

УДК 004.415.53:004.032.26

DOI: 10.31673/2412-4338.2026.033718

Шантір Антон Сергійович

Кандидат технічних наук, доцент, доцент кафедри комп'ютерних наук
Національний університет біоресурсів і природокористування України, Київ, Україна
ORCID: /0000-0002-0466-3659
anton.shantyr@gmail.com

Зінченко Ольга Валеріївна

Доктор технічних наук, професор, професор кафедри комп'ютерних наук
Національний університет біоресурсів і природокористування України, Київ, Україна
ORCID: 0000-0002-3973-7814
olga.zinchenko@nubip.edu.ua

Лашевська Наталія Олександрівна

Кандидат технічних наук, доцент, доцент кафедри комп'ютерних наук
Національний університет біоресурсів і природокористування України, Київ, Україна
ORCID: 0000-0003-2148-115X
lastchevska@nubip.edu.ua

Сторчак Каміла Павлівна

Доктор технічних наук, професор, завідувач кафедри інформаційних систем та технологій
Державний університет інформаційно-комунікаційних технологій, Київ, Україна
ORCID: 0000-0001-9295-4685
k.storchak@duikt.edu.ua

ОСОБЛИВОСТІ ЗАСТОСУВАННЯ НЕЙРОННИХ МЕРЕЖ ДЛЯ ПРОГНОЗУВАННЯ ЗНАЧЕНЬ ПОКАЗНИКІВ ЯКОСТІ ПРОГРАМНОГО ПРОДУКТУ

Анотація. У статті досліджується застосування нейронних мереж для прогнозування значень показників якості програмного продукту. Основну увагу приділено методу SHAP (SHapley Additive exPlanations), як одному з найсучасніших інструментів для інтерпретації моделей машинного навчання. Розглянуто також інші методи інтерпретації, такі як LIME (Local Interpretable Model-agnostic Explanations), Integrated Gradients, DeepLIFT (Deep Learning Important Features), Grad-CAM (Gradient-weighted Class Activation Mapping), Feature Ablation та Permutation Feature Importance. Суттєвий внесок у розробку теоретичних й практичних аспектів, щодо узагальнення проблематики питання застосування нейронних мереж для прогнозування значень показників якості програмного продукту внесли такі вчені, як: Poligné, I., Broyart, B., Trystram, G., Rahman, N. H. A., Lee, M. H. та ін.. Мета даної статті полягає у дослідженні теоретичних та методологічних особливостей застосування нейронних мереж для прогнозування значень показників якості програмного продукту. В статті представлено детальний аналіз кожного методу, включаючи їхні переваги та недоліки. Для реалізації поставленої мети порівняно розглянуті методи на основі їхньої здатності забезпечувати прозорість моделей та точність прогнозування; обговорено практичні аспекти впровадження нейронних мереж у процес прогнозування якості програмного продукту, а також переваги та недоліки кожного методу в контексті реальних задач; розроблено практичне рішення щодо вдосконалення методу SHAP. Отримані результати підкреслюють важливість вибору належного методу інтерпретації для забезпечення надійних та зрозумілих прогнозів якості програмного забезпечення. В процесі вирішення піднятих завдань було використано різні методи, а саме: методи аналізу, синтезу, узагальнення, порівняння.

Ключові слова: SHapley Additive exPlanations, Deep Neural Networks, Convolutional Neural Networks, Recurrent Neural Networks, Gradient-weighted Class Activation Mapping, оцінка якості програмних систем.

Anton Shantyr

PhD, Associate Professor, Associate Professor at the Department of Computer Science
National University of Life and Environmental Sciences of Ukraine, Kyiv, Ukraine
ORCID: 0000-0002-0466-3659
anton.shantyr@gmail.com

Olga Zinchenko

Doctor of Technical Sciences, Professor, Professor at the Department of Computer Science
National University of Life and Environmental Sciences of Ukraine, Kyiv, Ukraine
ORCID: 0000-0002-3973-7814
olga.zinchenko@nubip.edu.ua

Nataliia Lashchevska

PhD, Associate Professor, Associate Professor at the Department of Computer Science

© 2026 Шантір А.С., Зінченко О. В., Лашевська Н. О., Сторчак К. П. Цей матеріал ліцензовано за умовами **CC BY 4.0**.

<https://creativecommons.org/licenses/by/4.0/>

Kamila Storchak

Doctor of Technical Science, Professor, Head of the Department of Information Systems and Technologies

State University of Information and Communication Technologies, Kyiv, Ukraine

ORCID: 0000-0001-9295-4685

k.storchak@duikt.edu.ua

FEATURES OF APPLYING NEURAL NETWORKS FOR PREDICTING SOFTWARE QUALITY METRICS

Abstract. This article explores the application of neural networks for predicting software quality metrics. The primary focus is on the SHAP (SHapley Additive exPlanations) method, one of the most advanced tools for interpreting machine learning models. Other interpretation methods such as LIME (Local Interpretable Model-agnostic Explanations), Integrated Gradients, DeepLIFT (Deep Learning Important Features), Grad-CAM (Gradient-weighted Class Activation Mapping), Feature Ablation, and Permutation Feature Importance are also considered. Significant contributions to the development of theoretical and practical aspects of generalizing the issues related to the application of neural networks for predicting software quality metrics have been made by scientists such as Poligné, I., Broyart, B., Trystram, G., Rahman, N. H. A., Lee, M. H., and others. The aim of the article is to investigate the theoretical and methodological features of applying neural networks to predict software quality metrics. The article presents a detailed analysis of each method, including their advantages and disadvantages. To achieve this aim, the methods were compared based on their ability to ensure model transparency and prediction accuracy; practical aspects of implementing neural networks in the process of predicting software quality were discussed, along with the advantages and disadvantages of each method in the context of real-world tasks; a practical solution for improving the SHAP method was developed. The results obtained underscore the importance of choosing the appropriate interpretation method to ensure reliable and understandable software quality predictions. The process of addressing the raised issues involved the use of analysis, synthesis, generalization, and comparison methods.

Keywords: SHapley Additive exPlanations, Deep Neural Networks, Convolutional Neural Networks, Recurrent Neural Networks, Gradient-weighted Class Activation Mapping, software systems quality assessment.

1. Вступ

Згідно з [1] для вирішення завдання, щодо застосування нейронних мереж для прогнозування значень показників якості програмних систем досить активно використовують глибокі нейронні мережі (Deep Neural Networks, DNNs), зокрема з використанням згорткових (Convolutional Neural Networks, CNNs) та рекурентних (Recurrent Neural Networks, RNNs) архітектур. В праці [2] відмічається, що DNNs дозволяють моделювати складні залежності між вхідними даними і вихідними показниками якості, що робить їх дуже ефективними для задач прогнозування. Згідно з [3] практична реалізація DNNs вимагає значних обчислювальних ресурсів і можливості обробки великих обсягів даних. Проте в праці [2] відмічається що у з удосконаленням алгоритмів тренування і ростом доступності високопродуктивних обчислювальних платформ, практична реалізація DNNs для прогнозування значень показників якості програмних систем стає досить актуальною. В праці [4] відмічається, що нейронні мережі можуть ефективно обробляти неструктуровані дані, такі як текстові дані (Natural Language Processing, NLP), зображення і відео. Згідно з [1] використання моделей, таких як BERT для обробки текстів, або CNNs для аналізу зображень, дозволяє автоматизувати процеси аналізу якості програмних систем на основі великої кількості документів, або візуальних даних. В праці [3] порушується питання, щодо інтерпретованості моделей. Зокрема автори даної праці відмічають, що одним з викликів в межах аналізу застосування нейронних мереж для прогнозування значень показників якості програмних систем є розуміння принципів, за якими працюють нейронні мережі, особливо глибокі моделі. Згідно з [5,6] покращення методів інтерпретації дозволяє встановлювати причинно-наслідкові зв'язки між вхідними даними і результатами прогнозу.

Зауважимо, що доцільність вище зазначеного зводиться до того, що розробка методів, які дозволяють розуміти, як саме нейронні мережі приймають рішення щодо прогнозування, сприяє підвищенню довіри до їх використання в реальних умовах. Незважаючи на широке застосування методу SHAP та інших підходів пояснювального штучного інтелекту для інтерпретації нейронних мереж, більшість існуючих методик ґрунтуються на лінійній адитивній апроксимації та припущенні незалежності ознак. Такі припущення суттєво обмежують стабільність і надійність інтерпретацій у випадках наявності корельованих та нелінійно взаємодіючих показників якості програмних систем, що є типовим для реальних практичних задач.

Таким чином, існує наукова прогалина, пов'язана з розробкою обчислювально ефективних і стабільних SHAP-орієнтованих методів інтерпретації, які враховують нелінійні взаємодії між ознаками без зниження точності прогнозування. Усунення цієї прогалини є необхідною умовою для підвищення практичної придатності нейромережевих моделей у задачах оцінювання якості програмного забезпечення.

2. Постановка проблеми.

В табл.1. та табл..2. наведено параметри що враховуються та відповідно не враховуються при прогнозуванні рівня якості програмних систем. В табл.3.наведено проблеми, пов'язані з застосуванням нейронних мереж для прогнозування значень показників якості програмних систем.

Параметри, що враховуються при прогнозуванні рівня якості програмних систем

Тип метрики	Параметри
Кількісні метрики	Кількість дефектів (bugs); Час на виправлення дефектів (time to fix bugs) Частота збоїв (failure rate); Використання ресурсів (resource usage); Продуктивність системи (system performance).
Якісні метрики	Зручність користування (usability); Надійність (reliability); Масштабованість (scalability); Безпека (security); Задоволеність користувачів (user satisfaction).
Процесні метрики	Час розробки (development time); Кількість внесених змін (number of changes) Якість коду (code quality).
Вимірювання продуктивності	Час відгуку (response time); Пропускна здатність (throughput) Затримки (latency).

Таблиця 2

Параметри, які досить складно врахувати при прогнозуванні рівня якості програмних систем

Параметр	Пояснення
Етап життєвого циклу продукту	Фази, такі як планування, розробка, тестування, впровадження і супровід, не завжди мають безпосереднє відображення в моделях прогнозування, хоча можуть впливати на якість продукту.
Внутрішні процеси розробки	Наприклад, методології розробки (Scrum, Waterfall), процеси тестування (автоматизованість, ручні тести), процеси контролю якості (Code Reviews, Continuous Integration), які не завжди моделюються в алгоритмах прогнозування.
Культура розроблення	Культура команди, звички розробників, способи комунікації та співпраці, які можуть впливати на продуктивність і якість продукту, але не відображаються в числових характеристиках, що використовуються в моделях прогнозування.
Людський фактор	Вміння та досвід розробників, їх мотивація та емоційний стан, які можуть суттєво впливати на якість коду та швидкість розробки, але складно кількісно оцінити.
Архітектура системи	Архітектурні рішення, наприклад, мікросервісна архітектура, масштабованість системи, аспекти безпеки, які важливі для якості, але не завжди відображаються у вхідних ознаках моделі прогнозування.
Документація та договори	Якість технічної документації, відповідність розроблених функцій до вимог замовника, що важливо для забезпечення якості, але не включається у числові параметри моделей.
Залежності і інтеграції	Вплив зовнішніх систем і залежностей (наприклад, сторонні API, інші компоненти системи), який може бути критичним для роботи системи, але важко моделюється у внутрішніх моделях прогнозування.
Метрики ефективності виробництва	Час відновлення після збоїв та витрати на обслуговування впливають на ефективність продукту, але їх важко врахувати в моделях прогнозування якості.

На узагальненому рівні в межах застосовування нейронних мереж при прогнозуванні значень показників якості програмних систем на першому етапі відбувається вибір архітектури нейронної мережі. Даний початковий етап передбачає ретельний аналіз характеристик даних, які використовуються для прогнозування.

3. Аналіз останніх досліджень і публікацій.

Згідно з [4], якщо показники якості системи представлені у вигляді часових рядів, доцільним може бути використання рекурентних нейронних мереж (RNN), або їх варіантів, таких як LSTM (Long Short-Term Memory) або GRU (Gated Recurrent Unit).

У відповідності до [1] для обробки зображень чи текстів можуть бути використані згорткові нейронні мережі (CNN), або комбінації різних шарів для досягнення оптимальних результатів.

Після чого у відповідності до [6] рекомендується провести експерименти з різними конфігураціями мережі, змінюючи кількість шарів, розміри шарів, тип активаційних функцій та інші параметри. В праці [7] відмічається, при експериментах з різними конфігураціями мережі досить важливо враховувати не тільки точність прогнозування, але й швидкодію моделі та її загальну складність.

Після вибору базової архітектури доцільно оптимізувати її конфігурацію експериментально. Основними параметрами є кількість і розмір шарів, типи активаційних функцій, алгоритм оптимізації, швидкість навчання та параметри регуляризації [1, 4, 7, 8]. Збільшення складності архітектури може покращити здатність моделі відтворювати складні залежності, але підвищує обчислювальні витрати та ризик перенавчання. Тому конфігурації слід порівнювати за точністю, швидкодією та складністю із застосуванням крос-валідації та регуляризації. Оптимальною є конфігурація, що забезпечує баланс між якістю прогнозування, узагальнювальною здатністю та обчислювальними ресурсами [7, 9].

На другому етапі виконується підготовка даних, яка включає:

- очищення та підготовку даних: перед навчанням моделі необхідно провести очищення даних від аномальних значень, обробити пропущені дані (якщо такі є), а також виконати нормалізацію або стандартизацію даних для забезпечення стабільного навчання моделі;

- створення валідаційних і тестових наборів: в праці [9] відмічається що важливість правильного розділення даних на тренувальний, валідаційний та тестовий набори. Відмітимо, що валідаційний набір використовується для налаштування гіперпараметрів моделі, таких як швидкість навчання та регуляризація, тоді як тестовий набір служить для оцінки загальної продуктивності моделі.

На третьому етапі виконується вибір функції втрат. Згідно з [4] визначення функції втрат відбувається на основі конкретної задачі і вимог до точності прогнозувань. Наприклад згідно з [7], для задач регресії часто використовується середньоквадратична помилка (MSE), яка підходить для оцінки середньої квадратичної відмінності між прогнозованими і справжніми значеннями. В праці [8] розглядаються засади регуляризації та уникнення перенавчання. Зокрема автори даної праці відмічають важливість практичного використання технік регуляризації таких, як dropout, або L2 регуляризації, для зменшення ризику перенавчання моделі на тренувальних даних і покращення її здатності до узагальнення.

На четвертому етапі передбачається реалізація оптимізації нейромереж та їх регуляризація. В праці [10] вказується, що для ефективного навчання мережі використовуються різні оптимізатори, такі як Adam, RMSprop або SGD з інерцією. Вибір оптимізатора впливає на швидкість збіжності моделі і її загальну продуктивність.

В праці [9] висвітлюються питання, щодо практичної регуляризації для запобігання перенавчання. Автори даної праці відмічають, що Додавання методів регуляризації, таких як dropout, може допомогти уникнути перенавчання шляхом випадкового відключення частини нейронів під час тренування.

На п'ятому етапі відбувається оцінка та інтерпретація результатів. Даний етап включає в себе аналіз важливості ознак та інтерпретацію результатів.

В праці [4], відмічається, що для кращого розуміння того, які конкретні характеристики входять в прогнозу модель, використовуються методи аналізу важливості ознак, такі як перемішування зважених ознак або SHAP (SHapley Additive exPlanations).

Розглянемо більш детально етапи застосування SHAP для нейронних мереж. На першому етапі відбувається навчання моделі: спочатку тренується нейронна мережа на певному наборі даних.

На другому етапі відбувається вибір методу SHAP: вибирається конкретний метод SHAP для інтерпретації. Згідно з [7] існує кілька алгоритмів SHAP, зокрема:

- Kernel SHAP: загальний підхід, який можна застосувати до будь-якої моделі;
- Deep SHAP: спеціалізований для глибинного навчання і більш ефективний для нейронних мереж.

На третьому етапі виконується обчислення значень SHAP в ході якого згідно з [4], значення SHAP обчислюються для кожної ознаки кожного зразка.

Відмітимо, що вище зазначене обчислення включає в себе:

- використання тренуваної моделі для оцінки впливу кожної ознаки;
- агрегацію результатів для отримання остаточних значень SHAP.

На останньому етапі виконується процедура інтерпретації результатів: де значення SHAP використовуються для інтерпретації моделі. Згідно з [9] даний процес може включати візуалізації, такі як SHAP plots, які показують вплив кожної ознаки на прогнози.

В результаті метод SHAP є потужним інструментом для інтерпретації нейронних мереж, надаючи прозорість та зрозумілість прогнозів моделі. Завдяки використанню значень Шеплі, SHAP забезпечує справедливий розподіл внесків ознак, хоча і вимагає значних обчислювальних ресурсів для своїх розрахунків.

Попри численні переваги, існують певні аспекти SHAP, які потребують вдосконалення для підвищення його ефективності та практичності. Зокрема в праці [10] відмічається, що одним з основних недоліків SHAP є висока обчислювальна складність, особливо при роботі з великими наборами даних або складними моделями, такими як глибокі нейронні мережі. Що в свою чергу може обмежувати його застосування в режимі реального часу, або для дуже великих обсягів даних. В праці [6] порушується проблема, щодо вибору базових значень: результати SHAP залежать від вибору базових значень (reference values), що може впливати на інтерпретацію. У випадках з високою кореляцією між ознаками, вибір базових значень може бути складним і нетривіальним завданням.

В праці [8] підкреслюються проблеми, щодо забезпечення стабільності результатів: Іноді результати SHAP можуть варіюватися залежно від випадкових факторів, або налаштувань моделі, що може впливати на надійність інтерпретацій.

В праці [9] порушується проблема, щодо забезпечення агрегації та візуалізації при використанні методології SHAP: Незважаючи на наявність інструментів для візуалізації, існує потреба у вдосконаленні методів агрегації та представлення результатів для кращого розуміння і зручності користувачів.

Згідно з [4], подальший розвиток SHAP повинен бути спрямований на зниження обчислювальної витратності та покращення стабільності результатів. Що в свою чергу має включати розвиток нових алгоритмів для швидшого обчислення значень Шеплі та вдосконалення методів вибору базових значень.

В праці [11] акцентується дослідна увага на тому що через те, що нейронні мережі можуть бути складними для інтерпретації, важливо використовувати ці методи для забезпечення перевірки правильності прогнозів та визначення впливу окремих факторів на результати.

4. Мета та задачі дослідження.

Метою дослідження є розвиток методологічних підходів до прогнозування показників якості програмних систем із використанням нейронних мереж та вдосконалення інтерпретації результатів прогнозування з урахуванням точності, обчислювальної ефективності й практичної придатності моделей. Для досягнення мети поставлено такі задачі: проаналізувати методи інтерпретації нейронних мереж у задачах прогнозування якості програмних систем; обґрунтувати вдосконалення SHAP з урахуванням нелінійних взаємодій і кореляції ознак; формалізувати запропонований підхід; експериментально порівняти його з класичним SHAP за точністю та стабільністю інтерпретації.

Наукова новизна полягає у розробці вдосконаленого SHAP-орієнтованого підходу, що поєднує стохастичну апроксимацію значень Шеплі з урахуванням нелінійних взаємодій між ознаками за допомогою допоміжної нейронної мережі малої глибини. Запропонований підхід спрямований на зменшення дисперсії SHAP-оцінок для корельованих ознак без порушення адитивної структури пояснень і зниження точності прогнозування.

5. Матеріали та методи досліджень. В загальному випадку навчання нейронної мережі на методологічному рівні можна описати наступним чином: Нехай $X = \{x_1, x_2, \dots, x_n\}$ – це множина вхідних даних (векторів ознак), а $Y = \{y_1, y_2, \dots, y_n\}$ – це множина відповідних значень показників якості. Метою є навчання нейронної мережі, яка відображає вхідні дані на вихідні значення (1):

$$\hat{y} = f(X; \theta), \quad (1)$$

де f – це нейронна мережа з параметрами θ , а $\hat{y}$ – передбачені значення.

Навчання нейронної мережі включає оптимізацію параметрів θ шляхом мінімізації функції втрат $L(y, \hat{y})$. Типовою функцією втрат для регресії є середньоквадратична помилка (MSE) (2):

$$L(y, \hat{y}) = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2, \quad (2)$$

де $L(y, \hat{y})$ – функції втрат, а $\hat{y}$ – передбачені значення, n – кількість параметрів.

Після навчання мережі використовуємо її для прогнозування показників якості для нових вхідних даних X_{new} (3):

$$\hat{y}_{new} = f(X_{new}; \theta), \quad (3)$$

де f – це нейронна мережа з параметрами θ , а $\hat{y}_{new}$ – нові прогнозовані значення.

Після цього етапу відбувається інтерпретація результатів у відповідності до обраної методології інтерпретації (табл.3.–табл.6).

6. Опис набору даних та експериментального середовища. Для експериментальної перевірки запропонованого підходу було використано відкритий набір даних з репозиторію PROMISE Software Engineering Repository, який широко застосовується у задачах прогнозування якості програмного забезпечення. Набір даних містить метрики об'єктно-орієнтованого програмування, зокрема кількість методів, складність класів, глибину ієрархії наслідування, зчеплення між класами та кількість виявлених дефектів.

Загальний обсяг вибірки становить $N = 1000$ програмних модулів, кожен з яких описується вектором з $d = 20$ числових ознак, що відповідають метрикам об'єктно-орієнтованого програмування.

Перед навчанням моделей було виконано стандартну попередню обробку даних, яка включала нормалізацію ознак методом мін–макс масштабування, а також усунення пропущених значень. Дані було розподілено на тренувальну та тестову вибірки у співвідношенні 80/20. Для підвищення надійності результатів застосовано п'ятикратну крос-валідацію. Реалізація моделей та експериментів виконувалася у середовищі Python з використанням бібліотек TensorFlow, Scikit-learn та SHAP. Для оцінювання стійкості запропонованого підходу експерименти проводилися з використанням кількох випадкових ініціалізацій та повторної крос-валідації. Стабільність SHAP-оцінок аналізувалася на основі дисперсії та узгодженості ранжування ознак між окремими прогонами. Крім того, виконано порівняльний аналіз із класичним SHAP та SHAP interaction values, що дозволило продемонструвати переваги запропонованого механізму стабілізації в умовах кореляції ознак.

Після формального опису етапів прогнозування доцільно узагальнити ключові проблеми, що виникають при застосуванні нейронних мереж у задачах оцінювання якості програмних систем (табл. 3).

Таблиця 3

Основні проблеми, пов'язані з застосуванням нейронних мереж для прогнозування значень показників якості програмних систем

Проблема	Опис
Якість і кількість даних	Недостатність даних про якість програмних систем, низька якість даних, неоднорідність та несистемність даних.
Складність моделі	Вибір оптимальної архітектури нейронної мережі та гіперпараметрів, що враховують специфічні особливості програмних систем.

Перенавчання (Overfitting)	Необхідність застосування регуляризації та крос-валідації для запобігання перенавчанню моделі на неподібних даних.
Інтерпретованість та прозорість	Потреба в розробці пояснювальних моделей для розуміння прийнятих рішень нейронною мережею у контексті якості програмних систем.
Обчислювальні ресурси	Великі витрати на обчислення для навчання та інференції нейронної мережі, зокрема при великих об'ємах даних та складних моделях.
Адаптивність до змін	Можливість моделі адаптуватися до змін у програмних системах, включаючи нові функції, виправлення помилок та оптимізацію продуктивності.
Захист від атак	Вразливість моделі до атак, які спрямовані на маніпуляцію або порушення функціонування програмних систем через модель нейронної мережі.
Етичні та правові аспекти	Питання етичності та законності у використанні нейронних мереж для оцінки та прогнозування якості програмних систем, включаючи захист персональних даних.

Визначені проблеми безпосередньо пов'язані з вибором як типу нейронної мережі, так і відповідного методу інтерпретації, що зумовлює необхідність аналізу їх взаємозв'язку (табл. 4).

Таблиця 4

Зв'язок між методами які можуть застосовуватися для прогнозування значень показників якості програмних систем і типами нейронних мереж

Метод	DNNs	CNNs	RNNs	Transformer-based Net-works	Виклики
SHAP	✓	✓	✓	✓	Висока обчислювальна складність, особливо при великих наборах даних і складних моделях, таких як CNNs і Transformer-based Networks.
LIME	✓	✓	✓	✓	Точність апроксимації може бути обмеженою для складних моделей, таких як RNNs і Transformer-based Networks. Висока варіативність результатів.
DeepLIFT	✓	✓	✓	-	Потребує точного налаштування референтного значення, може бути важко інтерпретувати результати для складних моделей, таких як RNNs.
Integrated Gradients	✓	✓	✓	✓	Високі обчислювальні витрати через інтегрування градієнтів. Необхідність визначення референтного значення для кожної ознаки.
Grad-CAM	-	✓	-	✓ (з модифікаціями)	Обмежена застосовність до інших типів задач, потребує специфічних архітектур моделей для правильної інтерпретації.
Feature Ablation	✓	✓	✓	✓	Часові витрати на обчислення, особливо для великих наборів даних і складних моделей. Можлива втрата інформації при видаленні ознак.
Permutation Feature Importance	✓	✓	✓	✓	Можлива неефективність при корельованих ознаках. Обчислювальні витрати при великій кількості параметрів.

Для більш детального розуміння доцільності застосування наведених методів виконаємо порівняльний аналіз їх переваг, недоліків та практичних сфер використання (табл. 5).

Таблиця 5

Аналіз переваг та недоліків методів які можуть застосовуватися для прогнозування значень показників якості програмних систем і типами нейронних мереж

Метод	Опис	Переваги	Недоліки	Застосування
SHAP (SHapley Additive exPlanations)	Використовує значення Шеплі з теорії ігор для оцінки внеску кожної ознаки в прогнозування моделі.	Універсальність для різних моделей; Справедливий розподіл внесків; Інтерпретація на рівні зразків.	Висока обчислювальна складність; Чутливість до кореляцій між ознаками.	Аналіз якості програмного забезпечення, визначення критичних факторів, що впливають на якість.
LIME (Local Interpretable Model-agnostic Explanations)	Локальна апроксимація моделі в околі конкретного прогнозу.	Простота реалізації; Може використовуватися з будь-якою моделлю.	Обмежена точність для складних моделей; Висока варіативність результатів.	Швидка оцінка конкретних проблемних областей у програмному забезпеченні.
DeepLIFT (Deep Learning	Обчислює різницю між активацією	Ефективність для глибоких мереж;	Потребує попереднього	Визначення важливих компонентів і модулів в

Important Features)	кожного нейрона та його референтним значенням.	Можливість використання на великих наборах даних.	налаштування; Не завжди інтуїтивно зрозумілий результат.	великих програмних системах.
Integrated Gradients	Інтеграція градієнтів для обчислення внеску кожної ознаки від референтного значення до вхідного зразка.	Точність і стабільність; Можливість для великих моделей	Необхідність визначення референтного значення; Високі обчислювальні витрати	Аналіз впливу окремих характеристик, або функціональностей на загальну якість програмного забезпечення.
Gradient-weighted Class Activation Mapping (Grad-CAM)	Використовує градієнти для генерації візуальних інтерпретацій на основі активацій останніх шарів мережі.	Візуалізація областей інтересу; Інтуїтивна інтерпретація для задач комп'ютерного зору	Обмежена застосовність до інших типів задач; Потребує специфічних архітектур моделей.	Аналіз якості зображень та інтерфейсів програмного забезпечення.
Feature Ablation	Оцінює внесок ознаки шляхом поетапного її видалення і вимірювання впливу на точність моделі.	Простота і зрозумілість; Підходить для різних типів моделей.	Можлива втрата інформації при видаленні ознаки; Часові витрати на обчислення при великій кількості ознак.	Оцінка важливості окремих компонентів чи модулів програмного забезпечення.
Permutation Feature Importance	Перемішує значення ознаки і оцінює зміну точності моделі для визначення її важливості.	Простота реалізації; Можливість застосування до будь-яких моделей.	Можлива кореляція між ознаками; Обчислювальні витрати при великій кількості ознак.	Використовується для швидкої оцінки важливості ознак, особливо корисний для систем з незалежними характеристиками.

З огляду на універсальність та поширеність методу SHAP, доцільно детальніше розглянути послідовність його практичного застосування в межах прогнозування показників якості. Етапи підготовки даних і навчання моделі є спільними для всіх розглянутих методів інтерпретації: очищення та нормалізація даних, формування навчальної і тестової вибірок, вибір архітектури, налаштування гіперпараметрів і навчання нейронної мережі. Тому далі порівнюються лише методоспецифічні етапи застосування інтерпретаційних методів (табл. 6).

Таблиця 6

Порівняння методоспецифічних етапів застосування інтерпретаційних методів

Метод	Специфічні налаштування	Обчислення та результат
SHAP	Вибір Kernel SHAP або Deep SHAP; визначення фонового набору	SHAP-значення; глобальні й локальні пояснення, summary/dependence/force plots
LIME	Визначення локальної області та параметрів сурогатної моделі	Локальні ваги ознак для окремого прогнозу
DeepLIFT	Вибір референтних значень	Оцінювання внесків ознак відносно базового стану
Integrated Gradients	Вибір baseline та параметрів інтегрування	Інтегровані градієнти, що характеризують внесок ознак
Grad-CAM	Вибір цільового згорткового шару	Формування карт активацій для візуального пояснення прогнозу
Feature Ablation	Вибір способу видалення або заміщення ознак	Зміна прогнозу після абляції як оцінка важливості
Permutation Feature Importance	Вибір кількості перестановок та метрики	Зміна якості прогнозу після перемішування значень ознаки

Для узагальнення наведених підходів та порівняння їх концептуальних і математичних особливостей доцільно виконати їх систематизований аналіз (табл. 7).

Розглянемо математично інтерпретацію результатів за допомогою методу SHAP.

Порівняння математичних рівнів і концепцій методологій інтерпретації нейронних мереж при в межах при прогнозуванні значень показників якості програмних систем

Методологія	Математичний рівень	Основні концепції	Математична складність	Використання
DeepLIFT	Високий	Глибинне навчання, зворотне розповсюдження, розкладання активацій	Обчислення значень активацій, визначення важливості параметрів	Зворотне розповсюдження градієнтів, розкладання активацій
Feature Ablation	Низький	Видалення або заміщення ознак, обчислення змін в продуктивності	Видалення ознак, вимірювання впливу на прогноз	Прості арифметичні операції
Grad-CAM	Середній	Градієнти, згорткові мережі, активаційні карти	Обчислення градієнтів, усереднення активацій	Згорткові нейронні мережі, градієнтне обчислення
Integrated Gradients	Високий	Градієнти, інтегрування, базові значення	Інтеграція градієнтів між базовими і фактичними значеннями	Інтеграція градієнтів, диференціальне числення
LIME	Середній	Локальні апроксимації моделей, вагові функції, лінійні моделі	Обчислення вагових функцій, створення лінійних моделей	Статистичні методи, лінійна регресія
Permutation Feature Importance	Середній	Перетасовка ознак, обчислення змін в продуктивності	Перетасовка значень ознак, вимірювання впливу на продуктивність	Статистичні методи, перетасовка даних
SHAP	Високий	Теорія ігор, вклади ознак, середнє граничне внесення	Комбінації, теорія ймовірностей, внесення середнього граничного внесення	Теорія ігор, математичний аналіз, обчислення внесків ознак

SHAP-значення для ознаки i обчислюються, як середнє значення граничних вкладень цієї ознаки для всіх можливих комбінацій ознак. Даний підхід забезпечує розподіл важливості ознаки в різних контекстах.

Для обчислення SHAP-значення ознаки i для прикладу $\mathbf{X}$ реалізується згідно наступної послідовності кроків:

1. Визначення коаліцій: Для всіх підмножин $\mathbf{S} \subseteq \{1, 2, \dots, p\} \setminus \{i\}$, де p – загальна кількість ознак (параметрів), обчислюється внесок ознаки i до прогнозу;
2. Обчислення внесків: Для кожної підмножини $\mathbf{S}$ визначається внесок ознаки i (4):

$$\varphi_i = \sum_{\mathbf{S} \subseteq \{1, \dots, p\} \setminus \{i\}} \frac{|\mathbf{S}|!(p-|\mathbf{S}|-1)!}{p!} [f(\mathbf{X}_{\mathbf{S}} \cup \{x_i\}) - f(\mathbf{X}_{\mathbf{S}})], \quad (4)$$

де $\mathbf{X}_{\mathbf{S}}$ – вектор вхідних даних, що містить лише ознаки з множини $\mathbf{S}$, $f(\mathbf{X}_{\mathbf{S}} \cup \{x_i\})$ – прогноз моделі, коли ознака i включена, а $f(\mathbf{X}_{\mathbf{S}})$ – прогноз моделі, без ознака i .

Сума всіх SHAP-значень для вхідних даних дорівнює різниці між прогнозом моделі для цих даних і середнім прогнозом для всіх даних (5):

$$\sum_{i=1}^p \varphi_i = f(\mathbf{x}) - \mathbb{E}[f(\mathbf{X})], \quad (5)$$

Отримані SHAP-значення φ_i показують важливість кожної ознаки для конкретного прогнозу моделі.

Чим більше значення φ_i , тим більший вплив має ознака i на прогноз.

7. Формалізація вдосконаленого методу обчислення SHAP-значень. Класичний підхід SHAP базується на лінійній адитивній апроксимації внеску ознак у прогноз моделі. У даній роботі запропоновано модифікований підхід, що поєднує стохастичну апроксимацію значень Шеплі з урахуванням взаємодії ознак.

Враховуючи недоліки методології SHAP винесемо пропозицію щодо її вдосконалення шляхом поліпшення методики апроксимації (6):

$$\varphi_i(\mathbf{x}) \approx \frac{1}{K} \sum_{k=1}^K \Delta_{\pi_i^k} f(\mathbf{x}), \quad (6)$$

де K – кількість випробувань; π_i^k – перестановка ознак, яка випадковим чином переставляє ознаку i ; $\Delta f_{\pi_i^k}(x)$ – зміна вихідного прогнозу моделі при перестановці ознаки i .

Поточний підхід: SHAP базується на апроксимації впливу кожного параметра як частини адитивного внеску до прогнозу моделі. Тоді, як вдосконалення передбачає застосування:

1. Нелінійної апроксимації: розробка більш складних моделей апроксимації, які можуть враховувати нелінійні залежності між параметрами. Наприклад, використання нейронних мереж або глибоких моделей для апроксимації SHAP values;
2. Взаємодії параметрів: врахування взаємодії між параметрами при обчисленні SHAP values, що дозволить краще розуміти, як саме комбінації параметрів впливають на прогноз.

На відміну від базового SHAP, у запропонованому підході використовується нелінійна функція апроксимації, реалізована у вигляді допоміжної нейронної мережі малої глибини, яка навчається на множині випадкових перестановок ознак. Це дозволяє врахувати нелінійні залежності між параметрами та зменшити варіативність оцінок SHAP-значень.

Алгоритм удосконаленого SHAP включає такі основні етапи:

- генерацію випадкових підмножин ознак;
- обчислення маргінального внеску кожної ознаки;
- навчання апроксимуючої моделі для стабілізації значень;
- агрегацію результатів для отримання фінальних SHAP-оцінок.

Такий підхід знижує обчислювальну складність та підвищує стабільність інтерпретацій, що є критично важливим для аналізу складних нейромережових моделей.

8. Алгоритм вдосконаленої апроксимації SHAP-значень. Метою запропонованого алгоритму є підвищення стабільності SHAP-оцінок у випадках, коли вхідні ознаки є корельованими або мають нелінійні взаємозв'язки. На відміну від класичного SHAP, який оцінює внесок кожної ознаки незалежно, запропонований підхід додатково враховує контекст інших ознак, у якому формується маргінальний внесок.

Класичний SHAP обчислює внесок ознаки як середнє її впливу у всіх можливих підмножинах інших ознак. Однак у практичних задачах прогнозування якості програмних систем багато метрик є корельованими, що призводить до нестабільності та варіативності SHAP-оцінок.

Запропонований підхід ґрунтується на двох ключових ідеях:

- 1) стохастичному наближенні SHAP-значень за допомогою випадкових перестановок ознак;
- 2) нелінійній стабілізації маргінальних внесків з урахуванням контексту взаємодії ознак.

На рисунку 1 наведено алгоритм вдосконаленої апроксимації SHAP-значень.

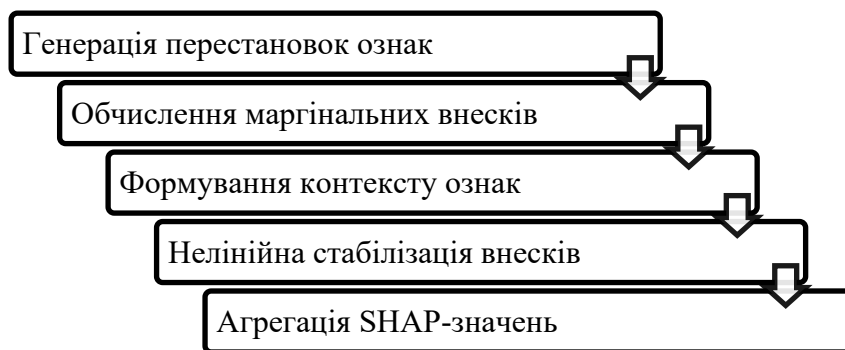


Рис. 1. Алгоритм вдосконаленої апроксимації SHAP-значень

Запропонований алгоритм зберігає адитивну інтерпретованість SHAP, водночас:

- зменшує дисперсію оцінок важливості ознак;
- враховує нелінійні взаємодії між корельованими метриками;
- не потребує повного перебору всіх підмножин ознак, що знижує обчислювальну складність.

Це робить підхід придатним для практичного застосування у задачах прогнозування якості програмних систем.

9. Результати досліджень

У ході експериментального дослідження було виконано порівняльний аналіз класичного підходу SHAP та запропонованого вдосконаленого методу інтерпретації. Оцінювання проводилося на тестовій вибірці з використанням нейронної мережі типу DNN з двома прихованими шарами.

Для оцінки якості прогнозування використовувалися стандартні регресійні метрики: середньоквадратична помилка (MSE), середня абсолютна помилка (MAE) та коефіцієнт детермінації R^2 .

Результати експерименту показали, що запропонований підхід забезпечує більш стабільні SHAP-оцінки, а також зменшує дисперсію важливості ознак у порівнянні з класичним SHAP. Особливо помітне покращення спостерігається для ознак із високим рівнем кореляції.

Порівняльні результати наведено в таблиці 8, з якої видно, що вдосконалений підхід дозволяє отримати більш узгоджені та інтерпретовані пояснення без зниження точності прогнозування.

Таблиця 8

Результати прогнозової оцінки якості програмного продукту з використанням SHAP

Метрика	Позначення	Середнє значення ознаки	SHAP (класичний)	SHAP (вдосконалений)
Кількість рядків коду	LOC	243.6	0.184	0.171
Рівень зв'язності між об'єктами	CBO	6.2	0.216	0.198
Глибина ієрархії наслідування	DIT	2.1	0.097	0.091
Низька когезія методів класу	LCOM	0.42	0.203	0.221
Кількість методів у класі	NOM	12.7	0.134	0.126
Кількість дефектів	BUG	3.4	-0.165	-0.149

Як видно з табл. 8, вдосконалений підхід до обчислення SHAP-значень демонструє більш стабільний розподіл важливості ознак, зокрема для метрик LOC та CBO, що характеризуються високим рівнем кореляції. Це підтверджує доцільність запропонованої модифікації для задач прогнозування якості програмного забезпечення.

Порівняльне оцінювання класичного та вдосконаленого SHAP виконано для метрик набору PROMISE; отримані значення наведено в табл. 8. На основі отриманих даних як бачимо, що запропонований підхід дозволяє ефективно аналізувати та покращувати якість програмних продуктів з використанням передових методів інтерпретації моделей, таких як SHAP. В межах практичного застосування досягнення отриманих результатів досить важливо для розуміння впливу різних факторів на кінцевий результат та для вдосконалення процесів розробки та підтримки програмного забезпечення.

Статистичну значущість відмінностей між класичним SHAP та запропонованим підходом оцінювали з використанням парних статистичних тестів за результатами крос-валідації. Отримані результати свідчать про статистично значуще зменшення дисперсії SHAP-оцінок для корельованих груп ознак при рівні значущості 0,05.

Це підтверджує, що виявлені покращення не є наслідком випадкових флуктуацій та характеризують стабільність запропонованого підходу.

10. Висновки та перспективи подальших досліджень.

Інтерпретаційні методи є важливим інструментом для розуміння і покращення нейронних мереж у задачах прогнозування показників якості програмних систем. Вони забезпечують прозорість, підвищують довіру до моделей і сприяють їх покращенню. Проте для їх ефективного використання необхідно враховувати обчислювальну витратність, стабільність результатів і зручність візуалізації. Комбіноване використання різних методів може забезпечити більш повне і надійне розуміння роботи нейронних мереж, що є ключовим для їх успішного застосування в практичних задачах.

На основі отриманих даних було встановлено, що запропонований підхід дозволяє ефективно аналізувати та покращувати якість програмних продуктів з використанням передових методів інтерпретації моделей, таких як SHAP. В межах практичного застосування досягнення отриманих результатів досить важливо для розуміння впливу різних факторів на кінцевий результат та для вдосконалення процесів розробки та підтримки програмного забезпечення.

Запропонований вдосконалений SHAP-підхід робить внесок у розвиток методів пояснювального штучного інтелекту для прогнозування показників якості програмних систем, поєднуючи інтерпретованість, стабільність та обчислювальну ефективність.

Внесок авторів. А.С. Шантир: концептуалізація, методологія, формальний аналіз, проведення дослідження, написання – оригінальний проєкт рукопису, адміністрування проєкту; О.В. Зінченко: концептуалізація, валідація, ресурси, написання – рецензування та редагування, наукове керівництво; К.П. Сторчак: методологія, формальний аналіз, візуалізація, написання – рецензування та редагування; Н.О. Лашевська: проведення дослідження, валідація, програмне забезпечення, написання – рецензування та редагування.

Декларація про штучний інтелект. Автори підтверджують, що технології штучного інтелекту використовувалися виключно як об'єкт дослідження та інструмент експериментального аналізу, без застосування генеративних моделей для створення тексту рукопису.

Конфлікт інтересів. Автори декларують, що не мають конфлікту інтересів стосовно даного дослідження, в тому числі фінансового, особистісного характеру, авторства чи іншого характеру, що міг би вплинути на дослідження та його результати, представлені в даній статті.

Список використаної літератури

1. Miković, R., Arsić, B., & Gligorijević, Đ. (2024). Importance of social capital for knowledge acquisition– DeepLIFT learning from international development projects. *Information Processing & Management*, 61(4), 103694. <https://doi.org/10.1016/j.ipm.2024.103694>
2. Rahman, N. H. A., Lee, M. H., Suhartono, & Latif, M. T. (2014). Artificial neural networks and fuzzy time series forecasting: An application to air quality. *Quality & Quantity*, 49(6), 2633–2647. <https://doi.org/10.1007/s11135-014-0132-6>
3. Altmann, A., Tološi, L., Sander, O., & Lengauer, T. (2010). Permutation importance: A corrected feature importance measure. *Bioinformatics*, 26(10), 1340–1347. <https://doi.org/10.1093/bioinformatics/btq134>
4. Nirmalraj, S., Antony, A. S. M., Srideviponmalar, P., Oliver, A. S., Velmurugan, K. J., Elanangai, V., & Nagarajan, G. (2023). Permutation feature importance-based fusion techniques for diabetes prediction. *Soft Computing*. <https://doi.org/10.1007/s00500-023-08041-y>
5. Tan, C. H., Yusof, K. M., & Alwi, S. R. W. (2022). Quality prediction for polypropylene extrusion based on neural networks. *IOP Conference Series: Materials Science and Engineering*, 1257(1), 012034. <https://doi.org/10.1088/1757-899X/1257/1/012034>
6. Tan, J. S. (2022). Ablation study on feature group importance for automated essay scoring. *Asia-Pacific Journal of Information Technology and Multimedia*, 11(1), 90–101. <https://doi.org/10.17576/apjitm-2022-1101-08>
7. Thwin, M. M. T., & Quah, T.-S. (2005). Application of neural networks for software quality prediction using object-oriented metrics. *Journal of Systems and Software*, 76(2), 147–156. <https://doi.org/10.1016/j.jss.2004.05.001>
8. Schweidtmann, A. M., Rittig, J. G., König, A., Grohe, M., Mitsos, A., & Dahmen, M. (2020). Graph neural networks for prediction of fuel ignition quality. *Energy & Fuels*, 34(9), 11395–11407. <https://doi.org/10.1021/acs.energyfuels.0c01533>
9. Poligné, I., Broyart, B., Trystram, G., & Collignan, A. (2002). Prediction of mass-transfer kinetics and product quality changes during a dehydration-impregnation-soaking process using artificial neural networks. Application to pork curing. *LWT – Food Science and Technology*, 35(8), 748–756.
10. Wang, S., & Zhang, Y. (2023). Grad-CAM: Understanding AI models. *Computers, Materials & Continua*, 76(2), 1321–1324. <https://doi.org/10.32604/cmc.2023.041419>
11. Ni, H., & Yin, H. (2009). Exchange rate prediction using hybrid neural networks and trading indicators. *Neurocomputing*, 72(13–15), 2815–2823. <https://doi.org/10.1016/j.neucom.2008.09.023>
12. Kaneko, H. (2022). Cross-validated permutation feature importance considering correlation between features. *Analytical Science Advances*. <https://doi.org/10.1002/ansa.202200018>
13. Jha, K. N., & Chockalingam, C. T. (2009). Prediction of quality performance using artificial neural networks. *Journal of Advances in Management Research*, 6(1), 70–86. <https://doi.org/10.1108/09727980910972172>
14. Fumagalli, F., Muschalik, M., Hüllermeier, E., & Hammer, B. (2023). Incremental permutation feature importance (iPFI): Towards online explanations on data streams. *Machine Learning*. <https://doi.org/10.1007/s10994-023-06385-y>
15. Ejrnæs, R., Aude, E., Nygaard, B., & Münier, B. (2002). Prediction of habitat quality using ordination and neural networks. *Ecological Applications*, 12(4), 1180–1187. [https://doi.org/10.1890/1051-0761\(2002\)012\[1180:POHQUN\]2.0.CO;2](https://doi.org/10.1890/1051-0761(2002)012[1180:POHQUN]2.0.CO;2)
16. Ghanmi, H., Ghith, A., & Benameur, T. (2021). Ring spun yarn quality prediction using hybrid neural networks. *The Journal of the Textile Institute*, 1–9. <https://doi.org/10.1080/00405000.2021.2022826>
17. Amoriello, T., Ciccoritti, R., & Ferrante, P. (2022). Prediction of strawberries' quality parameters using artificial neural networks. *Agronomy*, 12(4), 963. <https://doi.org/10.3390/agronomy12040963>

References

1. Miković, R., Arsić, B., & Gligorijević, Đ. (2024). Importance of social capital for knowledge acquisition– DeepLIFT learning from international development projects. *Information Processing & Management*, 61(4), 103694. <https://doi.org/10.1016/j.ipm.2024.103694>
2. Rahman, N. H. A., Lee, M. H., Suhartono, & Latif, M. T. (2014). Artificial neural networks and fuzzy time series forecasting: An application to air quality. *Quality & Quantity*, 49(6), 2633–2647. <https://doi.org/10.1007/s11135-014-0132-6>
3. Altmann, A., Tološi, L., Sander, O., & Lengauer, T. (2010). Permutation importance: A corrected feature importance measure. *Bioinformatics*, 26(10), 1340–1347. <https://doi.org/10.1093/bioinformatics/btq134>
4. Nirmalraj, S., Antony, A. S. M., Srideviponmalar, P., Oliver, A. S., Velmurugan, K. J., Elanangai, V., & Nagarajan, G. (2023). Permutation feature importance-based fusion techniques for diabetes prediction. *Soft Computing*. <https://doi.org/10.1007/s00500-023-08041-y>
5. Tan, C. H., Yusof, K. M., & Alwi, S. R. W. (2022). Quality prediction for polypropylene extrusion based on neural networks. *IOP Conference Series: Materials Science and Engineering*, 1257(1), 012034. <https://doi.org/10.1088/1757-899X/1257/1/012034>
6. Tan, J. S. (2022). Ablation study on feature group importance for automated essay scoring. *Asia-Pacific Journal of Information Technology and Multimedia*, 11(1), 90–101. <https://doi.org/10.17576/apjitm-2022-1101-08>
7. Thwin, M. M. T., & Quah, T.-S. (2005). Application of neural networks for software quality prediction using object-oriented metrics. *Journal of Systems and Software*, 76(2), 147–156. <https://doi.org/10.1016/j.jss.2004.05.001>
8. Schweidtmann, A. M., Rittig, J. G., König, A., Grohe, M., Mitsos, A., & Dahmen, M. (2020). Graph neural networks for prediction of fuel ignition quality. *Energy & Fuels*, 34(9), 11395–11407. <https://doi.org/10.1021/acs.energyfuels.0c01533>
9. Poligné, I., Broyart, B., Trystram, G., & Collignan, A. (2002). Prediction of mass-transfer kinetics and product quality changes during a dehydration-impregnation-soaking process using artificial neural networks. Application to pork curing. *LWT – Food Science and Technology*, 35(8), 748–756.
10. Wang, S., & Zhang, Y. (2023). Grad-CAM: Understanding AI models. *Computers, Materials & Continua*, 76(2), 1321–1324. <https://doi.org/10.32604/cmc.2023.041419>
11. Ni, H., & Yin, H. (2009). Exchange rate prediction using hybrid neural networks and trading indicators. *Neurocomputing*, 72(13–15), 2815–2823. <https://doi.org/10.1016/j.neucom.2008.09.023>
12. Kaneko, H. (2022). Cross-validated permutation feature importance considering correlation between features. *Analytical Science Advances*. <https://doi.org/10.1002/ansa.202200018>

13. Jha, K. N., & Chockalingam, C. T. (2009). Prediction of quality performance using artificial neural networks. *Journal of Advances in Management Research*, 6(1), 70–86. <https://doi.org/10.1108/09727980910972172>
14. Fumagalli, F., Muschalik, M., Hüllermeier, E., & Hammer, B. (2023). Incremental permutation feature importance (iPFI): Towards online explanations on data streams. *Machine Learning*. <https://doi.org/10.1007/s10994-023-06385-y>
15. Ejrnæs, R., Aude, E., Nygaard, B., & Münier, B. (2002). Prediction of habitat quality using ordination and neural networks. *Ecological Applications*, 12(4), 1180–1187. [https://doi.org/10.1890/1051-0761\(2002\)012\[1180:POHQUN\]2.0.CO;2](https://doi.org/10.1890/1051-0761(2002)012[1180:POHQUN]2.0.CO;2)
16. Ghanmi, H., Ghith, A., & Benameur, T. (2021). Ring spun yarn quality prediction using hybrid neural networks. *The Journal of the Textile Institute*, 1–9. <https://doi.org/10.1080/00405000.2021.2022826>
17. Amoriello, T., Ciccoritti, R., & Ferrante, P. (2022). Prediction of strawberries' quality parameters using artificial neural networks. *Agronomy*, 12(4), 963. <https://doi.org/10.3390/agronomy12040963>

Надійшла до редакції: 21.06.2026

Прийнята до друку: 15.09.2026

Опубліковано: 30.09.2026