ОБҐРУНТУВАННЯ ДОЦІЛЬНОСТІ ВИКОРИСТАННЯ ПЛАТФОРМИ TENSORFLOW LITE ДЛЯ МАШИННОГО НАВЧАННЯ НА ПРИСТРОЯХ

Целік Максим Романович
студент
Арсенюк Ігор Ростиславович
к. т. н., доцент
Вінницький національний технічний університет
м. Вінниця, Україна

Вступ. / Introductions. Розвиток машинного навчання на кінцевих пристроях (on-device AI) відкриває безліч можливостей для різних сфер діяльності, зокрема для мобільних додатків, Інтернету речей (IoT) та вбудованих систем. Виконання моделей TensorFlow Lite безпосередньо на пристрої користувача, а не на віддаленому сервері, надає суттєві переваги, такі як низька затримка, покращена конфіденційність (оскільки дані користувача не залишають пристрій) та можливість роботи в офлайн-режимі. Однак, кінцеві пристрої мають значні обмеження щодо обчислювальної потужності, обсягу пам’яті та енергоспоживання порівняно з хмарними серверами. Це створює виклик для розробників: як розгорнути складні моделі машинного навчання в таких обмежених умовах, зберігши при цьому їхню точність та продуктивність.

TensorFlow Lite (TFLite) була розроблена як провідна платформа для вирішення саме цієї проблеми. Вона пропонує широкий спектр інструментів для оптимізації, розгортання та керування моделями машинного навчання безпосередньо на пристроях. TFLite дозволяє розробникам створювати швидкі, ефективні та конфіденційні додатки зі штучним інтелектом для мільярдів пристроїв, включаючи мобільні платформи (Android, iOS) та вбудовані системи (Linux, Microcontrollers).

Мета роботи. / Aim. Метою даної роботи є дослідження ключових компонентів, архітектури та функціональних можливостей платформи TensorFlow Lite. Робота ставить за ціль детально проаналізувати основні механізми, які TFLite надає для розгортання моделей машинного навчання на

пристроях, зокрема:

1. Вивчити процес підготовки моделей за допомогою конвертера TFLite.

2. Розглянути принципи роботи інтерпретатора TFLite та його роль у виконанні виведення.

3. Проаналізувати основні методи оптимізації моделей, зокрема квантизацію, проріджування та кластеризацію.

4. Оцінити механізми апаратного прискорення (делегати GPU, DSP, NPU).

5. Визначити основні переваги та обмеження використання TFLite у порівнянні з традиційними хмарними API для машинного навчання.

Матеріали та методи. / Materials and methods. Для досягнення поставленої цілі в роботі використовувався метод аналізу технічної документації та архітектури платформи TensorFlow Lite. Дослідження базується на вивченні трьох основних компонентів системи: конвертера, інтерпретатора та інструментів оптимізації.

Конвертер TensorFlow Lite є фундаментальним інструментом, що перетворює навчені моделі TensorFlow (у форматах SavedModel, Keras H5 або з tf.function) у спеціальний оптимізований формат .tflite. Цей формат розроблений для ефективного виконання на пристроях.

Основними функціями конвертера є квантизація (Quantization) дозволяє конвертувати 32-бітні ваги з рухомою комою у 8-бітні цілі числа або 16-бітні з рухомою комою, також можливість конвертувати моделі, збережені як TensorFlow SavedModel, моделі Keras, або конкретні tf.function, не можна не згадати що сумісність операторів гарантує, що операції у моделі TensorFlow (TF ops) будуть або перетворені на еквівалентні операції TFLite, або (за потреби) модель включатиме необхідні операції TF для виконання, а генерація метаданих дозволяє вбудовувати метадані в .tflite модель, що полегшує її інтеграцію.

Інтерпретатор TFLite – це ядро середовища виконання (runtime), яке завантажує .tflite моделі та виконує на них виведення (inference). Він легкий,

швидкий та крос-платформний; має малий бінарний розмір та мінімальні залежності; надає C++ API та обгортки для мов Java/Kotlin (Android), Swift/Objective-C (iOS) і Python. Крім того, в інтерпретаторі передбачено апаратне прискорення (Delegates): він може передавати виконання частини або всієї моделі спеціалізованим апаратним прискорювачам (делегатам) для значного приросту продуктивності. Підтримувані делегати включають GPU Delegate, NNAPI Delegate (для Android використовується NPU), Core ML Delegate (для iOS) та Hexagon DSP.

Процес роботи наведено на рис. 1. Тут передбачається завантаження моделі, виділення тензорів для входу та виходу, надання вхідних даних, виклик інтерпретатора та зчитування результатів.

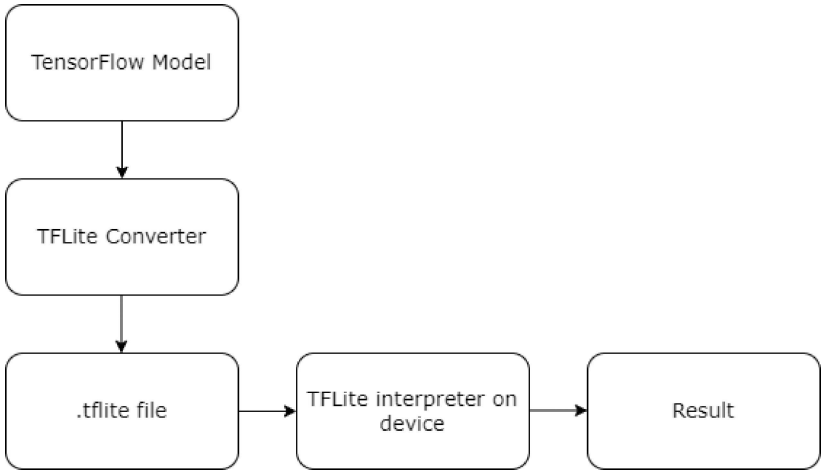

Рисунок 1 – Принцип роботи TFLite

Оптимізація є дуже важливим методом для адаптації моделей до обмежених ресурсів пристроїв. TFLite використовує кілька ключових технік (рисунок 2).

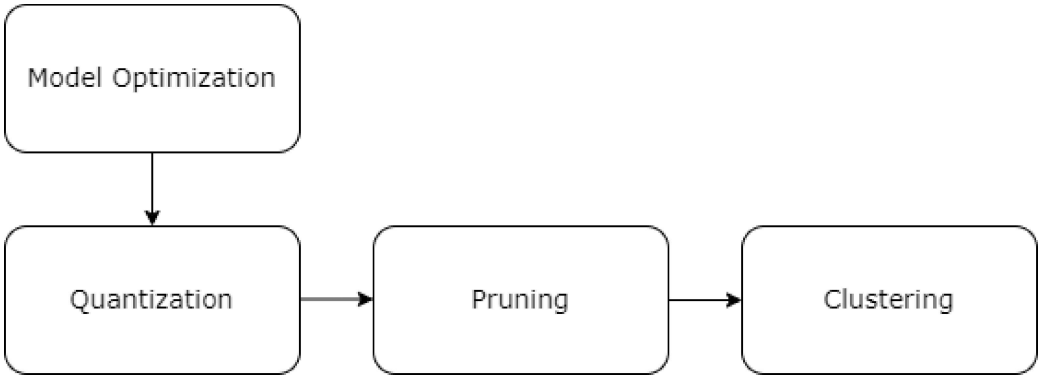

Рисунок 2 – Основні техніки оптимізації в TFLite

Квантизація (Quantization) передбачає зменшення точності чисел для представлення ваг (наприклад, з float32 до int8). Це значно зменшує розмір моделі (до 4 разів) та прискорює обчислення. Post-training quantization (квантизація після навчання): найпростіший спосіб, оптимізує вже навчену модель. Quantization-aware training: імітує ефекти квантизації під час навчання, що зазвичай приводить до вищої точності.

Проріджування (Pruning) передбачає видалення параметрів (вагів) моделі, які мають незначний вплив на результат, роблячи модель "рідкою" (sparse), що може значно зменшити розмір для стиснення.

Кластеризація (Clustering) передбачає групування ваг у кластери та використовує одне значення для всіх ваг у кластері, зменшуючи кількість унікальних значень, що також допомагає у стисненні.

Результати та обговорення./Results and discussion. Аналіз функціональних можливостей TensorFlow Lite показує, що платформа надає комплексне рішення для ефективного розгортання ШІ на кінцевих пристроях. Обговорення результатів зосереджено на перевагах, які дає використання наведених методів.

Використання інтерпретатора TFLite у поєднанні з апаратними делегатами (GPU, NPU) дозволяє досягти значного прискорення виведення. Замість того, щоб покладатися на центральний процесор, обчислення перенаправляються на спеціалізовані процесори. Це дуже важливо для завдань у реальному часі, таких як розпізнавання зображень з камери або обробка аудіо.

Методи оптимізації, особливо квантизація, дозволяють зменшити розмір моделі у 4 (float32 до int8), або у 2 рази (float16). Проріджування та кластеризація ще більше стискають модель. Це безпосередньо впливає на загальний розмір мобільного додатку, зменшуючи час завантаження та вимоги до пам'яті пристрою.

Оскільки виведення відбувається на пристрої, додатки, що використовують TFLite, не потребують постійного підключення до Інтернету. Це забезпечує надійну роботу в умовах нестабільного зв'язку або в авіарежимі,

що є неможливим для рішень, які покладаються виключно на серверне виведення.

TensorFlow Lite (TFLite) вирізняється серед інших рішень для on-device inference низкою архітектурних та практичних переваг, які роблять його оптимальним вибором для широкого спектра мобільних, вбудованих та TinyML-завдань.

Ключовими перевагами є повна екосистема оптимізації та розгортання [1]. TFLite поєднує потужний конвертер (.tflite), інтерпретатор із делегатами апаратного прискорення (GPU, NNAPI, Core ML) та інструменти для квантизації, проріджування та кластеризації – усе це «з коробки», що спрощує шлях від навчання до роботи на пристрої. Також до переваг можна віднести широку підтримку платформ – від смартфонів до мікроконтролерів. Окрема підсистема TensorFlow Lite for Microcontrollers дозволяє запускати моделі на пристроях з десятками кілобайт пам'яті, що ставить TFLite у вигідну позицію порівняно з фреймворками, орієнтованими лише на мобільні процесори. Також не можна оминути ефективність (розмір моделі та енергоспоживання) за рахунок широкої підтримки квантизації (post-training та quantization-aware training) та оптимізованих ядер [2] TFLite дозволяє зменшити розмір моделі в ~4× (float32 → int8) і часто зменшує затримку та енергоспоживання під час inference. Дослідження на вбудованих платформах показують, що TFLite входить до числа енергоефективних рішень поряд з ONNX Runtime, залишаючись при цьому дуже портативним. Готові делегати апаратного прискорення також є великою перевагою адже механізм делегатів дозволяє передавати частини моделі під спеціалізовані пристрої (GPU, NPU/Neural Engine, DSP), що дає значний приріст продуктивності у реальному застосуванні (наприклад, обробка відеопотоку в режимі реального часу).

Порівняння з основними альтернативами:

PyTorch Mobile [3]: краще підходить для швидкого прототипування та у випадках, коли модель проєктувалась у PyTorch; проте TFLite зазвичай має більше оптимізованих інструментів для мобільного inference, ширшу підтримку

делегатів і спеціалізовану підсистему для microcontrollers. У ряді випадків PyTorch Mobile виявляється більш «легким» для інтеграції в Python-орієнтований стек, але в задачах обмежених ресурсів (TinyML, енергоефективність) TFLite має перевагу.

Core ML (Apple): для iOS-платформи Core ML інколи дає кращу продуктивність завдяки тісній інтеграції з апаратурою Apple та оптимізованим Apple API; однак Core ML – це екосистема, прив'язана до Apple-пристроїв, тоді як TFLite є крос-платформним (Android, iOS, embedded, MCU). Тому якщо важлива портативність на різних платформах – перевага на боці TFLite; якщо ж ціль – максимальна швидкість тільки на iPhone – Core ML може випереджати.

ONNX Runtime (Mobile): ONNX став популярним як проміжний формат для перенесення моделей між фреймворками; ONNX Runtime Mobile іноді демонструє конкурентну або кращу енергоефективність у деяких сценаріях. Проте ONNX більше позиціонується як універсальний рантайм для моделей із різних бекендів, тоді як TFLite пропонує більш «цілісну» екосистему для мобільного/вбудованого розгортання, включаючи TF-специфічні оптимізації та TF Micro.

Загалом, використання платформи TFLite відкриває перед компаніями великі можливості для інновацій, підвищення ефективності додатків та покращення користувацького досвіду, ефективно управляючи обмеженими ресурсами пристрою.

Висновки./Conclusions. Використання платформи TensorFlow Lite може значно полегшити розробникам створення додатків з машинним навчанням на пристроях, забезпечуючи високу продуктивність, низьку затримку та конфіденційність. Її широкий спектр інструментів для оптимізації, крос-платформність та постійні інновації роблять її одним з найкращих виборів для будь-кого, хто шукає рішення в області мобільного ШІ та IoT. Зважаючи на всі наведені переваги саме TensorFlow Lite буде найкращим інструментом для багатьох розробників інноваційних продуктів.

Незалежно від масштабів проєкту, TFLite забезпечить інструменти,

необхідні для ефективного запуску моделей, і дозволить розгортати рішення на мільярдах пристроїв, не потребуючи значних витрат на серверну інфраструктуру для виведення. Крім того, TFLite пропонує різноманітні інструменти для оптимізації, тестування та моніторингу продуктивності моделей, що дозволяє розробникам зосередитися на покращенні функціональності, замість витрат на управління складними серверними рішеннями.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. TensorFlow Lite Core ML delegate enables faster inference on iPhones and iPads [Електронний ресурс] – Режим доступу до ресурсу: https://blog.tensorflow.org/2020/04/tensorflow-lite-core-ml-delegate-faster-inference-iphones-ipads.html?utm_source=chatgpt.com

2. Exploring energy consumption of AI frameworks on a 64-core RV64 Server CPU [Електронний ресурс] – Режим доступу до ресурсу: https://arxiv.org/html/2504.03774v1?utm_source=chatgpt.com

3. TensorFlow Lite vs PyTorch Mobile for On-Device Machine Learning [Електронний ресурс] – Режим доступу до ресурсу: https://www.analyticsvidhya.com/blog/2024/12/tensorflow-lite-vs-pytorch-mobile/?utm_source=chatgpt.com