MLP будь-якої глибини, але все ще не може застосовуватись до DNN з довільною архітектурою.

2.3.1 Опис алгоритму

На першому кроці алгоритму DeepRED, так саме, як і у випадку CRED, за допомогою C4.5, будуються дерева рішень на основі активацій нейронів останнього прихованого шару та виходів NN. Кожен вузол дерева представляє собою правило, що описує конкретний поріг активації відповідного нейрона прихованого шару, а кожен листок відповідає певному класу класифікації. Активації нейронів прихованого шару, що використовуються при побудові дерев

рішень, отримуються шляхом пропускання вхідної навчальної вибірки даних через нейронну мережу та запам’ятовування виходів усіх нейронів.

Для довільної мережі з k прихованими шарами, заради зручності, введемо позначення для кожного прихованого шару: $h_1, \dots, h_k$. Таким чином, в результаті першого кроку, отримується дерево рішень, що описує вихідний шар мережі через активації шару h_k. Тепер, оскільки мережа не обмежена лише одним прихованим шаром, наступним кроком алгоритму є обробка шару h_{k-1}. Для кожного терміна із набору правил, отриманого на попередньому етапі, застосовується алгоритм C4.5, 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 терміна вже було побудоване раніше, наявні результати будуть просто скопійовані.

В результаті проходу всіма шарами нейронної мережі, отримуються дерева рішень, що описують кожен шар нейронної мережі його попереднім шаром. Тепер, щоб отримати набір правил, що буде описувати виходи NN через її входи, усі ці правила потрібно об’єднати. Цей процес злиття відбувається пошарово у зворотному напрямку (починаючи з останнього шару). Подібним чином, до того, як проводяться об’єднання дерев рішень в CRED (див. *2.2.1 – CRED*), виконується об’єднання правил останнього і передостаннього шарів. Після цього, отримане, в результаті злиття, дерево рішень об’єднується з правилами наступного шару, і так, процес продовжується, доки не буде досягнуто вхідного шару мережі.

В процесі об'єднання наборів правил між собою, усі невідповідні та надлишкові проміжні правила видаляються. Це значно скорочує час обчислень та оптимізує використання пам’яті. В результаті об’єднання всіх проміжних

дерев рішень, на виході формується одне дерево рішень, що описує результати класифікації мережі відповідно до її входів, тобто описує поведінку самої нейронної мережі. [9]

На рисунку 2.3 зображено псевдокод реалізації DeepRED, представлений автором алгоритму в його роботі [9].

```
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
```

Рисунок 2.3 — Псевдокод оригінальної реалізації алгоритму DeepRED [9]

2.3.2 Покращення реалізації алгоритму

В деяких із робіт інших дослідників, які розглядали питання застосування алгоритму DeepRED, були запропоновані певні покращення щодо реалізації алгоритму. Далі буде описано два таких покращення, які мають підвищити універсальність та ефективність даного методу при його практичному застосуванні.

Ініціалізація DT

Як описувалось раніше, перший крок алгоритму полягає в ініціалізації DT, що забезпечує представлення вихідного шару мережі. Із псевдокоду, представленого на рисунку 2.3, видно, що в своїй первинній формі, алгоритм DeepRED для початкової ініціалізації використовував наближення за допомогою набору правил 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 абсолютно 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. [18]

Використання DDAG

DeepRED, аналогічно до алгоритму CRED, з якого він був розвинений, для представлення правил використовує дерева рішень. Проте, тепер, алгоритм розрахований на вилучення правил з глибинних нейронних мереж. Це означає роботу з багатьма прихованими шарами, які здатні будувати велику кількість внутрішніх зв'язків, формуючи досить складні послідовності правил.

Якщо розглянути загальну структуру багатошарової MLP, легко помітити, що взаємодії між нейронами різних шарів швидше нагадують форму графа, аніж

дерева. Через це, більш точний та якісний опис того, що відбувається в процесі злиття правил забезпечує застосування DDAG, отриманих з DT, замість самих DT. DDAG – decision directed acyclic graph, або орієнтований ациклічний граф рішень – це, по-суті, розширення до звичного 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. Завдяки такій структурі, DDAG має ряд переваг 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, які потрібні для представлення одного і того ж набору правил, порівняно з DT, DDAG набагато зручніші при їх аналізі, оскільки мають більш зручну для читання структуру.

2.3.3 Дерева рішень та графи рішень

Дерева рішень є одними з найпопулярніших та найпотужніших інструментів для класифікації та прогнозування. Дерево рішень представляє собою модель, що має структуру орієнтованого ациклічного графа деревовидної форми. Кожен внутрішній вузол дерева рішень відповідає за певну умову і має два вихідні ребра, які представляють можливе правило, в залежності від того, виконується умова вузла (true) чи ні (false). При цьому, кожен листовий вузол відповідає за один із класів фінального рішення.

Дерева рішень відносяться до моделей машинного навчання, що працюють за принципом “білого ящика”, оскільки мають просту в поясненні структуру. Древо рішень може бути легко перетворене в набір правил типу IF-THEN, для чого достатньо всього лише виконати проходи всіма шляхами від кореня до кожного листка та записати умови вузлів у послідовності правил. Проте, коли задача полягає в формуванні DT, еквівалентного певному набору правил, це уже може

бути більш складним завданням, оскільки правила можуть мати різні форми і не всі з них легко відтворити за допомогою дерева. [18]

Одним із найбільш показових прикладів проблеми представлення правил у вигляді дерева рішень, є N-бітові проблеми парності (N-bit parity problems). Частковим випадком N-бітових проблем парності є дуже відома проблема XOR (2-бітова проблема парності). Представлення таких логічних взаємозв'язків дуже зручно виконувати у вигляді правил M-of-N. Саме тому, серед підходів до вилучення правил з нейронних мереж, часто зустрічається використання цих правил. [21]

Якщо розглянути приклад представлення концепції правил M-of-N у вигляді класичної структури дерева рішень, то відразу стає помітною проблема необхідності використання великої кількості додаткових вузлів. На рисунку 2.4 продемонстровано принцип, за яким відбувається подання у вигляді дерева рішень правил *M-of-{A,B,C}*, де *A*, *B* і *C* відповідають 3-ом різним умовам. При цьому, якщо потрібно представляти такі правила, як, наприклад, "*at least 2-of-{A,B,C}*" чи "*at most 2-of-{A,B,C}*" ("*хоча б 2-із-{A,B,C}*" та "*не більше 2-із-{A,B,C}*" відповідно), тоді задовольняти умову буде відразу декілька листових вузлів. У випадку з "*at least 2-of-{A,B,C}*", умова задовольняється всіма листками "*3-of-3*" та "*2-of-3*".

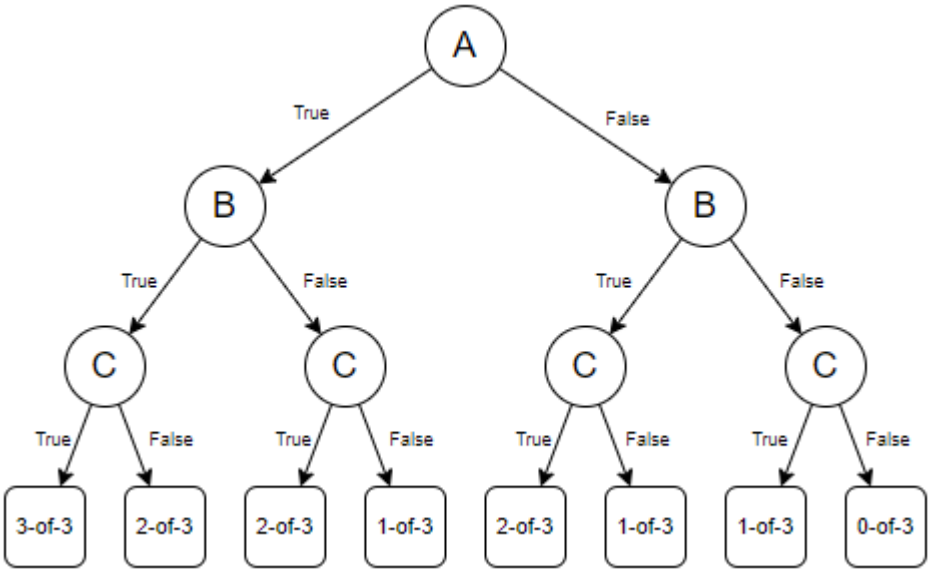


Рисунок 2.4 — Представлення правил M-of-{A,B,C} у вигляді DT

Як уже відмічалось раніше, для полегшення проблеми великої кількості надлишкових вузлів, DT може бути розширено до DDAG. Завдяки тому, що DDAG дозволяє в своїй структурі більше одного батьківського вузла, такий самий набір правил може бути представлений в набагато більш компактній формі (Рис. 2.5).

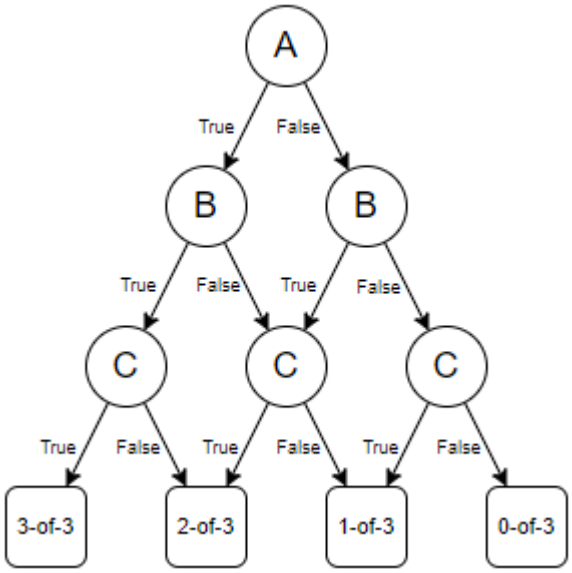


Рисунок 2.5 — Представлення правил M-of-{A,B,C} у вигляді DDAG

У випадку представлення правила “*at least 2-of-{A,B,C}*” за допомогою DDAG, листки, які відповідають одному результату будуть об’єднані, а зайві внутрішні вузли будуть видалені (Рис. 2.6).

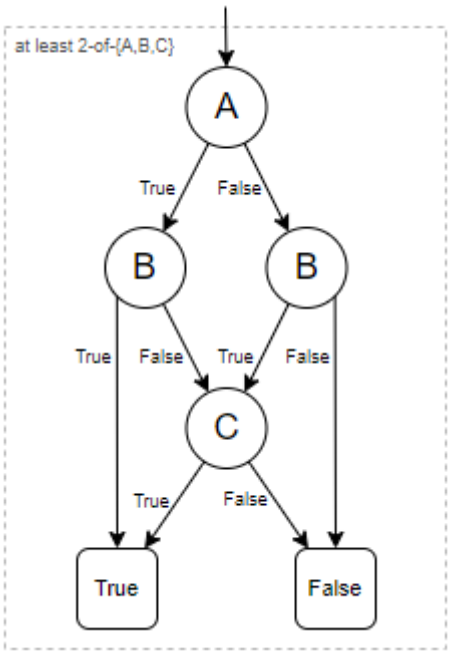


Рисунок 2.6 — Представлення правила “*at least 2-of-{A,B,C}*” у вигляді DDAG

3 ЗАСТОСУВАННЯ DEEPRED ДЛЯ ВИЛУЧЕННЯ ПРАВИЛ З NN

3.1 Вибірка даних

Для вилучення правил з нейронної мережі, алгоритм DeepRED використовує навчальну вибірку даних, на основі якої відбувалось навчання досліджуваної мережі. При роботі з DeepRED навчальна вибірка потрібна для забезпечення двох задач. По-перше, за допомогою навчальної вибірки отримуються активації внутрішніх шарів нейронної мережі. Основні деталі цього процесу будуть описані в наступному підрозділі. По-друге, оскільки очікується, що в якості результату буде отримано правила, що описують виходи нейронної мережі через її входи, необхідно чітко розуміти, що саме представляє собою кожен із входів нейронної мережі. Для цього, далі буде детальніше розглянуто вибірку даних, що буде використовуватись для навчання досліджуваних NN, та процес формування вектора вхідних значень, що подаватимуться на входи мережі.

3.1.1 MNIST

База даних MNIST (Modified National Institute of Standards and Technology database) – це велика база даних рукописних цифр, яка дуже часто використовується для навчання та тестування в області машинного навчання (Рис. 3.1). На сьогоднішній день, вона є стандартом для калібрування та порівняння штучних нейронних мереж у задачі класифікації. MNIST містить навчальну вибірку розміром 60000 зображень та тестову вибірку розміром 10000 зображень. Всі зображення бази монотонні (у відтінках сірого різної інтенсивності) розміром 28x28 пікселів.

Рисунок 3.1 — Приклади зображень бази даних MNIST

Усі зображення рукописних цифр в базі даних MNIST нормалізовані, центровані та не мають зайвого шуму. За рахунок цього і великої варіативності зразків, нейронні мережі дуже добре дістають із неї патерни та легко навчаються на її основі. Використання MNIST, в якості вхідної вибірки даних, добре підходить для дослідження роботи алгоритму DeepRED, оскільки дозволяє NN добре вивчати необхідні особливості, в рамках поставленої задачі, і, при цьому, не обов’язково мати складну архітектуру.

3.1.2 Формування вектора вхідних значень

Формування вектора вхідних значень, в даному випадку відбувається наступним чином: так, як зображення вибірки мають розміри 28x28 пікселів, для представлення кожного зображення формується вектор, що містить значення 784-ох пікселів у вигляді {*x_0, ..., x_783*} (Рис. 3.2). Кожен елемент вектора *x_i* приймає значення в діапазоні [0, 1], де 0 – означає порожній піксель, а 1 – повністю зафарбований піксель.

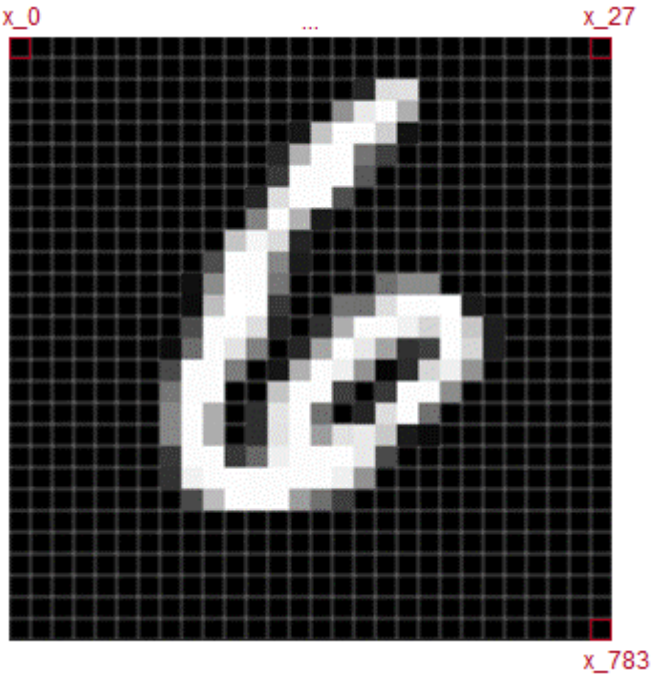


Рисунок 3.2 — Формування вектора вхідних значень

3.2 Нейронна мережа

Алгоритм DeepRED розрахований на вилучення правил прийняття рішень з глибинних нейронних мереж з архітектурою MLP (multilayer perceptron, або багатошаровий перцептрон). MLP – це модель штучної нейронної мережі з прямими зв’язками, що містить декілька шарів, кожен з яких повністю пов’язаний з наступним. Між вхідним і вихідним шарами моделі знаходиться один або кілька нелінійних прихованих шарів. Схематичне зображення мережі з такою архітектурою показано на рисунку 3.3.

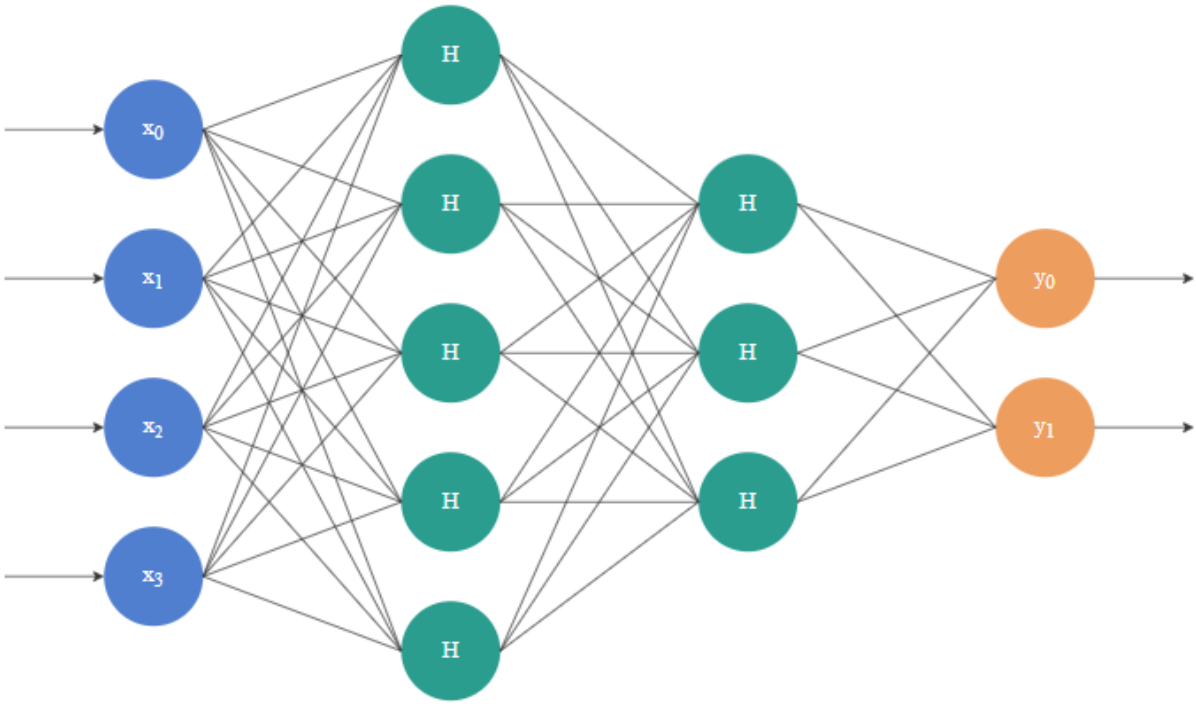


Рисунок 3.3 — Приклад нейронної мережі MLP з 2-ма прихованими шарами

Як уже описувалось раніше, отримання правил відбувається за рахунок навчання дерев рішень на основі активацій внутрішніх шарів нейронної мережі. Активації – це значення, що є результатами функцій активацій кожного нейрона під час обробки певного набору вхідних даних, іншими словами – це виходи нейронів, що подаються в якості входів у нейрони наступного шару (за винятком активацій вихідного шару, які є виходами нейронної мережі).

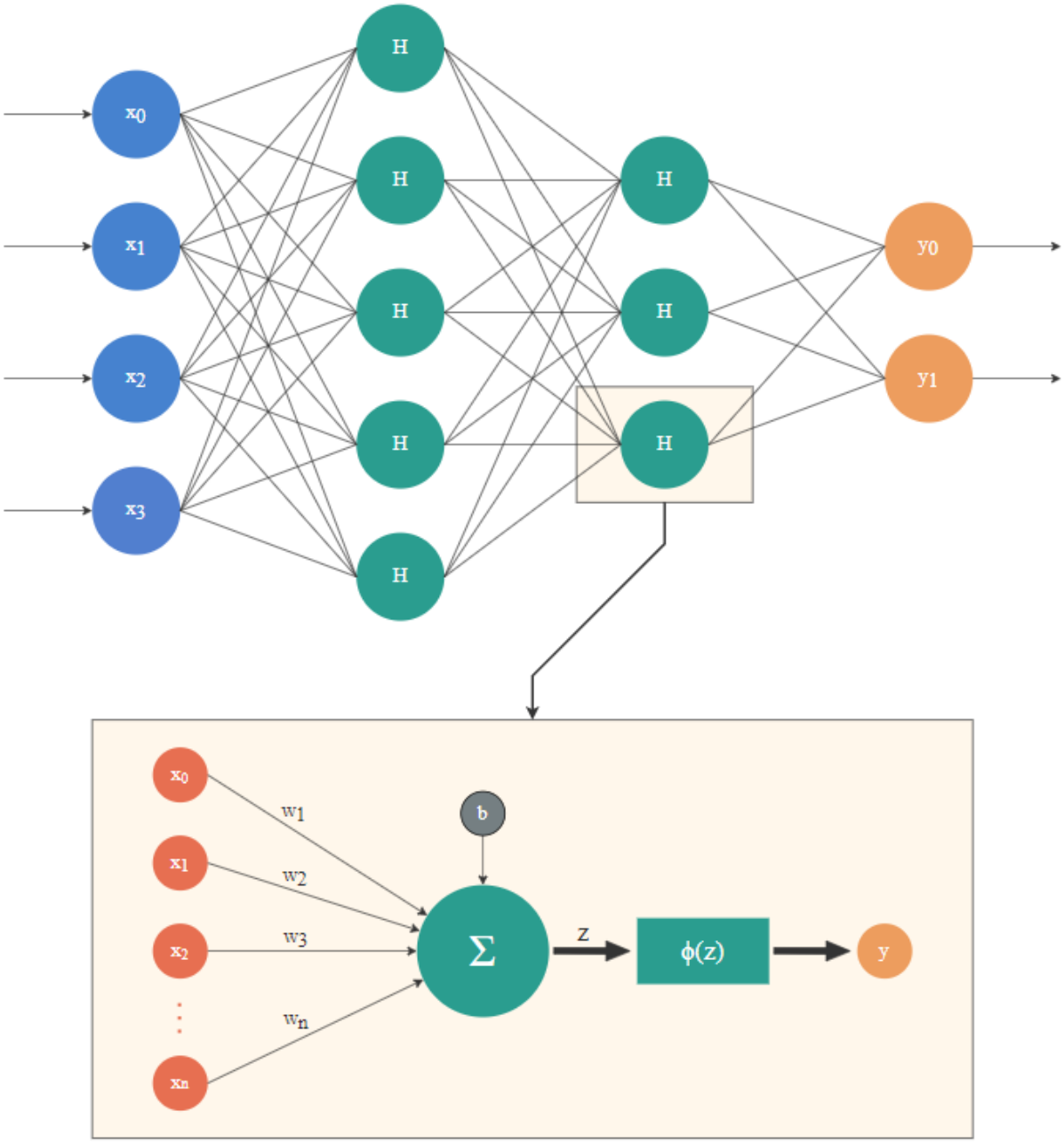


Рисунок 3.4 — Отримання активацій з нейронів штучної нейронної мережі

Зазвичай, при навчанні нейронної мережі, активації внутрішніх шарів не зберігають. Тому, для того, щоб застосовувати DeepRED, спершу потрібно ці активації отримати. Отримати активації всіх шарів нейронної мережі можна за допомогою пропускання навчальної вибірки через мережу та запам’ятовування виходів нейронів при всіх варіаціях вхідних значень.

Враховуючи, що DeepRED працює лише з MLP, активації дуже легко отримати маючи лише ваги та зміщення шарів нейронної мережі. Оскільки ваги

та зміщення напряму відповідають всім зв’язкам та нейронам мережі, за допомогою них, можна відтворити мережу, після чого, за допомогою пропускання навчальної вибірки даних, отримати необхідні активації.

3.3 Дослідження використання алгоритму для вилучення правил

В рамках даного підрозділу буде розглянуто приклад застосування алгоритму DeepRED до нейронної мережі, що вирішує задачу бінарної класифікації. Розглядати вилучення правил на основі бінарної класифікації було вирішено для більшої зручності аналізу отриманого результату. Таке спрощення дозволить отримати набагато менший граф з вилученими правилами, що, в свою чергу, полегшить ручний прохід його вузлами. Для того, щоб сформувати вхідну вибірку для бінарної класифікації, із бази даних MNIST було відібрано лише зображення, що містять цифри 0 та 1.

В якості нейронної мережі, що буде виконувати задачу класифікації, було взято MLP з двома прихованими шарами, розмірами в 100 та 30 нейронів. На всіх проміжних шарах мережі використовується функція активації ReLU, а в якості функції активації вихідного шару застосовується Softmax. Таким чином, враховуючи розмірності вхідних та вихідних векторів, отримано нейронну мережу 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).

3.3.1 Результати вилучення правил

В результаті застосування алгоритму DeepRED до описаної нейронної мережі, було отримано граф рішень з вилученими правилами, який зображено на рисунках 3.5, 3.6 та 3.7.

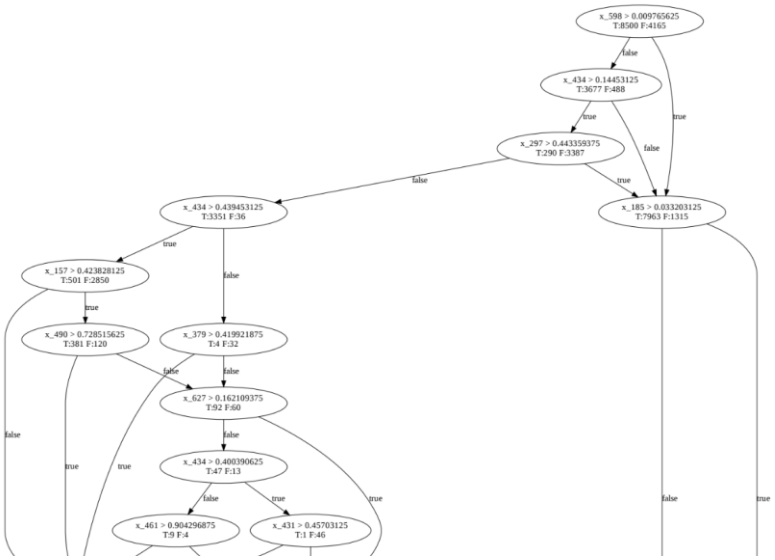


Рисунок 3.5 — DDAG з вилученими правилами NN (частина 1)

Рисунок 3.6 — DDAG з вилученими правилами NN (частина 2)

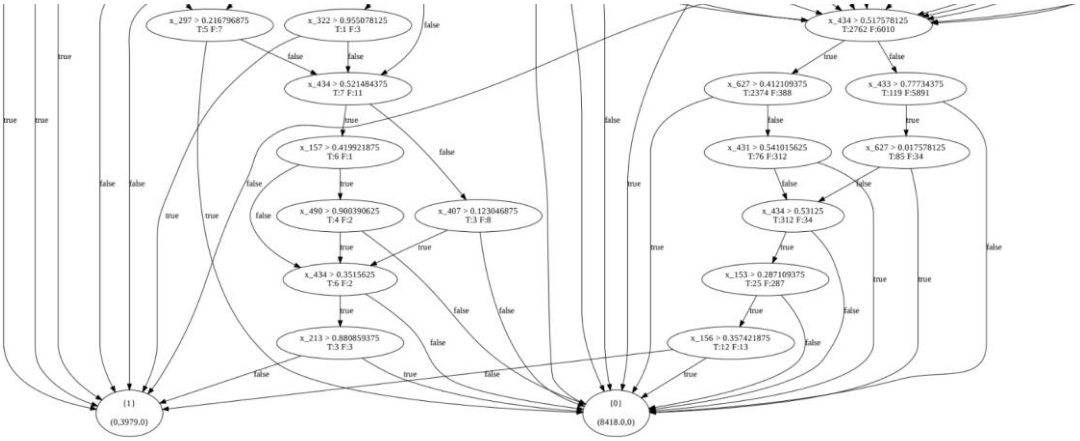


Рисунок 3.7 — DDAG з вилученими правилами NN (частина 3)

На отриманому графі, в даному випадку, листок “{0}” відповідає результату класифікації, який стверджує, що вхідне зображення містить цифру “0”, а листок “{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 і-того елемента вектора вхідних значень, що відповідає інтенсивності відповідного пікселя зображення (Рис. 3.2), а *intensity* – це певне порогове значення інтенсивності, вилучене з “логіки” нейронної мережі, з яким ведеться порівняння інтенсивності пікселя зображення.

3.3.2 Дослідження отриманого графа правил прийняття рішень

Для кращого розуміння отриманого результату, виконаємо ручний прохід вузлами отриманого графу на прикладі двох зображень з бази даних MNIST.

Зображення 1:

Спершу розглянемо випадок з зображенням, що містить цифру “1”. Для цього, з вхідної вибірки даних, випадковим чином обираємо одне із таких зображень. Це обране зображення можна побачити на рисунку 3.8.

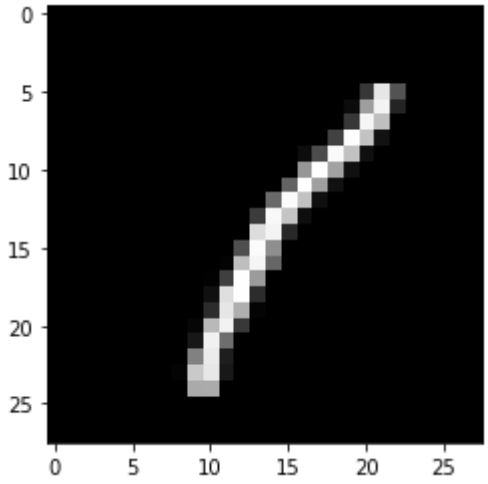


Рисунок 3.8 — Досліджуване зображення із бази даних MNIST, що містить цифру “1”

Дістаючи, крок за кроком, необхідні значення пікселів вхідного зображення, будемо проходити вузлами отриманого графа з вилученими правилами, відтворюючи процес прийняття рішення щодо класифікації зображення. Даний процес проходу вузлами продемонстровано на рисунках 3.9, 3.10 та 3.11.

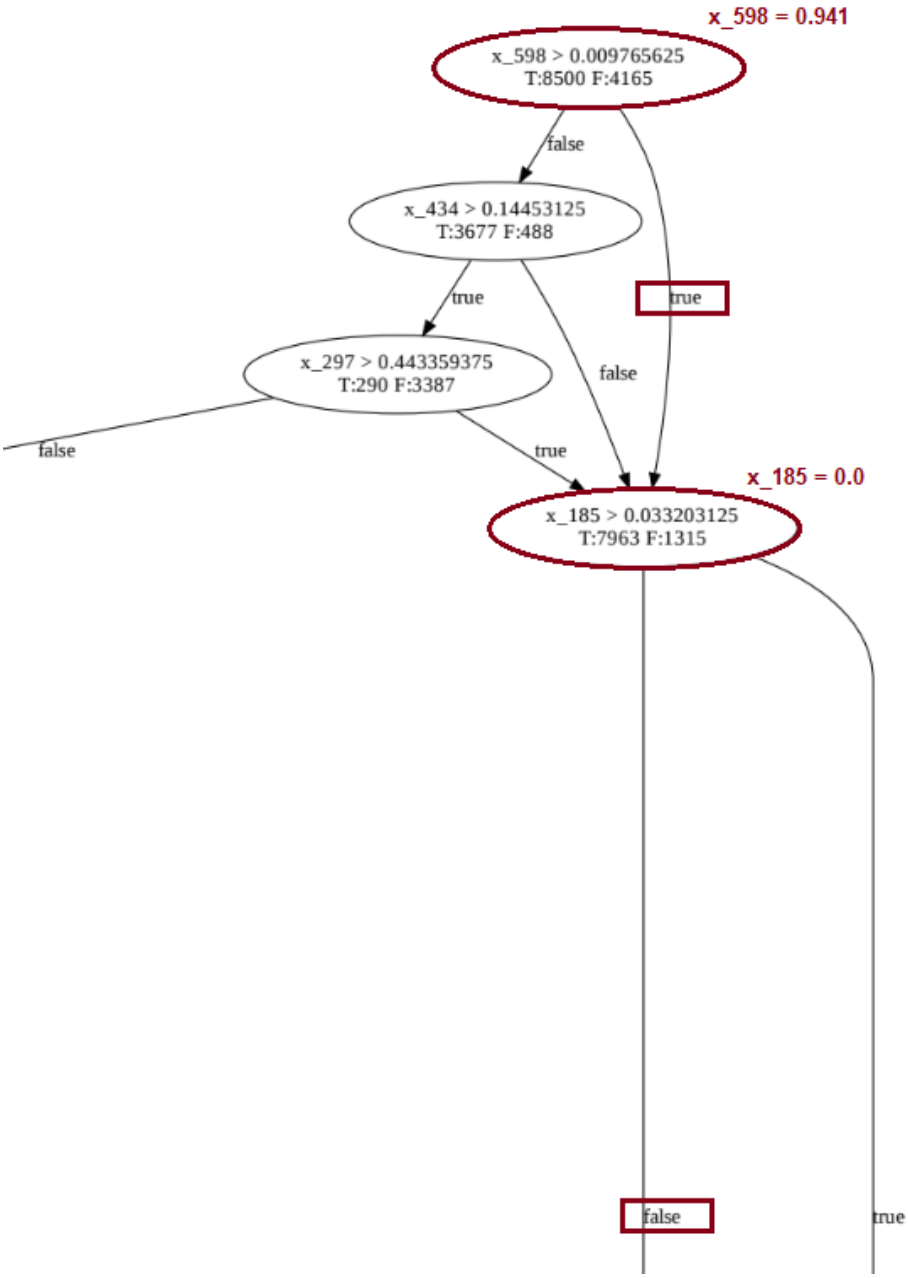


Рисунок 3.9 — Схематичне зображення проходу графом вилучених правил для класифікації зображення з цифрою “1” (частина 1)

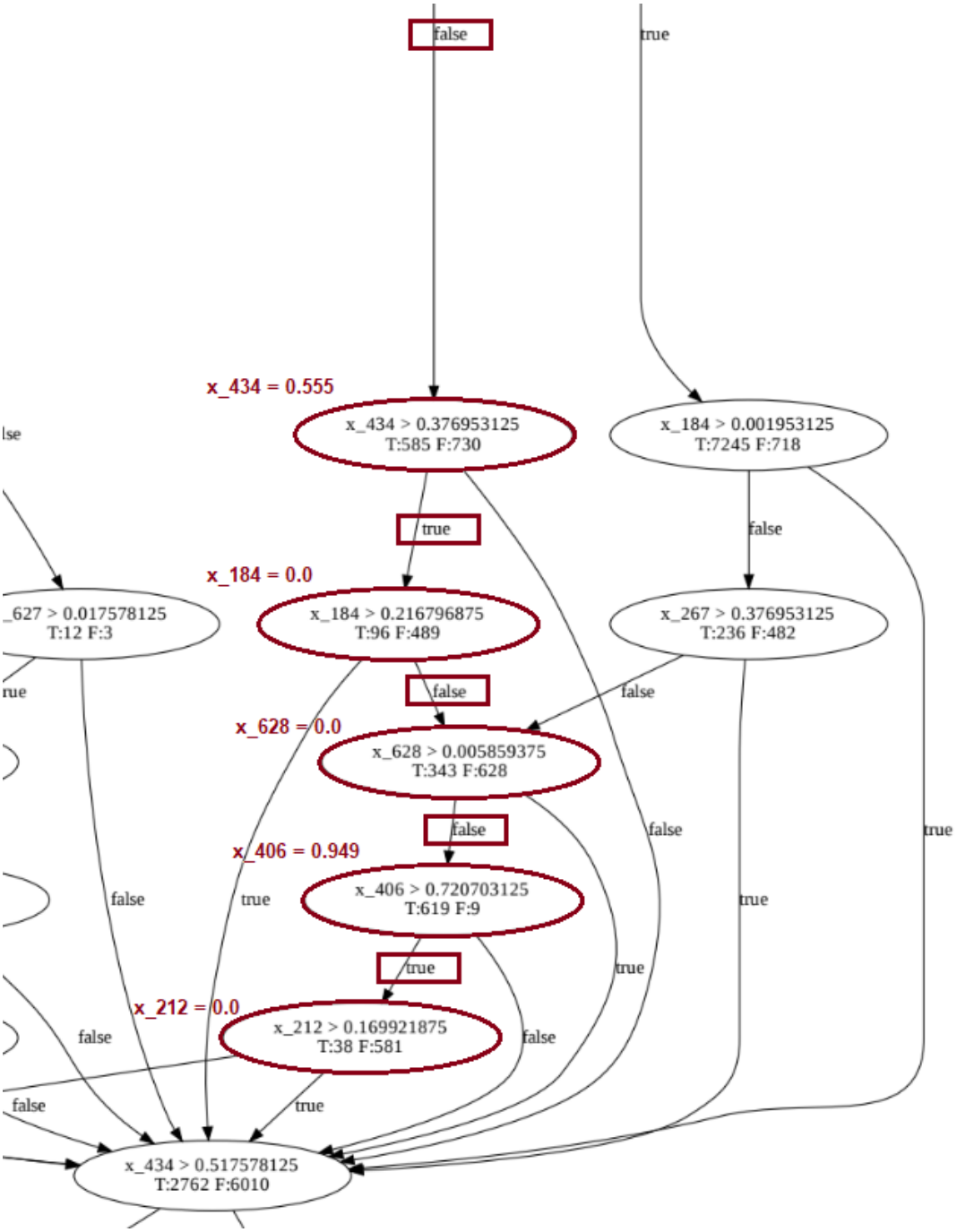


Рисунок 3.10 — Схематичне зображення проходу графом вилучених правил для класифікації зображення з цифрою “1” (частина 2)

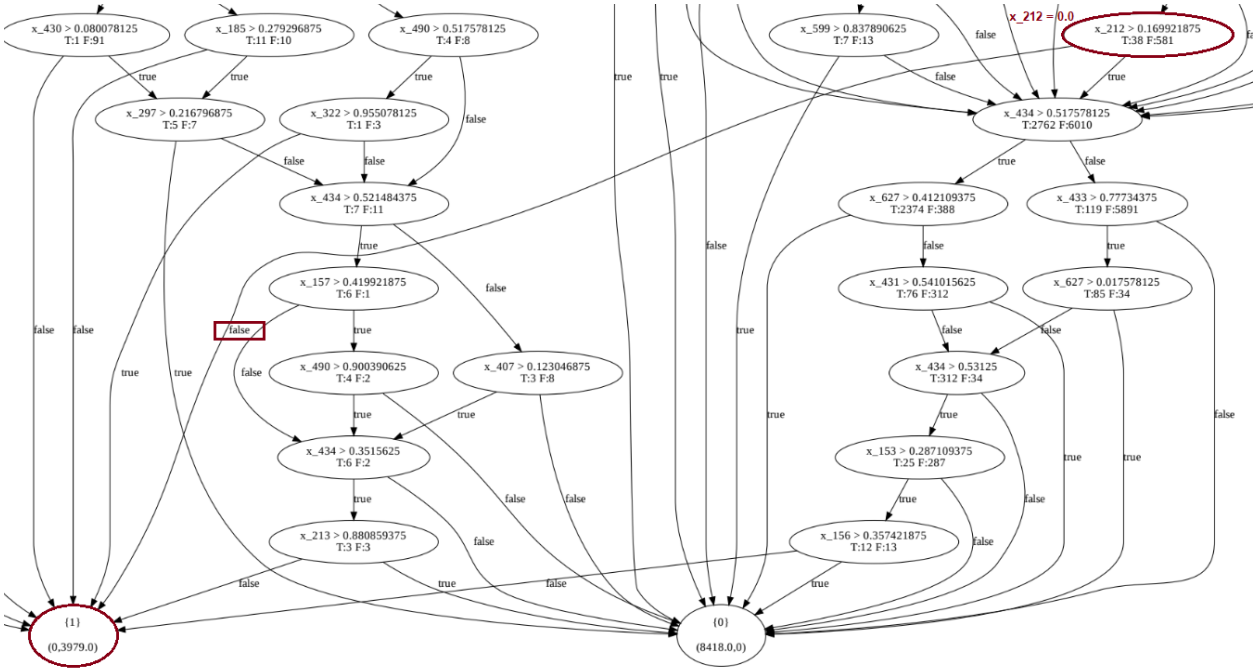


Рисунок 3.11 — Схематичне зображення проходу графом вилучених правил для класифікації зображення з цифрою “1” (частина 3)

Зображення 2:

Для другого випадку, аналогічно першому, випадковим чином обираємо зображення, що містить цифру “0”. Обране зображення показано на Рис. 3.12.

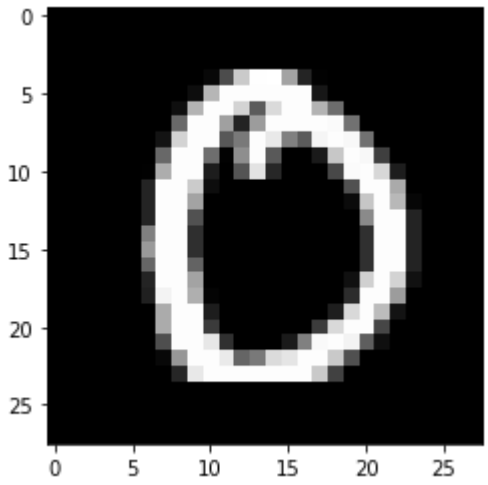


Рисунок 3.12 — Досліджуване зображення із бази даних MNIST, що містить цифру “0”

Після цього, виконаємо аналогічний прохід по тому ж графу рішень з вилученими правилами нейронної мережі, але тепер дістаючи значення пікселів нового зображення з цифрою “0”. Процес даного проходу відображено на рисунках 3.13, 3.14 та 3.15.

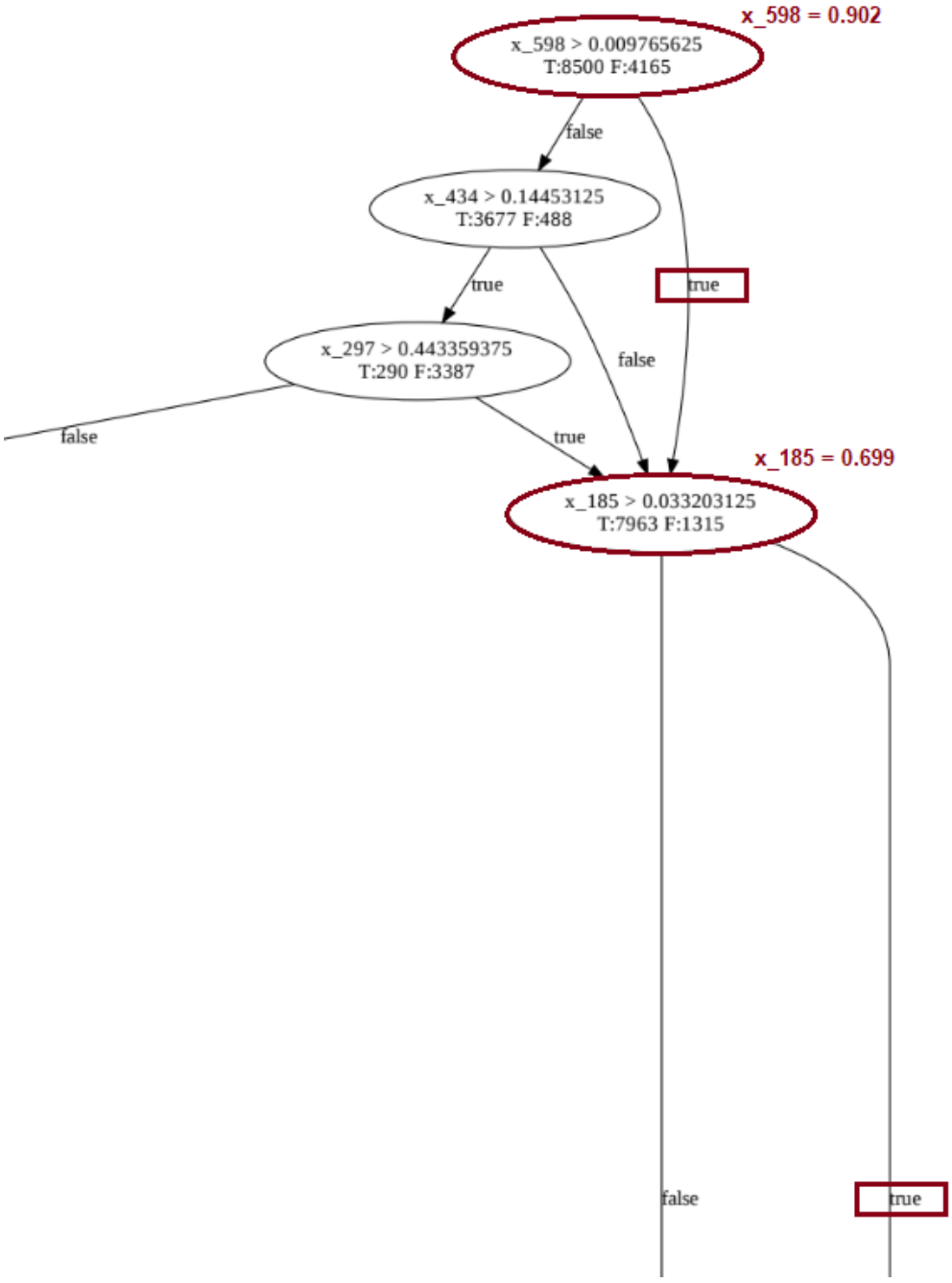


Рисунок 3.13 — Схематичне зображення проходу графом вилучених правил для класифікації зображення з цифрою “0” (частина 1)

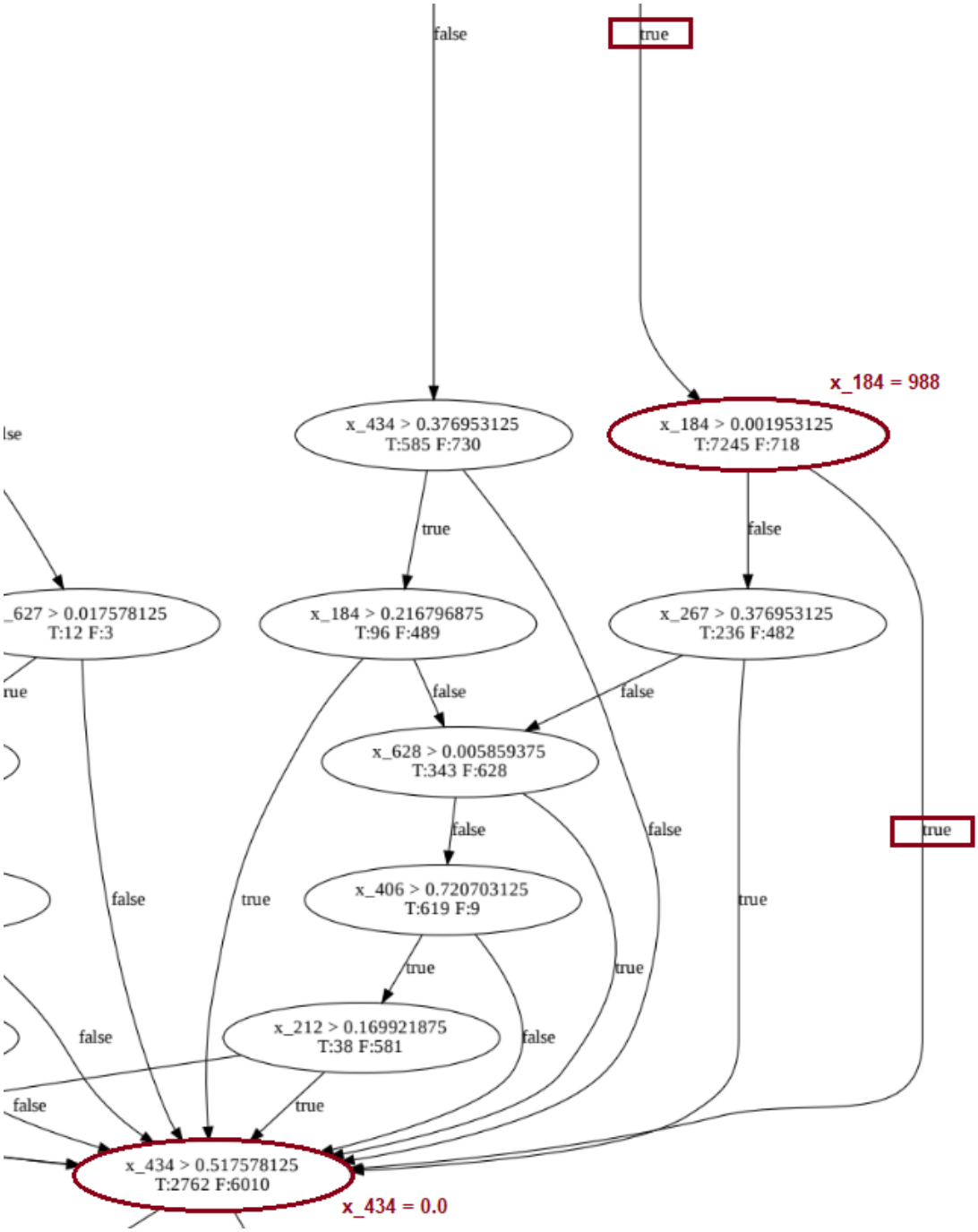


Рисунок 3.14 — Схематичне зображення проходу графом вилучених правил для класифікації зображення з цифрою “0” (частина 2)

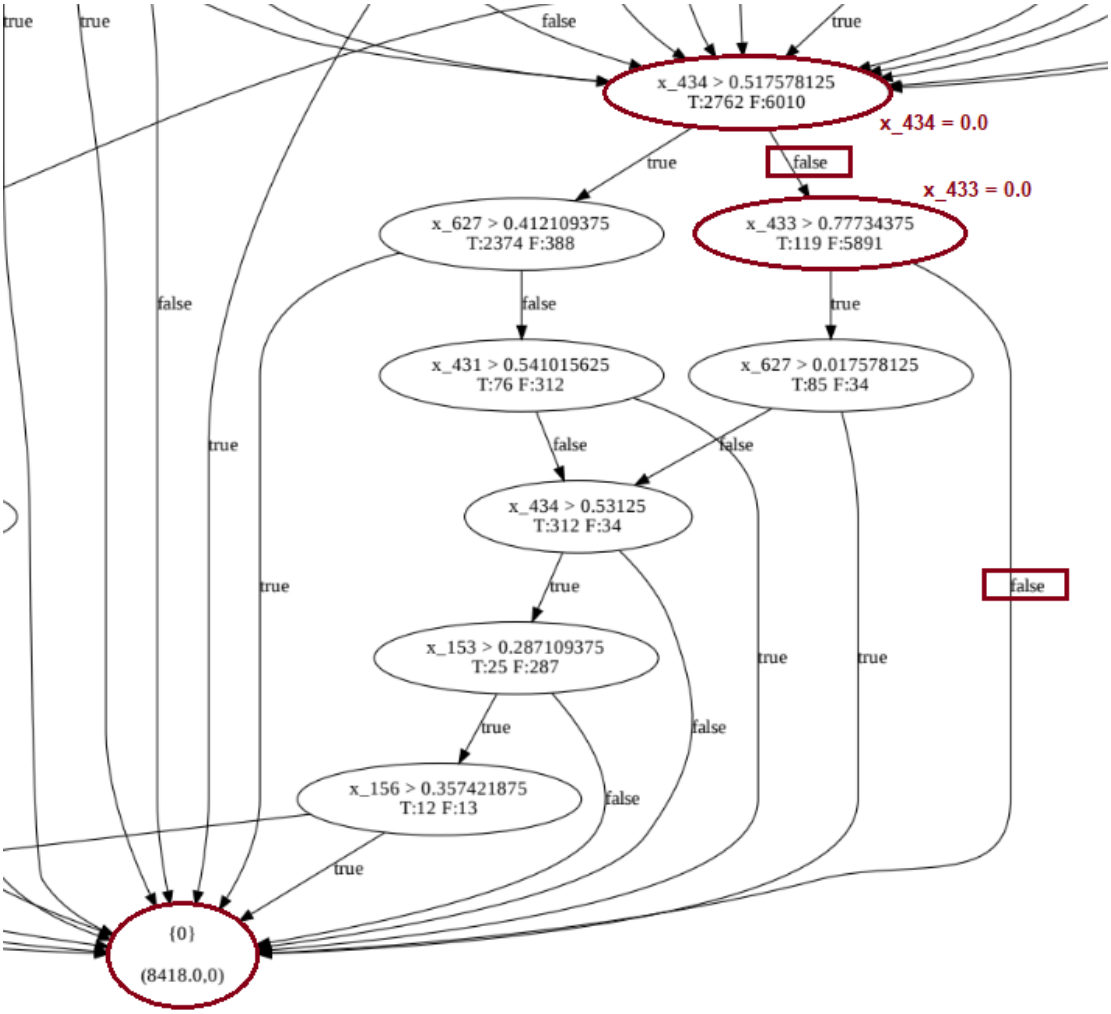


Рисунок 3.15 — Схематичне зображення проходу графом вилучених правил для класифікації зображення з цифрою “0” (частина 3)

3.4 Порівняльний аналіз серії запусків алгоритму при різних конфігураціях задачі

Проведемо порівняння результатів запуску алгоритму DeepRED на MLP різних розмірів, для випадків з бінарною та багатокласовою класифікаціями. Для того, щоб отримати потрібні навчальні вибірки, будемо відфільтровувати з бази даних MNIST набори даних так, щоб вони відповідали бажаній кількості результатів класифікації. Наприклад, для дослідження випадку з бінарною класифікацією, візьмемо із загальної вибірки лише набори значень, що відповідають зображенням з цифрами 1 та 3.

3.4.1 Вибір показників для порівняння

Перед тим, як проводити порівняння, необхідно вирішити, які властивості результатів нас цікавлять в першу чергу, та якими показниками вони будуть вимірюватись. Орієнтуючись на основні вимоги до процесу вилучення правил, які були виділені в ході аналізу існуючих проблем (2.3.3 Постановка задачі), приходимо до того, що в першу чергу нас цікавлять показники зрозумілості та точності вилучених правил.

Питання зрозумілості отриманих правил, по-суті, зводиться до питання їх складності. Так як, в даній реалізації алгоритму, вигляд правил чітко обмежений структурою графа рішень, з умовами, стандартними для DT, складність правил залежатиме лише від кількості умов та зв’язків в отриманому результаті. Таким чином, для порівняння складності вилучених правил, буде достатньо розглядати показник розміру фінального графа рішень – ***final DDAG size***.

Перевіряти точність отриманих правил – трохи складніше завдання, оскільки, через особливості самих нейронних мереж, точно знати їх внутрішні правила, щоб порівняти їх з отриманими, ми не можемо. Тому, для оцінки точності правил будуть використовуватись оцінки точності результатів класифікації, отриманих із побудованого графа рішень (DDAG). В даному випадку, будуть розглядатись такі показники точності:

- ***DDAG fidelity*** **—** показник точності результатів класифікації DDAG відносно результатів класифікації нейронної мережі на основі тестової вибірки. В якості назви цього показника застосовується термін *fidelity*, оскільки він часто використовується для позначення точності моделі симуляції відносно об’єкту, що симулюється.
- ***DDAG accuracy*** **—** показник точності результатів класифікації DDAG відносно еталонних значень із тестової вибірки (очікуваних результатів класифікації).

- ***DDAG recall*** — показник точності результатів класифікації DDAG відносно результатів класифікації нейронної мережі на основі навчальної вибірки.

Додатково, для кращого розуміння кожного із випадків, буде також відслідковуватись показник точності досліджуваної нейронної мережі на основі тестової вибірки – ***NN accuracy***. А також, будуть вестись спостереження за розмірами графа вилучених правил, перед тим, як до нього застосовано алгоритм видалення зайвих правил – ***DDAG size before pruning***. Це дозволить порівнювати обчислювальну складність розрахунків при різних конфігураціях мережі.

3.4.2. Проведення серії запусків алгоритму

Як уже відмічалося вище, порівняльні запуски алгоритму DeepRED будуть проводитись для класифікацій з різними кількостями класів. Було вирішено проводити дослідження для випадків бінарної класифікації, класифікації з 3-ма класами та класифікації з 4-ма класами. Відповідні результати запусків можна побачити у таблицях 3.1, 3.2 та 3.3. Через те, що обчислювальна складність алгоритму швидко зростає зі збільшенням можливих результатів класифікації, порівняння для випадків з більшою кількістю класів було вирішено не проводити, через досягнення ряду обмежень середовища виконання розрахунків.

Таблиця 3.1 — 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 3.2 — 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

Таблиця 3.3 — Результати запусків Deep RED у випадку класифікації з 4 класами (зображення з цифрами 0, 1, 2, 3)

Конфігурація MLP	**NN accuracy**	**DDAG fidelity**	**DDAG accuracy**	**DDAG recall**	**Final DDAG size**	**DDAG size before pruning**
MLP(20)	0.9875	0.6520	0.6568	0.6626	38	88
MLP(50)	0.9911	0.7806	0.8073	0.7978	64	68
MLP(100)	0.9923	0.7669	0.7763	0.7759	69	77
MLP(300)	0.9942	0.8648	0.8609	0.8519	65	66

MLP(100,30)	0.9932	0.6011	0.5018	0.4978	155	460
MLP(300,150)	0.9933	0.5093	0.3996	0.4000	155	570
MLP(50, 25, 9)	0.9930	0.6707	0.2345	0.2395	137	668
MLP(100,50,30)	0.9942	0.7775	0.2735	0.2725	77	924

Окрім того, що складність алгоритму сильно зростає зі збільшенням кількості класів, в результаті запусків з різними конфігураціями MLP, було також помічено, що розміри побудованих DDAG досить сильно зростають при збільшенні кількості шарів NN, або збільшенні їх розмірів, що також призводить до росту обчислювальної складності. Звісно, варто відмітити, що та реалізація алгоритму, на основі якої проводились всі експерименти, не була особливо зосереджена на оптимізації продуктивності алгоритму, і потенціал для покращення цього показника ще точно є. Але, в той же час, проблеми зі складністю даного алгоритму відмічаються також і в роботах інших авторів.

Із отриманих результатів, легко виділити закономірність, що зі збільшенням кількості класів класифікації відбувається відчутне падіння точності вилучених правил. Як бачимо, в рамках бінарної класифікації, DeepRED показує себе набагато краще ніж в інших випадках. При цьому, помітно, що при складніших конфігураціях нейронної мережі, було отримано гіршу точність результатів. Чим більша кількість нейронів у кожному шарі, та кількість самих шарів у NN, тим складніше отримати хороший результат вилучення правил. Таку закономірність можна пояснити тим, що зі збільшенням вхідних даних, піддерева, побудовані в процесі проходів алгоритму, починають накопичувати загальну помилку. Через це, для отримання адекватних результатів, бажано, щоб розміри шарів NN були невеликими. Наприклад, в рамках досліджуваної задачі, було помічено, що при конфігураціях, в яких розміри шарів на порядок більші за 100 нейронів, вилучити правила, з відносно хорошою точністю, досить складно.

3.5 Висновки

В рамках даного розділу, було досліджено застосування алгоритму DeepRED, для вилучення правил з нейронних мереж, що виконують класифікацію зображень бази даних MNIST. В ході проведених досліджень, було проаналізовано результати із вилученими правилами, отриманими у вигляді графа рішень, а також, було проведено невеликий порівняльний аналіз серії різних запусків DeepRED на базі MLP різних конфігурацій, при класифікації з двома, трьома та чотирма класами.

Не зважаючи на те, що в рамках виконаних дослідів було можливим проведення вдалих вилучень правил, експерименти показали також і ряд обмежень методу DeepRED. Окрім того, що алгоритм часто має проблеми з обчислювальною складністю, він не надто ефективний при будь-якій іншій класифікації, окрім бінарної. При цьому, особливо гостро це питання постає при роботі зі складнішими конфігураціями MLP (з більшою кількістю шарів, або з шарами великої розмірності). Іншою можливою проблемою є те, що алгоритм сильно залежить від умов, в яких він запускається. Наприклад, одна із таких, найбільш показових, умов – це вдалість початкової ініціалізації дерева рішень. По-суті, увесь процес проходу алгоритму спирається на те, як буде ініціалізовано дерево на першому кроці. А оскільки на одному і тому ж наборі даних, дерево рішень може бути побудоване по-різному (наприклад, при побудові можна обирати різні обмеження щодо максимальної глибини дерева), то і фінальні результати можуть досить сильно варіюватися.

4 РОЗРОБКА СТАРТАП ПРОЕКТУ

4.1 Опис ідеї проекту

Ідея проекту полягає в створенні бібліотеки для вилучення правил прийняття рішень з нейронних мереж. Така бібліотека допоможе проводити аналіз розробленої мережі за рахунок отримання зрозумілого набору правил. Потенційно це дозволить розширити можливості використання NN у сферах, де критичними є передбачуваність та можливість покриття усіх крайніх сценаріїв моделей, що використовуються.

Таблиця 4.1 — Суть ідеї стартапу

Зміст ідеї	**Напрямки застосування**	**Вигоди для користувача**
Застосування алгоритму DeepRED для вилучення правил прийняття рішень з нейронних мереж	Використання при дослідженні готових моделей, що уже використовуються, чи мають використовуватись у критичних сферах	Можливість знайти потенційні проблеми та непередбачувані сценарії в поведінці мережі за певних крайніх умов.
		Здатність отримання повного набору результатів моделі за всіх можливих комбінацій вхідних значень.
	Використання в якості додаткового інструменту для контролю якості навчання моделей, при їх розробці для застосування у критичних сферах	Додаткові можливості контролю якості навчання мережі.
		Можливість розробки більш передбачуваних моделей.

Аналіз потенційних техніко-економічних переваг ідеї (відмінність від рішень-аналогів) проведено у таблиці 4.2. Порівняння буде проводитись із двома бібліотеками, що забезпечують вилучення правил із нейронних мереж за допомогою педагогічних алгоритмів ANN-DT та HypInv (ще одного алгоритму, що не розглядався в рамках підрозділу 2.2).

Таблиця 4.2 — Суть ідеї стартапу

Техніко-економічні характеристики ідеї	**Проект, що розробляється**	**Проекти-конкуренти**		**W**	**N**	**S**
		ANN-DT	**HypInv**			
Висока точність результатів	+	-	-			+
Стабільність результатів	-	-	+		+	
Високий рівень зрозумілості правил	+	+	-			+
Можливість роботи з будь-якою NN	-	+	+	+		
Порівняно низькі вимоги щодо обчислювальної потужності	-	+	+	+		
Широкі можливості щодо подальшого розвитку алгоритму	+	-	-			+

Отриманий перелік порівняльних характеристик дозволяє краще зрозуміти можливі вектори подальшого розвитку проекту, а також дещо уточнює нішу ринку, де буде отримано потенційно кращі результати, порівняно з конкурентами. В даному випадку видно, що жодне із представлених рішень не має повного домінування над іншими. З одного боку, бібліотеки, що реалізують ANN-DT та HypInv є більш універсальними рішеннями, які можуть використовуватись до будь-яких NN і, при цьому, не мають високих вимог щодо обчислювальної потужності. З іншого – реалізація DeepRED має поєднання високої точності та зрозумілості результатів, через що, у тих випадках, де можливе її застосування, вона буде мати сильну конкурентну перевагу. При

цьому, важливо також відмітити, що сам алгоритм DeepRED має досить багато місця для покращень, за рахунок яких, потенційно, можна вирішити деякі із його основних мінусів.

4.2 Технологічний аудит ідеї

Запуск проекту опирається на реалізацію відповідної системи. Через це, важливим етапом є проведення аудиту технологій, які необхідні для виконання цієї реалізації. Результати даного аудиту, з розглядом ключових технологій, на які опиратиметься розробка системи, наведено у таблиці 4.3.

Таблиця 4.3 — Технологічна здійсненність проекту

Головна ідея	**Технології**	**Наявність**	**Доступність**
Використання алгоритму DeepRED для вилучення правил з нейронних мереж	Мова програмування Python	Так	Безкоштовна
	Бібліотека для роботи з нейронними мережами (наприклад, TensorFlow)	Так	Безкоштовна
	Бібліотека для роботи з деревами рішень (наприклад, Scikit-learn)	Так	Безкоштовна
	Середовище з високими обчислювальними ресурсами	Так	Від $300 на місяць

4.3 Аналіз ринкових можливостей

Для побудови правильної стратегії виходу на ринок, потрібно розуміти особливості цього ринку та розуміти потреби потенційних користувачів. У

таблицях 4.4 та 4.5 зібрано основні характеристики ринку та потенційних клієнтів продукту.

Таблиця 4.4 — Характеристика потенційного ринку

Показники стану ринку	**Характеристика**
Головні гравці на ринку (монополісти / великі компанії, що зайняли нішу)	Відсутні
Малі гравці на ринку (потенційні конкуренти / непрямі конкуренти, що можуть забезпечити альтернативні рішення)	Майже немає, < 5
Динаміка ринку (якісна оцінка)	Повільний ріст, спричинений відсутністю рішень, які можуть забезпечити покриття потреб в повній мірі
Наявність обмежень для входу	Складність забезпечення рішення, що зможе якісно покривати потреби потенційних користувачів
Специфічні вимоги до стандартизації та специфікації	Відсутні

Таблиця 4.5 — Характеристика потенційних клієнтів

Потреба	**Потенційні клієнти**	**Відмінності поведінки різних цільових груп потенційних клієнтів**	**Вимоги споживачів до товару**
Можливість контролю передбачуваності поведінки моделі. Можливість отримання повного набору можливих результатів, що може видавати використовувана система	Фінансові та медичні установи, авіакомпанії, сфера енергозабезпечення та представники інших сфер, у яких потенційні помилки систем проводять до високих ризиків.	Точність та зрозумілість вилучених правил є критичними, при цьому, допускається нехтування будь-якими іншими показниками, заради покращення цих.	Максимальна зрозумілість вилучених правил. Повна відповідність отриманих результатів та поведінки досліджуваної мережі.
Додаткові можливості контролю якості навчання нейронних мереж. Пришвидшення та покращення процесу розробки нейронних мереж під конкретні задачі.	Компанії-розробники програмного забезпечення на основі нейронних мереж.	Точність та відповідність поведінці моделі є найбільш важливими, при цьому, зрозумілість правил може бути не на найвищому рівні, якщо це дозволить отримувати більш широкий набір додаткових показників.	Достатній рівень відповідності отриманих результатів та поведінки досліджуваної мережі. Широкий спектр додаткових характеристик та показників, які можуть бути використаними для контролю процесу навчання мережі.

Важливим етапом, перед виходом на ринок, є проведення аналізу можливих загроз. Результати цього аналізу показані в таблиці 4.6.

Таблиця 4.6 — Аналіз загроз

Фактор	**Зміст загрози**	**Профілактичні заходи / можлива реакція**
Поява конкурентів	Необхідність поділу існуючого ринку з компаніями-конкурентами	Постійна робота над покращенням алгоритму
		Проведення досліджень та експериментів для пошуку нових рішень у сфері вилучення правил
Проблеми зі стабільністю отримання якісних результатів	Втрата лояльності клієнтів	Постійне тестування бібліотеки за різних умов
		Вдосконалення універсальності бібліотеки
		Регулярна підтримка та консультації клієнтів

Для кращого розуміння шляху розвитку проекту, буде розглянуто фактори можливостей (табл. 4.7)

Таблиця 4.7 — Фактори можливостей

Фактор	**Зміст можливостей**	**Потенційний вплив на компанію**
Залучення інвестицій	Потенційне розширення можливостей компанії, за рахунок отримання фінансування від інвесторів.	Масштабування команди; залучення зовнішніх експертів; покращення основних технологій.
Розширення цільових груп клієнтів	Отримання ширшої бази потенційних клієнтів та можливість уточнення/зміщення фокусу компанії.	Збільшення прибутку компанії; краще розуміння потреб на існуючому ринку; отримання додаткового досвіду та експертизи у галузі вилучення правил.

Для успішного виходу на ринок, дуже важливо розуміти його стан та всіх учасників, з якими потрібно буде конкурувати. Тому в таблицях 4.8 і 4.9 буде розібрано ці питання.

Таблиця 4.8 — Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	**Прояви характеристики**	**Вплив на компанію**
Майже повна відсутність конкуренції	Через складність якісного виконання задачі, повноцінних конкурентів на ринку немає, присутні лише деякі рішення, що злегка покривають представлені проблеми і не націлені на вирішення даної проблеми.	Відсутність аналогічних рішень сильно ускладнює можливості чіткого розуміння потреб потенційних користувачів, але при цьому дає досить широкі можливості для виходу на ринок на вигідних для компанії умовах.
Складність задачі; проблематичність покриття потреб користувачів	Популярність нейронних мереж дуже сильно зросла за останній час і той рівень розвитку, якого вони досягли на даний момент, має дуже великий потенціал для вирішення складних та важливих проблем. Але, в той же час, можливості щодо розширення сфер використання NN певною мірою упирається у непрозорість їх роботи. При цьому, не зважаючи на таку активність, все ще відсутні якісні рішення щодо можливості вирішення проблеми зі зрозумілістю нейронних мереж.	Складність задачі, яку має забезпечувати даний проект може спричинити певні проблеми з пошуком перших клієнтів, оскільки майже гарантовано буде потребувати проведення певної адаптації та корегування під потреби реальних задач. Проте, за умови успішного впровадження рішення ряду клієнтів, подальше розширення буде набагато простішим, оскільки проблема зрозумілості нейронних мереж добре відома та досить широка.

Таблиця 4.9 — Аналіз конкуренції в галузі за М. Портером

	Прямі конкуренти	**Потенційні конкуренти**	**Постачальники**	**Клієнти**	**Товари-замінники**
Складові аналізу	-	Деякі безкоштовні бібліотеки для вилучення правил за допомогою педагогічних підходів	-	Фінансові та медичні установи, авіакомпанії, сфера енергозабезпечення, розробники нейронних мереж, …	Деякі інші інструменти для роботи з нейронними мережами, що не опираються на вилучення правил
Висновки	На сьогоднішній день, прямих конкурентів на ринку не існує	Існують певні аналоги, що здатні вилучати правила з NN, але рівень їх точності не дозволяє забезпечувати вирішення існуючих потреб	Відсутні	Існує досить поширена проблема в продукції такого роду	Є певні рішення, що можуть частково покривати деякі з розглянутих проблем, але вони слабко підходять для нормального вирішення даної задачі

Фактори, що забезпечують продукту конкурентні переваги порівняно з іншими рішеннями на ринку – фактори конкурентоспроможності, а також, порівняння його сильних та слабких сторін, буде наведено в таблицях 4.10 та 4.11, відповідно.

Таблиця 4.10 — Фактори конкурентоспроможності

Фактор конкурентоспроможності	**Обґрунтування**
Висока точність результатів	Від того, на скільки точними будуть отримані результати залежить доцільність виконання вилучення правил взагалі.
Зрозумілість вилучених правил	Без отримання зрозумілих правил, що описують поведінку нейронної мережі буде проблемно (або неможливо) проводити аналіз результатів.
Універсальність рішення	Від того, на скільки універсальним буде представлене рішення, на прямо буде залежати кількість потенційних задач, у яких воно зможе використовуватися.

Таблиця 4.11 — Порівняльний аналіз сильних та слабких сторін

Фактор конкурентоспроможності	**Бали [1, 20]**	**Оцінка конкурентів у порівнянні з рішенням, що розробляється**						
		-3	**-2**	**-1**	**0**	**1**	**2**	**3**
Висока точність результатів	15			+				
Зрозумілість вилучених правил	18		+					
Універсальність рішення	10						+	

Далі буде проведено SWOT-аналіз описаного рішення (4.12).

Таблиця 4.12 — SWOT-аналіз

Сильні сторони	**Слабкі сторони**
Зрозумілість вилучених правил	Залежність від розмірів вхідних даних
Точність результатів	Необхідність в наявності апаратного забезпечення з достатніми обчислювальними потужностями
Великий потенціал покращення алгоритму	Необхідність проведення багатьох досліджень та експериментів
Можливості	**Загрози**
Перспектива зайняття ключового місця на вільному ринку з потенційно великою потребою в подібних рішеннях	Невдача у спробі забезпечити ринок рішенням з достатніми можливостями, щоб покрити головні вимоги потенційних користувачів

Також, розглянемо стратегії виходу на ринок для описаного стартап-проекту (табл. 4.13).

Таблиця 4.13 — Альтернативи ринкового впровадження

Альтернатива ринкової поведінки	**Ймовірність отримання ресурсів**	**Терміни реалізації**
Розробка нових способів вилучення правил з нейронних мереж	70%	24 місяці
Вдосконалення сервісу й усунення залежності, розширення функціоналу	60%	12 місяців

Інформація, описана в рамках таблиць 5.4 – 5.13 показує, що описаний стартап-проект має хороші перспективи виходу на ринок. При тому, що описана ідея має певні слабкості, прямих аналогів на ринку, що можуть забезпечити сильну конкуренцію рішенню, на даний момент немає.

4.4 Розробка ринкової стратегії

В рамках розробки ринкової стратегії, важливо чітко зрозуміти, які саме групи користувачів будуть розглядатись в якості цільової аудиторії для використання продукту. Розгляд цих груп наведено в таблиці 4.14.

Таблиця 4.14 — Вибір цільових груп потенційних користувачів

Опис профілю цільової групи потенційних клієнтів	**Готовність споживачів сприйняти продукт**	**Орієнтовний попит в межах цільової групи (сегменту)**	**Інтенсивність конкуренції в сегменті**	**Простота входу у сегмент**
Розробники нейронних мереж	Зацікавлені у продукті	Більшість представників зацікавлені у будь-яких вдосконаленнях та покращеннях, навіть за умов не ідеальної роботи рішення	Низька	Потребує пошуку лояльних, зацікавлених клієнтів, які готові розглядати користування рішенням на певних взаємовигідних умовах
Фінансові та медичні установи, авіакомпанії, сфера енергозабезпечення	Зацікавлені у продукті лише за умов високої точності в поєднанні з хорошою зрозумілістю правил	Частина представників сильно зацікавлена в подібних рішеннях вже, інша частина може бути зацікавлена лише після того, як з’являться успішні сценарії використання	Відсутня	Досить високі потреби, може потребувати наявності успішних впроваджень серед інших компаній

Також, розробка ринкової стратегії включає в себе такі етапи, як визначення базової стратегії розвитку, визначення базової стратегії конкурентної поведінки

та формування стратегії позиціонування. Ці етапи будуть описані далі, в таблицях 4.15, 4.16 та 4.17, відповідно.

Таблиця 4.15 — Визначення базової стратегії розвитку

Обрана альтернатива розвитку проекту	**Стратегія охоплення ринку**	**Ключові конкурентні позиції відповідно до обраної альтернативи**	**Базова стратегія розвитку**
Розвиток шляхом демонстрації можливостей рішення, пошуку партнерських відносин та формування рекомендацій існуючими клієнтами	Впливати на обрані цільові групи в порядку значимості та перспектив	Виступи на конференціях, розповсюдження дослідницько-наукових статей, формування спільноти серед зацікавлених в базовій ідеї.	Постійне проведення нових досліджень в галузі. Розробка нових, актуальних рішень в галузі вилучення правил.

Таблиця 4.16 — Визначення базової стратегії конкурентної поведінки

Чи є проект «першопрохідцем» на ринку?	**Чи може компанія забирати існуючих споживачів?**	**Компанія копіюватиме характеристики товару конкурента?**	**Стратегія конкурентної поведінки**
Так	Оскільки прямих аналогів на ринку немає, споживачів такого продукту також немає. Можливе знаходження користувачів, що застосовують певні суміжні рішення.	Ні	Постійно розвивати рішення, щоб мати сильний відрив прогресу у випадку появи рішень-конкурентів

Таблиця 4.17 — Визначення стратегії позиціонування

Вимоги до товару цільової аудиторії	**Базова стратегія розвитку**	**Ключові конкурентні позиції власного стартап-проекту**	**Вибір асоціацій, які мають сформувати комплексну позицію власного проекту**
Точність та зрозумілість правил	Застосування алгоритму DeepRED для вилучення правил прийняття рішень з нейронних мереж	Швидке вдосконалення алгоритму; виправлення недоліків, які будуть знайдені в ході використання	Точність та зрозумілість отримуваних правил, універсальність бібліотеки

4.5 Розробка маркетингової програми

Маркетингова складова є дуже важливою у формуванні привабливості продукту для потенційних користувачів. Основні переваги стартап-проекту, що розраховані на формування такої привабливості описано в таблиці 4.18.

Таблиця 4.18 — Визначення ключових переваг концепції потенційного товару

Потреба	**Вигода, що пропонується**	**Ключові преваги перед конкурентами (існуючі, або майбутні)**
Точність вилучених правил	Використання декомпозиційного алгоритму DeepRED	За рахунок використання декомпозиційного алгоритму, рішення буде забезпечувати максимально можливе наближення поведінки до моделі, що досліджується
Зрозумілість вилучених правил	Отримання правил у вигляді компактного графа рішень	За рахунок використання структури графа, користувач отримає зручну форму представлення правил, що легко піддається аналізу та може бути побудована у досить лаконічній формі
Стабільність отримання якісного результату	Застосування додаткових покращень до ініціалізації дерев рішень	Вдосконалення та покращення, внесені в етапи побудови дерев рішень дозволить мінімізувати проблеми зі здатністю формувати стабільні результати.

Опис трирівневої моделі товару, для рішення, що розглядається н адається в таблиці 4.19.

Таблиця 4.19 — Опис трьох рівнів моделі товару

Рівні товару	**Сутність та складові**	
Товар за задумом	Бібліотека алгоритму DeepRED для вилучення правил з нейронних мереж	
Товар у реальному виконанні	Властивості/характеристики	Розмір
	Модуль роботи з правилами у вигляді дерев та графів рішень	43МБ
	Модуль для роботи з NN та вилучення правил із неї	15МБ
	Якість: інструментарій для отримання звітності щодо якості вилучення правил, можливості отримання інформації про деталі проходу алгоритму, логіка для внутрішнього тестування.	
Товар із підкріпленням	До продажу: готова бібліотека зі зручним API та якісною документацією	
	Після продажу: регулярна підтримка та консультація користувачів; покращення рішення; регулярні оновлення	

Таблиця 4.20 надає картину встановлення цінової політики, що буде використовуватись.

Таблиця 4.20 — Визначення меж встановлення ціни

Рівень цін на товари-замінники	**Рівень цін на товари-аналоги**	**Рівень доходів цільової групи споживачів**	**Верхня та нижня межі становлення ціни на товар/послугу**
Відсутні на ринку	Від $400 за рік користування	Вище середнього, частіше високий	Від $500 за рік користування без підтримки, в залежності від масштабів. Від $1500 за рік користування з підтримкою, в залежності від масштабів та повноти підтримки

Після визначення цінової політики продукту, варто визначити систему збуту продукту, яка забезпечить якісне формування попиту (табл. 4.21).

Таблиця 4.21 — Формування системи збуту

Специфіка закупівельної поведінки цільових клієнтів	**Функції збуту, які має виконувати постачальник товару**	**Глибина каналу збуту**	**Оптимальна система збуту**
Покупка ліцензії на продукт через відділ продажів, або партнерів	Надання та підтримка програмного забезпечення, надання консультаційної підтримки	Канал збуту однорівневий	Право власності залишається у розробника ПЗ

Фінальною стадією формування маркетингової стратегії, для рішення, що розробляється, є визначення концепції маркетингових комунікацій. Результати цього етапу відображені у таблиці 4.22.

Таблиця 4.22 — Концепція маркетингових комунікацій

Специфіка поведінки цільових клієнтів	**Канали комунікацій, якими користуються цільові клієнти**	**Ключові позиції, обрані для позиціонування**	**Завдання рекламного повідомлення**	**Концепція рекламного звернення**
Використання демо-версії протягом місяця перед придбанням ліцензії	Презентації, зустрічі, конференції, тощо	Рішення, що дозволяє вилучати максимально точні та зрозумілі правила	Показати переваги використання даного рішення при розробці нейронних мереж, або задля їх перевірки	На інженерних і професійних сайтах

4.6 Висновки

В рамках даного розділу було досліджено основні переваги та недоліки проекту, описано стан конкуренції на ринку та проведено аналіз цільової сфери майбутнього продукту. Для вивчення напрямків подальшого розвитку, було проведено наступні етапи:

- Проаналізовано ідеї проекту та його ринкових можливостей. Спроектовано загальну ідею на потенційний ринок. проведено порівняння проекту з основними конкурентами.
- Проведено аналіз технологій, обов’язкових для реалізації ідеї проекту.
- Проаналізовано ринкові сценарії конкурентів, та основні ринкові можливості, досліджено конкурентоспроможність проекту та розроблено план і ринкову стратегію.
- Розроблено маркетингову програму для просування товару на ринку, описано канали продажів, специфіку клієнтів, цінову політику, а також, побудовано трирівневу модель товару.

Відповідно до результатів, що були отримані в рамках даного розділу, запланована ідея стартап-проекту має хороші перспективи щодо можливості існування на ринку.

ВИСНОВКИ

В процесі роботи з алгоритмом DeepRED було помічено, 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 розробки 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, дійсно має право на існування, і виглядає досить перспективно. Ті правила, які було отримано в якості результатів, 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 в інших розділах роботи, такі алгоритми постійно балансують в рамках компромісу точності та зрозумілості результатів.

Підсумовуючи всю інформацію, що була отримана в ході виконання даної роботи, 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 більш-менш 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, які можуть вирішувати велику кількість складних задач є мережі, що мають набагато складнішу архітектуру за MLP, наприклад, згорткові, або рекурентні нейронні мережі. 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 основано даний алгоритм, виглядає дуже перспективно, і, схоже на те, що можливостей щодо її покращень ще досить багато.

ПЕРЕЛІК ПОСИЛАНЬ

1. Eduardo R. Hruschka, Nelson F. F. Ebecken. Rule Extraction from Neural Networks in Data Mining Applications. *Transactions on Information and Communications Technologies*, 1998. Режим доступу: https://www.witpress.com/Secure/elibrary/papers/DATA98/DATA98021FU.pdf
2. Frank Emmert-Streib, Zhen Yang, Han Feng, Shailesh Tripathi, Matthias Dehmer. An Introductory Review of Deep Learning for Prediction Models With Big Data. *Front. Artif. Intell.*, 2020. Режим доступу: https://www.frontiersin.org/articles/10.3389/frai.2020.00004/full
3. Geoffrey G. Towell, Jude W. Shavlik. Extracting refined rules from knowledge-based neural networks. *Machine learning*, 1993. Режим доступу: https://link.springer.com/article/10.1007/BF00993103
4. Gregor P. J. Schmitz, Chris Aldrich, Francois S. Gouws. ANN-DT: an algorithm for extraction of decision trees from artificial neural networks. *IEEE Transactions on Neural Networks*, 1999. Режим доступу: https://ieeexplore.ieee.org/document/809084
5. Guido Bologna, Yoichi Hayashi. A Comparison Study on Rule Extraction from Neural Network Ensembles, Boosted Shallow Trees, and SVMs. Applied Computational Intelligence and Soft Computing, 2018. Режим доступу: https://downloads.hindawi.com/journals/acisc/2018/4084850.pdf
6. Hiroshi Tsukimoto. Extracting rules from trained neural networks. *IEEE Transactions on Neural networks*, 2000. Режим доступу: https://ieeexplore.ieee.org/document/839008
7. Ismail Taha and Joydeep Ghosh. Three techniques for extracting rules from feedforward networks. *Intelligent Engineering Systems Through Artificial Neural Networks*, 1996. Режим доступу: https://ieeexplore.ieee.org/document/774103

8. J.M. Benitez, J.L. Castro, I. Requena. Are artificial neural networks black boxes? *IEEE Transactions on Neural Networks*, 1997. Режим доступу: https://ieeexplore.ieee.org/document/623216
9. Jan Ruben Zilke. Extracting Rules from Deep Neural Networks. *Master's Thesis*, 2015. Режим доступу: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.724.6244&rep=rep1&type=pdf
10. Jordi Torres. Learning Process of a Deep Neural Network. *Towards Data Science*, 2020. Режим доступу: https://towardsdatascience.com/learning-process-of-a-deep-neural-network-5a9768d7a651
11. Kamal Kumar Sethi, Durgesh Kumar Mishra, Bharat Mishra. Extended Taxonomy of Rule Extraction Techniques and Assessment of KDRuleEx. *International Journal of Computer Applications*, 2012. Режим доступу: https://www.researchgate.net/publication/258652014_Extended_Taxonomy_of_Rule_Extraction_Techniques_and_Assessment_of_KDRuleEx
12. Kamal Kumar Sethi, Durgesh Kumar Mishra, Bharat Mishra. KDRuleEx: A Novel Approach for Enhancing User Comprehensibility Using Rule Extraction. *Third International Conference on Intelligent Systems Modelling and Simulation, IEEE*, 2012. Режим доступу: https://ieeexplore.ieee.org/document/6169675
13. LiMin Fu. Rule generation from neural networks. *IEEE Transactions on Systems, Man, and Cy-bernetics*, 1994. Режим доступу: https://ieeexplore.ieee.org/document/299696
14. M Gethsiyal Augasta, Thangairulappan Kathirvalavakumar. Reverse engineering the neural networks for rule extraction in classification problems. *Neural processing letters*, 2012. Режим доступу: https://www.researchgate.net/publication/216628594_Reverse_Engineering_the_Neural_Networks_for_Rule_Extraction_in_Classification_Problems

15. Makoto Sato, Hiroshi Tsukimoto. Rule extraction from neural networks via decision tree in-duction. *International Joint Conference on Neural Networks, IEEE*, 2001. Режим доступу: https://ieeexplore.ieee.org/document/938448
16. Mark W. Craven, Jude W. Shavlik. Extracting tree-structured representations of trained networks. *Advances in neural information processing systems*, 1996. Режим доступу: https://proceedings.neurips.cc/paper/1995/file/45f31d16b1058d586fc3be7207b58053-Paper.pdf
17. Mark W. Craven. Extracting comprehensible models from trained neural networks. *Technical report, University of Wisconsin-Madison Department of Computer Sciences*, 1996. Режим доступу: https://www.biostat.wisc.edu/~craven/papers/thesis.pdf
18. Matěj Fanta. Rules extraction from deep neural networks. *Master's thesis*, 2019. Режим доступу: https://www.researchgate.net/publication/335146775_Rules_extraction_from_deep_neural_networks_Master_thesis
19. Rudy Setiono, Huan Liu. Understanding Neural Networks via Rule Extraction. *National University of Singapore*. Режим доступу: https://www.ijcai.org/Proceedings/95-1/Papers/063.pdf
20. Rudy Setiono, Wee Kheng Leow. FERNN: An algorithm for fast extraction of rules from neural networks. *Applied Intelligence*, 2000. Режим доступу: https://link.springer.com/article/10.1023/A:1008307919726
21. Rudy Setiono. Extracting M-of-N Rules from Trained Neural Networks. *National University of Singapore*. Режим доступу: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.37.4104&rep=rep1&type=pdf
22. Sebastian Thrun. Extracting provably correct rules from artificial neural networks. *Technical report, University of Bonn, Institut für Informatik III*, 1993. Режим доступу: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.31.9441

23. Sebastian Thrun. Extracting rules from artificial neural networks with distributed representations. *Advances in neural information processing systems*, 1995. Режим доступу: https://proceedings.neurips.cc/paper/1994/file/bea5955b308361a1b07bc55042e25e54-Paper.pdf
24. Sophie Burkhardt, Jannis Brugger, Nicolas Wagner, Zahra Ahmadi, Kristian Kersting, Stefan Kramer. Rule Extraction From Binary Neural Networks With Convolutional Rules for Model Validation. *Front. Artif. Intell.*, 2021. Режим доступу: https://www.frontiersin.org/articles/10.3389/frai.2021.642263/full
25. Tameru Hailesilassie. Rule Extraction Algorithm for Deep Neural Networks: A Review. *National University of Science and Technology*, 2016. Режим доступу: https://arxiv.org/ftp/arxiv/papers/1610/1610.05267.pdf
26. Zhengping Che, Sanjay Purushotham, Robinder Khemani, Yan Liu. Distilling Knowledge from Deep Networks with Applications to Healthcare Domain. 2015. Режим доступу: https://arxiv.org/pdf/1512.03542.pdf
27. Zhi-Hua Zhou, Shi-Fu Chen, Zhao-Qian Chen. A statistics based approach for extracting priority rules from trained neural networks. *International Joint Conference on Neural Networks, IEEE*, 2000. Режим доступу: https://ieeexplore.ieee.org/document/861337
28. Zhi-Hua Zhou. Rule Extraction: Using Neural Networks or For Neural Networks? *Nanjing University*. Режим доступу: https://cs.nju.edu.cn/_upload/tpl/01/0b/267/template267/zhouzh.files/publication/jcst04.pdf