



DOI 10.28925/2663-4023.2026.35.1409

УДК 004

Богаченко Сергій Володимирович

аспірант кафедри кібербезпеки та програмного забезпечення

Центральноукраїнський національний технічний університет, Кропивницький, Україна

ORCID: 0009-0007-5787-0559

sergeybohachenkodevs@gmail.com

Смірнов Олексій Анатолійович

д.т.н., професор, завідувач кафедри кібербезпеки та програмного забезпечення

Центральноукраїнський національний технічний університет, Кропивницький, Україна

ORCID: 0000-0001-9543-874X

dr.smirnova@gmail.com

Козірова Наталія Леонідівна

викладач кафедри кібербезпеки та програмного забезпечення

Центральноукраїнський національний технічний університет, Кропивницький, Україна

ORCID: 0009-0005-8753-5132

natalidonchenko23@gmail.com

Смірнова Тетяна Віталіївна

к.т.н., доцент кафедри кібербезпеки та програмного забезпечення

Центрально український національний технічний університет, Кропивницький, Україна

ORCID: 0000-0001-6896-0612

sm.tetyana@gmail.com

Константинова Лілія Володимирівна

викладач кафедри кібербезпеки та програмного забезпечення

Центральноукраїнський національний технічний університет, Кропивницький, Україна

ORCID: 0000-0002-3305-2427

liliyashel1976@gmail.com

Смірнов Сергій Анатолійович

кандидат технічних наук, доцент, доцент кафедри кібербезпеки та програмного забезпечення

Центральноукраїнський національний технічний університет, Кропивницький, Україна

ORCID: 0000-0002-7649-7442

smirnov.ser.81@gmail.com

**МЕТОДИ ЗМЕНШЕННЯ ОБЧИСЛЮВАЛЬНОЇ СКЛАДНОСТІ ЛЕГКОВАГОВИХ НЕЙРОННИХ
МЕРЕЖ ІЗ ВИКОРИСТАННЯМ БАЗ ДАНИХ ДЛЯ МОБІЛЬНИХ ТА EDGE-ПЛАТФОРМ**

Анотація. Важливим напрямом оптимізації, окрім використання легковагових архітектур, є застосування методів зменшення обчислювальної складності вже наявних моделей машинного навчання. Для цього у даній роботі запропоновано комбінацію методів квантування, прорідження, дистиляція знань, низькорангове розкладання, оптимізація графа обчислень та fusion операторів. Важливим напрямом оптимізації, окрім використання легковагових архітектур, є застосування методів зменшення обчислювальної складності з використанням баз даних для збереження та ефективної вибірки стиснутих параметрів моделей машинного навчання. Кожен із перелічених підходів має власні переваги, недоліки та особливості практичного використання. Зазначені методи оптимізації можуть реалізовуватися як автономно, так і в сукупності та взаємодії один з одним. Зокрема, у моделях машинного навчання нерідко поєднують квантування з проріджуванням або застосовують дистиляцію знань разом із подальшою оптимізацією графа виконання. Така комбінована стратегія забезпечує вищу ефективність порівняно із застосуванням поодиноких підходів. Показано, що використання цих методів дозволяє суттєво зменшити розмір моделі машинного навчання, скоротити витрати пам'яті та прискорити виконання інференсу. Разом із цим встановлено, що більшість таких підходів формує статичну конфігурацію моделі машинного навчання до початку її використання. Після завершення оптимізації структура та поведінка моделі машинного навчання практично не змінюються. Таким чином визначено, що



ефективність оптимізації може змінюватися залежно від поточного стану пристрою та умов виконання.

Ключові слова: обмеження ресурсів, штучний інтелект, машинне навчання, бази даних, мобільні платформи, edge-платформи, розробка програмного забезпечення, комп'ютерні мережі

ВСТУП

Постановка завдання дослідження

Важливим напрямом оптимізації вже наявних моделей машинного навчання, окрім використання легковагових архітектур, є застосування методів зменшення обчислювальної складності. Навіть якщо архітектура спочатку створювалася для мобільних або edge-платформ, вона все одно може вимагати додаткової оптимізації перед практичним використанням. Це пояснюється тим, що характеристики реальних пристроїв можуть суттєво відрізнятися, а доступні ресурси не є постійними.

Постановка проблеми. Методи оптимізації дозволяють зменшувати обсяг пам'яті, кількість обчислень, затримки інференсу та рівень енергоспоживання без необхідності повного перепроєктування архітектури моделі машинного навчання. Особливо важливими такі підходи стають у випадках, коли вже існує навчена модель машинного навчання і виникає потреба адаптувати її до обмежених ресурсів конкретної платформи.

До таких методів належать квантування, прорідження, дистилляція знань, низькорангове розкладання, оптимізація графа обчислень та fusion операторів. Кожен із перелічених підходів має власні переваги, недоліки та особливості практичного використання.

У більшості випадків зазначені методи можуть застосовуватися окремо або комбінуватися між собою. Наприклад, модель машинного навчання може одночасно використовувати квантування та прорідження або поєднувати дистилляцію знань із подальшою оптимізацією графа виконання. Такий комбінований підхід дозволяє отримувати кращі результати порівняно з використанням лише одного способу оптимізації.

У даному дослідженні необхідно розв'язати завдання дослідження та комбінування наступних методів:

1. Методів квантування.
2. Методів прорідження та структурної оптимізації.
3. Методів дистилляції знань.

Аналіз останніх досліджень і публікацій. Дослідження Edge AI та периферійних обчислень охоплює широке коло технологічних рішень. Зокрема, у праці [1] розкрито сутність концепції Edge AI як застосування алгоритмів штучного інтелекту безпосередньо на кінцевих пристроях (смартфонах, відеокамерах, пристроях IoT) без залучення хмарної інфраструктури, що гарантує миттєву локальну обробку даних та ухвалення рішень. Подібний підхід перенесення обчислювальних потужностей безпосередньо до джерел інформації деталізовано в роботі [2], де наголошується на критичній важливості мінімізації затримок для сервісів реального часу. Принципи функціонування мобільних периферійних обчислень (Mobile Edge Computing, MEC) описано в публікації [3], а практичні рекомендації щодо підвищення їхньої продуктивності узагальнено в [4]. Окрему увагу в науковому співтоваристві приділено оптимізації архітектури нейронних мереж для обробки даних на пристроях. Дослідження [5, 6] присвячені розробці полегшених мережевих архітектур (Lightweight Network Architecture) та оптимізації моделей глибокого навчання. Застосування моделей машинного навчання з відкритим кодом для задач комп'ютерного зору на кінцевих пристроях розглянуто у праці [7]. Нарешті, питанню ефективного масштабування та побудови згорткових нейронних мереж присвячені праці [8, 9] (у контексті архітектури MobileNetV2) та [10] (на прикладі підходу EfficientNet). Праці [11-13] присвячені особливостям побудови динамічних і адаптивних механізмів для edge-систем. Методи інтеграції векторних баз даних у моделі генеративного штучного інтелекту (зокрема для задач семантичного пошуку, RAG та збереження векторних ембедингів), що дозволяє підвищити точність і швидкість обробки великих масивів знань досліджені у [14]. Питання обмеження ресурсів мобільних та edge-платформ при розробці програмного забезпечення для реалізації та оптимізації моделей машинного навчання розглянуті у [15], де визначено що розробка програмного забезпечення машинного навчання для мобільних та edge-платформ вимагає комплексного підходу, що поєднує системну оптимізацію архітектур моделей і використання спеціалізованих фреймворків для ефективного подолання апаратних обмежень та досягнення оптимального балансу між точністю, швидкістю і енергоефективністю. Робота [16] пропонує метод дистилляції архітектури для проектування компактної згорткової нейронної мережі (LightweightNet), що



дозволяє прискорити обчислення та зменшити обсяг моделі для використання на пристроях із обмеженими ресурсами. У статті [17] розроблено схему квантування та тренування нейромереж, яка дозволяє перевести обчислення з плаваючою комою у формат 8-бітних цілих чисел (integer-arithmetic-only), суттєво зменшуючи затримку й енергоспоживання під час виконання інференсу на мобільних пристроях. Праця [18] висвітлює концепцію «дистиляції знань» (Knowledge Distillation), за якої компактна та ефективна «модель-учень» тренується відтворювати «м'які» ймовірнісні розподіли складного ансамблю або великої «моделі-вчителя», зберігаючи високу точність при значно менших обчислювальних витратах.

Метою статті є оцінка обмеженості ресурсів мобільних і edge систем, пошук шляхів зменшення обчислювального навантаження полегшених нейромережових архітектур із використанням відповідних баз даних, а також визначення пріоритетних напрямків подальших досліджень.

Об'єктом дослідження є процес машинного навчання.

Предметом дослідження є зменшення обчислювальної складності легковагових нейронних мереж із використанням баз даних для мобільних та edge-платформ.

ТЕОРЕТИЧНІ ОСНОВИ ДОСЛІДЖЕННЯ

Одним із найпоширеніших методів є квантування. Квантування полягає у зменшенні розрядності представлення ваг і активацій. Наприклад, замість float32 можуть використовуватися int8 або float16. Це дозволяє зменшити розмір моделі машинного навчання, прискорити виконання на підтримуваному апаратному забезпеченні та знизити енергоспоживання [17].

Основна ідея квантування полягає у переході до менш точного, але більш компактного числового представлення. Менша кількість бітів дозволяє зменшити обсяг пам'яті та скоротити кількість операцій при виконанні моделі машинного навчання.

Сучасні підходи до квантування можуть використовувати різні стратегії. Деякі методи виконують квантування після завершення навчання моделі машинного навчання, тоді як інші інтегрують його безпосередньо у процес тренування. Другий підхід зазвичай дозволяє краще зберігати точність, однак потребує додаткових обчислювальних витрат під час навчання.

Водночас квантування може призводити до втрати точності, особливо якщо модель машинного навчання чутлива до чисельних похибок. Це особливо помітно для архітектур, де навіть незначні зміни ваг можуть суттєво впливати на результат прогнозування.

При квантуванні (quantization) ваги переводяться в нижчу точність (наприклад, INT8 чи INT4). Таблиці квантування (lookup tables) та квантовані параметри часто зберігають у спеціалізованих вбудованих БД (наприклад, SQLite, DuckDB, LMDB) для швидкого зчитування на edge-пристроях.

Іншим важливим методом є прорідження. Прорідження передбачає видалення малозначущих ваг, фільтрів або каналів. Основна ідея цього підходу полягає в тому, що не всі параметри моделі машинного навчання однаково впливають на кінцевий результат.

Після завершення навчання частина параметрів може мати дуже малий внесок у роботу мережі. Видалення таких компонентів дозволяє скоротити розмір моделі машинного навчання та зменшити кількість операцій.

Прорідження може бути неструктурованим або структурованим. Неструктуроване прорідження дозволяє сильно зменшити кількість параметрів, але не завжди добре прискорює інференс на мобільному hardware. Це пояснюється тим, що хоча кількість ваг зменшується, структура обчислень залишається складною для ефективного виконання.

Структуроване прорідження краще узгоджується з реальними апаратними платформами, оскільки видаляє цілі канали, блоки або шари. Завдяки цьому оптимізація стає більш сумісною із сучасними обчислювальними архітектурами.

При прорідженні (pruning) більшість вагових коефіцієнтів стають нульовими. Для їх ефективного збереження використовуються розріджені бази даних або векторні індекси (Sparse Matrix Formats/Vector DBs).

Ще одним важливим методом є дистиляція знань [18]. Дистиляція знань полягає у навчанні меншої моделі машинного навчання на основі більшої моделі-вчителя. При цьому компактна модель машинного навчання отримує не лише правильні відповіді, але й інформацію про поведінку складнішої моделі машинного навчання [16].

Такий підхід дозволяє компактній моделі машинного навчання краще відтворювати поведінку складнішої моделі машинного навчання. Це особливо корисно для мобільних задач, де потрібно отримати невелику модель машинного навчання з прийнятною точністю.



На практиці дистиляція дозволяє отримувати значно кращі результати порівняно із простим навчанням невеликої моделі машинного навчання з нуля. Завдяки використанню знань моделі-вчителя процес навчання стає більш інформативним.

Для навчання меншої ("молодшої") моделі від великої ("вчителя") часто створюють великі кешовані датасети передбачень або ембедингів. Для збереження, швидкої вибірки та завантаження цих знань застосовують бази даних ембедингів (Vector Databases) або реляційні/NoSQL БД.

Окремо варто згадати низькорангове розкладання. Цей метод базується на тому, що вагові матриці нейронної мережі можуть бути представлені у вигляді добутку декількох менших матриць. У результаті кількість параметрів і складність обчислень можуть бути суттєво зменшені. Особливо ефективним такий підхід може бути для великих шарів із значною кількістю параметрів.

Ще одним напрямом є оптимізація графа обчислень. Оптимізація графа включає видалення зайвих операцій, об'єднання операторів, попереднє обчислення сталих значень та адаптацію моделі машинного навчання до конкретного runtime. Подібні методи часто реалізуються у TensorFlow Lite, ONNX Runtime, Core ML та інших інструментах мобільного інференсу [5-10]. У багатьох випадках користувач може навіть не бачити подібних оптимізацій, оскільки вони виконуються автоматично.

Додатково використовується operator fusion. Його основна ідея полягає в об'єднанні декількох послідовних операцій у єдиний блок виконання. Це дозволяє скоротити кількість звернень до пам'яті та зменшити накладні витрати. Особливо важливим operator fusion є для мобільних платформ, де доступ до пам'яті часто стає вузьким місцем продуктивності.

Оптимізовані графи обчислень під конкретні процесори (NPU/GPU/CPU) або бази заздалегідь скомпільованих ядер (kernels) можуть зберігатися в вбудованій базі даних конфігурацій/кешу (наприклад, у вигляді key-value store).

Однак більшість зазначених методів мають спільну проблему: вони переважно виконуються до запуску моделі машинного навчання і формують статичну оптимізовану версію. Після завершення оптимізації структура моделі машинного навчання зазвичай більше не змінюється.

У реальних умовах мобільного середовища цього може бути недостатньо, оскільки стан пристрою змінюється вже під час виконання. Рівень заряду батареї, температура, навантаження CPU та доступність пам'яті можуть змінюватися навіть у межах одного сеансу роботи. У результаті оптимізація, яка була ефективною в одному стані системи, може виявитися менш корисною в іншому. Саме ця проблема поступово підвела сучасні дослідження до використання адаптивних підходів.

Тому виникає потреба у методах, які не лише зменшують складність моделі машинного навчання заздалегідь, але й дозволяють адаптувати процес інференсу в реальному часі.

Таблиця 1.2

Порівняння методів оптимізації моделей машинного навчання

Метод	Переваги	Недоліки
Квантування	Зменшує розмір моделі машинного навчання та енергоспоживання	Можлива втрата точності
Прорідження	Скорочує кількість параметрів	Не завжди прискорює інференс
Дистиляція знань	Покращує компактні моделі машинного навчання	Потребує навчання моделі-вчителя
Fusion операторів	Зменшує витрати виконання	Залежить від runtime
Оптимізація графа	Скорочує зайві операції	Прив'язана до фреймворку

Можна помітити, що більшість методів спрямована на попередню оптимізацію моделі машинного навчання до запуску. Під час виконання моделі машинного навчання можливість адаптації таких підходів є обмеженою.

Слід зазначити, методи зменшення обчислювальної складності дозволяють суттєво скорочувати витрати пам'яті, кількість обчислень та енергоспоживання моделей машинного навчання. Проте більшість із них реалізують оптимізацію на етапі підготовки моделі машинного навчання та формують статичну конфігурацію виконання. Це створює обмеження для використання у динамічному мобільному середовищі та формує потребу у розробці адаптивних методів інференсу.

Методи квантування моделей машинного навчання

Квантування є одним із найбільш практичних та поширених методів оптимізації моделей машинного навчання для мобільних та edge-платформ. Його основна ідея полягає у зменшенні точності числового представлення параметрів і проміжних значень моделі машинного навчання. Завдяки цьому



можна зменшити розмір моделі машинного навчання, скоротити кількість обчислень та знизити витрати енергії під час виконання інференсу [17].

Проблема високої точності числового представлення стала особливо актуальною зі збільшенням розміру сучасних нейронних мереж. Більшість моделей машинного навчання під час навчання використовують формат float32, який забезпечує високу точність обчислень, однак потребує значних ресурсів пам'яті та обчислювальної потужності. Для мобільних платформ такі вимоги можуть бути надмірними. Основна перевага квантування полягає не лише у зменшенні розміру файлу моделі машинного навчання. Менша розрядність також зменшує кількість переданих даних між пам'яттю та обчислювальними блоками. У мобільних пристроях доступ до пам'яті часто є одним із найбільш енерговитратних процесів, тому скорочення обсягів переданих даних може суттєво впливати на загальне енергоспоживання.

Найчастіше у мобільному інференсі використовується int8-квантування або float16-представлення. Формат int8 дає змогу суттєво зменшити розмір моделі машинного навчання та знизити витрати пам'яті. Крім того, сучасні мобільні процесори та нейронні прискорювачі часто мають апаратну підтримку операцій int8. Формат float16 є дещо простішим у застосуванні, оскільки він дозволяє частково зберігати переваги роботи з числами з плаваючою комою. У деяких випадках це дозволяє уникати значного погіршення точності. Проте ефективність використання float16 значною мірою залежить від наявності відповідної підтримки на рівні апаратного забезпечення.

Квантування може бути післянавчальним або таким, що враховується під час навчання. Післянавчальне квантування є простішим, оскільки застосовується до вже навченої моделі машинного навчання. Воно не потребує повторного навчання та може бути реалізоване досить швидко.

Разом із цим післянавчальний підхід може спричинити більшу втрату точності. Це особливо характерно для моделей машинного навчання, які чутливі до змін числового представлення параметрів.

Іншим підходом є Quantization-aware training. У цьому випадку ефекти квантування враховуються ще під час навчання моделі машинного навчання. Мережа поступово адаптується до майбутніх обмежень розрядності та вчиться компенсувати можливі похибки. Подібний підхід зазвичай забезпечує кращу якість моделі машинного навчання після оптимізації. Однак він потребує додаткового навчання або донавчання моделі машинного навчання, що збільшує складність розробки.

Недоліком квантування є те, що воно не завжди однаково ефективне для різних архітектур. Деякі моделі машинного навчання добре переносять зменшення розрядності, тоді як інші можуть суттєво втрачати точність. Крім того, енергетичний вигравш можливий лише за умови, що цільова платформа ефективно підтримує відповідний формат обчислень. Якщо така підтримка відсутня, очікуваний ефект може бути значно меншим.

Таким чином, квантування дозволяє суттєво скорочувати обсяг пам'яті та зменшувати обчислювальну складність моделей машинного навчання. Разом із цим ефективність підходу залежить від типу архітектури та підтримки апаратного забезпечення. У більшості випадків квантування формує статичну конфігурацію моделі машинного навчання та не враховує зміни ресурсів під час виконання.

Методи прорідження та структурної оптимізації

Прорідження моделей машинного навчання спрямоване на видалення надлишкових параметрів або структурних елементів нейронної мережі. У багатьох сучасних моделях частина ваг або фільтрів має відносно невеликий вплив на кінцевий результат, тому їх можна видалити з мінімальною втратою точності.

Дослідження показують, що нейронні мережі часто містять значний рівень надлишковості. Під час навчання модель машинного навчання може формувати велику кількість параметрів, які фактично дублюють функції один одного або мають незначний внесок у результат прогнозування. Основна ідея прорідження полягає у виявленні таких елементів та їх подальшому видаленні. Це дозволяє скоротити кількість параметрів та потенційно зменшити складність виконання.

Неструктуроване прорідження видаляє окремі ваги. Воно може значно зменшити кількість параметрів, іноді навіть у декілька разів. Проте на практиці такий підхід не завжди прискорює виконання на мобільних пристроях. Причина полягає в тому, що мобільні процесори та прискорювачі часто неефективно працюють з розрідженими структурами даних.

Після видалення ваг утворюється нерівномірна структура обчислень, яка не завжди добре узгоджується з особливостями сучасних процесорів. Структуроване прорідження є більш практичним для мобільного інференсу. Воно видаляє цілі канали, фільтри або блоки мережі. Такий підхід краще узгоджується з апаратними платформами, оскільки зменшує фактичний обсяг операцій і може реально знижувати latency.

Структуроване прорідження також простіше інтегрується у мобільні фреймворки та runtime-системи. Саме тому останніми роками воно активно використовується у практичних рішеннях. Разом із



цим надмірне прорідження може призводити до різкого падіння точності. Видалення занадто великої кількості структурних елементів може погіршити якість представлення ознак.

Тому важливо знаходити баланс між ступенем стиснення та якістю моделі машинного навчання. У контексті енергозберігаючого машинного навчання прорідження варто розглядати не лише як метод зменшення розміру моделі машинного навчання, але й як один з елементів ширшої системи адаптації.

Варто зазначити, прорідження дозволяє зменшити кількість параметрів і скоротити обчислювальні витрати моделей машинного навчання. Для мобільних платформ найбільш практичним є структурований підхід, оскільки він краще узгоджується з можливостями апаратного забезпечення. Разом із цим прорідження залишається статичним методом оптимізації та не враховує поточний стан системи.

Методи дистилляції знань

Дистилляція знань є підходом, при якому компактна модель машинного навчання навчається відтворювати поведінку більшої моделі машинного навчання. Велика модель машинного навчання, яку називають моделлю-вчителем, формує більш інформативні цільові значення, ніж звичайні мітки класів [16, 18].

На відміну від класичного навчання, де модель машинного навчання отримує лише правильний клас, під час дистилляції використовується додаткова інформація про розподіл ймовірностей між класами. Це дозволяє компактній моделі машинного навчання отримати більше інформації про структуру задачі.

Завдяки цьому менша модель машинного навчання може досягати вищої точності, ніж при стандартному навчанні. Для мобільних та edge-систем цей підхід є особливо корисним, оскільки дозволяє отримати компактну модель машинного навчання без повної втрати якості. Наприклад, велика модель машинного навчання може бути навчена у хмарному середовищі, де доступні потужні обчислювальні ресурси. Після цього її знання передаються меншій моделі машинного навчання, яка виконується вже на мобільному пристрої.

Подібний підхід дозволяє використовувати переваги великих моделей машинного навчання навіть у середовищах із жорсткими обмеженнями ресурсів. Останніми роками дистилляція знань почала використовуватися не лише для згорткових мереж, але й для трансформерних архітектур та великих мовних моделей машинного навчання.

Разом із цим метод має певні обмеження. Недоліком дистилляції є залежність від якості моделі-вчителя та складність налаштування процесу навчання. Крім того, вибір температурного параметра, функції втрат та інших характеристик може суттєво впливати на кінцевий результат.

Ще одним обмеженням є те, що дистилляція не вирішує проблему динамічності ресурсів під час виконання. Вона допомагає створити кращу компактну модель машинного навчання, але сама модель машинного навчання після цього зазвичай залишається статичною. Тобто після завершення навчання поведінка моделі машинного навчання практично не змінюється незалежно від поточного стану мобільного пристрою.

Таким чином, дистилляція знань дозволяє отримувати компактні моделі машинного навчання з вищою точністю порівняно зі стандартним навчанням. Метод є ефективним для мобільних та edge-систем, однак не враховує зміни умов виконання під час роботи моделі машинного навчання. Це створює необхідність доповнення подібних підходів адаптивними механізмами інференсу.

ВИСНОВКИ ТА ПЕРСПЕКТИВИ ПОДАЛЬШИХ ДОСЛІДЖЕНЬ

У даній роботі досліджено сучасні методи зменшення обчислювальної складності моделей машинного навчання. Розглянуто підходи квантування, прорідження, дистилляції знань, оптимізації графа обчислень та структурної оптимізації моделей машинного навчання. Показано, що використання цих методів дозволяє суттєво зменшити розмір моделі машинного навчання, скоротити витрати пам'яті та прискорити виконання інференсу. Разом із цим встановлено, що більшість таких підходів формує статичну конфігурацію моделі машинного навчання до початку її використання. Після завершення оптимізації структура та поведінка моделі машинного навчання практично не змінюються. У результаті ефективність оптимізації може змінюватися залежно від поточного стану пристрою та умов виконання. Таким чином за результатами проведеного дослідження визначено перспективний напрям подальших досліджень, який полягає у розробленні адаптивного ресурсно-орієнтованого підходу до керування процесом інференсу моделей машинного навчання.



СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ha, R. (2025, August 25). *Що take Edge AI? Переваги та виклики*. INVIAI.
2. ProIT. (n.d.). *Як edge computing змінює обчислювальні технології*.
3. European Telecommunications Standards Institute. (2019). *Multi-access Edge Computing (MEC); framework and reference architecture* (ETSI GS MEC 003 V2.1.1).
4. Google. (2026, May 28). *Performance best practices*. Google AI Edge.
5. Mao, Z. (2025). Lightweight network architecture—Optimization of deep learning models. *Applied and Computational Engineering*, 175, 178–185. <https://doi.org/10.54254/2755-2721/2025.AST26509>
6. Liu, L., & Xu, Z. (2025). Optimizing lightweight neural networks for efficient mobile edge computing. *Scientific Reports*, 15, Article 22056. <https://doi.org/10.1038/s41598-025-04652-7>
7. Howard, A. G., & Zhu, M. (2017, June 14). *MobileNets: Open-source models for efficient on-device vision*. Google Research.
8. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2018). MobileNetV2: Inverted residuals and linear bottlenecks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 4510–4520). IEEE. <https://doi.org/10.1109/CVPR.2018.00474>
9. Ma, N., Zhang, X., Zheng, H.-T., & Sun, J. (2018). ShuffleNet V2: Practical guidelines for efficient CNN architecture design. In V. Ferrari, M. Hebert, C. Sminchisescu, & Y. Weiss (Eds.), *Computer vision—ECCV 2018* (Lecture Notes in Computer Science, Vol. 11218, pp. 116–131). Springer. https://doi.org/10.1007/978-3-030-01264-9_8
10. Tan, M., & Le, Q. V. (2019). EfficientNet: Rethinking model scaling for convolutional neural networks. In K. Chaudhuri and R. Salakhutdinov (Eds.), *Proceedings of the 36th International Conference on Machine Learning* (Vol. 97, pp. 6105–6114). PMLR. <https://proceedings.mlr.press/v97/tan19a.html>
11. Kuznetsov, O., Frontoni, E., Kuznetsova, Y., Chevardin, V., & Smirnov, O. (2026). Architectural foundations for adaptive security in edge computing systems. In A. L. Imoize, F. Xhafa, M. S. Obaidat, and H. H. Song (Eds.), *Cybersecurity defensive walls in edge computing* (pp. 21–61). Academic Press. <https://doi.org/10.1016/B978-0-443-34109-0.00015-2>
12. Kuznetsov, O., Atzeni, G., Arnesano, M., Randieri, C., & Smirnov, O. (2025). Secure IoT-based smart wheelchair system: From implementation to security enhancement strategy. In A. L. Imoize, W. Montlouis, S. I. Popoola, and M. Hammoudeh (Eds.), *Security and privacy of cyber-physical systems: Emerging trends, technologies, and applications* (pp. 225–257). River Publishers. <https://doi.org/10.1201/9788743801306-8>
13. Kuznetsov, O., Smirnov, O., Kuznetsova, T., Shaikhanova, A., & Svatowsky, I. (2025). Privacy-utility trade-offs in IoT networks: A comparative analysis of differential privacy mechanisms for sensor data aggregation. In A. L. Imoize, W. Montlouis, S. I. Popoola, and M. Hammoudeh (Eds.), *Security and privacy of cyber-physical systems: Emerging trends, technologies, and applications* (pp. 589–622). River Publishers. <https://doi.org/10.1201/9788743801306-17>
14. Смірнов, О., Ткачук, Р., Козірова, Н., Константинова, Л., Коноплицька-Слободенюк, О., Якименко, Н., & Смірнов, С. (2026). Дослідження застосування векторних баз даних у генеративному штучному інтелекті. *Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка»*, 1(33), 667–682. <https://doi.org/10.28925/2663-4023.2026.33.1248>
15. Богаченко, С., Смірнов, О., Буравченко, К., Смірнова, Т., Коноплицька-Слободенюк, О., Якименко, Н., & Смірнов, С. (2026). Обмеження ресурсів мобільних та edge-платформ при розробці програмного забезпечення для реалізації та оптимізації моделей машинного навчання. *Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка»*, 2(34), 677–686. <https://doi.org/10.28925/2663-4023.2026.34.1360>
16. Xu, T.-B., Yang, P., Zhang, X.-Y., & Liu, C.-L. (2019). LightweightNet: Toward fast and lightweight convolutional neural networks via architecture distillation. *Pattern Recognition*, 88, 272–284. <https://doi.org/10.1016/j.patcog.2018.10.029>
17. Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H., & Kalenichenko, D. (2018). Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 2704–2713). IEEE. <https://doi.org/10.1109/CVPR.2018.00286>
18. Hinton, G., Vinyals, O., & Dean, J. (2015). *Distilling the knowledge in a neural network* [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.1503.02531>

**Serhii Bohachenko**

PhD graduate student of the Cybersecurity & Software Academic Department
Central Ukrainian National Technical University, Kropyvnytskyi, Ukraine
ORCID: 0009-0007-5787-0559
sergeybohachenkodevs@gmail.com

Oleksii Smirnov

Doctor of Technical Sciences, Professor,
Head of Cybersecurity & Software Academic Department
Central Ukrainian National Technical University, Kropyvnytskyi, Ukraine
ORCID: 0000-0001-9543-874X
dr.smirnovoa@gmail.com

Nataliia Kozirova

Lecturer of Cybersecurity & Software Academic Department
Central Ukrainian National Technical University, Kropyvnytskyi, Ukraine
ORCID: 0009-0005-8753-5132
natalidonchenko23@gmail.com

Tetiana Smirnova

PhD of Science (Engineering),
Associate Professor of Cybersecurity & Software Academic Department
Central Ukrainian National Technical University, Kropyvnytskyi, Ukraine
ORCID: 0000-0001-6896-0612
sm.tetyana@gmail.com

Liliia Konstantynova

Lecturer of Cybersecurity & Software Academic Department
Central Ukrainian National Technical University, Kropyvnytskyi, Ukraine
ORCID: 0000-0002-3305-2427
liliyashel1976@gmail.com

Serhii Smirnov

Candidate of Science (Engineering), Associate Professor,
Associate Professor of Cybersecurity & Software Academic Department
Central Ukrainian National Technical University, Kropyvnytskyi, Ukraine
ORCID: 0000-0002-7649-7442
smirnov.ser.81@gmail.com

METHODS FOR REDUCING THE COMPUTATIONAL COMPLEXITY OF LIGHTWEIGHT NEURAL NETWORKS USING DATABASES FOR MOBILE AND EDGE PLATFORMS

Abstract. In addition to the use of lightweight architectures, a key optimization direction involves applying methods to reduce the computational complexity of existing machine learning models. To this end, this paper proposes a combination of techniques: quantization, pruning, knowledge distillation, low-rank decomposition, computation graph optimization, and operator fusion. Another important optimization avenue-beyond lightweight architectures-involves reducing computational complexity by utilizing databases to store and efficiently retrieve compressed machine learning model parameters. Each of these approaches has its own advantages, disadvantages, and practical implementation nuances. These optimization methods can be implemented either independently or in combination and interaction with one another. For instance, machine learning models often combine quantization with pruning or employ knowledge distillation followed by execution graph optimization. Such a combined strategy yields greater efficiency than the use of isolated approaches. It has been demonstrated that employing these methods significantly reduces machine learning model size, lowers memory usage, and accelerates inference. However, it has also been observed that most of these approaches establish a static model configuration prior to deployment; once optimization is complete, the model's structure and behavior remain virtually unchanged. Consequently, it is evident that optimization efficiency may vary depending on the device's current state and execution conditions.

Keywords: resource constraints, artificial intelligence, machine learning, databases, mobile platforms, edge platforms, software development, computer networks.



REFERENCES (TRANSLATED AND TRANSLITERATED)

1. Ha, R. (2025, August 25). *What is Edge AI? Benefits and challenges*. INVIAI.
2. ProIT. (n.d.). *How edge computing is changing computing technology*.
3. European Telecommunications Standards Institute. (2019). *Multi-access Edge Computing (MEC); framework and reference architecture* (ETSI GS MEC 003).
4. Google. (2026, May 28). *Performance best practices*. Google AI Edge.
5. Mao, Z. (2025). Lightweight network architecture—Optimization of deep learning models. *Applied and Computational Engineering*, 175, 178–185. <https://doi.org/10.54254/2755-2721/2025.AST26509>
6. Liu, L., & Xu, Z. (2025). Optimizing lightweight neural networks for efficient mobile edge computing. *Scientific Reports*, 15, Article 22056. <https://doi.org/10.1038/s41598-025-04652-7>
7. Howard, A. G., & Zhu, M. (2017, June 14). *MobileNets: Open-source models for efficient on-device vision*. Google Research.
8. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2018). MobileNetV2: Inverted residuals and linear bottlenecks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 4510–4520). IEEE. <https://doi.org/10.1109/CVPR.2018.00474>
9. Ma, N., Zhang, X., Zheng, H.-T., & Sun, J. (2018). ShuffleNet V2: Practical guidelines for efficient CNN architecture design. In V. Ferrari, M. Hebert, C. Sminchisescu, & Y. Weiss (Eds.), *Computer vision—ECCV 2018* (Lecture Notes in Computer Science, Vol. 11218, pp. 116–131). Springer. https://doi.org/10.1007/978-3-030-01264-9_8
10. Tan, M., & Le, Q. V. (2019). EfficientNet: Rethinking model scaling for convolutional neural networks. In K. Chaudhuri & R. Salakhutdinov (Eds.), *Proceedings of the 36th International Conference on Machine Learning* (Vol. 97, pp. 6105–6114). PMLR. <https://proceedings.mlr.press/v97/tan19a.html>
11. Kuznetsov, O., Frontoni, E., Kuznetsova, Y., Chevardin, V., & Smirnov, O. (2026). Architectural foundations for adaptive security in edge computing systems. In A. L. Imoize, F. Khafa, M. S. Obaidat, & H. H. Song (Eds.), *Cybersecurity defensive walls in edge computing* (pp. 21–61). Academic Press. <https://doi.org/10.1016/B978-0-443-34109-0.00015-2>
12. Kuznetsov, O., Atzeni, G., Arnesano, M., Randieri, C., & Smirnov, O. (2025). Secure IoT-based smart wheelchair system: From implementation to security enhancement strategy. In A. L. Imoize, W. Montlouis, S. I. Popoola, & M. Hammoudeh (Eds.), *Security and privacy of cyber-physical systems: Emerging trends, technologies, and applications* (pp. 225–257). River Publishers. <https://doi.org/10.1201/9788743801306-8>
13. Kuznetsov, O., Smirnov, O., Kuznetsova, T., Shaikhanova, A., & Svatowsky, I. (2025). Privacy–utility trade-offs in IoT networks: A comparative analysis of differential privacy mechanisms for sensor data aggregation. *Security and privacy of cyber-physical systems: Emerging trends, technologies, and applications* (pp. 589–622). River Publishers. <https://doi.org/10.1201/9788743801306-17>
14. Smirnov, O., Tkachuk, R., Kozirova, N., Konstantynova, L., Konoplińska-Slobodeniuk, O., Yakymenko, N., & Smirnov, S. (2026). Research on the application of vector databases in generative artificial intelligence. *Cybersecurity: Education, Science, Technique*, 1(33), 667–682. <https://doi.org/10.28925/2663-4023.2026.33.1248>
15. Bohachenko, S., Smirnov, O., Buravchenko, K., Smirnova, T., Konoplińska-Slobodeniuk, O., Yakymenko, N., & Smirnov, S. (2026). Resource constraints of mobile and edge platforms in software development for the implementation and optimization of machine learning models. *Cybersecurity: Education, Science, Technique*, 2(34), 677–686. <https://doi.org/10.28925/2663-4023.2026.34.1360>
16. Xu, T.-B., Yang, P., Zhang, X.-Y., & Liu, C.-L. (2019). LightweightNet: Toward fast and lightweight convolutional neural networks via architecture distillation. *Pattern Recognition*, 88, 272–284. <https://doi.org/10.1016/j.patcog.2018.10.029>
17. Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H., & Kalenichenko, D. (2018). Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 2704–2713). IEEE. <https://doi.org/10.1109/CVPR.2018.00286>
18. Hinton, G., Vinyals, O., & Dean, J. (2015). *Distilling the knowledge in a neural network* [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.1503.02531>

Отримано редакцією журналу / Received: 30.05.26

Прорецензовано / Revised: 16.06.26

Схвалено до друку / Accepted: 26.10.26



This work is licensed under Creative Commons Attribution-noncommercial-sharealike 4.0 International License.