Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет __Комп'ютерної інженерії та управління_____
(повна назва)

Кафедра Автоматизації проектування обчислювальної техніки_____
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти ____другий (магістерський)_____
(рівень вищої освіти)

_Моделі та методи оптимізації машинного навчання з урахуванням квантового шуму для гібридних квантових нейронних мереж_____.
(тема)

Виконав: студент 2 курсу, групи СКСм-24-1

_____Кутній Михайло Сергійович____
(прізвище, ініціали)

Спеціальність _*123–Комп'ютерна інженерія*
(код і повна назва спеціальності)

Тип програми ____*освітньо-професійна* _____
(освітньо-професійна або освітньо-наукова)

Освітня програма _____
__*Спеціалізовані комп'ютерні системи*
(повна назва освітньої програми)

Керівник __проф.Литвинова Є. І._____
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ *Чумаченко С.В.*
(підпис) (прізвище, ініціали)

2025 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління

Кафедра Автоматизації проєктування обчислювальної техніки

Рівень вищої освіти другий (магістерський)

Спеціальність 123_ Комп'ютерна інженерія
(шифр і назва)

Тип програми Освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Спеціалізовані комп'ютерні системи
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____20 ____ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві *Кутньому Михайлу Сергійовичу*
(прізвище, ім'я, по батькові)

1. Тема роботи Моделі та методи оптимізації машинного навчання з урахуванням квантового шуму для гібридних квантових нейронних мереж
затверджена наказом по університету від *07 листопада* 2025 р. № *1012 Ст*

3. Вхідні дані до роботи (проекту)

Набір даних: MNIST (рукописні цифри)

QNN з кільцевою топологією

Кодування ознак через оберти Ry

MNIST, PCA-зменшення розмірності

Parameter-shift правило

Аналіз продуктивності в режимах 15/36/100 шотів

Шум квантових вимірювань (shot noise)

4. Перелік питань, що потрібно опрацювати в роботі

1.Проаналізувати сучасні оптимізатори для варіаційних квантових алгоритмів.

2. Виявити недоліки існуючих підходів в умовах квантового шуму.

3. Обґрунтувати і описати новий метод адаптивної оптимізації.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри)

18 комп'ютерна ілюстрація (слайди)

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи (проекту)	Термін виконання етапів роботи (проекту)	Примітка
1	Аналіз теми.	02.09.25 — 06.09.25	
2	Підготовка плану та структури роботи.	07.09.25 — 14.09.25	
3	Огляд літератури та збір теоретичних матеріалів.	15.09.25 — 01.10.25	
4	Розробка моделі та підходів до збору даних.	02.10.25 — 12.10.25	
5	Проведення експерименту. Обробка даних.	13.10.25 — 20.10.25	
6	Аналіз результатів і формулювання висновків.	21.10.25 — 25.10.25	
7	Написання основних розділів роботи.	26.10.25 — 24.11.25	
8	Оформлення рисунків, таблиць та посилань.	25.11.25 — 13.12.25	
9	Оформлення та редагування згідно вимог.	14.12.25 — 21.12.25	
10	Захист	22.12.25 — 24.12.25	

Дата видачі завдання *07 листопада* 2025 р.

Здобувач _____ *Кутній М.С.*
(підпис)

Керівник роботи _____ _____ *проф. Литвинова Є.І.*
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка містить 82 сторінок., 1 рис., 5 табл., 20 джерел.

КВАНТОВЕ МАШИННЕ НАВЧАННЯ, ВАРІАЦІЙНІ КВАНТОВІ АЛГОРИТМИ, ГІБРИДНІ НЕЙРОННІ МЕРЕЖІ, NISQ, ШУМ КВАНТОВИХ ВИМІРЮВАНЬ, АДАПТИВНА ОПТИМІЗАЦІЯ, QNA-ADAM, ГРАДІЄНТНА ДИСПЕРСІЯ.

Предметом дослідження є сучасні гібридні квантово-класичні нейронні мережі, проблеми їх оптимізації в умовах квантового шуму, а також підходи до покращення стійкості тренувального процесу. Метою магістерської кваліфікаційної роботи є теоретичне обґрунтування та концептуальна інтеграція алгоритму шумостійкої оптимізації Adam у навчальний цикл варіаційної квантової моделі.

У ході виконання роботи було досліджено наявні оптимізатори для варіаційних алгоритмів, проаналізовано їх недоліки в умовах NISQ-пристроїв, а також запропоновано формальну модель шумоадаптивного методу оновлення параметрів. Розроблено, реалізовано та проаналізовано узагальнену архітектуру гібридної QNN, до якої логічно інтегровано шумоадаптивний Adam без зміни базової структури обробки.

ABSTRACT

The explanatory note contains: 82 pages, 1 figures, 5 tables, 20 sources according to the list of links.

QUANTUM MACHINE LEARNING, VARIATIONAL QUANTUM ALGORITHMS, HYBRID NEURAL NETWORKS, NISQ, QUANTUM MEASUREMENT NOISE, ADAPTIVE OPTIMIZATION, QNA-ADAM, PARAMETER-SHIFT RULE, GRADIENT VARIANCE.

The subject of the research is modern hybrid quantum-classical neural networks, problems of their optimization in conditions of quantum noise, as well as approaches to improving the stability of the training process. The purpose of the master's qualification work is the theoretical justification and conceptual integration of the noise-resistant optimization algorithm Adam into the training cycle of the variational quantum model.

During the work, existing optimizers for variational algorithms were investigated, their shortcomings in the conditions of NISQ devices were analyzed, and a formal model of the noise-adaptive parameter update method was proposed. A generalized architecture of a hybrid QNN was developed, implemented and analyzed, into which the noise-adaptive Adam was logically integrated without changing the basic processing structure.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

QNN – quantum neural network.

VQA – variational quantum algorithm.

NISQ – Noisy Intermediate-Scale Quantum. (англ. Domain Name System).

PQC – Parametrized quantum cirquit.

ВСТУП

Швидкий прогрес у квантових обчисленнях відкрив нові горизонти для машинного навчання, обіцяючи обчислювальні переваги для завдань, які є складними для класичних алгоритмів. Серед найперспективніших підходів є варіаційні квантові алгоритми (VQA) та гібридні квантово-класичні нейронні мережі, які використовують сильні сторони як квантових, так і класичних ресурсів. Однак сучасне покоління квантових процесорів, які часто називають шумними проміжними квантовими пристроями (NISQ), принципово обмежене шумом, декогеренцією та апаратними обмеженнями. Зокрема, квантовий шум вимірювання (дробовий шум), вносить значну стохастичність у процес оптимізації, підриваючи ефективність традиційних алгоритмів навчання, розроблених для класичних, безшумних середовищ.

Оптимізація варіаційних квантових схем є центральною проблемою в квантовому машинному навчанні. Хоча класичні оптимізатори, такі як стохастичний градієнтний спуск (SGD) та Adam, були адаптовані для квантових умов, їхня продуктивність часто погіршується за наявності квантового шуму. Цей шум може призвести до нестабільного оновлення параметрів, повільної або нестабільної збіжності та поганого узагальнення. Стандартні адаптивні оптимізатори, такі як Adam, коригують швидкість навчання на основі історичної статистики градієнтів, але не розроблені безпосередньо для врахування унікальних характеристик шуму, властивих квантовим вимірюванням.

Ці проблеми стимулювали розробку спеціалізованих стратегій оптимізації з урахуванням шуму, які динамічно адаптуються до невизначеностей квантового обладнання. Серед них запропонована модифікація стандартного алгоритму Quantum Noise-Aware Adam (в подальшому - QNA-Adam) Запропонований оптимізатор Quantum Noise-Aware Adam (QNA-Adam) є модифікацією Adam, що враховує квантовий

вимірювальний шум. Ідея полягає в тому, щоб зменшувати крок навчання для параметрів із високою дисперсією градієнту (тобто сильним шумом), завдяки чому навчання стає стабільнішим.

У цій роботі досліджуються теоретичні основи оптимізації з урахуванням шуму в гібридних квантових нейронних мережах. Вона представляє комплексний аналіз алгоритму QNA-Adam, детально описуючи його мотивацію, математичне виведення, інтеграцію в модульний конвеєр QNN, а також його переваги та обмеження порівняно з існуючими підходами. За допомогою цієї роботи ми прагнемо вдосконалити надійні методології оптимізації для квантового машинного навчання та зробити внесок у практичну життєздатність квантово-розширених алгоритмів на пристроях NISQ.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Квантове машинне навчання (QML): поточний стан та перспективи

Квантове машинне навчання (QML) – це нова міждисциплінарна галузь на перетині квантових обчислень та штучного інтелекту, спрямована на використання квантових ресурсів для покращення завдань машинного навчання. З появою квантових процесорів, здатних виконувати складні алгоритми, що виходять за рамки можливостей класичних комп'ютерів, дослідники почали досліджувати, як квантові явища, такі як суперпозиція, заплутаність та квантовий паралелізм, можна використовувати для обробки, навчання та прогнозування даних ефективніше, ніж класичні методи [1, 2].

Однією з основних парадигм QML є варіаційний квантовий алгоритм (VQA), який використовує параметризовані квантові схеми (також звані ansätze) у поєднанні з класичними циклами оптимізації. VQA особливо підходять для сучасних пристроїв NISQ, оскільки вони спираються на відносно поверхневі схеми та можуть включати методи зменшення помилок [3]. Гібридні квантово-класичні нейронні мережі (QNN) є окремим випадком VQA, де квантова схема використовується як навчальний екстрактор ознак або класифікатор, з класичною попередньою та постобробкою для компенсації апаратних обмежень [4, 5]. Ця модульна структура дозволяє QNN поєднувати виразну силу квантових обчислень з надійністю та масштабованістю класичних фреймворків машинного навчання.

Моделі QML були запропоновані для різноманітних завдань, включаючи контрольовану класифікацію, регресію, генеративне моделювання та квантово-покращене навчання з підкріпленням [2, 6]. Ранні емпіричні дослідження показали, що для певних розподілів даних або відображень ознак квантові моделі можуть представляти складні шаблони ефективніше, ніж класичні мережі порівнянного розміру [7, 8]. Зокрема, гібридні квантові

нейронні мережі (QNN) продемонстрували багатообіцяючі результати на еталонних наборах даних, таких як MNIST, де вони здатні вивчати нетривіальні межі класифікації з набагато меншою кількістю параметрів або глибини схеми, ніж чисто класичні аналоги [4, 9].

Незважаючи на ці досягнення, галузь QML залишається на початковому етапі та стикається зі значними викликами. Більшість практичних квантових пристроїв все ще обмежені шумом, декогеренцією та кількістю кубітів, що не дає реалізувати бажану складність та глибину в квантових моделях [10]. Більше того, теоретичне розуміння того, коли і чому виникає квантова перевага в завданнях машинного навчання, все ще розвивається [1, 11]. Тим не менш, QML залишається сферою інтенсивних досліджень та оптимізму, маючи потенціал для трансформації технологій, керованих даними, у міру розвитку квантового обладнання.

1.2 NISQ: виклики та обмеження

Поточні пристрої для квантових обчислень представлені переважно шумними проміжними квантовими пристроями (NISQ) — квантовими процесорами, що мають від десятків до сотень кубітів, але працюють без повної корекції помилок [10]. Хоча ці системи знаменують собою історичну віху в розробці апаратного забезпечення, їхнє практичне застосування принципово обмежене кількома джерелами шуму та недосконалостей.

Пристрої NISQ схильні до низки помилок, включаючи декогеренцію кубітів, неточність вентиля, перехресні перешкоди, шум зчитування [4, 10].

Ці ефекти обмежують глибину схеми та алгоритмічну складність, які можна надійно реалізувати, накладаючи суворі обмеження на розробку квантових алгоритмів.

Зокрема, варіаційні квантові алгоритми (VQA) та гібридні квантово-класичні нейронні мережі (QNN) повинні бути ретельно спроектовані, щоб відповідати цим апаратно-встановленим обмеженням, зазвичай

використовуючи поверхневі, апаратно-ефективні анзац-схеми та низькокубітні кодування [3, 4, 9].

Однією з найвизначніших проблем ери NISQ є поширеність квантового шуму вимірювань, який зазвичай називають дробовим шумом. Кожне квантове вимірювання дає ймовірнісний результат, тому очікувані значення та градієнти оцінюються шляхом усереднення за багаторазовими виконаннями (пострілами) однієї й тієї ж схеми. Скінченна кількість доступних пострілів вносить статистичну невизначеність у кожне вимірювання, яка може поширюватися протягом процесу оптимізації та призводити до нестабільних або нерівномірних оновлень параметрів [3, 10]. Як результат, навіть найсучасніші класичні оптимізатори можуть працювати погано, оскільки їхні припущення щодо оцінки градієнта з низьким рівнем шуму більше не виконуються в цьому режимі.

Ще одним фундаментальним обмеженням є безплідні плато, явище, коли ландшафт оптимізації стає експоненціально плоским зі збільшенням розміру або глибини схеми. Цей ефект призводить до зникнення величин градієнтів, що робить практично неможливим для локальних оптимізаторів знайти покращені налаштування параметрів — навіть за відсутності апаратного шуму [3, 11]. Сукупний вплив апаратних помилок, дробового шуму та безплідних плато являє собою серйозну перешкоду для масштабування моделей квантового машинного навчання в еру NISQ.

Незважаючи на ці труднощі, ера NISQ сприяла сплеску інновацій в алгоритмічному проектуванні та усуненні помилок. Такі підходи, як поверхневі архітектури схем, повторне завантаження даних, апаратно-ефективні анзац-проекти та оптимізація з урахуванням шуму, стали стандартною практикою в дослідженнях QML [3, 9]. Розробка адаптивних оптимізаторів, які явно враховують квантовий шум вимірювань, таких як QNA-Adam, є критичним кроком до практичного застосування QML на сучасних недосконалих квантових пристроях.

1.3 Оптимізаторя в квантовому машинному навчанні

Оптимізація є центральним стовпом квантового машинного навчання (QML), що лежить в основі навчання варіаційних квантових схем та гібридних квантово-класичних моделей. Метою цих алгоритмів, як правило, є мінімізація функції втрат, визначеної за параметрами квантової схеми, аналогічно класичному машинному навчанню. Однак ландшафт оптимізації в QML принципово відрізняється і часто складніший, ніж у його класичному аналогу, головним чином через властивий шум і стохастичність, присутні в короткостроковому квантовому обладнанні, яке зазвичай називають шумними квантовими пристроями середнього масштабу (NISQ) [10].

Однією з визначальних проблем у цьому контексті є наявність статистичного шуму, що виникає внаслідок скінченних вибірок вимірювань (дробовий шум), на додаток до помилок обладнання та явища безплідних плато, де градієнти експоненціально зникають з розміром схеми [11, 12]. Як результат, методи оптимізації повинні мати справу не лише з неопуклими та багатовимірними просторами параметрів, але й зі значною невизначеністю в оцінці градієнтів.

Традиційні алгоритми оптимізації, такі як стохастичний градієнтний спуск (SGD) та Adam, широко використовуються в дослідженнях QML завдяки їхньому успіху в класичному глибокому навчанні. Тим не менш, їхня продуктивність часто обмежується унікальними шумовими характеристиками квантового обладнання, що призводить до розробки квантово-специфічних та шумоадаптивних оптимізаторів. Нещодавні досягнення призвели до появи постріло-адаптивних оптимізаторів, які динамічно коригують стратегії навчання у відповідь на квантовий шум, забезпечуючи більш надійне та ефективне навчання на пристроях NISQ [13, 14]. Ця постійна еволюція підкреслює вирішальну роль індивідуальних стратегій оптимізації у розкритті практичного потенціалу квантового машинного навчання.

1.4 Класичні оптимізатори в квантовому машинному навчанні

Класичні алгоритми оптимізації залишаються основними інструментами вибору для навчання моделей квантового машинного навчання. Серед них стохастичні градієнтні методи та адаптивні оптимізатори, такі як Adam, були безпосередньо імпортовані з класичного глибокого навчання в квантовий режим. У наступних підрозділах розглядаються їхні принципи та досліджується їхня ефективність і обмеження при застосуванні до параметризованих квантових схем.

1.4.1 Стохастичний градієнтний спуск (SGD)

Стохастичний градієнтний спуск (SGD) – це фундаментальний алгоритм оптимізації в машинному навчанні, який використовується для мінімізації цільових функцій шляхом ітеративного оновлення параметрів у напрямку негативного градієнта [2]. На відміну від пакетного градієнтного спуску, який обчислює градієнти, використовуючи весь набір даних, SGD працює з випадково вибірковими підмножинами (міні-пакетами) або навіть окремими точками даних, що забезпечує швидше оновлення та покращену масштабованість для великих наборів даних.

У контексті квантового машинного навчання (QML) SGD був прийнятий для навчання параметризованих квантових схем, таких як ті, що використовуються у варіаційних квантових алгоритмах та квантових нейронних мережах [6, 7, 9]. Типовий цикл навчання (формула 1.1) включає оцінку градієнтів функції вартості відносно параметрів квантової схеми, а потім оновлення цих параметрів відповідно до правила SGD, де θ - набір параметрів квантової схеми, L – функція втрат, η – швидкість навчання:

$$\theta_{t+1} = \theta_t - \eta \nabla_\theta L(\theta_t), \qquad (1.1)$$

Однак застосування SGD у квантових схемах створює унікальні проблеми, відсутні в класичній оптимізації. Через ймовірнісну природу квантових вимірювань, оцінки градієнтів піддаються дробовому шуму, який виникає через скінченну кількість оцінок схеми, що використовуються для оцінки спостережуваних середніх значень [3, 10]. Ця стохастичність додає додатковий рівень випадковості поверх властивої стохастичності SGD, що потенційно призводить до неточних або дуже мінливих оцінок градієнтів. Крім того, на квантове обладнання впливають шум пристрою та декогеренція, що може ще більше погіршити надійність оновлення параметрів [10, 11]. Як наслідок, поведінка конвергенції SGD в QML може бути нестабільною, і досягнення стабільного навчання часто вимагає ретельного налаштування швидкості навчання та стратегій вибірки. Крім того, зі збільшенням глибини схеми градієнти можуть експоненціально зникати — явище, відоме як проблема безплідного плато, — що ще більше ускладнює оптимізацію [3, 11].

Незважаючи на ці труднощі, SGD залишається широко використовуваною базовою лінією для оптимізації квантових схем і служить основою для розробки більш складних, адаптивних до шуму методів оптимізації в квантовому машинному навчанні [14].

1.4.2 Оптимізатор Adam

Оптимізатор Adam — це адаптивний стохастичний алгоритм оптимізації, який став стандартним вибором у класичному глибокому навчанні завдяки своїй здатності поєднувати переваги імпульсу та адаптивної швидкості навчання [2]. Adam підтримує окремі оцінки першого та другого моментів (середнього значення та нецентрованої дисперсії) градієнтів для кожного параметра, що забезпечує більш ефективну та стабільну конвергенцію, особливо за наявності шумних або розріджених градієнтів.

При застосуванні до моделей квантового машинного навчання Adam часто має перевагу над базовим стохастичним градієнтним спуском завдяки

своїй надійності та простоті використання [7, 9]. Правило оновлення для Adam задається згідно з формулою 1.2.

$$
\begin{aligned}
m_t &= \beta_1 \mathrm{m}_{t-1} + (1-\beta_1)\mathrm{g}_t \\
v_t &= \beta_2 \mathrm{v}_{t-1} + (1-\beta_2) g_t^2 \\
\widehat{m}_t &= \mathrm{m}_t/(1-\beta_1^t) \\
\widehat{v}_t &= \mathrm{v}_t/(1-\beta_2^t) \\
\theta_{t+1} &= \theta_t - \eta \widehat{m}_t/(\sqrt{\widehat{v}_t} + \epsilon)
\end{aligned}
\tag{1.2}
$$

Де g_t – градієнт на кроці t, m_t та v_t – перші та другі оцінки моментів, β_1 та β_2 – швидкості спаду, η – швидкість навчання, а ϵ – мала константа для числової стабільності.

Незважаючи на свою популярність, Adam стикається зі значними проблемами в квантових умовах. Квантові градієнти оцінюються за допомогою скінченної кількості вимірювань, що вносить значний статистичний шум, особливо на пристроях NISQ [3], [10]. Цей шум може призвести до ненадійних оцінок моментів, знижуючи ефективність адаптивного механізму Adam та потенційно спричиняючи нестабільну збіжність або погану кінцеву продуктивність [11]. Крім того, як і у випадку з SGD, Adam не застрахований від феномену безплідного плато, де градієнти зникають для глибоких або високоекспресивних квантових схем, що ще більше перешкоджає оптимізації [3, 11].

Недавні дослідження підкреслили необхідність модифікацій Adam з урахуванням шуму або адаптивних до кадрів модифікацій для вирішення цих проблем у квантовій оптимізації [14]. Тим не менш, класичний оптимізатор Adam залишається ключовою точкою відліку та базовим показником для оцінки продуктивності квантово-специфічних алгоритмів оптимізації.

1.5 Квантові оптимізатори з адаптацією до кількості шотів

Хоча класичні оптимізатори, такі як SGD та Adam, забезпечують відправну точку для навчання моделей квантового машинного навчання, їхня ефективність часто обмежується стохастичним шумом, властивим квантовим вимірюванням. Щоб подолати ці обмеження, було розроблено новий клас оптимізаторів — кадрово-адаптивні квантові оптимізатори, які спеціально розроблені для вирішення проблем скінченної вибірки та шуму в квантових схемах. У наступних підрозділах розглядається мотивація кадрово-адаптивної стратегії та виділяються кілька провідних алгоритмів у цій новій категорії.

1.5.1 Потреба в кадровій адаптивності

Алгоритми квантового машинного навчання спираються на повторні виконання та вимірювання параметризованих квантових схем для оцінки очікуваних значень та градієнтів. Кожне вимірювання за своєю суттю є ймовірнісним, а точність цих оцінок залежить від кількості виконань схеми або «пострілів» [3, 10]. На апаратному забезпеченні NISQ практичні обмеження часто обмежують загальну кількість пострілів, що призводить до значних статистичних коливань, які зазвичай називають дробовим (постріловим) шумом. Скінченна точність вимірювання через обмежену кількість оцінок схеми вносить дисперсію в оцінки градієнтів, що впливає на продуктивність оптимізації [14].

Дробовий шум може приховувати справжній напрямок оптимізації, особливо коли дисперсія оцінки градієнта порівнянна з величиною градієнта або перевищує її [14, 15]. Ця стохастична невизначеність є особливо проблематичною в квантових умовах, де дробовий шум може посилити феномен безплідного плато та збільшити складність оптимізації неопуклих квантових ландшафтів [15, 11]. В результаті, оновлення параметрів на основі градієнта з шумом можуть стати або занадто агресивними, або надмірно

консервативними, що в результаті призводить до повільної або нестабільної збіжності [3, 14]. Класичні оптимізатори, такі як SGD та Adam, не враховують цей квантово-специфічний шум, що може призвести до неефективного або нестабільного навчання при безпосередньому застосуванні до квантових схем.

Визнаючи це, були розроблені адаптивні до вимірювань квантові оптимізатори, які динамічно коригують стратегії навчання (швидкість навчання, розмір кроку або розподіл кадрів, на основі спостережуваної дисперсії градієнта або рівня шуму). Нещодавно з'явилося кілька адаптивних до квантового шуму алгоритмів для вирішення труднощів, спричинених шумом вимірювань та скінченною вибіркою на пристроях NISQ.

1.5.2 iCANS (Ітеративний спряжений адаптивний шумовий розв'язувач)

Алгоритм iCANS був одним з перших, який запропонував явну адаптацію до квантового шуму та обмежених ресурсів вимірювань. Замість того, щоб покладатися на фіксовану кількість кадрів для всіх параметрів та оновлень, iCANS динамічно розподіляє кадри під час процесу оптимізації, щоб збалансувати компроміс між обчислювальними витратами та точністю оцінки градієнта. Метод використовує адаптивні спряжені напрямки, використовуючи більше кадрів, де градієнтний сигнал слабкий, і менше, де він сильний, ефективно розподіляючи квантові ресурси для покращення оптимізації [4]. Емпіричні дослідження показали, що iCANS може досягати конкурентоспроможних результатів зі зменшеним загальним бюджетом кадрів, що робить його добре придатним для експериментів ери NISQ.

1.5.3 Rosalin (Надійне адаптивне навчання до кадрів у шумних квантових схемах)

Rosalin представила механізм оновлення з урахуванням дисперсії для оптимізації квантових схем. Оцінюючи дисперсію градієнта в режимі

реального часу, Rosalin модулює швидкість навчання та може адаптивно розподіляти більше кадрів на параметри з вищою невизначеністю. Цей адаптивний підхід безпосередньо спрямований на негативний вплив дробового шуму, стабілізуючи оптимізацію та допомагаючи уникнути розбіжностей, спричинених надмірно шумними оновленнями. Розробка Rosalin особливо ефективна для схем та задач, де шум є просторово або часово неоднорідним [16]. Метод продемонстрував покращену стійкість як у симульованих, так і в апаратних експериментах, демонструючи практичні переваги стратегій кадрово-адаптивного навчання.

1.5.4 ShotAdaptiveOptimizer від PennyLane

У рамках квантового машинного навчання PennyLane, ShotAdaptiveOptimizer пропонує прагматичну реалізацію кадрово-адаптивної оптимізації для варіаційних квантових алгоритмів. Цей оптимізатор контролює дисперсію в оцінках градієнта зсуву параметрів та збільшує кількість кадрів вимірювання для параметрів з високою дисперсією, одночасно зменшуючи кількість кадрів, коли оцінки стабільні [17]. Стратегія дозволяє здійснювати точніші оновлення параметрів без витрачання квантових ресурсів, що призводить до ефективнішого навчання. Завдяки автоматичному налаштуванню розподілу кадрів у відповідь на шум, ShotAdaptiveOptimizer особливо привабливий для пристроїв NISQ з обмеженим бюджетом на кадри та змінною продуктивністю схеми.

Важливо зазначити, що всі ці адаптивні до кадрів оптимізатори мають спільну мету: вони прагнуть зробити вимірювання ефективнішим, динамічно регулюючи розподіл кадрів на основі дисперсії або невизначеності градієнта. Однак, вони зазвичай не змінюють основну логіку того, як оптимізатор оновлює параметри, використовуючи виміряні градієнти — ці оптимізатори все ще використовують стандартні правила оновлення, такі як SGD або Adam. Це контрастує з оптимізаторами, такими як QNA-Adam, які не тільки

адаптують стратегії вимірювання, але й фундаментально змінюють самі правила оновлення параметрів, щоб явно включити інформацію про шум у динаміку навчання.

У сукупності, адаптивні до кадрів алгоритми є передовим етапом квантово-свідомої оптимізації, ілюструючи, як адаптивність до дробового шуму стає фундаментальною вимогою для масштабованого квантового машинного навчання.

1.5.5 KOALA (Оптимізатор Калмана для адаптивного налаштування швидкості навчання)

KOALA додатково узагальнює оптимізатори на основі Калмана для квантового машинного навчання [18]. Він застосовує фільтрацію Калмана спільно до оцінок параметрів та градієнтів, і може динамічно адаптувати як швидкість навчання, так і кількість пострілів на параметр та ітерацію. Розподіляючи квантові ресурси у відповідь як на шум вимірювання, так і на невизначеність параметрів, KOALA досягає високої ефективності використання ресурсів та стійкості на пристроях NISQ. Цей метод встановив нові стандарти стабільності оптимізації та ефективності пострілів, особливо у варіаційних квантових алгоритмах, таких як VQE та QAOA.

1.5.6 KAdam (Kalman Adam)

KAdam використовує інший підхід, інтегруючи фільтрацію Калмана в оптимізаційну структуру Adam [17]. У KAdam як середнє значення, так і невизначеність кожного параметра поширюються та оновлюються за допомогою рекурсій Калмана. Алгоритм адаптивно зменшує швидкість навчання для кожного параметра залежно від оціненої надійності вимірювання градієнта. KAdam є статистично принциповим, використовуючи байєсівський висновок для оптимального балансу нової та попередньої інформації.

Емпіричні дослідження показали, що KAdam пропонує значні покращення в стійкості та збіжності, особливо в умовах сильного дробового шуму.

1.6 Вплив шуму вимірювання та аргументи

Однією з визначальних рис квантових обчислень на пристроях NISQ є притаманна стохастичність результатів вимірювань. На відміну від класичних обчислень, де змінні можна зчитувати майже з ідеальною точністю, квантові вимірювання дають ймовірнісні результати, що визначаються станом системи та обраним базисом вимірювання. Щоб отримати очікувані значення та градієнти для навчання моделей квантового машинного навчання, один і той самий квантовий цикл має бути виконаний кілька разів — кожне повторення відоме як «постріл» — з усередненням результатів для оцінки спостережуваних величин [3, 10].

Цей процес, хоча й є фундаментальним для квантових обчислень, вводить шум вимірювання: статистичні коливання, що виникають через скінченну кількість виконань циклу. Вплив цього шуму особливо сильний у варіаційних квантових алгоритмах та гібридних квантово-класичних нейронних мережах, де оптимізація спирається на точну оцінку градієнтів для оновлення параметрів циклу. Недостатнє усереднення, через обмежену кількість кадрів або апаратні обмеження, може призвести до дуже шумних оцінок градієнтів, що спричиняє нестабільні або нестабільні траєкторії оптимізації [3, 4, 9].

Класичні адаптивні оптимізатори, такі як Adam, частково пом'якшують шумні оновлення, масштабуючи швидкість навчання відповідно до дисперсії минулих градієнтів. Однак ці алгоритми не були розроблені для явного розрізнення дисперсії, спричиненої ландшафтом оптимізації (сигналом), та дисперсії, що виникає внаслідок квантового шуму вимірювань [3, 10]. Як результат, при застосуванні до квантових параметрів вони можуть занадто

повільно або неефективно реагувати на флуктуації, які зумовлені виключно скінченною вибіркою, що часто призводить до марних обчислювальних зусиль, повільної збіжності або, в гіршому випадку, дивергенції.

Недавні дослідження підкреслили необхідність стратегій оптимізації з урахуванням шуму, які динамічно адаптуються до невизначеності, властивої квантовим вимірювання [3, 12]. Один із підходів полягає у виділенні більшої кількості кадрів для вимірювань або градієнтів з вищою дисперсією, тим самим зменшуючи їхній шум, але ціною збільшення споживання ресурсів. Інше, більш масштабоване рішення полягає у модифікації самого алгоритму оптимізації таким чином, щоб він штрафував оновлення в напрямках, де виміряні градієнти особливо невизначені. Оптимізатор Quantum Noise-Aware Adam (QNA-Adam) є прикладом цього підходу: шляхом явного включення спостережуваної дисперсії градієнта кожного параметра на кожній ітерації, він знижує швидкість навчання там, де шум високий, тим самим стабілізуючи та прискорюючи навчання [12].

Такі алгоритми з урахуванням шуму мають вирішальне значення не лише для досягнення надійної конвергенції на апаратному забезпеченні NISQ, але й для ефективного використання обмежених квантових ресурсів. Вони дозволяють практикам навчати глибші або більш виразні квантові моделі в межах одного бюджету кадрів, прокладаючи шлях для практичного квантового машинного навчання в найближчій перспективі. Оскільки квантові процесори продовжують удосконалюватися, ці методи, ймовірно, залишатимуться важливими для подолання розриву між можливостями апаратного забезпечення та алгоритмічними амбіціями.

1.7 Прогалини в дослідженнях та внесок цієї роботи

Хоча за останні роки досягнуто значного прогресу в квантовому машинному навчанні та розробці методів адаптивної оптимізації, взаємодія між квантовим шумом вимірювань та алгоритмічною конвергенцією

залишається важливою відкритою проблемою. Більшість сучасних досліджень зосереджені або на апаратно-ефективних квантових схемах, або на адаптації класичних оптимізаторів, таких як стохастичний градієнтний спуск (SGD), Adam та квантовий природний градієнт (QNG) до квантової області [3, 5, 9]. Хоча ці методи забезпечують основу для навчання варіаційних квантових алгоритмів та гібридних квантово-класичних нейронних мереж, вони не повністю вирішують унікальні проблеми, що виникають через квантовий дробовий шум та стохастичну градієнтну невизначеність, особливо ті, що зустрічаються на пристроях NISQ [3, 10, 12].

Класичні оптимізатори, такі як Adam, спочатку були розроблені для середовищ, де оцінки градієнтів є точними та лише незначно стохастичними. Однак у квантовій оптимізації градієнти можуть бути сильно спотворені шумом у результатах вимірювань, що призводить до високої дисперсії та потенційно оманливих оновлень параметрів. Існуючі адаптивні оптимізатори реагують, усереднюючи минулі градієнти, але їх коригування може бути недостатнім в умовах швидкозмінного або нестаціонарного квантового шуму [12]. Інші підходи, такі як методи адаптації до пострілів, включаючи iCANS, Rosalin та ShotAdaptiveOptimizer від PennyLane [13-15], адаптивно змінюють кількість оцінок схеми («пострілів»), що використовуються для оцінки градієнта, на основі спостережуваної дисперсії градієнта. Хоча це підвищує ефективність вимірювання, ці методи все ще покладаються на фіксоване правило оновлення параметрів у самому оптимізаторі, незалежно від фактичного шуму, виміряного в режимі реального часу.

Більш просунуті фреймворки, такі як KAdam та KOALA, використовують фільтрацію Калмана для поширення як значень параметрів, так і оцінок невизначеності під час навчання, і навіть можуть динамічно адаптувати кількість пострілів [18, 19]. Ці методи пропонують статистично принциповий підхід до адаптації шуму та показали високу стійкість у певних варіаційних квантових алгоритмах. Однак вони вносять додаткову складність, вимагають ретельного моделювання статистики шуму та можуть бути не

такими простими для реалізації або інтерпретації, як методи, подібні до Adam.

Отже, залишається ключова прогалина: жоден широко прийнятий оптимізатор не адаптує свою внутрішню логіку оновлення параметрів на основі явних вимірювань квантового шуму в реальному часі та при цьому зберігає практичну простоту та інтерпретованість класичних оптимізаторів, таких як Adam. Квантовий шум є не тільки випадковим, але й може бути систематично зміщений, і ефективний оптимізатор повинен вирішувати проблему шуму на алгоритмічному рівні — не лише за допомогою стратегій збору даних або розподілу вимірювань.

Головна мета цієї роботи — подолати цю прогалину, запропонувавши та систематично проаналізувавши оптимізатор Quantum Noise-Aware Adam (QNA-Adam), розширення Adam, яке явно включає вимірювання шуму для кожного параметра в процес адаптації швидкості навчання. QNA-Adam зберігає знайомі механізми відстеження моментів та корекції зміщення Adam, але безпосередньо коригує швидкість навчання для кожного параметра відповідно до виміряної дисперсії відповідного градієнта, визначеної квантовим дробовим шумом під час оцінок зсуву параметрів. Цей підхід пропонує кілька ключових переваг:

- він стабілізує навчання в режимах з високим рівнем шуму, не вимагаючи збільшення бюджетів на вимірювання;
- забезпечує прозорість та інтерпретованість того, як шум впливає на оновлення параметрів;
- такий підхід можна легко реалізувати в існуючих гібридних конвеєрах навчання QNN.

Базуючись на теорії баєсівського оцінювання та натхненна принципами фільтра Калмана [18, 19], ця робота демонструє, як QNA-Adam може бути безшовно інтегрована в модульні квантово-класичні архітектури для практичних завдань, таких як класифікація зображень. Підсумовуючи, основні досягнення цієї роботи такі:

- детальне дослідження впливу квантового шуму вимірювань на оптимізацію гібридної QNN та критичний аналіз обмежень існуючих класичних та квантово-специфічних оптимізаторів, включаючи адаптивні методи та методи на основі Калмана;
- виведення та обґрунтування оптимізатора QNA-Adam, який адаптує швидкість навчання для кожного параметра до спостережуваної дисперсії градієнта, з практичними рекомендаціями щодо вибору гіперпараметрів;
- розробка модульної гібридної архітектури QNN, адаптованої для апаратного забезпечення NISQ, що демонструє ефективну інтеграцію QNA-Adam у наскрізний цикл навчання;
- систематичне порівняння QNA-Adam з сучасними альтернативами, з аналізом збіжності, робустності та ефективності вимірювань в умовах реалістичного шуму;
- надання як теоретичних знань, так і практичних інструментів для підтримки масштабованого та ефективного квантового машинного навчання, що стосується як моделювання, так і майбутніх експериментів з реальним обладнанням.

Завдяки цим внескам ця робота просуває сучасний рівень оптимізації квантового машинного навчання, пропонуючи як міцну теоретичну основу, так і практичні рекомендації для майбутніх досліджень та практичного впровадження.

Попередні результати цієї роботи були представлені у вигляді тез на конф. «АПКН-2025» [20].

2 ТЕОРЕТИЧНІ ОСНОВИ ТА ВИВЕДЕННЯ АЛГОРИТМІВ

Розробка надійних та ефективних стратегій оптимізації для гібридних квантово-класичних нейронних мереж вимагає чіткого розуміння як механіки квантових схем, так і класичних парадигм навчання, з якими вони взаємодіють. У цьому розділі встановлюється теоретична основа, необхідна для аналізу та виведення квантових алгоритмів оптимізації з урахуванням шуму.

Ми починаємо з представлення параметризованих квантових схем (PQC) та їхньої центральної ролі у варіаційних квантових алгоритмах (VQA), які формують обчислювальне ядро більшості моделей квантового машинного навчання. Далі ми розглядаємо гібридну парадигму квантово-класичного навчання, висвітлюючи, як квантові схеми та класичні оптимізатори взаємодіють у сучасних конвеєрах машинного навчання. Далі йде обговорення оцінки градієнта в квантових схемах, зосереджуючись на статистичних характеристиках результатів вимірювань та фундаментальній проблемі дробового шуму. Нарешті, ми окреслюємо унікальні особливості ландшафтів оптимізації в квантовому машинному навчанні, включаючи феномен безплідного плато, який мотивує потребу в алгоритмах оптимізації, стійких до шуму.

Викладаючи ці фундаментальні концепції, ми забезпечуємо необхідний контекст для розуміння подальшого виведення та обґрунтування оптимізатора Quantum Noise-Aware Adam (QNA-Adam) та його інтеграції в практичні конвеєри квантового машинного навчання.

2.1.1 Параметризовані квантові схеми та варіаційні квантові алгоритми

Параметризовані квантові схеми (PQCs) є фундаментальними будівельними блоками більшості моделей квантового машинного навчання.

PQC складається з послідовності квантових вентилів — деякі фіксовані, деякі з налаштовуваними параметрами — що застосовуються до регістра кубітів. Ці параметри, зазвичай позначені як $\theta = (\theta_1, \theta_2, \ldots, \theta_N)$, кодують «навчальну» частину квантової моделі [3, 6].

У математичних термінах PQC реалізує унітарне перетворення U(θ) на початковому квантовому стані $|0\rangle^{\otimes n}$, створюючи параметризований квантовий стан (формула 2.1).

$$|\psi(\theta)\rangle = U(\theta)|\rangle^{\otimes n} \quad (2.1)$$

Тут n – це кількість кубітів у системі. Спостережувані величини – величини, що становлять інтерес (наприклад, результати вимірювань), оцінюються шляхом багаторазової підготовки та вимірювання цього стану, що дає очікувані значення (формула 2.2).

$$\langle O\rangle_\theta = \langle\psi(\theta)|O|\psi(\theta)\rangle \quad (2.2)$$

Тут O – хермітів оператор, що представляє спостережувану величину. Концепція VQA розширює PQC класичним циклом оптимізації [3]. У типовому ВКА класичний оптимізатор пропонує оновлення параметрів θ, щоб мінімізувати (або максимізувати) задану функцію вартості, яка зазвичай визначається як очікуване значення однієї або кількох квантових спостережуваних величин. Цей ітераційний процес утворює гібридний квантово-класичний цикл, де квантовий комп'ютер оцінює функцію вартості, а класичний комп'ютер виконує оновлення параметрів.

Виразність PQC визначається його ансатцем конкретною локацією та вибором параметризованих та фіксованих вентилів. Апаратно-ефективні ансатці розроблені відповідно до власного набору вентилів та зв'язності квантового обладнання, зменшуючи глибину схеми та ефекти декогеренції, а проблемно-інспіровані ансатці включають знання структури задачі [3, 6, 9].

VQA та PQC є центральними для широкого кола квантових застосувань, від оцінки енергії основного стану в квантовій хімії до класифікації та регресії в квантовому машинному навчанні [3, 4, 6]. Їхня модульна та гнучка структура робить їх особливо придатними для сучасних пристроїв NISQ, де глибина схеми та зменшення помилок є критичними міркуваннями.

2.1.2 Гібридне квантово-класичне машинне навчання

Гібридна квантово-класична парадигма навчання використовує сильні сторони як квантових, так і класичних обчислень, щоб подолати поточні обмеження квантового обладнання та забезпечити масштабовані програми машинного навчання [3, 4, 14]. У цьому підході квантові комп'ютери використовуються для завдань, які можуть отримати вигоду від квантового паралелізму або заплутаності, таких як оцінка параметризованих квантових схем та оцінка квантових спостережуваних величин, тоді як класичні комп'ютери обробляють завдання, що потребують великого обсягу пам'яті, розширеної логіки керування або усталених методів оптимізації.

Типовий гібридний робочий процес структурований наступним чином. Спочатку класичні дані проходять попередню обробку, таку як зменшення розмірності або нормалізація, щоб підготувати їх до квантового кодування. Потім оброблені дані відображаються в квантові стани за допомогою схеми кодування. Квантове обладнання виконує параметризовану квантову схему (PQC), яка генерує результати вимірювань, що використовуються для оцінки очікуваних значень спостережуваних величин, що стосуються завдання навчання. Ці результати потім піддаються постобробці та перетворюються на значення витрат або втрат, яке кількісно визначає продуктивність моделі. Зрештою, класичний оптимізатор оновлює параметри схеми на основі виміряних градієнтів або втрат, і процес повторюється в ітераційному циклі до досягнення збіжності.

Цей гібридний цикл можна підсумувати наступними кроками:

- класична попередня обробка: зменшення або перетворення даних у квантово-сумісний формат;
- квантова оцінка: виконання PQC з поточними параметрами на квантовому обладнанні;
- вимірювання: вибірка вихідних даних для оцінки спостережуваних величин або градієнтів;
- класична постобробка: Обчислення втрат/витрат та (за потреби) зіставлення квантових вихідних даних з класичними прогнозами;
- класична оптимізація: Оновлення параметрів за допомогою класичних алгоритмів (наприклад, Adam, QNA-Adam).

Модульна структура гібридного квантово-класичного навчання є особливо вигідною для пристроїв епохи NISQ. Вона дозволяє підтримувати квантові схеми менш поверхневими, пом'якшуючи вплив декогеренції та апаратного шуму, водночас використовуючи зрілі можливості класичної оптимізації та обробки даних. Ця філософія проектування лежить в основі багатьох сучасних алгоритмів квантового машинного навчання, включаючи варіаційні квантові класифікатори, квантові згорткові нейронні мережі та методи квантового ядра [3, 4, 14].

2.1.3 Оцінка градієнта в квантових схемах

Ефективне навчання параметризованих квантових схем вимагає точної оцінки градієнтів відносно параметрів схеми. У квантовому машинному навчанні найпоширенішим методом обчислення градієнтів є правило зсуву параметрів, яке дозволяє оцінювати похідну очікуваного значення відносно параметра вентиля шляхом виконання скінченного набору оцінок квантових схем зі зміщеними параметрами [3, 6, 14].

Враховуючи функцію вартості, визначену як очікуване значення спостережуваної O відносно параметризованого стану $|\psi(\theta)\rangle$, як введено в

рівнянні (2.1), градієнт відносно параметра θ_i можна оцінити за допомогою правила зсуву параметрів, де s — фіксований зсув параметра (часто $s = \pi/2$), а всі інші параметри залишаються постійними (формула 2.3).

$$\frac{(\partial C)}{\partial \theta_i} = \left(\frac{1}{2}\right)[C(\theta_i + s) - C(\theta_i - s)], \quad (2.3)$$

На практиці значення C(θ) оцінюється за допомогою повторних вимірювань («пострілів») виходу квантової схеми. Кожен окремий постріл дає випадковий результат, тому середнє значення багатьох таких вимірювань забезпечує емпіричну оцінку очікуваного значення. Скінченна кількість пострілів вносить статистичну невизначеність — дробовий шум — в оцінку градієнта, що призводить до дисперсії, яка масштабується обернено пропорційно кількості вимірювань [3], [10], [12]. Ця дисперсія стає домінуючим джерелом шуму в квантовій оптимізації, особливо зі збільшенням глибини схеми або оскільки доступні квантові апаратні ресурси обмежують кількість пострілів на вимірювання.

Існує кілька стратегій для зменшення градієнтного шуму, включаючи збільшення кількості пострілів (за рахунок обчислювальних ресурсів), використання методів групування вимірювань або використання адаптивних до пострілів алгоритмів, які розподіляють більше вимірювань на градієнти з високою дисперсією [16] [17]. Однак, у більшості практичних сценаріїв квантового машинного навчання, градієнтні оцінки залишаються значно більш шумними, ніж у класичних умовах, що робить надійну оптимізацію з урахуванням шуму необхідною для ефективного навчання.

2.1.4 Оптимізаційні ландшафти та безплідні плато

Ефективність оптимізації в квантовому машинному навчанні критично залежить від геометрії ландшафту функції вартості, визначеного параметризованими квантовими схемами. На відміну від класичних

нейронних мереж, де поверхні вартості можуть бути нерівними, але зазвичай доступними для навігації за допомогою сучасних оптимізаторів, квантові моделі можуть демонструвати області, де градієнти стають зникаюче малими — явище, відоме як проблема безплідного плато[11, 15].

Безплідне плато виникає, коли дисперсія градієнта функції вартості відносно параметрів схеми експоненційно зменшується з розміром системи або глибиною схеми. Це призводить до того, що оптимізатор отримує мало інформації або взагалі її не отримує, що зупиняє навчання незалежно від обраної стратегії оптимізації. Математично безплідне плато описується згідно з формулою 2.4.

$$Var\left(\frac{(\partial C)}{\partial \theta_i}\right) \to 0 \qquad (2.4)$$

Тут C — функція вартості, де дисперсія зменшується експоненційно з кількістю кубітів або глибиною схеми, що є типовою ознакою безплідного плато. Безплідні плато спостерігалися в різних умовах, включаючи випадково ініціалізовані глибокі квантові схеми та певні типи ансатце. Вони особливо поширені в неструктурованих або апаратно-ефективних схемах, де заплутаність та експресивність схеми високі, але величини градієнтів швидко зменшуються [11]. Важливо, що наявність шуму — чи то від недосконалих вентилів, декогеренції, чи вимірювання кінцевим кадром (шум пострілу) — може посилити ефект безплідного плато, що ще більше ускладнює оптимізацію на реальному обладнанні NISQ [15].

Пом'якшення безплідних плато залишається активною сферою досліджень. Стратегії включають використання проблемно-інспірованих або неглибоких ансатце, пошарове навчання, ретельну ініціалізацію та алгоритми оптимізації з урахуванням шуму, стійкі як до зникаючих, так і до шумових градієнтів. Розуміння взаємодії між проектуванням схеми, шумом вимірювання та геометрією ландшафту оптимізації є важливим для розвитку практичного квантового машинного навчання.

2.2 Математичне формулювання навчання гібридної квантово-класичної нейронної мережі (QNN)

Гібридні квантово-класичні нейронні мережі (QNN) спираються на тісний зв'язок між квантовими обчисленнями та класичними процедурами оптимізації. У цьому розділі формалізовано структуру, що лежить в основі гібридних моделей, визначено цілі оптимізації та введено позначення для оцінки градієнта шуму, що створює основу для оптимізатора, який буде представлено пізніше.

2.2.1 Функція вартості в гібридних квантово-класичних нейронних мережах

У варіаційному квантовому навчанні тренований квантовий модуль будується шляхом застосування параметризованого унітарного оператора U(θ) до квантового стану, що кодує вхід |ϕ(x)⟩, утворюючи правило 2.5.

$$|\psi(\theta, x)\rangle = U(\theta)|\phi(x)\rangle \tag{2.5}$$

,де $\theta \in \mathbb{R}^p$ — параметри моделі, а x — класичний вхід. Вихід моделі отримують шляхом вимірювання ермітового оператора $\hat{O}$, що дає значення функції вартості 2.6.

$$C(\theta;\, x) = \langle\psi(\theta, x)|\ \hat{O}\ |\psi(\theta, x)\rangle \tag{2.6}$$

На практиці це математичне сподівання точно невідоме — воно оцінюється шляхом багаторазових квантових вимірювань. Для M вимірювань оцінка $\tilde{C}$ має вигляд 2.7.

$$E(\bar{C}) = C(\theta;\, x),\ Var(\bar{C}) = \frac{Var[O]}{M} \qquad (2.7)$$

Такий статистичний шум від вимірювань безпосередньо впливає на оцінки градієнта, вводячи незмінну нижню межу дисперсії, яка зберігається навіть при багатьох ітераціях — це ключове обмеження для стандартних оптимізаторів і основна мотивація для нашого методу QNA-Adam.

Коли мова йде про задачу з учителем, квантовий вихід моделі може бути переданий через класичну функцію втрат L, де y — це правильна мітка (ground truth). В інших випадках, таких як мінімізація енергії, саме квантове математичне сподівання виступає цільовою функцією для оптимізації (формула 2.8).

$$L(\theta;\, x, y) = L(C(\theta;\, x), y) \qquad (2.8)$$

2.2.2 Оцінка градієнта та шуму вимірювання

У гібридній квантовій оптимізації градієнти квантової функції втрат C(θ) використовуються для оновлення параметрів θ. Зазвичай ці градієнти оцінюють за допомогою правила зсуву параметра, яке дає точний вираз для похідної очікуваного значення, коли квантові елементи побудовані на генераторних унітарних операторах із $G^2 = I$. Для одного параметра θ_i градієнт обчислюється як згадано в формулі 1.3, де де $\delta = \pi/2$, а всі інші параметри залишаються фіксованими. Для кожного параметра потрібно виконати два незалежних запуску квантової схеми. Кожне значення $C(\theta_i \pm \delta)$ не визначене точно, а оцінюється емпірично — тобто, шляхом виконання квантової схеми M разів (пострілів, shots) та усереднення результатів вимірювань спостережуваного Ô. Таким чином, отримується оцінювач із шумом , що має форму 2.9, де $o_j \in spec(\hat{O})$ — результати вимірювань (наприклад, ±1 для Паулі-Z операторів).

$$\tilde{C}(\theta_i \pm \delta) = 1/\mathrm{M} \sum_{j=1}^{M} o_j \quad (2.9)$$

Таким чином, оцінка градієнта є випадковою величиною (враховуючи рівняння 2.7) із дисперсією. Тепер рівняння набуває вигляду 2.10.

$$Var[\tilde{g}_i] = 1/4M(Var[O_+] + Var[O_-]) \quad (2.10)$$

Ця дисперсія, викликана дробовим шумом, принципово відрізняється від класичного міні-пакетного шуму. Вона не зменшується з епохами, якщо не використовується більше квантових ресурсів (пострілів), вона є постійною та адитивною, забруднюючи кожну оцінку градієнта, її можна аналітично апроксимувати та відняти, що мотивує оптимізацію з урахуванням шуму.

Зокрема, класичні оптимізатори, такі як Адам, інтерпретують дисперсію величин градієнтів як кривизну або невизначеність і відповідно коригують швидкість навчання. Однак, коли ця дисперсія зумовлена квантовим шумом вимірювань, це призводить до систематичного недооцінювання швидкості навчання, особливо коли навчання наближається до конвергенції. Це спостереження забезпечує основу для QNA-Adam, який безпосередньо компенсує шум вимірювань під час його оновлення у другий момент.

Ми спираємося на цю формальну модель градієнтного шуму, щоб вивести правило оновлення QNA-Adam у розділі 2.5.

2.3 Теоретична мотивація для швидкості навчання з урахуванням шуму

2.3.1 Градієнтний шум у квантовій оптимізації

У класичному машинному навчанні стохастичність оцінок градієнтів зазвичай виникає внаслідок випадкової вибірки пакетів даних або шуму міток. Ці форми шуму є алгоритмічними або залежними від набору даних, і їх статистичну поведінку часто можна пом'якшити за допомогою більших

розмірів пакетів або регуляризації. Однак у квантовому машинному навчанні оцінки градієнтів додатково спотворюються шумом, викликаним вимірюванням, що є фундаментальним наслідком процесу квантового вимірювання та зберігається навіть за ідеальних умов.

Як було представлено в розділі 2.1.3, оцінка градієнтів у варіаційних квантових алгоритмах зазвичай виконується за допомогою правила зсуву параметрів, яке вимагає багаторазового виконання квантової схеми для оцінки очікуваного значення спостережуваної величини. Кожне виконання, яке називається пострілом, дає один ймовірнісний результат, а очікуване значення оцінюється шляхом усереднення за скінченим числом S таких пострілів. Отриману оцінку градієнта для параметра θ_i на ітерації t можна записати у вигляді 2.11.

$$\tilde{g}_i^{(t)} = E\left[g_i^{(t)}\right] + \epsilon_i^{(t)} \tag{2.11}$$

де $E\left[g_i^{(t)}\right]$ — це істинний градієнт (тобто, математичне сподівання при нескінченній кількості вимірювань), а $\epsilon_i^{(t)}$ — випадкова змінна, що описує шум пострілів (shot noise). Дробовий шум виникає через постулат квантової проекції: вимірювання руйнує квантовий стан, і результат виводиться з розподілу ймовірностей, який залежить від поточних параметрів кола. Отже, навіть повторні оцінки одного й того ж кола дають різні результати, і дисперсія оціненого градієнта $\tilde{g}_i^{(t)}$ стає критичною величиною.

Важливо, що цей тип шуму не тільки динамічно змінюється під час навчання, оскільки параметри схеми змінюються на кожній ітерації, але й відрізняється залежно від параметрів, деякі градієнти можуть бути добре оцінені, тоді як інші мають високий рівень шуму, а ще не може бути довільно зменшений на пристроях NISQ через обмеження апаратного часу, декогеренцію або обмежений бюджет на виконання.

У таких умовах стає важливим розрізняти величину градієнтного компонента та нашу впевненість у його оцінці. Без цього розмежування оптимізатори можуть робити великі оновлення в ненадійних напрямках, що призводить до нестабільності, розбіжності або неефективного навчання. Це створює умови для розробки оптимізаторів, які явно враховують цей фактор, таких як алгоритм Adam з квантовим шумом, обговорений у наступних розділах.

2.4 Аргументи на користь адаптації швидкості навчання з урахуванням шуму

Класичні оптимізатори, такі як Adam, RMSProp або SGD, адаптують швидкість навчання на основі величин градієнтів або ковзних середніх квадратів градієнтів. Хоча ці алгоритми ефективні в багатьох класичних умовах, вони не враховують явно статистичну достовірність оцінок градієнтів. У квантовому режимі, де градієнти спотворюються дробовим шумом, викликаним вимірюванням (див. розділ 2.4.1), ця відсутність чутливості до шуму може призвести до нестабільних оновлень або марних витрат зусиль на навчання.

Щоб вирішити цю проблему, ми пропонуємо модулювати швидкість навчання дисперсією кожної оцінки градієнта. Інтуїція проста: коли градієнт вимірюється з високою невизначеністю, відповідне оновлення слід зменшувати, щоб уникнути поширення шуму. І навпаки, оновлення, засновані на більш надійних (низькодисперсійних) градієнтах, повинні мати можливість виконуватися з повним кроком.

Це приводить до розкладу швидкості навчання, адаптивного до дисперсії (variance-adaptive learning rate schedule), що є ключовим для оптимізатора Quantum Noise-Aware Adam (QNA-Adam). У QNA-Adam швидкість навчання для кожного параметра $\alpha_i^{(t)}$ на ітерації t визначається за формулою 2.12.

$$\alpha_i^{(t)} = \frac{\alpha_0}{1 + \lambda\sigma_{\{i,t\}}^2} \qquad (2.12)$$

де:

- α_0 — базова швидкість навчання (як у Adam);

- $\lambda > 0$ — налаштовуваний гіперпараметр чутливості до шуму;

- $\sigma_{\{i,t\}}^2$ — емпірично оцінена дисперсія градієнта $\tilde{g}_i^{(t)}$ для параметра θ_i на кроці t.

Ця формула гарантує:

- для напрямків з високим рівнем шуму $\left(\sigma_{\{i,t\}}^2 \gg 1\right)$ швидкість навчання пропорційно зменшується, запобігаючи хаотичним оновленням;

- для напрямків з низьким рівнем шуму ($\sigma_{\{i,t\}}^2 \to 0$) правило оновлення наближається до стандартного кроку Adam з швидкістю навчання α_0.

Цей підхід розглядає дисперсію градієнта як обернену величину довірчої впевненості: чим вищий шум, тим менше оптимізатор «довіряє» цьому напрямку. В результаті навчання стає більш надійним, з адаптивним демпфуванням нестабільних оновлень, які в іншому випадку могли б дестабілізувати навчання в режимах NISQ.

Важливою перевагою цього графіка швидкості навчання є його простота: його можна обчислити з мінімальними накладними витратами, використовуючи ту саму емпіричну дисперсію, яка вже доступна з оцінки градієнта на основі пострілів. Структура оптимізатора залишається практично незмінною, що дозволяє легко інтегруватися в існуючі квантово-класичні робочі процеси.

Ця схема з урахуванням дисперсії є відмінністю QNA-Adam, що відрізняє його від звичайних оптимізаторів та від складніших підходів на основі Калмана.

2.4.1 Порівняння з адаптивними методами пострілів, QNG, KOALA та Kadam

Зростаюча кількість досліджень визнає обмеження класичних оптимізаторів у квантовому середовищі, особливо коли вони стикаються з шумом вимірювань та обмеженнями ресурсів на пристроях NISQ. У відповідь з'явилося кілька підходів, які намагаються зробити оптимізацію більш надійною шляхом адаптації або процесу вибірки (розподілу вимірів), або самої логіки оновлення. У цьому розділі ми порівнюємо QNA-Adam з чотирма основними категоріями пов'язаних методів: адаптивними оптимізаторами, квантовими методами природних градієнтів (QNG) та двома адаптивними оптимізаторами, натхненними Калманом, KOALA та KAdam.

2.4.1.1 Адаптивні оптимізатори

Оптимізатори, такі як ShotAdaptiveOptimizer від PennyLane та Rosalin [16, 17], зосереджені на більш інтелектуальному розподілі ресурсів вимірювань. Ці алгоритми динамічно збільшують кількість вимірів для компонентів градієнта з вищою оціненою дисперсією. Ключова ідея полягає в зменшенні шуму шляхом витрачання більшого бюджету вимірювань на «невизначені» напрямки, зберігаючи при цьому ресурси там, де градієнти стабільні.

Однак, хоча методи адаптації до кадрів покращують фазу збору даних, вони не змінюють внутрішню логіку оновлення оптимізатора. Як правило, після оцінки градієнтів (більш-менш точно), ці оптимізатори все одно передають їх без змін до класичного правила оновлення (наприклад, стандартного SGD або Adam). Таким чином, хоча вони допомагають зменшити шум, вони залишаються вразливими до неправильної інтерпретації залишкових коливань градієнта як справжнього сигналу.

QNA-Adam принципово відрізняється тим, що залишає «бюджет» вимірювання незмінним, але безпосередньо змінює величину оновлення на основі оцінок градієнтного шуму. Замість перерозподілу кадрів, він «карає» ненадійні напрямки після оцінки градієнта, вбудовуючи усвідомлення шуму всередину самого оптимізатора.

2.4.1.2 Квантовий природний градієнт (QNG)

Метод квантового природного градієнта покращує збіжність шляхом перемасштабування градієнтів відповідно до геометрії квантового простору параметрів, використовуючи метрику, отриману з метрики Фубіні-Студі або інформаційної матриці Фішера [3]. Цей підхід ґрунтується на диференціальній геометрії та призводить до оновлень, які є інваріантними при репараметризації квантової схеми.

Хоча QNG виправляє проблеми, пов'язані з кривизною та чутливістю параметрів, він безпосередньо не вирішує проблему шуму вимірювання. Він припускає, що вхідні градієнти є достатньо точними, а його складність погано масштабується з кількістю параметрів через необхідність оцінки та інвертування матриці Фішера, часто за допомогою дорогих квантових вимірювань.

На противагу цьому, QNA-Adam не робить геометричних припущень і вводить чутливість до шуму через просте, прозоре зниження швидкості навчання, що робить його набагато легшим та сумісніше з обмеженими ресурсами NISQ.

2.4.1.3 Методи, натхненні фільтром Калмана: KOALA та KAdam

Два нещодавні алгоритми (KOALA [19] та KAdam (Kalman Adam) [18]) — виходять за рамки стратегій адаптації до шотів, вбудовуючи байєсівську модель оновлення безпосередньо в оптимізатор.

Обидва натхненні фільтром Калмана, який оцінює оптимальне злиття невизначених джерел інформації.

KAdam використовує рівняння в стилі Калмана для поєднання поточних шумних градієнтів з плинною оцінкою "істинного" напрямку градієнта, розглядаючи стан оптимізатора як ймовірнісне переконання, що прогресує. Він відстежує невизначеності та відповідно коригує імпульс і швидкість навчання.

KOALA базується на подібних принципах, але робить акцент на адаптивній модуляції швидкості навчання. Він підтримує оцінки дисперсії як сигналу, так і шуму для кожного параметра та оновлює швидкості навчання, використовуючи коефіцієнти посилення, подібні до коефіцієнтів Калмана, отримані з їх співвідношення.

KOALA та KAdam надійно працюють за високого рівня шуму, але вони також є обчислювально складнішими, вимагаючи рекурсивного оновлення дисперсій або коваріацій, а в деяких випадках налаштування більшої кількості гіперпараметрів. Їхні реалізації є складнішими та можуть бути менш придатними для навчання в реальному часі за жорстких апаратних або кадрових обмежень.

2.4.1.4 Позиціонування QNA-Adam

QNA-Adam можна розглядати як міст між класичними оптимізаторами та повною байєсівською/калманівською фільтрацією. Його характеристики:

- використовує емпіричну градієнтну дисперсію безпосередньо зі статистики вимірювань;
- застосовує просте раціональне зниження швидкості навчання без рекурсивного байєсівського злиття;
- не потребує додаткової пам'яті для станів довіри або розкладання сигнал-шум;
- залишається незалежним від фреймворку.

Таким чином, QNA-Adam пропонує легке, інтерпретоване та практичне рішення для квантової оптимізації в еру NISQ, особливо коли бюджети на розробку та доступ до обладнання обмежені (таблиця 2.1).

Таблиця 2.1 – таблиця порівняння оптимізаторів

Метод	**Шумо-адаптивність**	**Адаптивність за вимірюваннями**	**Адаптивний до оптимізатора?**	**Склад-ність**	**Notes**
SGD / Adam	-	-	-	Низька	Класичні базові методи
ShotAdaptiveOptimizer	+	+	-	Низька	Враховує лише вимірювання
Rosalin / iCANS	+	+	Partially	Середня	Деяка логіка стійкості до шуму
QNG	-	-	+	Висока	Не враховує шум
KOALA	+	-	+	Висока	Потребує моделювання сигнал–шум
KAdam	+	-	+	Висока	Злиття в стилі фільтра Калмана
QNA-Adam	+	-	+	Низька	Масштабування кроку на основі дисперсії

2.5 Інтерпретація байєсівського та калманівського фільтрів в контексті QNA-Adam

Формулу швидкості навчання QNA-Adam можна розглядати не просто як евристичний штраф для шумних градієнтів, а як рішення, засноване на баєсівській теорії оцінювання. Вона відображає правило оновлення фільтра Калмана — оптимальної рекурсивної оцінки в умовах квантового шуму.

У рамках фільтра Калмана оцінка прихованої змінної (наприклад, положення, імпульсу або значення параметра) ітеративно оновлюється на основі нових спостережень, зважених відповідно до їхньої відносної достовірності. Рівняння оновлення Калмана записується згідно з формулою 2.13.

$$x_t = x_{t-1} + K_t(z_t - x_{t-1}) \tag{2.13}$$

де x_{t-1} — попередня оцінка, z_t— шумне спостереження, а K_t — коефіцієнт підсилення Калмана (Kalman gain), значення від 0 до 1, що визначає, наскільки ми довіряємо новому спостереженню.

Коефіцієнт Калмана обчислюється згідно з формулою 2.14.

$$K_t = \frac{\Sigma_{\{t-1\}}}{\Sigma_{\{t-1\}} + R_t} \tag{2.14}$$

де $\Sigma_{\{t-1\}}$ — дисперсія апріорної оцінки (невизначеність поточної оцінки), а R_t— дисперсія шуму вимірювань. Коли вимірювальний шум високий, $K_t \to 0$, і ми довіряємо попередній оцінці. Коли шум низький, $K_t \to 1$, і ми повністю приймаємо нове спостереження. Ця структура напряму аналогічна до правила швидкості навчання з урахуванням шуму в QNA-Adam (рівняння 2.12).

З баєсівської точки зору, ми інтерпретуємо оновлення градієнта як зашумлене спостереження справжнього оптимального напрямку спуску. Якщо дисперсія цього спостереження висока, нам слід оновлювати консервативно, тоді як якщо спостереження чітке, ми можемо діяти більш впевнено.

Таким чином, QNA-Adam неявно трактує оновлення параметрів як рекурсивну баєсівську оцінку, де дисперсія градієнта служить показником невизначеності спостереження. На відміну від повних оптимізаторів на основі Калмана, таких як KOALA та KAdam, які можуть підтримувати повний

розподіл станів довіри та рекурсивно оновлювати апостеріорні дисперсії, QNA-Adam використовує легкий, прямий штраф, який апроксимує баєсівську довіру без необхідності матричних операцій або рекурсії.

Це забезпечує як теоретичну основу для оптимізатора, так і практичне обґрунтування його структури: QNA-Adam апроксимує адаптивне навчання в стилі Калмана без складності, що робить його особливо придатним для шумної та обмеженої ресурсами квантової оптимізації.

2.6 Властивості та аналіз правила оновлення QNA-Adam

Основою суттю оптимізатора QNA-Adam є його адаптивна до дисперсії швидкість навчання, яка вводить пряму реакцію на невизначеність вимірювання в оцінці квантового градієнта. У цьому розділі аналізуються ключові математичні та практичні властивості правила оновлення, зосереджуючись на його обмеженості, граничній поведінці та стійкості до шумових флуктуацій.

2.6.1 Поведінка у граничних режимах

Може розподілятись на режим низького шуму ($\sigma^2_{\{i,t\}} \to 0$) та режим високого шуму. За низького QNA-Adam відновлює поведінку класичного Adam (формула 2.15).

$$\alpha_i^{(t)} \to a_0 \tag{2.15}$$

Тобто повна швидкість навчання застосовується до надійних напрямків градієнта. За режиму високого шуму ($\sigma^2_{\{i,t\}} \gg 1$) швидкість навчання зменшується обернено пропорційно до рівня шуму (формула 2.16).

$$\alpha_i^{(t)} \approx \frac{\alpha_0}{\lambda \sigma^2_{\{i,t\}}} \tag{2.16}$$

Таким чином, оновлення ефективно пригнічуються у дуже невизначених напрямках. Ця поведінка дозволяє QNA-Adam гнучко адаптуватися на різних етапах навчання. Спочатку, коли градієнти зазвичай малі та шумні, штрафи за дисперсію запобігають нестабільності. За прогресу оптимізації й покращення оцінок градієнта оптимізатор природно застосовує агресивніші оновлення.

2.6.2 Стійкість до гострого або тимчасового градієнтного шуму

На практиці градієнтний шум у квантових схемах не завжди стабільний. Він може змінюватися від ітерації до ітерації через зміну параметрів схеми, обмеження розподілу кадрів або флуктуації на зворотному кінці. Оптимізатори, які надмірно реагують на такі флуктуації (наприклад, занадто агресивно регулюючи розміри кроків), можуть страждати від коливань, переналаштування під шум або навіть дивергенції.

QNA-Adam вирішує цю проблему, безпосередньо включаючи виміряну дисперсію, а не покладаючись на окремі значення градієнта. Так транзиентно велике $\tilde{g}_i^{(t)}$ має незначний вплив, якщо його дисперсія висока, а стабільні напрямки градієнтів посилюються не лише завдяки величині, але й через низький рівень шуму.

Цей контроль на основі дисперсії діє як механізм статистичної довіри, що дозволяє QNA-Adam уникати надмірної реакції на помилкові градієнтні піки.

2.6.3 Стійкість до гострого або тимчасового градієнтного шуму

Ключовий гіперпараметр λ модулює вплив шуму: малий λ змушує оптимізатор поводиться більше як класичний Адам, а великий λ агресивно

«карає» за шумні градієнти, і таким чином, він уповільнює нестабільні оновлення.

Однією з критичних переваг формули раціональної швидкості навчання є те, що вона запобігає неконтрольованій поведінці за наявності зникаючих градієнтів та шумних вимірювань. Так можна привести приклад, якщо компонент градієнта $\tilde{g}_i^{(t)} \approx 0$, але його оцінена дисперсія висока, класичний Adam все ще може посилити його внесок завдяки малому знаменнику в оновленні другого моменту. QNA-Адам повністю уникає цього, зменшуючи саму швидкість навчання, діючи консервативно в напрямках, де інформація є низькодостовірною, навіть коли градієнт здається оманливо малим.

3 АРХІТЕКТУРА ДЛЯ ЕКСПЕРИМЕНТУ

Розробка ефективної гібридної квантово-класичної нейронної мережі (QNN) є важливою для оцінки та порівняльного аналізу продуктивності запропонованого оптимізатора Quantum Noise-Aware Adam (QNA-Adam). У цьому розділі описано наскрізний конвеєр гібридної архітектури QNN, розробленої для цієї мети, включаючи попередню обробку даних, квантове кодування, варіаційну структуру схеми, стратегії вимірювання, класичну постобробку та інтеграцію оптимізатора з урахуванням шуму в цикл навчання.

Кожен модуль у конвеєрі розроблено з урахуванням обмежень епохи NISQ, включаючи малу глибину схеми, обмежену кількість кубітів та схильність до шуму вимірювання. Архітектура розроблена як модульна та розширювана, що дозволяє їй підтримувати експерименти з різними методами кодування, глибиною схеми та варіантами оптимізатора.

У цьому розділі ми висвітлюємо, як QNA-Adam включена в цикл оптимізації, пояснюємо її взаємодію з градієнтним шумом, оціненим за квантовими вимірюваннями. Цей модульний конвеєр служить не лише емпіричним тестовим платформою для QNA-Adam, але й універсальним шаблоном для навчання варіаційних квантових класифікаторів на класичних наборах даних, таких як MNIST.

3.1 Огляд архітктури

Процес навчання гібридної квантово-класичної нейронної мережі (QNN) поєднує класичну обробку даних з квантовим варіаційним виконанням схеми та класичною оптимізацією. Архутектура, що використовується в цій роботі, призначений для класифікації рукописних цифр з набору даних MNIST, одночасно забезпечуючи практичний контекст для оцінки ефективності оптимізатора QNA-Adam в реалістичних умовах шуму.

Загальна архітектура відповідає модульній конструкції, що дозволяє незалежно аналізувати або замінювати кожен етап. Високорівнева схема потоку даних показана на рисунку 3.1.

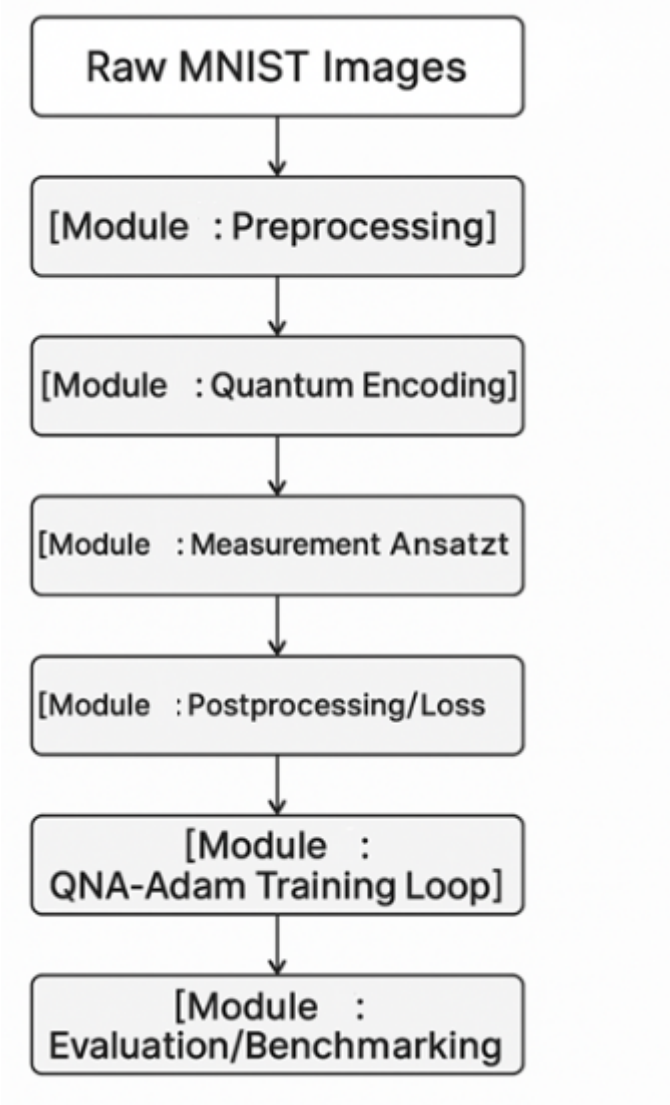

Рисунок 3.1 – Високорівнева схема потоку даних

Етапи конвеєра:

– необроблені вхідні дані (зображення MNIST): вхідні дані складаються з рукописних цифр у градаціях сірого розміром 28×28 пікселів. Для зменшення розмірності для кодування всі зображення перетворюються на низьковимірний вектор за допомогою класичної передобробки;
– класична попередня обробка: крок зменшення розмірності (наприклад,

PCA) витягує найважливіші ознаки та зменшує кількість вимірів, щоб вони відповідали доступній кількості кубітів (зазвичай 4–8). Вихід: вектор з дійсними значеннями, придатний для кодування в квантові стани;

- квантове кодування: класичний вектор відображається в квантовий стан;
- VQA: квантові дані обробляються за допомогою параметризованої квантової схеми $U(\theta)$, яка включає навчальні вентилі та шари переплутаності. Вихідним результатом є кінцевий квантовий стан, що представляє собою вивчене перетворення вхідних даних;
- зчитування: кубіти вимірюються у фіксованому базисі для отримання ймовірностей результату. Ці результати вимірювань використовуються для обчислення очікуваних значень спостережуваних величин або розподілів ймовірностей;
- постобробка та обчислення втрат: результати вимірювань перетворюються на прогнози;
- цикл оптимізації за допомогою QNA-Adam: Швидкість навчання коригується для кожного параметра (рівняння 14). Ця швидкість навчання використовується в оновленні параметрів, подібному до Адама. Забезпечує стабільнішу оптимізацію в умовах квантового шуму.

QNA-Adam замінює класичний оптимізатор (наприклад, Adam або SGD) у фінальному циклі навчання, вводячи механізм зворотного зв'язку, який динамічно коригує швидкість навчання на основі дисперсії градієнта для кожного параметра. Це дозволяє системі інтелектуально реагувати на шум вимірювання, викликаний обладнанням, без зміни інших компонентів конвеєра.

Кожен етап конвеєра розроблений для підтримки кількох конфігурацій, що робить робить конвеєр придатним не лише для оцінки продуктивності, але й для дослідницьких експериментів, де метою є розуміння взаємодії обмежень квантового обладнання та оптимізації з урахуванням шуму.

У цьому розділі формалізовано методологію та інструментарій, використані під час передатестаційної практики для побудови й оцінювання гібридної квантово-класичної моделі (QNN) з оптимізатором QNA-Adam. Мета розділу — зробити відтворюваними всі ключові рішення експерименту: від підготовки даних і архітектури варіаційного ланцюга до способу обчислення градієнтів, політики шотів, налаштування оптимізаторів, валідації та логування артефактів.

Подача організована «від даних до артефактів». Спершу описуємо набір даних і препроцесинг (PCA → стандартизація), далі — кодування ознак у квантову схему R_y, апаратно-ефективний анзац та класичну голову класифікатора. Окремий підпункт присвячено обчисленню градієнтів для параметрів квантової частини через Inline-DocPS з розділенням градієнтних шляхів і збором пер-параметрної дисперсії $\widehat{\mathrm{Var}}[g]$, що надалі використовується QNA-Adam для noise-aware масштабування кроків. Після цього фіксуємо вибір оптимізаторів і параметр-груп, політику шотів (у т.ч. лінійний аннілінг та eval_shots), механізми валідації та ранньої зупинки, а також структуру логів і артефактів експерименту (metrics.csv, чекпойнти, профайлінг, q-метрики).

Наприкінці розділу узагальнюємо вимоги до відтворюваності (сідування, детермінізм на рівні Torch/CUDA, версії ПЗ/заліза) і наводимо стислий псевдокод одного тренувального кроку, який однозначно відображає реалізацію в тренері. Така структура дозволяє прозоро співвіднести кожну експериментальну метрику з конкретним налаштуванням і частиною коду, а в подальшому (розд. 4–6) — коректно інтерпретувати отримані результати.

3.2 Дані та препроцесинг

Для навчання та оцінювання гібридної квантово-класичної моделі використовувався стандартний набір MNIST — зображення рукописних цифр розміром 28×28пікселів у відтінках сірого. Для цілей практики було обрано

підмножину набору MNIST із чотирьох класів, що дало змогу скоротити розмірність ознак і кількість кубітів без втрати загальності постановки задачі класифікації.

Дані були поділені на три підмножини:

- тренувальну (80 %);
- валідаційну (10 %);
- тестову (10 %).

Перед подачею до квантової частини дані проходили двоетапний препроцесинг:

- зменшення розмірності методом головних компонент (PCA);
- стандартизація (Z-score normalization).

Реалізаційно обидва етапи об’єднані в модуль PhiDataset, який при ініціалізації отримує конфігураційний словник (YAML-файл) і автоматично підвантажує попередньо збережені компоненти PCA та статистику стандартизації. Кожен запис у датасеті на виході має пару $(\mathbf{u}', y)$, де $\mathbf{u}'$— вектор стандартизованих ознак розмірності d, а y— цільова мітка класу.

Таким чином, на вхід квантової частини подається компактний, масштабно нормований вектор $\mathrm{u}' \in \mathbb{R}^d$, що дозволяє безпосередньо кодувати кожну компоненту в кут обертання одного кубіта при формуванні варіаційного ланцюга.

3.3 Архітектура моделі (QuantumClassifier)

Архітектура моделі побудована за принципом гібридної квантово-класичної нейронної мережі (QNN), де квантовий шар виконує роль нелінійного перетворювача ознак, а класична частина відповідає за остаточну класифікацію. Реалізація здійснена у вигляді класу QuantumClassifier, який містить два основні компоненти:

- quantum — квантову підмодель, побудовану на бібліотеці PennyLane;
- classifier — класичну «голову» на основі PyTorch.

3.3.1 Квантова частина

Квантовий модуль реалізує параметризовану схему (варіаційний анзац) з Lапаратно-ефективних шарів (HEA, Hardware-Efficient Ansatz). Кожен шар включає:

- локальні обертання навколо осі y, що кодують вхідні ознаки;
- ланцюжкове переплутування за допомогою вентилів CNOT.

Таким чином, схема зберігає топологічну сумісність навіть для NISQ-пристроїв і симуляторів, мінімізуючи глибину при збереженні експресивності.

Вхідний вектор $\mathbf{u}' \in \mathbb{R}^d$, отриманий після PCA та стандартизації, кодується в кути обертання вентилів R_yзгідно з правилом 3.1.

$$\phi_i = \alpha\, u_i', R_y(\phi_i) = e^{-i\phi_i \sigma_y/2}, \tag{3.1}$$

де α— масштабний коефіцієнт для контролю діапазону обертів. На кожному кубіті формується набір вентилів $R_y(\phi_i)$, після чого застосовується Lциклів (R_y, CNOT). Вимірювання здійснюється в базисі Z, і результатом є вектор очікуваних значень, який служить квантовими ознаками для класичного класифікатора (формула 3.2).

$$\mathbf{E} = [\, \langle Z_1 \rangle, \langle Z_2 \rangle, \ \ldots, \langle Z_r \rangle \,], \tag{3.2}$$

3.3.2 Класична частина (голова класифікатора)

Вектор квантових очікувань E подається на класичний лінійний шар, який виконує афінне перетворення, де W і b — ваги та зміщення голови (формула 3.3).

$$o = WE + b, \tag{3.3}$$

Після цього застосовується функція softmax для отримання ймовірностей класів (формула 3.4).

$$p_i = \frac{e^{o_i}}{\sum_j e^{o_j}} \tag{3.4}$$

Навчання здійснюється за функцією втрат крос-ентропії, спільною для класичних моделей (формула 3.5), де B— розмір батча, а $p_{y_n}^{(n)}$— передбачена ймовірність правильного класу.

$$\mathcal{L}(\theta) = -\frac{1}{B}\sum_{n=1}^{B} \log\, p_{y_n}^{(n)}, \tag{3.5}$$

3.3.3 Узгодження з конфігураційним файлом

Усі параметри архітектури моделі, квантового пристрою та оптимізації задаються централізовано через YAML-конфігурацію, що забезпечує відтворюваність експериментів і узгодженість між усіма компонентами.

Квантовий модуль реалізовано на симуляторі default.qubit, який відтворює поведінку шумного пристрою з обмеженою кількістю вимірювань (100 шотів під час навчання). Для валідації використовується розширений режим із 2048 шотами, що підвищує точність оцінювання очікуваних значень $\langle Z_i \rangle$. Анзац має два апаратно-ефективні шари з кільцевою топологією переплутування, яка зберігає локальність взаємодій між кубітами та мінімізує глибину схеми. Ознаки кодуються через обертання навколо осі Y з повторним завантаженням у кожному шарі (reuploading), що збільшує експресивність моделі. Вимірювання виконуються змішаним оператором ZX, а градієнти обчислюються методом parameter-shift при фіксованому сідуванні.

Оптимізація здійснюється за допомогою QNA-Adam — модифікації алгоритму Adam, яка враховує дисперсію градієнтів для кожного параметра (mode = param). Базова швидкість навчання становить 0.0003, параметри $\beta_1 = 0.9$, $\beta_2 = 0.999$, без регуляризації ваг. Межі масштабування швидкості навчання визначаються коефіцієнтами lr_min = 0.1 та lr_max = 1.0, а чутливість до шуму — параметром λ_var = 0. Для стабільності градієнтів задано межу норми та числову стабілізацію.

Кількість шотів у процесі тренування залишається сталою (mode = none), що гарантує порівнянність результатів між епохами. Такий підхід забезпечує повний контроль над гіперпараметрами, дозволяючи гнучко адаптувати експерименти до різних шумових моделей і конфігурацій квантових пристроїв.

3.4 Обчислення градієнтів та метод Inline-DocPS

Навчання варіаційних квантових моделей передбачає обчислення похідних функції втрат $\mathcal{L}(\theta)$за параметрами квантової схеми θ. На відміну від класичних нейронних мереж, градієнти тут не можна отримати прямою диференціацією — кожна оцінка ґрунтується на результатах вимірювань із квантового пристрою, що вводить стохастичний шум (shot noise).

Для уникнення додаткового обчислювального навантаження та контролю шуму в цьому проєкті реалізовано власну версію parameter-shift rule, інтегровану безпосередньо у тренер. Цей підхід дозволяє одночасно:

- обчислити градієнти $\partial\mathcal{L}/\partial\theta$для квантового шару;
- оцінити емпіричну дисперсію градієнтів $\widehat{\mathrm{Var}}[g_i]$для кожного параметра θ_i;
- зберегти значення цієї дисперсії для подальшого використання оптимізатором QNA-Adam.

3.4.1 Основна ідея parameter-shift

Для будь-якого параметра θ_k очікуване значення квантового вимірювання можна подати як періодичну функцію 3.6.

$$E_k(\theta_k) = \langle 0 \mid U^{\dagger}(\theta_k)\, Z\, U(\theta_k) \mid 0 \rangle. \tag{3.6}$$

Її аналітична похідна обчислюється через parameter-shift-формулу 3.7.

$$\frac{\partial E_k}{\partial \theta_k} = \frac{1}{2}\left[E_k\left(\theta_k + \frac{\pi}{2}\right) - E_k\left(\theta_k - \frac{\pi}{2}\right)\right]. \tag{3.7}$$

Таким чином, для оцінки градієнта достатньо двох запусків схеми з параметром, зсунутим на $\pm\pi/2$.

3.4.2 Реалізація Inline-DocPS у тренері

У тренерному циклі обчислення градієнтів квантового шару й оцінка shot-noise виконуються «вбудовано», без окремого глобального backward крізь квантову гілку. Для кожного батча спершу отримуються очікувані значення вимірювань E = model.quantum(xb) і логіти класифікатора logits = model.classifier(E), після чого рахується втрата. Далі вибираємо вектор-джекобіан-добуток $\partial L / \partial E$ через torch.autograd.grad(loss, E) — це дає чутливість функції втрат до квантових виходів без побудови повного графа через параметризовану схему. Цей градієнт передається у метод квантового шару backward_inline_docps(xb, dL_dE, reduce_var="param"), який параметр-shift-обчисленням заповнює ∇_θ «на місці» (тобто без другого проходу PyTorch) і водночас оцінює дисперсію вимірювань: повертає скалярний індикатор шуму $\tilde{V}$ та тензор var_param розміру, сумісного з параметрами θ. Отриманий var_param одразу «вшивається» в стан оптимізатора через

_push_qna_var_to_optimizer(...): для параметра θу optimizer.state[θ] кладеться qna_var_param (зі санітаризацією NaN/Inf, приведенням dtype/девайса й керованим fallback'ом форми). Якщо використовується QNA-Adam, він під час step() читає це поле та масштабує елемент-по-елементу базове оновлення $\text{update} = \widehat{m}/\sqrt{\widehat{v} + \varepsilon}$за правилом $\text{scale}_\lambda = 1/(1 + \lambda_{\text{var}} \cdot \tilde{V})$у межах $[\text{lr\_min\_mult}, \text{lr\_max\_mult}]$, після чого очищає qna_var_param. Паралельно у CSV фіксуються $\tilde{V}$та агрегована телеметрія QNA (scale_mean, scale_p90, частки на межах, upd_norm, cos_gu).

Щоб уникнути подвійного обчислення градієнтів у квантовій гілці, подальша оптимізація «голови» виконується на від'єднаних квантових виходах: обчислюються logits2 = model.classifier(E.detach()), втрата loss2 і викликається loss2.backward(), що дає градієнти лише для класичної частини. За потреби застосовується зовнішній кліпінг норм градієнтів (якщо внутрішній кліпінг у QNA-Adam вимкнено), після чого виконується optimizer.step(). Така організація дозволяє поєднати ефективний parameter-shift для ∇_θ, одноразову й узгоджену з кроком оцінку shot-noise для шум-адаптивного масштабування, а також чисте розділення оновлень квантових параметрів і класичного класифікатора без зайвого графа PyTorch і дублювання обчислень.

3.4.3 Розрахунок дисперсії градієнтів

Дисперсія shot-noise для кожного параметра θ_iоцінюється статистично з багатьох вимірювань у межах одного батча (формула 3.8)

$$\widehat{\text{Var}}[g_i] = \frac{1}{S-1} \sum_{s=1}^{S} (g_i^{(s)} - \bar{g}_i)^2, \tag{3.8}$$

де S— кількість шотів, $g_i^{(s)}$— окремі оцінки градієнта, $\bar{g}_i$— їх середнє.

3.5 Оптимізатори, групи параметрів і шум-адаптивне масштабування (QNA-Adam)

Для оптимізації параметрів гібридної моделі застосовувався алгоритм QNA-Adam (Quantum Noise-Aware Adam) — модифікація класичного *Adam*, розроблена для сценаріїв із квантовим шумом та стохастичними оцінками градієнтів. У порівнянні з базовим Adam, який оновлює параметри за правилом (1), QNA-Adam вводить додатковий тензор масштабів шуму S_λ, який регулює амплітуду оновлення кожного параметра (формула 3.9).

$$S_\lambda = \text{clip}\left(\frac{1}{1+\lambda_{\text{var}}\tilde{V}}, s_{\min}, s_{\max}\right), \theta_{t+1} = \theta_t - \eta\, S_\lambda * \frac{m_t/(1-\beta_1^t)}{\sqrt{v_t/(1-\beta_2^t)}+\varepsilon}. \quad (3.9)$$

Тут $\tilde{V}$ — оцінена дисперсія shot-noise для даного параметра, λ_{var} — коефіцієнт чутливості до шуму, $s_{\min} = lr_{\text{min\_mult}}$ і $s_{\max} = lr_{\text{max\_mult}}$ — межі допустимого масштабування.

3.5.1 Джерела оцінки шуму

QNA-Adam підтримує два режими отримання $\tilde{V}$: per-parameter та group-level.

Per-parameter (*рекомендований у проєкті*) — тренер (train_quantum.py) перед кожним кроком оптимізації він виконує виклик функції _push_qna_var_to_optimizer з параметрами «optimizer», «model.quantum» та «var_param», де var_param — тензор дисперсій $\text{Var}[g_i]$. Цей тензор в подальшому використовується в подальшому розрахунку S_λ.

Group-level (fallback) — якщо для параметра відсутній індивідуальний qna_var_param, використовується скалярне значення групи group[vtilde key].

У даному експерименті використано режим mode: param, тобто масштабування виконується для кожного параметра окремо.

3.5.2 Телеметрія та діагностика

Під час кожного кроку QNA-Adam формує статистику масштабування і зберігає її в полі group["qna_stats"]. Вони логуються у файл metrics.csv після кожної епохи та показують стабільність навчання. Значення (табл. 3.1):

Таблиця 3.1 – Зміст qna_stats

Поле	Зміст
scale_mean, scale_p90, scale_min, scale_max	статистики масштабу S_λ
scale_at_min_frac, scale_at_max_frac	частка параметрів, які досягли меж $s_{\min}$ або $s_{\max}$
update_norm, grad_norm	норми оновлень і градієнтів
cos_grad_update	косинусна близькість між градієнтом і оновленням (міра «ефективності» кроку)

3.5.3 Порівняння з класичним Adam

Порівняння надано з класичним Adam надано в таблиці 3.2

Таблиця 3.2 – Порівняння з класичним Adam

Властивість	Adam	QNA-Adam
Оцінка шуму градієнта	немає	використовується $\tilde{V}$(shot-noise)
Масштабування кроку	однакове для всіх параметрів	індивідуальне, шум-залежне
Телеметрія кроків	відсутня	групова статистика масштабу, норм, косинусів
Сумісність з PyTorch API	повна	повна (drop-in replacement)

Таким чином, QNA-Adam реалізує підхід noise-aware optimization, що забезпечує стабільніше оновлення параметрів у стохастичному квантовому середовищі. Його реалізація повністю сумісна з інфраструктурою PyTorch та дозволяє без змін коду переходити між Adam і QNA-Adam у конфігураційному файлі.

3.6 Політика шотів, стохастика та шедулінг

У варіаційних квантових моделях shot noise є головним джерелом стохастичності. Кожне очікуване значення $\langle Z_i \rangle$ отримується шляхом усереднення результатів вимірювань за фіксованою кількістю шотів S. У цьому експерименті основна кількість шотів під час навчання становила 100, що відповідає умовам обмеженого ресурсу вимірювань, а під час валідації — 2048, щоб зменшити статистичну похибку при оцінці точності.

Для забезпечення контрольованої повторюваності в симуляторі default.qubit використовується фіксоване зерно випадкових чисел (seed) на рівні проєкту (project.seed = 42). Параметр reseed_each_epoch: false фіксує однакову послідовність shot-noise для всіх епох, дозволяючи відокремити вплив оптимізатора від впливу статистичного шуму.

Таким чином, кожна епоха використовує однакові стохастичні умови, що робить результати безпосередньо порівнюваними між QNA-Adam і базовим Adam.

3.7 Валідація, рання зупинка та контроль стабільності

Для контролю узагальнювальної здатності моделі після кожної епохи проводиться валідаційний прохід із тим самим набором параметрів θ, але з більшою кількістю шотів (eval_shots = 2048). Розраховуються середні значення втрати ($\mathcal{L}_{\text{val}}$) і точності (Acc_{val}), після чого результати логуються у файл metrics.csv

Для запобігання перенавчанню застосовується клас EarlyStopper, який відстежує найкраще значення моніторингової метрики та зупиняє навчання після серії епох без покращення.

Алгоритм працює так:

- після кожної епохи порівнюється поточне val_lossз найкращим;
- якщо протягом patience епох не зафіксовано покращення більше ніж на min_delta, процес припиняється;
- якщо restore_best=true, модель автоматично повертається до стану з найменшою валідаційною втратою.

3.8 Логування, артефакти та відтворюваність

Система логування спроєктована для повної прозорості експериментів: усі проміжні й підсумкові результати автоматично зберігаються тренером у структурованій директорії. Для кожного запуску створюється окрема папка на кшталт runs/EXP_20251106_optqnaadam_enc_ry_nq8_L2_shots100/, у якій містяться ключові артефакти. Файл metrics.csv акумулює покрокові та епохальні метрики (зокрема loss, accuracy, норми градієнтів і телеметрію QNA-Adam), model.pt містить найкращу або фінальну версію моделі, а config.snapshot.yaml зберігає повний знімок конфігурації для відтворення. Додатково, у profiler.jsonl фіксуються часи виконання кожної епохи в мілісекундах разом із кількістю шотів, тоді як qmetrics.jsonl містить спеціалізовані квантові показники (φ-розподіли, очікування та градієнти). Каталог logs/ зберігає текстові журнали виконання з інформаційними повідомленнями й попередженнями, що полегшує діагностику та аудит.

Контроль відтворюваності забезпечується комплексом заходів. Кожен експеримент фіксує сіди для генераторів випадкових чисел мовою Python (random), бібліотеки NumPy та фреймворку Torch. Увімкнено параметр deterministic_torch=true, який примушує детермінізм у бібліотеці cuDNN і відключає режим TF32, зменшуючи недетермінізм на GPU. Окремо

протоколюється середовище виконання: версії Python, PyTorch, PennyLane, NumPy та scikit-learn, а також відомості про ОС і апаратну конфігурацію CPU/GPU. Для повноти відтворення разом з іншими артефактами завжди зберігається повна копія YAML-конфігурації у вигляді знімка (snapshot). Після завершення навчання автоматично записується чекпойнт найкращої моделі — за мінімальним значенням val_loss або, альтернативно, за максимумом val_acc; якщо найкращий чекпойнт не збігся з фінальною епохою, додатково зберігається і фінальна версія моделі. Така організація артефактів і контроль середовища гарантують можливість точного відтворення експериментів і коректного порівняння результатів.

3.9 Відтворюваність та середовище виконання

Відтворюваність експериментів є критичною для квантово-класичних систем, оскільки навіть незначна стохастика підрахунків шотів або генераторів випадкових чисел здатна помітно вплинути на оцінку стабільності оптимізації. Для мінімізації цих чинників у проєкті реалізовано багаторівневу політику фіксації випадковості та ізоляції середовища. Базове зерно задається безпосередньо в конфігурації (ключі project.seed = 42 та project.deterministic_torch = true), після чого спеціальна процедура _set_all_seeds() синхронно сідує бібліотеки random, NumPy і Torch та забезпечує повторювану ініціалізацію потоків завантажувача даних через worker_init_fn(seed + worker_id). Для зниження недетермінізму на GPU в PyTorch активуються детерміністичні режими: torch.backends.cudnn.deterministic = True, torch.backends.cudnn.benchmark = False, а також вимикається обчислення у форматі TF32 як для тензорних операцій CUDA (torch.backends.cuda.matmul.allow_tf32 = False), так і для cuDNN (torch.backends.cudnn.allow_tf32 = False). У сукупності ці налаштування відключають недетерміновані оптимізації та забезпечують ідентичні результати за повторних запусків.

Експерименти виконувалися на симуляторі default.qubit з фіксованою кількістю шотів і апаратно-незалежним бекендом PennyLane, що додатково спрощує контроль джерел випадковості. Базове середовище має такі параметри: Python 3.11, PennyLane 0.37, PyTorch 2.2, NumPy 1.26, scikit-learn 1.5, операційна система Windows 10 x64, процесор Intel Core i7-12700H, оперативна пам’ять 16 GB, а за наявності — GPU NVIDIA RTX 3060 Laptop (6 GB). Версії бібліотек фіксуються у файлі requirements.txt, який автоматично експортується з робочого середовища. Для повної відтворюваності кожен запуск зберігає знімок конфігурації (config.snapshot.yaml), що дозволяє відтворити експеримент на іншому обладнанні; разом із цим автоматично зберігаються чекпойнти найкращої моделі (визначеної за мінімальним val_loss або максимумом val_acc) і фінальна версія моделі, якщо ця подія не відбулася раніше. Таке поєднання фіксації зерен, детермінізму обчислень і протоколювання середовища забезпечує стабільність, прозорість і точну повторюваність отриманих результатів.

4 РЕАЛІЗАЦІЯ

4.1 Структура проєкту

Програмна частина дослідження реалізована у вигляді модульного Python-проєкту з чітким поділом відповідальностей між компонентами. Основні вихідні коди розташовано в директорії Code/, структура якої побудована за принципом чистої архітектури — кожен модуль виконує одну логічну функцію (дані, моделі, оптимізатори, тренування, утиліти тощо).

4.1.1 Рівень даних (datasets/, preprocess/)

Рівень даних відповідає за підготовку вхідних даних: від зчитування набору MNIST до зменшення розмірності методом головних компонент (PCA) і стандартизації ознак. Кінцевим виходом є тензори u', готові до кодування у квантові кути.

4.1.2 Рівень моделі (models/)

Рівень моделі містить гібридну архітектуру QuantumClassifier, яка поєднує квантовий модуль (QuantumLayer) і класичну голову класифікатора. Взаємодія між квантовою частиною та оптимізатором відбувається через внутрішній інтерфейс backward_inline_docps.

4.1.3 Рівень оптимізатора (optim/)

Рівень оптимізатора включає реалізацію класичного адаптивного оптимізатора Adam і його модифікації QNA-Adam — алгоритму, що адаптує крок оновлення з урахуванням рівня shot-noise. Обидва реалізовані в рамках

одного модулю, поведінка якого відрізняється виключно через передаваємі параметри (lambda_var). Модуль registry.py дозволяє динамічно підключати нові оптимізатори без змін основного коду тренера.

4.1.4 Рівень тренування (trainers/)

Рівень тренування відповідає за створення оркестратора, що відповідає виконання циклу навчання, логування метрик, контроль ранньої зупинки, перевірку узгодженості параметрів, керування шотами, формування артефактів (metrics.csv, model.pt, qmetrics.jsonl).

4.1.5 Рівень утиліт (utils/,logger.py)

Рівень утиліт забезпечує сервісні функції: роботу з файлами, сідування, збереження конфігурацій, централізоване логування з підтримкою рівнів INFO, WARNING, DEBUG.

5. ЕКСПЕРИМЕНТАЛЬНИЙ ПРОТОКОЛ

5.1 Мета та гіпотези

Мета. Кількісно оцінити вплив рівня shot-noise під час тренування (100, 36 і 15 шотів) на стабільність та якість навчання гібридної QNN і порівняти оптимізатори Adam та QNA-Adam з пер-параметрним урахуванням дисперсії градієнтів за однакової архітектури та протоколу валідації (eval_shots = 2048).

Гіпотеза 1. Зменшення кількості шотів із 100 до 36 і 15 підвищує стохастичність оцінок: зростає середній рівень Vtilde, збільшується варіативність grad_norm, криві train-loss стають більш пилкоподібними. Очікується, що валідаційна точність для 100 шотів не нижча за 36, а для 36 не нижча за 15.

Гіпотеза 2. За малих шотів і додатного значення lambda_var QNA-Adam забезпечує стабільнішу динаміку, ніж Adam: менший розкид grad_norm, плавніший спад loss, не гірша й часто вища val_acc наприкінці навчання, нижчий clip_rate завдяки зменшенню ефективного кроку в шумних координатах. За телеметрією scale_mean наближається до lr_min_mult при 15 шотах, а частка параметрів на межі мінімального масштабу зростає зі зменшенням кількості шотів.

Гіпотеза 3. Використання великих eval_shots під час валідації робить валідаційні криві loss та acc більш стабільними порівняно з тренувальними, тому відбір моделей слід проводити за валідаційними метриками (мінімальний val_loss за епоху).

Гіпотеза 4. Якщо у частині запусків для QNA-Adam встановлено lambda_var = 0, такий режим збігається з Adam за правилом оновлення, але зберігає телеметрію; ці запуски використовуються як контроль, що дозволяє відокремити ефект активного шум-масштабування від простого логування рівня шуму.

Критерії перевірки. Спостерігається монотонне зростання середнього Vtilde зі зменшенням кількості шотів; для QNA-Adam із додатним lambda_var фіксується менший розкид grad_norm, нижчий clip_rate та не гірша val_acc порівняно з Adam у режимах 36 і 15 шотів; показники scale_* демонструють очікуване зміщення у бік нижньої межі масштабу зі зменшенням кількості шотів.

5.2 Дизайн експерименту (фактори та умови)

В експериментальному дизайні передбачено два незалежні фактори:

- кількість шотів під час тренування: 100, 36, 15;
- тип оптимізатора: Adam, QNA-Adam з пер-параметрним урахуванням дисперсії градієнтів.

Це стається за наступних умови: повний перехресний план 3×2, загалом шість окремих запусків. Для кожної умови створюється власна папка артефактів з ідентифікатором експерименту.

У всіх шести умовах фіксуються однакові дані, препроцесинг, архітектура і бекенд: MNIST з підмножиною класів, PCA до кількості кубітів і Z-score, HEA з двома шарами, топологія ring, кодування ry з повторним завантаженням, вимірювання ZX, пристрій default.qubit, метод градієнтів parameter-shift. Гіперпараметри оптимізації: lr 3e-4, betas 0.9 і 0.999, eps 1e-8, weight_decay 0.0, узгоджена політика кліпінгу. Для валідації всюди використовується eval_shots 2048 і відбір найкращої моделі за мінімальним val_loss. Відтворюваність забезпечується фіксованим seed, reseed_each_epoch false та увімкненим детермінізмом у Torch.

Процедура для кожної умови:

- встановлюється кількість тренувальних шотів: 100 або 36 або 15;
- обирається оптимізатор: Adam або QNA-Adam; у QNA-Adam активується пер-параметрний канал передавання var_param з Inline-DocPS;

- виконується навчання на однаковій кількості епох і з тим самим розміром батча;
- після кожної епохи проводиться валідація з eval_shots 2048, зберігається найкраща модель за val_loss;
- логуються батчеві та епохальні метрики у metrics.csv; для QNA-Adam додатково зберігаються qna_stats;

Візуалізації, що будуються для кожного запуску:

- accuracy залежно від епохи з тренувальною та валідаційною кривими. Джерело: рядки train_epoch і val_epoch у metrics.csv, стовпчик acc;
- loss залежно від епохи з тренувальною та валідаційною кривими. Джерело: train_epoch і val_epoch, стовпчик loss;
- grad norm залежно від епохи. Джерело: train_epoch, стовпчик grad_norm; за потреби додатково grad_norm_raw;
- vtilde залежно від епохи. Джерело: train_epoch, стовпчик Vtilde, що є середнім за батчі в епосі.

Правила побудови графіківнаступні:

- для епохальних кривих використовуються лише рядки зі step = E та split ∈ {train_epoch, val_epoch};
- легенда містить назву умови у форматі Shots=S, Optimizer=Adam|QNA-Adam;
- масштаб осей уніфікується між умовами для коректного порівняння;
- за наявності суттєвого шуму на батчевих кривих допускається додаткове згладжування рухомим середнім невеликого вікна, але епохальні криві на основних рисунках наводяться без згладжування.

Повтори та рандомізація. Базовий план передбачає по одному запуску на умову.

Артефакти. Для кожної умови зберігаються model.pt, config.snapshot.yaml, metrics.csv, profiler.jsonl і, за потреби, qmetrics.jsonl; побудовані графіки додаються до звіту з посиланням на відповідні експерименти та конфігурацію.

5.3 Фіксована конфігурація

Усі експерименти виконуються за однакових налаштувань даних і препроцесингу (див. 3.2), архітектури HEA та кодування (3.1–3.2), квантового бекенду та диференціювання (див. 3.3), оптимізації й правил оновлення (3.4), політики шотів/валідації/логування (див. 3.5–3.7), а також відтворюваності й середовища (див. 3.9).

Змінювані фактори лише два: кількість тренувальних шотів $S \in \{100,36,15\}$та тип оптимізатора (Adam, QNA-Adam).

Незмінні параметри наступні: $\mathrm{lr} = 3 \cdot 10^{-4}$, $\beta = (0.9,0.999)$, $\mathrm{eps} = 10^{-8}$, $\mathrm{weight\_decay} = 0$, $\mathrm{eval\_shots} = 2048$, reseed_each_epoch = false, однаковий batch size/epochs.

5.4 Процедура навчання

Процедура навчання у нас наступна:

- підготовка: завантажити конфігурацію, зафіксувати seed і детермінізм; створити робочу директорію запуску з snapshot конфігурації;
- вибір умови: встановити тренувальні шоти S ∈ {100, 36, 15} та оптимізатор {Adam, QNA-Adam}. Інші параметри лишаються незмінними;
- дані: ініціалізувати PhiDataset для train/val із тими самими PCA та Z-score, що використовуються в усіх умовах;
- модель: створити QuantumClassifier з HEA (L=2, ring, ry, reupload, ZX) на default.qubit, parameter-shift;
- тренування (N епох): на кожному батчі виконати прямий прохід, обчислити dL/dE, викликати Inline-DocPS для ∇θ та var_param; для QNA-Adam передати var_param у стан оптимізатора; виконати backward для голови, кліпінг за нормою (як у конфігурації), крок оптимізатора; зберігати батчеві метрики;

- валідація: наприкінці кожної епохи оцінити val_loss і val_acc з eval_shots = 2048; оновити best-checkpoint за мінімальним val_loss;
- логування й артефакти: записати metrics.csv (батчеві й епохальні рядки), profiler.jsonl, model.pt, config.snapshot.yaml; для QNA-Adam — qna_stats у metrics.csv;
- візуалізації: для кожного запуску побудувати криві залежно від епох: accuracy (train/val), loss (train/val), grad_norm, Vtilde; використати однакові шкали та легенди;

Результатом є шість відтворюваних запусків (3 значення S × 2 оптимізатори) з ідентичною процедурою, уніфікованими артефактами та порівнюваними візуалізаціями.

5.5 Метрики та артефакти

Усі експерименти логуються в runs/EXP_* (деталі див. 3.7). На рівні епох фіксуємо train_loss, train_acc, val_loss, val_acc, середні grad_norm_raw, grad_norm, clip_rate, Vtilde; для QNA-Adam додатково scale_mean, scale_p90, scale_min/max, scale_at_min/max, update_norm, cos_grad_update. На рівні батчів записуються ті самі базові поля плюс lr, shots та час кроку.

Best-checkpoint визначається за мінімальним val_loss; для порівнянь між умовами використовуємо епохальні ряди з metrics.csv. Валідація завжди з eval_shots = 2048.

5.6 Візуалізації (організація фігур)

Для кожної експериментальної умови, що відповідає комбінації $\{S \in \{100,36,15\}\} \times \{\text{Adam,QNA-Adam}\}$, будується окремий рисунок із єдиним стилем оформлення. Кожен рисунок має структуру сітки, де підграфіки розміщуються за типом метрики: на панелі A подається точність (Accuracy) для навчальної та валідаційної вибірок, на панелі B — функція втрат (Loss) для

тих самих підмножин, панель C відображає норму градієнта (Grad-norm) за епоху. На графіках A і B маркером позначається епоха, що відповідає найкращій моделі за критерієм найменшої валідаційної втрати (val_loss). Якщо під час тренування застосовувалася рання зупинка, на рисунку додатково відображається тонка вертикальна лінія, що вказує момент завершення навчання.

Для кожного типу метрики використовується спільна шкала по осі удля всіх рисунків, що забезпечує коректне візуальне порівняння між різними умовами. На графіках A і B легенда обмежується лише позначеннями train та val, тоді як основна інформація про умову експерименту (значення Sta тип оптимізатора) виноситься у заголовок рисунка. Підпис під рисунком коротко зазначає ключові параметри конфігурації — $\mathrm{L} = 2$, ring-топологія, оператор вимірювання ZX, швидкість навчання $\mathrm{lr} = 3 \times 10^{-4}$, а також кількість шотів під час валідації (eval_shots = 2048) і ідентифікатор запуску (EXP_).

У разі наявності кількох сідів для однієї експериментальної умови відображається усереднена крива з напівпрозорою смугою, що відображає варіативність результатів. Такий підхід забезпечує узгоджене, інтерпретоване порівняння між різними конфігураціями моделі та стабільність візуального представлення результатів.

6 РЕЗУЛЬТАТИ

6.1 Підсумкова таблиця умов

Результати експериментів на 100, 36 та 15 шотах надані в таблицях 6.1 та 6.2.

Таблиця 6.1 – Підсумкова таблиця. Частина 1

№	opt	Shots train	Best epoch	Val loss	Val acc	Train loss	Train acc	Grad norm	Grad norm raw
0	QNA	100	15	0.778684	0.7180	0.788516	0.7072	0.211358	0.211358
1	QNA	36	15	0.801714	0.7076	0.827313	0.6821	0.224562	0.224562
2	QNA	15	15	1.041991	0.5923	1.092081	0.4959	0.187175	0.187175
3	Adam	36	15	0.798115	0.7092	0.823952	0.6855	0.224274	0.224274
4	Adam	100	13	0.807857	0.7116	0.819332	0.6971	0.204738	0.204738
5	Adam	15	15	1.029374	0.5947	1.079076	0.4977	0.184618	0.184618

Таблиця 6.2 – Підсумкова таблиця. Частина 2

№	opt	Shots train	Scale mean	Scale min	Scale at max	Upd norm	Cos gu
0	QNA	100	0.9840	0.9152	0.8095	3.234536	0.4442
1	QNA	36	0.9854	0.9222	0.8095	3.111603	0.4474
2	QNA	15	0.9475	0.8229	0.6923	2.495611	0.5126
3	Adam	36	1.0000	1.0000	1.0000	3.143932	0.4455
4	Adam	100	1.0000	1.0000	1.0000	3.484532	0.4757
5	Adam	15	1.0000	1.0000	1.0000	2.545749	0.5000

6.1.1 Вплив кількості шотів

Якість очікувано спадає зі зменшенням S: для обох оптимізаторів val_acc приблизно 0.71–0.72 при S=100, ~0.71 при S=36 і ~0.59–0.595 при S=15. Це узгоджується з тим, що оцінки стають шумнішими за малих шотів.

Порівняння оптимізаторів за фіксованих S:

- S=100: QNA-Adam кращий за val_loss (0.7787 vs 0.8079) і val_acc (+0.64 п.п.);
- S=36: практично паритет (0.7076 vs 0.7092, −0.16 п.п. для QNA);
- S=15: також паритет (0.5923 vs 0.5947, −0.24 п.п. для QNA).

Висновок: на цих даних виграш QNA у точності помітний на S=100; на 36/15 точність близька до Adam.

Якщо ж розглядати тренувальну та валідаційна точність, то val_acc стабільно вища за train_acc у всіх умовах (наприклад, S=100 QNA: 0.718 vs 0.707). Причина — валідація рахується з eval_shots=2048.

6.1.2 Норми градієнтів і кліпінг

Grad_norm у межах 0.18–0.23 на best-епосі для всіх умов; кліпінг не спрацьовував (clipped=0.0). Це свідчить про стабільне навчання без «зривів» кроку.

6.1.3 Телеметрія QNA (масштабування кроку)

Scale_mean у QNA знижується на S=15 (0.9475) проти ~0.985 при 36–100, а scale_at_max падає (0.6923 проти 0.8095). Отже, за сильнішого шуму QNA зменшує ефективний крок частіше/сильніше. При цьому upd_norm у QNA нижчий, ніж у Adam, для всіх S (наприклад, S=100: 3.23 vs 3.48), що узгоджується з м’якшими оновленнями.

6.1.4 Рівень шуму вимірювань (Vtilde)

У таблиці явно зафіксовано для S=15 (~0.0294), що відповідає високій стохастичності. Для S=36/100 очікується менше значення (приблизно пропорційно 1/S); це добре видно на кривих Vtilde за епохами.

6.1.5 Best-epoch та динаміка

Для S=100 у Adam найкраща епоха 13-а, тоді як у QNA — 15-а (QNA продовжує покращуватись довше). Для S=36/15 best — 15-а в обох, що типово при повільнішій збіжності у шумному режимі.

6.1.6 Загальний висновок

QNA-Adam на цих запусках дає чіткий плюс на S=100 за валідною якістю і пом’якшує оновлення (нижчий upd_norm). За дуже малих шотів (36/15) фінальна точність лишається на рівні Adam, але телеметрія показує більш консервативні, стабільні оновлення, що корисно з позиції керованості тренування.

6.2 Вплив кількості шотів на якість

Зменшення тренувальних шотів із 100 до 36 і 15 однозначно погіршує валідаційну точність і підсилює стохастичність оцінок. За val_acc маємо монотонне падіння і для QNA-Adam (0.718 → 0.708 → 0.592), і для Adam (0.712 → 0.709 → 0.595), див. табл. 6.1. За val_loss картина узгоджена для QNA-Adam (0.7787 < 0.8017 < 1.0420), а для Adam різниця між 100 і 36 шотами мінімальна й навіть трохи краща на 36 (0.7981 проти 0.8079), проте при 15 шотах спостерігається чітке зростання втрат (1.0294). Отже, основний тренд визначається саме дефіцитом шотів: 15 шотів істотно погіршують якість, 36 — близькі до 100 за val_loss, але все ж поступаються за val_acc. Це узгоджується з підвищеним рівнем вимірювального шуму (Vtilde; для S=15 ≈ 0.0294), який робить навчання більш «пилкоподібним» на тренуванні, тоді як валідація на eval_shots = 2048 стабілізує оцінку, але не усуває впливу малих шотів на кінцеву якість. Повний набір кривих для всіх умов наведено в Додатку А: S=100 (рис. А.1–А.2), S=36 (рис. А.3–А.4), S=15 (рис. А.5–А.6).

6.3 Порівняння оптимізаторів при фіксованому S

S=100. QNA-Adam демонструє кращу валідаційну якість: val_loss 0.7787 проти 0.8079 і val_acc 0.7180 проти 0.7116 у Adam. QNA-Adam продовжує поліпшуватись до 15-ї епохи (у Adam best — на 13-й) і має нижчий upd_norm (3.23 проти 3.48), тобто робить м’якші оновлення за збереженої стабільності (clip_rate = 0).

S=36. Практично паритет: val_loss 0.8017 (QNA) проти 0.7981 (Adam) та val_acc 0.7076 (QNA) проти 0.7092 (Adam). Телеметрія все ж фіксує трохи нижчий upd_norm у QNA-Adam (3.11 проти 3.14), що відповідає більш консервативним крокам без втрати стабільності (clip_rate = 0).

S=15. Різниця мінімальна на користь Adam: val_loss 1.0420 (QNA) проти 1.0294 (Adam), val_acc 0.5923 проти 0.5947. При однаковому рівні шуму (Vtilde ≈ 0.0294) QNA-Adam зменшує ефективний крок (scale_mean 0.9475; scale_at_max 0.6923) і має нижчий upd_norm (2.50 проти 2.55), забезпечуючи більш обережні оновлення за подібної кінцевої точності. Порівняння Adam vs QNA-Adam: S=100 — рис. А.1 vs А.2; S=36 — А.3 vs А.4; S=15 — А.5 vs А.6

6.4 Динаміка збіжності та градієнти

За 100 шотів обидва оптимізатори збігаються рівномірно; QNA-Adam утримує плавніший спад loss і продовжує покращуватися довше (best на 15-й епосі проти 13-ї в Adam). За 36 шотів з’являються невеликі коливання, а за 15 — помітна «пилкоподібність» тренувальних кривих; водночас валідаційні криві лишаються стабільнішими завдяки eval_shots=2048. Загалом, зменшення шотів сповільнює збіжність і робить її більш «зубчастою», але не призводить до зривів навчання.

Епохальні норми градієнтів у найкращих точках лежать у вузькому діапазоні ~0.18–0.23 для всіх умов; кліпінг не спрацьовував (clip_rate=0), отже вибрані lr та анзац стабільні. Для QNA-Adam характерні нижчі upd_norm

(наприклад, 3.23 vs 3.48 при 100 шотах; 2.50 vs 2.55 при 15 шотах) і помірне зменшення ефективного кроку за підвищеного шуму (менший scale_mean, нижчий scale_at_max). Значення cos_gu≈0.44–0.51 свідчать про послідовне, але не ідеально колінеарне узгодження оновлень з градієнтом, що відповідає очікуванням для шумного режиму. Форми збіжності (панелі B) та норми градієнтів (панелі C) для всіх умов див. рис. A.1–A.6 (відповідні панелі B/C)

6.5 Телеметрія шуму та масштабування

Рівень вимірювального шуму оцінюємо через $\tilde{V}$(Vtilde). Для S=15 зафіксовано середнє $\tilde{V} \approx 0.0294$, що відображає високу стохастичність тренування; для S=36 та S=100 значення менші (приблизно спад ~$1/S$) і це добре видно на відповідних кривих $\tilde{V}$за епохами. Валідація з eval_shots=2048 стабілізує оцінки якості, але не усуває самого впливу шуму, що зумовлює гірші підсумкові метрики при малих S.

Механізм QNA-Adam реагує на шум через масштабування кроку. За S=100 і S=36 середній масштаб близький до одиниці (scale_mean ≈ 0.984–0.985, scale_p90 = 1.0, scale_at_max ≈ 0.81), тобто більшість параметрів оновлюються з повною ефективною швидкістю. За S=15 спостерігаємо помірніше оновлення: scale_mean = 0.9475, scale_at_max = 0.6923, тоді як нижня межа майже не досягається (scale_at_min = 0.0, scale_min = 0.8229). Це означає, що за сильнішого shot-noise оптимізатор м’яко зменшує крок у шумних координатах, не «душачи» навчання — верхній дециль лишається на повній швидкості (scale_p90 = 1.0). Така поведінка узгоджується з тенденцією: менше шотів, частіше й сильніше активують демпфування кроку в QNA-Adam.

6.6 Зв’язки між метриками

Підвищення вимірювального шуму (більший $\tilde{V}$за менших S) узгоджено відбивається на телеметрії та якості. У режимі QNA-Adam зі зростанням

$\tilde{V}$ зменшується середній масштаб кроку (scale_mean) і частка параметрів на верхній межі (scale_at_max), що призводить до менших оновлень (upd_norm) за збереження подібних grad_norm. Кліпінг не спрацьовує (clip_rate = 0), отже різницю в динаміці кроку пояснює саме шум-адаптивне масштабування, а не відсікання піків градієнтів. Показник узгодженості кроку з градієнтом (cos_gu ≈ 0.44–0.51) лишається стабільним у всіх умовах; за найшумнішого режиму (S=15) він не погіршується, що свідчить про відсутність “розвороту” оновлень у випадковому напрямку. На рівні підсумкової якості спостерігається очікуваний зв’язок: більший $\tilde{V}$ корелює з гіршою val_acc; шум-адаптивне зменшення кроку в QNA-Adam згладжує динаміку навчання (менший upd_norm), але не може повністю компенсувати втрату інформації за дуже малих шотів.

6.7 Загрози валідності та обмеження

Основні обмеження:

- симулятор проти реального заліза: експерименти виконано на default.qubit без апаратного шуму (декогеренції, помилок гейтів, cross-talk). Ми враховуємо лише shot-noise; отже результати оптимістичніші за роботу на реальному пристрої;
- невідповідність режимів train vs eval: навчання ведеться з малими shots $\in \{100,36,15\}$, тоді як валідація — з eval_shots=2048. Це знижує дисперсію метрик на val і може завищувати очікувану якість у сценаріях, де інференс теж обмежений шотами;
- обмежена предметна постановка: класифікація виконувалась на підмножині MNIST (4 класи) з PCA-редукцією до n_qubits; узагальнення на повні 10 класів, інші датасети або інші розмірності ознак не гарантується;
- фіксований анзац і вимірювання: архітектура HEA з L=2, топологія ring, кодування Ry з reupload та вимірювання ZX зафіксовані. Інші

анзаци/топології/схеми вимірювань можуть дати іншу картину збіжності та якості;

- гіперпараметри без ретюнінгу: єдиний lr=3e−4, однакові β, відсутній підбір по shots/оптимізатору. Це чесно для порівняння, але може не показати пік можливостей кожної умови (особливо для QNA-Adam);
- сід та варіативність: якщо використано один seed, міжзапускова дисперсія не оцінена. Для повної картини потрібні кілька сідів з агрегацією (mean ± sd);
- метрики епох рівня: висновки робимо за епохальними середніми; локальні батчеві флуктуації (особливо при S=15) згладжені й можуть бути непомітні у тексті.

ВИСНОВКИ

У результаті виконання цієї роботи реалізовано й апробовано повний пайплайн гібридної квантово-класичної моделі (PCA → Ry-reupload → HEA L=2, ring, вимірювання ZX) на симуляторі default.qubit з parameter-shift та власним шум-адаптивним оптимізатором QNA-Adam, який працює через штрафування надмірно шумних напрямків. Досліджено роль кількості шотів і вибору оптимізатора для навчання на підмножині MNIST (цифри 0 – 4), а також показано, як телеметрія $\tilde{v}$(оцінка shot-noise) може керувати масштабуванням кроку оновлення параметрів.

Емпірично встановлено: зі зменшенням шотів (100 → 36 → 15) валідаційні метрики погіршуються, що узгоджується зі зростанням стохастичності вимірювань. В результаті маємо паритет за підсумковою точністю, але стабільніші оновлення (менший upd_norm). Система логування (metrics.csv, profiler.jsonl, snapshot конфігів) забезпечує відтворюваність і прозорість експериментів; валідація з eval_shots=2048 стабілізує оцінювання якості.

Варто сказати, що симулятор не враховує апаратні помилки (декогеренція, неточність вентилів, cross-talk); train-shots відрізняються від eval-shots (2048), тому реальна продуктивність при обмежених шотах інференсу може бути нижчою; досліджено фіксований анзац і частину класів MNIST.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. J. Biamonte, P. Wittek, N. Pancotti, et al., “Quantum machine learning,” Nature, vol. 549, no. 7671, pp. 195–202, 2017.

2. M. Schuld and F. Petruccione, Supervised Learning with Quantum Computers, Springer, 2018.

3. M. Cerezo, A. Arrasmith, R. Babbush, et al., “Variational quantum algorithms,” Nature Reviews Physics, vol. 3, no. 9, pp. 625–644, 2021.

4. V. Havlíček, A. D. Córcoles, K. Temme, et al., “Supervised learning with quantum-enhanced feature spaces,” Nature, vol. 567, pp. 209–212, 2019.

5. E. Farhi and H. Neven, “Classification with quantum neural networks on near term processors,” arXiv:1802.06002, 2018.

6. M. Benedetti, E. Lloyd, S. Sack, M. Fiorentini, “Parameterized quantum circuits as machine learning models,” Quantum Science and Technology, vol. 4, no. 4, 2019.

7. H. Beer, D. Bondarenko, T. Farrelly, et al., “Training deep quantum neural networks,” Nature Communications, vol. 12, 2021.

8. M. Schuld, “Quantum machine learning models are kernel methods,” arXiv:2101.11020, 2021.

9. K. Mitarai, M. Negoro, M. Kitagawa, K. Fujii, “Quantum circuit learning,” Physical Review A, vol. 98, no. 3, 032309, 2018.

10. J. Preskill, “Quantum computing in the NISQ era and beyond,” Quantum, vol. 2, p. 79, 2018.

11. J. R. McClean, S. Boixo, V. N. Smelyanskiy, et al., “Barren plateaus in quantum neural network training landscapes,” Nature Communications, vol. 9, p. 4812, 2018.

12. A. W. Harrow and A. Montanaro, “Quantum computational supremacy,” Nature, vol. 549, pp. 203–209, 2017.

13. L. Zhou, S. Wang, S. Zhang, P. Zhang, and D. Tao, “Quantum

Approximate Optimization Algorithm: Performance, Mechanism, and Implementation on Near-Term Devices," *Quantum Engineering*, vol. 1, no. 1, p. e5, 2019.

14. R. Sweke, F. Wilde, J. Meyer, et al., "Stochastic gradient descent for hybrid quantum-classical optimization," *Quantum*, vol. 4, p. 314, 2020.

15. S. Wang, S. E. Economou, E. Barnes, and S. Das Sarma, "Noise-induced barren plateaus in variational quantum algorithms," Nature Communications, vol. 12, no. 1, p. 6961, 2021.

16. J. V. Dorband, "Rosalin: Robust shot-adaptive learning in noisy quantum circuits," Physical Review A, vol. 105, 042403, 2022.

17. PennyLane Team, "PennyLane Documentation: ShotAdaptiveOptimizer," PennyLane Official Documentation, [Online]. Available: https://docs.pennylane.ai/en/stable/code/api/pennylane.ShotAdaptiveOptimizer.html

18. Z. Sun et al., "Kalman Adam: Quantum-Noise Robust Adam for Variational Quantum Algorithms," *arXiv preprint* arXiv:2208.05890, 2022.

19. D. Reuther et al., "KOALA: Kalman Optimizer for Adaptive Learning-rate Adjustment in Variational Quantum Algorithms," *arXiv preprint* arXiv:2403.09890, 24.

20. M. S. Kutnii and Ye. I. Lytvynova, "Шумо-чутливий Adam для гібридних квантових моделей," Матеріали конференції «Актуальні проблеми комп'ютерних наук – АПКН-2025», Харків, Україна, 2025, pp. 245–246.