УДК 004.8
DOI https://doi.org/10.32782/tnv-tech.2024.1.2

МЕТОДИКА ОПТИМІЗАЦІЇ АЛГОРИТМІВ МАШИННОГО НАВЧАННЯ ДЛЯ ВБУДОВАНИХ КІБЕРФІЗИЧНИХ СИСТЕМ

Бешлей М. І. *– доктор технічних наук, доцент кафедри телекомунікацій Національного університету «Львівська політехніка»*
ORCID ID: 0000-0002-7122-2319

Ковальчук О. В. *– студент кафедри телекомунікацій Національного університету «Львівська політехніка»*
ORCID ID: 0009-0003-8099-4956

Андрущак В. С. *– доктор філософії, старший викладач кафедри телекомунікацій Національного університету «Львівська політехніка»*
ORCID ID: 0000-0002-2185-0923

Бешлей Г. В. *– доктор філософії, доцент кафедри телекомунікацій Національного університету «Львівська політехніка»*
ORCID ID: 0000-0001-5392-3499

Розвиток цифрової трансформації та штучного інтелекту супроводжується зростаючою потребою впровадження вбудованих кіберфізичних систем, які використовують алгоритми машинного навчання, у критичну інфраструктуру, таку як енергетика, транспорт, виробництво та охорона здоров'я. Однак цей процес ускладнюється необхідністю розробки методів оптимізації для алгоритмів машинного навчання, які б забезпечили ефективну роботу вбудованих систем при обмежених обчислювальних ресурсах та забезпечили стійкість у критичних умовах. Це ставить перед розробниками завдання адаптації та оптимізації алгоритмів машинного навчання до обмежень, пов'язаних з обчислювальною потужністю, обсягом пам'яті, точністю прийняття рішення та енергоспоживанням. Саме тому, у роботі розроблено та оцінено методику оптимізації алгоритмів машинного навчання для використання у вбудованих кіберфізичних. Основний акцент дослідження робився на згорткових нейронних мережах, які використовуються у завданнях розпізнавання зображень. Використовуючи згорткові нейронні мережі, натреновані на датасеті Street View House Numbers (SVHN), дослідження демонструє, як моделі можуть ефективно виконувати задачі класифікації та розпізнавання цифр в реальному часі, при цьому оптимізуючи використання обмежених ресурсів вбудованих систем. Для оцінки ефективності пропонованої методики враховувалися критерії, такі як тривалість виконання для забезпечення точності вимірювань у кіберфізичних системах, енергоспоживання вбудованих систем, а також мінімізація дискового простору та оперативної пам'яті, необхідних для запуску моделей. В процесі дослідження методики застосовано методи оптимізації, такі як вагове скорочення та квантування, комбінація яких дає змогу зменшити розмір моделі та енергоспоживання без значної втрати точності. Оптимізовані моделі нейронних мереж протестовані на типовій вбудованій системі ESP32, демонструючи здатність до автономної роботи та розпізнавання об'єктів у реальному часі. Дослідження включало підготовку даних, розробку та тренування моделі, застосування різних варіантів оптимізації, та вимірювання їх впливу на кінцеві метрики системи. Використання фреймворка TensorFlow Lite дозволило адаптувати моделі для ефективного використання у вбудованих системах. Результати дослідження підтвердили ефективність запропонованої методики оптимізації, яка забезпечує зниження точності моделей на лише 2.1%, при цьому підвищуючи швидкість виконання на 30% та значно знижуючи енергоспоживання.

Ключові слова: *вбудовані системи, машинне навчання, оптимізація, енергоефективність, нейронна мережа, кіберфізична система.*

Beshley M. I., Kovalchuk O. V., Andrushchak V. S., Beshley H. V. Machine learning algorithms optimization methodology for embedded cyber-physical systems

The development of digital transformation and artificial intelligence is accompanied by a growing need to implement embedded cyber-physical systems that use machine learning algorithms in critical infrastructure such as energy, transportation, manufacturing, and healthcare. However, this process is complicated by the need to develop optimization methods for machine learning algorithms that would ensure the efficient operation of embedded systems with limited computing resources and ensure resilience in critical environments. This poses the challenge for developers to adapt and optimize machine learning algorithms to the limitations associated with computing power, memory space, decision accuracy, and power consumption. That is why we have developed and evaluated a methodology for optimizing machine learning algorithms for use in embedded cyber-physical systems. The main focus of the study was on convolutional neural networks used in image recognition tasks. Using convolutional neural networks trained on the Street View House Numbers (SVHN) dataset, the study demonstrates how models can effectively perform real-time digit classification and recognition tasks while optimizing the use of limited resources in embedded systems. To evaluate the effectiveness of the proposed methodology, criteria such as execution time to ensure the accuracy of measurements in cyber-physical systems, power consumption of embedded systems, and minimization of disk space and RAM required to run the models were taken into account. In the process of researching the methodology, optimization methods such as weight reduction and quantization were applied, the combination of which allows to reduce the model size and power consumption without significant loss of accuracy. The optimized neural network models were tested on a typical ESP32 embedded system, demonstrating the ability to operate autonomously and recognize objects in real time. The study included data preparation, model development and training, application of different optimization options, and measurement of their impact on the system's final metrics. The use of the TensorFlow Lite framework allowed us to adapt the models for effective use in embedded systems. The results of the study confirmed the effectiveness of the proposed optimization methodology, which ensures a decrease in model accuracy by only 2.1%, while increasing execution speed by 30% and significantly reducing power consumption.

Key words: *embedded systems, machine learning, optimization, energy efficiency, neural network, cyber-physical system.*

Постановка проблеми. У сучасному світі вбудовані кіберфізичні системи (ВКС) стають невід'ємною частиною багатьох секторів життєдіяльності, включаючи промислове виробництво, автомобільну промисловість, медицину, енергетику, смарт-будинки та міські інфраструктурні системи [1]. Ці системи інтегрують комп'ютерні алгоритми з фізичними процесами, надаючи можливість автоматизації, ефективного моніторингу та управління складними системами в реальному часі [2]. Інтеграція алгоритмів машинного навчання в ці системи відкриває нові можливості для автоматизації, адаптації та інтелектуалізації продуктів [3]. Проте, обмежені ресурси вбудованих систем, такі як обчислювальна потужність, пам'ять і енергоспоживання, ставлять перед розробниками завдання розробки ефективних методик оптимізації алгоритмів машинного навчання [4].

Аналіз останніх досліджень і публікацій. Розвиток штучного інтелекту, особливо через застосування глибоких нейронних мереж в компактних вбудованих системах, відкрив нові можливості для рішення складних задач. Проте, безпосереднє навчання цих моделей на вбудованих пристроях стикається з викликами, пов'язаними з обмеженими обчислювальними ресурсами та високою обчислювальною складністю задач. Відповідно, розвиток оптимізаційних методів для машинного навчання стає критичним. Один із основних підходів до оптимізації алгоритмів машинного навчання включає квантування, що дозволяє зменшити обсяг пам'яті моделі шляхом уніфікації значень параметрів, як показано в дослідженнях [5, 6]. Тензорна декомпозиція, описана в [7], подальше розширює можливості оптимізації шляхом розкладу мережевих матриць на менші компоненти, спрощуючи обчислення та знижуючи вимоги до пам'яті. Новаторський підхід LightweightNet, представлений у [8], пропонує структуру дистиляції архітектури,

яка замінює повнорозмірні шари на більш легкі еквіваленти, зменшуючи таким чином вимоги до обчислювальних ресурсів та пам'яті. Сучасні стратегії також зосереджуються на прискоренні моделей через вдосконалення згорткових операцій або архітектур, як зазначено в [9]. Прунінг (скорочення) ваг у попередньо навчених моделях згорткових нейронних мереж (CNN) виступає ключовою технікою, яка полягає у видаленні ваг з мінімальною значущістю, що не суттєво впливають на загальну ефективність моделі. Після видалення таких ваг модель проходить процес детального налаштування з використанням регуляризації, метою якого є підвищення загальної продуктивності. Цей підхід було продемонстровано у роботі [10], підкреслюючи його важливість для оптимізації моделей CNN.

Незважаючи на те, що розглянуті методи спрощують розгортання алгоритмів машинного навчання на вбудованих системах та забезпечують баланс між ефективністю нейронних мереж і вимогами до обчислювальних ресурсів, досі залишається відкритим питання розробки методики визначення критеріїв для оцінки значущості параметрів моделей нейронних мереж, що дасть змогу адаптувати оптимізаційні стратегії до конкретних вимог задачі та обчислювального середовища.

Таким чином, **метою роботи** є розробка методики адаптивної оптимізації алгоритмів машинного навчання таким чином, щоб вони могли ефективно працювати в умовах обмежених ресурсів на обладнанні вбудованих систем в залежності від цільового призначення, забезпечуючи при цьому достатню точність, енергоефективність та швидкість роботи. Це включає в себе зменшення розміру моделі, оптимізацію обчислень та зниження енергоспоживання без значної втрати продуктивності.

Виклад основного матеріалу дослідження. Для дослідження пропонованої методики оптимізації алгоритмів машинного навчання необхідно враховувати конкретні приклади їх використання для кіберфізичних систем. Алгоритми машинного навчання, зокрема нейронні мережі, використовуються для вирішення різноманітних завдань, таких як класифікація об'єктів, передбачення результатів, виявлення шаблонів тощо. Навчання моделей зазвичай відбувається на основі доступних даних, де алгоритм адаптується до вхідних даних, шукаючи залежності та взаємозв'язки, які допомагають вирішувати конкретні завдання.

Для оцінки ефективності моделі нейронних мереж у кіберфізичних вбудованих системах та дослідження пропонованої методики оптимізації алгоритмів машинного навчання, важливо врахувати наступні критерії:

- Модель повинна мати достатньо тривалий час виконання, щоб забезпечити точність вимірювання у кіберфізичних системах. Це є ключовим параметром для ефективної оптимізації енергоефективності системи.
- Кінцева модель повинна займати якнайменше дискового простору та оперативної пам'яті, щоб бути придатною для запуску в умовах вбудованих систем. Це дозволить забезпечити ефективне використання обмежених ресурсів, які характерні для кіберфізичних систем.

Отже, після врахування вищезазначених вимог обрано конкретне прикладне дослідження, що полягає у впровадженні згорткової нейронної мережі для розпізнавання зображень номерних знаків на будинках, з використанням моделі, яка була попередньо натренована на датасеті Strret View House Numbers (SVHN) [11]. Згорткові нейронні мережі добре підходять для розпізнавання об'єктів, оскільки вони враховують просторову ієрархію, характерну для зображень. Обрана модель навчалася на різноманітних даних, включаючи зображення номерних знаків на будинках, що відображається у реальних умовах спостереження.Типову вбудовану систему, в якій використовуються моделі нейронних мереж показана на рис. 1.

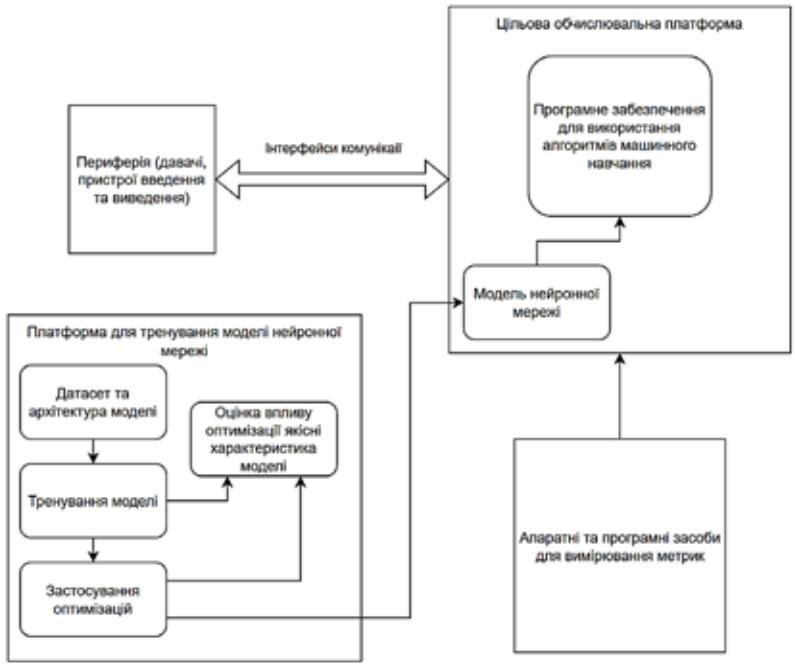

Рис. 1. Узагальнена схема вбудованої кіберфізичної системи з використанням нейронних мереж для обробки зображень в реальному часі

Така система включає в себе кінцевий пристрій з периферією, на якому виконуються всі обчислення використовуючи модель та системи, на якій відбувається побудова та тренування моделі. Після оптимізації нейромережевої моделі її розгортають на кінцевому пристрої. Розпізнавання об’єктів у реальному часі відбувається безпосередньо на пристрої, що дозволяє йому оперативно реагувати на події, не покладаючись на зовнішні обчислювальні ресурси. Вбудована система працює автономно, демонструючи потенціал нейронних мереж у створенні інтелектуальних периферійних пристроїв.

Для проведення дослідження розроблено методику оптимізації моделі нейронної мережі, яка включає в себе наступні етапи (рис. 2):

- Підбір датасету та підготовка даних для тренування. Для подальшого тренування та перевірки і валідації даних з датасету проведено попередню обробку даних, а саме конвертовано в кольоровий простір градацій сірого.
- Розробка та тренування моделі.
- Застосування різних варіантів оптимізації і формування моделей для тестування.
- Вимірювання впливу методів оптимізації на метрики кінцевої системи та формування результатів порівняння.

В процесі проведення дослідження використовувався фреймворк для машинного навчання від Google Tensorflow Lite, оскільки він добре адаптований для використання у вбудованих системах на основі різних архітектур та розповсюджується з набором інструментів для впровадження оптимізацій у вже існуючі моделі нейронних мереж.

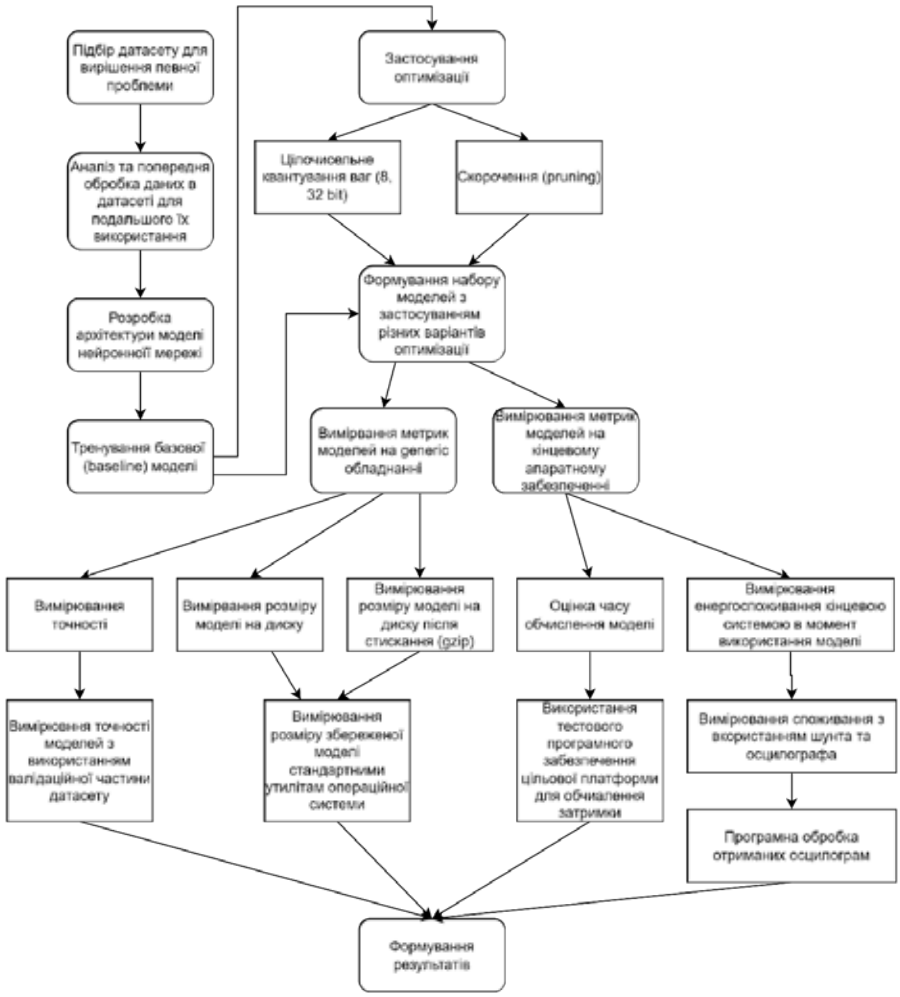

Рис. 2. Методика дослідження впливу оптимізації моделей нейронних мереж на показники вбудованих систем

Архітектура моделі показана на рис. 3. На вхід приймаються напівтонові зображення розміром 32×32 (зображені червоним кольором). Згодом ці зображення проходять ряд операцій. Спочатку застосовуються три згорткові шари, кожен з яких супроводжується Batch Normalization, Activation та MaxPooling2D, в результаті чого утворюється 12 функціональних згорткових шарів. Після операцій згортки застосовуються 3 шари Dense за якими слідує BatchNormalization та Activation.

Ця архітектура, що характеризується інтеграцією згорткових шарів з функціями пакетної нормалізації, активації та максимального об'єднання, полегшує вилучення ієрархічних ознак із вхідних зображень. Наступні щільні шари слугують для фіксації складних взаємозв'язків у представленнях ознак, що в кінцевому підсумку призводить до точної класифікації за визначеними вихідними категоріями. Включення пакетної нормалізації допомагає стабілізувати і прискорити процес навчання, нормалізуючи активації між шарами, пом'якшуючи такі проблеми,

як внутрішній зсув коваріацій. Крім того, використання функцій активації ReLU вносить нелінійність, дозволяючи моделі вивчати складні закономірності в даних.

Описана архітектура особливо добре підходить для задач класифікації зображень, оскільки вона використовує можливості ієрархічного виділення ознак за допомогою згорткових шарів, одночасно вводячи нелінійність і нормалізацію для підвищення стабільності та ефективності навчання.

Layer (type)	Output Shape	Param
input_image (InputLayer)	[(None, 32, 32, 3)]	0
conv_0 (Conv2D)	(None, 30, 30, 16)	432
bn_conv_0 (BatchNormalization)	(None, 30, 30, 16)	64
conv_act_0 (Activation)	(None, 30, 30, 16)	0
pool_0 (MaxPooling2D)	(None, 15, 15, 16)	0
conv_1 (Conv2D)	(None, 13, 13, 16)	2304
bn_conv_1 (BatchNormalization)	(None, 13, 13, 16)	64
conv_act_1 (Activation)	(None, 13, 13, 16)	0
pool_1 (MaxPooling2D)	(None, 6, 6, 16)	0
conv_2 (Conv2D)	(None, 4, 4, 24)	3456
bn_conv_2 (BatchNormalization)	(None, 4, 4, 24)	96
conv_act_2 (Activation)	(None, 4, 4, 24)	0
pool_2 (MaxPooling2D)	(None, 2, 2, 24)	0
flatten (Flatten)	(None, 96)	0
dense_0 (Dense)	(None, 42)	4032
bn_dense_0 (BatchNormalization)	(None, 42)	168
dense_act_0 (Activation)	(None, 42)	0
dense_1 (Dense)	(None, 64)	2688
bn_dense_1 (BatchNormalization)	(None, 64)	256
dense_act_1 (Activation)	(None, 64)	0
output_dense (Dense)	(None, 10)	650
output_softmax (Activation)	(None, 10)	0

Рис. 3. Архітектура моделі нейронної мережі, яка використовувалась для дослідження

Тренування моделі виконувалось з допомогою алгоритму машинного навчання під наглядом. Для тренування використовувались наступні гіперпараметри:

- Функція втрат CategoricalCrossentropy
- Оптимізатор Adam (швикість навчання 0.3*10^-3, beta_1=0.9, beta_2=0.999, epsilon=1*10^-07)
- Розмір серії (batch size) 1024

Результат тренування baseline моделі нейронної мережі без оптимізації показано на рис. 4 у вигляді ROC кривих (Receiver Operating Characteristic curve) для кожного з класів класифікатора. ROC-крива – це графічне представлення діагностичної здатності бінарного класифікатора при всіх можливих порогах класифікації. Вона відображає взаємозв'язок між часткою істинно позитивних результатів (True Positive Rate, TPR) та часткою хибно позитивних результатів (False Positive Rate, FPR) для різних порогових значень. Кожна крива представляє собою ефективність моделі у класифікації окремого класу . Криві мають різні кольори для ідентифікації, і кожна крива має підпис, який показує площу під кривою (Area under the curve, AUC).

Як бачимо, що загальна точність оригінальної моделі без оптимізації становить 89.7%, а показники AUC варіюється від 98.4% до 99.4% для різних класів. Це високі значення AUC, що свідчить про високу здатність моделі розрізняти між класами та можливість проведення попередньої оптимізації.

У контексті оптимізації нейронних мереж для зменшення витрат ресурсів можна розглянути два популярні підходи: вагове скорочення (model pruning) та квантування (quantization). Вагове скорочення це процес видалення несуттєвих ваг у нейронній мережі. Квантування полягає в зменшенні точності ваг моделі, зазвичай переходячи від 32-бітних змінних з плаваючою комою до 8-бітних

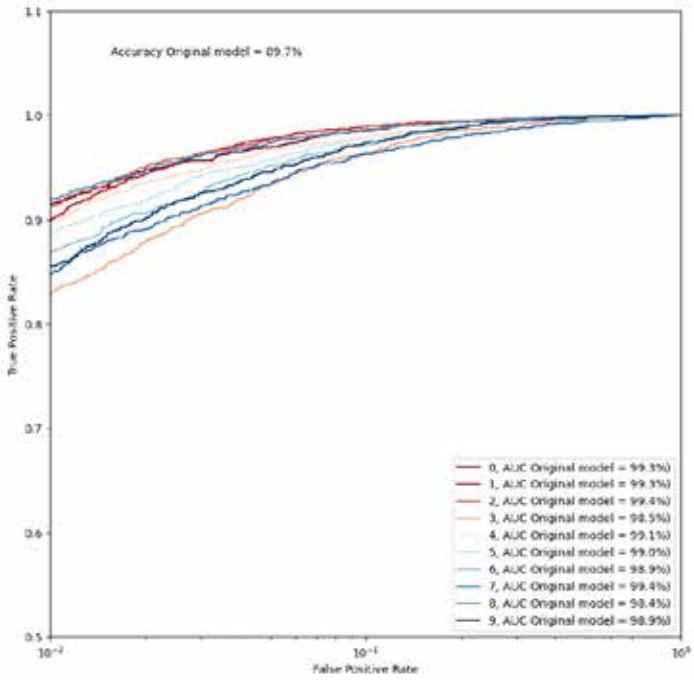

Рис. 4. Криві ROC натренованої baseline моделі (без оптимізації)

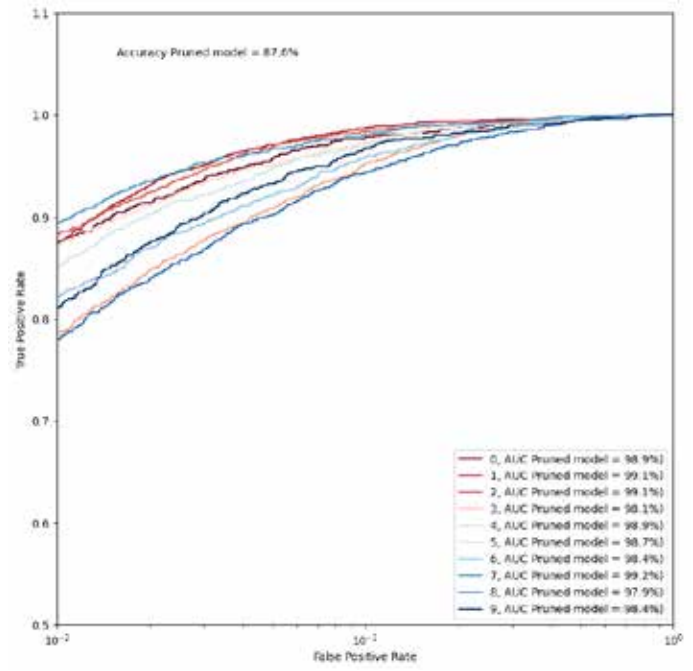

Рис. 5. Криві ROC натренованої моделі після застосування скорочення

цілочисельних значень. Дані підходи можуть бути застосовані окремо або разом, щоб досягти оптимального балансу між точністю моделі та ефективністю використання ресурсів.

На рис. 5 представлені ROC-криві для моделі після застосування методу вагового скорочення (pruning). Подібно до попереднього графіка, кожна лінія відповідає за певний клас в багатокласовій класифікаційній задачі. Після скорочення точність моделі становить 87.6%, що трохи нижче, ніж було зазначено для оригінальної моделі (89.7%). Значення AUC для різних класів після скорочення коливаються від 97.9% до 99.1%. Ці значення трохи нижче, ніж у оригінальної моделі, але все ще знаходяться на високому рівні, що свідчить про добру класифікаційну здатність моделі навіть після скорочення.

Ефект від застосування скорочення можна побачити на гістограмі розподілу ваг. Суть скорочення полягає у заміні ваг із значенням близько нуля, вагами з нульовим значенням. Вплив такої оптимізації на модель можна оцінити змінами в гістограмах розподілу ваг до та після застосування оптимізації. Дані гістограми показані на рис. 6 та рис. 7 відповідно.

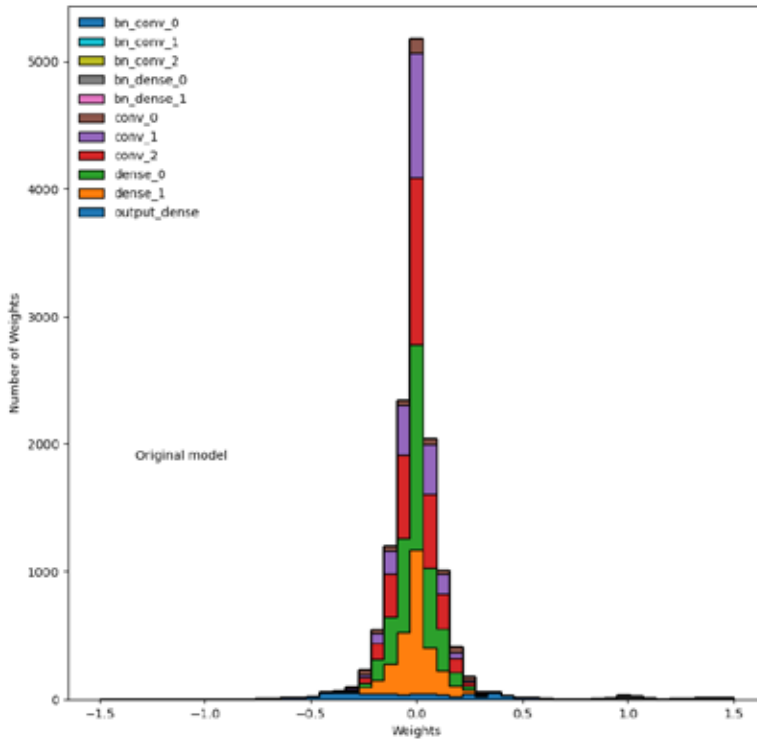

Рис. 6. Гістограма розподілу ваг моделі до застосуваня оптимізації

Для застосування оптимізації квантування використовується метод квантування після тренування, який виконується засобами Tensorflow Lite. В ході дослідження квантування виконується з наступними параметрами:

- Квантування ваг до 32-розрядних цілих чисел.

● Квантування ваг та входів/виходів функцій активації до 8-розрядних цілих чисел.

В результаті застосування оптимізацій, отримано наступний набір моделей для тестування:

● cnn_not_optimised-no_opt.tflite – модель без застосування оптимізації: без скорочення та без квантування;

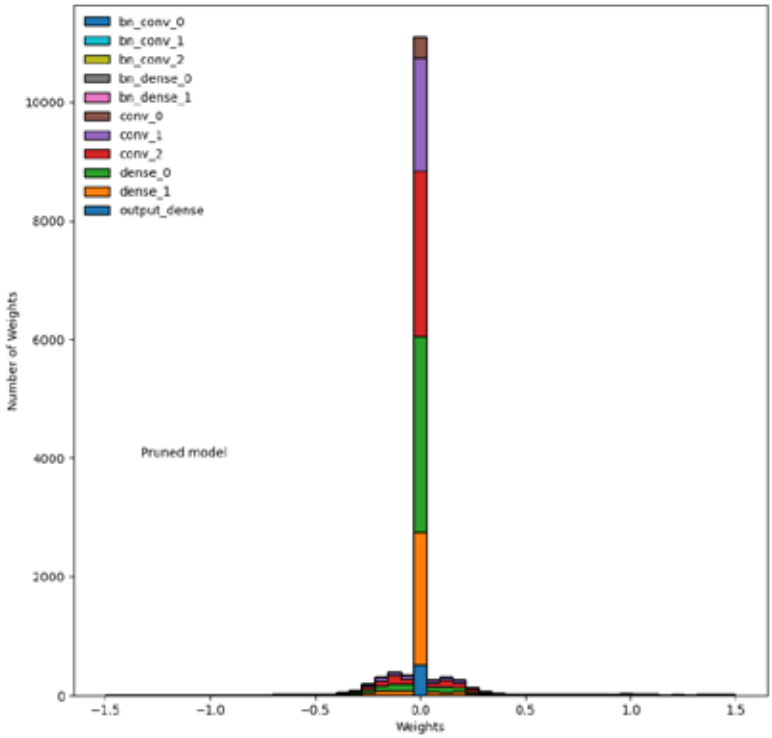

Рис. 7. Гістограма розподілу ваг моделі після застосуваня скорочення

● cnn_not_optimised-default_opt.tflite – модель з цілочисельним квантуванням з можливістю застосуваня операторів які використовують числа з плаваючою комою;

● cnn_not_optimised-int8_quant.tflite – модель з застосуванням квантування до ы8-розрядних цілих чисел;

● cnn_pruned-default_opt.tflite – модель, в якій застосовано оптимізації скорочення;

● cnn_pruned-int8_quant.tflite – модель, в якій застосовано оптимізації скорочення та цілочисельне квантування до 8 розрядних чисел;

● cnn_pruned-no_opt.tflite – модель, в якій застосовано оптимізації скорочення та цілочисельне квантування.

Для проведення аналізу використання алгоритмів навчання, а саме нейронних мереж у контексті вбудованих систем, важливо виокремити кілька чинників, які впливають на використання апаратного забезпечення вбудованих систем в цілому:

- Вимірювання точності моделі. Дану метрику важливо відслідковувати, оскільки вона показує вплив змін в моделі внаслідок застосування оптимізації на точність її роботи.
- Вимірювання розміру моделі до та після стискання.
- Час виклику моделі (inference time).
- Кінцевий розмір моделі на диску (у постійній пам’яті).
- Розмір моделі після стискання.
- Споживана потужність (динамічна).
- Усереднена потужність.

Вимірювання точності та обсягу моделі не вимагає безпосередньої взаємодії з цільовою платформою, оскільки ці характеристики є незалежними від обладнання, на якому модель застосовується. Таким чином, найбільш раціональним варіантом було провести ці вимірювання на тій самій платформі, де відбувається навчання моделі. Водночас, оцінки часу виконання та енергоспоживання проводились в умовах, що максимально наближені до реального використання, а тому – безпосередньо на вбудованій цільовій системі. Для проведення тестування даних параметрів розроблено тестове програмне забезпечення, в якому робиться кілька викликів розрахунку моделі з заданою частотою та розраховується середнє значення часу виконання. Приклад результату роботи тестової утиліти показано на рис. 8.

```
32  INFO: Running benchmark for at least 1 iterations and at least 0.5 seconds but terminate if exceeding 150 seconds.
33  Running benchmark for at least 50 iterations and at least 1 seconds but terminate if exceeding 150 seconds.
34
35  INFO: ==============Summary of All Runs w/ Different Performance Options==============
36  INFO: single core cpu: count=52 first=64710 curr=64066 min=64066 max=64710 avg=64079.2 std=88s
```

Рис. 8. Приклад виводу утиліти для аналізу затримки виконання моделей Tensorflow Lite встановленої на мікроконтролер

На рис. 9 показано результати вимірювання середнього часу виконання нейронної мережі для різних конфігурацій моделі на мікроконтролері ESP32 (позначено як "single core cpu" на легенді). Конфігурації моделі, які порівнюються, включають комбінації з ваговим скороченням та квантуванням. З лівої сторони графіка видно, що певні конфігурації моделі мають значно вищий час виконання, що вказує на менш оптимізовані версії моделі. В центрі графіка середній час виконання знижується, що свідчить про ефективність виконання оптимізованих версій моделей. Потім час виконання знову підвищується для останніх конфігурацій, вказаних на графіку. Отриманні результати можуть бути використані для визначення найбільш ефективної конфігурації моделі з точки зору часу виконання для мікроконтролерів, таких як ESP32.

Згідно запропонованої методики для вимірювання енергоспоживання мікроконтролера під час роботи моделі нейронної мережі використовується схема з шунтовим резистором на 0.05 Ом, який встановлено послідовно у ланцюг живлення плати з мікроконтролером відповідно. Вимірювання напруги, що впадає на шунтовому резисторі, проводиться за допомогою цифрового осцилографа, який може фіксувати та зберігати зняті показники на диск для наступного перетворення у значення споживаної потужності (приймаючи, що напруга живлення становить 5 В) та аналізу. Графіки енергоспоживання, які були отримані, представлені на рис. 10. На них чітко видно піки споживання струму в ланцюзі, які виникають у моменти активного використання моделі.

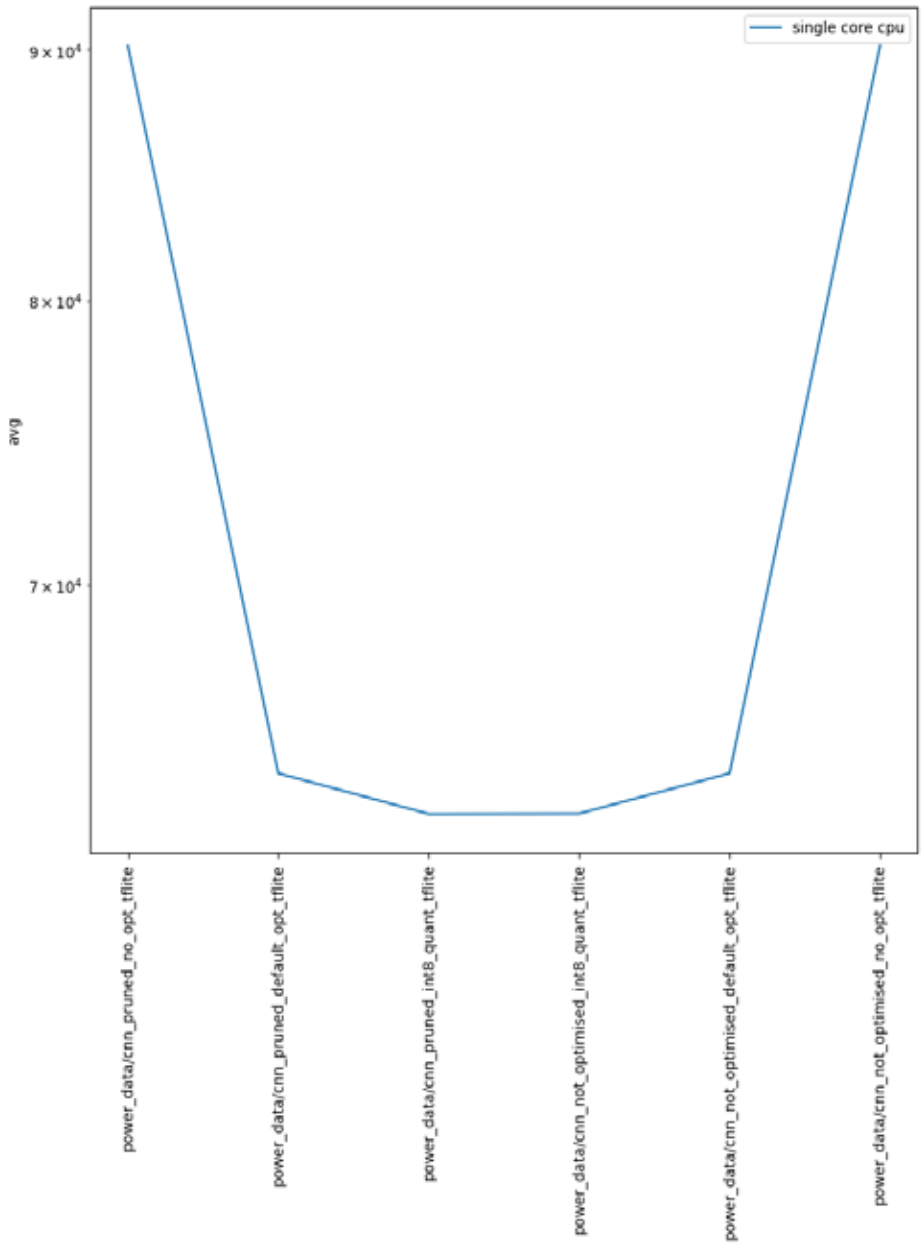

Рис. 9. Результат вимірювання часу виконання моделі на мікроконтролері ESP32

Провівши вимірювання для кожного з варіантів отриманих моделей нейронної мережі та узагальнивши результати сформовано таблицю 1, яка містить результати всіх вимірювань досліджуваних метрик.

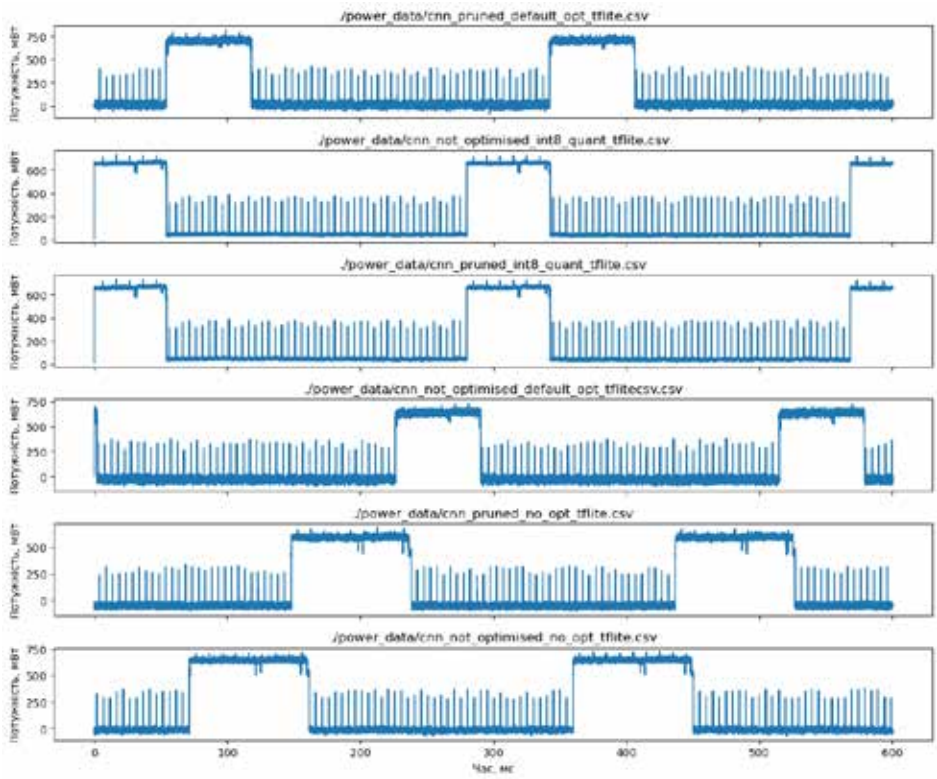

Рис. 10. Споживана потужність вбудованої системи з застосуванням різних варіантів оптимізації моделі

Таблиця 1

Узагальнення результатів впливу оптимізації нейронних мереж на метрики вбудованої системи

Тип оптимізації	Розмір моделі / розмір стиснутої моделі, КБ	Середня затримка виконання, мс	Середнє значення споживаної потужності, мВт
Без оптимізації	59.5 / 54.1	90.1	1946
Цілочисельне квантування	21.3 / 15.6	64.7	1237
Квантування до 8-розрядних цілих чисел	21 / 15.4	62.8	1384
Скорочення	59.5 / 19.3	90.1	1769
Скорочення + цілочисельне квантування	21.3 / 8.3	64	1468
Скорочення + квантування до 8-розрядних цілих чисел	21 / 8.2	62.8	1384

З рис. 9 можна зробити висновок, що внаслідок квантування час виклику нейронної мережі знизився з ~90мс до квантування до ~60мс після квантування до цілочисельних типів так і після квантування до 8 розрядних цілочисельних типів. Також слід зауважити про співпадіння величини часу виконання, яка рівна тривалості імпульсу на осцилограмі зі значеннями виміряними з допомогою тестового програмного забезпечення.

Паралельно з цим, дослідження скорочених ваг у моделі нейронної мережі, як на прикладі cnn_pruned-no_opt.tflite, висвітлює інший аспект оптимізації це зменшення розміру моделі. Скорочення, як метод, систематично усуває надлишкові зв'язки в нейронних мережах, в результаті чого моделі займають значно менше місця на флеш-пам'яті. Це особливо помітно при порівнянні вимог до пам'яті скорочений моделей (cnn_pruned-no_opt.tflite) з їх неоптимізованими аналогами (cnn_not_optimised-no_opt.tflite). Стиснені моделі не лише економлять дорогоцінний простір у сховищі, але й сприяють прискореному завантаженню та розгортанню моделей на вбудованих системах з обмеженими ресурсами.

Таким чином запропонована методика здатна гнучко адаптуватися до оптимізації залежно від специфічних потреб. Вона включає в себе стратегію, яка об'єднує квантування та скорочення для оптимізації нейронних мереж, що показано на прикладах моделей, таких як "cnn_not_optimised-int8_quant.tflite" та "cnn_pruned-int8_quant.tflite". Це поєднання дає змогу втілити в одній моделі переваги обох технік оптимізації, що робить її значно швидшою та компактнішою порівняно з її первісною версією. Такі моделі не тільки прискорюють процес виведення, але й ефективно знижують використання пам'яті. Це забезпечує комплексне вдосконалення ефективності нейронних мереж, зокрема для застосування в умовах обмежених ресурсів вбудованих систем.

Висновки. У роботі запропоновано методику оптимізації алгоритмів машинного навчання для вбудованих кіберфізичних систем. Акцент зроблено на значущості програмних оптимізацій для покращення функціональності вбудованих систем, використовуючи нейронні мережі. Для аналізу потенціалу оптимізацій розглянуто архітектуру типової вбудованої системи, модель нейронної мережі, визначено критичні метрики для моніторингу оптимізацій, ідентифіковано ряд потенційних оптимізацій, що можуть знизити вимоги до ресурсів системи. Також визначено набір середовищ для тестування оптимізацій, охоплюючи широкий спектр популярних програмно-апаратних платформ. На основі експериментальних даних досягнуто висновку, що запропонована методика оптимізації є ефективною, оскільки мінімально впливає на точність моделей нейронних мереж, знижуючи її лише на 2.1%, при цьому підвищуючи швидкість виконання моделі на 30%, що у свою чергу значно знижує енергоспоживання.

ПОДЯКА

Дана стаття підготовлена завдяки грантової підтримки Національного Фонду Досліджень України, реєстраційний номер проєкту 2022.01/0009 «Оцінювання та прогнозування загроз відбудові та сталому функціонуванню об'єктів критичної інфраструктури» за конкурсом «Наука для відбудови України у воєнний та повоєнний періоди».

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ:

1. A Cyber-Physical Systems Paper Survey About the Concept, Architecture and Challenges for the Deployment within the Concept of Industry 4.0 / M. Melicher et al. *Research Papers Faculty of Materials Science and Technology Slovak University of Technology*. 2019. Vol. 27, no. 45. P. 49–54.

2. Cyber-Physical Power System (CPPS): A Review on Modeling, Simulation, and Analysis With Cyber Security Applications / R. V. Yohanandhan et al. *IEEE Access*. 2020. Vol. 8. P. 151019–151064.

3. Conrad C., Al-Rubaye S., Tsourdos A. Intelligent Embedded Systems Platform for Vehicular Cyber-Physical Systems. *Electronics*. 2023. Vol. 12, no. 13. P. 2908.

4. Batzolis E., Vrochidou E., Papakostas G. A. Machine Learning in Embedded Systems: Limitations, Solutions and Future Challenges. *2023 IEEE 13th Annual Computing and Communication Workshop and Conference (CCWC)*, Las Vegas, NV, USA, 8–11 March 2023. 2023.

5. Wiedemann S., Muller K.-R., Samek W. Compact and Computationally Efficient Representation of Deep Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*. 2020. Vol. 31, no. 3. P. 772–785.

6. Tung F., Mori G. Deep Neural Network Compression by In-Parallel Pruning-Quantization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2020. Vol. 42, no. 3. P. 568–579.

7. Compressing by Learning in a Low-Rank and Sparse Decomposition Form / K. Guo et al. *IEEE Access*. 2019. Vol. 7. P. 150823–150832.

8. LightweightNet: Toward fast and lightweight convolutional neural networks via architecture distillation / T.-B. Xu et al. *Pattern Recognition*. 2019. Vol. 88. P. 272–284.

9. ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices / X. Zhang et al. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Salt Lake City, UT, 18–23 June 2018. 2018.

10. EIE: Efficient Inference Engine on Compressed Deep Neural Network / S. Han et al. *Computer Architecture News*. 2016. Vol. 44(3). P. 243–254.

11. Yang H., Yao H. Street View House Number Identification Based on Deep Learning. *International Journal of Advanced Network, Monitoring and Controls*. 2019. Vol. 4, no. 3. P. 47–52.

REFERENCES:

1. Melicher, M., Šišmišová, D., Vachálek, J., & Belavý, C. (2019). A Cyber-Physical Systems Paper Survey About the Concept, Architecture and Challenges for the Deployment within the Concept of Industry 4.0. *Research Papers Faculty of Materials Science and Technology Slovak University of Technology*, *27*(45), 49–54.

2. Yohanandhan, R. V., Elavarasan, R. M., Manoharan, P., & Mihet-Popa, L. (2020). Cyber-Physical Power System (CPPS): A Review on Modeling, Simulation, and Analysis With Cyber Security Applications. *IEEE Access*, *8*, 151019–151064.

3. Conrad, C., Al-Rubaye, S., & Tsourdos, A. (2023). Intelligent Embedded Systems Platform for Vehicular Cyber-Physical Systems. *Electronics*, *12*(13), 2908.

4. Batzolis, E., Vrochidou, E., & Papakostas, G. A. (2023). Machine Learning in Embedded Systems: Limitations, Solutions and Future Challenges. In *2023 IEEE 13th Annual Computing and Communication Workshop and Conference (CCWC)*. IEEE.

5. Wiedemann, S., Muller, K.-R., & Samek, W. (2020). Compact and Computationally Efficient Representation of Deep Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*, *31*(3), 772–785.

6. Tung, F., & Mori, G. (2020). Deep Neural Network Compression by In-Parallel Pruning-Quantization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *42*(3), 568–579.

7. Guo, K., Xie, X., Xu, X., & Xing, X. (2019). Compressing by Learning in a Low-Rank and Sparse Decomposition Form. *IEEE Access*, *7*, 150823–150832.

8. 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.

9. Zhang, X., Zhou, X., Lin, M., & Sun, J. (2018). ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

10. Han, S., Liu, X., Mao, H., Pu, J., Pedram, A., Horowitz, M. A., & Dally, W. J. (2016). EIE: Efficient inference engine on compressed deep neural network. *Computer Architecture News*, *44*(3), 243–254.

11. Yang, H., & Yao, H. (2019). Street View House Number Identification Based on Deep Learning. *International Journal of Advanced Network, Monitoring and Controls*, *4*(3), 47–52.