

Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет _____ Комп'ютерних наук
(повна назва)
Кафедра _____ Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти _____ перший (бакалаврський)

Гібридний квантово-класичний Vision Transformer для
малопараметричної класифікації зображень
(тема)

Виконав:
здобувач _____ четвертого _____ року навчання,
групи _____ ІТШ-22-4

Максим ВАСІЛІВ
(власне ім'я, прізвище)

Спеціальність 122 Комп'ютерні науки

(код і повна назва спеціальності)

Тип програми _____ освітньо-професійна

Освітня програма _____ Штучний інтелект

(повна назва освітньої програми)

Керівник проф. Олександр ШЕВЧЕНКО
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри ШІ

(підпис)

Лариса ЧАЛА
(власне ім'я, прізвище)

2026 р.

Харківський національний університет радіоелектроніки

Факультет _____ Комп'ютерних наук _____

Кафедра _____ Штучного інтелекту _____

Рівень вищої освіти _____ перший (бакалаврський) _____

Спеціальність _____ 122 Комп'ютерні науки _____
(код і повна назва)

Тип програми _____ освітньо-професійна _____

Освітня програма _____ Штучний інтелект _____
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____

(підпис)

«_____» _____ 20 ____ р.

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві _____ Васіліву Максиму Олександровичу _____
(прізвище, ім'я, по батькові)

1. Тема роботи _____ Гібридний квантово-класичний Vision Transformer для
малопараметричної класифікації зображень _____

затверджена наказом університету від 5 червня 2026 р. № 488Ст

2. Термін подання здобувачем роботи до екзаменаційної комісії 9 липня 2026 р.

3. Вихідні дані до роботи електронні ресурси за обраною тематикою, науково-технічні джерела з питань глибокого навчання, архітектур Vision Transformer, квантового машинного навчання та варіаційних квантових схем; мова програмування Python; бібліотеки PyTorch, TorchVision, PennyLane, NumPy, scikit-learn, pandas, matplotlib; набір даних MNIST та Fashion-MNIST для первинної та додаткової експериментальної перевірки; персональний комп'ютер для локального навчання та тестування моделей.

4. Перелік питань, що потрібно опрацювати в роботі _____

1) Аналіз предметної галузі та постановка задачі _____

2) Розробка варіаційної квантової схеми для гібридної моделі _____

3) Програмні засоби для реалізації гібридної квантово-класичної моделі _____

4) Опис програмної реалізації _____

5) Проведення експерименту та аналіз результатів _____

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / термін виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	08.06.2026	виконано
2	Аналіз предметної галузі та постановка задачі	10.06.2026	виконано
3	Аналіз архітектури Vision Transformer	12.06.2026	виконано
4	Аналіз варіаційних квантових схем та можливостей їх застосування	15.06.2026	виконано
5	Розробка математичної моделі квантової класифікаційної голови	17.06.2026	виконано
6	Вибір програмних засобів для реалізації моделі	17.06.2026	виконано
7	Програмна реалізація класичної моделі TinyViT з лінійною класифікаційною головою	19.06.2026	виконано
8	Програмна реалізація гібридної моделі QuantumTinyViT з варіаційною квантовою схемою	20.06.2026	виконано
9	Проведення первинного експерименту та аналіз результатів	22.06.2026	виконано
10	Проведення повторних експериментів за різних значень random seed	22.06.2026	виконано
11	Проведення додаткового експерименту на наборі Fashion-MNIST	24.06.2026	виконано
12	Побудова графіків, аналіз кількісних показників та формування висновків	24.06.2026	виконано
13	Підготовка пояснювальної записки	25.06.2026	виконано
14	Підготовка презентації та доповіді	26.06.2026	виконано
15	Захист перед ЕК	09.07.2026	

Дата видачі завдання 8 червня 2026 р.

Здобувач _____

(підпис)

Керівник роботи _____

(підпис)

проф. Олександр ШЕВЧЕНКО

(посада, власне ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка: 73 с., 16 рис., 2 дод., 17 джерел.

ВАРІАЦІЙНА КВАНТОВА СХЕМА, ГІБРИДНА КВАНТОВО-КЛАСИЧНА МОДЕЛЬ, КВАНТОВА НЕЙРОННА МЕРЕЖА, КЛАСИФІКАЦІЯ ЗОБРАЖЕНЬ, КУБІТ, МАЛОПАРАМЕТРИЧНА КЛАСИФІКАЦІЯ, PENNYLANE, PYTHON, PYTORCH, VISION TRANSFORMER.

Об'єкт дослідження – процес автоматичної класифікації зображень методами глибинного та квантового машинного навчання.

Предмет дослідження – гібридна квантово-класична архітектура малого Vision Transformer із варіаційною квантовою класифікаційною головою.

Мета роботи – розробка та дослідження малопараметричної гібридної моделі для класифікації зображень, що поєднує класичне формування ознак із квантовим варіаційним перетворенням.

Методи дослідження – аналіз джерел, математичне моделювання варіаційної квантової схеми, програмна реалізація моделей мовою Python із використанням PyTorch, TorchVision і PennyLane, а також експериментальне порівняння моделей за метриками accuracy, precision, recall і F1-score.

У результаті роботи розроблено модель QuantumTinyViT, яка поєднує малий Vision Transformer із варіаційною квантовою схемою. Класична частина формує компактне представлення зображення, а квантовий шар виконує кутове кодування, параметризовані операції та вимірювання Паулі-Z. Проведено порівняння з класичною моделлю TinyViT з лінійною головою та оцінено доцільність використання варіаційної квантової схеми для малопараметричної класифікації зображень.

ABSTRACT

Bachelor's thesis contains: 73 pp., 16 fig., 2 ann., 17 references.

HYBRID QUANTUM-CLASSICAL MODEL, IMAGE CLASSIFICATION, LOW-PARAMETER CLASSIFICATION, PENNYLANE, PYTHON, PYTORCH, QUANTUM NEURAL NETWORK, QUBIT, VARIATIONAL QUANTUM CIRCUIT, VISION TRANSFORMER.

The object of research is automatic image classification using deep learning and quantum machine learning methods.

The subject of research is a hybrid quantum-classical architecture of a small Vision Transformer with a variational quantum classification head.

The purpose of the work is to develop and investigate a low-parameter hybrid model for image classification that combines classical feature extraction with quantum variational transformation.

The research methods include source analysis, mathematical modeling of a variational quantum circuit, Python implementation using PyTorch, TorchVision and PennyLane, and experimental comparison using accuracy, precision, recall and F1-score.

As a result, the QuantumTinyViT model was developed. It combines a small Vision Transformer with a variational quantum circuit. The classical part forms a compact image representation, while the quantum layer performs angle embedding, parameterized operations and Pauli-Z measurements. The model was compared with a classical TinyViT with a linear head, and the feasibility of using a variational quantum circuit for low-parameter image classification was evaluated.

ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень і термінів	8
Вступ.....	11
1 Аналіз предметної галузі та постановка задачі.....	13
1.1 Класифікація зображень як задача комп'ютерного зору	13
1.2 Vision transformer та його застосування для обробки зображень	14
1.3 Обмеження класичних моделей та актуальність малопараметричного підходу.....	16
1.4 Використання варіаційних квантових схем у задачах машинного навчання	18
1.5 Постановка задачі.....	19
2 Розробка варіаційної квантової схеми для гібридної моделі	21
2.1 Основні поняття квантового машинного навчання	21
2.2 Кубіт, квантові оператори та вимірювання стану	22
2.3 Кутове кодування ознак у квантовий стан	25
2.4 Варіаційна квантова схема з параметризованими обертаннями та заплутуванням	26
2.5 Математична модель квантової класифікаційної голови	28
3 Програмні засоби для реалізації гібридної квантово-класичної моделі ..	31
3.1 Мова програмування python	31
3.2 Фреймворк pytorch та бібліотека torchvision.....	32
3.3 Бібліотека pennylane для моделювання квантових схем.....	33
3.4 Метрики оцінювання якості класифікації	34
4 Опис програмної реалізації	36
4.1 Загальна структура програмного рішення.....	36
4.2 Підготовка даних для експерименту	37
4.3 Реалізація малопараметричної моделі TinyViT	38
4.4 Реалізація класичної моделі з лінійною класифікаційною головою .	39
4.5 Реалізація варіаційної квантової схеми	40

4.6 Реалізація гібридної моделі QuantumTinyViT.....	41
4.7 Навчання, оцінювання та збереження результатів	42
5 Проведення експерименту та аналіз результатів	44
5.1 Первинний експеримент на двокласовій задачі класифікації	44
5.1.1 Умови проведення первинного експерименту.....	44
5.1.2 Динаміка навчання моделей.....	45
5.1.3 Порівняння якості класифікації.....	46
5.1.4 Аналіз матриць помилок	48
5.1.5 Аналіз кількості параметрів і часу навчання	50
5.1.6 Узагальнення результатів первинного експерименту	53
5.2 Перевірка стійкості результатів за різних random seed.....	54
5.2.1 Умови повторного експерименту	54
5.2.2 Аналіз середніх значень метрик на MNIST.....	54
5.2.3 Аналіз часу навчання на MNIST	56
5.2.4 Узагальнення результатів повторного експерименту	57
5.3 Додатковий експеримент на наборі Fashion-MNIST.....	57
5.3.1 Умови додаткового експерименту	57
5.3.2 Аналіз середніх значень метрик на Fashion-MNIST.....	58
5.3.3 Аналіз часу навчання на Fashion-MNIST	60
5.3.4 Порівняння результатів між mnist та Fashion-MNIST.....	61
5.3.5 Узагальнення результатів додаткового експерименту.....	62
Висновки	64
Перелік джерел посилання	68
Додаток А Код реалізації варіаційної квантової голови моделі QuantumTinyViT	70
Додаток Б Відомість кваліфікаційної роботи.....	73

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

Квантовий шар – компонент гібридної моделі, що виконує обчислення за допомогою квантової схеми або її симулятора;

Кубіт – базова одиниця квантової інформації, яка може перебувати у суперпозиції базових станів;

Патч – фрагмент зображення фіксованого розміру, який у Vision Transformer розглядається як окремий елемент послідовності;

ШІ – штучний інтелект;

AI – Artificial Intelligence – штучний інтелект;

AngleEmbedding – angle embedding – метод кодування класичних числових ознак у квантовий стан шляхом використання значень ознак як кутів обертання кубітів;

Ассурасу – accuracy – метрика, що показує частку правильно класифікованих прикладів серед усіх прикладів;

CLS-токен – classification token – спеціальний токен у transformer-архітектурах, представлення якого використовується для задач класифікації;

CNN – Convolutional Neural Network – згортова нейронна мережа;

CV – Computer Vision – комп'ютерний зір, напрям штучного інтелекту, що досліджує методи автоматичного аналізу, обробки та розпізнавання зображень;

DL – Deep Learning – глибинне навчання, підхід машинного навчання, що використовує багатoshарові нейронні мережі;

Embedding – embedding – векторне представлення об'єкта, наприклад патча зображення або узагальненого представлення всього зображення;

F1-score – F1-score – гармонійне середнє між precision і recall;

Fashion-MNIST – Fashion Modified National Institute of Standards and Technology database – набір зображень предметів одягу, який використовується для перевірки моделей класифікації зображень;

Logits – logits – ненормалізовані вихідні значення моделі перед визначенням імовірностей належності до класів;

ML – Machine Learning – машинне навчання, сукупність методів побудови моделей, здатних навчатися на основі даних;

MLP – Multilayer Perceptron – багатошаровий перцептрон, нейронна мережа з повнозв'язними шарами;

MNIST – Modified National Institute of Standards and Technology database – стандартний набір зображень рукописних цифр, який використовується для навчання та тестування моделей класифікації зображень;

NISQ – Noisy Intermediate-Scale Quantum – сучасний етап розвитку квантових пристроїв з обмеженою кількістю кубітів і наявністю шумів;

Pauli-Z – Pauli-Z operator – квантовий оператор, очікуване значення якого використовується для отримання класичного числового виходу з квантового шару;

PennyLane – PennyLane – програмна бібліотека для квантового машинного навчання та побудови гібридних квантово-класичних моделей;

PQC – Parameterized Quantum Circuit – параметризована квантова схема, що містить квантові гейти з керованими параметрами;

Precision – precision – метрика, що показує частку правильних позитивних передбачень серед усіх позитивних передбачень моделі;

PyTorch – PyTorch – програмний фреймворк для побудови, навчання та тестування нейронних мереж;

QML – Quantum Machine Learning – квантове машинне навчання, напрям досліджень, що поєднує методи машинного навчання та квантових обчислень;

QuantumTinyViT – Quantum Tiny Vision Transformer – гібридна квантово-класична модель, у якій малий Vision Transformer поєднується з варіаційною квантовою схемою;

Qubit – qubit – кубіт, базова одиниця квантової інформації;

Recall – recall – метрика, що показує частку знайдених позитивних прикладів серед усіх справжніх позитивних прикладів;

StronglyEntanglingLayers – strongly entangling layers – шаблон варіаційної квантової схеми, який поєднує параметризовані обертання кубітів і операції заплутування;

TinyViT – Tiny Vision Transformer – у межах цієї роботи малий Vision Transformer, що використовується як малопараметрична класична модель для класифікації зображень;

TorchVision – TorchVision – бібліотека екосистеми PyTorch для роботи з наборами даних, трансформаціями та моделями комп'ютерного зору;

Trainable parameters – trainable parameters – параметри моделі, які оновлюються під час навчання;

ViT – Vision Transformer – трансформерна архітектура для обробки зображень, у якій зображення подається як послідовність окремих патчів;

VQC – Variational Quantum Circuit – варіаційна квантова схема з параметризованими квантовими операціями, параметри яких оптимізуються під час навчання моделі.

ВСТУП

Класифікація зображень є однією з базових задач комп'ютерного зору та використовується в системах розпізнавання рукописних символів, медичної діагностики, відеоспостереження, промислового контролю якості та інших напрямках штучного інтелекту. Розвиток глибинного навчання значно підвищив якість автоматичного аналізу зображень, однак сучасні нейромережеві моделі часто мають велику кількість параметрів і потребують значних обчислювальних ресурсів. Тому актуальним є дослідження компактних моделей, здатних виконувати класифікацію зображень за обмеженого параметричного бюджету.

Одним із перспективних напрямів комп'ютерного зору є використання трансформерних архітектур. Vision Transformer показав, що механізм самоуваги може бути ефективно застосований не лише до текстових послідовностей, а й до зображень [1], [2]. У таких моделях зображення подається як послідовність патчів, після чого transformer-encoder формує узагальнене представлення для класифікації. Перевагою цього підходу є здатність враховувати глобальні залежності між різними частинами зображення.

Додатковий інтерес становить поєднання класичних нейромережевих моделей із методами квантового машинного навчання. Повністю квантові моделі поки що мають обмежене практичне застосування, проте гібридні квантово-класичні підходи можуть досліджуватися за допомогою симуляторів [4], [6]. У таких моделях класична частина виконує основну обробку даних, а квантовий модуль реалізує параметризоване перетворення компактного вектора ознак.

У межах цієї роботи розглядається гібридний квантово-класичний Vision Transformer для малопараметричної класифікації зображень. Основна ідея полягає у використанні малого Vision Transformer як класичного модуля формування ознак та варіаційної квантової схеми як елемента

класифікаційної голови. Зображення подається у вигляді послідовності патчів, transformer-encoder формує компактне представлення, після чого воно проєктується до малої кількості ознак і передається до квантового шару [5], [6].

Метою роботи є розробка та дослідження гібридної квантово-класичної моделі для малопараметричної класифікації зображень, у якій малий Vision Transformer поєднується з варіаційною квантовою схемою. Для досягнення мети необхідно проаналізувати предметну галузь, розглянути архітектуру Vision Transformer, дослідити принципи побудови варіаційних квантових схем, реалізувати класичну та гібридну моделі мовою Python і провести експериментальне порівняння їхньої якості.

Об'єктом дослідження є процес автоматичної класифікації зображень методами глибинного та квантового машинного навчання. Предметом дослідження є гібридна квантово-класична архітектура малого Vision Transformer, у якій варіаційна квантова схема використовується як елемент класифікаційної голови. Основну увагу приділено побудові компактної моделі з невеликою кількістю навчуваних параметрів.

Методами дослідження є аналіз науково-технічних джерел, математичне моделювання варіаційної квантової схеми, програмна реалізація моделей та експериментальна перевірка результатів.

Для реалізації використано Python, PyTorch, TorchVision і PennyLane [7], [10], [16]. Якість моделей оцінюється за метриками accuracy, precision, recall, F1-score, функцією втрат, кількістю параметрів і часом навчання.

Практична значущість роботи полягає у створенні програмного прототипу QuantumTinyViT для дослідження гібридних квантово-класичних моделей комп'ютерного зору. Запропонований підхід дозволяє оцінити доцільність використання варіаційної квантової схеми як альтернативи лінійній класифікаційній голові.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Класифікація зображень як задача комп'ютерного зору

Класифікація зображень є однією з основних задач комп'ютерного зору. Її суть полягає у визначенні класу, до якого належить вхідне зображення. На відміну від задач детекції об'єктів або сегментації, де необхідно локалізувати об'єкти чи виділяти їхні межі, класифікація передбачає формування одного або кількох підсумкових рішень щодо всього зображення. Такий тип задачі часто використовується як базовий етап для перевірки ефективності нових архітектур нейронних мереж, оскільки дозволяє оцінити здатність моделі виділяти характерні ознаки та узагальнювати їх для прийняття рішення.

У класичному підході до розв'язання цієї задачі модель отримує зображення у вигляді матриці пікселів і повинна перетворити його на числове представлення, достатнє для віднесення до певного класу. Для рукописних цифр такими ознаками можуть бути форма символу, товщина ліній, просторове розташування штрихів і співвідношення між окремими частинами зображення. У складніших наборах даних ознаки можуть відповідати текстурам, локальним деталям, кольоровим патернам або взаємному розташуванню об'єктів.

Тривалий час основним підходом до обробки зображень були згорткові нейронні мережі. Їхньою перевагою є здатність ефективно виділяти локальні ознаки за допомогою згорткових фільтрів. Проте розвиток трансформерних архітектур показав, що для зображень також можна застосовувати підхід, заснований на поданні даних у вигляді послідовності. Саме ця ідея лежить в основі Vision Transformer, де зображення розбивається на окремі фрагменти, які далі обробляються подібно до токенів у мовних трансформерах [1], [2].

У межах цієї роботи задача класифікації зображень розглядається не з позиції побудови максимально великої та точної моделі, а з позиції дослідження малопараметричної архітектури. Це означає, що важливими є не лише значення accuracy або F1-score, а й кількість навчуваних параметрів, простота архітектури та можливість локального запуску експерименту на звичайному комп'ютері.

1.2 Vision Transformer та його застосування для обробки зображень

Vision Transformer є адаптацією трансформерної архітектури до задач комп'ютерного зору [2]. Головна ідея такого підходу полягає в тому, що зображення не обробляється як єдина двовимірна структура, а попередньо розбивається на невеликі фрагменти, які називаються патчами. Кожен патч перетворюється на вектор ознак, після чого вся послідовність патчів подається до transformer-encoder [1], [2]. Завдяки цьому задача аналізу зображення зводиться до задачі обробки послідовності векторів.

У стандартних реалізаціях Vision Transformer до послідовності патчів додається спеціальний CLS-токен [2]. Після проходження transformer-encoder саме його представлення використовується як узагальнене представлення всього зображення. Такий підхід дозволяє моделі поступово накопичувати інформацію про взаємозв'язки між різними частинами зображення. На відміну від згорткових мереж, які природно орієнтовані на локальні ознаки, механізм самоуваги дозволяє враховувати взаємодію між усіма патчами, навіть якщо вони розташовані далеко один від одного [1].

Для великих задач комп'ютерного зору зазвичай використовуються значні за розміром Vision Transformer моделі. Вони можуть мати мільйони або десятки мільйонів параметрів і потребувати попереднього навчання на великих наборах зображень [2], [3]. Однак у цій роботі використання великої моделі не є основною метою. Навпаки, дослідження спрямоване на створення малого трансформерного класифікатора, який зберігає ключову

ідею Vision Transformer, але має невелику кількість параметрів і може бути навчений локально.

У запропонованій архітектурі використовується спрощена модель TinyViT. Вхідне зображення розбивається на патчі, після чого кожен патч перетворюється у векторне представлення за допомогою шару patch embedding. До послідовності додається CLS-токен і позиційне кодування, що дозволяє моделі враховувати порядок розташування патчів. Далі послідовність обробляється одним transformer-encoder шаром. На виході модель отримує компактний вектор ознак, який використовується для класифікації.

Такий підхід є достатнім для первинної експериментальної перевірки, оскільки дозволяє зберегти трансформерну природу моделі та водночас уникнути надмірної складності. У результаті можна окремо дослідити вплив класифікаційної голови: звичайної лінійної або квантово-класичної. Це важливо для коректного порівняння, оскільки основна відмінність між моделями має бути зосереджена саме в тій частині, де використовується варіаційна квантова схема.

У контексті цієї роботи схема Vision Transformer використовується не лише як теоретичне пояснення, а і як основа для побудови власної малопараметричної моделі. Важливо, що у запропонованій реалізації зберігаються ключові елементи ViT: поділ зображення на патчі, формування patch embedding, додавання CLS-токена та обробка послідовності transformer-encoder шаром. Водночас розмір embedding, кількість шарів і кількість attention heads навмисно зменшені, щоб модель залишалася компактною. Це дозволяє надалі коректно порівняти класичну та гібридну квантово-класичну класифікаційні голови за близької архітектурної основи.

Класична архітектура Vision Transformer передбачає поділ зображення на фіксовані патчі, лінійну проєкцію кожного патча у векторне представлення, додавання позиційного кодування та подальшу обробку отриманої послідовності transformer-encoder блоком. Після цього

узагальнене представлення зображення передається до MLP-голови для виконання класифікації. Базову схему архітектури Vision Transformer наведено на рисунку 1.1.

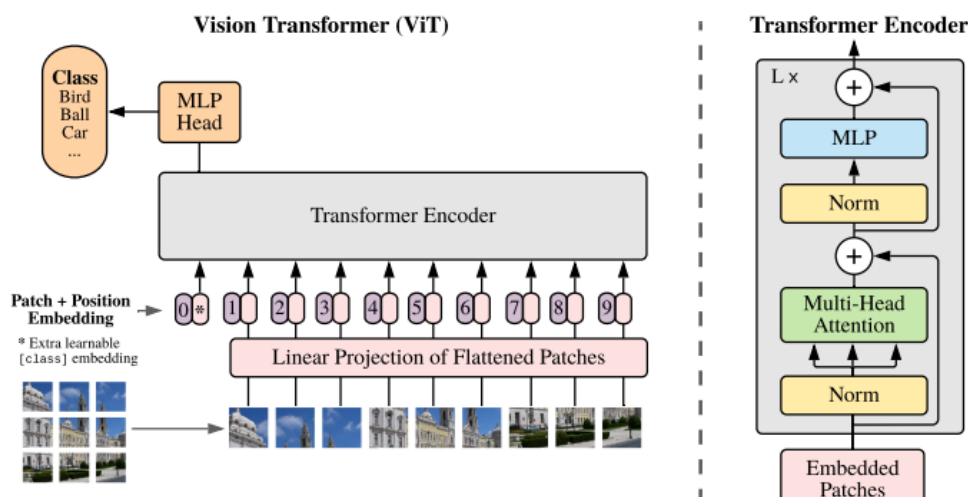


Рисунок 1.1 – Базова архітектура Vision Transformer для класифікації зображень [2]

Наведена схема узагальнює основні етапи обробки зображення у Vision Transformer. Для цієї роботи важливо, що така архітектура може бути зменшена до малопараметричного варіанта без втрати її базового принципу: подання зображення як послідовності патчів. Саме ця ідея використовується далі під час побудови TinyViT та гібридної моделі QuantumTinyViT.

1.3 Обмеження класичних моделей та актуальність малопараметричного підходу

Сучасні моделі комп'ютерного зору демонструють високу якість класифікації, однак часто потребують значних обчислювальних ресурсів [2], [3]. Збільшення кількості параметрів може покращувати здатність моделі до навчання складних залежностей, але водночас призводить до зростання часу навчання, вимог до пам'яті та ризику

перенавчання на малих наборах даних. Для дослідницьких і навчальних задач це створює додаткові труднощі, оскільки не завжди доцільно використовувати великі попередньо навчені моделі для перевірки окремої архітектурної ідеї.

Малопараметричний підхід передбачає побудову моделі, у якій кількість навчуваних параметрів свідомо обмежується. У такому випадку особливу роль відіграє не абсолютна потужність моделі, а ефективність використання кожного параметра. Для цієї роботи це має принципове значення, оскільки мета полягає не в тому, щоб отримати найвищу можливу точність на складному наборі даних, а в тому, щоб порівняти класичну та квантово-класичну класифікаційну голову за близького параметричного бюджету.

Класична модель у цій роботі використовує малий Vision Transformer і лінійну класифікаційну голову. Гібридна модель має таку саму основну структуру, але замість прямої лінійної класифікації після проєкції ознак використовується варіаційна квантова схема. Таким чином порівняння виконується не між великою класичною моделлю і малою квантовою моделлю, а між двома компактними моделями зі схожою кількістю навчуваних параметрів. Це робить експеримент більш коректним і дозволяє оцінити внесок саме квантового компонента.

Актуальність такого підходу також пов'язана з тим, що сучасні квантові обчислення перебувають на етапі, коли повністю квантові рішення для великих задач ще не є практичними [4]. Водночас малі варіаційні квантові схеми вже можна моделювати на класичному комп'ютері за допомогою спеціалізованих бібліотек [5], [6]. Тому дослідження гібридних моделей з невеликою кількістю кубітів є реалістичним напрямом, який дозволяє перевіряти принципову доцільність використання квантових шарів у задачах машинного навчання.

1.4 Використання варіаційних квантових схем у задачах машинного навчання

Варіаційні квантові схеми є одним із основних інструментів квантового машинного навчання [5], [6]. Така схема складається з квантових операцій, частина параметрів яких налаштовується під час навчання. У гібридній моделі класична частина готує вхідні ознаки, після чого вони кодуються у квантовий стан. Далі параметризовані квантові операції змінюють цей стан, а результат вимірювання повертається у класичну частину моделі для подальшого використання.

У запропонованій роботі варіаційна квантова схема використовується як частина класифікаційної голови. Це означає, що квантовий шар не працює безпосередньо з усім зображенням. Спочатку зображення обробляється малим Vision Transformer, який формує компактний вектор ознак. Після цього вектор проєктується до розмірності, що відповідає кількості кубітів. У першому експерименті використовується двокубітна схема, тому до квантового шару подаються дві числові ознаки.

Для кодування класичних ознак у квантовий стан використовується кутове кодування [8]. У цьому підході значення ознак інтерпретуються як кути обертання кубітів. Після кодування застосовується варіаційний шар із параметризованими квантовими операціями та операціями заплутування [9]. Наприкінці для кожного кубіта вимірюється очікуване значення оператора Паулі-Z. Отримані значення повертаються у класичну модель і використовуються для формування фінального передбачення.

Перевагою такого підходу є компактність квантового шару. Кількість його параметрів залежить від кількості кубітів і глибини схеми. У першій версії моделі використовується мала глибина та невелика кількість кубітів, що відповідає малопараметричній постановці задачі. Водночас квантовий шар є нелінійним перетворенням ознак, тому його порівняння зі звичайною

лінійною головою дозволяє перевірити, чи може така схема покращити якість класифікації без значного збільшення кількості параметрів.

1.5 Постановка задачі

Виходячи з аналізу предметної галузі, у роботі поставлено задачу розробити та дослідити гібридну квантово-класичну модель для малопараметричної класифікації зображень. Основна ідея полягає у поєднанні малого Vision Transformer, який виконує формування ознак із зображення, та варіаційної квантової схеми, що використовується як частина класифікаційної голови [2], [5], [6]. Такий підхід дозволяє зберегти трансформерну структуру моделі, але водночас перевірити можливість використання квантового шару замість звичайного класичного класифікаційного перетворення.

У роботі передбачається реалізувати дві моделі з близькою архітектурою. Перша модель є класичною та складається з малого Vision Transformer, проєкційного шару і лінійної класифікаційної голови. Друга модель є гібридною та використовує той самий принцип формування ознак, але після проєкції ознаки подаються до варіаційної квантової схеми. Після квантового шару результат передається до фінального класифікатора, який формує логіти для двох класів.

Для первинної перевірки обрано набір даних MNIST і двокласову постановку задачі, у якій класифікуються зображення цифр 0 та 1. Такий вибір дозволяє зменшити складність першого експерименту й зосередитися на перевірці працездатності гібридної архітектури. У подальших дослідженнях такий підхід може бути розширений на більшу кількість класів, складніші набори зображень і різні конфігурації квантового шару.

Експериментальне порівняння повинно враховувати не лише точність класифікації, а й кількість навчуваних параметрів, значення функції втрат, F1-score, матрицю помилок і час навчання. Це необхідно для того, щоб

оцінити не тільки якість передбачень, а й практичну доцільність використання варіаційної квантової схеми. Оскільки квантовий шар моделюється на класичному комп'ютері, очікується, що гібридна модель може працювати повільніше за класичну, однак при цьому важливо перевірити, чи здатна вона забезпечити конкурентну якість за близької кількості параметрів.

Таким чином, поставлена задача має як теоретичну, так і практичну складову. Теоретична складова полягає в аналізі можливості інтеграції варіаційної квантової схеми у малий Vision Transformer. Практична складова полягає у програмній реалізації класичної та гібридної моделей, їх навчанні на однакових даних і порівнянні результатів. Отримані результати повинні дати змогу зробити висновок щодо доцільності використання квантової класифікаційної голови у задачах малопараметричної класифікації зображень.

2 РОЗРОБКА ВАРІАЦІЙНОЇ КВАНТОВОЇ СХЕМИ ДЛЯ ГІБРИДНОЇ МОДЕЛІ

2.1 Основні поняття квантового машинного навчання

Квантове машинне навчання є напрямом досліджень, у якому методи машинного навчання поєднуються з принципами квантових обчислень [4], [6]. У межах сучасного етапу розвитку квантових технологій найбільш практичним є не повністю квантовий, а гібридний квантово-класичний підхід. У такій моделі частина обчислень виконується класичною нейронною мережею, а окремий блок реалізується у вигляді квантової схеми або її симуляції на класичному комп'ютері.

Особливість гібридної моделі полягає у тому, що квантова схема не працює безпосередньо з великим обсягом вхідних даних. Для зображень це особливо важливо, оскільки навіть просте зображення складається з великої кількості пікселів. Пряме кодування всіх пікселів у квантовий стан вимагало б значної кількості кубітів або складної процедури підготовки квантового стану. Тому в цій роботі використовується інший підхід: класична частина моделі спочатку формує компактний вектор ознак, а вже потім цей вектор передається до варіаційної квантової схеми.

Варіаційна квантова схема є параметризованою квантовою схемою, параметри якої оптимізуються під час навчання [5]. За логікою використання вона подібна до шару нейронної мережі: має вхідні значення, внутрішні навчувані параметри та вихід, який може бути використаний у наступній частині моделі. Водночас обчислення всередині такого шару виконуються не звичайними матричними операціями нейронної мережі, а послідовністю квантових операцій над кубітами.

У запропонованій архітектурі квантова схема використовується як частина класифікаційної голови. Модель TinyViT формує векторне представлення зображення, після чого це представлення стискається до

малої кількості ознак. У першій реалізації використовується дві ознаки, що відповідає двом кубітам квантової схеми. Далі ознаки кодуються у квантовий стан, проходять через варіаційне квантове коло, а результати вимірювання повертаються до класичної частини моделі для формування кінцевого передбачення.

2.2 Кубіт, квантові оператори та вимірювання стану

Базовою одиницею квантової інформації є кубіт. На відміну від класичного біта, який може перебувати лише в одному з двох станів, кубіт може бути у суперпозиції базових станів. Стан одного кубіта можна подати у вигляді:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \quad (2.1)$$

де $|\psi\rangle$ – квантовий стан кубіта;

α – комплексна амплітуда ймовірності для базового стану $|0\rangle$;

$|0\rangle$ – перший базовий стан кубіта;

β – комплексна амплітуда ймовірності для базового стану $|1\rangle$;

$|1\rangle$ – другий базовий стан кубіта.

Для фізично коректного квантового стану виконується умова нормування:

$$|\alpha|^2 + |\beta|^2 = 1, \quad (2.2)$$

де α – комплексна амплітуда ймовірності для базового стану $|0\rangle$;

β – комплексна амплітуда ймовірності для базового стану $|1\rangle$;

1 – одиничне значення повної ймовірності вимірювання кубіта в одному з базових станів.

Ця умова означає, що після вимірювання кубіт буде знайдений або у стані $|0\rangle$, або у стані $|1\rangle$, а сума відповідних ймовірностей дорівнює одиниці. До моменту вимірювання квантовий стан може містити інформацію про суперпозицію, але саме вимірювання повертає класичний результат.

Геометрично стан одного кубіта зручно інтерпретувати за допомогою сфери Блоха. У такому поданні чистий стан кубіта відповідає точці на поверхні одиничної сфери, а його положення визначається кутами повороту. Полюси сфери відповідають базовим станам $|0\rangle$ та $|1\rangle$, а інші точки поверхні описують можливі суперпозиції цих станів. Таке подання є корисним для цієї роботи, оскільки під час кутового кодування числові ознаки задають початкові повороти кубітів, після чого варіаційні квантові оператори додатково змінюють їхній стан у квантовій схемі.

Геометричне подання стану кубіта на сфері Блоха наведено на рисунку 2.1.

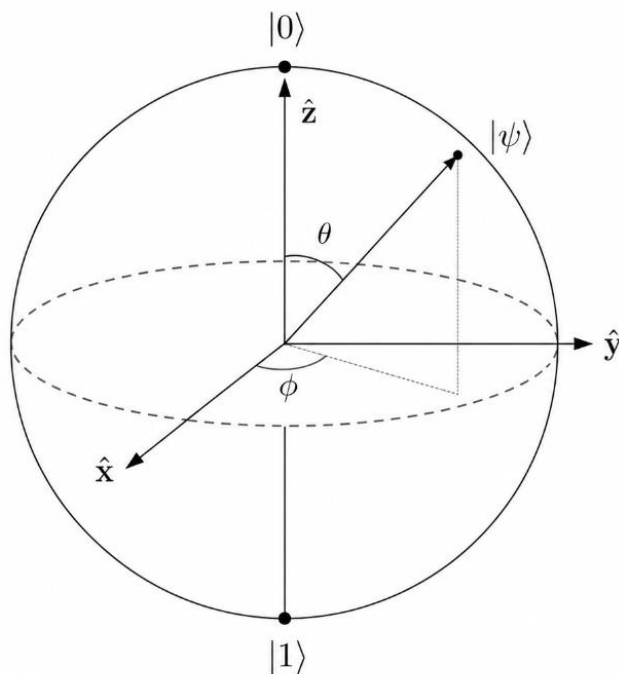


Рисунок 2.1 – Подання стану кубіта на сфері Блоха

Операції над кубітами виконуються за допомогою квантових операторів, або квантових гейтів. У цій роботі важливими є поворотні оператори, які змінюють стан кубіта залежно від заданого кута, а також оператори заплутування, що створюють взаємозв'язок між станами кількох кубітів. Поворот навколо осі Y можна подати оператором:

$$RY(\theta) = \begin{bmatrix} \cos\left(\frac{\theta}{2}\right) & -\sin\left(\frac{\theta}{2}\right) \\ \sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{bmatrix}, \quad (2.3)$$

де $RY(\theta)$ – оператор повороту стану кубіта навколо осі Y;

θ – кут повороту кубіта;

$\cos\left(\frac{\theta}{2}\right)$ – косинус половини кута повороту;

$\sin\left(\frac{\theta}{2}\right)$ – синус половини кута повороту.

Саме такий тип операції використовується для кодування класичних числових ознак у квантовий стан через кутове кодування.

Після виконання квантової схеми необхідно отримати класичні числові значення, які можуть бути використані нейронною мережею. Для цього застосовується вимірювання очікуваного значення оператора. У роботі використовується вимірювання очікуваного значення оператора Паулі-Z для кожного кубіта. У загальному вигляді очікуване значення оператора O у стані $|\psi\rangle$ записується як:

$$\langle O \rangle = \langle \psi | O | \psi \rangle, \quad (2.4)$$

де $\langle O \rangle$ – очікуване значення оператора O ;

$\langle \psi |$ – спряжений до $|\psi\rangle$ квантовий стан;

O – оператор, очікуване значення якого обчислюється;

$|\psi\rangle$ – квантовий стан системи.

Для оператора Паулі-Z результат належить діапазону від -1 до 1 . Це зручно для інтеграції квантового шару в нейронну мережу, оскільки вихід квантової схеми можна розглядати як звичайний числовий вектор ознак.

2.3 Кутове кодування ознак у квантовий стан

Для використання квантової схеми в задачі машинного навчання необхідно перетворити класичні ознаки у квантовий стан. У цій роботі використовується кутове кодування, оскільки воно є простим і придатним для малих варіаційних квантових схем [8]. У такому підході кожне числове значення з вхідного вектора використовується як кут повороту відповідного кубіта.

Нехай після класичної частини моделі отримано вектор ознак:

$$r = (r_1, r_2, \dots, r_q), \quad (2.5)$$

де r – вектор ознак, що подається до квантового шару;

r_1, r_2 – окремі компоненти вектора ознак;

r_q – q -та компонента вектора ознак;

q – кількість кубітів квантової схеми.

У першій реалізації $q = 2$, тому модель передає до квантового шару дві ознаки. Для кожної ознаки застосовується поворотний оператор R_Y , у результаті чого початковий стан кубітів змінюється відповідно до значень вхідного вектора. У цьому випадку кожна компонента вектора ознак керує поворотом окремого кубіта, тобто числове значення ознаки визначає кут квантової операції. Такий підхід дозволяє безпосередньо пов'язати класичні ознаки, сформовані TinyViT, зі станом квантової системи. Після виконання цих обертань інформація про вхідний вектор уже міститься у станах кубітів, які надалі можуть бути змінені варіаційною частиною схеми. Таким чином,

етап кутового кодування виконує роль переходу від класичного векторного представлення до квантового стану:

$$|0\rangle \otimes q \rightarrow U_{\text{enc}}(r)|0\rangle \otimes q, \quad (2.6)$$

де $|0\rangle$ – початковий базовий стан одного кубіта;

q – кількість кубітів квантової схеми;

$U_{\text{enc}}(r)$ – оператор кодування класичного вектора ознак у квантовий стан;

r – вектор ознак, який кодується у квантовий стан.

Тут $U_{\text{enc}}(r)$ позначає операцію кодування, що залежить від вхідного вектора r . У практичній реалізації це відповідає використанню функції `AngleEmbedding` з параметром `rotation='Y'` у бібліотеці `PennyLane` [8].

Перевагою кутового кодування є те, що воно потребує невеликої кількості операцій і природно узгоджується з малою кількістю кубітів. Його недоліком є те, що кількість ознак, які можна безпосередньо закодувати, зазвичай відповідає кількості кубітів. Саме тому перед квантовим шаром використовується проєкційний лінійний шар, який стискає представлення, сформоване `TinyViT`, до двох ознак. Таким чином зображення не кодується в квантовий стан напряду; до квантового шару потрапляє тільки компактне представлення, вже підготовлене класичною частиною моделі.

2.4 Варіаційна квантова схема з параметризованими обертаннями та заплутуванням

Після кодування ознак у квантовий стан застосовується варіаційна частина квантової схеми. Її завдання полягає у виконанні параметризованого перетворення квантового стану. Параметри цього перетворення є навчуваними, тобто вони оновлюються під час оптимізації так само, як ваги класичної нейронної мережі.

У цій роботі використовується схема, що складається з параметризованих обертань і операцій заплутування [9]. Такий підхід реалізується через шаблон StronglyEntanglingLayers у PennyLane [9]. Один шар такої схеми містить параметризовані однокубітні обертання та двокубітні операції, які забезпечують взаємодію між кубітами. Саме наявність операцій заплутування дозволяє квантовій схемі моделювати залежності між окремими ознаками.

Позначимо параметри варіаційної квантової схеми як ϕ . Тоді після кодування вхідного вектора r та застосування варіаційного перетворення стан квантової системи можна записати у вигляді:

$$|\psi(r, \phi)\rangle = U(\phi)U_{enc}(r)|0\rangle \otimes q, \quad (2.7)$$

де $|\psi(r, \phi)\rangle$ – квантовий стан після кодування ознак і застосування варіаційного перетворення;

r – вектор класичних ознак, що подається до квантової схеми;

ϕ – набір навчуваних параметрів варіаційної квантової схеми;

$U(\phi)$ – параметризоване квантове перетворення.

У першій версії моделі використовується невелика глибина квантової схеми. Це відповідає малопараметричній постановці задачі та дозволяє запускати експеримент локально на звичайному комп'ютері. Кількість навчуваних параметрів квантової частини залежить від кількості кубітів, глибини схеми та кількості параметризованих обертань у кожному шарі. Для використаної схеми з двома кубітами та одним шаром кількість параметрів квантового блоку є малою, що дозволяє порівнювати гібридну модель із класичною моделлю за близького параметричного бюджету.

Крім того, мала глибина квантової схеми зменшує ризик надмірної складності моделі та спрощує інтерпретацію отриманих результатів. У межах цієї роботи така конфігурація розглядається як базова, оскільки вона

дозволяє перевірити саму можливість використання VQC Head без ускладнення архітектури.

2.5 Математична модель квантової класифікаційної голови

У запропонованій гібридній моделі квантова схема не є окремим самостійним класифікатором, а входить до складу класифікаційної голови. Спочатку малий Vision Transformer перетворює зображення x на вектор ознак h :

$$h = B\theta(x), \quad (2.8)$$

де h – вектор ознак, сформований класичною частиною моделі;

$B\theta$ – класичний backbone на основі малого Vision Transformer;

θ – параметри класичної трансформерної частини моделі;

x – вхідне зображення.

Далі вектор h проєктується до малої розмірності, яка відповідає кількості кубітів:

$$r = W_p h + b_p, \quad (2.9)$$

де r – вектор ознак після проєкції до малої розмірності;

W_p – матриця ваг проєкційного шару;

h – вектор ознак, отриманий із класичної трансформерної частини;

b_p – вектор зміщення проєкційного шару.

Після цього вектор r кодується у квантовий стан і проходить через варіаційну квантову схему. На цьому етапі параметризовані квантові операції змінюють стан кубітів відповідно до навчуваних параметрів моделі. Після завершення квантового перетворення необхідно отримати класичні числові значення, які можуть бути передані до наступного шару

нейронної мережі. Для цього для кожного кубіта вимірюється очікуване значення оператора Паулі-Z. Таке вимірювання не повертає сам квантовий стан, а формує числову характеристику стану кожного кубіта, придатну для подальшої класичної обробки:

$$u_j = \langle \psi(r, \varphi) | Z_j | \psi(r, \varphi) \rangle, j = 1, \dots, q, \quad (2.10)$$

де u_j – вихідне значення квантового шару для j -го кубіта;

$\langle \psi(r, \varphi) |$ – квантовий стан після кодування ознак і застосування варіаційного шару;

Z_j – оператор Паулі-Z для j -го кубіта;

j – індекс кубіта.

Отримані значення формують вектор:

$$u = (u_1, u_2, \dots, u_q), \quad (2.11)$$

де u – вихідний вектор квантового шару;

u_1, u_2 – значення, отримані після вимірювання перших кубітів;

u_q – значення, отримане після вимірювання q -го кубіта.

Цей вектор подається до фінального класичного шару, який формує логіти класів:

$$\hat{y} = W_o u + b_o, \quad (2.12)$$

де $\hat{y}$ – вектор логітів, сформований моделлю;

W_o – матриця ваг вихідного класифікатора;

b_o – вектор зміщення вихідного класифікатора.

У задачі двокласової класифікації цей вектор має дві компоненти, що відповідають двом класам.

Таким чином, повна гібридна модель може бути подана як послідовність класичного формування ознак, проєкції до малої розмірності, квантового варіаційного перетворення та фінальної класичної класифікації. Така структура дозволяє ізолювати вплив квантової частини та порівняти її зі звичайною лінійною класифікаційною головою за близької кількості навчуваних параметрів.

Структурно квантова класифікаційна голова складається з проєкційного шару, етапу кодування ознак у квантовий стан, варіаційного квантового блоку, вимірювання очікуваних значень операторів Паулі-Z та фінального класичного класифікатора. Таке представлення наведено на рисунку 2.2.

Структура квантової класифікаційної голови

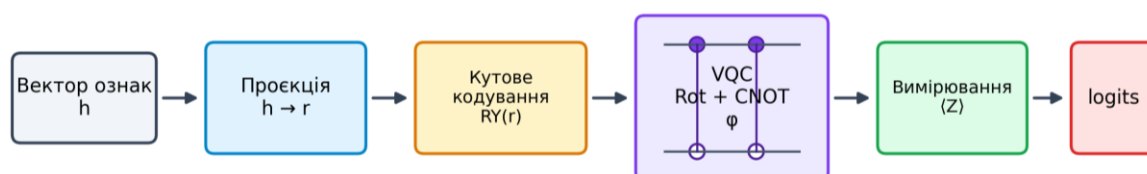


Рисунок 2.2 – Структура квантової класифікаційної голови VQC Head

Наведена структура показує, що квантова класифікаційна голова є частиною загального нейромережевого pipeline, а не окремим незалежним класифікатором. Вона приймає компактний вектор ознак, виконує квантове перетворення та повертає числовий вектор, придатний для подальшої класичної класифікації. На основі цієї математичної моделі далі розглядаються програмні засоби, необхідні для її реалізації.

3 ПРОГРАМНІ ЗАСОБИ ДЛЯ РЕАЛІЗАЦІЇ ГІБРИДНОЇ КВАНТОВО-КЛАСИЧНОЇ МОДЕЛІ

3.1 Мова програмування Python

Для програмної реалізації гібридної квантово-класичної моделі у роботі використовується мова програмування Python. Її вибір зумовлений широким застосуванням у сфері машинного навчання, глибинного навчання, комп'ютерного зору та квантових обчислень. Python має розвинену екосистему бібліотек, що дозволяє реалізовувати повний цикл дослідження: завантаження даних, попередню обробку зображень, побудову нейронної мережі, створення квантової схеми, навчання моделі, обчислення метрик і збереження результатів експерименту.

Важливою перевагою Python є можливість поєднання різних інструментів в одному програмному середовищі. У межах цієї роботи класична частина моделі реалізується засобами PyTorch, робота зі стандартними наборами зображень виконується через TorchVision, квантова схема моделюється за допомогою PennyLane, а метрики якості обчислюються з використанням scikit-learn. Така інтеграція дозволяє будувати гібридну модель як єдиний програмний об'єкт і навчати її у стандартному циклі оптимізації.

Оскільки метою роботи є не промислове розгортання моделі, а дослідження архітектурного підходу, Python є доцільним вибором для експериментальної реалізації. Він забезпечує достатню гнучкість для швидкої зміни структури моделі, кількості кубітів, глибини квантової схеми, параметрів навчання та набору метрик. Це особливо важливо для дослідницької роботи, у якій необхідно порівнювати класичну та гібридну квантово-класичну моделі в однакових умовах.

3.2 Фреймворк PyTorch та бібліотека TorchVision

Для побудови класичної частини моделі використовується фреймворк PyTorch [11], [12], [13], [14], [15]. Він надає засоби для створення нейронних мереж, роботи з тензорами, автоматичного обчислення градієнтів і реалізації циклу навчання. У цій роботі PyTorch використовується для реалізації малого Vision Transformer, проєкційного шару, класичної лінійної класифікаційної голови, а також для навчання обох моделей за допомогою оптимізатора та функції втрат.

Основною класичною частиною моделі є малий Vision Transformer, який реалізовано у вигляді окремого модуля. У ньому зображення спочатку проходить через шар patch embedding, який розбиває зображення на патчі та перетворює їх у векторні представлення [12]. Далі до послідовності додається спеціальний CLS-токен і позиційне кодування. Після цього послідовність обробляється transformer-encoder шаром, а вихідне представлення CLS-токена використовується як компактний опис усього зображення [11].

PyTorch також використовується для реалізації двох порівнюваних моделей. Перша модель є класичною і містить малий Vision Transformer, проєкційний шар та лінійний класифікатор. Друга модель має таку саму класичну основу, але після проєкційного шару ознаки передаються до варіаційної квантової схеми. Завдяки цьому обидві моделі мають близьку структуру, а основна відмінність між ними зосереджена саме в класифікаційній голові.

Для завантаження та підготовки зображень використовується бібліотека TorchVision [16]. Вона надає доступ до стандартних наборів даних комп'ютерного зору, зокрема MNIST, Fashion-MNIST і CIFAR-10. Для первинної експериментальної перевірки доцільно використовувати простий набір даних, оскільки це дозволяє швидко перевірити працездатність запропонованої архітектури та коректність інтеграції

квантового шару. Надалі та сама програмна структура може бути використана для складніших наборів зображень без принципової зміни архітектури моделі.

Перед поданням до моделі зображення перетворюються у тензори та нормалізуються. Така попередня обробка необхідна для стабільнішого навчання нейронної мережі. Далі дані передаються до DataLoader, який формує батчі для навчання та тестування. Це дозволяє організувати стандартний процес навчання, у якому модель поступово проходить через набір прикладів, обчислює функцію втрат і оновлює навчувані параметри.

3.3 Бібліотека PennyLane для моделювання квантових схем

Для реалізації квантової частини моделі використовується бібліотека PennyLane [7], [8], [9], [10]. Вона призначена для побудови, моделювання та оптимізації квантових схем, а також підтримує інтеграцію з фреймворками машинного навчання. У межах цієї роботи PennyLane використовується для створення варіаційної квантової схеми, яка виконує роль квантового шару в гібридній моделі.

Квантова схема реалізується на симуляторі, тобто для проведення експерименту не потрібен реальний квантовий комп'ютер. Це важливо для практичної реалізації роботи, оскільки модель можна навчати локально на звичайному персональному комп'ютері. У першій версії експерименту використовується невелика кількість кубітів, тому симуляція є технічно можливою навіть без спеціалізованого квантового обладнання.

У запропонованій моделі квантовий шар приймає на вхід невеликий вектор ознак, сформований класичною частиною моделі. Для кодування цих ознак у квантовий стан використовується кутове кодування. У PennyLane це реалізується за допомогою AngleEmbedding, де кожна ознака інтерпретується як кут обертання відповідного кубіта [8]. Після цього

застосовується варіаційна частина схеми, що містить параметризовані квантові операції та операції заплутування.

Для побудови варіаційного шару використовується шаблон `StronglyEntanglingLayers` [9]. Він поєднує параметризовані обертання кубітів з операціями, які створюють взаємодію між кубітами. Параметри цього шару є навчуваними і змінюються під час оптимізації моделі. На виході квантової схеми для кожного кубіта вимірюється очікуване значення оператора Паулі-Z. Отримані числові значення повертаються у класичну частину моделі та використовуються для формування кінцевого передбачення.

Важливою властивістю `PennyLane` є можливість інтеграції квантової схеми в `PyTorch`-модель. Завдяки цьому варіаційна квантова схема може бути використана як звичайний шар нейронної мережі. У програмній реалізації квантова схема оформлюється як `TorchLayer`, після чого її можна викликати всередині методу `forward` гібридної моделі [7], [8], [9], [10]. Це дозволяє навчати класичні та квантові параметри в межах одного загального циклу навчання.

3.4 Метрики оцінювання якості класифікації

Для оцінювання результатів роботи моделей використовуються стандартні метрики класифікації. Основною метрикою є *accuracy*, яка показує частку правильно класифікованих прикладів серед усіх прикладів тестової вибірки. Ця метрика є зрозумілою для задач класифікації та дозволяє швидко оцінити загальний рівень якості моделі.

Окрім *accuracy*, у роботі використовуються *precision*, *recall* та *F1-score*. *Precision* характеризує, яка частина прикладів, віднесених моделлю до позитивного класу, справді належить до цього класу. *Recall* показує, яку частину справжніх позитивних прикладів модель змогла правильно знайти.

F1-score є узагальненою метрикою, яка враховує одночасно precision і recall, тому її доцільно використовувати для порівняння моделей разом з accuracy.

Додатково аналізується значення функції втрат на навчальній і тестовій вибірках. Функція втрат показує, наскільки передбачення моделі відрізняються від правильних міток класів. У задачі класифікації використовується CrossEntropyLoss, яка є стандартною функцією втрат для моделей, що формують логіти класів [14]. Зменшення значення цієї функції під час навчання свідчить про те, що модель поступово краще підлаштовується під навчальні дані.

Для більш повного аналізу результатів також використовується матриця помилок. Вона дозволяє побачити не тільки загальну точність, а й те, які саме класи модель плутає між собою. У двокласовій постановці матриця помилок показує кількість правильно та неправильно розпізнаних прикладів кожного класу.

Оскільки тема роботи пов'язана з малопараметричною класифікацією, окремо враховується кількість навчуваних параметрів моделі. Це дозволяє оцінити не лише якість класифікації, а й те, наскільки компактною є модель. Крім того, фіксується час навчання, оскільки гібридна модель із квантовим шаром може працювати повільніше через симуляцію квантової схеми на класичному комп'ютері. Таким чином, порівняння класичної та квантово-класичної моделей виконується не за однією метрикою, а за сукупністю показників якості, складності та практичної ефективності.

4 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

4.1 Загальна структура програмного рішення

Програмна реалізація роботи побудована як експериментальний pipeline для порівняння класичної малопараметричної моделі та гібридної квантово-класичної моделі. Основна мета програмної частини полягає не в розробці великої промислової системи класифікації зображень, а в перевірці працездатності запропонованої архітектурної ідеї: заміни звичайної класифікаційної голови малого Vision Transformer на голову, що містить варіаційну квантову схему.

Програма реалізована мовою Python із використанням бібліотек PyTorch, TorchVision, PennyLane, NumPy, pandas та scikit-learn. Класична частина моделі реалізована засобами PyTorch, підготовка зображень виконується через TorchVision, а квантовий шар створюється за допомогою PennyLane. Усі компоненти поєднані в єдиний цикл навчання, у якому модель отримує батч зображень, формує передбачення, обчислює функцію втрат, виконує зворотне поширення помилки та оновлює навчувані параметри.

У програмі реалізовано дві основні моделі. Перша модель є класичною і складається з малого Vision Transformer, проєкційного шару та лінійної класифікаційної голови. Друга модель є гібридною квантово-класичною і має ту саму трансформерну основу, але після проєкції ознак використовує варіаційну квантову схему. Такий підхід дозволяє зробити порівняння коректним, оскільки обидві моделі мають близьку кількість навчуваних параметрів, а основна відмінність між ними полягає саме у класифікаційній голові.

Загальну структуру програмного рішення, у якому однакова трансформерна основа використовується для класичної та гібридної моделей, наведено на рисунку 4.1. Схема відображає основний pipeline:

підготовку зображення, формування ознак за допомогою TinyViT, проєкцію до малої розмірності та подальше використання або лінійної, або квантово-класичної голови.

Загальна структура програмного рішення

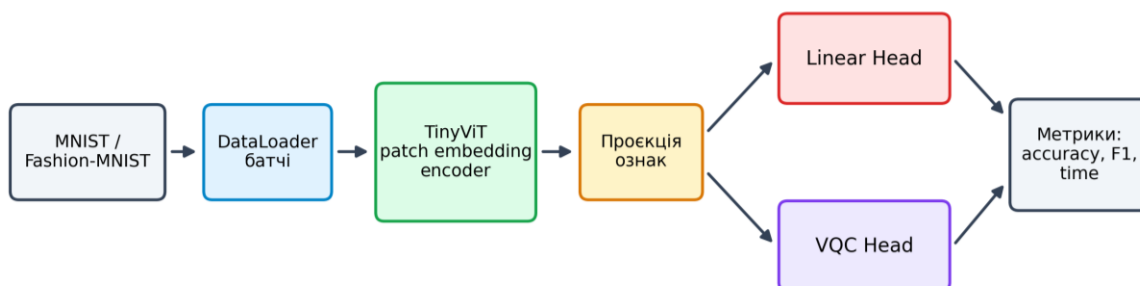


Рисунок 4.1 – Загальна структура програмного рішення для порівняння TinyViT і QuantumTinyViT

Наведена схема демонструє, що обидві моделі мають спільну трансформерну основу, а відмінність між ними зосереджена у класифікаційній голові. Це дозволяє проводити коректне порівняння між класичним і гібридним підходами. Далі окремо розглядається підготовка даних, реалізація компонентів моделі та організація навчального циклу.

4.2 Підготовка даних для експерименту

Для первинної експериментальної перевірки використовується стандартний набір зображень, доступний через бібліотеку TorchVision [16]. На початковому етапі доцільно використовувати просту двокласову задачу, оскільки вона дозволяє швидко перевірити коректність роботи моделі, навчального циклу та квантового шару. Така постановка не обмежує запроповану архітектуру лише одним набором даних, оскільки в подальшому той самий pipeline може бути застосований до складніших наборів зображень.

Зображення перед поданням до моделі перетворюються у тензори та нормалізуються. Перетворення у тензор необхідне для подальшої обробки засобами PyTorch, а нормалізація допомагає стабілізувати процес навчання. Після цього зображення передаються до DataLoader, який формує батчі фіксованого розміру для навчання та тестування. Використання батчів дозволяє ефективніше організувати обчислення та виконувати оновлення параметрів моделі після обробки групи прикладів.

У початковому експерименті використовується обмежена кількість навчальних і тестових прикладів. Це зроблено для того, щоб експеримент можна було виконати локально на звичайному персональному комп'ютері без потреби у спеціалізованому обладнанні. Такий підхід є достатнім для перевірки працездатності гібридної моделі, оскільки на цьому етапі основним завданням є не досягнення максимально можливої точності, а порівняння поведінки класичної та квантово-класичної класифікаційних голів.

4.3 Реалізація малопараметричної моделі TinyViT

Класична частина обох моделей реалізована у вигляді малого Vision Transformer. У програмі вона винесена в окремий модуль, який виконує роль backbone, тобто модуля формування ознак із вхідного зображення. Вхідне зображення спочатку проходить через шар patch embedding [12]. Цей шар розбиває зображення на окремі патчі та перетворює кожен патч у вектор фіксованої розмірності.

Після формування послідовності патчів до неї додається спеціальний CLS-токен. Його призначення полягає в тому, щоб після обробки transformer-encoder отримати узагальнене представлення всього зображення. Також до послідовності додається позиційне кодування, яке дозволяє моделі враховувати розташування патчів у зображенні. Без

позиційного кодування transformer-encoder не мав би явної інформації про порядок елементів послідовності.

Далі послідовність токенів передається до transformer-encoder шару [11]. У цій роботі використовується малий варіант такої архітектури з невеликою розмірністю embedding, малою кількістю attention heads і обмеженою кількістю transformer layers. Це відповідає малопараметричній постановці задачі та дозволяє навчати модель локально. Після проходження transformer-encoder з виходу береться представлення CLS-токена, яке далі використовується як компактний вектор ознак зображення.

4.4 Реалізація класичної моделі з лінійною класифікаційною головою

Класична модель використовується як baseline для порівняння з гібридною квантово-класичною моделлю. Вона складається з малого Vision Transformer, проєкційного шару та звичайної лінійної класифікаційної голови. Після того як backbone формує вектор ознак зображення, цей вектор передається до проєкційного шару, який зменшує його розмірність до невеликої кількості ознак.

Проекційний шар потрібен для того, щоб зробити класичну та гібридну моделі максимально подібними за структурою. У класичній моделі після проєкції ознаки одразу передаються до лінійного класифікатора, який формує логіти класів [13]. У гібридній моделі на цьому самому місці між проєкційним шаром і фінальним класифікатором розміщується варіаційна квантова схема.

Таким чином, класична модель дає змогу оцінити, якої якості можна досягти без використання квантового шару. Вона є необхідною для коректного експерименту, оскільки результат гібридної моделі має порівнюватися не з абстрактним значенням точності, а з конкретною класичною моделлю, що має близьку архітектуру та близький параметричний бюджет.

4.5 Реалізація варіаційної квантової схеми

Квантова частина моделі реалізована за допомогою бібліотеки PennyLane [7], [8], [9], [10]. У програмі створюється квантовий пристрій-симулятор `default.qubit`, який дозволяє моделювати роботу квантової схеми на класичному комп'ютері. У початковій реалізації використовується мала кількість кубітів, що відповідає кількості ознак, які подаються до квантового шару після проєкційного шару.

Вхідні ознаки кодуються у квантовий стан за допомогою `AngleEmbedding` [8]. У цій роботі використовується кутове кодування, за якого кожне числове значення інтерпретується як кут повороту відповідного кубіта. Такий спосіб кодування є простим для реалізації та добре підходить для невеликих варіаційних квантових схем.

Після кодування ознак застосовується варіаційний шар `StronglyEntanglingLayers` [9]. Він містить параметризовані квантові операції та операції заплутування між кубітами. Параметри цього шару є навчуваними, тому під час оптимізації вони змінюються разом з іншими параметрами моделі. Саме ця частина квантової схеми виконує нелінійне перетворення вхідних ознак у квантовому просторі.

На виході квантової схеми для кожного кубіта вимірюється очікуване значення оператора Паулі-Z. Отримані значення є звичайними числовими величинами, які можуть бути передані назад у класичну частину моделі. У програмній реалізації квантова схема оформлюється як `TorchLayer`, що дозволяє використовувати її всередині PyTorch-модуля так само, як звичайний шар нейронної мережі [7], [8], [9], [10].

Схематичне зображення реалізованої двокубітної варіаційної квантової схеми наведено на рисунку 4.2. На вході кожна ознака кодується у відповідний кубіт за допомогою обертання RY , після чого застосовується варіаційний блок із параметризованими обертаннями та операціями

заплутування. На виході для кожного кубіта виконується вимірювання, результат якого повертається до класичної частини моделі.

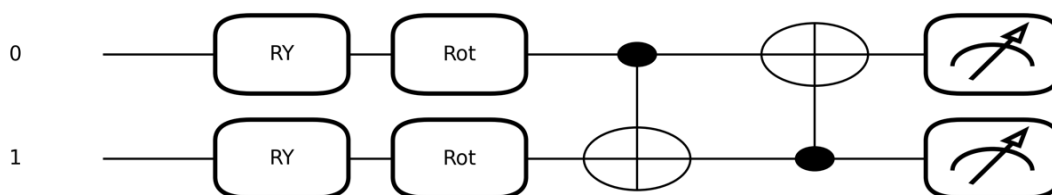


Рисунок 4.2 – Варіаційна квантова схема для двох кубітів у моделі QuantumTinyViT

Наведена схема відповідає квантовій частині моделі, яка використовується після проєкційного шару. Вона показує послідовність операцій, що забезпечують кодування ознак, варіаційне перетворення та отримання числового виходу через вимірювання. Далі ця квантова схема інтегрується до повної гібридної архітектури QuantumTinyViT.

Фрагмент програмної реалізації варіаційної квантової голови, що містить створення квантового шару, використання AngleEmbedding, StronglyEntanglingLayers та інтеграцію квантової схеми в модель QuantumTinyViT, наведено у додатку А.

4.6 Реалізація гібридної моделі QuantumTinyViT

Гібридна модель QuantumTinyViT має ту саму класичну основу, що й baseline-модель, але відрізняється структурою класифікаційної голови. Спочатку зображення проходить через малий Vision Transformer, який формує компактне представлення на основі CLS-токена. Потім це представлення подається до проєкційного шару, який зменшує розмірність вектора до кількості ознак, сумісної з кількістю кубітів.

Після проєкції ознаки передаються до варіаційної квантової схеми. Квантовий шар виконує параметризоване перетворення ознак і повертає

вектор очікуваних значень операторів Паулі-Z. Цей вектор передається до фінального лінійного шару, який формує логіти класів. Таким чином, QuantumTinyViT поєднує класичну обробку зображення та квантове перетворення компактного вектора ознак.

Перевагою такої структури є те, що квантовий шар не працює з повним зображенням або великою кількістю пікселів. Він отримує лише невелику кількість ознак, уже сформованих класичною частиною моделі. Це робить експеримент обчислювально реалістичним і дозволяє використовувати малу кількість кубітів. Крім того, така архітектура дозволяє ізолювати вплив квантової частини, оскільки класична та гібридна моделі мають однакову основу.

4.7 Навчання, оцінювання та збереження результатів

Навчання обох моделей виконується в однакових умовах. Для цього використовується однаковий набір даних, однакова попередня обробка зображень, однаковий розмір батчу, однакова кількість епох і однакова функція втрат. Такий підхід потрібен для того, щоб різниця в результатах пояснювалася саме архітектурною відмінністю між моделями, а не різними умовами експерименту.

Як функція втрат використовується CrossEntropyLoss, що є стандартним вибором для задач класифікації [14]. Для оновлення параметрів застосовується оптимізатор Adam [15]. У процесі навчання модель отримує батч зображень, формує логіти класів, після чого обчислюється значення функції втрат. Далі виконується зворотне поширення помилки та оновлення параметрів моделі.

Після кожної епохи модель оцінюється на тестовій вибірці. Для оцінювання використовуються accuracy, precision, recall, F1-score, значення функції втрат і матриця помилок. Крім того, для кожної моделі обчислюється кількість навчуваних параметрів і час навчання. Ці показники

є важливими, оскільки робота присвячена саме малопараметричній класифікації зображень, а не лише максимізації точності.

Результати експерименту зберігаються у файл формату CSV. Це дозволяє надалі аналізувати їх у вигляді таблиць, будувати графіки та порівнювати різні конфігурації моделей. Такий формат збереження є зручним для подальшого розширення експериментів, наприклад при зміні кількості кубітів, глибини квантової схеми або набору даних.

5 ПРОВЕДЕННЯ ЕКСПЕРИМЕНТУ ТА АНАЛІЗ РЕЗУЛЬТАТІВ

5.1 Первинний експеримент на двокласовій задачі класифікації

5.1.1 Умови проведення первинного експерименту

Для перевірки працездатності запропонованої гібридної квантово-класичної архітектури було проведено первинний експеримент, у якому порівнювалися дві малопараметричні моделі класифікації зображень. Перша модель є класичною та складається з малого Vision Transformer, проєкційного шару і лінійної класифікаційної голови. Друга модель має таку саму класичну трансформерну основу, але після проєкційного шару використовує варіаційну квантову схему, реалізовану засобами бібліотеки PennyLane.

Метою експерименту було не досягнення максимально можливої точності на складному наборі даних, а перевірка того, чи може варіаційна квантова схема бути інтегрована в класифікаційну голову малого Vision Transformer та забезпечити конкурентну якість класифікації за близької кількості навчуваних параметрів. Для первинної перевірки було використано двокласову постановку задачі, що дозволяє зменшити обчислювальну складність і виконати експеримент локально на звичайному персональному комп'ютері.

Обидві моделі навчалися в однакових умовах. Кількість епох становила 5, розмір батчу – 32, швидкість навчання – 0,001. Для оптимізації параметрів використовувався оптимізатор Adam, а як функція втрат застосовувалась CrossEntropyLoss. Якість моделей оцінювалася за accuracy, precision, recall, F1-score, значенням функції втрат, матрицею помилок, кількістю навчуваних параметрів і часом навчання.

5.1.2 Динаміка навчання моделей

Під час навчання обох моделей спостерігалось поступове зменшення значення функції втрат. Це свідчить про коректну роботу навчального циклу та здатність як класичної, так і гібридної моделі адаптувати свої параметри до навчальних даних. Для моделі TinyViT з лінійною класифікаційною головою значення train loss зменшилося з 0,4747 на першій епосі до 0,1219 на п'ятій епосі. Для моделі TinyViT з варіаційною квантовою головою train loss зменшилося з 0,4312 до 0,1575.

На тестовій вибірці класична модель зменшила test loss з 0,3167 до 0,0795. Гібридна квантово-класична модель зменшила test loss з 0,3276 до 0,1167. Отже, обидві моделі демонструють стабільну динаміку навчання, однак у межах цього запуску класична модель має нижче фінальне значення функції втрат. Водночас нижче значення loss не означає автоматично вищу якість класифікації за всіма метриками, тому додатково необхідно аналізувати accuracy, precision, recall і F1-score.

Аналіз динаміки train loss і test loss дозволяє оцінити не лише кінцевий результат, а й сам процес збіжності моделей. У цьому експерименті обидві криві втрат мають спадний характер, що свідчить про відсутність різких нестабільностей під час навчання. Важливо також, що test loss зменшується разом із train loss, тобто моделі не демонструють явних ознак перенавчання в межах п'яти епох. Це дає підстави використовувати отримані результати для подальшого порівняння класифікаційних метрик.

Динаміка зміни функції втрат для обох моделей наведена на рисунку 5.1. З графіка видно, що в обох випадках значення функції втрат поступово зменшується протягом усіх п'яти епох. Це свідчить про те, що обидві архітектури навчаються коректно, а варіаційна квантова схема може бути використана як частина навчуваної класифікаційної голови.

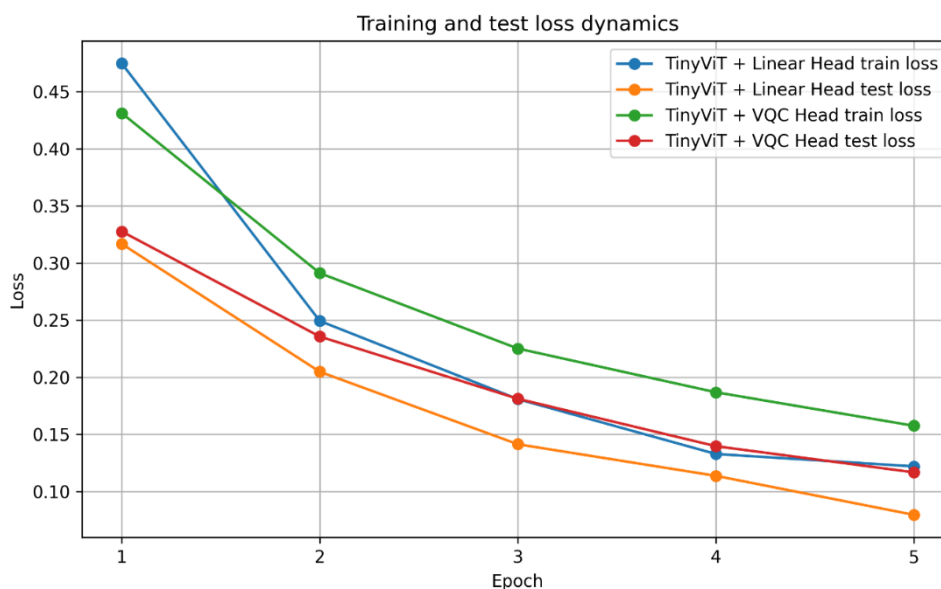


Рисунок 5.1 – Динаміка функції втрат на навчальній і тестовій вибірках

Отримана динаміка функції втрат підтверджує, що обидві моделі здатні навчатися в заданих умовах експерименту. При цьому нижче фінальне значення loss у класичної моделі не є достатньою підставою для остаточного висновку щодо якості класифікації. Тому наступним кроком є аналіз основних метрик якості.

5.1.3 Порівняння якості класифікації

За результатами первинного експерименту класична модель TinyViT з лінійною класифікаційною головою досягла accuracy 0,9740 та F1-score 0,9775. Значення precision для цієї моделі становило 0,9659, а recall – 0,9895. Гібридна модель TinyViT з варіаційною квантовою головою досягла accuracy 0,9820 та F1-score 0,9843. Precision для гібридної моделі становив 0,9792, а recall залишився на рівні 0,9895.

Отримані результати показують, що в межах поточного запуску гібридна модель продемонструвала дещо вищу якість класифікації за accuracy, precision і F1-score. Різниця за accuracy становить 0,0080, а різниця

за F1-score – приблизно 0,0068. Значення recall в обох моделях однакове, тобто обидві моделі однаково добре знаходили приклади позитивного класу. Водночас гібридна модель мала вище значення precision, що означає меншу кількість помилкових позитивних передбачень.

Порівняння основних метрик якості наведено на рисунку 5.2. Графік показує, що гібридна модель має невелику перевагу за трьома з чотирьох метрик, тоді як recall є однаковим для обох моделей. Проте цей результат слід розглядати як попередній, оскільки один запуск не є достатньою підставою для твердження про загальну перевагу квантової моделі над класичною.

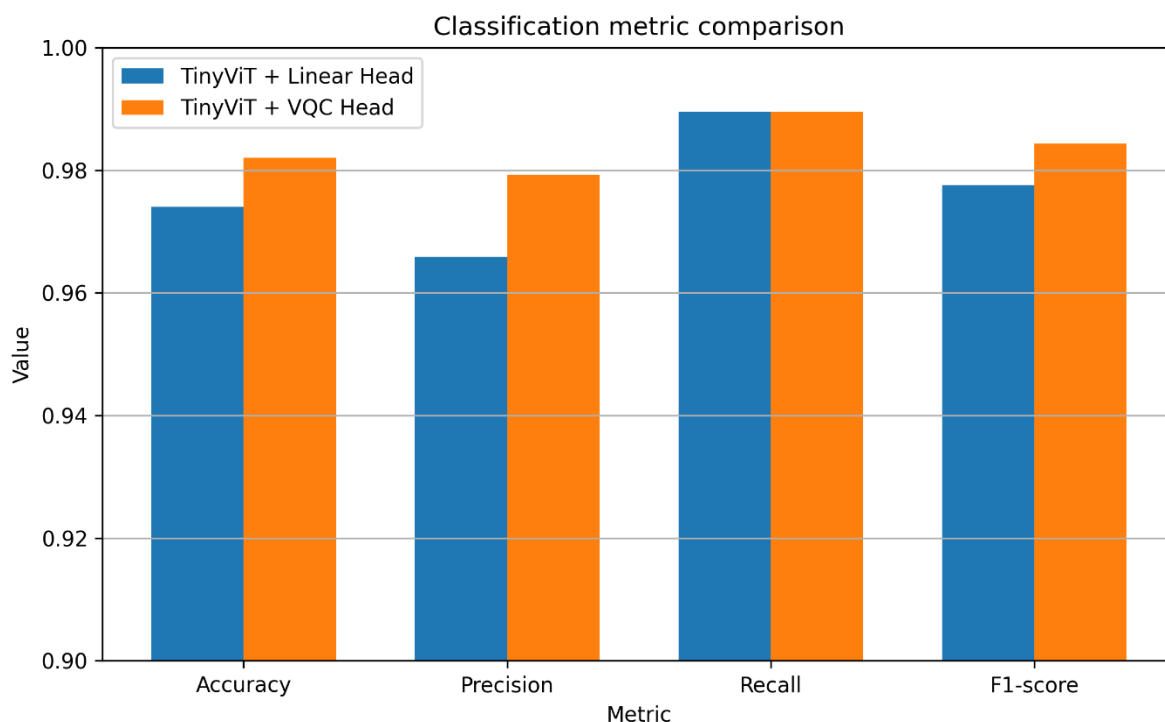


Рисунок 5.2 – Порівняння основних метрик якості класифікації

Наведене порівняння показує попередню перевагу гібридної моделі в межах одного запуску. Однак різниця між моделями є невеликою, тому її потрібно додатково перевірити через аналіз помилок класифікації та

повторні експерименти. Для цього далі розглядаються матриці помилок обох моделей.

5.1.4 Аналіз матриць помилок

Для детальнішого аналізу результатів було побудовано матриці помилок для обох моделей. Матриця помилок дозволяє побачити не лише загальну точність класифікації, а й характер помилок для кожного класу. У двокласовій задачі вона показує, скільки прикладів кожного класу було класифіковано правильно та скільки було віднесено до іншого класу.

Для класичної моделі TinyViT з лінійною головою було правильно класифіковано 204 приклади класу 0 та 283 приклади класу 1. При цьому 10 прикладів класу 0 були помилково віднесені до класу 1, а 3 приклади класу 1 були помилково віднесені до класу 0. Отже, основна частина помилок класичної моделі пов'язана саме з помилковим віднесенням прикладів класу 0 до класу 1.

Матрицю помилок для класичної моделі TinyViT з лінійною класифікаційною головою наведено на рисунку 5.3.

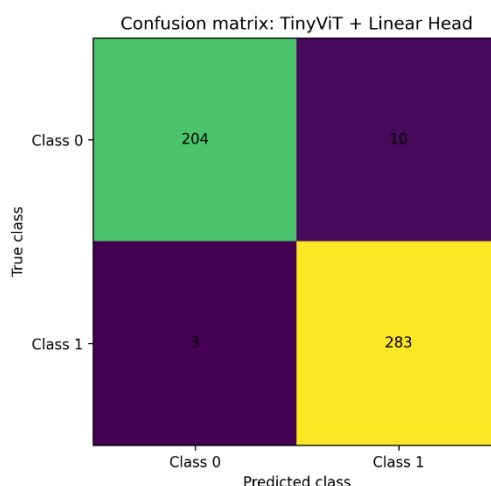


Рисунок 5.3 – Матриця помилок для TinyViT з лінійною класифікаційною ГОЛОВОЮ

Для гібридної моделі TinyViT з варіаційною квантовою головою було правильно класифіковано 208 прикладів класу 0 та 283 приклади класу 1. Кількість помилок для класу 0 зменшилася з 10 до 6, тоді як кількість помилок для класу 1 залишилася незмінною і становила 3. Саме це зменшення кількості помилкових передбачень для класу 0 пояснює покращення accuracy та precision у гібридної моделі в межах поточного запуску.

Матрицю помилок для гібридної моделі TinyViT з варіаційною квантовою головою наведено на рисунку 5.4.

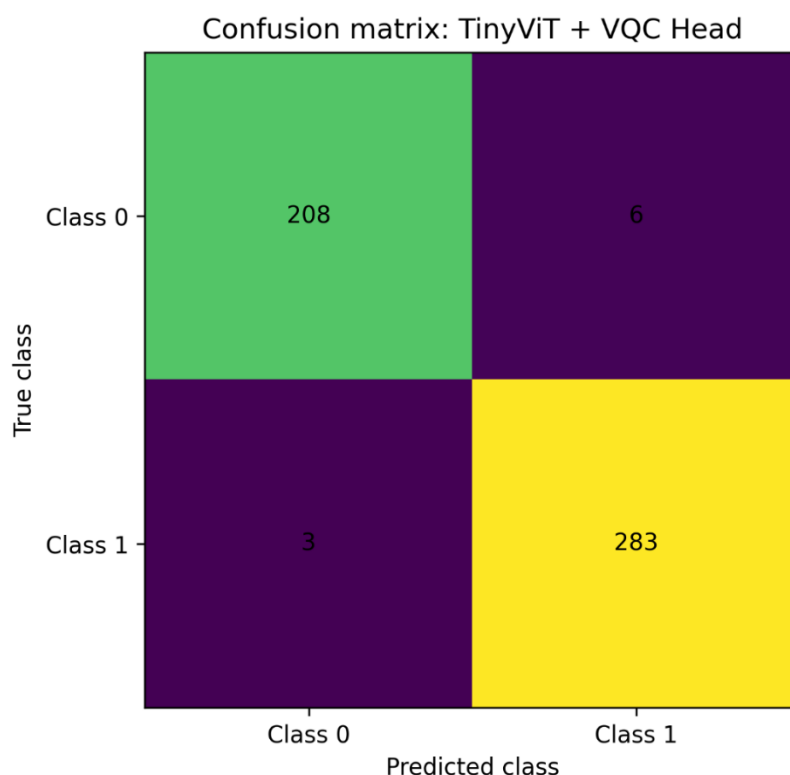


Рисунок 5.4 – Матриця помилок для TinyViT з варіаційною квантовою ГОЛОВОЮ

Порівняння матриць помилок показує, що обидві моделі добре розв'язують двокласову задачу, однак гібридна модель у цьому запуску допустила меншу кількість помилок для одного з класів. При цьому різниця

між моделями становить лише декілька прикладів, тому її не можна вважати остаточною доказом переваги гібридної моделі.

Таким чином, аналіз матриць помилок підтверджує результати, отримані за основними метриками якості. Гібридна модель показала дещо кращий результат у межах одного запуску, але ця перевага має перевірятися на кількох random seed. Тому подальший аналіз у роботі спрямований на оцінювання стабільності моделей за повторних експериментів.

5.1.5 Аналіз кількості параметрів і часу навчання

Оскільки тема роботи пов'язана з малопараметричною класифікацією зображень, важливо оцінювати не лише якість класифікації, а й кількість навчуваних параметрів. Класична модель TinyViT з лінійною головою має 3384 навчувані параметри. Гібридна модель TinyViT з варіаційною квантовою головою має 3390 навчуваних параметрів. Різниця становить лише 6 параметрів, що відповідає параметрам варіаційної квантової схеми з двома кубітами та одним шаром.

Таким чином, моделі мають майже однаковий параметричний бюджет. Це робить порівняння достатньо коректним, оскільки покращення або погіршення результату не можна пояснити значним збільшенням кількості параметрів. У межах проведеного запуску гібридна модель досягла вищих значень accuracy та F1-score за збільшення кількості навчуваних параметрів лише на 6.

Окремий аналіз кількості параметрів є важливим для підтвердження малопараметричного характеру обох моделей. Якщо б гібридна модель мала значно більшу кількість параметрів, її порівняння з класичною моделлю було б менш коректним. У цьому випадку різниця між моделями є мінімальною, тому подальше порівняння можна зосередити саме на впливі типу класифікаційної голови.

Порівняння кількості навчуваних параметрів класичної та гібридної моделей наведено на рисунку 5.5.

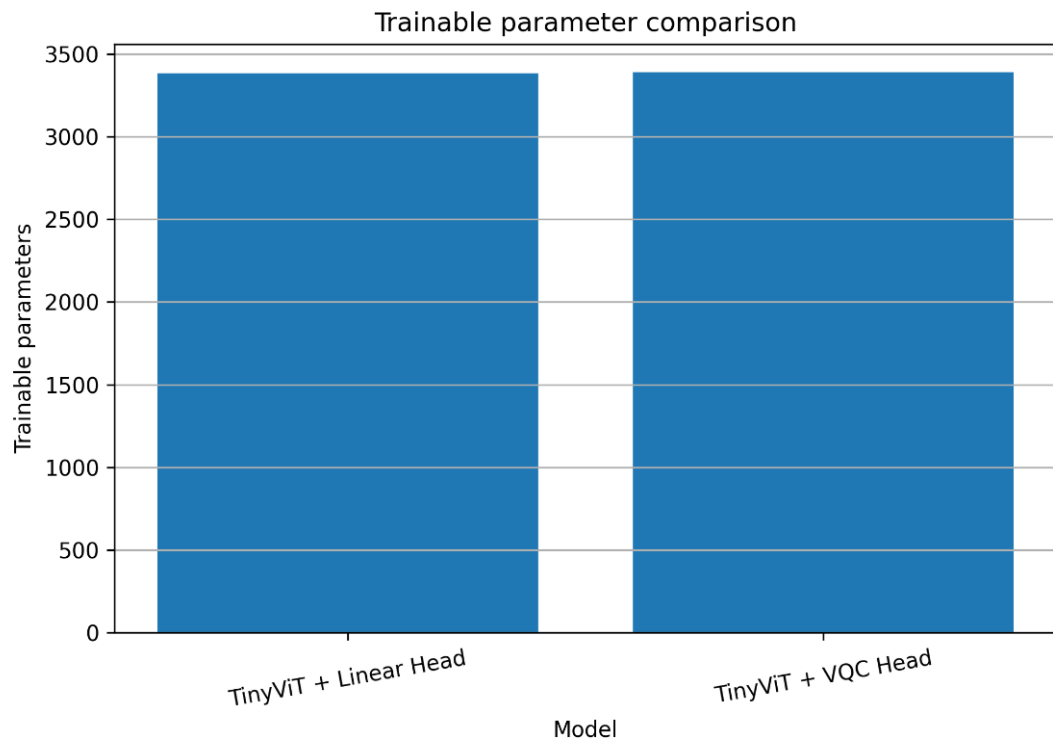


Рисунок 5.5 – Порівняння кількості навчуваних параметрів моделей

Водночас час навчання гібридної моделі був більшим. Класична модель навчалася приблизно 3,05 секунди, тоді як гібридна модель потребувала приблизно 4,85 секунди. Це очікуваний результат, оскільки квантова схема моделюється на класичному процесорі, а симуляція квантових обчислень створює додаткові обчислювальні витрати.

Аналіз часу навчання дозволяє оцінити практичну сторону використання квантової голови. Незважаючи на майже однакову кількість навчуваних параметрів, гібридна модель виконує додаткові операції, пов'язані з моделюванням варіаційної квантової схеми. Тому збільшення часу навчання є важливим обмеженням, яке необхідно враховувати під час оцінювання доцільності такого підходу.

Порівняння часу навчання класичної та гібридної моделей наведено на рисунку 5.6.

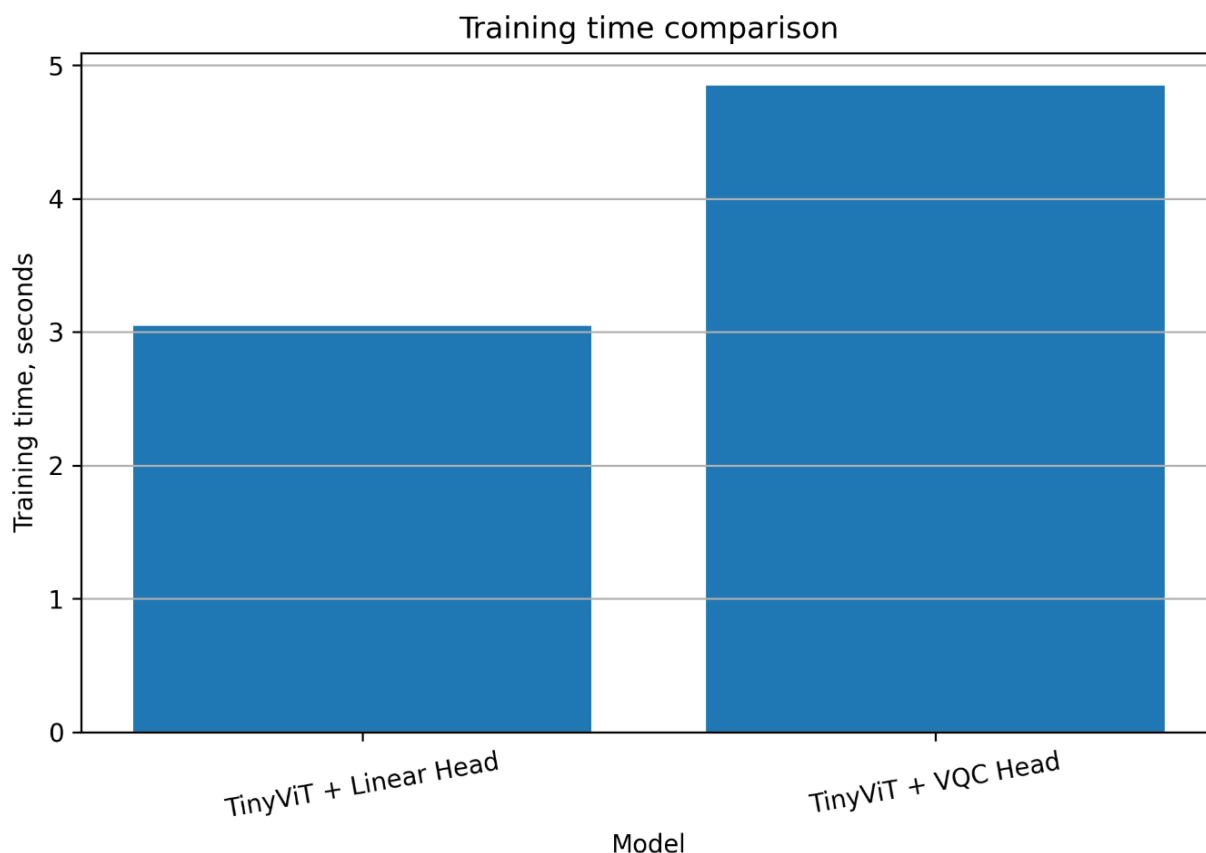


Рисунок 5.6 – Порівняння часу навчання класичної та гібридної моделей

Отже, з погляду кількості параметрів моделі є майже однаковими, але з погляду часу навчання класична модель є швидшою. Це важливе практичне обмеження гібридного підходу: навіть за малої кількості кубітів симуляція квантової схеми на класичному комп'ютері збільшує час виконання експерименту. Водночас отримана різниця в часі не скасовує працездатності запропонованої архітектури, а лише показує її обчислювальну вартість. Тому під час оцінювання гібридної моделі необхідно враховувати не тільки метрики якості, а й співвідношення між точністю, кількістю параметрів і часом навчання. Це дає змогу перейти до

узагальнення результатів первинного експерименту та сформулювати проміжні висновки щодо доцільності використання VQC Head.

5.1.6 Узагальнення результатів первинного експерименту

Проведений первинний експеримент підтвердив працездатність запропонованої архітектури QuantumTinyViT. Гібридна квантово-класична модель була успішно навчена в одному циклі разом із класичними параметрами, а варіаційна квантова схема коректно виконувала роль частини класифікаційної голови. Це підтверджує технічну можливість інтеграції PennyLane-схеми в PyTorch-модель для задачі класифікації зображень.

У межах первинного запуску гібридна модель показала дещо вищі значення accuracy, precision та F1-score порівняно з класичною моделлю. Однак цей результат не можна розглядати як остаточне підтвердження переваги квантової голови. Різниця між моделями є невеликою, а експеримент потребує повторення з різними random seed та, за потреби, на складніших наборах даних. Тому на цьому етапі коректним висновком є не твердження про перевагу квантової моделі, а підтвердження технічної можливості інтеграції варіаційної квантової схеми в малопараметричну класифікаційну голову Vision Transformer.

Основним позитивним результатом є те, що гібридна модель продемонструвала конкурентну якість класифікації за майже однакової кількості навчуваних параметрів. Основним недоліком є більший час навчання, що пов'язано з симуляцією квантової схеми на класичному процесорі. Отримані результати створюють основу для подальшого дослідження впливу кількості кубітів, глибини квантової схеми, випадкової ініціалізації та складності набору даних на якість малопараметричної класифікації зображень.

5.2 Перевірка стійкості результатів за різних random seed

5.2.1 Умови повторного експерименту

Для перевірки стійкості отриманих результатів було проведено серію повторних запусків експерименту на наборі MNIST. На відміну від первинного експерименту, який відображав результат одного запуску, у цьому підрозділі аналізуються усереднені результати декількох запусків із різними значеннями random seed. Такий підхід дозволяє оцінити, наскільки результати моделей залежать від випадкової ініціалізації параметрів, порядку перемішування навчальних даних та інших стохастичних факторів.

Експеримент було виконано для п'яти значень random seed: 42, 123, 2024, 7 та 99. У кожному запуску навчалися дві моделі: TinyViT з лінійною класифікаційною головою та TinyViT з варіаційною квантовою головою. Архітектура моделей, кількість епох, розмір батчу, швидкість навчання, обсяг навчальної вибірки та обсяг тестової вибірки залишалися однаковими для всіх запусків.

Для кожної моделі було обчислено середні значення accuracy, precision, recall, F1-score, test loss і часу навчання, а також стандартне відхилення цих показників. Використання середнього значення разом зі стандартним відхиленням дає змогу оцінити не лише загальний рівень якості моделі, а й стабільність її роботи за різних початкових умов.

5.2.2 Аналіз середніх значень метрик на MNIST

За результатами повторного експерименту на MNIST класична модель TinyViT з лінійною класифікаційною головою досягла середнього значення accuracy 0,9844 зі стандартним відхиленням 0,0057. Середнє значення F1-score цієї моделі становило 0,9865 зі стандартним відхиленням 0,0049. Гібридна модель TinyViT з варіаційною квантовою головою досягла

середнього значення accuracy 0,9740 зі стандартним відхиленням 0,0113, а середнє значення F1-score становило 0,9777 зі стандартним відхиленням 0,0094.

Отримані результати показують, що за серією запусків на MNIST класична модель у середньому продемонструвала вищі значення accuracy та F1-score. Крім того, стандартне відхилення для класичної моделі було меншим, що свідчить про більш стабільну поведінку за різних random seed. Для гібридної моделі значення стандартного відхилення є більшим, отже її результат сильніше залежав від випадкової ініціалізації та особливостей конкретного запуску.

Середнє значення precision для класичної моделі становило 0,9815, тоді як для гібридної моделі – 0,9651. Значення recall для обох моделей були близькими: 0,9916 для класичної моделі та 0,9909 для гібридної. Це означає, що обидві моделі добре знаходили приклади позитивного класу, однак класична модель у середньому мала меншу кількість помилкових позитивних передбачень.

Порівняння середніх метрик якості для MNIST за різних random seed наведено на рисунку 5.7.

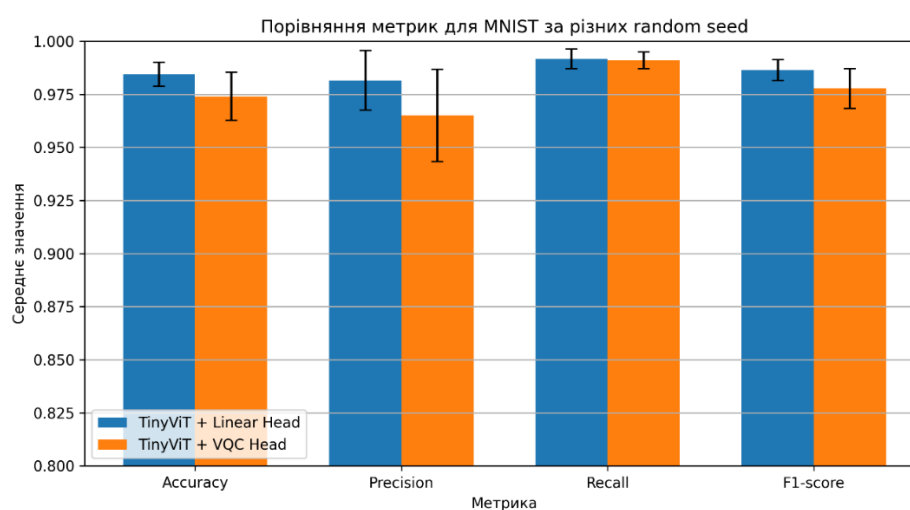


Рисунок 5.7 – Порівняння середніх метрик якості для MNIST за різних random seed

Графік підтверджує, що за серією запусків на MNIST класична модель у середньому має вищі та стабільніші значення основних метрик. Це показує, що результат первинного запуску не можна узагальнювати без повторної перевірки. Далі доцільно проаналізувати, як така різниця між моделями проявляється за часом навчання.

5.2.3 Аналіз часу навчання на MNIST

Окрім якості класифікації, важливим показником є час навчання. За результатами серії запусків середній час навчання класичної моделі TinyViT з лінійною головою становив приблизно 3,01 с. Для гібридної моделі TinyViT з варіаційною квантовою головою середній час навчання становив приблизно 4,78 с.

Таким чином, гібридна модель навчалася повільніше, хоча кількість навчуваних параметрів у двох моделей була майже однаковою. Це пояснюється тим, що варіаційна квантова схема моделюється на класичному процесорі. Навіть за малої кількості кубітів симуляція квантового шару створює додаткові обчислювальні витрати.

Порівняння середнього часу навчання для MNIST наведено на рисунку 5.8.

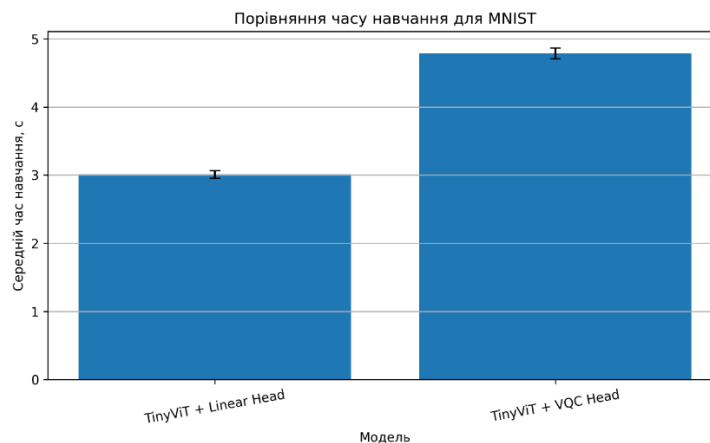


Рисунок 5.8 – Порівняння середнього часу навчання для MNIST

Наведене порівняння показує, що гібридна модель потребує більшого часу навчання за майже однакового параметричного бюджету. Отже, додавання варіаційної квантової схеми створює додаткові обчислювальні витрати. Це враховується під час узагальнення результатів повторного експерименту.

5.2.4 Узагальнення результатів повторного експерименту

Повторний експеримент за різних random seed показав, що результати одного запуску не є достатніми для остаточного висновку щодо переваги тієї чи іншої класифікаційної голови. У серії запусків на MNIST класична модель TinyViT з лінійною головою показала вищі середні значення accuracy та F1-score, а також менший test loss і менший час навчання. Гібридна модель при цьому залишилася конкурентною за якістю, але не продемонструвала сталої переваги над класичною моделлю.

Отже, за результатами повторного експерименту можна зробити висновок, що варіаційна квантова схема може бути технічно інтегрована в класифікаційну голову малого Vision Transformer, однак її використання не гарантує покращення якості класифікації. У межах цього експерименту VQC Head забезпечує порівнювану якість за близької кількості параметрів, але має більші обчислювальні витрати під час симуляції.

5.3 Додатковий експеримент на наборі Fashion-MNIST

5.3.1 Умови додаткового експерименту

Для перевірки запропонованого підходу на складнішому та більш варіативному наборі зображень було проведено додатковий експеримент на Fashion-MNIST. На відміну від MNIST, який містить зображення рукописних цифр, Fashion-MNIST містить зображення предметів одягу [17].

Такі зображення мають складніші візуальні ознаки, тому цей набір даних є корисним для додаткової перевірки узагальнювальної здатності моделей.

Важливою перевагою Fashion-MNIST у межах цієї роботи є те, що він має такий самий формат зображень, як MNIST: одноканальні зображення розміром 28×28 пікселів. Завдяки цьому додатковий експеримент можна провести без принципової зміни архітектури TinyViT. Як і в попередньому повторному експерименті, було використано двокласову постановку задачі та п'ять значень random seed: 42, 123, 2024, 7 та 99.

Умови навчання для Fashion-MNIST були такими самими, як і для MNIST: кількість епох становила 5, розмір батчу – 32, швидкість навчання – 0,001. Для порівняння використовувалися ті самі дві моделі: TinyViT з лінійною класифікаційною головою та TinyViT з варіаційною квантовою головою.

5.3.2 Аналіз середніх значень метрик на Fashion-MNIST

За результатами експерименту на Fashion-MNIST класична модель TinyViT з лінійною головою досягла середнього значення accuracy 0,9440 зі стандартним відхиленням 0,0032. Середнє значення F1-score цієї моделі становило 0,9435 зі стандартним відхиленням 0,0034. Гібридна модель TinyViT з варіаційною квантовою головою досягла середнього значення accuracy 0,9448 зі стандартним відхиленням 0,0069, а середнє значення F1-score становило 0,9440 зі стандартним відхиленням 0,0070.

Отримані значення показують, що на Fashion-MNIST обидві моделі продемонстрували майже однакову якість класифікації. Гібридна модель мала незначно вищі середні accuracy та F1-score, однак різниця є дуже малою і перебуває в межах варіативності результатів між різними random seed. Тому цей результат не слід інтерпретувати як підтвердження переваги квантової голови.

Середнє значення precision для класичної моделі становило 0,9598, а для гібридної – 0,9656. Водночас recall для класичної моделі становив 0,9286, а для гібридної – 0,9238. Це означає, що гібридна модель у середньому робила трохи менше помилкових позитивних передбачень, але водночас трохи гірше знаходила всі приклади позитивного класу. У підсумку F1-score моделей виявився майже однаковим.

Порівняння середніх метрик якості для Fashion-MNIST за різних random seed наведено на рисунку 5.9.

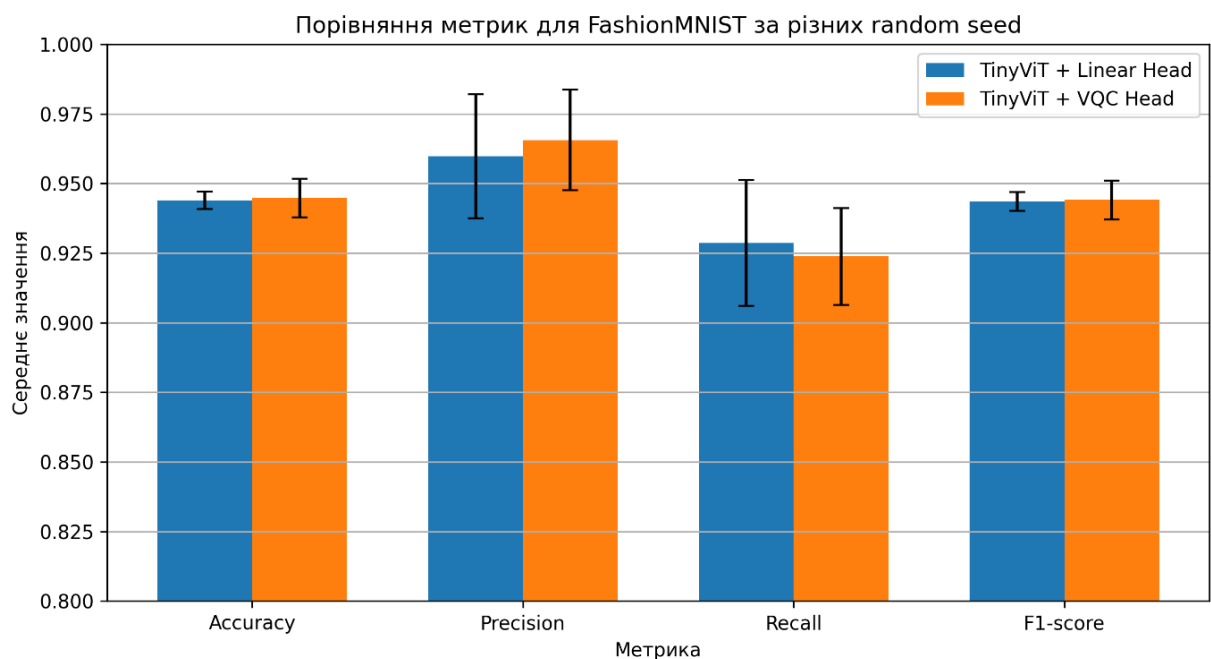


Рисунок 5.9 – Порівняння середніх метрик якості для Fashion-MNIST за різних random seed

Графік показує, що на Fashion-MNIST різниця між класичною та гібридною моделями є мінімальною. Це свідчить про конкурентну якість VQC Head на складнішому наборі даних, але не підтверджує його сталої переваги. Для повнішої оцінки необхідно також порівняти час навчання моделей.

5.3.3 Аналіз часу навчання на Fashion-MNIST

На Fashion-MNIST збереглася та сама тенденція щодо часу навчання, що й у повторному експерименті на MNIST. Класична модель навчалася в середньому приблизно 3,00 с, тоді як гібридна модель потребувала приблизно 4,78 с. Це підтверджує, що основним практичним недоліком використання варіаційної квантової голови є збільшення часу навчання через симуляцію квантової схеми.

Оскільки архітектура гібридної моделі містить квантовий шар, кожен прямий прохід через модель потребує додаткового обчислення квантової схеми. У цьому експерименті квантова частина виконувалася на симуляторі, тому збільшення часу навчання є очікуваним результатом. Водночас різниця в часі не пов'язана зі значним збільшенням кількості параметрів, а насамперед із вищою обчислювальною складністю моделювання VQC.

Порівняння середнього часу навчання для обох моделей на Fashion-MNIST наведено на рисунку 5.10.

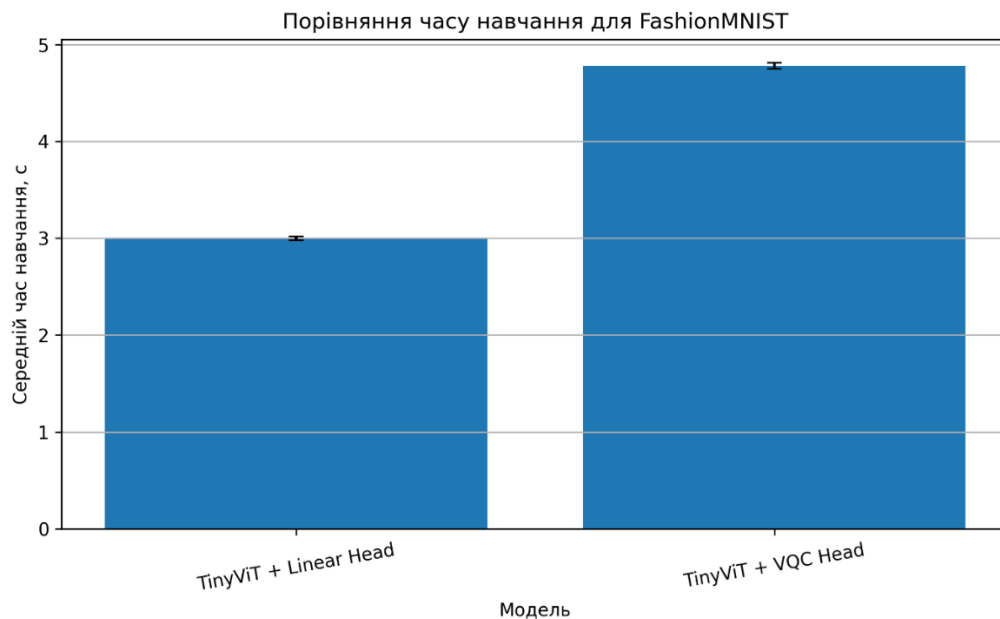


Рисунок 5.10 – Порівняння середнього часу навчання для Fashion-MNIST

Отримані результати підтверджують, що збільшення часу навчання гібридної моделі зберігається не лише на MNIST, а й на Fashion-MNIST. Це вказує на системний характер обчислювальних витрат, пов'язаних із симуляцією квантового шару. Далі результати двох наборів даних порівнюються між собою за узагальненою метрикою F1-score.

5.3.4 Порівняння результатів між MNIST та Fashion-MNIST

Порівняння результатів між MNIST і Fashion-MNIST показує, що обидві моделі досягають вищих значень F1-score на MNIST. Це очікувано, оскільки двокласова задача на рукописних цифрах є простішою за класифікацію зображень предметів одягу. На Fashion-MNIST середній F1-score зменшився приблизно до 0,943–0,944 для обох моделей.

На MNIST класична модель мала вищий середній F1-score порівняно з гібридною моделлю. На Fashion-MNIST різниця між моделями практично зникла, і обидві моделі показали майже однакову якість. Це свідчить про те, що ефективність квантової голови залежить від конкретного набору даних, випадкової ініціалізації та особливостей навчання.

Для наочнішого зіставлення результатів доцільно окремо порівняти середні значення F1-score на двох наборах даних. Саме ця метрика є зручною для узагальнення якості двокласової класифікації, оскільки враховує одночасно precision і recall. Таке порівняння дозволяє побачити не лише різницю між класичною та гібридною моделями, а й вплив складності самого набору даних на кінцеву якість класифікації.

Таким чином, результати на двох наборах даних показують, що поведінка моделей не є повністю однаковою для різних типів зображень. Це підкреслює необхідність перевіряти гібридну архітектуру не лише на одному простому наборі даних, а й на більш варіативних задачах класифікації.

Порівняння середнього F1-score між наборами MNIST та Fashion-MNIST наведено на рисунку 5.11.

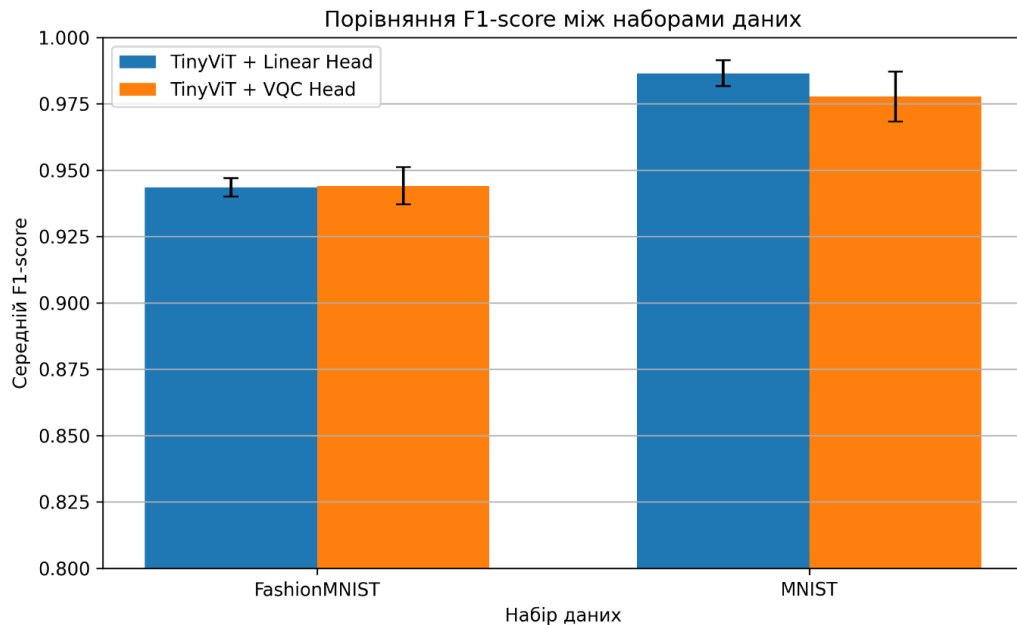


Рисунок 5.11 – Порівняння середнього F1-score між наборами даних

Наведене порівняння показує, що обидві моделі досягають вищої якості на MNIST, тоді як на Fashion-MNIST значення F1-score зменшуються. Це узгоджується з більшою складністю зображень предметів одягу порівняно з рукописними цифрами. Отримані результати дають змогу перейти до загального узагальнення додаткового експерименту.

5.3.5 Узагальнення результатів додаткового експерименту

Додатковий експеримент на Fashion-MNIST показав, що запропонована гібридна квантово-класична модель зберігає працездатність на більш варіативному наборі зображень. Якість класифікації гібридної моделі виявилася близькою до якості класичної моделі, що підтверджує

можливість використання VQC Head не лише на простому наборі MNIST, а й на складнішому наборі зображень.

Водночас результати не підтверджують сталої переваги квантової голови. На MNIST класична модель показала кращі середні результати, а на Fashion-MNIST обидві моделі виявилися практично рівними. Отже, варіаційна квантова схема в цій роботі розглядається не як гарантовано краща заміна класичної голови, а як альтернативний малопараметричний компонент, який може забезпечувати конкурентну якість класифікації за невеликої кількості додаткових параметрів.

Основним обмеженням гібридної моделі залишається час навчання. В обох наборах даних модель з VQC Head навчалася повільніше за класичну модель. Це пов'язано з тим, що квантова схема симулюється на класичному комп'ютері. Тому практична доцільність такого підходу потребує подальшого дослідження, зокрема з використанням інших конфігурацій квантової схеми, більшої кількості класів і додаткових класичних baseline-моделей.

ВИСНОВКИ

У кваліфікаційній роботі було розроблено та досліджено гібридну квантово-класичну модель для малопараметричної класифікації зображень. Поставлене завдання виконано повністю: проведено аналіз предметної галузі, розглянуто архітектуру Vision Transformer, досліджено принципи побудови варіаційних квантових схем, реалізовано класичну та гібридну моделі мовою Python, виконано експериментальне порівняння моделей і проаналізовано отримані результати.

У процесі виконання роботи було побудовано малопараметричну архітектуру TinyViT, яка використовується як класична основа для формування ознак зображення. Модель виконує розбиття зображення на патчі, формування embedding-представлень, додавання CLS-токена та позиційного кодування, після чого transformer-encoder формує компактний вектор ознак. На основі цієї архітектури реалізовано дві моделі: класичну TinyViT з лінійною класифікаційною головою та гібридну QuantumTinyViT, у якій після проєкції ознак використовується варіаційна квантова схема.

Квантовий компонент гібридної моделі реалізовано у вигляді VQC Head. Для кодування класичних ознак у квантовий стан використано кутове кодування, а для варіаційного перетворення – параметризовані квантові операції та операції заплутування. На виході квантового шару виконувалося вимірювання очікуваних значень операторів Паулі-Z, після чого отримані числові значення передавалися до фінального класичного класифікатора. Таким чином, у роботі підтверджено технічну можливість інтеграції варіаційної квантової схеми в класифікаційну голову малого Vision Transformer.

Первинний експеримент показав, що обидві моделі здатні успішно навчатися та виконувати двокласову класифікацію зображень. У межах одного запуску гібридна модель показала дещо вищі значення асигуру та F1-score порівняно з класичною моделлю. Водночас такий результат не було

розглянуто як остаточне підтвердження переваги квантової голови, оскільки результат одного запуску може залежати від випадкової ініціалізації параметрів, порядку перемішування даних та інших стохастичних факторів.

Для перевірки стійкості результатів було проведено повторний експеримент на MNIST із п'ятьма різними значеннями random seed. За результатами цього експерименту класична модель TinyViT з лінійною головою досягла середнього значення accuracy 0,9844 та середнього F1-score 0,9865. Гібридна модель TinyViT з варіаційною квантовою головою досягла середнього значення accuracy 0,9740 та середнього F1-score 0,9777. Отже, у серії запусків на MNIST класична модель продемонструвала вищі середні показники якості та меншу варіативність результатів.

Для додаткової перевірки запропонованого підходу було використано набір Fashion-MNIST, який є більш варіативним порівняно з MNIST, але має такий самий формат зображень. На Fashion-MNIST класична модель досягла середнього значення accuracy 0,9440 та F1-score 0,9435, а гібридна модель – accuracy 0,9448 та F1-score 0,9440. Отримані результати показали, що на складнішому наборі даних обидві моделі мають практично однакову якість класифікації. Це свідчить про конкурентоспроможність VQC Head, але не дає підстав стверджувати про його стабільну перевагу над класичною головою.

Важливим кількісним результатом є те, що обидві моделі мають майже однакову кількість навчуваних параметрів. Класична модель містить 3384 навчувані параметри, а гібридна – 3390. Різниця становить лише 6 параметрів, що відповідає параметрам варіаційної квантової схеми. Це дозволяє вважати порівняння моделей коректним з погляду малопараметричної постановки задачі. Водночас за часом навчання класична модель виявилася ефективнішою: у повторних експериментах вона навчалася приблизно за 3 с, тоді як гібридна модель потребувала приблизно 4,78 с. Це пояснюється тим, що квантова схема моделювалася на класичному процесорі, що створювало додаткові обчислювальні витрати.

Порівняно з відомими світовими підходами до використання Vision Transformer і гібридних квантово-класичних моделей, виконана розробка має навчально-дослідницький характер і орієнтована не на побудову великої високопродуктивної моделі, а на перевірку малопараметричної архітектурної ідеї. На відміну від великих попередньо навчених моделей комп'ютерного зору, запропонований підхід може бути навчений локально на звичайному персональному комп'ютері. Порівняно з типовими класичними класифікаторами, розроблена модель демонструє можливість включення квантового шару до стандартного pipeline глибинного навчання. У вітчизняному навчальному контексті така робота може бути корисною як приклад практичного дослідження на межі комп'ютерного зору, трансформерних архітектур і квантового машинного навчання.

Виконана робота пов'язана з актуальними напрямками науково-дослідних робіт у сфері штучного інтелекту, комп'ютерного зору, інтелектуального аналізу даних і квантових обчислень. Отримані результати можуть бути використані як основа для подальших досліджень на кафедрі або в навчальних проєктах, пов'язаних із розробкою малопараметричних нейромережових моделей, аналізом трансформерних архітектур і дослідженням гібридних квантово-класичних підходів. Окремі положення роботи можуть бути використані під час підготовки лабораторних або практичних завдань з дисциплін, пов'язаних із машинним навчанням, комп'ютерним зором і квантовими обчисленнями.

У межах роботи отримано прикладний науково-технічний результат, який полягає у побудові та експериментальній перевірці гібридної квантово-класичної класифікаційної голови для малого Vision Transformer. Результати роботи не оформлювалися у вигляді винаходів або окремих наукових статей, однак можуть бути використані як основа для подальшої публікації або розширеного дослідження. Подальша робота в цьому напрямі може бути пов'язана з перевіркою більшої кількості random seed, додаванням класичної MLP-головки як додаткового baseline, використанням

більшої кількості класів, тестуванням на складніших наборах даних, зміною кількості кубітів і глибини квантової схеми, а також аналізом впливу різних способів кодування ознак у квантовий стан.

Практична цінність роботи полягає у створенні програмного прототипу, який дозволяє порівнювати класичні та гібридні квантово-класичні моделі в однакових умовах. Матеріали роботи можуть бути використані за призначенням для подальших експериментів із малопараметричною класифікацією зображень, а також у навчальному процесі університету як приклад реалізації повного дослідницького циклу: від аналізу предметної галузі та математичного опису моделі до програмної реалізації, проведення експерименту, побудови графіків і формування висновків.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

- 1) Attention is all you need / A. Vaswani et al. *Advances in neural information processing systems*. 2017. URL: <https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html> (date of access: 08.06.2026).
- 2) An image is worth 16x16 words: transformers for image recognition at scale / A. Dosovitskiy et al. *International conference on learning representations*. 2021. URL: <https://openreview.net/forum?id=YicbFdNTTy> (date of access: 08.06.2026).
- 3) Training data-efficient image transformers & distillation through attention / H. Touvron et al. *Proceedings of the 38th international conference on machine learning*. PMLR. 2021. Vol. 139. P. 10347–10357. URL: <https://proceedings.mlr.press/v139/touvron21a.html> (date of access: 08.06.2026).
- 4) Preskill J. Quantum computing in the NISQ era and beyond. *Quantum*. 2018. Vol. 2. P. 79. URL: <https://quantum-journal.org/papers/q-2018-08-06-79/> (date of access: 08.06.2026).
- 5) Data re-uploading for a universal quantum classifier / A. Pérez-Salinas et al. *Quantum*. 2020. Vol. 4. P. 226. URL: <https://quantum-journal.org/papers/q-2020-02-06-226/> (date of access: 06.06.2026).
- 6) Transfer learning in hybrid classical-quantum neural networks / A. Mari et al. *Quantum*. 2020. Vol. 4. P. 340. URL: <https://quantum-journal.org/papers/q-2020-10-09-340/> (date of access: 08.06.2026).
- 7) qml.qnn.TorchLayer documentation. PennyLane documentation. URL: <https://docs.pennylane.ai/en/stable/code/api/pennylane.qnn.TorchLayer.html> (date of access: 10.06.2026).
- 8) qml.AngleEmbedding documentation. PennyLane documentation. URL: <https://docs.pennylane.ai/en/stable/code/api/pennylane.AngleEmbedding.html> (date of access: 10.06.2026).

- 9) qml.StronglyEntanglingLayers documentation. *PennyLane documentation*. URL: <https://docs.pennylane.ai/en/stable/code/api/pennylane.StronglyEntanglingLayers.html> (date of access: 10.06.2026).
- 10) PyTorch interface. *PennyLane documentation*. URL: <https://docs.pennylane.ai/en/stable/introduction/interfaces/torch.html> (date of access: 10.06.2026).
- 11) TransformerEncoderLayer documentation. *PyTorch documentation*. URL: <https://docs.pytorch.org/docs/2.12/generated/torch.nn.TransformerEncoderLayer.html> (date of access: 10.06.2026).
- 12) Conv2d documentation. *PyTorch documentation*. URL: <https://docs.pytorch.org/docs/2.12/generated/torch.nn.Conv2d.html> (date of access: 10.06.2026).
- 13) Linear documentation. *PyTorch documentation*. URL: <https://docs.pytorch.org/docs/2.12/generated/torch.nn.Linear.html> (date of access: 10.06.2026).
- 14) CrossEntropyLoss documentation. *PyTorch documentation*. URL: <https://docs.pytorch.org/docs/2.12/generated/torch.nn.CrossEntropyLoss.html> (date of access: 10.06.2026).
- 15) Adam documentation. *PyTorch documentation*. URL: <https://docs.pytorch.org/docs/2.12/generated/torch.optim.Adam.html> (date of access: 10.06.2026).
- 16) Datasets documentation. *TorchVision documentation*. URL: <https://docs.pytorch.org/vision/stable/datasets.html> (date of access: 10.06.2026).
- 17) Fashion-MNIST: a MNIST-like fashion product database. *Zalando Research*. URL: <https://github.com/zalando-research/fashion-mnist> (date of access: 10.06.2026).