Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра прикладної математики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Застосування нейронних мереж для оптимізації комплексних електромеханічних систем
(тема)

Виконав:
студент 2 курсу, групи ПМм-21-1
Лукашов Д.С.
(прізвище, ініціали)
Спеціальність
113 Прикладна математика
(код і повна назва спеціальності)
Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма
Прикладна математика
(повна назва освітньої програми)
Керівник доц. Наумейко І.В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри ПМ _____ Сидоров М.В.
(підпис) (прізвище, ініціали)

2022 р.

Харківський національний університет радіоелектроніки

Факультет _інформаційно-аналітичних технологій та менеджменту
Кафедра _прикладної математики
Рівень вищої освіти _другий (магістерський)
Спеціальність _113 Прикладна математика
(код і повна назва)
Тип програми _освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма _Прикладна математика
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри ПМ _____
(підпис)
“ 07 ” листопада 2022 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Лукашову Дмитру Сергійовичу
(прізвище, ім’я, по батькові)

1. Тема роботи Застосування нейронних мереж для оптимізації комплексних електромеханічних систем

затверджена наказом по університету від 25 жовтня 2022 р. № 1412 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 7 грудня 2022 р.

3. Вихідні дані до роботи математична модель електричного двигуна з постійним магнітом та щіточками

4. Перелік питань, що потрібно опрацювати в роботі
1. Аналіз предметної області
2. Вибір і обґрунтування методу розв’язання
3. Програмна реалізація
4. Результати обчислювального експерименту
5. Аналіз можливих застосувань

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій

1. Актуальність теми роботи
2. Постановка задачі
3. Аналіз предметної області
4. Метод чисельного аналізу
5. Результати обчислювального експерименту

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Підбір та вивчення технічної літератури за темою роботи	7 – 13 листопада 2022 р.	виконано
2	Вибір та обґрунтування методу	14 – 20 листопада 2022 р.	виконано
3	Розробка алгоритму і програми	21 – 27 листопада 2022 р.	виконано
4	Проведення аналітичних досліджень та розрахунків	28 листопада – 4 грудня 2022 р.	виконано
5	Робота над текстом пояснювальної записки	28 листопада – 6 грудня 2022 р.	виконано
6	Представлення роботи на рецензію в ЕК	7 грудня 2022 р.	виконано

Дата видачі завдання _7_ листопада 2022 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Наумейко І.В.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 52 с., 1 табл., 7 рис., 1 дод., 44 джерела.

НЕЙРОННІ МЕРЕЖІ, ГЛИБОКЕ НАВЧАННЯ З ПІДКРІПЛЕННЯМ, ОПТИМІЗАЦІЯ, МОДЕЛЮВАННЯ, ЕЛЕКТРОМЕХАНІКА, ПРИКЛАДНА МАТЕМАТИКА, ПРОГРАМНА ІНЕЖЕНРІЯ

Об’єкт дослідження – моделі комплексних електромеханічних систем.

Предметом дослідження є глибокі нейроні мережі в контексті оптимізації різноманітних систем, в тому числі електромеханічних.

Метою роботи є перевірка працездатності (proof of concept) запропонованого методу оптимізації електромеханічних систем за допомогою глибокого навчання.

Методи дослідження: комп’ютерне моделювання електромеханічної системи, оптимізаційні алгоритми, глибоке навчання з підкріпленням та нейронні мережі.

У роботі запропоновано новий підхід до оптимізації складних багато параметричних систем за допомогою нейронних мереж та методика, що дуже схожа на навчання з підкріпленням. Працездатність запропонованого підходу перевірено на задачі оптимізації параметрів змодельованого двигуна з постійними магнітами із щітками. В результаті перевірки концепції працездатність підходу підтверджено та визначено сторони, що потребують подальшого дослідження.

ABSTRACT

Introductory note: 52 pages, 1 table, 7 figures, 1 appendix, 44 sources.

NEURAL NETWORKS, DEEP REINFORCEMENT LEARNING, OPTIMIZATION, MODELING, ELECTROMECHANICS, APPLIED MATHEMATICS, SOFTWARE ENGINEERING.

The object of research is models of complex electromechanical systems.

The subject of research is deep neural networks in the context of optimization of various systems, including electromechanical ones.

Purpose of the work is to proof the proposed concept of optimizing electromechanical systems using deep learning.

Methods of research are computer simulating of electromechanical system, deep reinforcement learning and neural networks.

This work proposes a new approach to optimization of complex systems with large number of parameters using neural networks and approach which closely resembles deep reinforcement learning. Workability of the proposed approach is verified on the task of parameters optimization of permanent magnet motor with brushes. As the result of this work, workability of the approach has been confirmed and points that require additional research have been identified. Subsequent research on this topic will be focused on the identified points.

ЗМІСТ

ВСТУП

Актуальність теми. Електричні та електромеханічні пристрої вже досить широко поширені в повсякденному житті та перед лицем зміни клімату тенденція електрифікації всього, що може бути електрифіковано буде продовжена. Особливо це стосується транспортних систем та інших машини, які все ще покладаються на викопне паливо, тому що скорочення викидів є найважливішим в наші дні. Ця тенденція частково пояснює, чому так багато уваги приділяється електротехніці і чому оптимізація проектів нових систем є такою складною та має обмежений прогрес (через велику увагу даному полю існує багато досліджень з оптимізації).

Ще один фактор, який значною мірою обмежує можливість оптимізації електромеханічних систем – це довга історія галузі та її загальна зрілість. Так багато досліджень вже зроблено, так багато речей було вдосконалено до того, що здається максимальною ефективністю. Через це досить складно знайти суттєві зрушення в поставленій задачі. Також ця зрілість несе з собою багато усталених «шкіл думок» і змушує застосовувати ті самі методи з дуже обмеженою кількістю змін. Більш того, галузь електромеханіки значно відстає у впровадженні останніх досягнень у сфері розробки програмного забезпечення (хмарного обчислення та глибоко навчання) у порівнянні з іншими галузями, як, наприклад, біомедицинська [1-8] яка вже отримує багато переваг від нових технологій і підходів.

Мета і завдання кваліфікаційної роботи. Метою роботи є перевірка працездатності (proof of concept) запропонованого методу оптимізації електромеханічних систем за допомогою глибокого навчання.

Для досягнення мети та перевірки гіпотези обрано оптимізацію електричного двигуна, бо, як було наведено вище, сфера електричної інженерії є актуальною, потребує більшої оптимізації якої складно досягти, а також потребує “свіжого погляду” на використання останніх розробок в сфері машинного навчання та програмної інженерії.

Отже, для того щоб досягнути поставленої мети, необхідно виконати наступні завдання:

– розробити модель електричного двигуна із відносно великою кількістю параметрів, що можна оптимізувати;

– запрограмувати розроблену модель;

– створити та навчити нейронні мережі, за допомогою яких буде здійснено оптимізацію параметрів електричного двигуна;

– проаналізувати отримані результати та зробити висновок щодо того, чи можливо використовувати запропонований метод в цілому.

Необхідно відмітити, що вичерпного порівняння запропонованого методу оптимізації із іншими вже існуючими методами наведено не буде, оскільки, як вже було сказано, метою даної роботи є протестувати саме працездатність запропонованого методу. Таке тестування є необхідним оскільки представлений підхід принципово новий (як мінімум в контексті електромеханічних систем) і в представленому вигляді ще не був застосований. Однак, деякі ремарки та коротке порівняння із іншими методами буде наведено задля того, щоб продемонструвати деякі принципові відмінності та великий потенціал, який має представлений підхід.

Об'єктом дослідження є моделі комплексних електромеханічних систем.

Предметом дослідження є глибокі нейроні мережі в контексті оптимізації різноманітних систем, в тому числі електромеханічних.

Методи дослідження. У кваліфікаційній роботі використовується SaaS (Software as a Service) платформа хмарних обчислень Paper Space, яка за дуже низьку ціну надає доступ до потужних комп'ютерів, що налаштовані на роботу із даними у вигляді багатовимірних таблиць та на роботу із нейронними мережами. У роботі було використано мову програмування Python та бібліотеку для роботи із нейронними мережами та багатовимірними таблицями Tensorflow. А також бібліотеки для вирішення диференціальних рівнянь (scipy) та візуалізації даних matplotlib. Таким чином, методи дослідження відповідають

меті використати нові підходи та технології для оптимізації електромеханічних систем.

Публікації. Результати, отримані у кваліфікаційній роботі, було представлено на III Міжнародній студентській науковій конференції «Теоретичне та практичне застосування результатів сучасної науки» (м. Біла Церква, 7 жовтня 2022 р.) [9].

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ

1.1 Аналіз предметної області

Електромеханічні системи використовуються людством вже досить довгий час тому оптимізації таких систем було приділено велику увагу та написано багато праць. Однак через відсутність необхідної обчислювальної потужності, недосконаліть програмної інженерії та неприйняття новинок програмної інженерії та машинного навчання суспільством електричної інженерії дослідження оптимізації електромеханічних систем було сконцентровано на окремих компонентах цих систем [10-20] а не системі в цілому. Також моделі в наведених роботах мають зазвичай від трьох до п’ятнадцяти параметрів, що підлягають оптимізації.

Така невелика кількість параметрів говорить про те, що моделі для оптимізації були створені за допомогою великою людської експертизи та якихось припущень, які люди вважають логічними. Також це означає, що простір можливих параметрів зменшуються на порядки, що значно зменшує вірогідність знайти найкращий можливий варіант. Як було доведено нейронними мережами AlphaGo [21] та AlphaZero [22] люди можуть упускати найкращі рішення через обмеженість свого бачення навіть у таких сферах, які вони практикують тисячоліттями, як наприклад, шахи, де нейронна мережа AlphaZero показала небачені стратегій та тактичні маневри, які не були знайдені найкращими шаховими гравцями та алгоритмами того часу. Це назавжди змінило світ професійних шахів. Все що було необхідно для цього зробити – надати нейронні мережі “креативну свободу” та звільнити від людського бачення рішень поставленої задачі.

Повертаючись до оптимізації композитних та електромеханічних систем із більш великою кількістю параметрів, є деяка невелика кількість робіт, що досліджують цей напрямок [23-28]. Підхід який використовується у наведених

роботах називається багаторівневою оптимізацією. Даний метод уникає роботи одразу з великою кількістю параметрів за допомогою поетапної ієрархічної оптимізації електромеханічної системи. Альтернативою або додатком до багаторівневого підходу є фокус на оптимізації параметрів до яких чутливість цільової функції є найбільшою [24-28]. Більш детально процес аналізу чутливості наведено в [29, 30]. Таким чином, оптимізація більш складних систем із більшою кількістю можливих параметрів для оптимізації вже досліджується. Однак неможливо не відмітити, що у наведених дослідженнях все ще використовуються досить старі (хоча і дуже вдосконалені) методи оптимізації, а також майже або взагалі не використовуються досягнення програмної інженерії та глибокого навчання останнього десятиліття або двох, що значно обмежує потенціал наведених досліджень.

Вже існують дослідження які показують можливість використання глибокого навчання з підкріпленням для вирішення задачі мультицільової оптимізації [31-34] в цілому, не враховуючи специфіку електромеханіки. Також є дуже цікаві дослідження [35-37] щодо використання генеративних нейронних мереж для створення новітніх дизайнів компонентів електромеханічних систем. Такий підхід є безперечно цікавим та потребує подальшого вивчення, однак необхідно відмітити, що наведений метод на даний момент включає в себе використання досить великої кількості людської експертизи та застосовується для генерації тільки одного компонента. Головними напрямками досліджень цього методу мають стати надання більшої “креативної свободи” нейронним мережам, а також генерація більшої кількості компонентів або ж навіть повноцінних композитних систем. Наведені вище роботи показують позитивну тенденцію прийняття спільнотою електричних інженерів прогресу в сфері глибокого навчання та програмної інженерії, хоча таке прийняття і є досить повільним порівняно із іншими сферами інженерії та науки.

Підсумовуючи попередній аналіз предметної галузі, можна сказати, що тенденція до використання нових технологій та нейронних мереж в електромеханіці є, однак, для того щоб більш детально оцінити перспективи

запропонованого підходу до оптимізації необхідно розглянути теоретичні та практичні проблеми вже існуючих методів, для того щоб потім була можливість проаналізувати те, як представлений підхід оптимізації нівелює або зменшує знайдені проблеми.

1.2 Проблеми існуючих методів оптимізації

Для того щоб надати свободу методу оптимізації та знайти більш якісне рішення ніж існує зараз для кожного компонента електромеханічної системи та для системи в цілому необхідно надати кожному компоненту велику кількість ступенів свободи. Такий підхід може призвести до того, що результуюча модель матиме тисячі або десятки тисяч параметрів, або інакше кажучи простір можливих параметрів буде дуже і дуже багатовимірним. Це має дозволити пошук більш оптимальних рішень але в той же час зробить використання існуючих алгоритмів проблематичним або взагалі неможливим.

Однією із найбільших проблем наведених у попередньому підрозділі алгоритмів оптимізації, є так зване “прокляття розмірності” [38], яке виражається у недостатній “ємності” методу для описання складних взаємозв’язків між параметрами та цільовою функцією або ж асимптотична залежність часу роботи алгоритму від кількості вимірів простору параметрів. У першому випадку алгоритм взагалі не здатен вирішити поставлену задачу. У другому випадку алгоритм теоретично може вирішити задачу, але на практиці це займе надзвичайно великий проміжок часу, що в результаті робить алгоритм неможливим для використання.

Іншою доволі помітною проблемою є те, що немає ніяких гарантій щодо “поведінки” цільової функції. Звичайно можна вивчити її чутливість до різних параметрів, але такої інформації щодо неперервності, диференційованості і т.п. немає. Це значно обмежує список можливих алгоритмів оптимізації, наприклад, неможливо використовувати методи, що потребують похідних. Емпірично

раніше було показано, що такі методи є швидшими у пошуку локального оптимуму аніж інші способи оптимізації.

Таким чином, величезна розмірність простору параметрів є великою проблемою, яку необхідно вирішити для того, щоб знаходити більш оптимальні параметри для електромеханічних моделей. Також було б добре, якщо запропонований метод міг би якимось чином надати гарантії щодо гарних властивостей цільової функції. Нейронні мережі та глибоке навчання як раз заточені під вирішення прокляття розмірності. А запропонована архітектура рішення знаходить якісне наближення до цільової функції, що гарантує наявність гарних властивостей. Найбільш цікавою з таких властивостей є диференційованість, оскільки вона дозволяє значно розширити перелік доступних алгоритмів оптимізації. Ця додаткова властивість запропонованого рішення підвищує його цінність та актуальність адже дозволяє використовувати не підхід в цілому, а лише його частину; це розширює можливості оптимізації електромеханічних систем та дозволяє підходити до вирішення задачі більш креативно та варіативно.

1.3 Аналітична модель електричного двигуна

1.3.1 Побудова аналітичної моделі електричного двигуна

Перед побудовою моделі двигуна, що буде оптимізуватися, необхідно зазначити, що система, що розглядається в даній роботі не є дуже багатовимірною (має лише 28 вимірів) та багато компонентною (адже розглядається тільки двигун). Як вже зазначалось раніше, мета даної роботи перевірити концептуальну працездатність запропонованого метода. А його використання для оптимізації багатокомпонентних систем, що мають багато вимірні простори компонентів потребує подальших досліджень. Також

необхідно зазначити, що дана модель двигуна містить більше параметрів для оптимізації аніж моделі наведені в [10-20].

Розглянемо схему двигуна, представленого на рисунку 1.1.

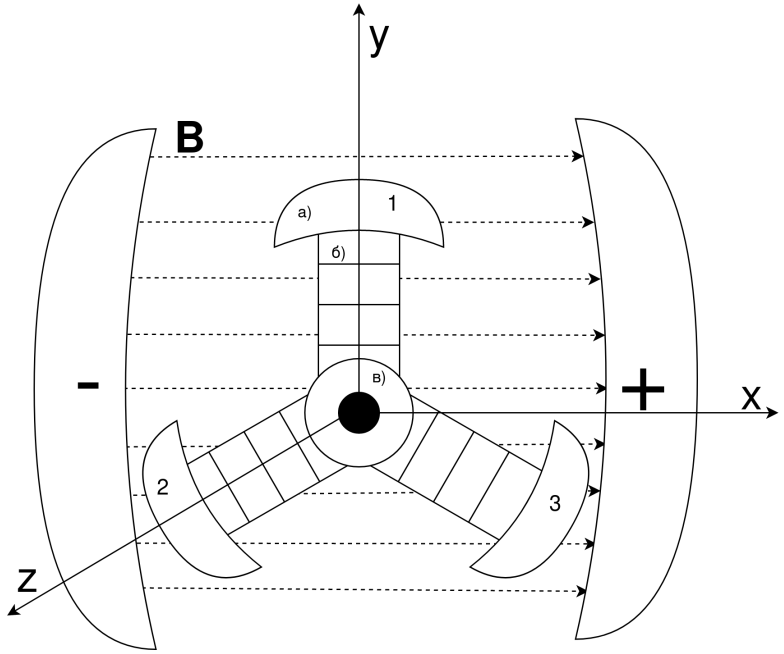

Рисунок 1.1 – Схема електричного двигуна із постійним магнітом та щіточками

Схему наведено з такого ракурсу, що момент обертання двигуна буде спрямовано в негативну сторону осі z. Літерами а) та б) позначено верхівку якоря та ніжку якоря відповідно (ніжка та верхівка разом називаються якір), а полоси на ніжці якоря – обмотка по якій протікає струм. Літерою в) позначено вал. Початок координат знаходиться в центрі валу. Вал, ніжки якорів та верхівки якорів знаходяться в середині постійного магніту, що генерує вектор індукції магнітного поля $\boldsymbol{B}$. Тільки x компонента вектору $\boldsymbol{B}$ не дорівнює нулю і має значення B_x. Числами позначено якоря двигуна. Двигун продовжується на деяку довжину в сторону негативної осі z.

Цільовою функцією, що розглядається буде ККД електричного двигуна в залежності від геометричної конфігурації ніжок та верхівок якорів. ККД електричного двигуна визначається рівнянням:

$$\eta = \frac{P_r}{P_i},$$

де η – ККД,

P_r – результуюча потужність двигуна;

P_i – вхідна потужність двигуна.

Вхідна потужність двигуна визначається наступним чином:

$$P_i = \mathrm{IV},$$

де I – сила струму;

V – напруга.

Результуючу потужність двигуна можна визначити як

$$P_r = \tau(t) \cdot \omega(t), \qquad (1.1)$$

де $\tau(t)$ – вектор моменту обертання двигуна;

$\omega(t)$ – вектор кутової швидкості обертання двигуна;

ω – скалярний добуток векторів;

t – час.

Вектор моменту обертання двигуна можна визначити наступним чином:

$$\tau(t) = \tau_1 + \tau_2 + \tau_3 = J\frac{d\omega}{dt}, \qquad (1.2)$$

де τ_i – моменти обертання створенні взаємодією магнітного поля що згенеровано струмом в обмотці якоря і та взаємодією якорів із полем тяжіння;

J – сумарний момент інерції двигуна та навантаження (в контексті даної роботи навантаження не враховується).

Моменти обертання виражаються наступним чином:

$$\tau_i = m_i \times \mathrm{B} + (\mathrm{R_i r}) \times \mathrm{F}, \qquad (1.3)$$

де m_i – магнітний момент створений якорем i;

B – вектор індукції магнітного поля згенерований постійними магнітами (визначення надано на початку підрозділу);

r – радіус вектор від початку координат до центру мас якоря i виражений в координатах якоря 1;

$\mathrm{R_i}$ – матриця повороту центру мас за годинниковою стрілкою;

F – сила гравітації що діє на якір.

Магнітний момент виражається наступним чином:

$$m_i = NI\mu S\hat{r}_i\delta_i, \qquad (1.4)$$

де N – кількість витків у обмотці;

I – сила струму;

μ – коефіцієнт посилення магнітного поля серцевиною обмотки (якорем);

S – площа одного витку котушки;

$\hat{r}_i$ – одиничний радіус вектор, що задає положення якоря двигуна відносно магнітного поля;

δ_i – функція, що визначає чи є в даний проміжок часу струм в обмотці (струм одночасно є лише в двох якорях, на початку це якоря 1 та 2).

Вирази $\hat{r}_i$ та δ_i задано наступними рівняннями:

$$\hat{r}_i = \begin{pmatrix} \cos(\omega t + \frac{\pi}{2} + \frac{2\pi(i-1)}{3}) \\ \sin(\omega t + \frac{\pi}{2} + \frac{2\pi(i-1)}{3}) \\ 0 \end{pmatrix},$$

$$\delta_i = \begin{cases} 1, \text{якщо} -\frac{\pi}{6} \le \omega t + \frac{\pi}{2} + \frac{2\pi(i-1)}{3} \le \frac{7\pi}{6}, \\ 0, \text{в інших випадках,} \end{cases} \qquad (1.5)$$

де $\omega = |\omega|$.

Функцію δ_i розраховано так, що при повороті на 120 градусів в сторону годинникової стрілки перший якір перестане отримувати струм, третій почне, а другий продовжуватиме отримувати струм.

Визначення центру мас r буде описано в наступній секції підрозділу.

Матрицю повороту R_i можна визначити за допомогою:

$$R_i = \begin{pmatrix} \cos(\omega t + \frac{\pi}{2} + \frac{2\pi(i-1)}{3}) & \sin(\omega t + \frac{\pi}{2} + \frac{2\pi(i-1)}{3}) & 0 \\ -\sin(\omega t + \frac{\pi}{2} + \frac{2\pi(i-1)}{3}) & \cos(\omega t + \frac{\pi}{2} + \frac{2\pi(i-1)}{3}) & 0 \\ 0 & 0 & 0 \end{pmatrix}. \qquad (1.6)$$

Силу гравітації, що діє на якір можна визначити як

$$F = \begin{pmatrix} 0 \\ -g \\ 0 \end{pmatrix}, \qquad (1.7)$$

де g – гравітаційна постійна.

Сумарний момент інерції J двигуна можна визначити наступним чином:

$$\mathrm{J} = m\sum_{i=1}^{3}(R_i r)\cdot(R_i r), \tag{1.8}$$

де m – маса якоря i (маси якорів вважаються однаковими).

Використовуючи визначення надані вище та виконавши обчислення отримаємо наступну сумарну формулу для моменту обертання двигуна:

$$\tau = \begin{pmatrix} 0 \\ 0 \\ -\mathrm{BNI}\mu\mathrm{S}\left(\sum_{i=1}^{3}\delta_i \sin\left(\omega t + \frac{\pi}{2} + \frac{2\pi(i-1)}{3}\right)\right) - 10m\left(\sum_{i=1}^{3} r_x \sin\left(\omega t + \frac{\pi}{2} + \frac{2\pi(i-1)}{3}\right)\right) + r_y \sin\left(\omega t + \frac{\pi}{2} + \frac{2\pi(i-1)}{3}\right) \end{pmatrix} =$$

$$= m\left(\sum_{i=1}^{3}(R_i r)\cdot(R_i r)\right)\frac{d\omega}{dt}, \tag{1.9}$$

де r_x – x компонента вектору r;

r_y – y компонента вектору r.

Аналізуючи отриману модель можна побачити, що момент обертання буде спрямовано по осі z. Це цілком очікувано оскільки момент обертання – це псевдовектор, такий що при умові, що z компонента буде від'ємною це означатиме обертання за годинниковою стрілкою і навпаки. В даній моделі, якщо вона створена правильно, очікуються обертання за годинниковою стрілкою. Також логічно очікувати, що псевдовектор кутової швидкості матиме лише z компоненту, оскільки в іншому разі у двигуна було б обертання і по іншим осям, що вступає у протиріччя із наявним моментом обертання. Як буде показано пізніше під час симуляції моделі, двигун обертається як і очікується. На далі розглядатиметься тільки z компонента моменту обертання. Також слід зазначити, що модель не враховує тертя, що створюється при обертанні двигуна.

1.3.2 Розрахунок центру мас та параметризація

Маючи модель двигуна, необхідно обрати параметри для оптимізації. Враховуючи те, що момент обертання лінійно залежить від таких параметрів, як сила струму, кількість витків обмотки і т.п. Вибір цих параметрів для оптимізації може бути нецікавим. Тому як ціль оптимізації обрано геометричну конфігурацію якорів двигуна.

Розглянемо якір двигуна, як на рис. 1.2.

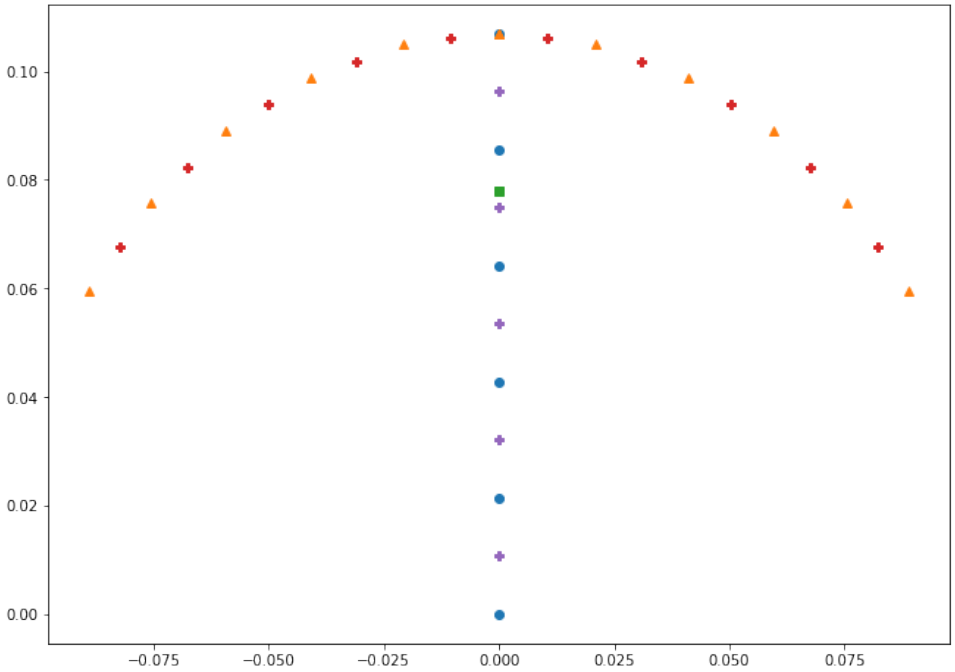

Рисунок 1.2 – Розбиття якоря двигуна на елементи

Трикутники представляють собою опорні точки якоря, круги представляють собою опорні точки ніжки якоря, хрестики – центри маси між двома сусідніми точками, а квадрат – центр мас системи верхівка якоря плюс ніжка якоря. Між двома сусідніми кругами та трикутниками лежить відрізок металу, який може мати різну густину.

Параметрами даної системи є густина відрізків якоря та кут між двома сусідніми опорними точками верхівки якоря. Радіус якоря залишається незмінним та дорівнює 0,107 м.

Параметри, що утворюють схему якоря показану на рис. 1.2 називатимемо канонічними. Вони мають наступні значення:

$$\theta_i = 25{,}276\frac{кг}{м},\ i = 0,1,...,15. \tag{1.10}$$

Індекси від 1 до 5 відповідають відрізками ніжки якоря знизу вверх. Від 6 до 15 відповідають верхівці якоря з центру наліво, а потім з центру на право.

Опорні точки верхівки якоря задаються рівняннями:

$$p = \begin{cases} \begin{pmatrix} 0.107\cos\left(\frac{\pi}{2} + \left(\frac{\pi}{\theta_i}\right)(i-16)\right) \\ 0.107\sin\left(\frac{\pi}{2} + \left(\frac{\pi}{\theta_i}\right)(i-16)\right) \end{pmatrix}, i = 16,17,...,21, \\ \begin{pmatrix} 0.107\cos\left(\frac{\pi}{2} - \left(\frac{\pi}{\theta_i}\right)(i-22)\right) \\ 0.107\sin\left(\frac{\pi}{2} - \left(\frac{\pi}{\theta_i}\right)(i-22)\right) \end{pmatrix}, i = 22,23,...,27, \end{cases} \tag{1.11}$$

де $\theta_i = 16, i = 16,...,27,$ – безвимірний параметр.

Центральну точку верхівки якоря враховано двічі для того, щоб вирахувати центр мас усіх відрізків коректно.

Прямими обмеженнями параметрів системи є проміжок [5; 50] для θ_{0-15} та проміжок [16; 26] для θ_{16-27}. Непрямими обмеженнями є те, що допустима маса якоря із урахуванням всіх параметрів має лежати на проміжку [4; 8] кг.

Таким чином, геометричну модель якоря розбито на 28-вимірний вектор параметрів θ, що необхідно оптимізувати, а також визначено набори прямих та непрямих обмежень для параметрів.

1.4 Постановка задачі дослідження

Задачею дослідження даної роботи є перевірка можливості використання нейронних мереж та технік глибокого навчання для вирішення поставленої задачі оптимізації. В контексті даної роботи пропонується перевірити працездатність методу, що буде описано в наступному розділі. А саме знайти такий вектор параметрів θ, що максимізує ККД $\mu(\theta)$.

Для того, щоб здійснити таку задачу її треба розбити на наступні підзадачі:

а) створити комп'ютерну модель (симуляцію) аналітичної моделі наведеної в розділах 1.3.1 та 1.3.2;

б) згенерувати випадкові параметри;

в) відфільтрувати ті параметри які не задовольняють прямим та непрямим умовам;

г) за допомогою симуляції обчислити ККД двигуна для тих параметрів, що залишились;

д) на основі отриманих даних навчити нейронну мережу (чи мережі) оптимізувати параметри для поставленої задачі;

ж) спробувати оптимізувати параметри;

к) проаналізувати отримані результати.

Підсумовуючи, в першому розділі наведено аналіз галузі оптимізації електромеханічних систем. Виявлено проблеми (технічні та не технічні), які стримують подальший потенціальний розвиток даної галузі, а також проаналізовано існуючи кроки щодо подолання проблем та найближчі до аналогів протестовані методи розробки дизайну компонентів електромеханічних систем.

В ході аналізу проблем оптимізації комплексних електромеханічних систем наведено можливість їх вирішення чи зменшення за допомогою нейронних мереж. Також визначено чітку ціль даної роботи та надано математичну модель електромеханічної системи, що буде використовуватись як «вимірювальний пристрій» для запропонованого методу оптимізації.

2 ВИБІР ТА ОБҐРУНТУВАННЯ МЕТОДУ РОЗВ'ЯЗАННЯ

2.1 Нейронні мережі

Представимо нейронну мережу як множину «вагів» $W \subset R$ та множину функцій активації $A \subset C^1(R,R)$, де $C^k(X,Y)$ – множина хоча б k –разів диференційних функцій $f: X \to Y$.Функцію активації будемо називати нейроном, а ваги що з'єднують два нейрона – зв'язком. Візуально нейронну мережу можна представити подібно зображенню на рис. 2.1.

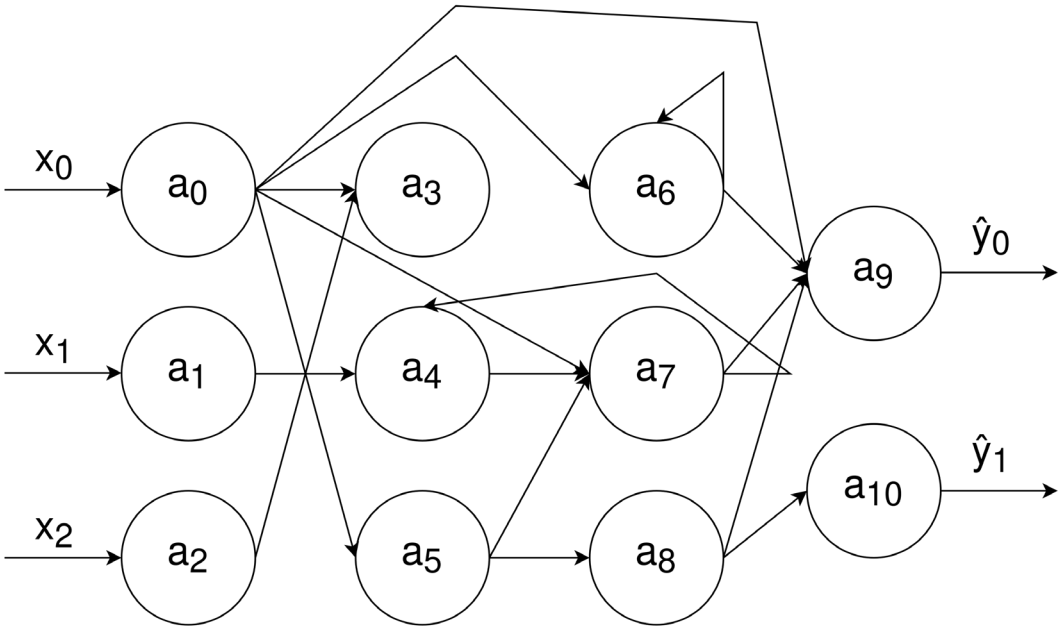

Рисунок 2.1 – Візуальне зображення нейронної мережі

На рис. 2.1 під x_i розуміються вхідні дані мережі, $a_i \in A$ – нейрони, стрілочки, що з'єднують нейрон i із нейроном j – це $w_{ij} \in W$, а $\hat{y}_i$ – вихідні значення нейронної мережі. Математично це можна виразити наступним чином:

$$\hat{y}_j = a_j\left(\sum_{i \in L_j} a_i w_{ij}\right) = v_j\left(x_0, x_1, ..., x_i; W, A\right) = v_j\left(x; W, A\right), \tag{2.1}$$

де L_j – множина індексів з'єднань, що поєднують a_i та a_j;

v_j – просто скорочена форма (у векторній формі позначається як v) для того, щоб не вказувати деталі із конкретними функціями активацій та з'єднаннями.

Необхідно відмітити перенавантаження на визначення нейрона у формулі (1.13), оскільки тут нейрон використовується і як функція активації і як вже обчислене значення функції активації. Також необхідно звернути увагу на те, що формула нейронної мережі має рекурсивну структуру, хоча ця структура і неявна.

Ключової властивістю нейронної мережі є те, що вона є диференційованою відносно будь-якого з'єднання w_{ij}, оскільки нейронна мережа складається із лінійних комбінації та композиції диференційованих функцій відносно w_{ij}. Ця властивість дозволяє використовувати головний інструмент для навчання нейронних мереж – зворотнє розповсюдження [37]. В контексті даної роботи зворотному розповсюдженню та процесу навчання нейронної мережі надано наступне дуже узагальнене визначення:

$$w_{ij}^{t+1} = u\left(w_{ij}^{t}, w_{ij}^{t-1}, \ldots, w_{ij}^{0}, \frac{\partial e}{\partial w_{ij}^{t}}, \frac{\partial e}{\partial w_{ij}^{t-1}}, \ldots, \frac{\partial e}{\partial w_{ij}^{0}} \right), \tag{2.2}$$

де w_{ij}^{t} – значення ваги між нейронам i та j на момент t–ої ітерації навчання нейронної мережі;

e – диференційовна функція похибки, яка оцінює на скільки вихідні значення нейронної мережі неправильні порівняно із реальними очікуваними значеннями;

u – деяка диференційовна функція, що комбінує значення своїх аргументів для генерації потенційно кращих значень для з'єднань між нейронами.

Надавши формальне визначення тому, як працює та навчається нейронна мережа можна переходити до описання алгоритмів необхідних для розв'язання поставленої задачі оптимізації.

2.2 Опис алгоритму “PropEvOp”

Розглянемо алгоритм PropEvOp (optimize, evaluate, propagate), представлений на рис. 2.2.

Гарантується: $\hat{\theta}$, таке що оптимізує цільову функцію.

1. $t = 0$
2. **while** $t < T$ **do**
3. $\hat{\theta} = v(\hat{\theta}; W_p, A_p)$
4. $r = v(\hat{\theta}; W_e, A_e)$
5. $q = e_p(r)$
6. $w_{ij}^{t+1} = u\left(w_{ij}^{t}, w_{ij}^{t-1}, \dots, w_{ij}^{0}, \frac{\partial q}{\partial w_{ij}^{t}}, \frac{\partial q}{\partial w_{ij}^{t-1}}, \dots, \frac{\partial q}{\partial w_{ij}^{0}} \right), w_{ij}^{k} \in W_p$
7. $t = t + 1$
8. **end while**
9. **return** $\hat{\theta}$

Рисунок 2.2 – Алгоритм PropEvOp

Як можна побачити алгоритм використовує дві нейронні мережі, що називатимуться оптимізатор (optimizer) та оцінювач (evaluator) яким відповідають набори з’єднань та нейронів (W_p, A_p) та (W_e, A_e) відповідно. Також використовується функція похибки оптимізатору e_p, яка показує наскільки параметри згенеровані оптимізатор дають гірший результат аніж уявні оптимальні параметри θ^* (в реальності значення цих параметрів невідомі, тому обирається функція що вважає більший вихідний результат алгоритму кращим для максимізації і навпаки). Під час виконання описаного алгоритму змінюються лише ваги оптимізатору. Зміна з’єднань має дозволити згенерувати більш

оптимальні параметри на наступній ітерації алгоритму. Кількість операцій T обирається довільно або за допомогою якоїсь евристики.

Оскільки даний алгоритм показує процес навчання лише оптимізатору, залишається відкритим питання навчання оцінювача. Можна підібрати багато різних алгоритмів, але один із найпростіших наведено в наступному алгоритмі, представленому на рис. 2.3.

Дано: початкові A_e та W_e. $\Theta_v, \Theta_l, e_e, f^*, \in$

Гарантується: такі A_e та W_e, що $\frac{\sum_{\theta \in \Theta_v} e_e\left(v(\theta; A_e, W_e), f^*(0)\right)}{|\Theta_v|} < \in$

1. $t = 0$
2. $b = +\infty$
2. **while** $\frac{b}{|\Theta_v|} > \in$ **do**
3. **for** $\theta \in \Theta_l$ **do**
4. $g = v(\theta; A_e, W_e)$
5. $r = f^*(\theta)$
6. $q = e_e(g, r)$
7. $w_{ij}^{t+1} = u\left(w_{ij}^t, w_{ij}^{t-1}, \ldots, w_{ij}^0, \frac{\partial q}{\partial w_{ij}^t}, \frac{\partial q}{\partial w_{ij}^{t-1}}, \ldots, \frac{\partial q}{\partial w_{ij}^0}\right)$
8. **endfor**
9. $b = 0$
10. **for** $\theta \in \Theta_v$ **do**
11. $g = v(\theta; A_e, W_e)$
12. $r = f^*(\theta)$
13. $q = e_e(g, r)$
14. $b = b + q$
15. **endfor**
16. **endwhile**
17. **return** (W_e, A_e)

Рисунок 2.3 – Алгоритм Навчання оцінювача

В наведеному алгоритмі $\Theta_l \subset \Theta_r$ – це датасет параметрів для навчання нейронної мережі, $\Theta_v \subset \Theta_r$ – датасет параметрів для валідації нейронної мережі, $\Theta_r \subset \Theta$ – множина випадково згенерованих параметрів для навчання нейронних мереж. Функція e_e – відповідає за обчислення похибки обчислювача відносно симуляції системи, f^* – цільова функція, яку необхідно оптимізувати, а $\in$ – числовий параметр, що відповідає за точність наближення нейронної мережі оцінювача до цільової функції. Позначення $|S|$ означає кількість елементів у множині S. Необхідно відмітити, що зазвичай кількість ітерацій алгоритму визначається заздалегідь, але для збереження математичної точності в Алгоритмі 2.2 вказано, що він виконується допоки не буде досягнуто заданої точності.

Підсумовуючи, алгоритм оптимізації PropEvOp складається з двох етапів. Спочатку навчається нейронна мережа оцінювач, а потім відбувається навчання нейронної мережі оптимізатора та одночасно із цим пошук більш оптимальних параметрів. Маючи формальний опис алгоритму, можна провести його аналіз з математичної точки зору для того щоб визначити теоретичне підґрунтя завдяки якому PropEvOp має працювати на практиці.

Необхідно зазначити, що аналіз алгоритмічної складності відносно часу виконання та використовуваної пам’яті наведено не буде, оскільки основою алгоритму є звичайний процес тренування нейронних мереж для яких цей аналіз уже зроблено.

2.3 Теоретичне обґрунтування працездатності алгоритму

Без втрати загальності розглянемо задачу максимізації функції $f^*: \Theta \to R$. Для цього треба довести, що $f^*\left(\hat{\theta}^{t+1}\right) \geq f^*\left(\hat{\theta}^{t}\right)$. Розкриємо дану нерівність більш детально:

$$f^*\left(v\left(\hat{\theta}^t, W_p^t, A_p^t\right)\right) \geq f^*\left(\hat{\theta}^t\right). \quad (2.3)$$

Оскільки обчислення цільової функції може бути занадто дорогим, а також відсутні гарантії того, що цільова функція диференційована, необхідно замінити її на нейронну мережу оцінювач (як це зроблено у 4-ому рядку коду алгоритму 1). Диференційованість мережі оцінювача очевидна, однак правомірність заміни необхідно довести.

Якщо визначити $v(\cdot; A_e, W_e) \approx f^*(\cdot)$ тоді і тільки тоді коли $\frac{\sum_{\theta \in \Theta_v} e_e\left(v(\theta; A_e, W_e), f^*(0)\right)}{|\Theta_v|} < \in$. Тоді виходить, що нейронна мережа оцінювач має бути правомірною заміною для цільової функції. Алгоритм 2.2, який навчає оцінювач використовує стандартну процедуру для навчання нейронної мережі та зворотнє розповсюдження. Відомо, що аналітично строго довести працездатність не можливо, однак емпіричну працездатність доведено багаторазово, завдяки чому нейронні мережі у комбінації зі зворотнім розповсюдженням і мають таку величезну популярність.

Тож, тепер маємо нерівність:

$$v\left(v\left(\hat{\theta}^t, W_p^t, A_p^t\right); W_e, A_e\right) \geq v\left(\hat{\theta}^t; W_e, A_e\right). \quad (2.4)$$

Якщо Алгоритм 2.1 зробить нерівність правдивою (2.4) після деякої кількості ітерацій t, то будуть знайдені такі параметри, стосовно яких очікується, що вони оптимізують цільову функцію.

Принциповою компонентою працездатності алгоритму 1 є функція похибки e_p, яка має бути обрана такою, що чим менше значення наближення цільової функції, тим більшою буде похибка. В самому навчанні оптимізатор також спирається на зворотнє розповсюдження, про працездатність якого було сказано вище.

Таким чином, очікувана працездатність запропонованого методу доведено хоча із деякими обмовками (яких неможливо уникнути). Ці обмовки треба брати до уваги і розуміти, що PropEvOp не гарантує знаходження будь-якого розв’язку поставленої задачі, не кажучи вже про глобально оптимальний. Однак такі самі обмовки застосовуються і до інших нейронних мереж, які все одне показують надзвичайні результати у своїх сферах. Тож, відсутність строго доведення працездатності не означає практичну відсутність цієї працездатності.

2.4 Метрики оцінки оптимізації

Очевидно, що для порівняння того на скільки гарно запропонований метод оптимізації справляється зі своєю задачею, його необхідно порівняти із іншими вже перевіреними методами оптимізації. Метрики, що необхідно виміряти та порівняти наступні:

а) якість знайдених параметрів;

б) швидкість роботи алгоритму;

в) швидкість реалізації алгоритму;

г) можливість горизонтального та вертикального скейлингу алгоритму;

д) вимоги до апаратного та програмного забезпечення;

ж) складність задачі. Мається на увазі порівняння за всіма критеріями описаними вище на якихось «простих» задачах, на яких існуючі методи вже досить гарно працюють. Та порівняння на «складних» задачах (багатокомпонентні багатопараметричні електромеханічні системи) під які і розроблявся запропонований метод.

Однак, як вже було сказано раніше, мета даної роботи перевірити принципову практичну працездатність методу. Якщо метод є працездатним, тоді вже можна буде порівнювати із іншими методами оптимізації. Тож необхідно визначитись із метриками для того, щоб сказати чи є метод принципово працездатним?

Як було описано вище, для навчання нейронних мереж необхідно згенерувати велику кількість параметрів та обчислити їх результуючий ККД. Ці обчислені значення можна використати, як метрику. Наприклад, порівняти середній ККД випадкових параметрів та ККД параметрів, що генерує PropEvOp. Із цими даними можна зробити декілька порівнянь:

а) чи здатен запропонований алгоритм згенерувати параметри, що обходять ККД випадкових параметрів хоча б на два стандартні відхилення;

б) чи здатен запропонований алгоритм згенерувати параметри, що перевершують або наближаються до найкращого вектору параметрів згенерованого випадково;

в) на скільки в середньому параметри згенеровані PropEvOp кращі аніж випадково згенеровані параметри;

г) чи генерує PropEvOp параметри, що не вписуються у задані обмеження.

У перших трьох пунктах йдеться тільки про згенеровані алгоритмом параметри, що задовольняють заданим обмеженням.

Перші два пункти мають показати чи працездатний алгоритм принципово. Другі два пункти мають показати на скільки запропонований алгоритм стабільний в плані генерації якісних параметрів та мають емпірично продемонструвати, що це не просто генератор випадкових значень, якому щастить працювати. На даний момент немає якихось значень для обраних метрик, які б показали, що алгоритм працює достатньо гарно. Такі значення мають бути визначеними під час аналізу даних.

Підбиваючи підсумки даного розділу, можна відмітити, що було надано визначення нейронних мереж та процесу їх навчання в контексті даної роботи. Ці визначення було використано для формального опису запропонованого алгоритму, який названо PropEvOp.

За допомогою формального опису алгоритму доведено, що теоретично алгоритм працездатний, хоча із деякими обмовками, що застосовуються до усіх алгоритмів, що використовують нейронні мережі, автоматичну диференціацію та зворотнє розповсюдження для навчання. Також наведено метрики, які

дозволять оцінити принципову працездатність алгоритму та розпочати вивчення якості його роботи більш загально. Наведено метрики за допомогою яких можна буде порівняти PropEvOp із іншими алгоритмами оптимізації, якщо запропонований метод виявиться принципово та “достатньо” працездатним для подальшого його вивчення.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Вибір технологій для реалізації

Основою даної роботи є застосовування глибокого навчання для вирішення поставленої задачі оптимізації складної електромеханічної системи. Тож логічно обрати інструменти програмної реалізації, що максимально спрощують роботу із нейронними мережами і, по можливості, дозволяють створювати симуляцію електромеханічних систем. На щастя, всі ці параметри комбінуються у мові програмування Python та такі бібліотеки як SciPy та Tensorlfow. Іншою важливою складовою є апаратне забезпечення, оскільки робота із нейронними мережами потребує досить великих обчислювальних потужностей та і симуляція багато параметричних електромеханічних систем не відстає у своїх апетитах на ресурси. До того ж, обраний спосіб забезпечення апаратних потреб має бути ефективним з точку зору затрати коштів та можливості розширення у майбутньому. Наведені вимоги, в цілому, задовольняються будь-яким хмарним провайдером, однак, є ще один параметр – це зручність розробки. Саме через нього обрано хмарного провайдера під назвою Paperspace.

Мова програмування Python – це строго динамічно типізована інтерпретована мова програмування, яка використовується вже багато років і завоювала визнання як мова, якій під силу найрізноманітніші завдання: від автоматизації відсилання електронних листів та створення API для вебсайтів до розробки найскладніших нейронних мереж та обробки петабайтів даних. Для даної мови існує безліч різних бібліотек та інструментів для роботи, які адаптують її під будь-які виклики. В контексті даної роботи важливими є бібліотеки SciPy, NumPy та TensorFlow.

SciPy – це бібліотека для наукового програмування. В ній реалізовані різноманітні алгоритми оптимізації, чисельної апроксимації диференціальних рівнянь, лінійної алгебри, перетворень Фур’є та багато чого іншого. Саме

ця бібліотека є основою для симуляції електромеханічної системи в даній роботі. А основою бібліотеки SciPy є бібліотека NumPy, яка представляє собою набір засобів для роботи із векторами, матрицями, тензорами та базовою лінійною алгеброю взагалі. Ці дві бібліотеки мають зручні інтерфейси, вони безкоштовні та досить гарно працюють із великими об’ємами даних, що робить є очевидним вибором для багатьох науковців. Такий вибір не обійшов і дану роботу.

TensorFlow – це бібліотека для створення та роботи із нейронними мережами, розроблена компанією Google. Даний інструмент є одним із найпопулярнішим у своїй ніші. Він зручний у використанні, безкоштовний, дозволяє працювати із колосальними об’ємами даних та створювати нейронні мережі, що показують приголомшливі результати, які демонструють дослідники із компанії Google. До того ж, ця бібліотека, як і всі використані у цій роботі має відкрити похідний код та активно підтримується незалежними розробниками з усього світу. Такий підхід забезпечує надзвичайно гарну документацію, низький поріг входу та отримати допомогу по будь-якому питанню. Це є надзвичайно важливим, адже, як було наведено на початку роботи, однією з причин стагнації сфери оптимізації в електричній інженерії є брак використання нових технологій. Тож, для того, щоб спільнота прийняла такі технології, необхідно, щоб вони були доступними, простими в експериментуванні та мали гарну підтримку. На щастя, зазначені вище бібліотеки мають всі ці характеристики.

Також в даній роботі використано ще декілька додаткових бібліотек. Таких як pandas, matplotlib та інші, однак вони не потребують такого детального опису, оскільки виконують лише допоміжні функції і не є цікавими в контексті даної роботи.

Для того щоб ефективно використати описані програмні інструменти потрібне потужне апаратне забезпечення. Для доступу до такої апаратної складової використано хмарний провайдер Paperspace. Це дозволило отримати доступ до 8 віртуальних ядер процесора, відеокарти Nvidia A4000 та

45 гігабайт оперативної пам’яті. Це досить потужні ресурси але вони є далеко не максимумом, який може надати ця платформа. Однак цих ресурсів виялвилось достатньо для виконання поставленої задачі. Однією з визначних характеристик хмарного провайдеру Paperspace є те, що всі необхідні для роботи інструменти та бібліотеки вже встановлено та сконфігуровано на віртуальній машині у дуже зручному середовищі для роботи JupyterLab. Дане середовище та різні його варіації є стандартом для роботи із машинним навчанням, глибоким навчанням та великими даними із використанням мови програмування Python. Той факт, що все встановлено та коректно сконфігуровано дозволяє зекономити час та зосередитись не виявленні та ліквідації проблем роботи інструментів, а на корисній роботі по дослідженню та вирішенню поставленої задачі. Даний постачальник хмарних послуг є максимально простим у використанні, дуже дешевим порівняно із іншими конкурентами, має гарну документацію та службу підтримки, що швидко реагує на запити. Як вже було вказано вище, такі параметри мають дуже велике значення, як і функціональні параметри платформи.

Розглянувши усі інструменти, що були використані при розробці програмного забезпечення для даної роботи, можна перейти до розгляду різних етапів реалізації.

3.2 Симуляція електромеханічної системи

Першим етапом реалізації поставленої задачі, є симуляція програмної електромеханічної системи. Даний етап необхідний для того, щоб була можливість по перше, згенерувати дані для навчання нейронних мереж, по друге, для того, щоб потім оцінити результат роботи нейронних мереж.

Симуляції електромеханічної системи складається з двох етапів:

а) розрахунок центру мас, використовуючи параметри, описані в розділі 1;

б) симуляція поведінки електромеханічної системи з урахуванням знайденого центру мас та інших параметрів системи.

Код першого етапу можна знайти в [40], а код другого етапу в [41], увесь код можна знайти у Додатку А роботи, а увесь код та всі проміжні дані, згенеровані в процеси роботи можна знайти в [42].

Розрахунок центру мас не має в собі нічого особливого, він дуже чітко реалізує все, що описано у підпункті 1.3.2. А симуляція системи чітко реалізовує рівняння (10).

Таблиця 3.1 – Постійні параметри електромеханічної системи

Літера	Розмірність	Опис	Значення
B	Тл	Індукція магнітного поля постійного магніту	1
N	1	Кількість витків обмотки якоря	50
I	А	Сила струму	18
V	В	Напруга	52
μ	1	Ефективна магнітна проникність якоря	2,08
s	$м^2$	Площа перерізу ніжки якоря (одного витку обмотки)	0,015

3.3 Генерація даних

Для того, щоб ефективно навчити нейронну мережу оцінювач поводитись, як симуляція електромеханічної системи, було вирішено згенерувати приблизно 200 тис пар параметрів та результуючого ККД.

Для отримання бажаного результату спочатку згенеровано 1 мільйон випадкових параметрів (за рівномірним розподілом) відповідно до прямих обмежень описаних у підпункті 1.3.2. Потім проведено валідацію їх відповідності непрямим обмеженням. В результаті залишилось лише 168676 векторів параметрів.

Наступним кроком стала симуляція результуючого ККД цих параметрів. Ця задача виявилось дещо складнішою ніж очікувалось. Одна симуляція електромеханічної системи потребувала значного часу для отримання результатів для одного набору параметрів. Це зробило час очікування на результати 168676 занадто великим, приблизно 60 годин. Однак, при лише одній паралельній симуляції системи не використовувався увесь потенціал віртуальної машини наданої хмарним провайдером, тому було прийнято рішення розпаралелити симуляцію за допомогою стандартного підходу – MapReduce. «Map» задачею стала обробка однієї восьмої датасету, а «Reduce» задачею стало зливання результатів в один датасет. Кількістю «Map» задач було обрано 8, таким чином, що на одне ядро процесору – одна симуляція. За допомогою такого підходу, вдалось значно зменшити час очікування з приблизно 60 годин до 8, а також максимально використати усі надані ресурси, особливо процесор, як можна побачити на наступних графіках, зображених на рис. 3.1 та рис. 3.2.

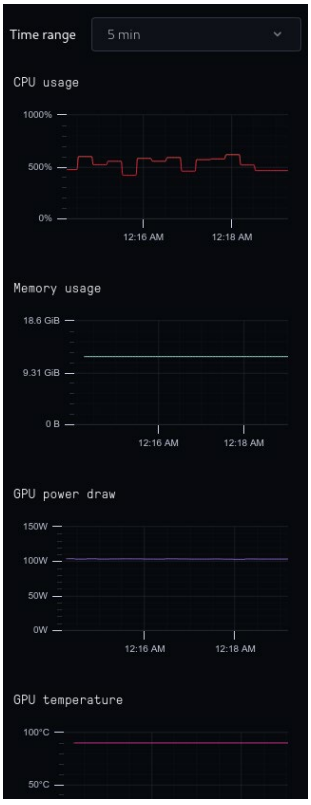

Рисунок 3.1 – Графік використання процесору, оперативної пам’яті, споживання електроенергії відеокартою та температури відеокарти

Рисунок 3.2 – Графік використання обчислювальної потужності відеокарти та відео пам’яті

Таким чином, було створено необхідний датасет для тренування нейронних мереж, що є наступним етапом. Код першого етапу генерації даних можна знайти в [43], код другого етапу в [44].

3.4 Створення та використання нейронних мереж

Нейронні мережі створено відповідно до алгоритмів описаних в розділі 2. Якихось складних та цікавих архітектурних рішень при проектуванні нейронних мереж робити не довелось, оскільки електромеханічна система, що розглядається – не така велика і складна для нейронних мереж.

Необхідно відмітити, що після тренування середня абсолютна похибка оцінювача на валідаційному датасеті становить трохи менше трьох відсотків, що є задовільним в контексті даної роботи. А нейронна мережа оптимізатор була спроможна згенерувати такі параметри, які досить гарно оптимізували електромеханічну систему. Більш детальний огляд результатів наведено в розділі 4. Код реалізації нейронних мереж можна знайти в [45].

4 РЕЗУЛЬТАТИ ОБЧИСЛЮВАЛЬНОГО ЕКСПЕРИМЕНТУ ТА ЇХ АНАЛІЗ

4.1 Аналіз працездатності алгоритму

Відповідно до критеріїв працездатності алгоритму, викладених у пункті 2.4, необхідно перевірити, чи здатен алгоритм обійти якість випадково згенерованих параметрів. Статистичні відомості, щодо значення ККД випадково згенерованих параметрів наведено на рис. 4.1.

```
count    1.686760e+05
mean     5.136112e-01
std      6.173697e-02
min      1.417582e-08
25%      4.838380e-01
50%      5.138777e-01
75%      5.464295e-01
90%      5.747556e-01
95%      6.018999e-01
99%      6.387804e-01
max      8.224433e-01
```

Рисунок 4.1 – Статистичні відомості ККД випадково згенерованих параметрів

На рисунку 4.1 дано інформацію про кількість значень, середня значення, стандартне відхилення, мінімальне значення, 25, 50, 75, 90, 95 та 99 перцентилі, а також максимальне значення.

Одне з перших значень згенерованих алгоритмом має параметри, що призводять до ККД, що дорівнює 0,85488015. Як можна побачити, алгоритм не бачив багато значень, які б призводили до великого ККД, це має означати, що заданий метод навчився створювати якісні параметри, а не просто копіювати ті, що бачив. Також, можна чітко побачити, що принципово, алгоритм здатен згенерувати параметри, що обходять як мінімум два стандартні відхилення випадково згенеровані параметри.

4.2 Аналіз стабільності працездатності алгоритму

Для аналізу стабільності алгоритму необхідно вивчити статистику згенерованих ним ККД, що наведено на рис. 4.2.

count	9.500000e+02
mean	6.496342e-01
std	1.800711e-01
min	1.358666e-08
25%	5.473915e-01
50%	6.638775e-01
75%	7.745798e-01
90%	8.837490e-01
95%	9.345906e-01
99%	9.598385e-01
max	9.977619e-01

Рисунок 4.2 – Статистичні відомості ККД параметрів згенерованих алгоритмом

На рисунку 4.2 дано інформацію про кількість даних, середнє значення, стандартне відхилення, мінімальне значення, 25, 50, 75, 90, 95 та 99 перцентилі.

Як можна побачити з цих даних, в середньому алгоритм генерує кращі параметри аніж просто параметри, що вибрані випадково. Однак справжній потенціал можна побачити у 75 перцентилі, де алгоритм обійшов випадкові параметри на 2 стандартні відхилення. В той же час 99 перцентиль показують надто гарні результати і скоріш за все є похибкою моделі.

Необхідно відмітити, що алгоритм досить сильно залежить від початкових випадкових параметрів з яких починається оптимізація і, коли ці параметри досить гарні, то алгоритм здатен знайти досить якісне рішення. Одним з можливих покращень алгоритму може стати використання не одного набору параметрів на початку пошуку, а декількох для того, щоб зменшити вплив випадковості на знайдене рішення та покращити стабільність роботи.

Іншою невеликою проблемою алгоритму стало те, що він може генерувати параметри, що не задовольняють прямим або непрямим обмеженням. Дану проблему можна спробувати вирішити за допомогою зміни функції похибки алгоритму. Вона має включати в себе якусь залежність від того, на скільки неправильними є згенеровані параметри. Це дозволить направити навчання алгоритму в правильну сторону та зменшити кількість параметрів, що були згенеровані не правильно.

Підбиваючи підсумки експерименту, можна із упевненістю сказати, що алгоритм принципово здатен вирішувати поставлену задачу. Є певні проблеми зі стабільністю, які можливо вирішити простими методами, однак це потребує подальшого дослідження. Також можна сказати, що в подальшому дослідженні є сенс, оскільки навіть в такій ранній версії алгоритм здатен знаходить досить якісні рішення за досить невеликий проміжок часу. Подальші дослідження в цьому напрямку мають бути спрямовані в першу чергу на його застосуванні на більш складних електромеханічних (та й не тільки системах) за для додаткової валідації принципової працездатності. А також на покращенні стабільності алгоритму.

4.3 Аналіз можливих застосувань

Аналізуючи запропонований алгоритм, можна помітити, що він є досить загальним. Немає ніякої прив’язки до конкретної сфери. Електромеханіку обрано для надання додаткового контексту даній роботі, як одну зі сфер, що потенційно має велику необхідність у свіжому погляді на вже існуючі рішення.

Також запропонований метод оптимізації використовує нові рішення в сфері програмної інженерії, комп’ютерних наук та хмарних застосунків. Ці рішення дозволяють досить легко реалізовувати взаємодію із іншими, більш старими інструментами, що на разі використовуються для симуляції різноманітних систем. Також така підбірка технологій відкриває доступ

до великої кількості якісної документації та можливості отримати допомогу від інших людей, що вже знаються на цих технологіях. І додатковим бонусом є те, що запропонований метод не вимагає використовувати його повністю, можливо навчити мережу-оцінювач та використовувати її у комбінації з іншими методами оптимізації.

Таким чином, запропонований метод оптимізації можливо застосовувати для будь-якої задачі оптимізації. Головною умовою є можливість отримати необхідну кількість даних для навчання нейронних мереж, інших особливих умов, що стосуються конкретно електромеханіки немає. Також те, що метод покладається на нові технології, які добре масштабуються та взаємодіють з іншими інструментами, що можуть бути використані інженерами та науковцями у різних сферах розширює і так велику можливу сферу застосування та знижує поріг для входу.

ВИСНОВКИ

В результаті даної роботи проаналізовано сферу оптимізації складних механічних систем. Аналіз виявив, що електромеханіка має деякий набір проблем, що стосуються оптимізації. Ці проблеми можна поділити на дві категорії: велика зрілість сфери та небажання або занадто велика складність у використанні нових розробок у сфері програмної інженерії та глибокого навчання. На основі цього аналізу виявлено критерії яким мав би відповідати алгоритм, що намагається вирішити згадані вище проблеми. Після чого було створено модель для перевірки принципової працездатності наведеного методу.

Основною інновацією роботи є наведення алгоритму для розв’язку загальної задачі оптимізації без прив’язки до електромеханіки чи будь-якої іншої сфери. Наведено теоретичне обґрунтування працездатності запропонованого алгоритму. А також метрики, що необхідно використати для емпіричної оцінки принципової працездатності алгоритму.

Проведено експеримент із математичною моделлю двигуна та використанням запропонованого методу оптимізації. В результаті цього експерименту виявлено, що принципово алгоритм працює і здатен знаходити дуже якісні параметри. Також виявлено деякі проблеми алгоритму, які має бути можливо розв’язати за допомогою простих змін. Підсумком експерименту стало те, що запропонований алгоритм є працездатним, а подальше його вивчення та покращення має сенс. Також виявлено потенційні напрямки для майбутніх робіт за даною та суміжними темами.

В результаті роботи розроблено новий алгоритм для оптимізації, емпірично показано, що він працює, а також пояснено якій потенціал він має та чого необхідно продовжувати дослідження.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Stein L. D. The case for cloud computing in genome informatics. Genome Biology. 2010. № 11 (5). P. 1–7.

2. Langmead B., Nellore A. Cloud computing for genomic data analysis and collaboration // Nature Reviews Genetics. 2018. № 19. P. 208–219.

3. Driscoll A. O., Daugelaite J., Sleator R. D. 'Big data', hadoop and cloud computing in genomics // Journal of Biomedical Informatics. 2013. № 46 (5). P. 774–781.

4. Rainbow: A tool for large-scale whole-genome sequencing data analysis using cloud computing / Zhao S, Prenger K, Smith L. [etc.] // BMC Genomics. 2013. № 14. P. 1–11.

5. Schatz M. C., Langmead B., Salzberg S. L. Cloud computing and the dna data race // Nature Biotechnology. 2010. № 28. P. 691–693.

6. Navale V., Bourne P. E. Cloud computing applications for biomedical science: A perspective // PLoS Computational Biology. 2018. № 14 (6). P. 1–14.

7. A virtual machine for automated and portable sequence analysis from the desktop using cloud computing / Angiuoli S. V., Matalka M., Gussman A. [etc.] // BMC Bioinformatics. 2011. № 12 (1). P. 1–15.

8. Searching for SNPs with cloud computing / Langmead B., Schatz M. C., Lin J. [etc.] // Genome Biology. 2009. № 10 (11). P. 1–10.

9. Lukashov D, Naumejko I. Practical requirements for composite electromechanical systems optimization // Матеріали III Міжнародної студентської наукової конференції, м. Біла Церква, 7 жовтня, 2022. С. 191-194.

10. Ekinci S., Demiroren A., Hekimoglu B. Parameter optimization of power system stabilizers via kidney-inspired algorithm // Transactions of the Institute of Measurement and Control. 2019. № 41. P. 1405–1417.

11. Ekinci S., Demiroren A., Hekimoglu B. Parameter optimization of power system stabilizer via Salp Swarm algorithm // 5th International Conference on Electrical and Electronic Engineering (ICEEE). 2018. P. 143–147.

12. Robust design of multimachine power system stabilizers based on improved non-dominated sorting genetic algorithms / Guesmi T., Farah A., Abdallah H. H. [etc.] // Electrical Engineering. 2018. № 100. P. 1351–1363.

13. Optimization of thermal modes and cooling systems of the induction traction engines of trams / Liubarskyi B, Petrenko O., Lakunin D. [etc.] // Eastern-European Journal of Enterprise Technologies. 2018. № 3 (9). P. 59–67

14. Analysis of optimal operating modes of the induction traction drives for establishing a control algorithm over a semiconductor transducer / Liubarskyi B., Petrenko O., Shaida V. [etc.] // Eastern-European Journal of Enterprise Technologies. 2017. № 3 (9). P. 50–57.

15. Beniakar M. E., Kakosimos P. E., Kladas A. G. Strength pareto evolutionary optimization of an in-wheel pm motor with unequal teeth for electric traction // IEEE Transactions on Magnetics. 2015. № 51 (3). P. 1–4.

16. Electric vehicle powertrain architecture and control global optimization / Janiaud N., Vallet F.-X., Petit M. [etc.] // World Electric Vehicle Journal. 2009. № 3 (4). P. 682–693.

17. Yıldız B. S., The mine blast algorithm for the structural optimization of electrical vehicle components // Materials Testing. 2020. № 62. P. 497–502.

18. Hekimoglu B., Ekinci S. Grasshopper optimization algorithm for automatic voltage regulator system // 5th International Conference on Electrical and Electronic Engineering (ICEEE). 2018. № 41. P. 152–156.

19. Hekimoglu B. Sine-cosine algorithm-based optimization for automatic voltage regulator system // Transactions of the Institute of Measurement and Control. 2019. № 41. P. 1761–1771.

20. Implementation of particle swarm optimization (pso) algorithm for tuning of power system stabilizers in multimachine electric power systems / Verdejo H., Pino V., Kliemann W. [etc.] // Energies. 2020. № 13 (8). P. 1–29.

21. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play / Silver D., Hubert T., Schrittwieser J. [etc.] // Science. 2018. № 362 (6419). P. 1140–1144.

22. Mastering the game of go with deep neural networks and tree search / Silver D., Huang A., Maddison C. J. [etc.] // Nature. 2016. № 529. P. 484–489.

23. A review of design optimization methods for electrical machines / Lei G., Zhu J., Guo Y. [etc.] // Energies. 2017. № 10. P. 1–32.

24. System-level design optimization methods for electrical drive systems: Deterministic approach / Lei G., Wang T., Guo Y. [etc.] // IEEE Transactions on Industrial Electronics. 2014. № 61. P. 6591–6602.

25. System-level design optimization method for electrical drive systems – robust approach / Lei G., Wang T., Zhu J. [etc.] // IEEE Transactions on Industrial Electronics. 2014. № 62. P. 4702–4713.

26. Multi-objective optimization of an outer-rotor v-shaped permanent magnet flux switching motor based on multilevel design method / Zhu X., Shu Z., Quan L. [etc.] // IEEE Transactions on Magnetics. 2016. № 52. P. 24–30.

27. Multilevel design optimization and operation of a brushless double mechanical port flux-switching permanent-magnet motor / Xiang Z., Zhu X., Quan L. [etc.] // IEEE Transactions on Industrial Electronics. 2016. № 63. P. 6042–6054.

28. Co-reduction of torque ripple for outer rotor flux-switching pm motor using systematic multi-level design and control schemes / Zhu X., Xiang Z., Zhang C. [etc.] // IEEE Transactions on Industrial Electronics. 2017. № 64. P. 1102–1112.

29. Augusto O. B., Bennis F., Caro S. Multiobjective engineering design optimization problems: A sensitivity analysis approach // Pesquisa Operacional. 2012. № 32 (3). P. 575–596.

30. Spagnol A., Riche R. L., Veiga S. D. Global sensitivity analysis for optimization with variable selection // SIAM/ASA Journal on Uncertainty Quantification. 2019. № 7 (2). P. 417-443.

31. Li K., Zhang T., Wang R. Deep reinforcement learning for multiobjective optimization // IEEE Transactions on Cybernetics. 2021. № 51. P. 3103–3114.

32. Meta-learning-based deep reinforcement learning for multiobjective optimization problems / Zhang Z., Wu Z., Zhang H. [etc.] // IEEE Transactions on Neural Networks and Learning Systems. 2022. P. 1–14.

33. Yonekura K., Hattori H.. Framework for design optimization using deep reinforcement learning // Structural and Multidisciplinary Optimization. 2019. № 60. P. 1709–1713.

34. Modrl/d-el: Multiobjective deep reinforcement learning with evolutionary learning for multiobjective optimization / Zhang Y., Wang J., Zhang Z. [etc.] // IEEE Transactions on Cybernetics. 2021. P. 1–8.

35. Deep generative design: Integration of topology optimization and generative models / Oh S., Jung Y., Kim S. [etc.] // Journal of Mechanical Design. 2019. № 141 (11). P. 1–22.

36. Integrating deep learning into cad/cae system: Case study on road wheel design automation / Yoo S., Lee S., Kim S. [etc.] // Structural and Multidisciplinary Optimization. 2021. № 64 (1). P. 1–21.

37. Integrating deep learning into cad/cae system: Generative design and evaluation of 3d conceptual wheel / Yoo S., Lee S., Kim S. [etc.] // Structural and Multidisciplinary Optimization. 2021. № 64 (4). P. 1–29.

38. Goodfellow I., Bengio Y., Courville A. Deep Learning (Adaptive Computation and Machine Learning series) / Publisher : The MIT Press, 2016. 800 p.

39. Лукашов Д. С. Код розрахунку центру мас [Електронний ресурс] / Дмитро Сергійович Лукашов – URL : https://github.com/DimonLuk/am-diploma-dev/blob/main/center_of_mass_calculation.ipynb. (Дата звернення 05.12.2022).

40. Лукашов Д. С. Код симуляції електромеханічної системи [Електронний ресурс] / Лукашов Д. С. – URL : https://github.com/DimonLuk/am-diploma-dev/blob/main/3phase_pm_engine_simulation.ipynb. (Дата звернення 05.12.2022).

41. Лукашов Д. С. Код роботи, проміжні та кінцеві дані [Електронний ресурс] / Лукашов Д. С. – URL : https://github.com/DimonLuk/am-diploma-dev. (Дата звернення 05.12.2022).

42. Лукашов Д. С. Код генерації даних, етап 1 [Електронний ресурс] / Лукашов Д. С. – URL : https://github.com/DimonLuk/am-diploma-dev/blob/main/data_generation.ipynb. (Дата звернення 05.12.2022).

43. Лукашов Д. С. Код генерації даних, етап 2 [Електронний ресурс] / Лукашов Д. С. – URL : https://github.com/DimonLuk/am-diploma-dev/blob/main/processing.py. (Дата звернення 05.12.2022).

44. Лукашов Д. С. Код реалізації нейронних мереж [Електронний ресурс] / Лукашов Д. С. – URL : https://github.com/DimonLuk/am-diploma-dev/blob/main/neural_networks.ipynb. (Дата звернення 05.12.2022).