Київський національний університет імені Тараса Шевченка
Факультет радіофізики, електроніки та комп’ютерних систем
Кафедра комп’ютерної інженерії

Випускна кваліфікаційна робота
«Прогнозування викидів шкідливих сполук CO та NOx з газових турбін»

Виконав студент 4 курсу
спеціальності 123 «Комп’ютерна інженерія»
Антон Мацишин

_____

Науковий керівник,
к. ф.-м. наук, асистент
Андрій КОНОВАЛОВ

_____

Рецензент

_____

До захисту допускаю

Завідувач кафедри
к.ф.-м.н., доцент
Юрій БОЙКО

Київ 2022

Реферат

Дипломна робота: 44 сторінки, 16 рисунків, 2 таблиці, 3 додатки, 13 джерел.

Мета роботи - це побудова та оптимізація моделі регресії на основі багатошарового персептрона для прогнозування кількості викидів CO та NOx з газових турбін.

Дослідження кращих моделей регресії та відповідні розробки у сфері систем прогнозування кількості викидів надають можливість відмовитися від складних у налаштуванні та технічному обслуговуванні систем немерервного моніторингу на ТЕС.

У першому розділі проведено огляд інсуючих наукових робіт у цій сфері. Розглянуто моделі та підходи, що ще не використовувалися для прогнозування кількості шкідливих викидів CO та NOx, обрано метрики якості прогнозування.

У другому розділі виизначено методологію розробки та оптимізації моделей регресії, оптимізовано архітектуру нейронної мережі та її гіперпараметри. Також проведено роботу із набором даних, його розділення відповідно до задач оптимізації та тестування моделі.

У третьому розділі проведено оцінку якості прогнозування оптимізованих нейронних мереж, досліджено вплив кількості попередніх прикладів, ознаки яких використовуються для прогнозування, на метрики якості моделі регресії. Результати прогнозування найкращих моделей було порівняно із результатами авторів набору даних.

Зміст

Умовні позначення

ТЕС - теплові електростанції

CEMS - системи неперервного моніторингу кількості викидів

PEMS - системи прогрозування кількості викидів

MAE - mean absolute error

Вступ

Наше суспільство успішно росте і розвивається, з кожним днем потребуючи все більше і більше електроенергії. Лише за останні двадцять років світове виробництво електроенергії виросло більше ніж у 1,6 рази [1].

Незважаючи на зростаючу популярність «зелених» способів виробництва в останні роки, незмінним лідером у цій сфері залишаються теплові електростанції (ТЕС), що постачають близько 59% усієї електроенергії світу [1].

Проблемою у використанні ТЕС (теплові електростанції) є велика кількість забруднювачів повітря, що потрапляють до атмосфери під час згорання палива. Сполуки CO та NOx вважаються головними забруднювачами атмосфери. Саме ці речовини спричиняють фотохімічний смог, знищення озонового шару, кислотні дощі і, як наслідок, глобальне потепління [2]. До того ж, оксиди карбону та нітрогену (CO та NOx) є шкідливими для здоров'я не тільки людини, а і більшості живих організмів на нашій планеті.

Саме через це сьогоднішнє суспільство приділяє увагу контролю і обмеженню викидів CO та NOx у атмосферу. Наприклад, Європейський Союз документами Large Combustion Plant Directive та Industrial Emission Directive забовзязує усі ТЕС з потужністю більше 50МВт заміряти викиди CO та NO_x в реальному часі; обмежувати їх кількість до 25 ppmdv (кількість мільйонних долей на одиницю сухого об'єму) [3].

Увага суспільства до проблем забрудювачів атмосфери та способів їх вимірювання стимулює наукові дослідження і розробки у сфері систем моніторингу кількості викидів.

У даній роботі демонструється процес розробки системи прогрозування кількості викидів газових турбін використовуючи технології машинного навчання. За основу було взято модель багатошарового персептрону, реалізовану засобами бібліотеки sklearn на мові програмування Python.

Розділ 1. Огляд предметної області

1.1 Існуючі способи виміру кількості викидів на ТЕС

На сьогодні існують три різні підходи замірювання кількості викидів CO та NOx [4]:

a) Періодичні заміри. Виконуються у спеціальних лабораторіях за отриманими зразками повітря поблизу ТЕС. Даний способ недійсний для ТЕС з потожністю більше 50МВТ.
b) Системи неперервного моніторингу кількості викидів (CEMS - continuous emissions monitoring system). Такі системи встановлюються на самих ТЕС та потребують регулярного догляду для забезпечення достатньої точності вимірювань.
c) Системи прогрозування кількості викидів (PEMS - predictive emission monitoring systems). Ідея PEMS полягає у прогнозуванні кількості викидів за вхідними параметрами турбіни. тому, щоб за вхідними параметрами, що заміряються на турбіні (наприклад, тиск на виході турбіни, температура на вході турбіни) надати прогноз щодо кількості викидів шкідливих речовин.

1.2 Системи прогнозування кількості викидів

Системи прогнозування кількості викидів – це системи, які, використовуючи математичні та статистичні моделі, аналізують параметри, наприклад, газової турбіни для прогнозування кількості викидів тих чи інших забруднювачів у атмосферу [5].

Незважаючи на відсутність єдиного стандарту, який дозволяв би використовувати дані про викиди, спрогнозовані PEMS, як офіційні, зацікавленість у цій технології з кожним роком лише зростає. На відміну від

CEMS, які потребують регулярного догляду, перевірки та доналаштувань, PEMS налаштовується лише один раз, на початку роботи; час використання цієї технології необмежений.

На зараз PEMS використовуються як альтернатива і резерв систем неперервного моніторингу. Як у наукових дослідженнях, так і на практиці, такі системи частіше за все використовуються для передбачення кількості NO_x, виділеного у процесі згорання палива (хоча існують і моделі, навчені передбачувати кількість інших забруднювачів також).

PEMS має над CEMS наступні переваги:

a) Капітальні витрати. Оскільки PEMS для навчання використовує дані, що вже були зібрані CEMS, початкова вартість встановлення такої системи набагато нижча [6].
b) Час встановлення. CEMS має електричні, механічні та електронні компоненти, на встановлення яких потрібен час. Зазвичай це становить близько 3-4 місяців. Для роботи PEMS потрібен лише 1 комп'ютер з усім необхідним програмним забезпеченням. Налаштування такої системи проходить швидше ніж за день.
c) Технічне обслуговування. PEMS потребує лише сезонного очищення комп'ютера, який використовується для обробки даних. CEMS в свою чергу має електричні та механічні частини, які потребують регулярного та висококваліфікованого обслуговування.
d) Калібрувальні гази. На відміну від PEMS, для роботи якої достатньо лише вхідних даних, CEMS для коректної роботи необхідні калібрувальні гази. Це зразки речовин, заміри яких CEMS виконує. Зберігання та таких речовин потребує додаткового часу і витрат.

е) Відмовостійкість. CEMS – це hardware інструмент, що містить у собі велику кількість різних компонентів. Якщо хоча б один із них вийде із ладу – система не зможе аналізувати дані в реальному часі, і виміри буде втрачено. PEMS в свою чергу не має такої проблеми: у разі збою програми дані про викиди буде відтворено, за наявності збережених вхідних даних.

1.3 Аналіз існуючих наукових робіт за тематикою PEMS

Ідея прогнозування кількості шкідливих викидів за вхідними параметрами системи не є революційною. Наприклад, у 1999 було розроблено нейронну мережу для прогнозування викидів HC, NOx і CO що виділяються під час згорання палива у двигуні [7].

У 2013 році в університеті Люблину було використано лінійні моделі (модель авторегресії, модель авторегресії ковзного середнього та АРКС) та нелінійну (метод опорних векторів) для прогнозування кількості викидів NOx на турбінах з вугільним типом палива [8]. Протягом проведення роботи було виявлено, що нелінійні моделі не надають більшої точності порівняно з лінійними.

У 2016 році для прогнозування викидів забруднюючих речовин із турбін з вугільним типом палива було використано модель карти самоорганізації [9].

Однією з найновіших робіт у сфері розробки PEMS є робота працівників університету імені Наміка Кемаля, 2019 року. У даній роботі було використано ансамбль нейронних мереж типу “екстремальні машини навчання” (extreme learning machine, ELM) для прогнозування кількості шкідливих викидів [4].

Слід зазначити, що у даній роботі були отримані високі показники точності при прогнозуванні кількості викидів CO та NOx, як і у випадку моделі

карт самоорганізації [9]. Цей факт дає підстави для гіпотези, що саме нейронні мережі підходять для подібного типу задач.

Модель багатошарового персептрону ще не використовувалася у науковій спільноті для прогнозування кількості викидів із газових турбін. А саме у цього підходу є потенціал у настільки комплексній задачі: за універсальною теоремою апроксимації Хорника для будь-якої неперервної функції *f(x)* існує нейронна мережа, що апроксимує функцію *f(x)* із заданою точністю. Для отримання заданої точності необхідно визначити оптимальну архітектуру мережі та підібрати синаптичні ваги.

1.4 Нейронні мережі

Нейронні мережі – це математична модель, що використовується для широкого спектру аналітичних задач, побудована за принципом організації та функціонування біологічних нейронних мереж.

Зважаючи на те, що біологічні нейронні мережі – це занадто комплексна модель, математична модель ґрунтується на наступних припущеннях:

a) Передача сигналів у мережі є синхронізованою. Тобто для будь-якого з’єднання між нейронами час розповсюдження сигналу повинен бути однаковий.
b) Для кожного нейрона визначена деяка функція, що визначає вихідний сигнал в залежності від вхідного.
c) Синапси використовуються для зв'язку між нейронами і перемножують отриманий сигнал на деяке число – вагу синапсу.

Той факт, ваги синапсів можуть змінюватися у часі означає різну реакцію нейрона на один і той же вхідний сигнал. Що, по суті, і є навчанням.

1.5 Персептрон

Персептрон – математична або комп'ютерна модель, запропонована Френком Розенблатом у 1957 році, реалізована у 1960 році як ЕОМ «Марк-1».

Розглянемо випадок одношарової нейронної мережі як окремий персептрон (рис. 1.1).

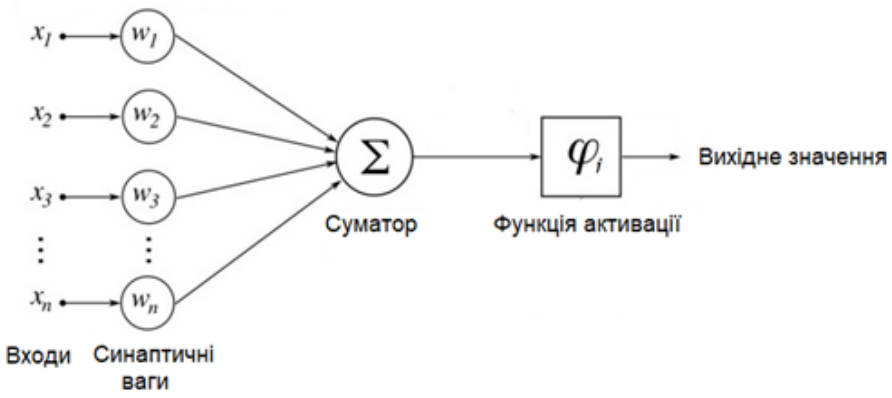

Рис. 1.1 Схематичне зображення нейрона.

На рис. 1.1 $[x_1, \dots, x_n]$ – сигнали, які подаються на вхід нейрона, $[w_1, \dots, w_n]$ – ваги синапсів (що можуть коригуватися) [10, стор. 12-15].

Суматор складає добутки входів і синаптичних ваг (b – значення зміщення):

$$S(x, w) = \sum_{i=1}^{n} w_i \cdot x_i + b \quad .$$

Результат функції S є єдиним аргументом функції активації, що є нелінійною неперервною функцією, яка і повертає прогнозований результат:

$$Y = \varphi(S(w, x)).$$

Як зрозуміло із формули, прогнозування персептрону однозначно визначається сигналами на входах та вагами синапсів.

1.6.1 Функції активації

Існує велика кількість функцій, які використовуються як функції активації. Найчастіше ці функції є нелінійними (рис 1.2). Також у випадку нейронних мереж з більше ніж одним нейроном можуть використовуватися різні функції активації для різних груп нейронів. Нижче наведені найпоширеніші функції активації:

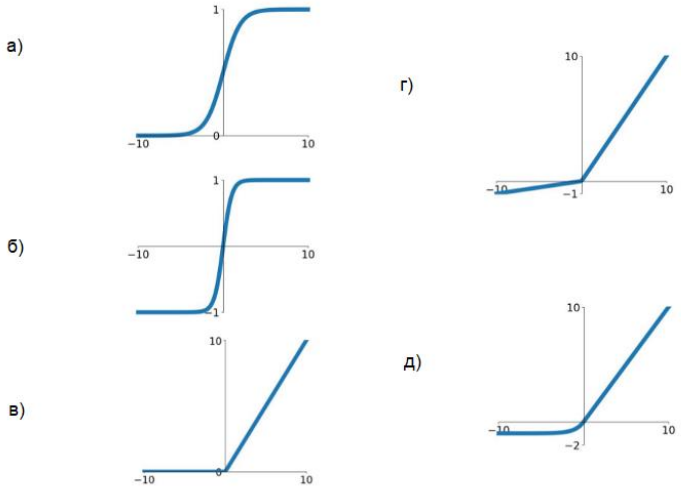

Рис. 1.2. Приклад функцій активації: а – сигмоід; б – гіперболічний тангенс; в – функція лінійного випрямлення (ReLU); г – Leaky ReLU; д – ELU.

1.6.2 Багатошаровий персептрон

Багатошаровий персептрон – це нейронна мережа прямого розповсюдження. Вхідний сигнал в такій мережі поширюється у прямому напрямку, від шару до шару [10, стор. 12].

Багатошаровий персептрон має наступну архітектуру:

a) один шар вхідних елементів, на які подаються вхідні сигнали,
b) один або більше прихованих шарів нейронів,
c) один шар вихідних нейронів (вихідні вузли).

Багатошарові персептрони успішно використовуються для вирішення як задач класифікації, так і задач регресії.

Багатошарові персептрони (рис. 1.3) мають ряд відмінних ознак:

a) Нелінійні функції активації. Для кожного нейрону прихованого шару в мережі функція активації повинна бути нелінійною, а також диференційованою. Наприклад, гіперболічний тангенс.
b) Наявність прихованого шару. Багатошаровий персептрон має містити як мінімум один прихований шар нейронів. Саме ці нейрони дозволяють мережі навчатися вирішенню складних задач.
c) Висока зв'язність. Мережа повинна мати високу синаптичну зв'язність. Окремим випадком високої синаптичної зв'язності є повна зв'язність: вихідні значення нейронів *n – 1* шару подаються на усі нейрони *n* шару.

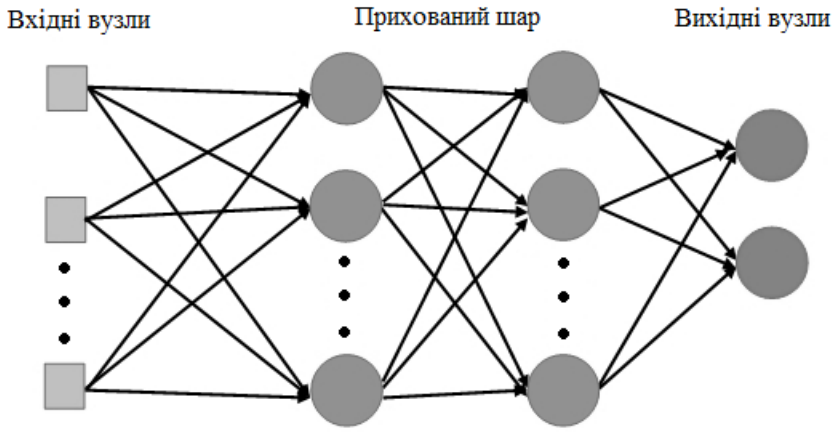

Рис. 1.3. Приклад архітектури багатошарового персептрона

1.6.3 Методи оптимізації параметрів нейронної мережі

Одним із методів оптимізації параметрів нейронної мережі є використання функції помилки (або функції втрат) *Q(w)*. Ця функція залежить від синаптичних ваг та вхідних сигналів і визначає похибку, яку має нейронна

мережа при прогнозуванні [10, стор. 38-39]. Ось деякі приклади функції помилки:

$$Q(w) = \sum_{i=1}^{N} |a(x_i, w) - y_i|(\quad)\quad|,$$

$$Q(w) = \sum_{i=1}^{N} (a(x_i, w) - y_i)^2(\quad)\quad,$$

N – розмір вибірки, x_i – і-тий приклад вибірки, y_i – очікуваний результат.

Основною задачею при розробці нейронної мережі є пошук таких синаптичних ваг, при яких значення функції помилки буде мінімальним.

1.7 Метрики якості

Обрано дві метрики якості: коефіцієнт детермінації R^2 та середня абсолютна помилка MAE (mean absolute error). Дані метрики демонструють наскільки точно модель апроксимує вхідний набір даних.

Формулу R^2 записують наступним чином:

$$R^2 = 1 - \frac{\sigma^2}{\sigma_y^2},$$

де σ_y^2 – дисперсія концентрації шкідливих викидів, σ^2 – середньоквадратичне відхилення прогнозованої концентрації викидів. R^2 набуває значень у діапазоні [-1, 1].

Нижче наведена формула MAE:

$$MAE = \frac{\sum_1^N (y_i - y_{0i})}{N},$$

де y_i - реальне значення прогнозованої величини i-го прикладу, y_{0i} - спрогнозоване значення, N - кількість екземплярів.

1.8 Аналіз даних

Набір даних, що використовувався для навчання [4], був зібраний протягом 5 років (з 2011 по 2015 роки включно). Усього у наборі представлено близько 36 000 прикладів.

Кожен приклад містить у собі 9 вхідних ознак та 2 прогнозовані величини (густина CO та NOx). У таблиці 1.1 наведено назви ознак та їх розмірність.

Таблиця 1.1. Вхідні ознаки та прогнозовані величини набору даних [4].

Назва ознаки	Абревіатура	Розмірність
Температура навколишнього середовища	AT	◦C
Тиск	AP	mbar
Вологість	AH	%
Різниця тиску на повітряному фільтрі	AFDP	mbar
Тиск вихлопних газів турбіни	GTEP	mbar
Температура на вході турбіни	TIT	◦C
Темпереτура на виходітурбіни	TAT	◦C
Тиск розряду компресора	CDP	mbar
Кількість виробленої енергії	TEY	MWH
Густина оксиду карбону	**CO**	mg/m3
Густина оксиду нітрогену	**NOx**	mg/m3

На рис. 1.4 показано розташування датчиків вимірювання параметрів турбіни, погодних умов та концентрації CO і NOx у газовій турбіні.

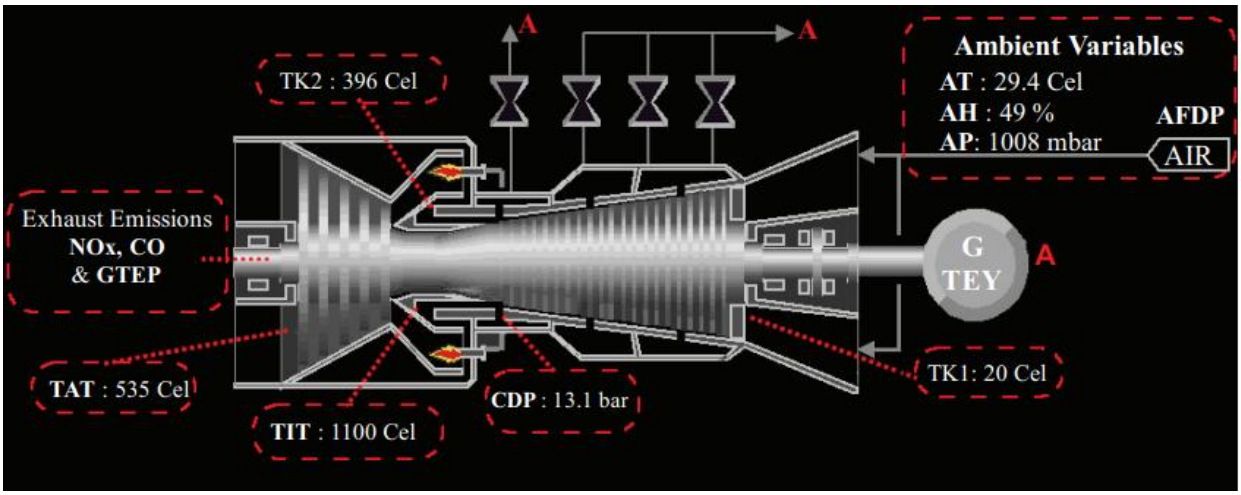

Рис. 1.4. Розташування датчиків вимірювання вхідних ознак та прогнозованих змінних [4].

1.9 Використання ознак у попередні моменти часу

Дані, опрацьовані у роботі [4], були зібрані зі сталим інтервалом у годину. Це дає підстави розглянути доцільність використання значень ознак (характеристик турбіни), виміряних у попередні моменти часу, для прогнозування значень концентрації викидів у поточний момент часу.

У такому підході при прогнозуванні, наприклад, концентрації CO у поточний момент часу, додатково використовуються ознаки і значення концентрації CO за кілька попередніх моментів часу. Кількість таких попередніх моментів часу, що використовується при прогнозуванні, називають вікном ознак. Розмір вікна у цій роботі позначається як m.

У таблиці 1.2 наведено приклад обробки даних у такому підході. Для прогнозування кількості викидів CO у прикладі 3 додатково

використовуються значення ознак і CO із прикладу 1 та прикладу 2. Розмір вікна (m) у даному прикладі дорівнює 2.

Таблиця 1.2. Візуалізація підходу з використанням ознак у попередні моменти часу.

	AT	AP	AH	AFDP	GTEP	TIT	TAT	TEY	CDP	CO
0	4.5878	1018.7	83.675	3.5758	23.979	1086.2	549.83	134.67	11.898	0.32663
1	4.2932	1018.3	84.235	3.5709	23.951	1086.1	550.05	134.67	11.892	0.44784
2	3.9045	1018.4	84.858	3.5828	23.990	1086.5	550.19	135.10	12.042	0.45144
3	3.7436	1018.3	85.434	3.5808	23.911	1086.5	550.17	135.03	11.990	0.23107
4	3.7516	1017.8	85.182	3.5781	23.917	1085.9	550.00	134.67	11.910	0.26747

Варто зазначити, що такий підхід не використовувався у роботі авторів даного набору даних [4].

1.10 Постановка задачі

Метою випускної кваліфікаційної роботи є побудова та оптимізація моделі регресії на основі багатошарового персептрона для прогнозування кількості викидів CO та NOx з газових турбін.

Для досягнення поставленої мети виконувалися такі задачі:

1. Побудова моделі регресора для прогнозування кількості викидів CO та NOx на основі поточних значень вхідних ознак стану газової турбіни і значень цих ознак у m попередніх моментів часу.
2. Оптимізація параметрів побудованої моделі для максимізації точності прогнозування.
3. Обчислення метрик якості моделей на тестовій вибірці і їх порівняльний аналіз з літературними даними.

Розділ 2. Методи досліджень

2.1 Засоби розробки програмної моделі прогнозування викидів

Мовою розробки моделі прогнозування було обрано Python. Це інтерпретована мова програмування високого рівня загального призначання зі строгою динамічною типізацією. Нижче наведені її переваги:

1. Python – це сучасна мова програмування, стандарти якої постійно оновлюються. Це дозволяє використовувати у розробці найновіші підходи та практики.
2. Велика кількість бібліотек для обробки даних та машинного навчання, що реалізують більшість необхідних методів для розробки нейронних мереж. До того ж, через широку популярність використання, саме бібліотеки на Python мають одні з кращих документацій.
3. Python зручно використовувати для розв’язання математичних задач. Ця мова програмування може оперувати з комплексними числами, цілими числами довільної велечини, тощо.

Бібліотекою для машинного навчання було обрано scikit-learn. Scikit-learn – це колекція одних з найефективніших інструментів для статистичного моделювання та машинного навчання [11].

Одними з головних переваг даного інструменту є:

1. Велика ступінь інкапсуляції методів машинного навчання і, як наслідок, простота використання. У користувача бібліотеки немає необхідності розуміти логіку низького рівня, scikit-learn надає абстрактні інтерфейси для роботи з моделями машинного навчання.

2. Універсальність використання. Scikit-learn постачає велику кількість інструментів для проектування нейронних мереж та моделей машинного навчання, що можуть бути застосовані для широкого спектру цілей.

До того ж, scikit-learn розповсюджується під ліцензією BSD, що робить цю бібліотеку повністю безкоштовною для некомерційного використання.

Середовищем розробки було обрано платформу Google Colaboratory. Це безкоштовний хмарний сервіс, що дозволяє запускати скрипти Python із браузера.

Google Colaboratory працює на базі процесора Xenon з одним ядром, двома потоками та тактовою частотою 3 ГГц. Під обчислення виділено 12 ГБ (можна збільшити до 25 ГБ) оперативної пам’яті DDR5 та 33 ГБ пам’яті на диску.

Також програма може використовувати графічний процесор – і це головна особливість даного сервісу. Google Colaboratory надає доступ до GPU Tesla K80 з 2496 ядрами та 12 ГБ пам’яті GDDR5 [12].

2.2 Архітектура багатошарового персептрона

У цій роботі розробляється підхід для прогнозування двох величин: CO та NOx. Вирішити таку задачу можна наступним чином:

1. Розробити одну модель багатошарового персептрона з двома виходами.
2. Розробити дві окремі моделі.

Було обрано другий варіант, так як він дозволяє використовувати різні налаштування нейронної мережі для прогнозування різних величин.

Дослідження було вирішено проводити на персептроні з одним прихованим шаром. Кількість нейронів у прихованому шарі змінювалася від 1

до 10. Така архітектура моделі була обрана для боротьби з перенавчанням, яке властиве моделям з більшою кількістю прихованих шарів на нейронів у них.

Функцією активації було обрано ReLu:

$$f(x) = max\ (0, x).$$

Нейромережі з функцією активації ReLu швидше навчаються в тому числі і через спрощений обрахунок градієнту (0 або 1 в залежності від знаку x).

2.3 Метод ранньої зупинки

Метод ранньої зупинки – це одна з форм регуляризації, метою якої є боротьба із явищем перенавчання.

Перенавчання – це явище, при якому модель апроксимує навчальні дані з високою точністю, а тестові – з дуже низькою [10, стор. 38]. Причиною перенавчання може бути зависока складність моделі або занадто велика кількість ітерацій навчання.

У методі ранньої зупинки з навчальних даних виділяється деяка вибірка для додаткової перевірки метрики якості (валідаційна вибірка). Коли метрика якості перестає покращуватись (рис. 2.1) на валідаційній вибірці деяку кількість ітерацій (n_iter_no_change), метод ранньої зупинки зупиняє навчання моделі та відтворює ваги, при яких метрика якості була найкращою.

У роботі використовувався метод ранньої зупинки за параметром n_iter_no_change = 100 (оптимальне значення визначене методом підбору). На валідаційну вибірку для CO було виділено 10% тестової вибірки, для NO - 30%.

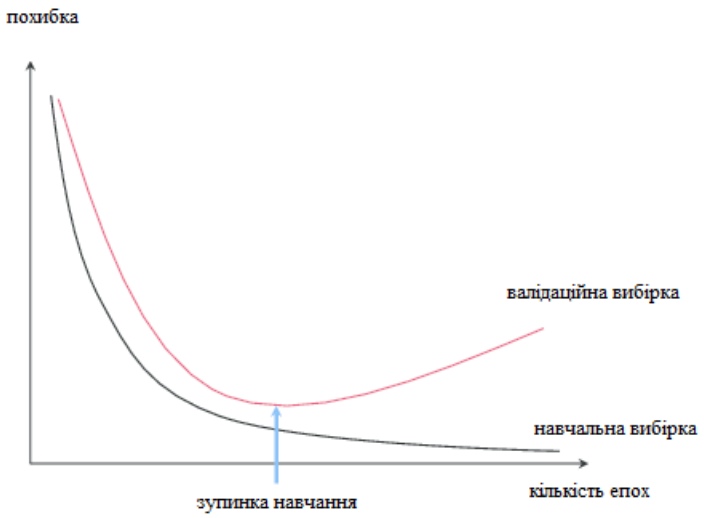

Рис. 2.1. Візуалізація підходу ранньої зупинки

2.4 Метод пониження ваг

Одним із симптомів перенавчання є великі абсолютні значення вагових коефіцієнтів. Це дає підстави ввести у модель штраф за великі значення синаптичних ваг. У такому випадку при навчанні мережі наше завдання не тільки мінімізувати функцію помилки *Q(W)*, а і функцію *R(W)*, що називають регуляризатором:

$$Q(W) + \alpha \cdot R(W) \to min.$$

Параметр регуляляризації α характеризує значущість штрафу *R(W)* відносно доданку *Q(W)*.

Існують наступні типи регуляризаторів:

1. L_1-регуляризатор, lasso-регуляризатор:

$$R(W) = L_1(W) = \|W\|_1 = \sum_{i=0}^{K} |w_i|.$$

Цей регуляризатор не є гладким, але успішно використовується у великій кількості чисельних градієнтних методів. Після lasso-регуляризації деяків ваги можуть дорівнювати 0.

2. L_2-регуляризатор, квадратичний, або ridge:

$$R(W) = L_2(W) = \|W\|^2 = \sum_{i=0}^{K} w_i{}^2.$$

Цей регуляризатор є гладким, випуклим, його можна диференціювати. Як наслідок – L_2-регуляризатор зручно використовувати у аналітичних виразах.

У роботі використовувався L_2-регуляризатор з коефіцієнтом регуляризації α = 0,1. Значення коефіцієнту підбиралося експериментально.

2.5 Оптимізація архітектури нейронної мережі та її гіперпараметрів

Основна задача при розробці нейронної мережі - це оптимізація її архітектури та гіперпараметрів для покращення точності моделі. Головна складність цього процесу полягає у великій кількості гіперпараметрів та їх нелінійній залежності між собою. Бібліотека scikit-learn надає інструмент автоматизації цього процесу - GridSearchCV.

За допомогою цього інструменту проводилася оптимізація кількості нейронів прихованого шару та гіперпатаметру learning_rate_init.

2.5.1 Перевірка оптимізованих моделей за допомогою крос-валідації

Моделі, оптимізовані за допомогою GridSearchCV, додатково перевірялися методом крос-валідації із розділенням даних TimeSeriesSplit (зі значенням cv 5) задля унеможливення “підгону” гіперпараметрів до валідаційної вибірки.

У такому підході відбувається cv ітерацій навчання (рис. 2.2) на тестовій вибірці та перевірок на валідаційній вибірці. При цьому зберігається порядок прикладів (на відміну від роботи [4]), що важливо при використанні ознак у m попередніх моментів часу.

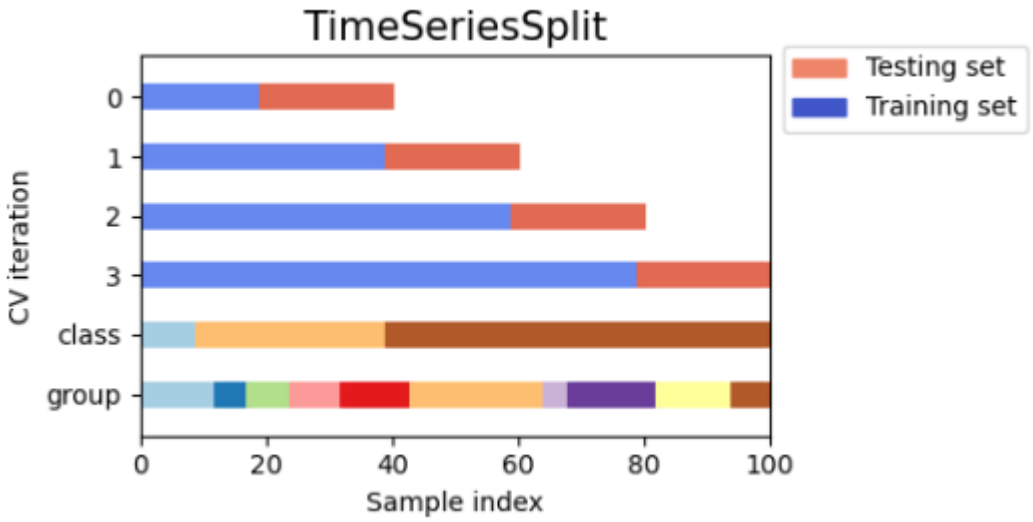

Рис. 2.2. Розділення тестого і валідаційного набору даних при використанні TimeSeriesSplit [13].

2.6 Робота з даними

У роботі використовувався набір даних зібраний протягом 5 років [4], з 2011 по 2015 рік включно. Виміри атрибутів проводилися кожну годину, усього було зібрано близько 36 тисяч об’єктів.

Весь набір даних було розділено на 3 частини. На навчальну вибірку було виділено 2011 та 2012 роки, на валідаційну - 2013 рік. Останні два роки було використано як тестову вибірку.

Проміжна оцінка якості прогнозування моделей регресії відбувалася на валідаційній вибірці задля унеможливлення “підгону” архітектури і

гіперпараметрів нейронної мережі під тестову вибірку. На тестовій вибірці відбувалося оцінка точності прогнозування вже оптимізованої моделі.

Розділ 3. Результати досліджень

У цьому розділі наведено результати прогнозування моделей регресії кількості викидів CO та NOx з різними значеннями параметру розміру вікна вхідних ознак m.

3.1 Результати прогнозування CO.

Проведемо дослідження залежності метрик якості при прогнозуванні CO від кількості нейронів прихованого шару при різних значеннях параметру m. На рис. 3.1 наведено таку залежність для метрики MAE.

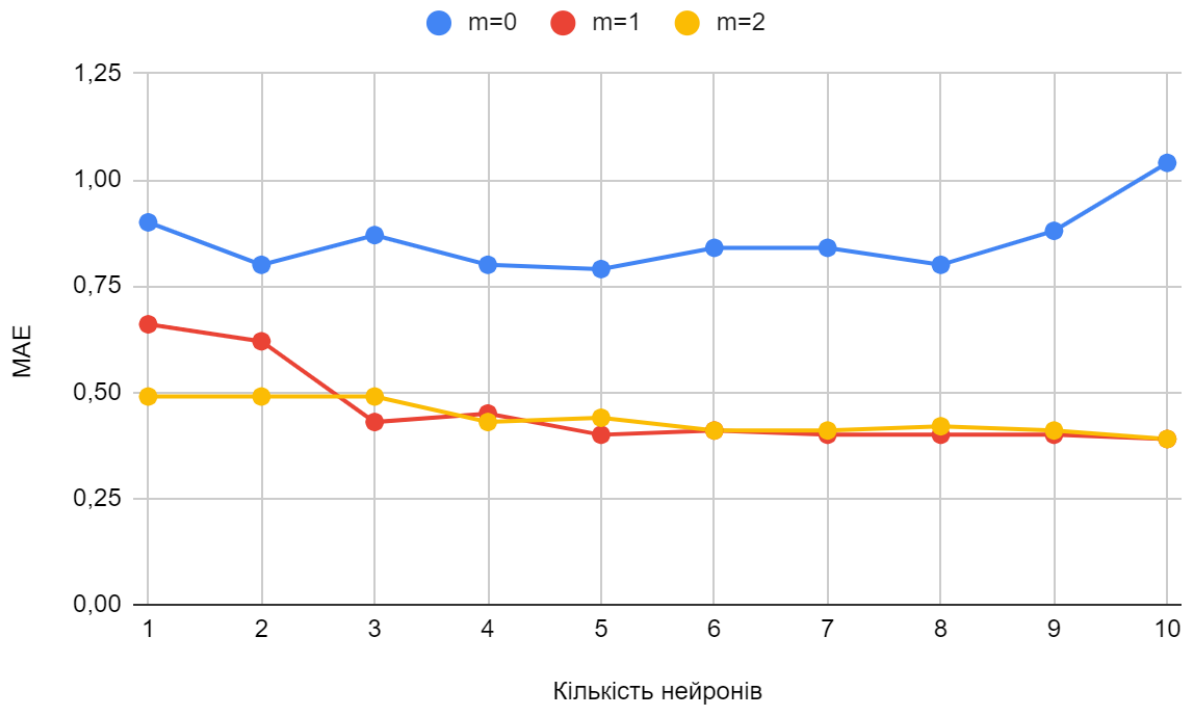

Рис. 3.1. Залежність MAE від кількості нейронів прихованого шару при прогнозуванні CO для різних значеннях параметру m.

Мінімальне значення MAE (0,39) було отримане з 10 нейронами у прихованому шарі при значеннях параметру m=1 та m=2. Без використання

ознак у попередні моменти часу значення MAE при архітектурі з 10 нейронами прихованого шару було максимальним серед вибірки (1,04); саме зазначений підхід дав такий приріст у точності.

На рис. 3.2 наведено аналогічну залежність для метрики R^2.

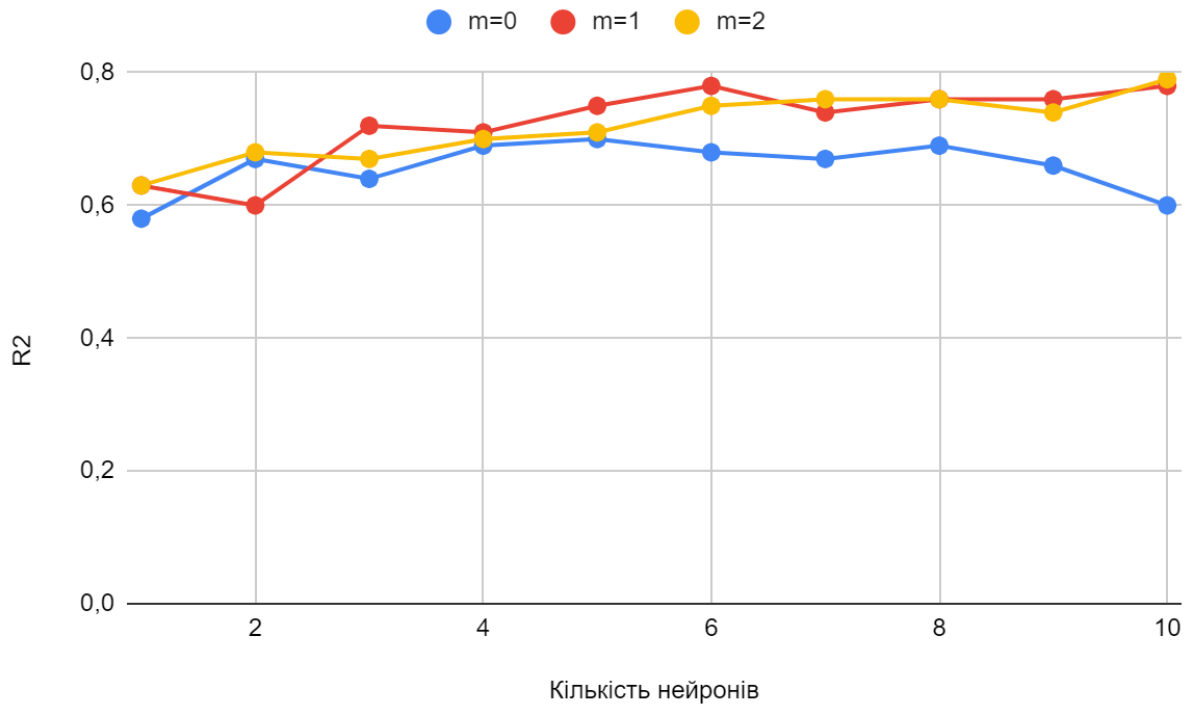

Рис. 3.2. Залежність R^2 від кількості нейронів прихованого шару при прогнозуванні CO для різних значеннях параметру m.

Як і у випадку MAE, найоптимальніше значення R^2 (0,79) було досягнуте при 10 нейронах у прихованому шарі з параметром m=2.

Дослідимо вплив параметру m на точність прогнозування нейронної мережі з 10 нейронами у приховану шарі більш докладно. На рис. 3.3 наведено графік залежності MAE від параметру m при зазначеній архітектурі.

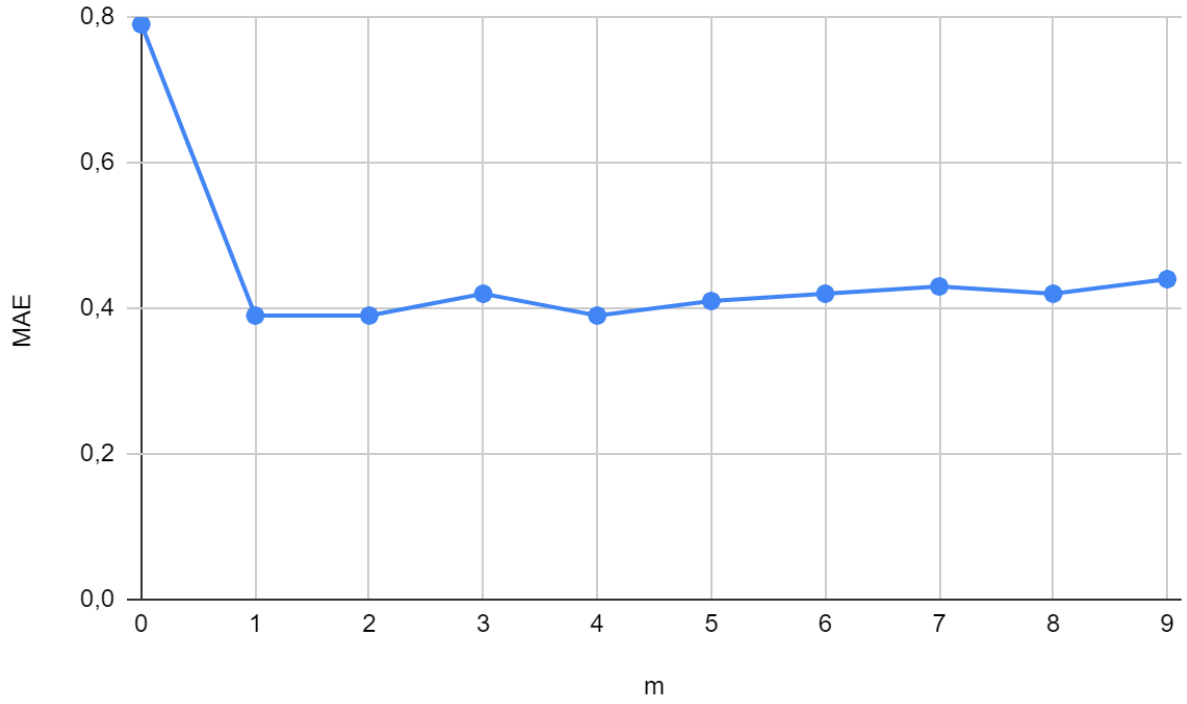

Рис. 3.3. Залежність MAE від параметру m при прогнозуванні CO.

З рис. 3.3 видно, що використання ознак у попередні моменти часу для прогнозування дає великий приріст у точності (при m=0 ознаки попередніх прикладів не використовуються). Мінімальне значення MAE (0,39) досягнуте із m=2. Подальше збільшення параметру m призводить до поступового погіршення метрики MAE.

На рис. 3.4 наведено графік залежності метрики R^2 від параметру m.

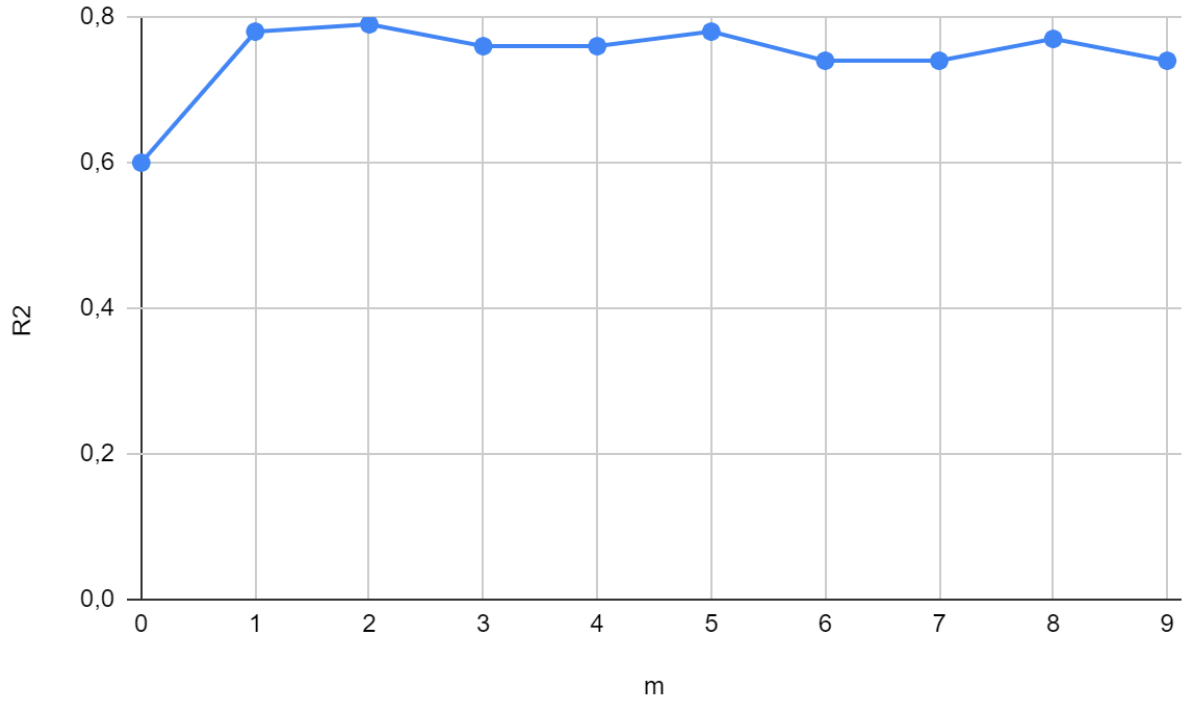

Рис. 3.4. Залежність R^2 від параметру m при прогнозуванні CO.

Найкращі результати прогнозування на валідаційній вибірці отримані з параметром m=2 та 10 нейронами у єдиному прихованому шарі. MAE та R^2дорівнюють 0,39 та 0,79 відповідно.

3.2 Результати прогнозування NOx

Проведемо дослідження залежності метрик якості при прогнозуванні NOx від кількості нейронів прихованого шару при різних значеннях параметру m. На рис. 3.5 наведено таку залежність для метрики MAE.

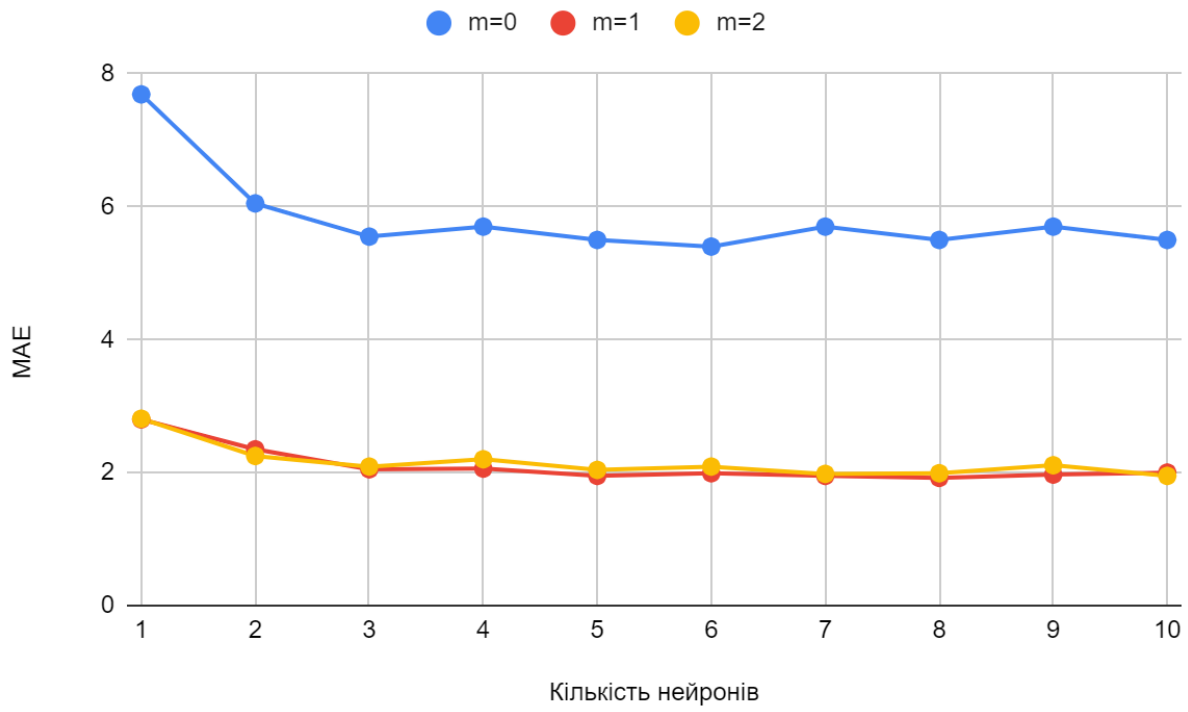

Рис. 3.5. Залежність MAE від кількості нейронів прихованого шару при прогнозуванні NOx для різних значеннях параметру m.

З рис. 3.5 бачимо, що використання ознак прикладів у попередні моменти часу для прогнозування поточного значення NOx кратно зменшує похибку при прогнозуванні кількості викидів NOx. На рис. 3.6 наведено аналогічний графік залежності метрики R^2.

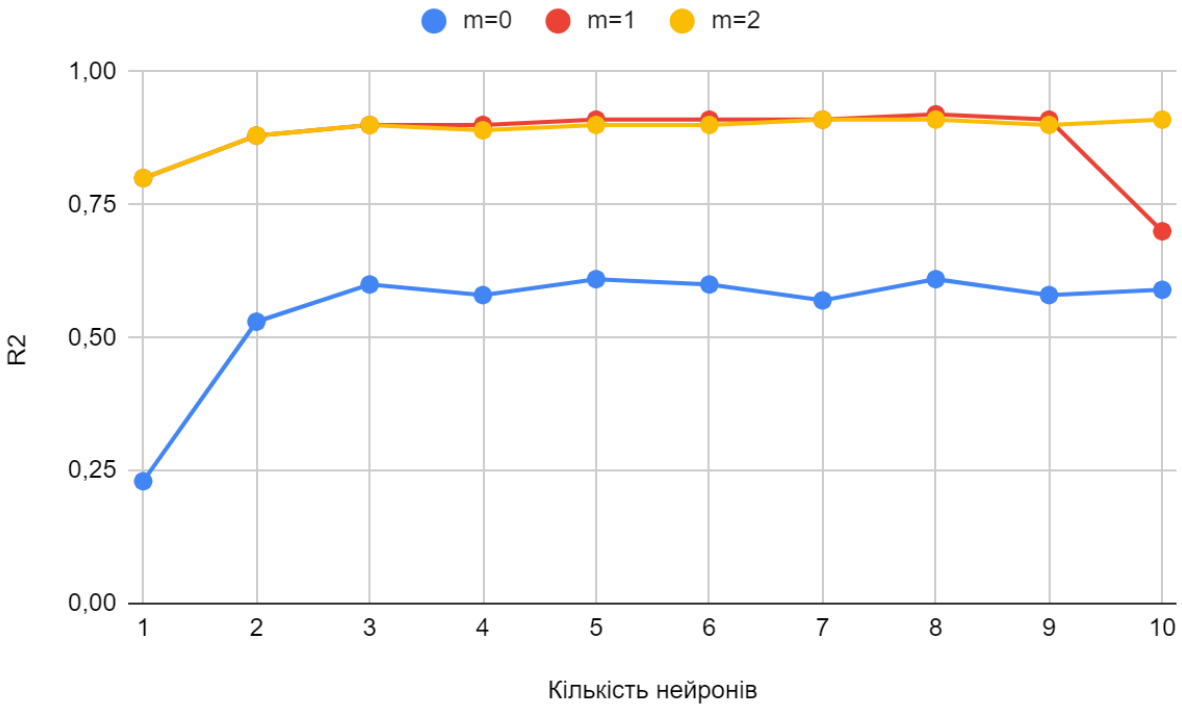

Рис. 3.6. Залежність R^2 від кількості нейронів прихованого шару при прогнозуванні NOx для різних значеннях параметру m.

Найоптимальніших значень метрик було досягнуто з 8 нейронами у прихованому шарі. Метрики MAE та R^2 дорівнюють 1,92 та 0,92 відповідно.

Проведемо дослідження залежності точності прогнозування нейронної мережі із цією архітектурою від параметру m. На рис. 3.7 наведено залежність метрики MAE від параметру m.

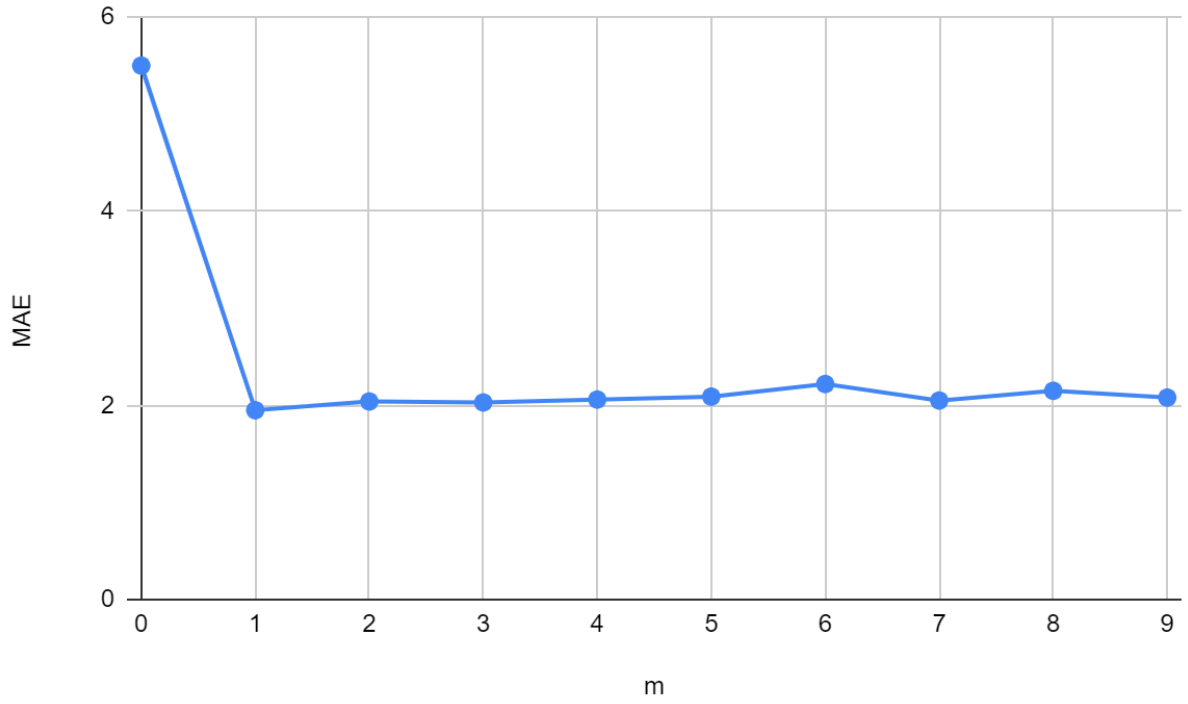

Рис. 3.7. Залежність MAE від параметру m при прогнозуванні NOx.

Як і у випадку CO, використання ознак у попередні моменти часу для прогнозування теперішнього значення NOx дає суттєве покращення точності. Подальше збільшення параметру m поступово погіршує значення MAE. Схожим чином поводить себе і величина R^2 (рис. 3.8). Із початком використання зазначеного підходу (m=1) спостерігається покращення метрики із 0,61 до 0,91. Подальше збільшення m суттєво на метрику R^2 не впливає.

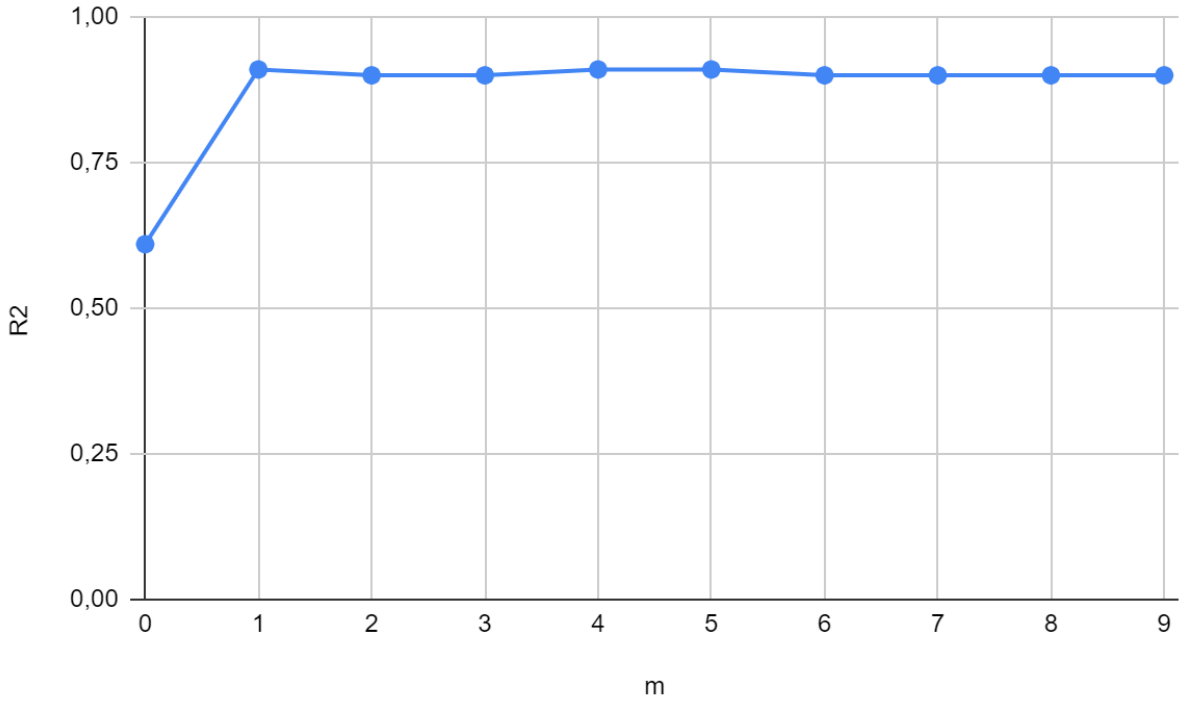

Рис. 3.8. Залежність R^2 від параметру m при прогнозуванні NOx.

Найкращі результати прогнозування NOx на валідаційній вибірці отримані з параметром m=1 та 8 нейронами у єдиному прихованому шарі MAE та R^2 дорівнюють 1,95 та 0,91 відповідно.

3.3. Перевірка оптимізованих моделей на тестовій вибірці.

Після оптимізації архітектури та гіперпараметрів нейронної мережі перевіримо точність прогнозування найкращих моделей на тестовій вибірці. Варто зазначити, що тестова вибірка не використовувалася ані для навчання моделей, ані для підбору їх гіперпараметрів. Це було зроблено задля забезпечення коректності перевірки оптимізованих моделей.

На рис. 3.9 наведено значення метрики MAE для моделей регресії кількості викидів CO та NOx, порівняння їх значень із результатами авторів набору даних [4].

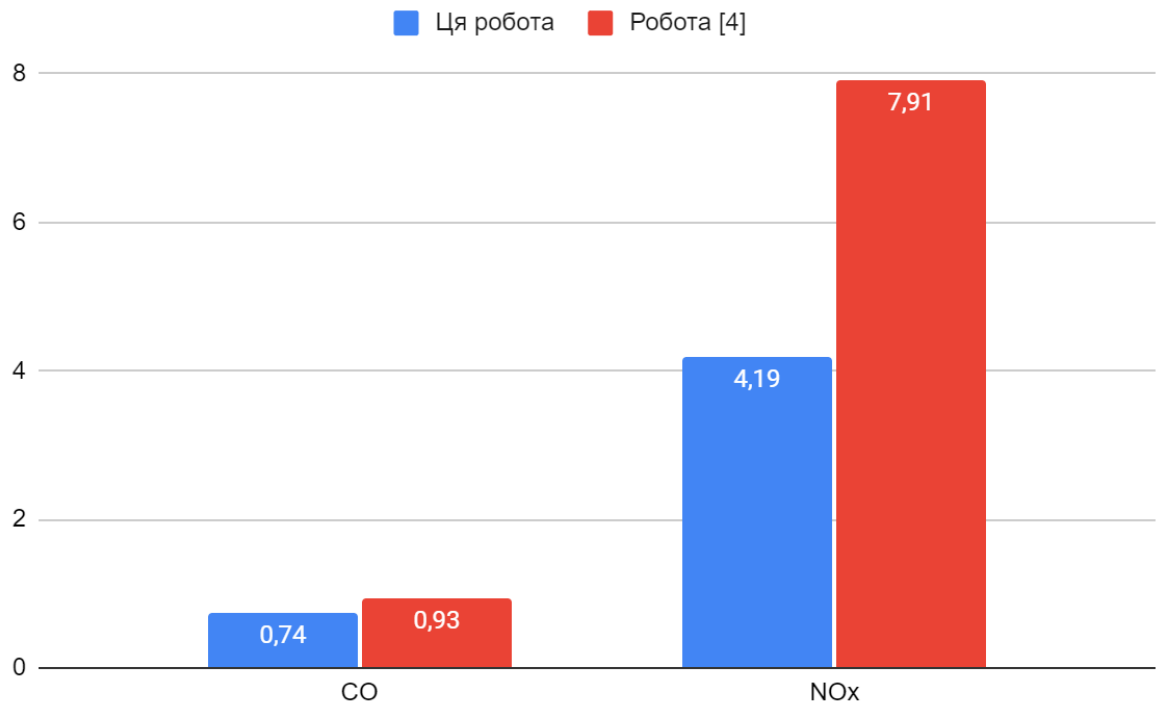

Рис. 3.9. Значення метрики MAE, обраховані на тестовій вибірці.

Спостерігається суттєве покращення метрики MAE для прогнозування обох величин.

На рис. 3.10 наведені значення метрики R^2, отримані на тестовій вибірці, та їх порівняння із значеннями з роботи 2019 року [4].

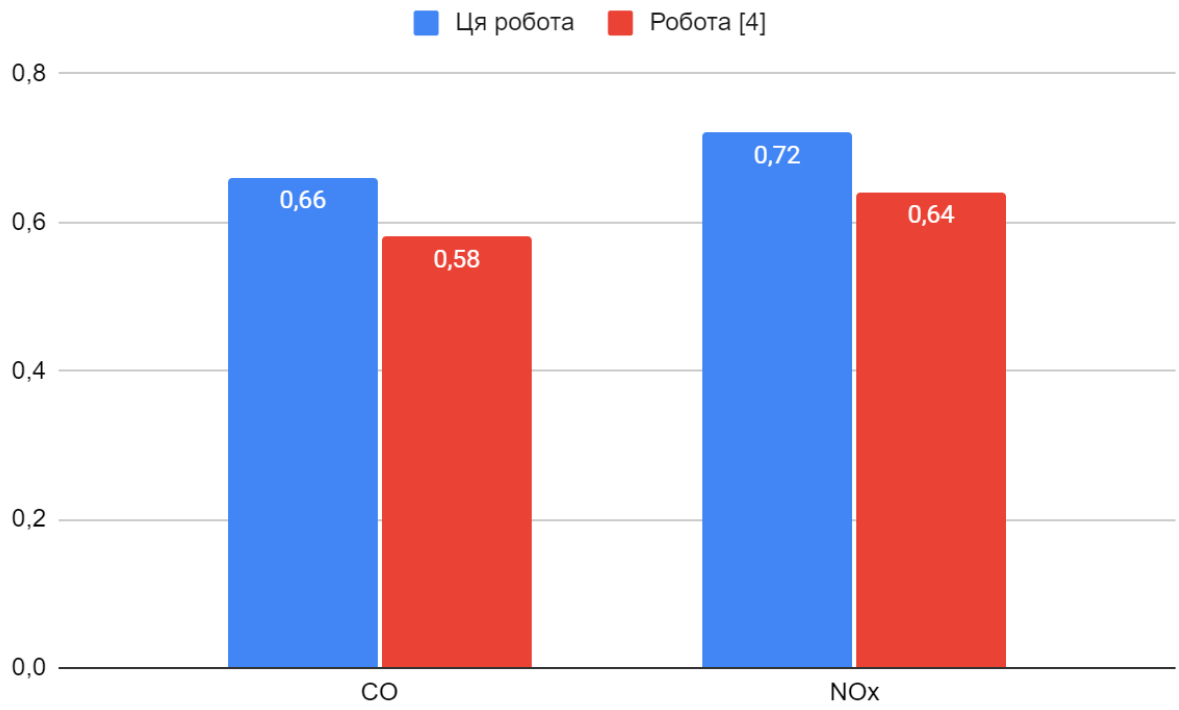

Рис. 3.10. Значення метрики R^2, обраховані на тестовій вибірці.

Як і у випадку MAE, значення R^2 вдалося покращити.

Значення метрик якості, що були отримані на тестовій вибірці, виявилися кращими за значення метрик, що були отримані у роботі авторів набору даних. Такого результату було досягнуто без використання ансамблю нейронних мереж та кількох різних моделей регресії. Кратний приріст точності забезпечило саме використання ознак попередніх екземплярів для прогнозування поточного значення кількості викидів.

Висновки

У результаті виконання випускної кваліфікаційної роботи:

1. Побудовано дві окремі моделі багатошарового персептрона для прогнозування кількості викидів CO і NOx з використанням значень ознак у m попереднім моментів часу.
2. Оптимізовано архітектуру та гіперпараметри нейронної мережі, максимізовано значення метрик якості на валідаційній вибірці.
3. Визначені метрики якості прогнозування оптимізованих моделей MAE і R^2 на тестовій вибірці, значення яких перевищують літературні дані.

Перелік посилань

1. Electricity Mix [Електронний ресурс] // OurWorldInData. – 2020. – Режим доступу до ресурсу: https://ourworldindata.org/electricity-mix (дата звернення 06.11.2021).

2. Skalska K, Miller JS, Ledakowicz S. Trends in NOx abatement: a review, 2010.

3. Fichet V, Kanniche M, Plion P, Gicquel O. A reactor network model for predicting NOx emissions in gas turbines, 2010.

4. Heysem Kaya, Pınar Tüfekci, Erdinç Uzun, Predicting CO and NOx emissions from gas turbines: novel data and a benchmark PEMS, 2019.

5. EMC: Continuous Emission Monitoring Systems [Електронний ресурс] // United States Environmental Protection Agency. – 2021. – Режим доступу до ресурсу: https://www.epa.gov/emc/emc-continuous-emission-monitoring-systems (дата звернення 18.11.2021).

6. CEMS vs PEMS [Електронний ресурс] // CTI Controltech Industrial Combustion and Process Control Blog. – 2019. – Режим доступу до ресурсу: https://blog.cti-ct.com/2017/09/cems-vs-pems.html (дата звернення 18.11.2021).

7. Traver ML, Atkinson RJ, Atkinson CM. Neural network-based diesel engine emissions prediction using in-cylinder combustion pressure, 1999.

8. Smrekar J, Potočnik P, Senegačnik A. Multi-step-ahead prediction of NOx emissions for a coal-based boiler, 2013.

9. Liukkonen M, Hiltunen T. Monitoring and analysis of air emissions based on condition models derived from process history, 2016.

10. Gulli A., Pal S. Deep learning with Keras. Pact, 2017. C. 296.

11. Scikit-learn in Python: Features, Prerequisites, Pros & Cons [Електронний ресурс] // upGrad. – 2020. – Режим доступу до ресурсу: https://www.upgrad.com/blog/scikit-learn-in-python/#:~:text=%20Pros%3A%20%201%20The%20library%20is%20distributed,and%20a%20vast%20international%20online%20community.%20More%20 (дата звернення 01.02.2022).

12. Google Colaboratory: Made for Cloud Based Deep Learning and Big Data Analytics [Електронний ресурс] // OpenSourceForU. – 2018. – Режим доступу до ресурсу: https://www.opensourceforu.com/2018/11/google-colaboratory-made-for-cloud-based-deep-learning-and-big-data-analytics/ (дата звернення 14.09.2021).

13. Cross-validation: evaluating estimator performance [Електронний ресурс] // Scikit learn. – 2021. – Режим доступу до ресурсу: 3.1. Cross-validation: evaluating estimator performance — scikit-learn 1.1.1 documentation (дата звернення 29.05.2022).

Додаток А

Код програми, що прогнозує величину CO

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.neural_network import MLPRegressor
from sklearn.metrics import mean_absolute_error

df1 = pd.read_csv('gt_2011.csv', usecols=['AT', 'AP',
'AH', 'AFDP', 'GTEP', 'TIT', 'TAT', 'TEY', 'CDP',
'CO'])
df2 = pd.read_csv('gt_2012.csv', usecols=['AT', 'AP',
'AH', 'AFDP', 'GTEP', 'TIT', 'TAT', 'TEY', 'CDP',
'CO'])
df_train = pd.concat([df1, df2])
(X_train, y_train) = toTimeSeries(df_train, 3)

df4 = pd.read_csv('gt_2014.csv', usecols=['AT', 'AP',
'AH', 'AFDP', 'GTEP', 'TIT', 'TAT', 'TEY', 'CDP',
'CO'])
df5 = pd.read_csv('gt_2015.csv', usecols=['AT', 'AP',
'AH', 'AFDP', 'GTEP', 'TIT', 'TAT', 'TEY', 'CDP',
'CO'])
df_test = pd.concat([df4, df5])
(X_test, y_test) = toTimeSeries(df_test, 3)

scalerX = StandardScaler()
X_train_scaled = scalerX.fit_transform(X_train)
X_test_scaled = scalerX.transform(X_test)

scalerY = StandardScaler()
y_train_scaled =
scalerY.fit_transform(y_train.reshape(-1, 1)).flatten()
```

```
y_test_scaled = scalerY.transform(y_test.reshape(-1,
1)).flatten()

co_regr = MLPRegressor(hidden_layer_sizes=(10, ),
alpha=0.1,
                       early_stopping=True,
validation_fraction=0.1, n_iter_no_change=100,
                       max_iter=2000,
                       random_state=1, verbose=0)
co_regr.fit(X_train_scaled, y_train_scaled)

r2=co_regr.score(X_test_scaled, y_test_scaled)

y_result =
scalerY.inverse_transform([co_regr.predict(X_test_scale
d)])
y_true = scalerY.inverse_transform([y_test_scaled])
mae = mean_absolute_error(y_result, y_true)
```

Додаток Б

Код програми, що прогнозує величину NOx

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.neural_network import MLPRegressor
from sklearn.metrics import mean_absolute_error

df1 = pd.read_csv('gt_2011.csv', usecols=['AT', 'AP',
'AH', 'AFDP', 'GTEP', 'TIT', 'TAT', 'TEY', 'CDP',
'NOX'])
df2 = pd.read_csv('gt_2012.csv', usecols=['AT', 'AP',
'AH', 'AFDP', 'GTEP', 'TIT', 'TAT', 'TEY', 'CDP',
'NOX'])
df_train = pd.concat([df1, df2])
(X_train, y_train) = toTimeSeries(df_train, 2)

df4 = pd.read_csv('gt_2014.csv', usecols=['AT', 'AP',
'AH', 'AFDP', 'GTEP', 'TIT', 'TAT', 'TEY', 'CDP',
'NOX'])
df5 = pd.read_csv('gt_2015.csv', usecols=['AT', 'AP',
'AH', 'AFDP', 'GTEP', 'TIT', 'TAT', 'TEY', 'CDP',
'NOX'])
df_test = pd.concat([df4, df5])
(X_test, y_test) = toTimeSeries(df_test, 2)

scalerX = StandardScaler()
X_train_scaled = scalerX.fit_transform(X_train)
X_test_scaled = scalerX.transform(X_test)

scalerY = StandardScaler()
y_train_scaled =
scalerY.fit_transform(y_train.reshape(-1, 1)).flatten()
```

```
y_test_scaled = scalerY.transform(y_test.reshape(-1,
1)).flatten()


no_regr = MLPRegressor(hidden_layer_sizes=(8, ),
random_state=1, max_iter=2000, alpha=0.1,
learning_rate_init=0.1,
                    early_stopping=True,
validation_fraction=0.3, n_iter_no_change=100,
                    verbose=0)
no_regr.fit(X_train_scaled, y_train_scaled)


r2=no_regr.score(X_test_scaled, y_test_scaled)


y_result =
scalerY.inverse_transform([no_regr.predict(X_test_scale
d)])
y_true = scalerY.inverse_transform([y_test_scaled])
mae = mean_absolute_error(y_true, y_result)
```

Додаток В
Код функції toTimeSeries

```
def toTimeSeries(dataset, takeBy=2):
  skip = 0
  take = takeBy * 10

  _X = []
  _y = []

  np_dataset = np.array(dataset).flatten()

  while skip + take - 10 < np_dataset.size:
    current = np_dataset[skip : skip + take]
    localX = list(current[:-1])
    localY = current[-1:][0]
    _X.append(localX)
    _y.append(localY)
    skip = skip + 10

  return (np.array(_X), np.array(_y))
```