УДК 519.65

ПОРІВНЯННЯ ПРОДУКТИВНОСТІ ПОПУЛЯРНИХ ЗГОРТКОВИХ МОДЕЛЕЙ ДЛЯ ДЕТЕКТУВАННЯ МЕХАНІЧНИХ ОБ'ЄКТІВ

Д.М. Миронюк[1], Б.Я. Благітко[1], І.М. Заячук[2]

[1]Львівський національний університет ім. Івана Франка, вул. Університетська, 1, Львів, 79000, Україна,
[2]Центр математичного моделювання ІППММ ім. Я. С. Підстригача НАН України, вул. Дж. Дудаєва, 15, Львів, 79005, Україна

Досліджено і проаналізовано особливості сучасних платформ на основі графічних карт для глибинного навчання. За результатами проведеного аналізу оцінено швидкісні параметри роботи різних моделей. З досліджених запропоновано ефективні моделі детектування механічних об'єктів на різних моделях відеокарт. На прикладах продемонстровано ефективність роботи попередньо навчених моделей програмного каркасу для глибинного навчання PyTorch. Дано рекомендації що до вибору графічних процесорів для ефективної роботи над масивом даних.

Ключові слова: *графічний процесор, згорткова модель, PyTorch.*

Постановка проблеми. Сьогодні існує достатньо багато ефективних засобів роботи з високими об'ємами різних типів даних (графічних, текстових та ін.). Одним із найбільш ефективних засобів обробки даних (особливо графічних) в останні роки стали графічні процесори (GPU), перевагою яких є високі показники швидкості під час роботи над масивом даних під однією операцією. Особливості, які визначають таку швидкодію наведено у таблиці 1.

Таблиця 1

Порівняльна характеристика обчислень на CPU та GPU

CPU	GPU
Багатоядерний процесор	Мультипроцесор
MIMD	SIMD
Однопотоковість	Багатопотоковість
Великий кеш	Невеликий кеш
Швидка пам'ять	Повільна пам'ять
Випадковий доступ до пам'яті	Послідовний доступ до пам'яті

Як бачимо, перевагою графічного процесора над центральним процесором є насамперед його кількість процесорних ядер та їхнє швидке перемикання між потоками (для обробки послідовних сусідніх текстур). Оскільки такі карти використовуються для обробки зображення, яке виводиться на монітор комп'ютера, то їхня система передбачає велику кількість арифметико-логічних пристроїв, що прив'язуються до спільної пам'яті. Велика кількість власної

пам'яті на графічному процесорі зумовлена основною його основною спеціалізацією — рендерингом текстур. Так само пояснюється відсутність великої кількості кеш-пам'яті. Доступ до пам'яті організовано послідовно для послідовності у генеруванні зображення.

Основне призначення ЦП — швидке виконання великої кількості команд. Для цього там встановлено кеш-пам'ять різних рівнів та організовано чергу операцій. Кеш-пам'ять центрального процесора є набагато швидшою, ніж на графічному процесорі, а максимальна швидкодія потоку команд забезпечується організацією черги та паралелізмом між обчислювальними ядрами.

На практиці отримуємо набагато швидше виконання простих математичних операцій над великими масивами даних на ГП, що розширює застосування графічних процесорів на велику кількість задач (майнінг, глибинне навчання, прискорення матричних операцій та ін). З іншого боку, якщо об'єм даних недостатній або є велика кількість різних операцій до застосування, центральний процесор показує вищу швидкість виконання (на обмін даними між ЦП і ГП також витрачається певний час, який у таких випадках суттєво впливає на загальну швидкість виконання). Способи організації обчислень на ЦП та ГП наведено на рисунку 1.

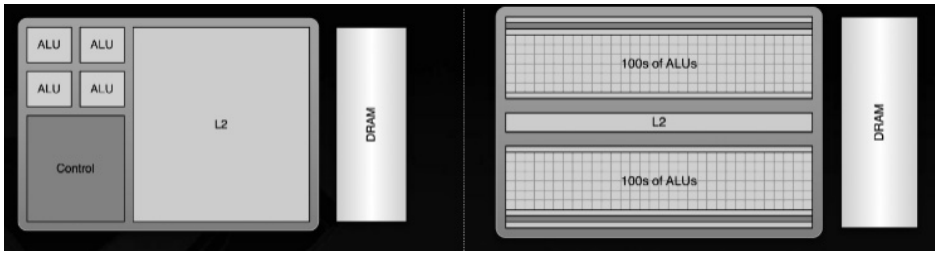


Рис.1. Способи організації обчислень на ЦП та ГП

1. Огляд сучасних платформ для виконання обчислень на графічних процесорах

Для організації обчислень на графічних процесорах сучасні виробники графічних адаптерів розробили власні системи, які вбудовуються у драйвер самого графічного процесора та дозволяють без великих зусиль використовувати всі переваги графічних процесорів. Таких платформ можна виділити дві: Nvidia CUDA та AMD ROCm

1.1. Платформа Nvidia CUDA

Nvidia CUDA – це програмно-апаратна платформа для виконання обчислень на графічних процесорах, яка була розроблена та підтримується компанією Nvidia. Містить у собі широкий набір інструментів для виконання різноманітних загальних обчислювальних задач, а також пакети для розпаралелення та вирішення конкретних задач. У програмному вигляді вона представлена у вигляді розширення мови програмування С. Для траснляції

коду з цього розширення використовується власний компілятор nvcc, який був створений на основі відкритого компілятора Open64.

Основні характеристики CUDA:

- основне уніфіковане рішення для виконання загальних задач з використанням графічних процесорів Nvidia;
- великий набір рішень, що підтримуються;
- великий набір стандартних бібліотек для чисельного аналізу (включно з BLAS та FFT);
- оптимізована для ефективного обміну даних між центральним та графічним процесором;
- взаємодія з графічним API OpenGL та DirectX;
- можливість низькорівневої розробки;
- підтримка широкого набору операційних систем;
- висока документованість і широкий набір прикладів коду для початківців.

1.2 Апаратно-програмна платформа AMD ROCm

AMDm ROCm – достатньо нова апаратно-програмна платформа (перший вихід було анонсовано у 2016 році) для виконання загальних задач на графічних процесорах. Містить у собі широкий набір рішень та API, які роблять використання графічних процесорів більш абстрагованими. Позиціонується як найбільш уніфікована платформа для розгортання, розширення та підтримки рішень з використанням графічних процесорів, однією з основних характеристик якої можна назвати підтримку контейнерних рішень. Всі рішення, використовувані у функціонуванні платформи, а також бібліотеки для виконання обчислювальних задач є високо оптимізовані та повністю відкриті (існує спеціальний репозиторій проекту [3] та повна документація по екосистемі [4]. Платформа була розроблена для швидкого та зручного масштабування рішень, вона підтримує обчислення на множині графічних карт що входять і виходять із зв'язку з вузлом через RDMA.

Важливі характеристики:

- основна екосистема для виконання обчислень на графічних процесорах на відеокартах AMD;
- відкрита система, весь код знаходиться у вільному доступі;
- великий набір стандартних бібліотек для чисельного аналізу (включно з BLAS та FFT) ;
- інтегровані популярні програмні каркаси для машинного навчання (TensorFlow, PyTorch) ;
- сумісність з HIP та власний компілятор для конвертації коду (HCC) під платформові рішення AMD;
- висока оптимізація під швидке масштабування рішень;
- широкий набір бібліотек для виконання чисельного аналізу.

На рисунку 2 приведений перелік інструментів екосистеми AMD ROCm.

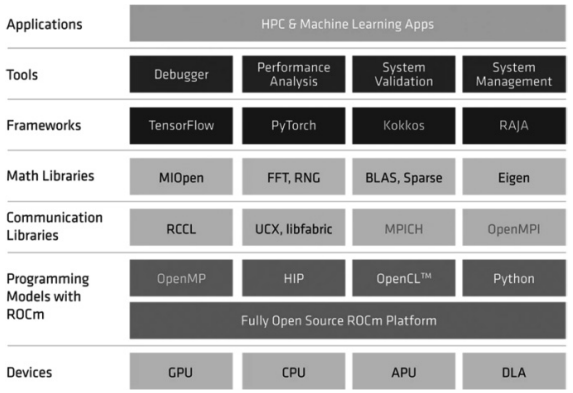


Рис. 2 Складові екосистеми AMD ROCm

2. Використані ресурси для виконання тестів

Для виконання тестів використовувались стандартні попередньо натреновані моделі на наборі даних ImageNet із пакету стандартних моделей програмного каркасу Pytorch. Також було використано фотографії для тестування з відкритого набору даних ImagenetV2 testing.

З точки зору апаратних платформ було використано 2 конфігурації одної апаратної платформи (ноутбука) на основі процесора Intel Core i5 сьомого покоління. Всі характеристики платформи наведено у таблиці 2.

Таблиця 2

Характеристики обчислень на основі процесора Intel Core i5 сьомого покоління

Parameter	Value
Processor	Intel Core i5 7300 HQ 2.5-3.4 GHz
RAM	8 GB
ROM	SSHD TOSHIBA MQ02ABD1 1 Tb
GPU	Nvidia GTX 1050 4 GB
CUDA	v. 9.0.76 with CuDNN v.7.0
Driver GPU	v.390.77
OS	Ubuntu 18.04
CUDA Cores	640

Конфігурація 2

Parameter	Value
Processor	Intel(R) Core(TM) i5-7500 CPU @ 3.40GHz
RAM	32 GB
ROM	HDD WD10EZEX 3.5 SATA III
GPU	Nvidia GTX 1080 8 GB
CUDA	V10.2.89 with CuDNN v.7.2
Driver GPU	v.415.23
OS	Ubuntu 18.04
CUDA Cores	2560

Експеримент 1. Список моделей для тестування:

- resnet101
- vgg19
- alexnet

Тести проводились на попередньо натренованих моделях з використанням GPU. Результати тестування приведені в таблиці 3.

Таблиця 3

Результати тестування на основі процесора Intel Core i5 сьомого покоління

Config1(GPU)		Config2(GPU)		Config1(CPU)	
Model	Mean time (ms)	Model	Mean time	Model	Mean time
resnet50	76.87	resnet50	64.2	resnet50	699
vgg19	136.32	vgg19	68.64	vgg19	1184
alexnet	84.43	alexnet	40.86	alexnet	130

На рисунку 3 показані результати порівняння середньої швидкості роботи програм на конфігурації 1 (ліва колонка) та конфігурації 2 (права колонка).

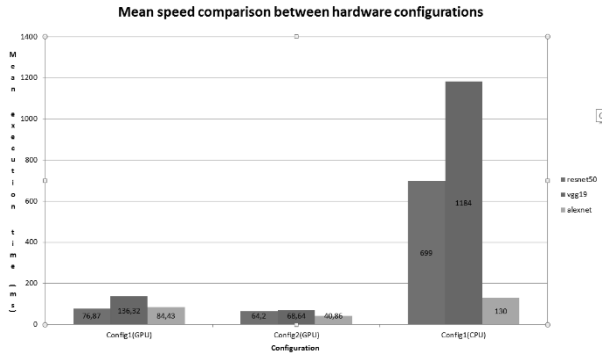
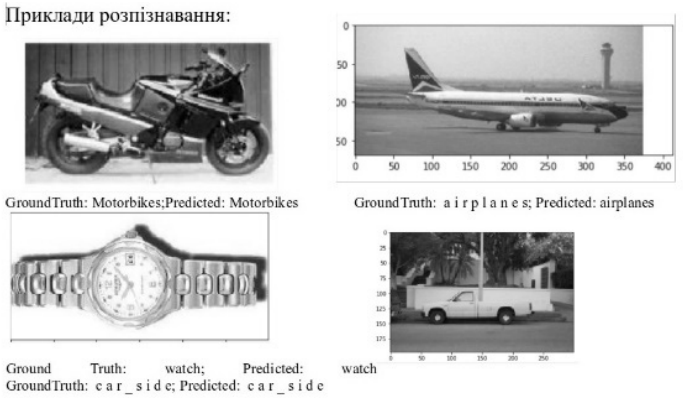


Рис.3 Результати порівняння середньої швидкості розпізнавання об'єктів на GPU та CPU

На рисунку 4 наведено приклади розпізнавання об'єктів з набору даних [6].



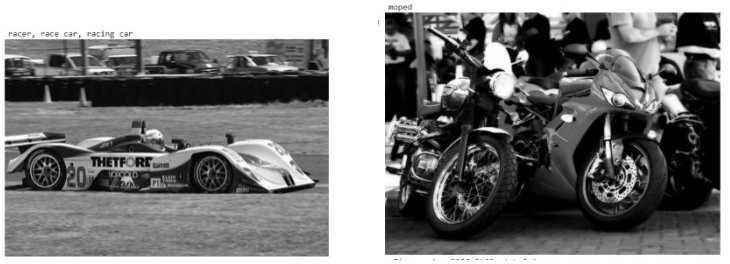


Рис 4. Приклади розпізнавання

Висновок. Запропоновано методику вибору програмно – апаратних платформ для обробки масивів графічної інформації. Протестовано популярні моделі для класифікації зображень з набору попереднього тренування програмного каркасу PyTorch. Результати тестування показують, що збільшення кількості шейдерних ядер CUDA збільшує продуктивність на наборах даних.

Список використаних джерел

1. CUDA Toolkit documentation .Retrieved from: https://docs.nvidia.com/cuda/
2. PyTorch documentation. Retrieved from: https://pytorch.org/docs/stable/index.html
3. Radeon Open Compute (ROCm). Retrieved from: https://github.com/Radeon OpenCompute/ROCm
4. ROCm documentation Retrieved from: https://rocm-documentation.readthedocs.io/en/latest/index.html
5. Ian Goodfellow, Yoshua Bengio,Aaron Courville Deep Learning. A MIT Press Book //Ian Goodfellow, Yoshua Bengio,Aaron Courville . - MIT Press, 2016. - 716 p.
6. ImageNet. Retrieved from: http://www.image-net.org/

REFERENCES

1. CUDA Toolkit documentation .Retrieved from: https://docs.nvidia.com/cuda/ (in English)
2. PyTorch documentation. Retrieved from: https://pytorch.org/docs/stable/index.html (in English)
3. Radeon Open Compute (ROCm). Retrieved from: https://github.com/RadeonOpenCompute/ROCm (in English)
4. ROCm documentation Retrieved from: https://rocm-documentation.readthedocs.io/en/latest/index.html (in English)
5. Ian Goodfellow, Yoshua Bengio,Aaron Courville (2016). Deep Learning. A MIT Press Book //Ian Goodfellow, Yoshua Bengio,Aaron Courville . - MIT Press - 716 p. (in English)
6. ImageNet. Retrieved from: http://www.image-net.org/ (in English)

DOI 10.32403/2411-9210-2020-2-44-112-118

PERFORMANCE COMPARISON OF POPULAR CONVOLUTIONAL MODELS FOR DETECTION OF MECHANICAL OBJECTS

D.M. Myronyuk[1], B.Ya. Blahitko[1], I.M. Zaiachuk[2]

[1]Lviv National University named after I.Franko
1, Universytetska St., Lviv, 79000, Ukraine,
[2]Center of Mathematical Modelling within Ya. S. Pidstryhach Institute of Applied Problems of Mechanics and Mathematics of NAS of Ukraine
15, Dudaieva St., Lviv, 79005, Ukraine
myronyukdmytro@gmail.com, blagitko@gmail.com, igorzajachj@gmail.com

The peculiarities of modern platforms based on graphic maps for in-depth learning have been researched and analysed. According to the results of the analysis, the speed parameters of different models have been evaluated. Effective models of detection of mechanical objects on various models of video cards are offered after the research. The examples demonstrate the effectiveness of pre-trained models of the software framework for deep learning PyTorch. Recommendations are given for the choice of graphics processors for efficient work on the data set.

Keywords: *graphics processor, convolutional model, PyTorch.*

Стаття надійшла до редакції 15.11.2020
Received 15.11.2020