https://doi.org/10.31891/2307-5732-2025-355-94
УДК 004.8

ФАРІОН МИКОЛА
Національний університет «Львівська політехніка»
https://orcid.org/0009-0004-2897-0495
e-mail: mykola.y.farion@lpnu.ua

ОГЛЯД ФРЕЙМВОРКІВ ДЛЯ МОБІЛЬНОЇ ІНФЕРЕНЦІЇ МОДЕЛЕЙ МАШИННОГО НАВЧАННЯ НА ПЛАТФОРМІ iOS

У статті проведено аналіз розвитку бібліотек машинного навчання, виділено основні ML фреймворки, які можна використовувати для мобільних додатків на платформі iOS. Описано принципові відмінності та рекомендації застосовування. Проаналізовано еволюцію інструментів глибокого навчання, починаючи з Theano та Caffe, до сучасних рішень, таких як Core ML, TensorFlow Lite (LiteRT), ONNX Runtime, PyTorch Mobile та ExecuTorch. Здійснено порівняння їхніх можливостей, форматів моделей, способів інтеграції та обмежень. Особливу увагу приділено питанню гнучкості, простоти впровадження та відповідності обраному фреймворку до практичних потреб розробників мобільних додатків.

Ключові слова: машинне навчання, мобільна інференція, iOS, ML фреймворки

FARION MYKOLA
Lviv Polytechnic National University

A REVIEW OF MACHINE LEARNING INFERENCE FRAMEWORKS FOR iOS-BASED MOBILE APPLICATIONS

This paper provides a comprehensive review of current frameworks for machine learning (ML) inference on the iOS platform. With the growing computational capabilities of mobile devices, on-device ML inference has become a viable and increasingly popular approach, reducing reliance on server-side processing. The review identifies and evaluates key ML frameworks suitable for mobile deployment, including Core ML, TensorFlow Lite (recently rebranded as LiteRT), ONNX Runtime, PyTorch Mobile, and ExecuTorch. The study outlines the historical evolution of ML tools, from early frameworks such as Theano and Caffe to modern, production-ready solutions.

The analysis focuses on core differences in model format support, integration complexity, hardware acceleration capabilities, and compatibility with the iOS ecosystem. Core ML, as Apple's native framework, is optimized for seamless integration and hardware performance, while TensorFlow Lite and ONNX Runtime offer cross-platform potential and flexibility. PyTorch-based solutions provide dynamic graph execution and are favored by the research community, especially after the introduction of ExecuTorch, which enhances mobile efficiency and hardware support.

The article also discusses the decline of legacy frameworks and the consolidation around more versatile and actively maintained tools. The strengths and limitations of each framework are compared based on use-case suitability, ease of integration, and optimization support.

This work aims to guide mobile developers in making informed decisions when selecting an ML inference framework for iOS apps. It highlights how different tools meet the practical demands of mobile AI applications and stresses the importance of considering factors such as platform constraints, energy efficiency, and maintainability. Future research directions include benchmarking model performance across frameworks and evaluating inference speed and resource usage on real-world iOS devices.

Keywords: Machine Learning, Mobile Inference, iOS, ML Frameworks

Стаття надійшла до редакції / Received 19.05.2025
Прийнята до друку / Accepted 26.06.2025

Постановка проблеми у загальному вигляді та її зв'язок із важливими науковими чи практичними завданнями

У сучасному світі мобільні пристрої відіграють ключову роль у повсякденному житті, а їхні обчислювальні можливості з кожним роком зростають. Це відкриває широкі перспективи для реалізації задач машинного навчання безпосередньо на мобільних пристроях — без необхідності постійного з'єднання з сервером. Водночас така реалізація вимагає адаптації моделей штучного інтелекту до обмежених ресурсів мобільних платформ, а також врахування специфіки операційних систем, зокрема iOS.

При збільшенні кількості можливостей зростає ризик правильного вибору. Розвиток технологій машинного навчання призвів до ситуації, де загальний інформаційний шум породжує нову недостовірну інформацію, оскільки при генерації контенту використовуються застарілі дані.

Науковий інтерес до цієї проблематики пов'язаний із необхідністю глибшого розуміння можливостей існуючих фреймворків (таких як TensorFlow Lite, PyTorch Mobile, ONNX Runtime та CoreML), їх продуктивності у різних умовах, простоти інтеграції, підтримки сучасних форматів моделей та сумісності з інструментами розробки iOS-додатків. У практичному сенсі це знання дає змогу створювати ефективніші, продуктивніші та енергоощадні мобільні застосунки на базі штучного інтелекту.

Таким чином, проблема полягає у відсутності систематизованого порівняння сучасних фреймворків мобільної інференції в контексті їхньої придатності до використання на iOS-платформі. Її вирішення дозволить розробникам приймати обґрунтовані технічні рішення, оптимізувати процес розробки та інтеграції моделей машинного навчання, а також сприятиме подальшому поширенню технологій штучного інтелекту в мобільному середовищі.

Аналіз досліджень та публікацій

Бурхливий розвиток технологій машинного навчання повязаний з інтересом до можливостей

глибинних нейронних мереж, а також з можливістю захоплення нової ринкової ніші, яка була недоступна з класичними методами навчання. Розробкою фреймворків займалась велика кількість компаній та команд - від ранніх академічних розробок Caffe та Theano до масивних, підтримуваних індустрією, PyTorch та TensorFlow [1]. Після стрімкої появи численних розробок у період з 2012 по 2019 спостерігається процес виокремлення явних лідерів, в той час як деякі інші продукти навпаки зупинили розробку і підтримку.

Nguyen та Dlugolinsky у своїй роботі [2] зазначають, що немає одного універсального інструменту для вирішення усіх задач та виділяють TensorFlow, CNTK, Caffe, Caffe2, Torch, PyTorch, MXNet, Theano, Chainer, PaddlePaddle, Keras бібліотеки як приклад найбільш перспективних лідерів у розробці машинного навчання.

Для задоволення різних потреб у процесі машинного навчання доступний широкий спектр ресурсів - від великих бібліотек, до спеціалізованих платформ. Однак, вибір відповідного фреймворку для конкретної програми все ще є значною перешкодою - на це рішення впливають різні фактори, такі як тип проблеми, обсяг та складність інформації, необхідна швидкість та рівень кваліфікації користувача [3].

Результати дослідження можливостей перенесення основних фреймворків машинного навчання на різні типи обладнання [4] показали, що фреймворки можуть втратити понад 40% своїх ключових функцій під час перенесення. І навіть коли функції є портативними, значне уповільнення при перенесенні може знизити продуктивність до неприйнятного рівня. Це доводить, що вибір фреймворку, який найбільш відповідає вимогам дослідження, може бути визначальним для його успішної реалізації.

Формулювання цілей статті

Метою роботи є надання вичерпного огляду підходів до інференції машинного навчання на мобільній платформі, особливостей вибору актуальних фреймворків. Стаття має на меті допомогти розробникам iOS зрозуміти доступні варіанти та прийняти обґрунтоване рішення щодо вибору оптимального фреймворку для їхніх конкретних потреб у машинному навчанні на пристроях.

Виклад основного матеріалу

У цій статті розглянемо поточний стан та перспективи подальшого використання бібліотек машинного навчання для застосування у розробці на мобільних платформах.

Theano, випущений у 2007 році, був одним з перших популярних фреймворків для глибокого навчання. У 2017 через конкуренцію з іншими фреймворками команда Монреальского інституту алгоритмів навчання (MILA) припинила розробку **Theano**. Продовження проєкту доступні у відгалуженнях **Aesara**, а потім **PyTensor**, що використовується як основа для бібліотеки ймовірнісного програмування PyMC.

Бібліотека **Torch,** створена у 2002 році, використовувала Lua та C для наукових розрахунків, а пізніше для нейронних мереж. У 2017 **Torch** бібліотека була переведена на мову Python як проєкт **PyTorch**.

Caffe це фреймворк глибокого навчання, створений з урахуванням модульності та швидкодії. Він розроблений Berkeley AI Research (BAIR)/The Berkeley Vision and Learning Center (BVLC) та учасниками спільноти.

Caffe2 це покращена версія **Caffe**. Фреймворк,анонсований Facebook у квітні 2017 року, включав нові можливості, такі як підтримка рекурентних нейронних мереж (RNN). У березні 2018 року Caffe2 був об'єднаний з PyTorch, і його розвиток як окремого фреймворку припинився [5].

У 2015 році команда Google Brain випустила TensorFlow фреймворк. Цей проект стартував як модернізація DistBelief і використовувався для задач машинного навчання компанією Google. Перша версія базувалась на статичних графах, підтримувала розподілене навчання, але мала високий поріг входження.

Цього ж року Франсуа Шолле, розробник Google, створив **Keras** - високорівневий API, що використовується для створення та навчання нейронних мереж. Перевагою Keras v1 була можливість вибору низькорівневих бібліотек для бекенду (Theano, TensorFlow, CNTK), простота та гнучкість. У 2019 Keras v2 була інтегрована у TensorFlow v2, ця зміна покращила взаємодію з TensorFlow, але не надавала попередніх можливостей вибору бекенду. У 2023 з виходом Keras v3 ця можливість відновилась.

Для виконання TensorFlow моделей на мобільних пристроях було розроблено TensorFlow Lite (TFLite).

Фреймворк TFLite зберігає модель у .tflite форматі, що оптимізований для мобільних пристроїв. Перша iOS версія TFLite виконувалась на CPU, з розвитком бібліотеки додали підтримку GPU. У 2024 року TensorFlow Lite зазнав ребрендингу і змінив назву на LiteRT (скорочення від Lite Runtime) [6].

У 2015 році був створений **Chainer** - фреймворк на мові Python, що використовував динамічні обчислювальні графи. Це робило його інтуїтивним, а також гнучким у застосуванні в порівнянні з Theano чи TensorFlow, які на той момент були статичними. У грудні 2019 команда Chainer перевела проєкт у режим підтримки, а сама перейшла на PyTorch.

У квітні 2015 року був випущений **CNTK** (Microsoft Cognitive Toolkit), фреймворк комерційного рівня , що реалізовував ефективне навчання глибоких нейронних мереж для текстових даних, мовлення, зображень.

У 2017 році Microsoft разом з Facebook анонсують формат **ONNX** (Open Neural Network Exchange), який спочатку мав назву Toffee. Метою цього формату є забезпечення сумісності між фреймворками машинного навчання, що дозволить використовувати моделі розроблені на одному фреймворку та експортовані у цей формат легко імпортувати іншими фреймворками.

ONNX — це відкритий стандарт, який визначає спільний набір операторів та спільний формат файлу

для представлення моделей глибокого навчання в широкому спектрі фреймворків, включаючи PyTorch та TensorFlow. Коли модель експортується у формат ONNX, ці оператори використовуються для побудови обчислювального графа (проміжного представлення), який відображає потік даних через нейронну мережу [7].

У 2018 компанія Microsoft випустила крос-платформний фреймворк під назвою **ONNX Runtime** (ORT), що використовується для інференцій моделей на основі ONNX формату. У 2019 році компанія Microsoft оголосила про завершення розробки **CNTK**, попередньо реалізувавши експорт моделей у ONNX.

Deeplearning4j бібліотека написана на мові Java не підтримує iOS платформу. Бібліотека була популярна до 2019, однак після скорочення команди, цей проєкт підтримується лише одним з авторів.

MXNet фреймворк було випущено у 2015 році. Його особливістю були можливість працювати з кількома мовами програмування (Python, C++, R, та інші), підтримка розподіленого навчання та масштабованість. У 2023 році через зниження популярності на фоні PyTorch та TensorFlow проєкт MXNet був зупинений.

Bender - вузькоспеціалізований фреймворк машинного навчання на платформі iOS, створений у 2017 році, що виконувався на GPU мобільного пристрою завдяки використанню низькорівневої бібліотеки MetalPerformanceShaders. Його перевагою над CoreML була відкритість коду, швидкодія та гнучкість. Фреймворк складався з шарів, що дозволяли адаптовувати TensorFlow код, який на той час не мав підтримки GPU на мобільні пристрої [8]. У 2018 проєкт призупинив активну розробку, а конкуренти доповнили свої фреймворки можливістю виконання на GPU.

Thakkar та Gallagher [9, 10] у своїх книгах посилаються на **Turi Create** як засіб розробки моделей машинного навчання під мобільну платформу iOS. У 2017 після придбання Turi Create компанія Apple виклала бібліотеку у відкритий доступ, та продовжила її адаптацію зі своїм новим продуктом CoreML. Після 2020-го Turi Create проєкт перейшов до режиму підтримки, а у 2023 році компанія зупинила проєкт повністю. Станом на зараз Turi Create не використовується, а CoreML є основою нативною бібліотекою для роботи з моделями машинного навчання.

Станом на зараз до існуючих бібліотек машинного навчання для інференції на мобільному можна віднести PyTorch, TensorFlow, ONNX Runtime, MNN, PaddlePaddle, MACE, Bender, CoreML. Однак не всі ці фреймворки підтримують iOS платформу, інші мають обмежений регіон використання, а деякі з них втратили свою перевагу з часом. Розглянемо більш детально версійність, формати експорту моделей [11-13] для актуальних фреймворків в таблиці 1.

Таблиця 1

Актуальні фреймворки для мобільної інференції

Фреймворк	Версія	Рік випуску	Розширення файлу для експорту моделі
TensorFlow	1.x	2015 - початок реліз процесу, 2017 - 1.0 реліз	.pb (frozen graphs), .ckpt (checkpoints)
	2.x	2019	.pb, .h5, SavedModel (формат папки)
TensorFlow Lite	1.x	2017	.tflite
	2.x	2020	.tflite
	3.x	2023	.tflite
LiteRT	1.x	2024	.tflite
Keras	1.x	2015	.h5
	2.x	2017	.h5, .keras
	3.x	2023	.keras, .h5, SavedModel (формат папки), .pt, .jax (JAX model format)
PyTorch	1.x	2018	.pt, .pth, .pkl
	2.x	2023	.pt, .pth, .onnx
PyTorch Mobile	1.x	2019	.ptl
ExecuTorch	0.6	2023	.pte
ONNX Runtime	1.x	2018 - початок реліз процесу, 2019 - 1.0 реліз	.onnx, .ort
CoreML Tools	1.x	2017	.mlmodel
	4.x	2020	.mlmodel, .mlpackage
	8.x	2024	.mlpackage, .mlmodel (для iOS < 15.0)

Завдяки своїй простоті інтеграції, гнучкості, а також підтримці динамічних обчислювальних графів **PyTorch** фреймворк популярний серед науковців та дослідників. Для конвертації PyTorch моделі до моделі для мобільного інференсу PyTorch Mobile її спочатку переводять у TorchScript представлення, яке можна оптимізовувати та зберігати як модель для мобільного додатку. Такий підхід дозволяє спростити розробку моделей для мобільних пристроїв до автоматичного конвеєра, що окрім оптимізації може застосовувати

квантування та обрізку ваг.

Початково PyTorch використовував PyTorch Mobile моделі у форматі .ptl для мобільних додатків, але у 2023 для розширення підтримки апаратних платформ, а також оптимізації продуктивності було розроблено ExecuTorch. Архітектура нового фреймворка дозволяє компактне представлення моделей у форматі .pte та ефективне використання апаратних можливостей мобільних пристроїв та вбудованих систем.

У 2024 році виходить PyTorch 2.0, де фреймворк зазнає суттєвих змін з точки зору оптимізації моделей та продуктивності за рахунок написання нових модулів TorchDynamo, AOTAutograd, PrimTorch, TorchInductor та їх використання для компіляції моделі.

Комбінація PyTorch 2.x та ExecuTorch зберігає простоту використання, але значно покращує продуктивність та оптимізацію моделей.

Create ML - це інструмент платформи macOS, створений у 2018 році, що використовується для створення та тренування моделей. У першій версії це був редактор, з обмеженим набором моделей та алгоритмів. Наступні версії працюють без необхідності писати код, але з можливостями контролю процесу навчання. Користувачу надається вибір типу моделі (з попередньо визначених типів задач), параметрів навчання. На основі даних користувача інструмент може створити (шляхом донавчання загальної моделі або ж з нуля) кастомізовану модель для обраного типу задач. Наразі підтримуються моделі для задач класифікації зображень, виявлення об'єктів, передачі стилю, класифікації дій, тексту, звуків, а також класифікація та регресія на основі числових даних [14].

Інструменти CoreML Tools можуть конвертувати навчені моделі з інших фреймворків у формат .mlmodel або .mlpackage, що використовуються у додатках для iOS, macOS та visionOS платформ [11].

Інтеграція Core ML моделі у нативний Xcode проєкт не потребує надлишкових зусиль. Достатньо перемістити файл моделі у форматі .mlpackage або .mlmodel у проєкт, а система згенерує інтерфейс з функцією для виконання інференції моделі. У останній версії було додано можливість виконання декількох задач однією моделлю. Під час створення мобільного додатку файл моделі теж компілюється і зберігається у контейнері додатку з розширенням .mlmodelc, що є оптимізованим для пристроїв екосистеми Apple.

Підсумовуючи можна зазначити, що CreateML є найпростішим інструментом для інференції на мобільному пристрої, але потребує JSON формату для розмітки даних. Якщо розробнику не достатньо базових моделей, бо наприклад CreateML використовує YOLOv3 для детекції об’єктів, то потрібно обирати фреймворк для навчання та інференції моделі (Рис.1).

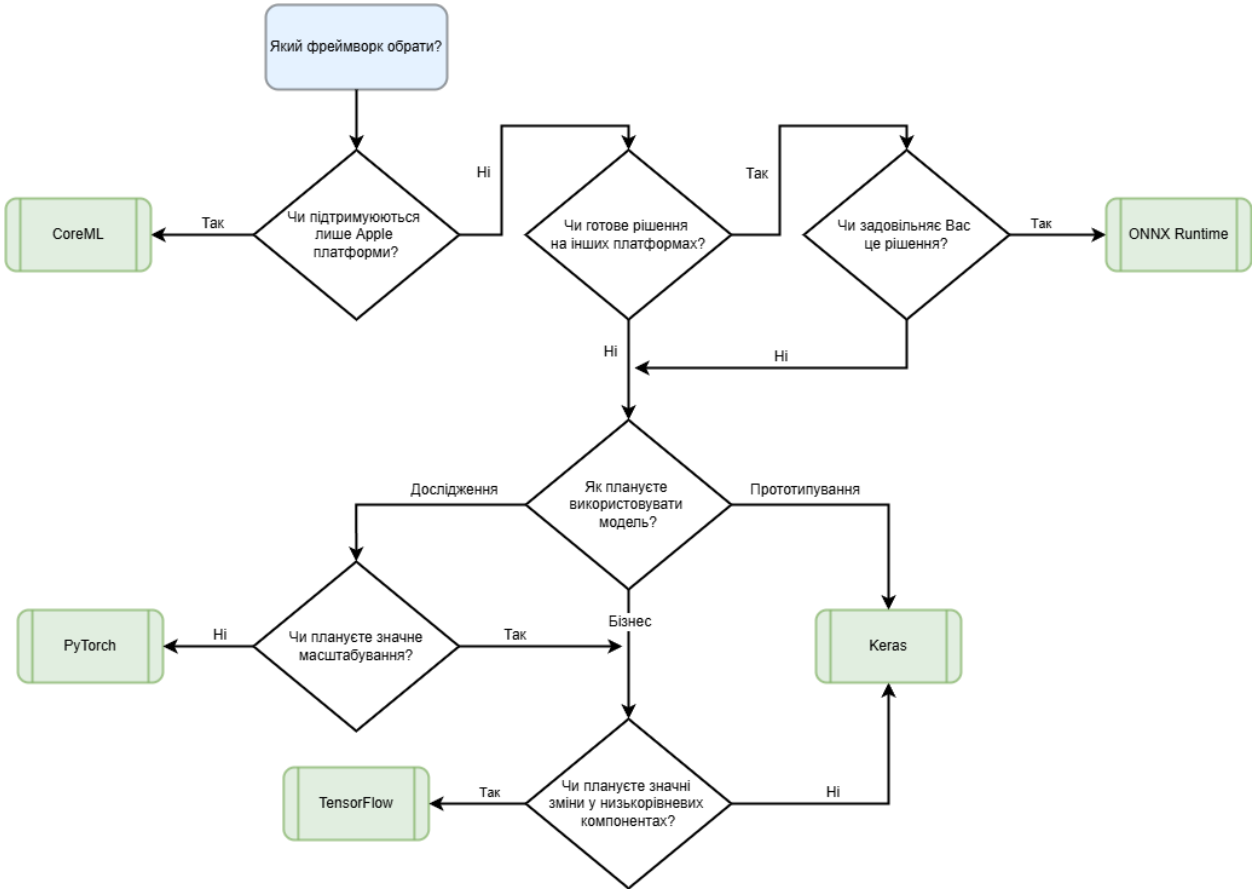

Рис. 1. Схема прийняття рішень для вибору фреймворка

Серед кількісних характеристик вибору фреймворку важливими є точність, продуктивність, час підвантаження моделі, час інференції, споживання ресурсів системи.

Окрім вищезгаданих критеріїв вибору варто також згадати складність конвертації моделі, що спричиняє більш як 20% усіх помилок застосування моделі машинного навчання на мобільному. Якщо у розробника недостатньо досвіду, то безпечнішим варіантом буде використання ONNX Runtime ніж конвертація засобами CoreML Tool. При можливості варто обирати стабільні фреймворки з великою спільнотою та відкритим кодом. Характерною відмінністю ONNX Runtime/ExecuTorch/RTLite в порівнянні з CoreML є відкритий код.

Також слід брати до уваги архітектуру та мови, які використовуються у фреймворку. У ONNX Runtime та ExecuTorch використовують динамічні бібліотеки, які необхідно компілювати та інтегрувати в проєкт для роботи з файлом моделі у форматі (.ort / .pte). RTLite вирізняється тим, що містить інтерпретатор на платформенному рівні та спільну базову бібліотеку на C. Для роботи з ONNX Runtime/ExecuTorch розробник повинен знати Python, C++, Objective-C та Swift мови, тоді як для CoreML та RTLite достатньо лише знання Python та Swift.

Висновки з даного дослідження
і перспективи подальших розвідок у даному напрямі

У статті проведено аналіз сучасних фреймворків машинного навчання, придатних для реалізації інференції моделей на мобільній платформі iOS. Розглянуто можливості, обмеження та особливості інтеграції таких популярних рішень, як Core ML, PyTorch Mobile, TensorFlow Lite (LiteRT) та ONNX Runtime. Зіставлення архітектурних підходів, форматів моделей, підтримки оптимізацій та вимог до середовища виконання дозволяє зробити низку практичних висновків.

Core ML є нативним фреймворком для платформи iOS, що забезпечує високу продуктивність, зручність інтеграції та глибоку оптимізацію під апаратні можливості пристроїв Apple. Він є оптимальним вибором у випадках, коли розробка відбувається виключно під iOS і не вимагає реалізації на кількох платформах або динамічного оновлення моделей. Суттєвим недоліком є закритий код.

TensorFlow Lite (LiteRT) забезпечує широку гнучкість, активну підтримку спільнотою, а також наявність інструментів для автоматичної оптимізації моделей. Водночас, його інтеграція в iOS-проєкти потребує додаткових зусиль в порівнянні з CoreML.

ExecuTorch надає високу продуктивність, компактне збереження моделей та ефективне використання ресурсів системи. При використанні для декількох платформ зв'язка ExecuTorch з PyTorch 2.x є максимально результативною. Недоліками є ускладнений підхід щодо інтеграції - компілювання бібліотек та їх динамічне завантаження.

Основною перевагою ONNX Runtime є сумісність із різними фреймворками розробки моделей, однак це потребує більшої технічної компетентності та врахування особливостей платформи, зокрема ряду обмежень щодо динамічного завантаження коду.

З урахуванням поточного стану технологій, вибір фреймворку повинен ґрунтуватися на пріоритетах конкретного проєкту: простота інтеграції, продуктивність, потреба в кросплатформеності, підтримка інструментів оптимізації, тощо. Результати цього дослідження можуть бути корисними як для розробників мобільних застосунків, так і для науковців, які працюють над впровадженням моделей машинного навчання у реальні мобільні середовища.

У подальших дослідженнях планується провести експериментальне тестування продуктивності моделей однакової архітектури, розгорнутих на різних фреймворках, а також оцінити споживання ресурсів та час виконання інференції на мобільних пристроях з iOS системою.

Література

1. He H. The State of Machine Learning Frameworks in 2019. *The Gradient*. URL: https://thegradient.pub/state-of-ml-frameworks-2019-pytorch-dominates-research-tensorflow-dominates-industry/ (дата звернення: 10.05.2025).

2. Nguyen G., Dlugolinsky S., Bobák M., Tran V., García Á. L., Heredia I., Malík P. and Hluchý L. Machine Learning and Deep Learning frameworks and libraries for large-scale data mining: a survey. *Artificial Intelligence Review*. 2019. Vol. 52, № 1. P. 77–124. URL: https://doi.org/10.1007/s10462-018-09679-z (дата звернення: 10.05.2025).

3. Rane N. L., Mallick S. K., Kaya O., Rane J. Tools and frameworks for machine learning and deep learning: A review.. *In Applied Machine Learning and Deep Learning: Architectures and Techniques.* 2024. P. 80-95. URL: https://doi.org/10.70593/978-81-981271-4-3_4 (дата звернення: 10.05.2025).

4. Mince, F., Dinh, D., Kgomo, J., Thompson, N., & Hooker, S. The Grand Illusion: The Myth of Software Portability and Implications for ML Progress. 2023. ArXiv, abs/2309.07181. URL: https://doi.org/10.48550/arXiv.2312.03120 (дата звернення: 10.05.2025).

5. Caffe2 Blog. [Електронний ресурс] – Режим доступу: https://caffe2.ai/blog (дата звернення: 10.05.2025).

6. TensorFlowLite is now LiteRT. [Електронний ресурс] – Режим доступу https://developers.googleblog.com/en/tensorflow-lite-is-now-litert/ (дата звернення: 10.05.2025).

7. ONNX. [Електронний ресурс] – Режим доступу

https://huggingface.co/docs/optimum/onnxruntime/concept_guides/onnx (дата звернення: 10.05.2025).

8. Bender. [Електронний ресурс] – Режим доступу https://xmartlabs.github.io/Bender/ (дата звернення: 10.05.2025).

9. Whole model saving & loading. [Електронний ресурс] – Режим доступу https://keras.io/2/api/models/model_saving_apis/model_saving_and_loading/ (дата звернення: 10.05.2025).

10. Torch ONNX. [Електронний ресурс] – Режим доступу: https://docs.pytorch.org/docs/2.7/onnx.html (дата звернення: 10.05.2025).

11. CoreML Tools. Source and Conversion Formats. [Електронний ресурс] – Режим доступу https://apple.github.io/coremltools/docs-guides/source/target-conversion-formats.html (дата звернення: 10.05.2025).

12. Thakkar M. Beginning machine learning in iOS: CoreML Framework. *Apress Berkeley, CA*. 2019. P.157. URL: https://doi.org/10.1007/978-1-4842-4297-1.

13. Gallagher A., Hollemans M., Tam A. Machine Learning by Tutorials (Second Edition): Beginning Machine Learning for Apple and IOS. *Amazon Digital Services LLC - KDP Print US*. 2021.

14. CreateML [Електронний ресурс] – Режим доступу https://developer.apple.com/documentation/createML (дата звернення: 10.05.2025).

References

1. He H. The State of Machine Learning Frameworks in 2019. *The Gradient*. URL: https://thegradient.pub/state-of-ml-frameworks-2019-pytorch-dominates-research-tensorflow-dominates-industry/ (last accessed: 10.05.2025).

2. Nguyen G., Dlugolinsky S., Bobák M., Tran V., García Á. L., Heredia I., Malík P. and Hluchý L. Machine Learning and Deep Learning frameworks and libraries for large-scale data mining: a survey. *Artificial Intelligence Review*. 2019. Vol. 52, № 1. P. 77–124. URL: https://doi.org/10.1007/s10462-018-09679-z (last accessed: 10.05.2025).

3. Rane N. L., Mallick S. K., Kaya O., Rane J. Tools and frameworks for machine learning and deep learning: A review.. *In Applied Machine Learning and Deep Learning: Architectures and Techniques*. 2024. P. 80-95. URL: https://doi.org/10.70593/978-81-981271-4-3_4 (last accessed: 10.05.2025).

4. Mince, F., Dinh, D., Kgomo, J., Thompson, N., & Hooker, S. The Grand Illusion: The Myth of Software Portability and Implications for ML Progress. 2023. ArXiv, abs/2309.07181. URL: https://doi.org/10.48550/arXiv.2312.03120 (last accessed: 10.05.2025).

5. Caffe2 Blog. URL: https://caffe2.ai/blog (last accessed: 10.05.2025).

6. TensorFlowLite is now LiteRT. URL: https://developers.googleblog.com/en/tensorflow-lite-is-now-litert/ (last accessed: 10.05.2025).

7. ONNX. URL: https://huggingface.co/docs/optimum/onnxruntime/concept_guides/onnx (last accessed: 10.05.2025).

8. Bender. URL: https://xmartlabs.github.io/Bender/ (last accessed: 10.05.2025).

9. Whole model saving & loading. URL: https://keras.io/2/api/models/model_saving_apis/model_saving_and_loading/ (last accessed: 10.05.2025).

10. Torch ONNX. URL: https://docs.pytorch.org/docs/2.7/onnx.html (last accessed: 10.05.2025).

11. CoreML Tools. Source and Conversion Formats. URL: https://apple.github.io/coremltools/docs-guides/source/target-conversion-formats.html (last accessed: 10.05.2025).

12. Thakkar M. Beginning machine learning in iOS: CoreML Framework. *Apress Berkeley, CA*. 2019. P.157. URL: https://doi.org/10.1007/978-1-4842-4297-1.

13. Gallagher A., Hollemans M., Tam A. Machine Learning by Tutorials (Second Edition): Beginning Machine Learning for Apple and IOS. *Amazon Digital Services LLC - KDP Print US*. 2021.

14. CreateML. URL: https://developer.apple.com/documentation/createML (last accessed: 10.05.2025).