УДК 004.3

С. Д. Погорілий, М. В. Чечула
Київський національний університет імені Тараса Шевченка
Проспект Глушкова, 4-Г, 03022 Київ, Україна

Агрегація та аналіз графічних даних у розподіленій мережі мобільних пристроїв

Проаналізовано напрямки досліджень у галузі концептів Big Data, розподілених мереж мобільних пристроїв і Deep Learning. Кількісно охарактеризовано та порівняно інтенсивність розвитку сучасних бібліотек нейронних мереж для використання технологій Deep Learning, Big Data, GPGPU. Розроблено застосування для дослідження роботи згорткових нейронних мереж різної архітектури із використанням потужностей мобільних CPU та GPU і з використанням API для нейронних мереж на операційній системі Android. Розроблено застосування для агрегації проаналізованих у реальному часі даних, структуризації даних на сервері та досліджено роботу застосування в мережах WiFi, 3G та 4G. Проведено аналіз різних шляхів агрегації даних.

Ключові слова: *концепт Big Data, технологія Deep Learning, технологія GPGPU, операційна система Android, бібліотека TensorFlow, бібліотека CNTK, бібліотека PyTorch, бібліотека MxNet, бібліотека Caffe, нейронна мережа Mobile Net, нейронна мережа Squeeze Net, нейронна мережа Inception Net, нейронна мережа NAS Net, нейронна мережа Dense Net, нейронна мережа Res Net.*

Вступ

Протягом останніх десятиліть стрімко зростає попит на засоби збору інформації як для державного використання так і для бізнесу. Традиційні методи, такі як: камери спостереження чи телефонні мережі є переважно державними, потребують оновлення та спеціалістів для підтримки їхньої працездатності. Однак з мініатюризацією персональних пристроїв збору та обробки інформації з'являються нові методи отримання даних з меншими фінансовими та людськими витратами. На початок 2019 року в світі нараховується понад 2,5 мільярди користувачів, що мають хоча би 1 смартфон [1]. Згідно звіту Google більше 1,2 мільярда фото було завантажено тільки на сервери Google Photos за 2017 рік [2]. Враховуючи те, що значна частка населення планети користується Amazon AWS, Microsoft Azure та іншими сервісами як засобами зберігання інформації, можна стверджувати, що

сумарна кількість тільки фото, які було викладено на хмарні сервіси, становить мільярди за рік [3]. Крім того, ці цифри зростають за експоненційним законом [3]. Відтак постає питання їхньої обробки. Для його вирішення з 2008 року активно проводяться дослідження з аналізу та використання інформації у Big Data [4]. Трансконтинентальними корпораціями вже створено кластерні системи обчислень по всьому світу для обробки інформації, що до них надходить від користувачів. Однак, незважаючи на це, доля оброблених даних у Big Data залишається на надзвичайно низькому рівні — менше 1 % [5]. Це означає, що значна частка інформації, що є шуканою, не може бути знайдена в необробленому об'ємі даних.

Проте, причина виникнення такого стану речей — мобільні пристрої усіх класів і форм — може бути рішенням проблеми низької щільності проаналізованих даних. Перехід до 10 нм технології розробки мобільних процесорів і застосування GPU як акселератора обчислень на них послугувало імпульсом для перенесення деяких технологій аналізу даних на мобільні платформи. Серед них доцільно підкреслити можливості використання технологій Machine Learning та Deep Learning у реальному часі та появу підтримки інтернету поколінь 4G та 5G.

Відтак постає актуальна задача розробки та аналізу рішень з обробки даних в реальному часі у мережах гетерогенних мобільних пристроїв.

Аналіз напрямків Big Data та Deep Learning

З початку 2010-х років все більше уваги дослідників і розробників у сфері Big Data привертає технологія Deep Learning, концепт Big Data та застосування технології GPU в обчисленнях загального призначення — технологія GPGPU. Як можна побачити на рис. 1 в останні роки спостерігається швидке зростання кількості публікацій щодо Deep Learning та Big Data, а кількість публікацій з GPU залишається на стабільно високих позиціях [6]. Це пов'язано зі стрімким підвищенням продуктивності GPU, нюанси архітектури яких доцільно використовувати в Deep Learning та BigData, особливо в науковій підгалузі цих сфер [7, 8]. Проте, специфічні напрямки, такі як, наприклад, застосування мобільних платформ для виконання аналізу даних і використання ОС Android чи IOS у розробці складних систем обробки Big Data знаходяться на ранніх стадіях розвитку (рис. 2) [6].

Такий інтерес до технологій аналізу даних на мобільних платформах пов'язаний зі зростанням не тільки кількості самих пристроїв а й продуктивності CPU та GPU на них.

Як зображено на рис. 3, продуктивність процесорів на обох платформах за 8 років зросла більш ніж 10 разів як в мультипроцесорному так і в однопроцесорному режимах [9]. Це поставило їх в один ряд з такими процесорами як Intel Core i5-6300HQ [10], що сьогодні є поширеним типом процесорів для Desktop та Laptop. Наявність мультипроцесорного режиму та допоміжних GPU в мобільних процесорах останніх поколінь дозволяє виконувати обчислення, що піддаються паралелізації у реальному часі. Отже з'являється можливість переносити деякі задачі, які до цього були прерогативою тільки ПК, на мобільні платформи та впроваджувати рішення на них в існуючу середу обчислювального обладнання.

Незважаючи на те, що продуктивність мобільних пристроїв і кількість їхніх виробників невпинно зростає, кількість операційних систем, що обслуговують апаратні архітектури, зменшується. На рис. 4 показано тренд щодо збільшення

сегменту Android та IOS-пристроїв з перевагою Android та зменшення сегменту непопулярних ОС, таких як Windows Phone, Symbian, Blackberry та ін. [11]. Відтак вибір операційної системи (ОС) для проведення досліджень на мобільних ОС de facto відбувається тільки між ОС Android та ОС IOS.

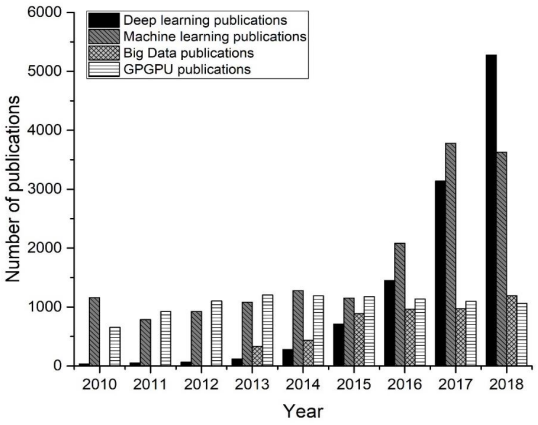


Рис. 1. Кількість публікацій за напрямками: Deep Learning, Machine Learning Big Data та GPGPU

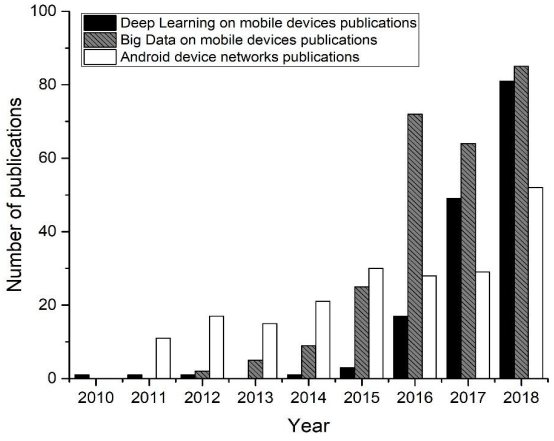


Рис. 2. Динаміка кількості публікацій у галузях обробки даних на мобільних пристроях

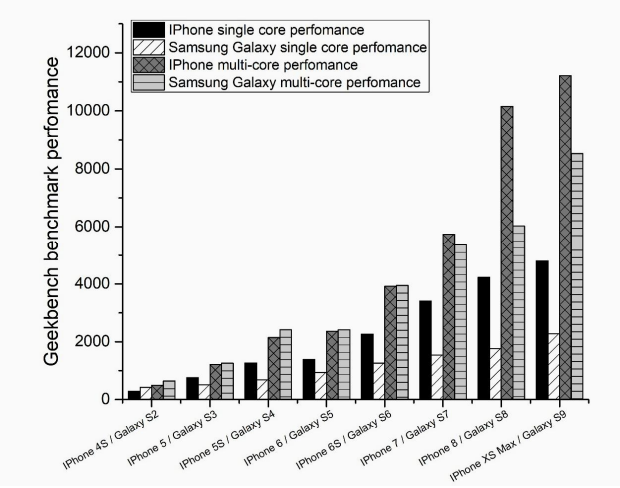


Рис. 3. Порівняння продуктивності мобільних пристроїв з періоду 2011–2018 рр.

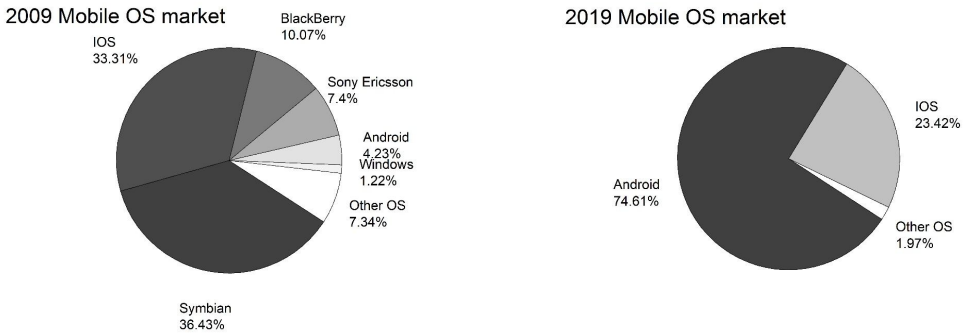


Рис. 4. Порівняння сегментації ринку мобільних ОС у 2009 та 2019 роках

Причиною такого успіху ОС Android можна вважати, перш за все, зручність та економічну вигідність розробки застосувань. На відміну від Apple, Google надає більше свободи у використанні IDE та при використанні функцій самого пристрою. Кількість застосувань для Android зростає швидше ніж в інших ОС, а отже є перспективним середовищем для створення як комерційних, так і дослідницьких робіт [12, 13].

Аналіз бібліотек для нейронних мереж

Наразі нейронні мережі набувають широкого поширення, оскільки тільки зараз з'явилися обчислювальні потужності, що здатні їх обслуговувати в глобальному масштабі. І якщо вибір мобільної ОС наразі визначається більшою мірою вже складеною ситуацією на ринку на сьогоднішній день, то вибір інструмента обробки даних у мережах мобільних пристроїв, особливо, якщо ці інструменти новітні, визначається темпами розвитку, ступенем підтримки та швидкістю розробки при використанні.

Перші публічні бібліотеки для створення нейронних мереж почали з'являтися ще в 1987 р. [14], проте вони не надавали змогу швидко та ефективно оперувати їхніми можливостями на звичайних ПК. Однак в останні роки, у зв'язку з різким зростанням IT-індустрії, з'явилися більш потужні та більш досконалі аналоги. На рис. 5 [15–19] показано порівняння темпів розвитку та популярності таких сучасних бібліотек Deep Learning як: TensorFlow, PyTorch, MxNet, Caffe та CNTK.

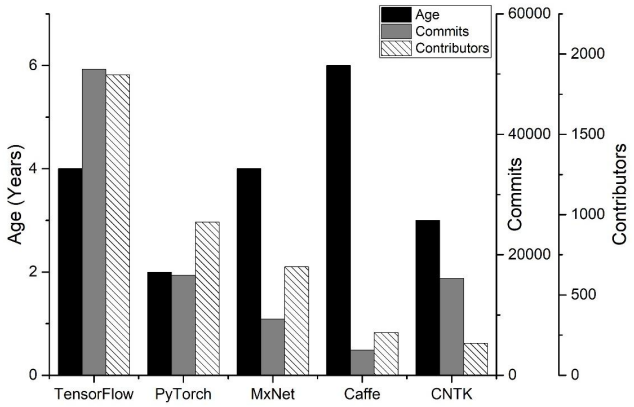


Рис. 5. Порівняння сучасних бібліотек для роботи з Deep Learning

Як бачимо з діаграми, TensorFlow є найбільш швидко зростаючим інструментом, який розробляють уже більше 4 років тисячі інженерів Google та світу. На початок 2019 року в середньому кожну годину відбувається 2 зміни програмного забезпечення TensorFlow [15].

Швидкодія ж застосувань, що засновані на бібліотеках залежить більше від мови програмування, оскільки математична модель, яка закладена в середину нейронної мережі майже не змінюється залежно від бібліотеки. Бібліотеки підтримують як мови програмування низького рівня, такі як C, так і високого, такі як Python. Реалізація ж залежить від мети користувача [15–19].

Після аналізу бібліотек за критеріями зручності розробки, підтримки нейромереж для обробки зображень — згорткових нейронних мереж (англ. CNN) та до-

кументації, як основи подальших розробок і досліджень, було обрано бібліотеку TensorFlow.

Аналіз нейронних мереж для мобільних пристроїв

У сучасних гетерогенних мережах обробки даних, особливо, якщо серед них є відносно малопотужні смартфони, важливо оптимізувати кількість ресурсів, задіяних для виконання обробки чи передачі даних.

У підході, що описується, основним джерелом навантаження для обчислювальних пристроїв є нейронна мережа, а саме її використання. Оскільки згорткові нейронні мережі, які є ефективними при обробці зображень, засновані на матричних перетвореннях, застосування додаткових потоків CPU та ядер мобільних GPU може надати необхідне прискорення залежно від архітектури та типу нейронної мережі. Точність же розпізнавання суттєво залежить як від параметрів нейронної мережі, так і від типу архітектури.

Для проведення досліджень з продуктивності використання тієї чи іншої нейронної мережі було створено застосування, в яких реалізовані різні підходи до використання нейронних мереж.

Як апаратні платформи було доцільно використати такі мобільні пристрої:

— Huawei P20 Pro. Специфікації [20];

— Meizu M3 Note. Специфікації [21];

— Samsung S4. Специфікації [22].

Як ядро застосування було використано нейронні мережі двох видів: Float — з обчисленнями на плаваючій точці (адаптованих під GPU) та Integer (Quant) — з обчисленнями в цілих числах (адаптованих для CPU).

Причиною появи Quant-моделей нейронних мереж стала відсутність оптимізації сучасних високоточних нейронних мереж до використання на пристроях низької і середньої потужності. Сучасні розробки в сфері Deep Learning сфокусовані здебільшого на збільшенні точності, що збільшує розмір таких нейронних мереж до гігабайтів, а час виконання розпізнавання до секунд на середньо-потужних пристроях. Задля оптимізації таких обчислень і зменшення кількості пам'яті для роботи з нейронною мережею використовується 2 підходи [23]:

1) використання операндів, ефективних за співвідношенням *точність обчислення*/*затрати пам'яті (Float)*;

2) перехід від використання 32-бітних змінних до менших за кількістю біт цілих для системи ваг і функції активації — так зване квантування (Quant/Integer).

Для більшого розуміння суттєвої відмінності підходів до оптимізації нейромереж коротко розглянемо підхід квантування. Основою підходу квантування є те, що він дозволяє ефективно використовувати всі арифметичні операції над квантованими значеннями в цілих числах. Це еквівалентно тому, що схема квантування є афінним відображенням чисел q до дійсних чисел r, тобто форми

$$r = S(q - Z), \tag{1}$$

де S та Z — деякі константи, які є параметрами квантування.

Для прикладу, якщо виконується 8-бітне квантування, то q буде 8-бітною змінною.

Константа S — додатна дійсна змінна, що функціонує як коефіцієнт.

Константа Z — 8-бітна константа, як є квантованим значенням змінної q, що відповідає дійсному значенню 0.

Оскільки левова частка обчислень у нейронних мережах стосується саме матричних операцій, розглянемо приклад множення матриць $r_1 r_2$ розміром $N \times N$ з реальними числами, де результатом є матриця $r_3 = r_1 r_2$. Нехай усі значення в квадратних матрицях r_α, де α є 1,2 чи 3, є $r_\alpha^{(i,j)}$ для $1 \le i, j \le N$, а параметрами їхньої квантизації є S_α, Z_α. Нехай також значення q мають форму $q_\alpha^{(i,j)}$.

Рівняння (1) тепер має вигляд:

$$r_\alpha^{(i,j)} = S_\alpha (q_\alpha^{(i,j)} - Z_\alpha). \tag{2}$$

Для множення матриць маємо:

$$S_3 (q_3^{(i,k)} - Z_3) = \sum_{j=1}^{N} S_1 (q_1^{(i,j)} - Z_1) S_2 (q_2^{(j,k)} - Z_2). \tag{3}$$

Останнє рівняння може бути переписане як

$$q_3^{(i,k)} = Z_3 + M \sum_{j=1}^{N} (q_1^{(i,j)} - Z_1)(q_2^{(j,k)} - Z_2), \tag{4}$$

де мультиплікатор M визначається як

$$M = \frac{S_1 S_2}{S_3}. \tag{5}$$

У рівнянні (4) єдиною нецілою величиною є мультиплікатор M. Так як M залежить тільки від констант квантування, то він може бути обрахований завчасно. Він завжди буде знаходитись у межах від 0 до 1, тому може бути поданий в нормалізованому вигляді

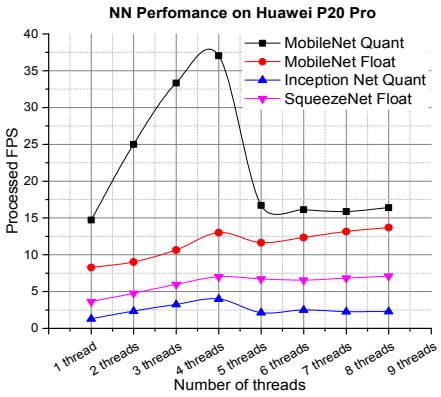
$$M = 2^{-n} M_0, \tag{6}$$

де M_0 існує в інтервалі $[0,5;\ 1)$, а n — невід'ємне ціле число. Відтак M може бути відображений як число з фіксованою точкою. Залежно від апаратної платформи це може бути Integer32 чи Integer16. У випадку 32-розрядного цілого, враховуючи те, що M більше 0,5, маємо щонайменше 30 біт точності, чого достатньо в більшості випадків [23].

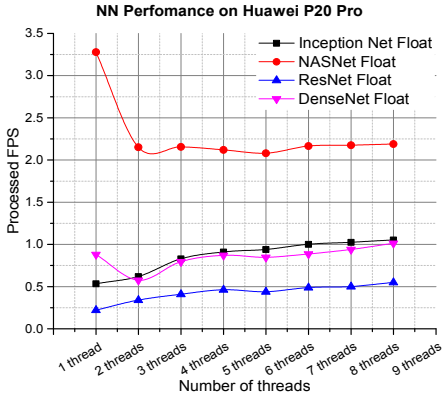
Отже, як ядро застосування було використано сучасні відкриті нейронні мережі обох типів, такі як:

— Inception Net Float [24];
— Inception Net Quant [25];
— Mobile Net Float [23];
— Mobile Net Quant [26];
— Squeeze Net Float [27];
— NAS Net Float [28];
— Dense Net Float [29];
— Res Net Float [30].

Результати використання нейронних мереж представлено на рис. 6–8.

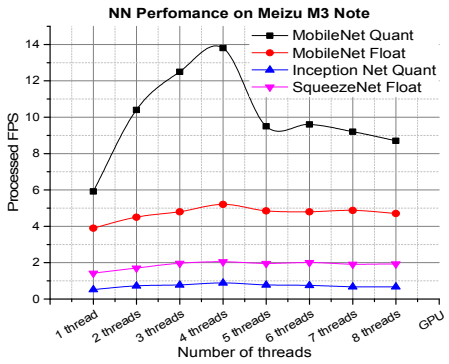


а)

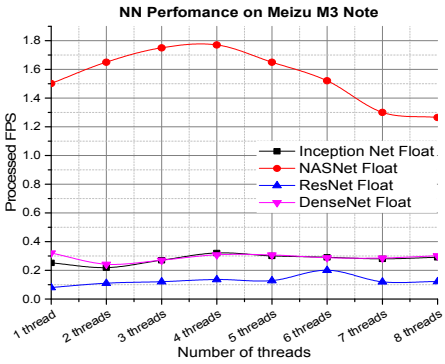


б)

Рис. 6. Вплив багатопоточності на швидкість обробки фото на Huawei P20 Pro:
а) швидкі нейронні мережі; б) повільні нейронні мережі

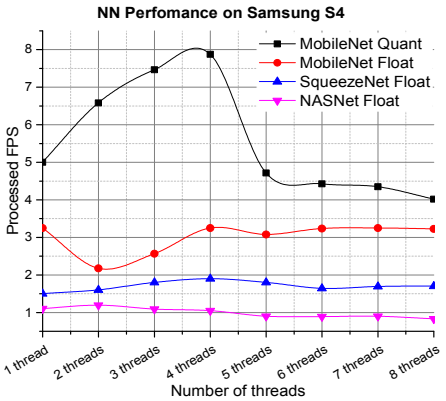


а)

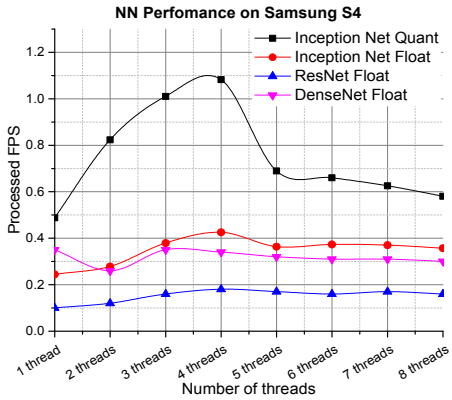


б)

Рис. 7. Вплив багатопоточності на швидкість обробки фото на Meizu M3 Note:
а) швидкі нейронні мережі; б) повільні нейронні мережі



а)



б)

Рис. 8. Вплив багатопоточності на швидкість обробки фото на Samsung Galaxy S4:
а) швидкі нейронні мережі; б) повільні нейронні мережі

Як можна побачити продуктивність деяких типів нейронних мереж непропорційна загальній продуктивності процесора. Наприклад, нейронна мережа NAS Net показує кращі результати ніж Inception Net на процесорі Samsung S4. Це свідчить про внутрішню оптимізацію операції на процесорі виробником. Також треба підкреслити, що після досягнення максимуму при кількості потоків, що дорівнює кількості ядер, продуктивність практично не зростає. Безумовним лідером за швидкістю роботи є нейронна мережа Mobile Net з обчисленнями на цілих числах. Проте швидкість корелює з точністю, що досягається. Показники точності наведені на рис. 9 [31].

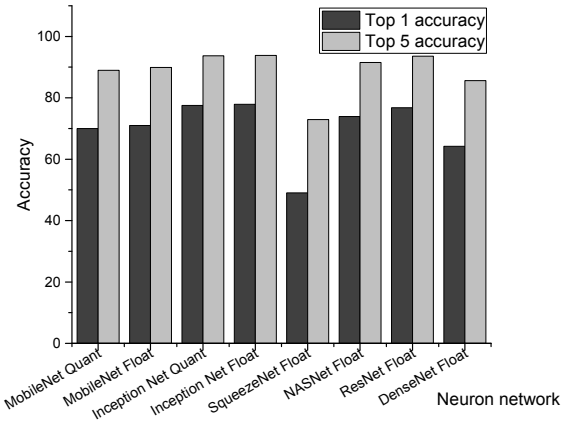


Рис. 9. Залежність точності нейронної мережі від її архітектури

Як бачимо з гістограми нейронні мережі типу Inception, мають на 10 % кращі показники точності розпізнавання об’єктів, проте, якщо брати до уваги 5 об’єктів, які нейронна мережа вважає вірними, то різниця між відносно швидкою Mobile Net та Inception Net неістотна, менша за 5 %.

В останній час з’являється все більше мобільних процесорів з підтримкою вбудованих GPU. Метою такого підходу було збільшення кількості операцій, які мобільний пристрій здатний виконувати в графічних програмах. Але цей механізм може бути використаний для реалізації технології GPGPU — обчислень загального характеру на GPU.

Важливо відзначити, що поряд з технологією GPGPU для виконання операцій з нейронними мережами, з виходом ОС Android 8.1 у розробників з’явилася можливість використовувати інтерфейс високого рівня абстракції для роботи з нейронними мережами — NNAPI. Архітектура NNAPI зображена на рис. 10 [32].

Як можна побачити з рис. 10 Архітектура NNAPI дозволяє автоматично перевірити наявність спеціалізованих процесорів або GPU для використання в машинному навчанні. Проте треба зауважити, що складний механізм комунікації може істотно зменшувати ефективність такого підходу порівняно з обчисленнями безпосередньо на CPU.

Для перевірки ефективності використання мобільних GPU та NNAPI для обчислень у нейронних мережах було реалізовано додатковий функціонал для їхнього застосування у розпізнаванні зображень. Результати можна побачити на рис. 11.

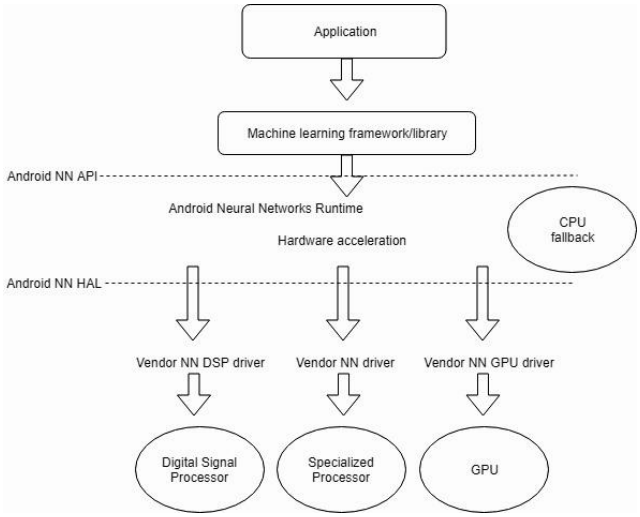
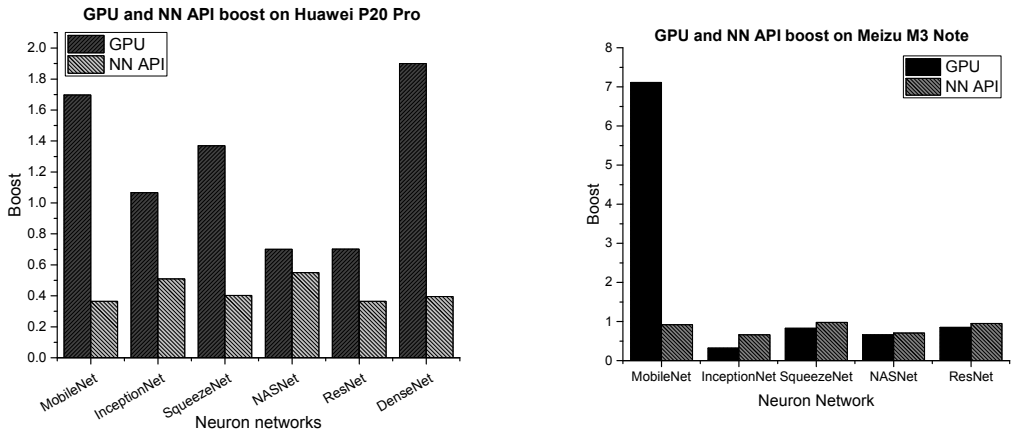


Рис. 10. Архітектура NNAPI



а) б)

Рис. 11. Прискорення обробки даних з використанням GPU та NNAPI на:
а) Huawei P20 Pro; б) Meizu M3 Note

Аналіз гістограм свідчить про те, що на різних платформах використання GPU може надавати значне прискорення, та у такий спосіб робити використання нейронних мереж з плаваючою точкою більш ефективним — використання GPU на Meizu M3 Note у нейронній мережі Mobile Net з обчисленнями без використання квантування показало результат у 37 оброблених фото за секунду, в той час як при використанні CPU у нейронній мережі Mobile Net з використанням квантування результат складав 14. Якщо порівняти 1-потоковий режим розпізнавання Mobile Net з використанням квантування та GPU-режим MobileNet без використання квантування, то прискорення останнього становить 9,5 разів — цей показник є подібним до прискорення, яке GPU надає при навчанні нейромереж на пристроях типу Laptop [33, 34]. Використання ж API високого рівня абстракції для роботи з нейронними мережами у Android не надавало прискорення, проте й не

змен-шувало продуктивність більше ніж в 2,5 рази від максимально доступної при використанні CPU та приблизно відповідало продуктивності при задіянні 1 ядра CPU.

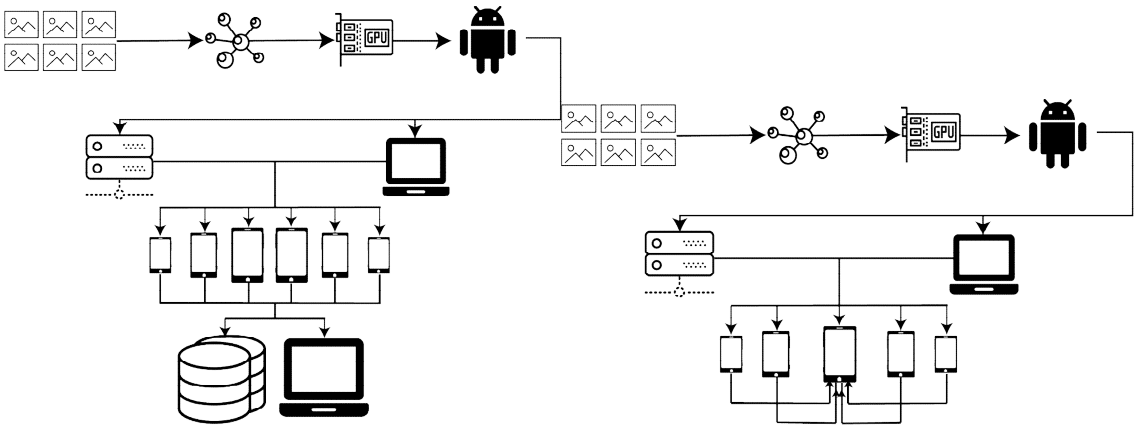
У підсумку, тип нейронної мережі для застосування на мобільних платформах залежить від апаратної платформи в цілому, а тип і клас нейронної мережі для використання на конкретній платформі з найбільшою ефективністю можна визначати тільки персоніфіковано.

Метод агрегації даних на ОС Android

Для перевірки роботи методу в реальних умовах вирішувалася технічна задача — деякий об'єкт потрібно локалізувати в реальному світі в режимі реального часу, не маючи при цьому системи відеоспостереження в необхідній зоні.

Було створено застосування на ОС Android із застосуванням бібліотеки TensorFlow, що містили заздалегідь навчені на високопродуктивних апаратних системах нейронні мережі класу Mobile Net, які, в свою чергу, виконували аналіз отриманих з камери пристрою фото, збирали дані щодо розташування самого пристрою та передавали отриману стиснуту за алгоритмом JPEG2000 інформацію в високій роздільній здатності, а саме 1.4MP на сервер. Було досліджено 2 варіанти серверної частини. В першому було використано віртуальний виділений сервер на ОС Linux та структуровано дані в базу даних NoSQL, як сучасний спосіб створювати швидкі масштабовані бази даних, що зберігають дані не як записи в таблиці (SQL), а як файли, що вигідно у випадку роботи з великою кількістю зображень. У другому варіанті як сервер було використано один із мобільних пристроїв, де отримані дані індексувалися та зберігалися на внутрішній швидкісний накопичувач. Усі дані, що передавалися на сервер, передавалися із SSL-шифруванням за TCP/IP протоколом. Зв'язок між усіма пристроями було використано у 3-х варіантах: WiFi, 3G, 4G. Як дистриб'ютор застосувань було використано сервер і ПК.

Схематично структуру обох варіантів побудови мережі мобільних пристроїв зображено на рис. 12.



а) б)

Рис. 12. Ілюстрація функціонування мережі мобільних пристроїв із використанням:
а) виділеного сервера; б) одного з пристроїв як сервера

Для зручного інтерфейсу керування даними на сервері (варіант 1) було розроблено веб-застосування для пошуку місця та часу перебування об'єктів, схожих на шуканий, і виводу на екран декількох останніх знімків цих об'єктів (рис. 13).

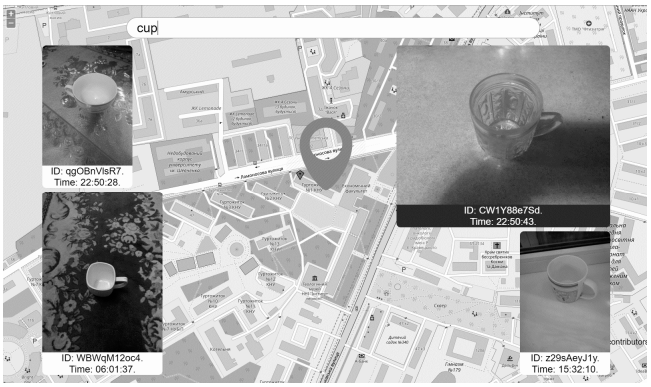
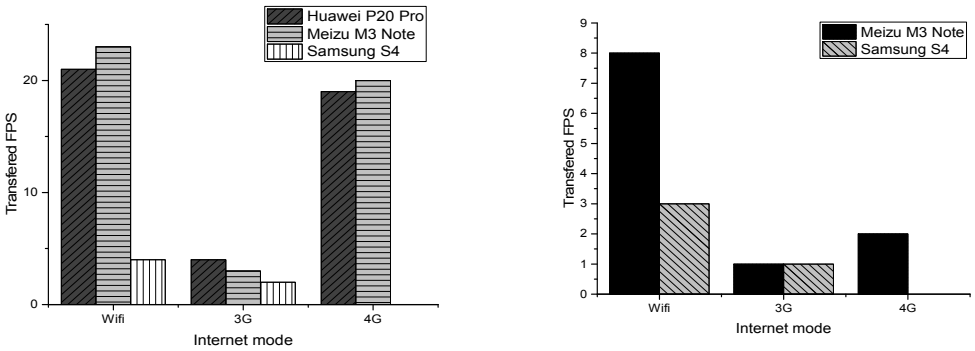


Рис. 13. Приклад роботи застосування для пошуку даних про місцезнаходження об'єктів

Слід відзначити, що аналогічний інтерфейс можливо створити і для варіанта 2, де сервером є один з мобільних пристроїв, однак у такому випадку доцільніше створювати спеціальний застосунок до ОС Android.

Результати експериментів з обробки та передачі даних за допомогою мережі з 3 мобільних пристроїв і сервера Linux представлено на рис. 14,*а*. Результати ж експериментів з мобільним пристроєм Huawei P20 Pro як сервера представлено на рис. 14,*б*.



а) б)

Рис. 14. Швидкість передачі оброблених нейронною мережею фото на:
а) виділений сервер; б) мобільний пристрій

Як можна спостерігати на рис. 14,*б* мережі останнього покоління мобільного Інтернету 4G надають змогу передавати інформацію про навколишнє середовище у високій якості з частотою до 20 фото на секунду, що більше за середню кількість фото, що робить середньостатистична камера спостереження.

Система із застосуванням мобільного пристрою очікувано істотно зменшила кількість переданої інформації, проте проблема полягала у високих затримках у мережі, особливо коли одночасно йдуть передача й отримання інформації на один

і той самий пристрій. Ця проблема поступово нівелюється з впровадженням мереж 5G, що працюють з наднизькою затримкою при передачі даних.

Висновки

1. Використання мобільних пристроїв в обробці даних із застосуванням концепту Big Data та технології Deep Learning набуває широкого поширення, кількість публікацій з цих тем зростає в середньому на 40 % за рік. Сучасні мобільні пристрої і бібліотеки для роботи з нейронними мережами надають можливості проводити аналіз даних у реальному часі, що дозволяє створювати до 40 проаналізованих фото за секунду навіть на пристроях середньої продуктивності

2. Застосування мобільних GPU дозволяє пришвидшити процес обробки даних (до 10 разів) порівняно з мобільним CPU. Дослідження роботи нейронних мереж у мережах мобільних пристроїв різної обчислювальної потужності показало, що на початок 2019 року доцільним є використання мереж мобільних пристроїв для пошуку інформації про місцерозташування об'єктів у реальному часі, навіть за умови відсутності високопродуктивного сервера.

3. Впровадження мобільного Інтернету наступних поколінь 4G та 5G надає змогу об'єднувати потоки оброблених даних з мобільних пристроїв і використовувати мережу мобільних пристроїв для пошуку інформації у місцях, недосяжних для традиційних способів збору інформації.

4. З метою експериментальної перевірки застосування мереж мобільних пристроїв для вирішення проблеми малої щільності проаналізованих даних у Big Data (їхньої здатності до обробки графічних даних у режимі реального часу та агрегації цих даних для подальшого структурування інформації) на основі проведеного аналізу було розроблено застосування, яке за заданими параметрами пошуку знаходило місцерозташування всіх схожих за параметрами об'єктів у режимі реального часу за допомогою мобільних камер у мережі мобільних пристроїв. У результаті було показано, що сучасні мобільні пристрої на ОС Android, з використанням нейронної мережі з архітектурою Mobile Net здатні (з вірогідністю понад 80 %) надавати інформацію про шукані об'єкти з частотою передачі розпізнаних кадрів у мережі WiFi до 20 кадрів на секунду. Експеримент також підтвердив можливість розгортання гетерогенних розподілених систем збору та аналізу графічної інформації у важкодоступних місцях.

1.Кількість користувачів смартфонів. URL: http://www.statista.com/statistics/330695/number-of-sma-phone-users-worldwide/

2. Кількість завантажених фото на сервери Google Photos. URL: http://www.blog.google/products/photos/google-photos-500-million-new-sharing/

3. Популярність хмарних сервісів. URL: https://www.forbes.com/sites/louiscolumbus/2018/09/23/roundup-of-cloud-computing-forecasts-and-market-estimates-2018/#79b146ca507b

4. Історія Big Data. URL: https://www.forbes.com/sites/gilpress/2013/05/09/a-very-short-history-of-big-data/#4a8694d365a1

5. Big Data статистика. URL: https://www.theguardian.com/news/datablog/2012/dec/19/big-data-study-digital-universe-global-volume

6. Бібліотека комп'ютерних наук DBLP. URL: https://dblp.uni-trier.de/

7. Погорелый С.Д., Бойко Ю.В., Трибрат М.И., Грязнов Д.Б. Анализ методов повышения производительности компьютеров с использованием графических процессоров и программно-аппаратной платформы CUDA. *Математичні машини і системи*. 2010. № 1. С. 40–54.

8. Погорілий С.Д. Програмне конструювання. Підручник серії «Автоматизація наукових досліджень» за ред. академіка АПН України Третяка О.В. ВПЦ Київський університет, 2007. 438 с.

9. Рейтинг продуктивності мобільних процесорів за методикою. URL: Geekbench https://browser.geekbench.com/android-benchmarks

10. Рейтинг продуктивності процесорів для ПК за методикою Geekbench. URL: https://browser.geekbench.com/processor-benchmarks

11. Статистика розповсюдженості мобільних ОС. URL: http://gs.statcounter.com/os-market-share/mobile/worldwide

12. Динаміка кількості застосувань в Google Play Store. URL: https://www.statista.com/ statistics/266210/number-of-available-applications-in-the-google-play-store/

13. Динаміка кількості застосувань в Apple App Store. URL: https://www.statista.com/ statistics/268251/number-of-apps-in-the-itunes-app-store-since-2008/

14. Історія Wolfram Research ОС. URL: http://www.wolfram.com/company/background.html?source=nav

15. Офіційний відкритий репозиторій TensorFlow. URL: https://github.com/tensorflow/tensorflow

16. Офіційний відкритий репозиторій PyTorch. URL: https://github.com/pytorch/pytorch

17. Офіційний відкритий репозиторій CNTK. URL: https://github.com/Microsoft/CNTK

18. Офіційний відкритий репозиторій Caffe. URL: https://github.com/BVLC/caffe

19. Офіційний відкритий репозиторій MxNet. URL: https://github.com/apache/incubator-mxnet

20. Специфікації Huawei P20 Pro. URL: https://consumer.huawei.com/en/phones/p20-pro/specs/

21. Специфікації Meizu M3 Note. URL: https://www.meizu.com/en/products/m3note/spec.html.

22. Специфікації Samsung S4. URL: https://www.gsmarena.com/samsung_i9500_galaxy_s4-5125.php

23. Howard Andrew G., Zhu Menglong, Chen Bo, Kalenichenko Dmitry, Wang Weijun, Weyand Tobias, Andreetto Marco, Adam Hartwig. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *Computer Vision and Pattern Recognition section*. ArXiv.org. Cornell University, 2017.

24. Szegedy Christian, Vanhoucke Vincent, Ioffe Sergey, Shlens Jonathon. Rethinking the Inception Architecture for Computer Vision. *Computer Vision and Pattern Recognition section*. ArXiv.org. Cornell University, 2016.

25. Krishnamoorthi Raghuraman. Quantizing deep convolutional networks for efficient inference: A whitepaper. *Machine Learning section*. ArXiv.org. Cornell University, 2018.

26. Jacob Benoit, Kligys Skirmantas, Chen Bo, Zhu Menglong, Tang Matthew, Howard Andrew, Adam Hartwig, Kalenichenko Dmitry. Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference. *Machine Learning section*. ArXiv.org. Cornell University, 2017.

27. Iandola Forrest N., Han Song, Moskewicz Matthew W., Ashraf Khalid, Dally William J., Keutzer Kurt. Squeezenet: Alexnet level accuracy with 50x fewer parameters and <0.5mb model size. *Computer Vision and Pattern Recognition section*. ArXiv.org. Cornell University, 2016.

28. Zoph Barret, Vasudevan Vijay, Shlens Jonathon, Le Quoc V. Learning Transferable Architectures for Scalable Image Recognition. *Computer Vision and Pattern Recognition section*. ArXiv.org. Cornell University, 2018.

29. Huang Gao, Liu Zhuang, Laurens van der Maaten. Densely Connected Convolutional Networks. *Computer Vision and Pattern Recognition section*. ArXiv.org. Cornell University, 2018.

30. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun – Identity Mappings in Deep Residual Networks. *Computer Vision and Pattern Recognition section*, ArXiv.org, Cornell University, 2016.

31. Порівняння точності розпізнавання нейронних мереж на Google Pixel 2 URL: https://www.tensorflow.org/lite/guide/hosted_models

32. NNAPI. URL: https://developer.android.com/ndk/guides/neuralnetworks

33. Погорілий С.Д., Чечула М.В. Застосування технології GPGPU при роботі з BigData. *Наукові праці Донецького національного технічного університету*. 2018. № 2. С. 98–102.

34. Чечула М.В., Погорілий С.Д. Розробка методів роботи з BigData на мобільних пристроях з використанням технології GPGPU. System Analysis and Information Technologies. Institute for Applied System Analysis. 2018. С. 159.

Надійшла до редакції 02.05.2019