

Yuliia Ihnatieva

Kharkiv National University of Radio Electronics, Kharkiv, Ukraine

TINY MACHINE LEARNING (TINYML) AS A TOOL FOR ENERGY-EFFICIENT DATA PROCESSING IN EDGE INTERNET OF THINGS DEVICES

Abstract. The paper examines TinyML, an approach that runs machine-learning inference directly on low-power microcontrollers and edge Internet of Things devices. The benefits of local inference include lower dependence on permanent connectivity, reduced cloud data transfer, lower latency for simple reactive scenarios, and greater control over raw sensor data. The main constraints are systematized: memory, compute capacity, energy budget, and hardware heterogeneity. A practical development cycle is proposed, covering task selection, data collection, model compression and quantization, memory profiling, latency measurement, and field validation.

Keywords: TinyML, machine learning, Internet of Things, edge computing, microcontrollers, energy efficiency.

Юлія Ігнат'єва

Харківський національний університет радіоелектроніки, Харків, Україна

МАЛОРЕСУРСНЕ МАШИННЕ НАВЧАННЯ (TINYML) ЯК ІНСТРУМЕНТ ЕНЕРГОЕФЕКТИВНОЇ ОБРОБКИ ДАНИХ У ПЕРИФЕРІЙНИХ ПРИСТРОЯХ ІНТЕРНЕТУ РЕЧЕЙ

Анотація. Досліджено підхід TinyML, що передбачає виконання моделей машинного навчання безпосередньо на малопотужних мікроконтролерах і периферійних пристроях Інтернету речей. Визначено переваги локального інференсу: зменшення залежності від постійного мережевого з'єднання, скорочення обсягу даних, що передаються в хмару, нижча затримка для простих сценаріїв реагування та кращий контроль над первинними даними. Систематизовано основні технічні обмеження TinyML — пам'ять, обчислювальну потужність, енергоспоживання й різноманітність апаратних платформ. Запропоновано практичний цикл розробки від вибору задачі та збору даних до квантування, профілювання пам'яті, вимірювання затримки й польового тестування.

Ключові слова: TinyML, машинне навчання, Інтернет речей, периферійні обчислення, мікроконтролери, енергоефективність.

Поширена архітектура Інтернету речей передбачає передавання значної частини даних із сенсорів до шлюзу або хмарної платформи, де виконується аналіз. Така модель зручна для централізованого керування, але має обмеження у випадках нестабільного зв'язку, високої частоти вимірювань або потреби в негайній локальній реакції. TinyML розвиває інший підхід: модель машинного навчання виконує інференс безпосередньо на мікроконтролері чи іншому мало-ресурсному пристрої біля джерела даних. Відповідно, до мережі передається не весь потік вимірювань, а подія, агрегований показник або результат класифікації.

За визначенням, яке закріпилося в технічній літературі, TinyML орієнтоване на перенесення задач машинного навчання на пристрої з жорсткими обмеженнями пам'яті, енергії й обчислювальної потужності (Warden & Situnayake, 2019; Ray, 2022). Типовими сценаріями є розпізнавання ключового слова,

виявлення аномалії в сигналі, класифікація простих зображень, контроль вібрацій, акустичний моніторинг або оцінювання стану обладнання. Важливо, що TinyML не намагається перенести на мікроконтролер будь-яку велику модель. Завдання має бути сформульоване так, щоб невелика модель давала достатню для практики якість за прийнятних ресурсних витрат.

Розробка TinyML-системи починається з оцінювання інформаційної цінності сигналу. Якщо рішення можна прийняти за декількома ознаками або коротким часовим вікном, локальна модель є доцільною. Якщо ж потрібне тривале зіставлення великої кількості джерел або складна генеративна обробка, краще залишити частину логіки на шлюзі чи сервері. Тому ефективна архітектура часто є гібридною: мікроконтролер виконує первинне розпізнавання, відсіює фонові дані й передає вищому рівню лише події, що потребують складнішого аналізу.

Найважливішим ресурсним обмеженням є пам'ять. Під час інференсу необхідно розмістити ваги моделі, проміжні тензори та буфери в обмеженому обсязі Flash і RAM. Через це практичними засобами оптимізації стають квантування чисел, зменшення кількості каналів і шарів, вибір компактних операторів та попередня обробка сигналу. Огляд TinyML підкреслює, що розвиток цього напрямку пов'язаний не лише зі зменшенням моделей, а й з інтеграцією інструментів, апаратних прискорювачів і стандартизованих методів оцінювання (Ray, 2022).

Таблиця 1. Критерії вибору TinyML для IoT-задачі

Критерій	Коли TinyML доцільне	Коли потрібен вищий рівень обчислень
Затримка	Потрібна реакція без мережевого запиту	Допустима серверна обробка
Обсяг даних	Потік великий, а подій мало	Потрібне зберігання всього потоку
Складність моделі	Компактна класифікація або аномалія	Велика мультимодальна чи генеративна модель
Зв'язок	Нестабільний або дорогий канал	Стабільний широкосмуговий канал

Джерело: складено автором на основі Warden and Situnayake (2019), Ray (2022) та Banbury et al. (2021).

Окреме значення має вимірювання не лише точності моделі. Для мікроконтролера модель із найвищою точністю може бути неприйнятною через затримку, розмір або енергоспоживання. Саме тому MLPerf Tiny сформував набір еталонних задач і запропонував розглядати разом точність, latency та energy, тобто показники, що характеризують компроміс між якістю та ресурсами (Banbury et al., 2021). Для практичного проекту до цього переліку варто додати максимальне використання RAM, розмір прошивки, час активної роботи процесора і стабільність у реальних умовах.

Важливо враховувати, що польові дані часто відрізняються від лабораторного набору. Мікрофон має інший шумовий профіль, датчик вібрації встановлено під іншим кутом, освітлення змінюється, а температура впливає на характеристики обладнання. Тому кінцева валідація повинна виконуватися на тому самому типі пристрою і в умовах, близьких до робочих. Для систем із тривалим життєвим циклом доцільно передбачити збір прикладів помилкових спрацьовувань, щоб періодично оновлювати модель централізовано, а потім розгортати нову версію на пристроях.

Перевагою локального інференсу є також контроль над сирими даними. Якщо пристрій аналізує аудіо або сенсорний потік локально і передає лише ознаку події, зменшується потреба зберігати первинні дані на сервері. Водночас це не робить систему автоматично безпечною: прошивка, модель і механізм оновлення повинні бути захищені від підміни. Таким чином, TinyML слід розглядати як частину повної IoT-архітектури, а не як ізольований алгоритм.

Отже, TinyML є практичним інструментом для задач, у яких рішення має прийматися близько до джерела даних і де передавання повного потоку в хмару є недоцільним. Найкращий результат забезпечує спільне проєктування моделі, апаратної платформи та сценарію експлуатації. Критеріями успіху мають бути не максимальна точність у лабораторному наборі, а достатня якість у реальних умовах, прогнозоване використання пам'яті, прийнятна затримка та стабільне енергоспоживання.

REFERENCES / СПИСОК ДЖЕРЕЛ

1. Banbury, C., Reddi, V. J., Torelli, P., Holleman, J., Jeffries, N., Kiraly, C., Montino, P., Kanter, D., Ahmed, S., Pau, D., Thakker, U., Torrini, A., Warden, P., Cordaro, J., Di Guglielmo, G., Duarte, J., Gibellini, S., Parekh, V., Tran, H., Tran, N., Wenxu, N., & Xuesong, X. (2021). MLPerf Tiny Benchmark. Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks, 1.
2. Ray, P. P. (2022). A review on TinyML: State-of-the-art and prospects. Journal of King Saud University - Computer and Information Sciences, 34(4), 1595–1623. <https://doi.org/10.1016/j.jksuci.2021.11.019>
3. Warden, P., & Situnayake, D. (2019). TinyML: Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers. O'Reilly Media.