Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління
(повна назва)

Кафедра електронних обчислювальних машин
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Метод федеративного навчання на основі блокчейну

(тема)

Виконав:

студент II курсу, групи СПм-22-6

Бохан І.А.
(прізвище, ініціали)

Спеціальність

123 «Комп’ютерна інженерія»
(код і повна назва спеціальності)

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

Освітня програма

Системне програмування
(повна назва освітньої програми)

Керівник: доц. Ляшенко О.С.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри ЕОМ _____ Коваленко А.А.
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління

Кафедра електронних обчислювальних машин

Рівень вищої освіти другий (магістерський)

Спеціальність 123 «Комп’ютерна інженерія»
(код і повна назва)

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

Освітня програма Системне програмування
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
“____” _____ 20___ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

студенту Бохану Івану Анатолійовичу
(прізвище, ім’я, по батькові)

1. Тема роботи Метод федеративного навчання на основі блокчейну

затверджена наказом по університету від “ 01 ” квітня 2024 р. № 257 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 15 червня 2024 р.

3. Вхідні дані до роботи Штучний інтелект, розподільні системи, навантаження, кластер федеративне навчання

4. Перелік питань, що потрібно опрацювати у роботі

Аналіз предметної області

Розробка методу BCFL

Розробка смарт-контактів

Експериментальне тестування та оцінка

Висновки

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) 13

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

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

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Постановка задачі	1.04.24	
2	Аналіз літературних джерел	2.04-12.04.24	
3	Підготовка теоретичних аспектів за темою	12.04-30.04	
	дослідження		
4	Побудова алгоритмів методу BCFL	1.05-18.05.24	
5	Розробка смарт-контрактів	18.05-22.05	
6	Тестування оцінка системи	23.05-31.05	
7	Підготовка публікації за темою дослідження	1.06-13.06.24	
8	Формування презентації	3.06.24	
9	Перевірка на антіплагіат	13.06.24	
10	Відправка на рецензування	14.06	

Дата видачі завдання 01 квітня 2024 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Ляшенко О.С.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 60 с., 16 рис., 1 табл., 1 дод., 27 джерел.

ФЕДЕРАТИВНЕ НАВЧАННЯ; ЕФЕКТИВНІСТЬ КОМУНІКАЦІЇ; КЛАСТЕРНЕ НАВЧАННЯ; БЛОКЧЕЙН.

Метою кваліфікаційної роботи є реалізація метода федеративного навчання на основі блокчейн

У ході виконання кваліфікаційної роботи пропонується метод об’єднаного навчання, заснований на непарно-парному навчанні кластера. Розділивши клієнтів на кластери та застосувавши частково серіалізований метод навчання в кластерах, ми можемо прискорити конвергенцію моделі. Перед передачею параметрів файл моделі розріджується та квантується, щоб зменшити витрати на зв’язок і підвищити ефективність зв’язку FL.

Архітектура BCFL більше не покладається на центральний сервер і використовує алгоритм балансування навантаження для планування клієнта, відповідального за агрегацію, у кожному раунді. Він представляє ланцюжок консорціуму для запису процесу FL і оптимізує проблему великих накладних витрат на зберігання ланцюга консорціуму шляхом поєднання з IPFS.

ABSTRACT

Master’s thesis: 60 pages, 16 figures, 1 tables, 1 appendices, 27 sources.

FEDERATED LEARNING; COMMUNICATION EFFICIENCY; CLUSTER TRAINING; BLOCKCHAIN

The major goal of this thesis is implementation of the federated learning method based on blockchain

In the course of the qualification work, a combined learning method based on odd-even cluster learning is proposed. By dividing the customers into clusters and applying a partially serialized learning method to the clusters, we can speed up the convergence of the model. Before transmitting parameters, the model file is sparse and quantized to reduce communication overhead and improve FL communication efficiency.

The BCFL architecture no longer relies on a central server and uses a load balancing algorithm to schedule the client responsible for aggregation in each round. It introduces a consortium chain to record the FL process and optimizes the problem of high storage overhead of the consortium chain by combining with IPFS.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

IPFS – InterPlanetary File System

FL – федеративне навчання

SWRR – Smooth Weighted Round-Robin

ВСТУП

Федеративне навчання (FL) – це новий метод машинного навчання, у якому всі учасники можуть спільно навчати модель, не передаючи свої необроблені дані, таким чином розбиваючи дані й уникаючи проблем конфіденційності, спричинених централізованим зберіганням даних. У практичних програмах клієнтські дані не є незалежними й однаково розподілені, що призводить до того, що FL потребує кількох раундів зв'язку для конвергенції, що тягне за собою високі витрати на зв'язок. Крім того, централізована архітектура традиційного FL залишається сприйнятливою до порушень конфіденційності, перевантаження мережі та одноточкових збоїв. Щоб вирішити ці проблеми, у цьому документі пропонується фреймворк FL, заснований на технології блокчейн і алгоритм навчання кластера, який називається BCFL. Спершу ми вдосконалили алгоритм FL на основі навчання кластерів із парним і непарним циклом, який прискорює конвергенцію моделі шляхом поділу клієнтів на кластери та впровадження серіалізованого навчання в кожному кластері. Тим часом операції стиснення були застосовані до параметрів моделі перед передачею, щоб зменшити витрати на зв'язок і підвищити ефективність зв'язку. Потім була розроблена та розроблена децентралізована архітектура FL на основі блокчейну та міжпланетної файлової системи (IPFS), де блокчейн записує процес FL, а IPFS оптимізує високі витрати на зберігання, пов'язані з блокчейном. Експериментальні результати демонструють перевагу фреймворку з точки зору точності та ефективності зв'язку.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Теоретичні аспекти

У березні 2023 року Асоціація глобальної системи мобільного зв’язку (GSMA) опублікувала свій щорічний звіт «Мобільна економіка» [1], у якому зазначено, що на кінець 2022 року понад 5,4 мільярда людей у всьому світі були підписані на послуги мобільного зв’язку, причому 4,4 мільярда також за допомогою мобільного Інтернету. Очікується, що до 2027 року кількість смартфонів, підключених до Інтернету, досягне 8 мільярдів із 6 мільярдами користувачів мобільних терміналів. Очікується, що до 2030 року технологія 5G принесе майже 1 трильйон доларів світовій економіці, що принесе користь усім галузям. Протягом наступних 7 років зі збільшенням кількості термінальних пристроїв, які підключаються до Інтернету, спостерігатиметься вибухове зростання мережевих даних. З настанням ери великих даних питання захисту конфіденційності даних стає все більш важливим. Однак у сфері машинного навчання для навчання високопродуктивних прогностичних моделей часто потрібні великі обсяги даних, і ці дані часто містять конфіденційну інформацію користувача, витік якої може мати серйозні наслідки для конфіденційності. Традиційні методи машинного навчання зазвичай інтегрують набори даних для навчання, що ускладнює захист конфіденційності даних. Щоб вирішити широко поширену проблему конфіденційності даних, Європейський Союз запровадив Загальний регламент захисту даних (GDPR) [2], перший закон про захист конфіденційності даних, який встановлює комплексні правила захисту приватних даних. У результаті федеративне навчання з’явилося як новий децентралізований метод машинного навчання, спрямований на вирішення проблем ізоляції та конфіденційності даних. У федеративному навчанні дані зберігаються окремо на різних локальних пристроях, а за допомогою таких методів, як шифрування

та безпечне обчислення, можна досягти обміну даними та навчання моделі без необхідності зберігати набори даних на будь-якому централізованому сервері. Цей метод не тільки захищає конфіденційність даних користувачів, але й використовує величезні обсяги даних для спільного навчання більш точних і надійних моделей, забезпечуючи кращу підтримку та допомогу для розробки та прийняття рішень у різних галузях.

Хоча федеративне навчання досягло значного прогресу в захисті конфіденційності [3], у практичних сценаріях для досягнення конвергенції моделі потрібні численні раунди зв'язку через участь кількох терміналів у передачі великих моделей нейронної мережі та незалежних і однаково розподілених ((не IID) характер даних, залучених до навчання більшості терміналів, тому комунікаційні витрати стають проблемою, яку не можна ігнорувати. У об'єднаному навчанні на загальний час навчання моделі в основному впливає час навчання кожної моделі терміналу та час передачі зв'язку. Завдяки постійному вдосконаленню обчислювальної потужності пристроїв швидкість навчання клієнтів поступово зростає, а ефективність зв'язку стала головним вузьким місцем, що обмежує швидкість навчання.

Blockchain – це технологія розподіленої бази даних, яка використовує структуру даних, подібну до ланцюга, для зберігання та керування даними. Дані упаковуються в окремі блоки, які потім з'єднуються в хронологічному порядку, утворюючи ланцюжок, що постійно зростає. Блокчейн використовує різні технології, включаючи протоколи однорангової мережі, криптографію, механізми консенсусу та смарт-контракти, щоб створити безпечне та надійне середовище для зберігання та передачі даних. Він характеризується такими характеристиками, як децентралізація, безпека, прозорість, незмінність і можливість аудиту [4]. Традиційне федеративне навчання – це централізована архітектура, у якій центральний сервер відповідає за збір, агрегування та трансляцію нової глобальної моделі. Ця архітектура має проблеми, такі як витік конфіденційності, перевантаження мережі та єдина точка відмови. Щоб вирішити ці проблеми, децентралізована об'єднана архітектура навчання

зазвичай не потребує центрального сервера і покладається головним чином на блокчейн для виконання відповідної роботи. Цей підхід може краще захистити конфіденційність користувачів, покращує ефективність і безпеку федеративного навчання, а також має хорошу масштабованість і відмовостійкість.

Мета дослідження, описаного в цій роботі, полягала в тому, щоб покращити ефективність інтегрованого навчання, зокрема ефективність спілкування. У практичних застосуваннях характеристики клієнтських даних, не пов’язані з IID, у федеративному навчанні збільшують кількість раундів зв’язку, необхідних для досягнення конвергенції. Необхідно вивчити ефективний алгоритм об’єднаного навчання, який підходить для реальних сценаріїв об’єднаного навчання. Крім того, у федеративному навчанні відсутній механізм довіри, що може призвести до таких проблем, як витік даних і підробка моделі. Тому технологія блокчейн потрібна для забезпечення довіри та безпеки систем федеративного навчання. Щоб вирішити ці проблеми, у цьому документі пропонується інноваційний підхід як до алгоритмів федеративного навчання, так і до архітектури, пропонуючи структуру федеративного навчання на основі блокчейну та кластерного навчання (BCFL). Внески цього документа підсумовуються таким чином:

- з точки зору алгоритмів, ця робота покращує алгоритм об’єднаного навчання, заснований на навчанні кластера з парним і непарним циклом, шляхом включення блокчейну замість центрального сервера. Це вдосконалення забезпечує вищий рівень конфіденційності, безпеки та відстеження даних, одночасно знижуючи ризик єдиної точки збою;

- з точки зору стиснення даних, параметри моделі, отримані в результаті локального навчання на стороні клієнта, піддаються операціям розрідженого квантування та інтегруються у формат квантування та зберігання з розділеними значеннями (QVCSS), запропонований у цьому документі, перед передачею, тим самим зменшуючи зв’язок вартість;

- з точки зору архітектури пропонується децентралізована архітектура федеративного навчання на основі блокчейну та IPFS. По-перше, замість того, щоб покладатися на центральний сервер, смарт-контракт розроблено для реалізації алгоритму Smooth Weighted Round-Robin (SWRR) для вибору клієнта, відповідального за глобальне агрегування градієнтів у кожному раунді, що ефективно дозволяє уникнути одноточкової відмови центральний сервер у традиційному інтегрованому навчанні. По-друге, приймається ланцюжок консорціуму, який більше підходить для таких сценаріїв, як інтегроване навчання, що потребує ефективної обробки великих обсягів даних, оскільки ланцюжок консорціуму може покращити продуктивність за рахунок обмеження кількості учасників і використання ефективніших алгоритмів консенсусу. Крім того, використання IPFS оптимізує проблему великих накладних витрат на зберігання файлів параметрів моделі в ланцюжку консорціуму.

1.2 Аналіз сучасних досліджень

Щоб підвищити ефективність зв’язку, вітчизняні та іноземні дослідники провели поглиблені дослідження оптимізації алгоритмів, стиснення даних і федеративного навчання в сценаріях даних, не пов’язаних з IID. У режимі початкового об’єднаного навчання клієнтські пристрої, які беруть участь у навчанні, запускають один раунд алгоритму стохастичного градієнтного спуску та завантажують локально оновлені параметри моделі на центральний сервер. Часте спілкування може спричинити надмірну передачу даних на центральний сервер і перевантаження мережі. Тому, починаючи з оптимізації алгоритму федеративного навчання, в [5] запропонувано алгоритм FedAvg. Цей алгоритм дозволяє клієнтським пристроям, які беруть участь у навчанні, виконувати кілька раундів алгоритму стохастичного градієнтного спуску локально перед тим, як надсилати параметри моделі на центральний сервер для агрегації за допомогою зваженого усереднення. У порівнянні з алгоритмом

FedSGD [6], алгоритм FedAvg покладає обчислювальне навантаження на локальні клієнтські пристрої, тоді як центральний сервер відповідає лише за агрегацію, що зменшує кількість раундів зв’язку більш ніж у 10 разів. Базуючись на алгоритмі FedAvg, в [7] поєднали ідею ієрархічної кластеризації з федеративним навчанням, використовуючи подібність між локальними оновленнями та глобальною моделлю для кластеризації та розділення клієнтських пристроїв, і досягнув зменшення загальної кількості раундів зв’язку.

З точки зору стиснення даних, існуючі дослідження показують, що розмір моделі машинного навчання має найбільший вплив на витрати на зв’язок. Ефективного скорочення обміну даними по висхідній і низхідній лінії зв'язку можна досягти за допомогою стиснення, що зменшує загальну вартість зв'язку. Однак стиснення даних також може призвести до часткової втрати інформації, тому необхідно знайти оптимальний баланс між точністю моделі та ефективністю зв'язку. В [8] запропонувано фіксовану швидкість розрідженості, сортуючи градієнти, отримані за допомогою локальних клієнтських пристроїв під час навчання, і передаючи лише верхній P% градієнтів на центральний сервер, а решту зберігаючи в залишковому векторі.

В [9] запропонував адаптивну стратегію квантування під назвою AdaQuantFL, яка динамічно змінює кількість рівнів квантування під час процесу навчання для досягнення балансу між точністю моделі та ефективністю зв’язку. В [10] представлено Compressed Sensing (CS) у федеративне навчання та досяг хороших показників ефективності зв’язку та точності моделі.

Дані в інтегрованому навчанні надходять від різних кінцевих користувачів, і дані, створені цими користувачами, часто мають незалежний і ідентичний розподіл. Дані, зібрані на їхніх незалежних пристроях, не можуть відображати розподіл даних інших наборів даних учасників. Робота з даними, не пов’язаними з IID, може спричинити такі проблеми, як труднощі з конвергенцією під час навчання та надмірні раунди зв’язку. Розглядаючи

проблеми, викликані даними, що не є IID, на клієнтських пристроях, в [11] запропонувано метод розрідженого потрійного стиснення (STC), який розширює існуючі методи top-K і поєднує кодування Голомба для розрідження зв'язку вниз по потоку. Цей метод може допомогти швидше об'єднати моделі об'єднаного навчання. В [12] запропонувано набір алгоритмів із періодичним стислим зв'язком і ввів локальне відстеження градієнта для налаштування напрямку градієнта клієнтських пристроїв у сценаріях без IID. В [13] зменшив дисбаланс класів у клієнтських пристроях за допомогою локальної вибірки даних і між клієнтськими пристроями шляхом вибору клієнтів і визначення обсягу даних, які використовуються для навчання в сценаріях розповсюдження даних без IID. Це забезпечує збалансований набір розподілу класів навчальних даних у кожному раунді, досягаючи високої точності та низьких обчислень. Щоб краще усунути погіршення продуктивності федеративного навчання при використанні наборів даних зображень без IID, в [14] представлено FedDIS, який використовує стратегію розподіленого обміну інформацією із збереженням конфіденційності для клієнтів для створення незалежних і однаково розподілених (IID) наборів даних. Вони навчили остаточну модель класифікації, використовуючи локальні та доповнені набори даних у формі об'єднаного навчання, що значно покращило продуктивність моделі.

Машинне навчання та технологія блокчейн були двома дуже помітними технологіями останніх років. Було проведено значну кількість досліджень, які поєднують машинне навчання та технологію блокчейн, що демонструє ефективність їхньої співпраці [15, 16]. Водночас інтегроване навчання, як нині популярна техніка машинного навчання, також викликало багато досліджень архітектур децентралізованого федеративного навчання, які поєднують його з технологією блокчейн. В [17] розглядав труднощі реалізації принципів відповідального штучного інтелекту в системах федеративного навчання та проблеми несправедливості, які виникають при роботі з різними зацікавленими сторонами та даними, не пов'язаними з IID. Вони

запропонували базовану на блокчейні архітектуру підзвітності та справедливості для систем федеративного навчання. Система розробляє смарт-контракт реєстру джерела моделі даних, використовує блокчейн для зберігання даних, навчених за допомогою локальних моделей, і хеш-значень локальних і глобальних версій моделі, відстежує та записує джерела моделі даних і реалізує підзвітність аудиту. Jiang та ін. запропонував модель виявлення загроз для обміну інформацією про кіберзагрози (CTI), яка називається BFLS [18]. Вони використовували блокчейн для подолання ризиків відмови одного сервера та шкідливих вузлів під час спільного процесу моделі. Щоб покращити практичну візантійську відмовостійкість (PBFT) у ланцюжку консорціуму, вони перевірили та відібрали високоякісну інформацію про загрози для участі у федеративному навчанні, а потім автоматично агрегували та оновили модель за допомогою смарт-контрактів, відфільтровуючи низькоякісні та помилкові дані про загрози. захищаючи конфіденційність організації. В [19] запропонували метод навчання децентралізованої федеративної передачі на основі блокчейну для вирішення проблеми спільної діагностики механічних несправностей у промислових сценаріях. Вони представили блокчейн, щоб замінити центральний сервер у федеративному навчанні, значно підвищивши безпеку обміну даними. Для оптимізації моделі та мотивації учасників було запропоновано консенсус комітету та схему агрегування моделі. Крім того, вони розробили алгоритм навчання передачі без вихідних даних для створення локальних індивідуальних моделей. В Інтернеті речей (IoT) деякі дані в Mobile Edge Computing (MEC) є дуже конфіденційними, тому прямий обмін порушує конфіденційність. Чжан та ін. запропонував архітектуру вертикального федеративного навчання (VFL), засновану на блокчейні консорціуму для обміну даними в MEC [20]. Ланцюг консорціуму замінює третю сторону в традиційному VFL і відповідає за розповсюдження ключів і завдання дешифрування навчання. Вони вдосконалили алгоритм Raft на основі випадкових функцій, які можна перевірити, і запропонували новий алгоритм

узгодженості, V-Raft, який може швидко та стабільно вибирати лідерів для допомоги в обміні даними VFL. Проаналізувавши ключові питання, які необхідно вирішити в управлінні ресурсами в майбутньому IoT, в [21] використав федеративне навчання для виконання розподіленого інтелектуального управління ресурсами IoT і запропонував структуру управління ресурсами IoT, що поєднує блокчейн і федеративне навчання. Вони використовували Support Vector Machine (SVM) для виявлення шкідливих вузлів і оптимізації відбору учасників інтегрованого навчання. Крім того, вони забезпечили безпеку обміну параметрами моделі шляхом завантаження локальних параметрів моделі та глобальних параметрів моделі в блокчейн.

2 РОЗРОБКА МЕТОДУ BCFL

2.1 Архітектура системи

У цій роботі застосовано блокчейн і IPFS до архітектури FL на основі алгоритму навчання кластера з парним і непарним циклом і поєднує операції розрідженого квантування параметрів моделі зі структурою сітчастої топології для подальшого підвищення ефективності зв'язку, забезпечуючи при цьому надійність FL. Як показано на рисунку 2.1, децентралізована архітектура BCFL складається з чотирьох рівнів: рівня навчання кластера, рівня блокчейну та рівня зберігання моделі.

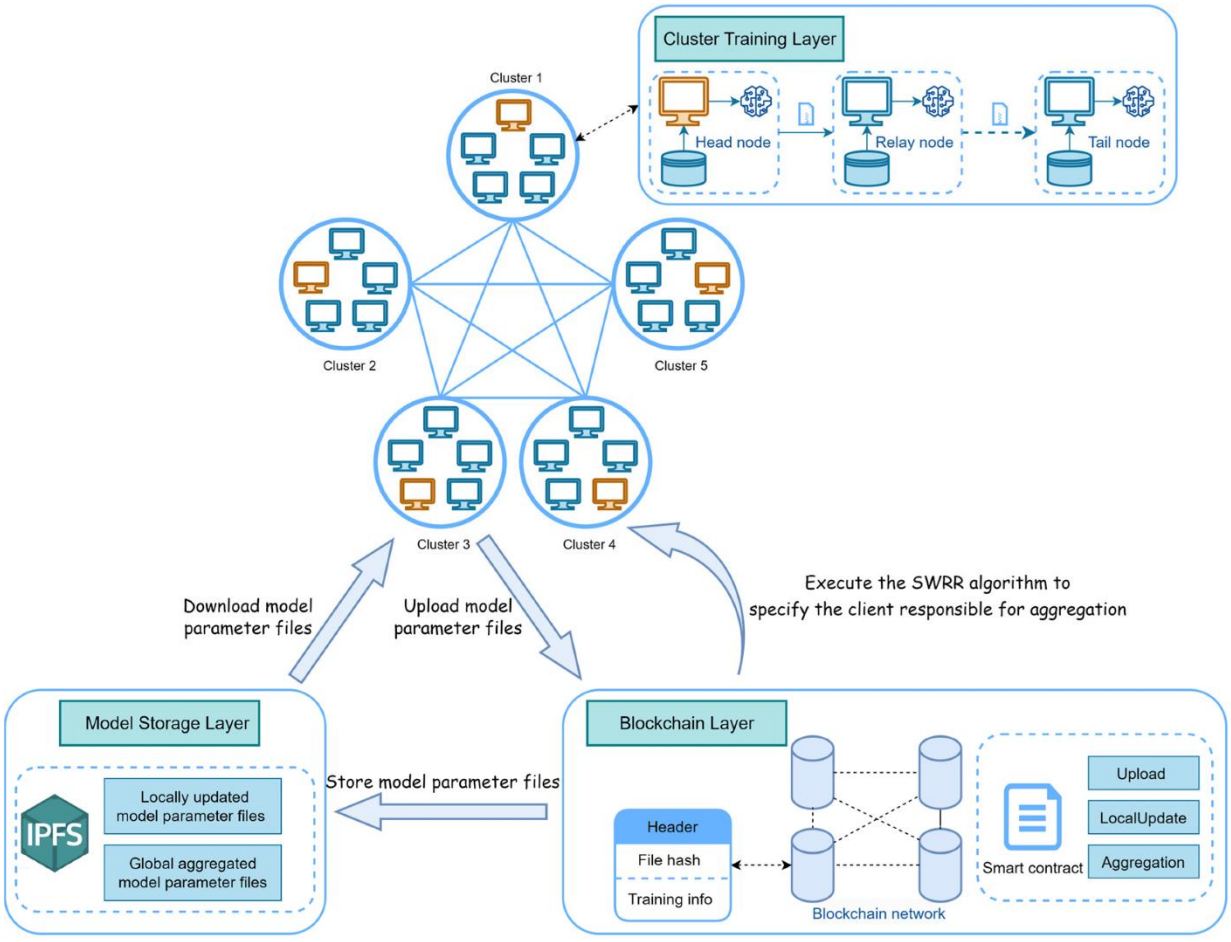

Рисунок 2.1 – Архітектура системи BCFL

На рівні навчання кластера клієнтські вузли дотримуються правил алгоритму навчання кластера з парним і непарним циклом для проведення навчання моделі та передачі параметрів у кластері. Клієнти-учасники поділяються на кілька кластерів, а відповідні клієнти обираються для участі в навчанні в парних і непарних раундах у кожному кластері. Під час кожного раунду навчання параметри моделі передаються від головного вузла до релейного вузла і, нарешті, до хвостового вузла, тоді як кожен вузол оновлює параметри моделі на основі свого локального набору даних. Зокрема, перед передачею параметрів моделі клієнт інтегрує параметри моделі у формат QVCSS і виконує стиснення без втрат на $indices$ у файлі параметрів, щоб ще більше зменшити простір, зайнятий файлом параметрів моделі, тим самим покращуючи ефективність зв'язку.

На рівні блокчейну смарт-контракти призначені для планування взаємодії між клієнтами FL, блокчейном і IPFS. Рівень навчання кластера викликає смарт-контракти, щоб забезпечити завантаження файлів параметрів моделі, виведених клієнтськими вузлами, і завантаження необхідних файлів параметрів моделі для локальних оновлень. Під час завантаження файлів параметрів моделі їхні відповідні хеш-значення та відповідна навчальна інформація упаковуються та зберігаються в блоках у блокчейні консорціуму. Крім того, через відсутність центрального сервера в архітектурі BCFL для агрегації параметрів моделі смарт-контракт розроблений для реалізації алгоритму SWRR для визначення клієнта, відповідального за агрегацію в кожному раунді.

На рівні зберігання моделі IPFS відповідає за зберігання реальних файлів параметрів моделі, згенерованих локальними оновленнями та глобальною агрегацією. Клієнт FL може знайти відповідний повний файл в IPFS на основі хешу файлу, що зберігається в блокчейні, і завантажити його. Цей підхід не тільки вирішує проблеми зберігання блокчейну консорціуму, але й забезпечує децентралізоване зберігання.

2.2 Алгоритм об’єднаного навчання на основі кластерного навчання непарних і парних груп

Базуючись на алгоритмі федеративного навчання з парно-непарним циклом і градієнтним розрідженням (FedOES) [22], у цій роботі пропонується алгоритм федеративного навчання на основі кластерного навчання парно-непарного циклу (FedOEC) шляхом видалення центрального сервера. Основна ідея запропонованого FedOEC полягає в тому, щоб розділити клієнтів-учасників на кілька кластерів і вибрати підмножину клієнтів для послідовного навчання в кожному кластері відповідно до парних і непарних раундів. Крім того, щоразу, коли параметри моделі передаються, алгоритм виконує операції стиснення градієнта, щоб зменшити обсяг даних зв’язку.

Потім інформація про градієнт, створена кожним кластером, агрегується. Цей процес ефективно покращує комунікаційну ефективність федеративного навчання, а кластерний метод показує високу надійність у обробці сценаріїв, не пов’язаних із IID. Алгоритм FedOEC можна розділити на три етапи: ініціалізація, навчання кластера та агрегація моделі. Потік алгоритму детально проаналізовано нижче, а псевдокод алгоритму для FedOEC показано в рисунку 2.2.

Ініціалізація: клієнти випадковим чином розподіляються на N кластери, припускаючи, що є $2k$ клієнтів у кожному кластері, а клієнтські вузли в кластері позначаються як $\mathrm{C} = \{\mathrm{C}_n^1, \mathrm{C}_n^2, \dots, \mathrm{C}_n^{kn}\}$. У кожному кластері є три типи вузлів: головний вузол, хвостовий вузол і релейний вузол. Оскільки процес навчання поділено на парні та непарні раунди, кожен кластер має два головних вузла, два хвостових вузла та кілька релейних вузлів. У кластері n , головний вузол позначається як C_n^{head}, хвостовий вузол позначається як C_n^{tail} $Ct\ a\ i\ ln$, а решта вузлів реле позначені як $\mathrm{C}_n^{mid} = \{\mathrm{C}_n^3, \mathrm{C}_n^4, \dots, \mathrm{C}_n^{2k-2}\}$. Зокрема, головний вузол, який бере участь у навчанні непарного раунду, позначається як C_n^1, а хвостовий вузол позначено як C_n^{2k-1}; головний вузол, який бере участь у тренуванні парного раунду, позначений як C_n^2 , а хвостовий вузол позначено

як C_n^{2k}. Глобальна модель w_0 ініціалізується та надсилається до головних вузлів N кластери, які позначаються як $C^{head} = \{C_1^{head}, C_2^{head}, \dots, C_N^{head}\}$.

Input: ω^0: initialize global model;
Output: ω^R: model after R rounds of aggregations;
Initialization: each client downloads global model ω^0;
for round r = 1,2,3…R, do
    execute in-cluster training;
    receive N tails of clusters: $\omega^*_1, \omega^*_2, \dots, \omega^*_N$;
    select client for aggregation by SWRR;
    model aggregation: $\omega^r = \frac{\sum_{n=1}^{N} \omega^*_n}{N}$;
    gradient compression: $\omega^{*r} = Sparse_k(|\omega^r|)$;
    if r = R − 1
        send model ω^{*r} to heads of clusters;
    else
        $\omega^R = \omega^{*r}$;
        return ω^R;
    end
end
In-cluster training:
for each cluster n = 1, 2, 3, … N, do
    if r%2 = 1 then
        for each client 1, 3, 5, … $2k$ − 1, do
            if k = 1 then
                $\omega_n^k = \omega_n^{r-1} - \eta \nabla_\omega \ell(\omega^{*r-1}, x_n^k, b)$
            else
                $\omega_n^k = \omega_n^{r-1} - \eta \nabla_\omega \ell(\omega^{*k-2}, x_n^k, b)$
            end
            $\omega^{*k}_n = Sparse_k(|\omega_n^k|)$
        end
    else
        /* similar to above process */
    end
end
$\omega^*_n = \omega_n^{tail}$;
return;

Рисунок 2.2 – Алгоритм FedOEC

Навчання кластеру: проходження непарного раунду кластераn як приклад, головний вузол C_n^1 виконує локальне навчання після отримання глобального градієнта, отриманого з попереднього раунду агрегації. Формула локального тренування головного вузла така:

$$w_n^1 = w^{*r-1} - \eta \nabla_w F(w^{*r-1}, x_n^1, b),$$

де w_n^1 представляє локальні параметри моделі, оновлені головним вузлом у непарному раунді; w^{*r-1} являє собою глобальний градієнт, підданий операції стиснення після попереднього раунду агрегації, де r позначає поточний тренувальний раунд; η і F представляють швидкість навчання та функцію втрат відповідно; x_n^1 представляє навчальні дані на клієнті головного вузла; і b являє собою кількість локальних тренувальних раундів.

Щоб зменшити витрати на передачу параметрів моделі, кожному клієнту потрібно виконати градієнтне стиснення локальних параметрів моделі. На прикладі головного вузла формула виглядає наступним чином:

$$w_n^{*1} = Sparse_k(|w_n^1|),$$

де w_n^{*1}являє собою параметри, отримані в результаті виконання $Sparse_k()$ робота з локально оновленими параметрами моделі w_n^1 за допомогою головного вузла. $Sparse_k()$ зменшує обсяг пам'яті, зайнятої параметрами моделі, що сприяє їх передачі. У кожному кластері клієнти виконують послідовне навчання в певному порядку, як показано на рисунку 2.3.

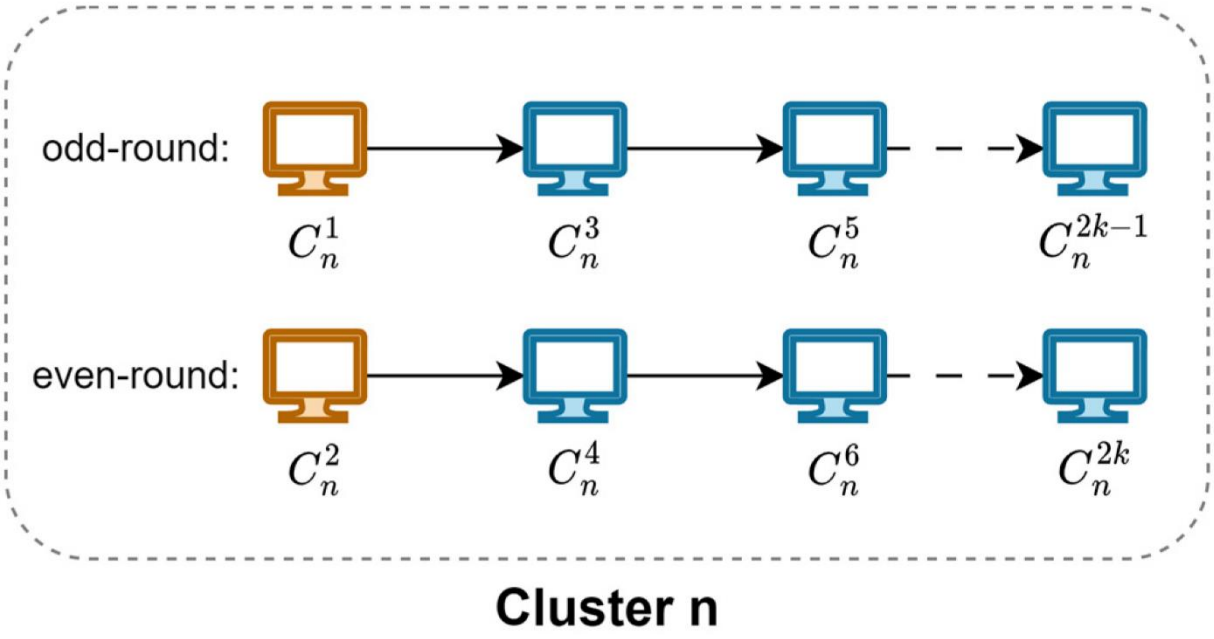

Рисунок 2.3 – Послідовний процес навчання в кластері n

Взявши, наприклад, непарний раунд після головного вузла C_n^1 навчений отримувати локальну модель і виконує градієнтне стиснення параметрів локальної моделі, він передає параметри моделі $w_n^{*\,1}$до релейного вузла C_n^3 , і так далі, до хвостового вузла C_n^{tail} завершує локальне оновлення та виконує $Sparse_k()$ функція. Таким чином кластеризуйтеn отримає остаточні параметри моделі $w_n^{*\,2k-1}$ цього раунду, тобто w_n^*. Вузли ретрансляції та хвостовий вузол, які беруть участь у цьому раунді навчання, виконуватимуть таке локальне оновлення та градієнтне стиснення:

$$w_n^k = w_n^{*\,k-2} - \eta \nabla_w F(w^{*\,k-2}, x_n^k, b),$$

$$w_n^{*\,k} = Sparse_k(|w_n^k|).$$

Агрегація моделі: все-таки N кластери завершили навчання, хвостовий вузол C_n^{tail} кожного кластера передаватиме параметри моделі w_n^* що представляє кластер n . Після отримання набору параметрів моделі $w_n^* =$

$\{w_1^*, w_2^*, \ldots, w_N^*\}$ з кожного кластера клієнтський вузол вибирається за допомогою алгоритму SWRR для агрегації моделі для отримання нового глобального параметра моделі w^r . Формула для агрегування моделі така:

$$w^r = \frac{\sum_{n=1}^{N} w_n^*}{N},$$

де r позначає поточний раунд, а алгоритм SWRR детально описано в розділі 3.3.1. Після завершення агрегації моделі операція $Sparse_k()$ виконується наwr отримати w^{*r} , який потім передається до головних вузлів N кластерів, які беруть участь у наступному раунді навчання.

2.3 Стиснення параметрів моделі

Згідно з робочим процесом алгоритму FedOEC, можна побачити, що нам потрібно виконати функцію градієнтного стиснення $Sparse_k()$ на оновлені параметри моделі після локального навчання на стороні клієнта або агрегування кожної моделі. $Sparse_k()$ функція складається з двох кроків:

- виконайте операцію top-K над інформацією про градієнт, тобто виберіть верхні градієнти K з найбільшими абсолютними значеннями;
- перетворення щільної матриці, отриманої в результаті операції top-K, у розріджену матрицю, збережену у форматі QVCSS.

2.3.1 Операція Топ-К

Параметри моделі зберігаються в словнику як пари ключ-значення, де ключ представляє назву параметра, наприклад Weight, Bias тощо, а значення представляє відповідне значення параметра, як правило, тензор. У процесі навчання необхідно оновити велику кількість параметрів, але не всі вони істотно впливають на продуктивність моделі. Таким чином, під час оновлення

параметрів можна використовувати техніку top-K для вибору підмножини градієнтів із більшими абсолютними значеннями для оновлення моделі, тоді як решту градієнтів можна ігнорувати, таким чином зменшуючи накладні витрати на зв’язок і обсяг завантажуваних даних. Процес виконання операції top-K над інформацією про градієнт показано на рисунку 2.4.

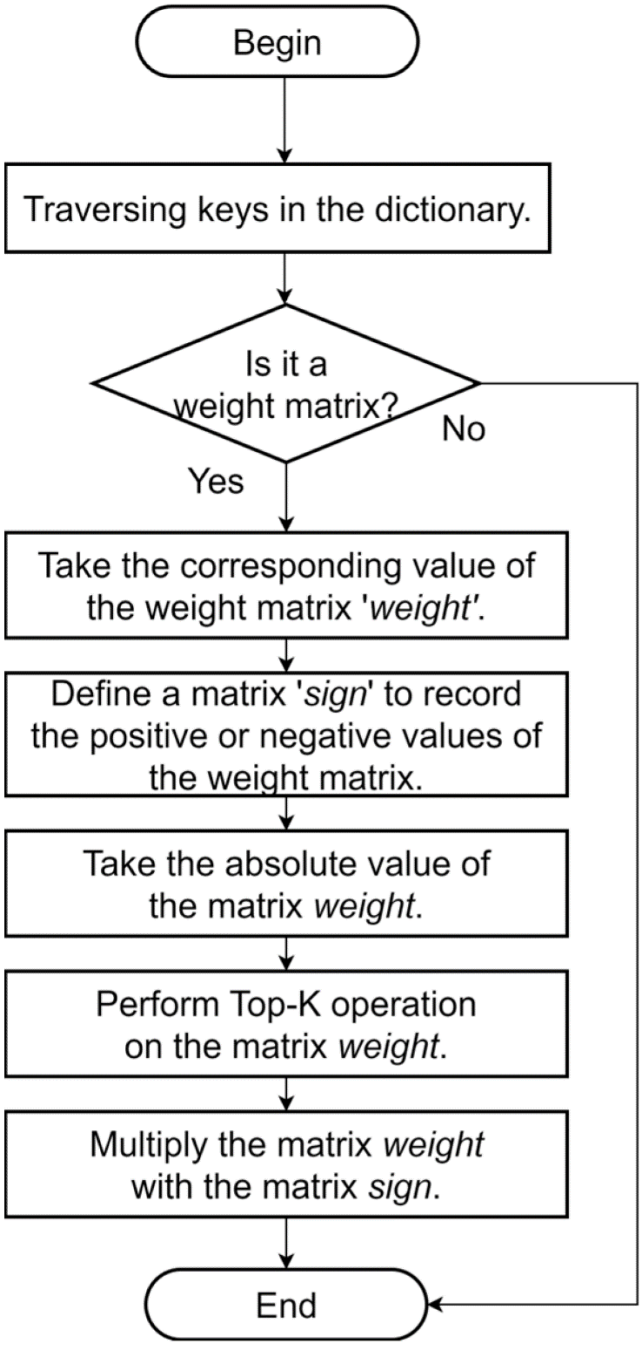

Рисунок 2.4 – Блок-схема операції top-K

Тут діапазон значень елементів матриці $sign$ є$\{-1, 0, 1\}$, де -1 вказує на від'ємне число і 1 вказує на додатне число. У цій роботі для представлення операції top-K на багатовимірних тензорах використовується наступна формула:

$$x_{i,j,\dots} = \begin{cases} x_{i,j,\dots} \in top(K) \\ x_{i,j,\dots} \overline{\in} top(K) \end{cases},$$

де $x_{i,j,\dots}$ представляє елементи в $weight'$ отримане шляхом взяття абсолютного значення початкового тензора, і $top(K)$ являє собою набір вершини K найбільші елементи в тензорі $weight'$. Якщо $x_{i,j,\dots}$знаходиться в $top(K)$ набір, елемент зберігається; інакше для елемента встановлюється значення 0. Результуючий тензор $weight*$ зберігає лише верх K найбільші елементи з тензора $weight'$.

2.3.2 Формат зберігання на основі квантування та розділення значень-координат

Виконуючи операцію top-K, щільний тензор $weight*$ отримано, для якого значення багатьох параметрів є 0, що займатиме великий обсяг пам'яті та призведе до марної витрати ресурсів пам'яті. На відміну від щільних тензорів, розріджені тензори зберігають лише ненульові елементи та інформацію про їх положення, що може значно зменшити використання пам'яті. Традиційний спосіб зберігання розріджених матриць полягає у збереженні координат і значень разом як кортежі, відомі як формат координат (COO) [23]. Однак COO менш підходить для роботи з розрідженими тензорами великої розмірності, оскільки він вимагає зберігання кількох значень індексу, що збільшує витрати пам'яті.

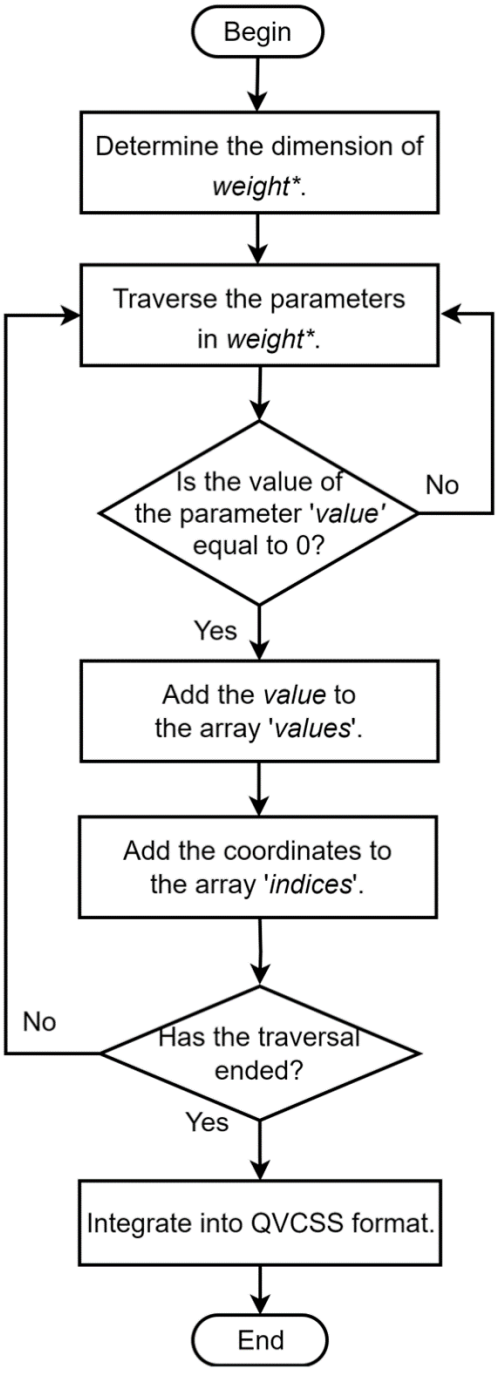

Рисунок 2.5 – Блок-схема перетворення формату QVCSS

У робочому процесі алгоритму FedOEC кожному клієнту потрібно відновити вагу $weight*$ в отриманих параметрах моделі формату QVCSS для отримання повних вагових параметрів, які потім використовуються як початковий градієнт для локальних оновлень у поточному раунді.

2.4 Зберігання параметрів моделі

Зберігання та оновлення параметрів моделі особливо важливі, і в цьому розділі описано режим зберігання, який поєднує ланцюг консорціуму з IPFS для зберігання параметрів моделі об’єднаного навчання.

Процес розділений на три етапи: ініціалізація, збереження параметрів моделі та синхронізація параметрів моделі. Ініціалізація: налаштуйте ланцюжок консорціуму, одночасно запустіть вузол IPFS і приєднайтеся до мережі IPFS.

Зберігання параметрів моделі: у кожному раунді об’єднаного навчання учасники оновлюють параметри моделі на основі своїх локальних даних, а оновлені параметри моделі зберігаються в мережі IPFS. Конкретні кроки показано на рисунок 2.6.

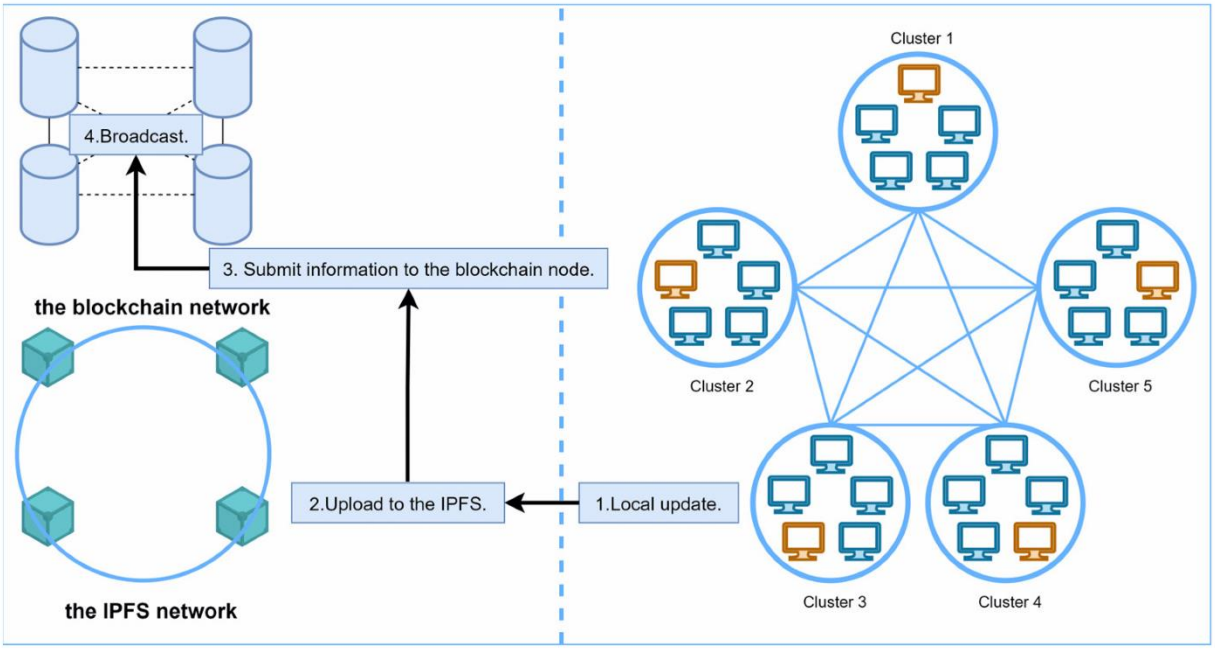

Рисунок 2.6 – Принципова схема зберігання параметрів моделі

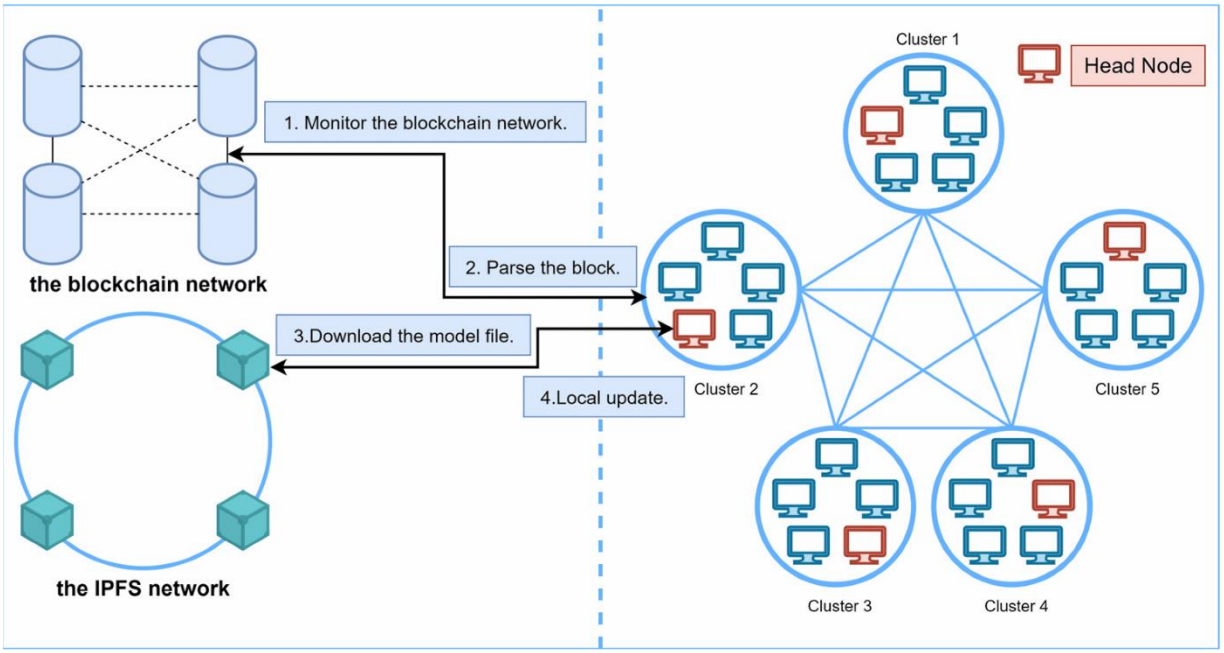

Рисунок 2.7 – Діаграма синхронізації параметрів моделі

Синхронізація параметрів моделі: щоб гарантувати, що головний вузол у кожному кластері може своєчасно отримувати останні параметри моделі, необхідно розробити механізм синхронізації. На рисунку 2.7 показано процес синхронізації.

3 РОЗРОБКА СМАРТ-КОНТРАКТУ

У цій структурі смарт-контракт відповідає за виконання основних функцій, таких як завантаження моделі, локальне оновлення та глобальне агрегування моделі. Рівень смарт-контракту включає три ключові функції: $Upload(\ )$, $LocalUpdate(\ )$, $Aggregation(\ )$.

3.1 Функція Upload().

$Upload(\ )$ функція відповідає за реалізацію завантаження параметрів моделі, яка виконується клієнтом-учасником у процесі навчання. Процес виглядає наступним чином (рисунок 3.1):

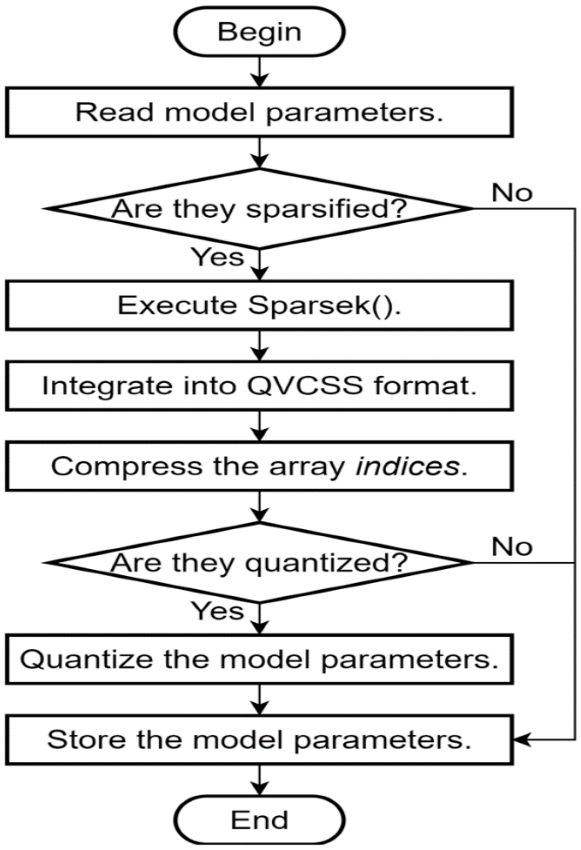

Рисунок 3.1 – Блок-схема $Upload(\ )$ функція

Читайте параметри моделі: $Upload()$ функція виконується після локального оновлення або глобальної агрегації параметрів моделі та отримує параметри моделі з локального клієнта.

Виконайте розріджену обробку та обробку квантування: вирішуйте, чи виконувати операції розрідженості чи квантування, і за необхідності виконайте $Sparsek()$ функція для інтеграції параметрів моделі у формат QVCSS. Зокрема, для стиснення використовується алгоритм стиснення deflate $indices$ масив у форматі QVCSS, що додатково зменшує простір для зберігання та вартість передачі параметрів моделі.

Збережіть параметри моделі: збережіть оброблені параметри моделі в IPFS, а потім згенеруйте хеш-значення, що відповідає файлу параметрів моделі. Хеш-значення та відповідна навчальна інформація упаковуються та зберігаються в блоці ланцюжка консорціуму, щоб інші учасники могли отримати до них доступ.

3.2 Функція LocalUpdate().

Функція $LocalUpdate()$ відповідає за обробку оновлення локальної моделі та виконується клієнтами, які беруть участь у процесі навчання. На рисунку 3.2 показаний потік функції.

Процес виглядає наступним чином:

- завантажте файл моделі: спочатку визначте, чи є клієнтський вузол головним у кластері; якщо так, то завантажте файл глобальної моделі попереднього раунду з IPFS відповідно до хеш-значення. В іншому випадку завантажте файл моделі вузла-попередника в поточному раунді;
- перевірте хеш файлу: визначте хеш-значення завантаженого файлу моделі та порівняйте обчислене значення з наданим значенням. Якщо вони збігаються, то файл повний і правильний;
- завантажити параметри моделі: завантажити параметри моделі з файлу моделі та застосувати їх до локального навчання.

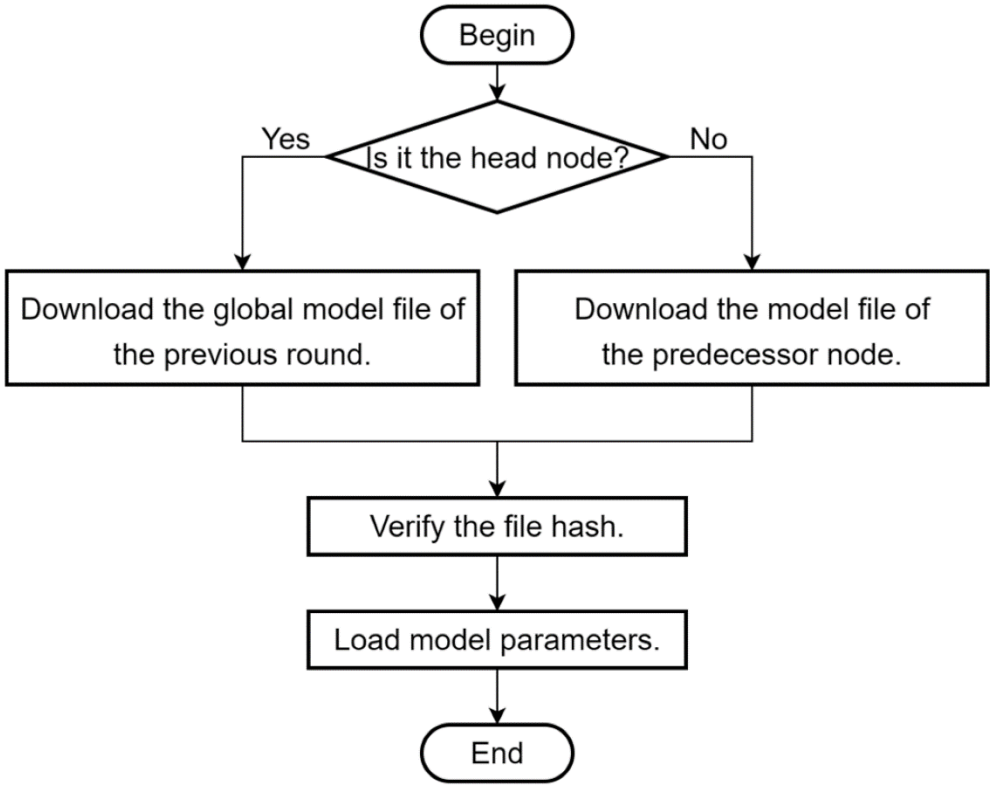

Рисунок 3.2 – Блок-схема $LocalUpdate()$ функція

3.3 Функція Aggregation()

3.3.1 Метод вибору вузла агрегації

FL на основі алгоритму SWRR Щоб досягти справді децентралізованої архітектури, BCFL скасовує центральний сервер, відповідальний за агрегування моделей у традиційній FL. Тому необхідно вибрати один із залучених клієнтських вузлів, який відповідатиме за агрегування параметрів моделі в кожному раунді.

Ми пропонуємо метод вибору вузла на основі алгоритму SWRR для визначення клієнтського вузла, відповідального за агрегацію моделі в кожному раунді. Конкретні кроки такі:

Крок 1: Ініціалізація ваг. Кожен кластер, який бере участь у FL, повинен обрати клієнтський вузол для участі в агрегації глобальної моделі. Для вибору

відповідного клієнта необхідно кожному клієнту присвоїти відповідну вагу, проаналізувавши такі показники, як його обчислювальна потужність, надійність і доступність. Зокрема, обчислювальну потужність клієнта можна оцінити на основі таких показників, як тип процесора, об'єм пам'яті та якість відеокарти, а клієнтам з більшою обчислювальною потужністю зазвичай призначають вищі ваги. Якщо є N кластери, набір клієнтських вузлів, відповідальних за агрегацію, обирається та позначається як $\mathrm{C}^{A} = \{\mathrm{C}_{n}^{\mathrm{A}}, \mathrm{C}_{n}^{\mathrm{A}}, \dots, \mathrm{C}_{N}^{\mathrm{A}}\}$.

Ініціалізовані ваги можна виразити у вигляді двох масивів, масиву відображення ваг $WeightMap$ і масив відображення динамічної ваги $Current.WeightMap$ представляє ваги, призначені клієнтам у C^{A} , а початкове значення, що відповідає кожному клієнту в $Current$ є 0. Ваги в $WeightMap$ фіксовані, а значення в $Current$ буде оновлено після агрегації моделі із зазначенням динамічної ваги кожного клієнта в поточний момент часу.

Крок 2: Виберіть кластер, відповідальний за агрегацію. Якщо припустити, що на даний момент є п'ять кластерів, після завершення першого раунду локального навчання кожне значення входить $Current$ додається до відповідної фіксованої ваги в $WeightMap$, так що $Current$ оновлюється до{"$Cluster1$": 1, "$Cluster2$": 1, "$Cluster3$": 3, "$Cluster4$": 2, "$Cluster5$" : 1}. Алгоритм SWRR визначає кластер, відповідальний за агрегацію в цьому раунді, на основі ключа, що відповідає найбільшому значенню в $Current$, який $Cluster3$.

Крок 3: Агрегування та оновлення ваги. На попередньому кроці $Cluster3$ було обрано як кластер, відповідальний за агрегацію в першому раунді. Щоб збалансувати навантаження між клієнтськими вузлами, відповідальними за агрегацію, потрібно виконати спеціальну операцію оновлення $Current$ після агрегації. Як показано на рисунку 3.3, є п'ять клієнтських вузлів, відповідальних за агрегацію, з ваговими коефіцієнтами 1, 1, 3, 2 і 1 із загальною вагою 8. Якщо клієнт C_{3}^{A} з найбільшою вагою в $Cluster3$

вибрано для обробки в кожному раунді агрегації глобальної моделі, мережеве навантаження в усьому FL не буде добре збалансованим, і клієнтські вузли з меншою вагою, наприклад C_1^A, C_2^A і C_5^A, навряд чи буде використовуватися. Тому після вибору клієнта C_3^A для агрегації моделі її вагу потрібно відняти від загальної ваги всіх учасників C^A . Це означає, що вага C_3^A буде оновлено до -5, а оновлене відображення динамічної ваги записується як $Current1'$. Потім, $Current1'$ додається до відповідної ваги в $WeightMap$, в результаті чого $Current2$={“$Cluster1$”: 2, ”$Cluster2$”: 2, “$Cluster3$”: -2, “$Cluster4$”: 4, ”$Cluster5$”: 2}. Цей метод може більш рівномірно розподілити глобальні запити агрегації.

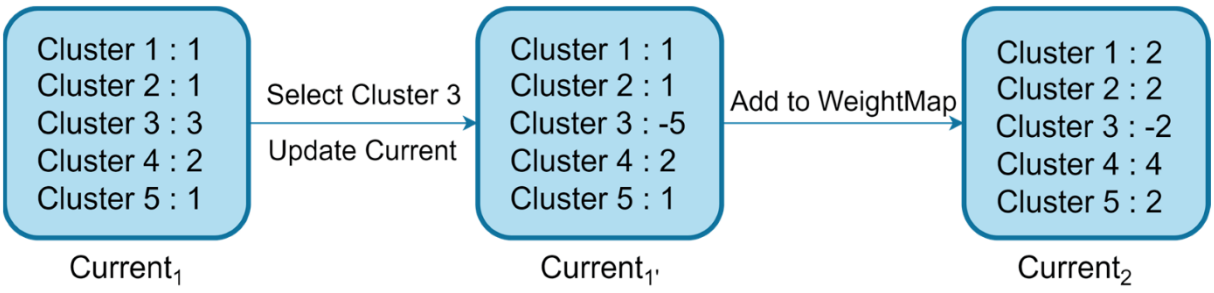

Рисунок 3.3 – Діаграма динамічного оновлення ваги

Деталі оновлення алгоритму SWRR для перших дев’яти раундів представлені в додатку, де $Current_R$ представляє ваговий масив початкового вузла агрегації в раундіR інтегрованого навчання та $Current_{R'}$ представляє оновлений масив ваг після вибору вузла агрегації. $Current_R$ для наступного раунду виходить додаванням $Current_{R'}$ до відповідної ваги в $WeightMap$.

Наведені вище кроки повторюються, доки рівень конвергенції моделі FL не досягне попередньо встановленої цілі. В алгоритмі SWRR кількість разів, коли кожен вузол вибирається для агрегації, обчислюється відповідно до вагових коефіцієнтів. Є N вузлів у наборі клієнтських вузлів C^A, а їх вага набору становить $w^A = \{w_1^A, w_2^A, \dots, w_N^A\}$. Кількість разів, коли і-й клієнтський

вузол відповідає за агрегацію, позначається як $count_i$ і розраховується за формулою:

$$count_i = round\left(\frac{w_i^A}{sum(w^A)} * R\right),$$

де R позначає раунд зв'язку, $sum(w^A)$ – загальна вага всіх вузлів, відповідальних за агрегацію, і $round()$ є поверховою функцією. Метод вибору вузла агрегації FL на основі SWRR може точніше збалансувати навантаження між вузлами та більш плавно розподілити завдання агрегації моделі.

3.3.2 Функція $Aggregation()$

Функція $Aggregation()$ відповідає за обробку глобальної агрегації моделі. Процес показаний на рисунку 3.4, а конкретні кроки такі:

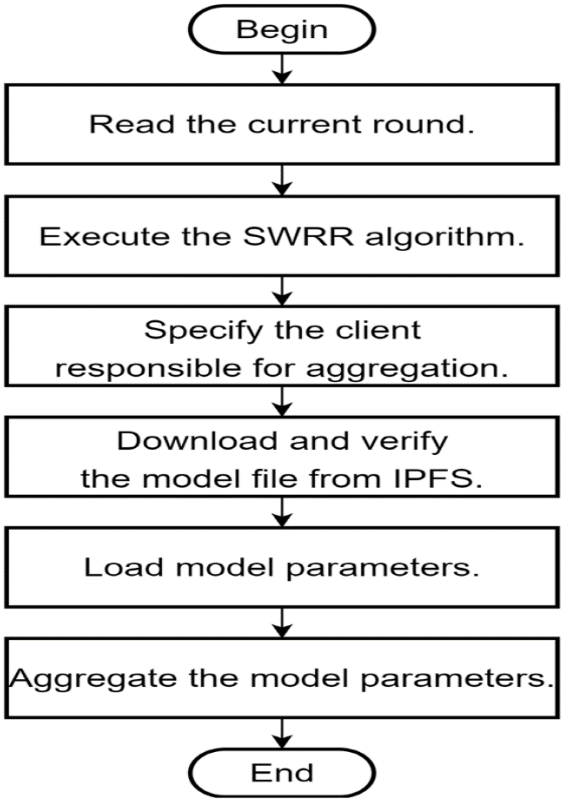

Рисунок 3.4 – Блок-схема $Aggregation()$ функція

Прочитайте поточний раунд: спочатку прочитайте інформацію про кожен кластер у поточному раунді.

Виконайте алгоритм SWRR: використовуйте інформацію поточного раунду як вхідні дані алгоритму SWRR і обчисліть масив ваг, що відповідає вузлам, відповідальним за агрегацію.

Вкажіть клієнта, відповідального за агрегацію: відповідно до результату алгоритму SWRR вкажіть вузол агрегації для поточного раунду.

Завантажте та перевірте файл моделі з IPFS: вузол агрегації завантажує файли параметрів моделі, які були завантажені хвостовими вузлами інших кластерів у поточному раунді з IPFS, і виконує перевірку хешу.

Завантажити параметри моделі: вузол агрегації завантажує параметри моделі з файлів моделі.

Агрегувати параметри моделі: вузол агрегації виконує зважене усереднення параметрів моделі всіх кластерів для отримання глобального параметра моделі.

4 ЕКСПЕРИМЕНТАЛЬНЕ ТЕСТУВННЯ ТА ОЦІНКА

4.1. Експериментальне середовище

В якості експериментального середовища використовувався комп'ютерний сервер, оснащений NVIDIA GeForce RTX 3090 (24 ГБ відеопам'яті) і процесором Intel(R) Xeon(R) Platinum 8358P @ 2,60 ГГц з 80 ГБ пам'яті. Операційною системою була Linux, а Anaconda 1.11.1 слугувала менеджером середовища Python. Python 3.8 використовувався як мова розробки, а PyTorch 1.11.0+cu113 був застосований фреймворк глибокого навчання для реалізації та навчання моделей нейронної мережі. У цьому експерименті використовувалася версія CUDA PyTorch, яка підтримує прискорення графічного процесора та значно покращує швидкість навчання та ефект моделей глибокого навчання. Платформу ланцюга консорціуму FISCO BCOS 3.2.0 було обрано для створення та керування мережею блокчейнів, а IPFS 0.10.0 _linux-arm64 було налаштовано для зберігання файлів моделі.

3.2. Експериментальний набір даних

В експерименті використовувалися два набори даних: MNIST і CIFAR-10.

1. Набір даних MNIST Набір даних MNIST містить 60 000 навчальних зображень розміром 28×28 пікселів і 10 000 тестових зображень, кожне з яких представляє рукописну цифру, кожна цифра позначена категорією від 0 до 9. Кількість міток у навчальному наборі даних становить близько 6000, а дані розподіл не зовсім рівномірний.

2. Набір даних CIFAR-10 CIFAR-10 – це широко використовуваний набір даних класифікації зображень, що складається з 60000 кольорових зображень розміром 32×32 у 10 категоріях. Набір даних поділено на

навчальний набір із 50000 зображень і набір для тестування з 10 000 зображень, і кожна категорія в навчальному наборі містить 5000 зображень. Однією з характеристик набору даних CIFAR-10 є те, що розмір і роздільна здатність зображень відносно малі, але категорії та розміри вибірки різноманітні, що робить його репрезентативним і складним.

4.3 Експериментальна метрика

Експеримент використовував точність тестування після кожного раунду агрегації моделі та вартість зв'язку за раунд як метрики оцінки. У цьому розділі наведено визначення точності та вартості зв'язку. Точність тестування є найважливішим показником для вимірювання продуктивності алгоритмів об'єднаного навчання, як показано у формулі:

$$Accuracy = \frac{Count(\widehat{y_n} = y_n)}{N},$$

де y_n являє собою етикетку зразка n, $\widehat{y_n}$ представляє прогнозоване значення моделі для y_n, і $Count(\widehat{y_n} = y_n)$ представляє кількість зразків, для яких прогнозовані результати моделі збігаються з істинною міткою. Припускаючи, що є N учасників і M раундів у навчальному процесі FL вартість зв'язку можна розрахувати за такими формулами:

$$upload\ communication\ cost = \sum_{i=1}^{N}\sum_{j=1}^{M} n_{i,j},$$

$$download\ communication\ cost = \sum_{i=1}^{N}\sum_{j=1}^{M} d_{i,j}.$$

Ці формули представляють вартість зв'язку завантаження та

завантаження даних, де $n_{i,j}$ представляє обсяг завантажених даних і $d_{i,j}$ представляє обсяг даних, завантажених і-м учасником у j-му раунді.

4.4 Постановка та впровадження експерименту

У цьому експерименті використовувалися моделі нейронних мереж, показані на рисунку 4.1, для навчання наборів даних MNIST і CIFAR-10. Оскільки набір даних CIFAR-10 має вищу роздільну здатність зображення та більше вхідних каналів, для вивчення особливостей зображень потрібна глибша та складніша нейронна мережа. Згорточна нейронна мережа, яка використовувалася для навчання набору даних CIFAR-10, відрізнялася від тієї, що використовувалася для навчання набору даних MNIST: вона мала три згорткові шари, три шари об'єднання та один повністю зв'язаний рівень. Було створено моделі згорткової нейронної мережі двох розмірів, і шляхом порівняння продуктивності цих двох моделей з різними значеннями параметрів у сценарії об'єднаного навчання можна більш чітко продемонструвати перевагу запропонованого алгоритму з точки зору зв'язку.

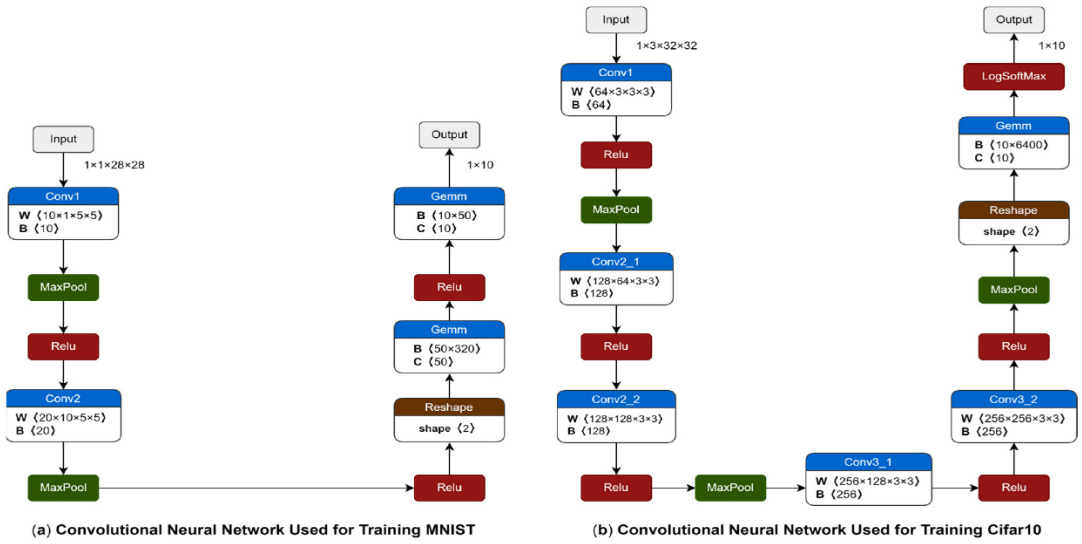

Рисунок 4.1 – Порівняння моделей згорткової нейронної мережі в цьому експерименті

Для моделювання різних сценаріїв даних, не пов’язаних із IID, навчальний набір даних був відсортований за мітками та розділений на 400 рівних частин. Під час тренінгу FL кожен із 100 клієнтів-учасників випадковим чином обирає чотири частини. На рисунках 4.2 і 4.3 гістограми представляють розподіл міток наборів даних MNIST і CIFAR-10, зайнятих 100 клієнтами. Різні кольори на гістограмах представляють різні типи міток у наборах даних.

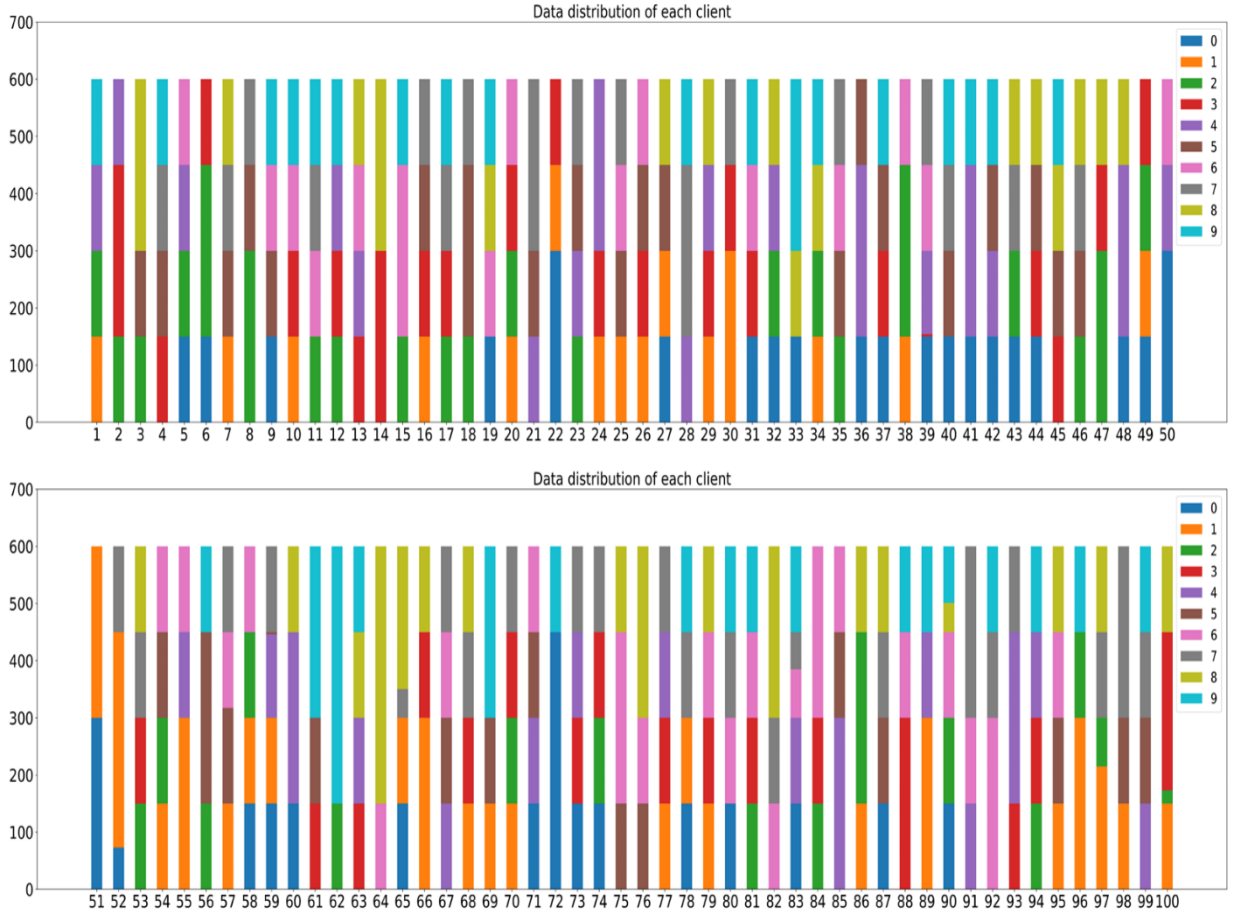

Рисунок 4.2 – Розподіл міток набору даних MNIST для кожного клієнта

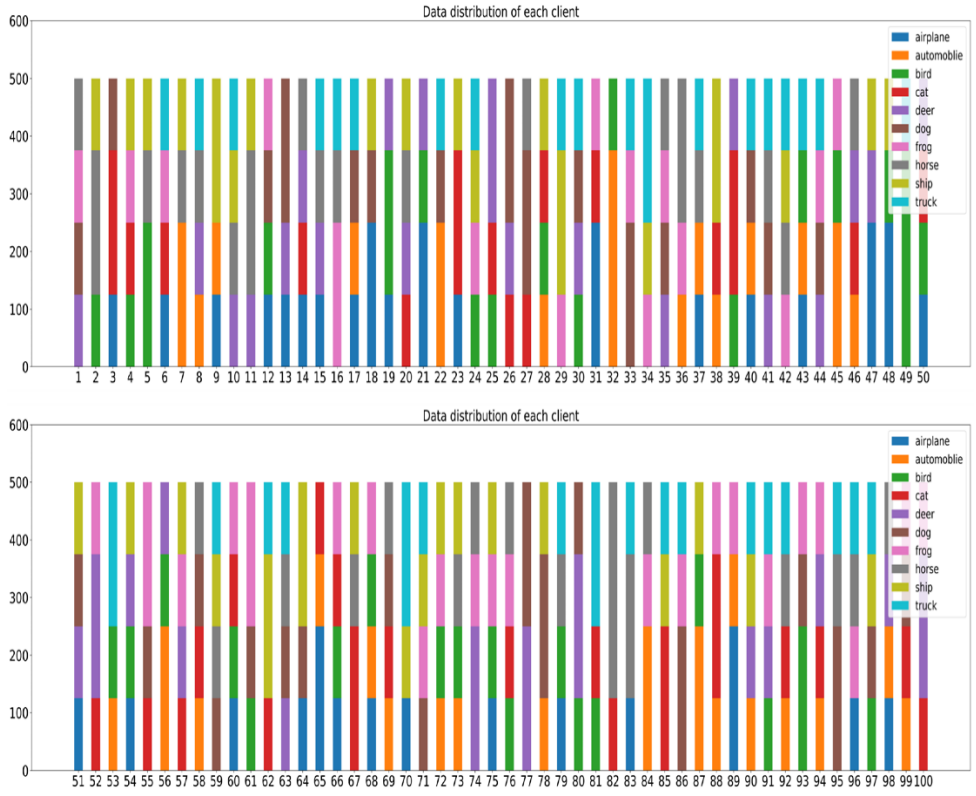

Рисунок 4.3 – Розподіл міток набору даних CIFAR-10 для кожного клієнта

В експериментальних умовах необхідно розглянути, чи є параметри моделі розрідженими, і в цьому експерименті було обрано коефіцієнт розрідженості 0,5. У той же час, також необхідно розглянути питання про квантування значень параметрів моделі від чисел з плаваючою комою повної точності до чисел половинної точності. Таким чином, для встановлення параметрів моделі в BCFL було використано три методи обробки, які наведено в таблиці 4.1.

Таблиця 4.1 Параметри обробки параметрів моделі для BCFL.

Pattern	Використовується розрідженість?	Використовується квантування?
Шаблон 1 (P1)	ні	ні
Шаблон 2 (P2)	так	ні
Шаблон 3 (P3)	так	так

4.5 Оцінка безпеки

Децентралізована структура федеративного навчання, запропонована в цьому документі, починається з принципів розробки моделі безпеки інформаційного потоку та поєднує різноманітні механізми безпеки, включаючи захист цілісності, захист механізму консенсусу та децентралізоване зберігання, щоб забезпечити цілісність і безпеку параметрів моделі, а опір різноманітним потенційним викликам і нападам.

Захист цілісності.

Відповідно до принципу захисту цілісності в моделі безпеки інформаційного потоку всі параметри моделі повинні спочатку бути оброблені хеш-алгоритмом SHA256, і ці хеш-значення зберігаються в блокчейні. Перед локальним оновленням параметри моделі, завантажені з IPFS, знову хешуються та порівнюються з хеш-значеннями, що зберігаються в блокчейні. Цей метод захищає цілісність параметрів моделі. Якщо параметри моделі змінено під час передачі або зберігання, хеш-значення не збігатимуться, тому система може ідентифікувати таке втручання та допомогти запобігти атакам підробки даних.

Захист механізму консенсусу.

PBFT використовується як механізм консенсусу. Він може терпіти до 1/3 збоїв вузлів або зловмисної поведінки [24], забезпечуючи високий ступінь безпеки та інклюзивності. Це допомагає запобігти візантійським атакам під час процесу консенсусу та забезпечує узгодженість даних.

Децентралізоване зберігання.

Реальні параметри моделі зберігаються в IPFS. Завдяки характеристикам IPFS параметри моделі зберігаються розподіленим способом на кількох вузлах, що забезпечує децентралізоване зберігання. Цей метод зберігання не покладається на одну сутність і може підвищити здатність системи протистояти одноточковим атакам.

4.6 Експериментальні результати

Експериментальні результати в основному включають рівень точності та вартість зв’язку, а також час виконання для передачі файлу та реконструкції в моделях об’єднаного навчання.

Аналіз точності та вартості зв'язку.

Були проведені експерименти з двома розподілами даних MNIST і CIFAR-10, щоб перевірити ефективність навчання фреймворку BCFL на алгоритмічному рівні в FL. Алгоритм FL, який найчастіше використовується, FedAvg, порівнювався з BCFL за різними шаблонами обробки параметрів. Під час аналізу вартості зв’язку розглядалася лише передача файлів параметрів моделі між клієнтами, де комунікація вгорі відноситься до передачі файлів параметрів моделі в межах кластера, а зв’язок у низхідній частині стосується глобального файлу параметрів моделі, який передається вузлом агрегатора головний вузол кожного кластера. Результати експерименту набору даних MNIST Результати експерименту показано на рисунку 4.4.

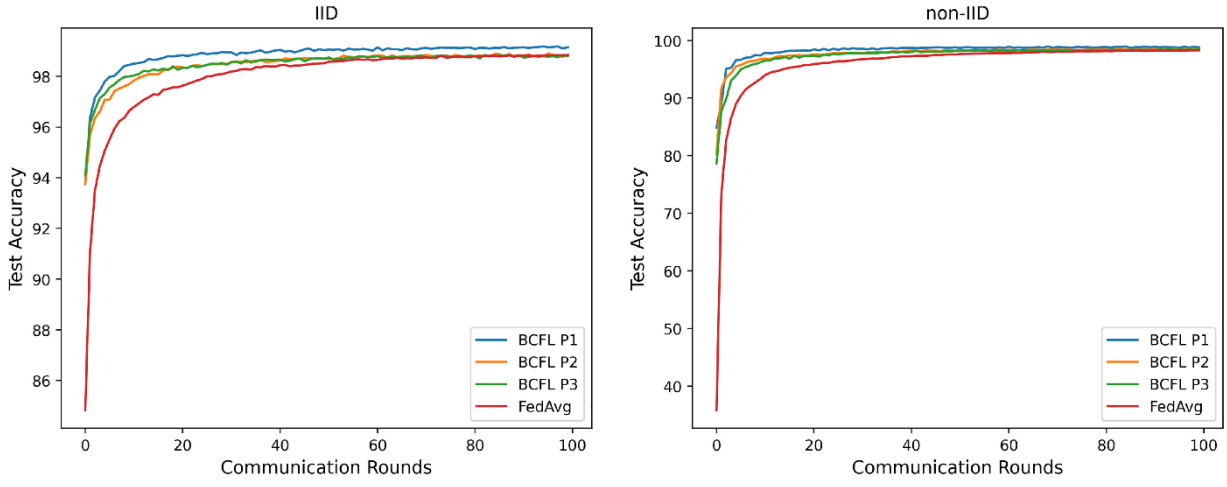

Рисунок 4.4 – Конвергенція BCFL і FedAvg на наборі даних MNIST.

Результати експерименту набору даних CIFAR-10. Результати показано на рисунку 4.5

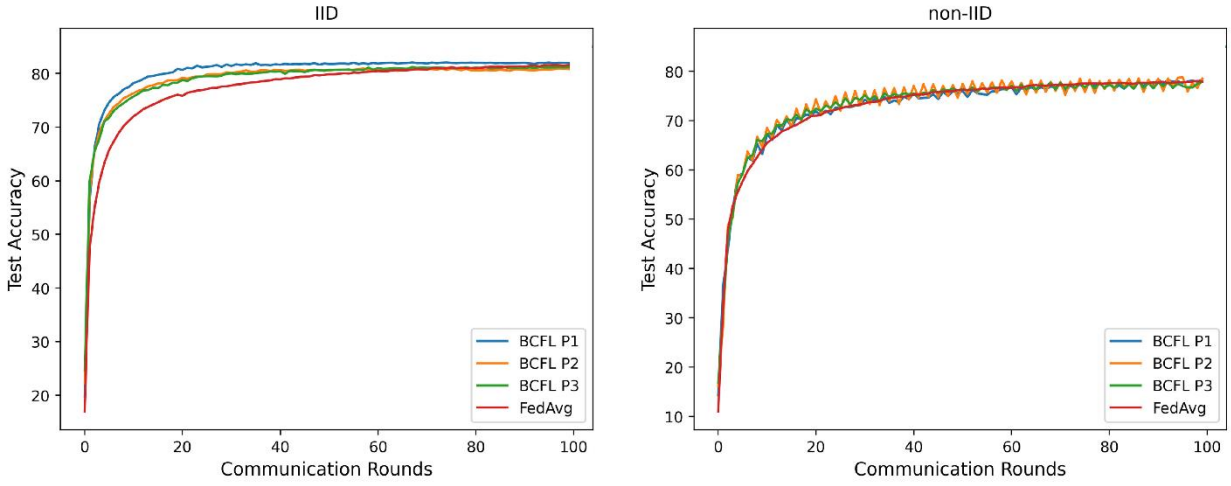

Рисунок 4.5 – Конвергенція BCFL і FedAvg на наборі даних CIFAR-10

Аналіз результатів експерименту

У цих двох експериментах точність і вартість зв'язку для різних алгоритмів (BCFL(p1), BCFL(p2), BCFL(p3) і FedAvg) у різних раундах зв'язку (5-й, 20-й і 100-й раунди) на MNIST і CIFAR -10 наборів даних було порівняно. Набір даних MNIST: Точність: як показано на рисунку 4.6, у всіх раундах BCFL(p1) мав вищу точність, ніж інші алгоритми в налаштуваннях IID і без IID. BCFL(p2) і BCFL(p3) мали дещо нижчу точність, ніж BCFL(p1) у тих самих раундах.

Зі збільшенням кількості раундів зв'язку точність усіх алгоритмів підвищувалася. У 100-му раунді точність BCFL(p1) досягла 99,15% і 98,84% у налаштуваннях IID і без IID відповідно, що було вищим порівняно з FedAvg. Вартість зв'язку: як показано в таблиці 3 , алгоритм BCFL продемонстрував значні переваги з точки зору вартості зв'язку. Порівняно з FedAvg, BCFL(p1), BCFL(p2) і BCFL(p3) значно знизили витрати на зв'язок, а BCFL(p3) мав найнижче значення.

У такій самій кількості раундів зв'язку витрати на зв'язок FedAvg були в 2, 2,76 і 4,17 рази вищими, ніж у BCFL(p1), BCFL(p2) і BCFL(p3), відповідно. З точки зору низхідного зв'язку вартість зв'язку алгоритму BCFL значно

зменшилася в 9, 12,8 і 19,9 разів порівняно з FedAvg. Таким чином, алгоритм BCFL перевершив FedAvg з точки зору вартості зв'язку.

Набір даних CIFAR-10: Точність: як показано на рисунку 4.7, подібно до набору даних MNIST, BCFL(p1) перевершив інші алгоритми на різних розподілах даних набору даних CIFAR-10. Коли було менше раундів зв'язку, алгоритм BCFL мав значно вищу точність, ніж алгоритм FedAvg.

Наприклад, у 5-му раунді точність алгоритму BCFL у трьох моделях за IID становила 73,06%, 71,15% та 71,01%, що вище, ніж FedAvg, тоді як за не-IID точність алгоритму BCFL становить 58,77 %, 58,98 % і 57,27 % відповідно, що також вище, ніж FedAvg. Зі збільшенням кількості раундів зв'язку точність FedAvg в основному була на рівні з BCFL. У 100-му раунді різниця в точності між BCFL(p1), BCFL(p2), BCFL(p3) і FedAvg під IID була невеликою, зі значеннями 81,98%, 80,82%, 81,20% і 81,44% відповідно. Загалом, за не-IID, BCFL мав дещо кращі показники, ніж FedAvg, з BCFL(p1), досягнувши точності 78,41%, вище, ніж FedAvg.

Вартість зв'язку, з точки зору вартості зв'язку, алгоритм BCFL мав певну перевагу над алгоритмом FedAvg на наборі даних CIFAR-10. Зокрема, після 100 раундів зв'язку FedAvg визначив загальну вартість зв'язку вгорі до 47,14 ГБ, що в 2, 2,29 і 3,29 рази більше, ніж у BCFL(p1), BCFL(p2) і BCFL(p3), відповідно.

У низхідній лінії зв'язку FedAvg також згенерував вартість зв'язку 47,14 ГБ, що в 10, 11,44 і 16,43 разів більше, ніж у BCFL(p1), BCFL(p2) і BCFL(p3), відповідно. Загалом загальна вартість зв'язку вгору та вниз за FedAvg була в 3,33, 3,81 та 5,47 разів вищою за три шаблони BCFL.

Аналіз витрати часу.

Окрім точності та вартості зв'язку, час виконання також є дуже важливим показником для оцінки продуктивності алгоритму BCFL. У структурі алгоритму BCFL файл параметрів моделі обробляється за допомогою розрідженості та квантування для отримання файлу моделі, який займає менше пам'яті, який потім передається іншим клієнтам. Після

отримання обробленого файлу параметрів моделі клієнт аналізує та відновлює його в придатні для використання параметри моделі для локальних оновлень. Під час цього процесу час, необхідний клієнтському вузлу для передачі параметрів моделі іншому вузлу, зменшується, але час, необхідний для обробки файлу параметрів моделі, збільшується. У цьому розділі аналізується час, необхідний для передачі файлів параметрів моделі між клієнтами та для обробки файлів, щоб додатково продемонструвати перевагу алгоритму BCFL у спілкуванні. У цьому експерименті час виконання було визначено за такою формулою:

$$T = T_{quantization} + T_{SC} + T_{transmission} + T_{restoration},$$

де $T_{quantization}$ час, необхідний для квантування параметрів моделі клієнтом, T_{SC} це час, необхідний для розрідження файлу параметрів моделі клієнтом, $T_{transmission}$це час, необхідний для передачі файлу параметрів моделі між клієнтами за допомогою мережевого зв'язку, і $T_{restoration}$ час, необхідний клієнту для відновлення файлу параметрів моделі після його отримання.

Далі в основному аналізується час виконання різних файлів моделі за різних методів обробки параметрів. Час виконання файлів параметрів моделі, згенерованих за допомогою алгоритму BCFL за різними шаблонами для наборів даних MNIST і CIFAR-10. Тут тип файлу p1 вказує на те, що алгоритм BCFL не виконував операції розрідженості або квантування у файлі моделі; тип файлу p2 вказує на те, що алгоритм виконав операції розрідженості та квантування файлу моделі; а тип файлу p3 вказує на те, що алгоритм додатково виконував операції стиснення. Через різні розміри згорткових нейронних мереж, які використовуються для навчання наборів даних, файли параметрів моделі, створені для набору даних MNIST, були меншими, ніж файли для набору даних CIFAR-10. Можна помітити, що операція квантування виконується дуже швидко, займаючи лише невелику частину загального часу

виконання. Час T_{SC} необхідний для операції розрідженості збільшився разом із кількістю параметрів моделі, але операція розрідженості зменшила розмір файлу моделі, тим самим зменшивши час передачі. Велика частина часу виконання відображається в $T_{transmission}$, який зазвичай збільшується, коли пропускна здатність мережі перевантажена. Цей експеримент імітував стабільну швидкість передачі даних у мережі 1 Мбіт/с.

Що стосується продуктивності файлів моделі MNIST, операції розрідженості та квантування зменшили загальний час виконання. Однак для файлів моделі CIFAR-10 через велику кількість параметрів моделі операція розрідженості вимагала певного часу та не зменшувала суттєво загальний час виконання за ідеальних умов мережі. Однак у поєднанні з операцією квантування розмір файлу моделі було значно зменшено, заощадивши багато часу на передачу. У порівнянні з CIFAR-10 (p2), CIFAR-10 (p3) зменшив загальний час виконання приблизно на 10 с.

ВИСНОВКИ

У відповідь на поточні виклики, з якими стикається федеративне навчання, ми пропонуємо структуру FL під назвою BCFL на основі блокчейну та кластерного навчання. У цьому документі пропонується алгоритм об’єднаного навчання, заснований на непарно-парному навчанні кластера. Розділивши клієнтів на кластери та застосувавши частково серіалізований метод навчання в кластерах, ми можемо прискорити конвергенцію моделі. Перед передачею параметрів файл моделі розріджується та квантується, щоб зменшити витрати на зв’язок і підвищити ефективність зв’язку FL. Архітектура BCFL більше не покладається на центральний сервер і використовує алгоритм балансування навантаження для планування клієнта, відповідального за агрегацію, у кожному раунді. Він представляє ланцюжок консорціуму для запису процесу FL і оптимізує проблему великих накладних витрат на зберігання ланцюга консорціуму шляхом поєднання з IPFS. Експерименти з наборами даних MNIST і CIFAR-10 демонструють, що запропонована структура відрізняється точністю та ефективністю зв’язку, а аналіз часу показує, що запропонована модельна схема обробки файлів є корисною для підвищення ефективності передачі FL. Незважаючи на те, що ця робота має певні досягнення в оптимізації продуктивності FL і проектуванні децентралізованої архітектури, все ще є багато напрямків для подальших досліджень і вдосконалень. У майбутніх дослідженнях можна розглянути наступні аспекти: Оптимізація стратегій поділу кластерів клієнтів: в алгоритмі, запропонованому в цьому документі, поділ кластерів клієнтів базується на простій стратегії випадкового групування.

Подальші дослідження можуть вивчити більш оптимальні стратегії розподілу клієнтських кластерів, враховуючи вимоги конкретних додатків і розподіл даних. Це може ще більше підвищити швидкість і точність конвергенції моделі;

- посилення механізмів безпеки: хоча архітектура BCFL, запропонована в цьому документі, демонструє високий рівень надійності та надійності, вона все ще має певні потенційні ризики безпеки та вразливості. Використання технології блокчейн у федеративному навчанні може зіткнутися з проблемами, пов'язаними зі зловживанням децентралізованими повноваженнями. Крім того, механізми консенсусу в технології блокчейн можуть бути чутливі до атак, таких як атака 51%, а смарт-контракти можуть містити вразливі місця в безпеці, якими можуть скористатися зловмисники. Майбутні дослідження мають бути зосереджені на впровадженні ефективних механізмів нагляду та управління, постійному вдосконаленні та оновленні механізмів консенсусу, а також на проведенні аудитів безпеки та виправленні вразливостей для розробки смарт-контрактів – усе це спрямовано на посилення механізмів безпеки системи;

- адаптація до неоднорідних пристроїв і обмеження ресурсів: у сценаріях практичного застосування клієнти, які беруть участь у тренінгах з об'єднаного навчання, можуть мати різні обчислювальні можливості та обмеження ресурсів, наприклад, пристрої IoT і периферійні обчислювальні системи. Майбутні дослідження можуть зосередитися на кількох аспектах, щоб підвищити адаптивність і практичність алгоритмів федеративного навчання. Ці аспекти включають зменшення витрат на зв'язок, що виникають через обмежену пропускну здатність пристрою, оптимізацію розподілу ресурсів і підвищення здатності боротися зі зловмисними учасниками;

- міждоменне застосування FL: ця робота головним чином зосереджена на фундаментальних питаннях і методах федеративного навчання без заглиблення в конкретні області застосування. Технологія FL має широкий потенціал застосування в таких сферах, як охорона здоров'я, фінанси та транспорт тощо. У майбутньому стане можливим розробляти міждоменні методи FL і інфраструктури додатків, адаптовані до характеристик і вимог різних областей.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

3. The Mobile Economy 2023. Available online: https://data.gsmaintelligence.com/research/research/research-2023/the-mobile-economy-2023.

4. Complete Guide to GDPR Compliance. Available online: https://gdpr.eu/ .

5. Yin, X.; Zhu, Y.; Hu, J. A comprehensive survey of privacy-preserving federated learning: A taxonomy, review, and future directions. ACM Comput. Surv. (CSUR) 2021, 54, 1–36.

6. Kouhizadeh, M.; Sarkis, J. Blockchain practices, potentials, and perspectives in greening supply chains. Sustainability 2018, 10, 3652.

7. McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; Arcas, B.A. Communication-efficient learning of deep networks from decentralized data. In Artificial Intelligence and Statistics; PMLR: New York, NY, USA, 2017; pp. 1273–1282.

8. Konecny, J.; McMahan, H.B.; Felix, X.Y.; Richtárik, P.; Suresh, A.T.; Bacon, D. Federated learning: Strategies for improving communication efficiency. arXiv 2016, arXiv:1610.05492.

9. Briggs, C.; Fan, Z.; Andras, P. Federated learning with hierarchical clustering of local updates to improve training on non-IID data. In Proceedings of the 2020 International Joint Conference on Neural Networks (IJCNN), Glasgow, UK, 19–24 July 2020; pp. 1–9.

10. Ляшенко О.С. Методи федеративного навчання для оптимізації розподілених систем. Ляшенко О.С., Нго За Фат., Назарова І.О // Проблеми інформатизації Тези доповідей одинадцятої міжнародної науково-технічної конференції (16 – 17 листопада 2023 року) Том 3. С. 102

11. Jhunjhunwala, D.; Gadhikar, A.; Joshi, G.; Eldar, Y.C. Adaptive quantization of model updates for communication-efficient federated learning. In

Proceedings of the ICASSP 2021—2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Toronto, ON, Canada, 6–11 June 2021; pp. 3110–3114.

12. Li, C.; Li, G.; Varshney, P.K. Communication-efficient federated learning based on compressed sensing. IEEE Internet Things J. 2021, 8, 15531–15541. [Google Scholar] [CrossRef]

13. Sattler, F.; Wiedemann, S.; Müller, K.R.; Samek, W. Robust and communication-efficient federated learning from non-iid data. IEEE Trans. Neural Netw. Learn. Syst. 2019, 31, 3400–3413.

14. Haddadpour, F.; Kamani, M.M.; Mokhtari, A.; Mahdavi, M. Federated learning with compression: Unified analysis and sharp guarantees. PMLR 2021, 130, 2350–2358.

15. Seol, M.; Kim, T. Performance Enhancement in Federated Learning by Reducing Class Imbalance of Non-IID Data. Sensors 2023, 23, 1152.

16. Zhao, L.; Huang, J. A distribution information sharing federated learning approach for medical image data. Complex Intell. Syst. 2023, 1–12.

17. Chen, F.; Wan, H.; Cai, H.; Cheng, G. Machine learning in/for blockchain: Future and challenges. Can. J. Stat. 2021, 49, 1364–1382.

18. Tsai, C.W.; Chen, Y.P.; Tang, T.C.; Luo, Y.C. An efficient parallel machine learning-based blockchain framework. Ict Express 2021, 7, 300–307.

19. Lo, S.K.; Liu, Y.; Lu, Q.; Wang, C.; Xu, X.; Paik, H.Y.; Zhu, L. Toward trustworthy ai: Blockchain-based architecture design for accountability and fairness of federated learning systems. IEEE Internet Things J. 2022, 10, 3276–3284.

20. Jiang, T.; Shen, G.; Guo, C.; Cui, Y.; Xie, B. BFLS: Blockchain and Federated Learning for sharing threat detection models as Cyber Threat Intelligence. Comput. Netw. 2023, 224, 109604.

21. Ляшенко О.С., Бохан І.А., Крюкова І.В. Метод федеративного навчання на основі блокчейн. Сучасний стан наукових досліджень і технологій в промисловості. 2(27). 2024. (в редакції)

22. Zhang, W.; Wang, Z.; Li, X. Blockchain-based decentralized federated

transfer learning methodology for collaborative machinery fault diagnosis. Reliab. Eng. Syst. Saf. 2023, 229, 108885.

23. Zhang, Y.; Wu, Y.; Li, T.; Zhou, H.; Chen, Y. Vertical Federated Learning Based on Consortium Blockchain for Data Sharing in Mobile Edge Computing. CMES-Comput. Model. Eng. Sci. 2023, 137, 345–361.

24. Fu, X.; Peng, R.; Yuan, W.; Ding, T.; Zhang, Z.; Yu, P.; Kadoch, M. Federated learning-based resource management with blockchain trust assurance in smart IoT. Electronics 2023, 12, 1034.

25. Li, Y.; Liu, Z.; Huang, Y.; Xu, P. FedOES: An Efficient Federated Learning Approach. In Proceedings of the 2023 3rd International Conference on Neural Networks, Information and Communication Engineering (NNICE), Guangzhou, China, 24–26 February 2023; pp. 135–139.

26. Davis, T.A.; Hu, Y. The University of Florida sparse matrix collection. ACM Trans. Math. Softw. 2011, 38, 1–25.

27. Zheng, Z.; Xie, S.; Dai, H.; Chen, X.; Wang, H. An overview of blockchain technology: Architecture, consensus, and future trends. In Proceedings of the 2017 IEEE International Congress on Big Data (BigData Congress), Honolulu, HI, USA, 25–30 June 2017; pp. 557–564.

28. Li, X.; Jiang, P.; Chen, T.; Luo, X.; Wen, Q. A survey on the security of blockchain systems. Future Gener. Comput. Syst. 2020, 107, 841–853.

29. Lim, W.Y.B.; Luong, N.C.; Hoang, D.T.; Jiao, Y.; Liang, Y.C.; Yang, Q.; Niyato, D.; Miao, C. Federated learning in mobile edge networks: A comprehensive survey. IEEE Commun. Surv. Tutor. 2020, 22, 2031–2063.