Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет комп'ютерної інженерії та управління
(повна назва)

Кафедра електронних обчислювальних машин
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Алгоритми підвищення ефективності згорткових нейронних мереж

(тема)

Виконав:

студент II курсу, групи КСМм-21-1

Пономаренко Р.Д.
(прізвище, ініціали)

Спеціальність

123 «Комп'ютерна інженерія»
(код і повна назва спеціальності)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма

Комп'ютерні системи та мережі
(повна назва освітньої програми)

Керівник: проф. Міхаль О.П.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри ЕОМ _____ Коваленко А.А.
(підпис) (прізвище, ініціали)

2022 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління

Кафедра електронних обчислювальних машин

Рівень вищої освіти другий (магістерський)

Спеціальність 123 «Комп’ютерна інженерія»
(код і повна назва)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Комп’ютерні системи та мережі
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
“____” _____ 20__ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

студенту Пономаренку Роману Дмитровичу
(прізвище, ім’я, по батькові)

1. Тема роботи Алгоритми підвищення ефективності згорткових нейронних мереж

затверджена наказом по університету від “ 07 ” листопада 2022 р. № 1453 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 13 грудня 2022 р.

3. Вхідні дані до роботи
згорткова нейронна мережа
CP-декомпозиція
архітектура

4. Перелік питань, що потрібно опрацювати у роботі
Згорткові нейронні мережі
Аналіз алгоритмів прискорення згорткових нейронних мереж
Реалізація алгоритм декомпозиції вагів у згорткових нейронних мережах

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) 17 слайдів

6. Консультанти розділів роботи (заповнюється за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Отримання завдання. Аналіз літератури.	07.11.2022–15.11.2022	
2	Огляд існуючих алгоритмів та методів.	16.11.2022–22.11.2022	
3	Аналіз існуючих архітектур.	23.11.2022–28.11.2022	
4	Реалізація розглянутих методів.	29.11.2022–02.12.2022	
5	Вибір баз даних для тестування.	03.12.2022–04.12.2022	
6	Отримання результатів	05.12.2022–06.12.2022	
7	Оформлення ПЗ	07.12.2022–12.12.2022	

Дата видачі завдання 07 листопада 2022 р.

Студент _____
(підпис)

Керівник роботи _____ проф. Міхаль О.П.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 65 с., 13 рис., 1 дод., 8 джерел.

ТЕНЗОР, ЗГОРТКОВА НЕЙРОННА МЕРЕЖА, НАБІР ДАНИХ, ALEXNET, CP-ДЕКОМПОЗИЦІЯ, ILSVRC, MNIST.

Метою кваліфікаційної роботи є аналіз методів та алгоритмів підвищення ефективності згорткових нейронних мереж.

У ході виконання кваліфікаційної роботи проведено порівняльний аналіз популярних архітектур згорткових нейронних мереж та їх блоків. Проведено огляд існуючих алгоритмів підвищення ефективності згорткових нейронних мереж, а також методів тензорної декомпозиції; обрано набори даних для проведення експериментів; проведено порівняльний аналіз та реалізовано розглянуті алгоритмів.

ABSTRACT

Master’s thesis: 65 pages, 13 figures, 1 appendices, 8 sources.

TENSOR, CONVOLUTIONAL NEURAL NETWORK, DATASET, ALEXNET, CP-DECOMPOSITION, ILSVRC, MNIST.

The major goal of this thesis is the analysis of methods and algorithms for improving the efficiency of convolutional neural networks.

In order to comparative analysis of popular architectures of convolutional neural networks and their blocks was carried out. An overview of existing algorithms for increasing the efficiency of convolutional neural networks, as well as tensor decomposition methods, was conducted; selected data sets for conducting experiments; a comparative analysis was carried out and the considered algorithms were implemented.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

CNN – згорткова нейронна мережа (ЗНМ)

CP – тензорна декомпозиція (від англ. Canonical Polyadic Decomposition)

CPU – центральний процесор

GPU – графічний процесор

VGG – згорткова мережа для виділення ознак

ВСТУП

Здатність багатошарових нейронних мереж, навчених методом градієнтного спуску до побудови складних багатовимірних областей на основі великої кількості навчальних прикладів, дозволяє застосовувати їх як класифікатор для розпізнавання образів. Незважаючи на це, у традиційній повнозв'язній нейронній мережі є ряд недоліків, що знижують ефективність її роботи.

Насамперед, це великий розмір зображень (під зображенням розуміється графічне подання розпізнаваного образу, представлене у вигляді набору пікселів), який може досягати кількох тисяч (або набагато більше). Для коректного навчання таким даним потрібно збільшити кількість прихованих нейронів, що призводить до збільшення числа параметрів, і, як наслідок, знижує швидкість навчання, вимагає більшої навчальної вибірки. Але найбільшим обмеженням таких мереж є те, що вони не відрізняються. інваріантністю до різних деформацій, наприклад, перенесення або незначного спотворення вхідного сигналу.

Слід зазначити, що варіації написання символів містять такі деформації (однієї з них можна вважати індивідуальний почерк, який суттєво змінює написання символу). В принципі, цю проблему можна вирішити за рахунок поповнення навчальної вибірки прикладами таких спотворених символів, проте це призведе до низької швидкості навчання і, що гірше, може призвести до поганої узагальнюючої здатності мережі [1].

Ще одним недоліком класичних повнозв'язкових мереж є те, що вони ігнорують топологію вхідного зображення. Пікселі можна подавати на вхід у будь-якому фіксованому порядку, і це не вплине на кінець навчання. З іншого боку, зображення мають чітку двовимірну структуру: сусідні пікселі пов'язані між собою, і ця структура несе у собі цінну інформацію про зображення. Локальний зв'язок пікселів – основна причина використання

певних механізмів вилучення локальних ознак на певній області зображення з подальшим формуванням певної системи таких ознак.

Така система інтуїтивно зрозуміла: різні конфігурації сусідніх пікселів формують певні категорії (кути, краї і т.і.). Очевидно, що ефективна система розпізнавання образів має ґрунтуватися на алгоритмі, що враховує такі особливості вхідних даних. У цій роботі розглянуті згорткові мережі – один із підвидів нейронних мереж, які значною мірою усувають вищеописані недоліки пов'язаних нейронних мереж та гарантує швидке навчання та розпізнавання образів.

Також слід зазначити, що згорткові нейронні мережі (CNN) є надзвичайно потужними моделями, які домінують у сучасному комп'ютерному зорі. CNN використовуються для класифікації зображень, сегментації, виявлення, фільтрації та створення завдань. Так само глибокі методи навчання процвітають у мові обробки сигналів і навчання з підкріпленням загального призначення.

Деякі застосунки комп'ютерного зору характеризуються великим обсягом даних, що підлягають обробці. Це можна реалізувати шляхом обчислення дескрипторів усіх зображень у базі даних, а потім порівняти дескриптор зображення запиту з дескрипторами бази даних.

Для сучасних пошукових систем ця база даних містить усі зображення в Інтернеті, і потрібні тисячі запитів обробляються одночасно, що вимагає величезної обчислювальної потужності. Навіть якщо доступна необхідна кількість потужних графічних процесорів, швидші моделі все одно корисні як міра для збереження електроенергії. У спробі вирішити ці проблеми було створено новий напрямок досліджень: нейронні мережі прискорення та стиснення.

Метою кваліфікаційної роботи є аналіз алгоритмів підвищення ефективності згорткових нейронних мереж. Також в роботі присутній опис підходів для прискорення CNN. Об’єктом дослідження є згорткові нейронні мережі. Завданнями є: аналіз популярних архітектур згорткових нейронних

мереж та їх блоків; огляд існуючих алгоритмів підвищення ефективності згорткових нейронних мереж; вибір наборів даних для проведення експериментів; порівняльний аналіз та реалізація розглянутих алгоритмів.

1 ЗГОРТКОВІ НЕЙРОННІ МЕРЕЖІ

1.1 Набори даних для використання

Прискорення CNN актуальне для всіх сфер їх застосування, в тому числі і для обробки та класифікації, виявлення об'єктів, сегментації тощо. Класифікація зображень вважається найбільш типовим завданням для CNN та більшості наукових публікацій по даній темі використовують класифікаційні завдання для демонстрації своїх досягнень. Наступні набори даних часто використовуються для цих демонстрацій.

1.1.1 MNIST

База даних рукописних цифр MNIST є, ймовірно, найвідомішою набір даних у машинному навчанні. Вона складається з 70000 (60000 тренувань + 10000 тестів) 28×28 пікселів зображення у відтінках сірого, кожне з яких належить до одного з 10 класів. MNIST використовувався в статті про згорткові нейронні мережі [LeCun та ін., 1989], і досі залишається популярною оскільки її невеликий розмір і відносна простота дозволяють проводити експерименти та досягати результатів швидко.

1.1.2 CIFAR10 та CIFAR100

CIFAR10 і CIFAR100 є позначеними підмножинами великого немаркованого набору даних із 80 мільйонів крихітних зображень. Існують десять класів у CIFAR10 такі як: літак, автомобіль, птах, кіт, олень, собака, жаба, кінь, корабель і вантажівка. Ці набори даних подібні до MNIST за розміром (32 × 32 пікселя), але мають колір, і набагато різноманітніші. Наприклад, зображення, позначене як птах, може зображати птаха, що летить

у небі, або крупним планом страуса. Об’єкти, створені людиною, наприклад вантажівки і човни, можуть бути пофарбовані в різні кольори. Ця різноманітність робить CIFAR10 і особливо CIFAR100 набагато складнішими, ніж MNIST.

1.1.3 ILSVRC2012

ImageNet Large Scale Visual Recognition Challenge (ILSVRC), була запущена ще в 2010 році, хоча версія 2012 року є найбільш широко використовуваною. Це стало стандартним еталоном для великомасштабних об'єктів визнання. Найскладніший набір даних у цьому списку, він складається з 1,2 мільйона зображень наборів поїздів і 50000 у наборі перевірки з 1000 класів. Розмір зображень різний, але вони зазвичай змінюються до 256 × 256 пікселів. Набір даних ILSVRC є підмножиною ImageNet (ще більшої бази даних із 14 мільйонами зображень у 21841 ієрархічно організованих категорій), яка була позначена за допомогою краудсорсингу. Навчання нейронних мереж на наборах даних такого розміру стало можливим завдяки переходу з обчислення CPU до GPU але на практиці часто вимагається розгортання CNN на пристроях лише з CPU.

У той час як у цій кваліфікаційній роботі ми зосереджуємося виключно на процесорі та GPU, інші обчислювальні архітектури можна використовувати для прискорення роботи нейронних мереж. Ці архітектури включають програмовану вентильну матрицю (FPGA) і спеціальні інтегральні схеми для застосування (найпомітнішим прикладом є Tensor від Google. Блок обробки (TPU), спеціально розроблений для тензорного потоку. Хоча ці рішення забезпечують значне прискорення в деяких випадках, їх гнучкість обмежена дизайном, що є очевидним недоліком для застосування в дослідженнях. Точність ILSVRC2012 вважається адекватним показником ефективності моделі в реальних завданнях, і цей набір даних часто стає кінцевою метою для загальної мети CNN – прискорення алгоритми. CIFAR і

MNIST зазвичай використовуються для попередніх експериментів. Однак між ILSVRC2012 і ними є кардинальна різниця в наборах даних.

1.2 Блоки CNN

Нейронні мережі організовані як стек перетворень (шарів) і ключового компоненту CNN, який дав назву моделі, є згортковий шар. Нейронні мережі працюють з даними, організованими в 2D-масиви, які також називають картами або каналами, 3D або 4D масиви. N-вимірний масив також називають тензором n-го порядку, хоча в глибокому навчанні ці два терміни іноді змішуються. Згортка в CNN базується на концепції лінійної фільтрації зображень, яка була використовувався задовго до сучасної ери CNN. Лінійний фільтр приймає вхідний двовимірний масив (карта, канал) U, застосовує до нього 2D-фільтр (або ядро) і створює інший 2D-масив V. Фільтрування можна визначити як згортку

$$V(x,y) = \sum_{i,j} W(x-i, y-j)U(i,j) = \sum_{i,j} W(i,j)U(x-i, y-j) \tag{1.1}$$

або як взаємне кореляція

$$V(x,y) = \sum_{i,j} W(i,j)U(x+i, y+j) \tag{1.2}$$

(1.1) можна перетворити на (1.2) і навпаки, перевернувши фільтр W. У багатьох CNN у рамках навчання, а також у решті роботи використовується формула крос-кореляції, але за традицією відповідний шар ще називають згортковим. Межі підсумовування визначаються розміром фільтра W, який тут передбачається квадратом d × d. Індекси поза межами в U обробляються шляхом доповнення вхідних даних тензору, зазвичай з нулями. Ця підкладка

зазвичай невелика і не заважає алгоритмів, розглянутих у цій кваліфікаційній роботі.

Тоді як (1.1) і (1.2) обробляють одноканальні (відтінки сірого) зображення, кольорові зображення представлені у вигляді стека з 3 каналів (або карт) або одного 3D-масиву. Так само дані, що проходять між згортковими шарами, також представлені тривимірним масивом із кількістю каналів, які традиційно більші за 3. Із запровадженням третього виміру на вхід U фільтр W також стає тривимірним масивом:

$$V(x,y) = \sum_{i=1}^{d}\sum_{j=1}^{d}\sum_{c=1}^{C} W(i,j,c)U(x+i,y+j,c) \quad . \qquad (1.3)$$

Нарешті, застосування кількох 3D-фільтрів призводить до кількох вихідних карт, які є зібрані в єдиний вихідний масив 3D. Потім фільтри організовуються як єдиний 4D-масив:

$$V(x,y,k) = \sum_{i=1}^{d}\sum_{j=1}^{d}\sum_{c=1}^{C} W(i,j,c,k)\,U(x+i,y+j,c) \quad . \qquad (1.4)$$

Це перетворення тривимірних масивів (або тензорів третього порядку) називається узагальненою згорткою. Він входить до складу нейронних мереж як згортковий шар, так і масив W розглядається як параметр моделі. Альтернативні назви для W можуть бути: згортка ваги, згорткове ядро, ядерний тензор. У згорткових шарах кожен нейрон з’єднаний з невеликою підгрупою нейронів у попередній шар, і ця підмножина називається рецептивним полем. Число з плаваючою комою операції в згортці можна оцінити як $HWCNd^2$, де H, W і C є розміри вхідного масиву, N – кількість фільтрів, d – розмір фільтра. Інші блоки CNN включають:

- у повнозв’язному шарі кожен нейрон з’єднаний з усіма нейронами

попереднього шару. Цей шар приймає вхідний вектор x з елементами C і множить його на матрицю вагових коефіцієнтів $W \epsilon R^{C \times N}$, що створює вихідний вектор y з N елементів у CN операції з плаваючою комою. Якщо вхід не є вектором, він змінюється та обробляється як вектор;

- нелінійність. Лінійні шари, такі як згортковий і повнозв'язаний шар, перемежовуються нелінійностями. Найпопулярнішою нелінійністю є функція випрямленої лінійної одиниці (ReLU) f(x) = max(0,x). Це обчислювально дешева операція застосовується поелементно, тому її вартість незначна порівняно з іншими компонентами CNN. Оскільки нелінійність майже завжди присутня після повнозв'язного або згорткового рівня, її часто опускають в описі архітектури;

- пакетна нормалізація. Шар пакетної нормалізації нормалізує вхідні дані для нульового середнього та одиниці стандартного відхилення. Запровадження пакетної нормалізації може значно прискорити конвергенцію навчання та покращити фінальний результат. Нормалізація така ж повсюдна, як і нелінійність у сучасних архітектурах, її також можна опустити в схемах архітектури. Згорткові шари зазвичай мають найбільшу кількість операцій і споживають найбільше пам'яті та часу в CNN, як показано на рисунку 1.2. Кілька конкретних випадків і модифікацій загальної згортки, які використовуються для швидшого збереження збірки моделі, показані на рисунку 1.1 і детально описані нижче.

1×1 згортка. Складність згортки зменшується для менших просторових розмірів ядра, а найменший можливий розмір дорівнює 1. У цьому випадку згортка (1.1) зменшується до лінійної комбінації вхідних каналів:

$$V(x, y, k) = \sum_{c=1}^{C} W(c, k)\, U(x, y, c) \quad . \qquad (1.5)$$

Окрім невеликої кількості операцій, цю операцію можна ефективно реалізувати множення матриці.

Групова згортка. Ще один спосіб зменшити вартість згортки – скоротити її з'єднання між вхідним і вихідним каналами. Ідея реалізована шляхом поділу вхідних і вихідних каналів на кілька груп G_k і розрізання всіх з'єднянь між різними групами:

$$V(x,y,k) = \sum_{i=1}^{d} \sum_{j=1}^{d} \sum_{c \in G_k} W(i,j,c,k)\, U(x+i, y+j, c) \quad . \qquad (1.6)$$

Групова згортка спочатку була запропонована як спосіб побудови моделі, яка може бути розпаралелена між двома графічними процесорами, а потім ця концепція була відроджена як блок швидких CNN.

Поглиблена згортка. Якщо кількість груп збігається з кількістю вхідного та вихідного каналів, згортка називається згорткою по глибині. У цьому випадку кожен канал фільтрується незалежно одним фільтром:

$$V(x,y,k) = \sum_{i=1}^{d} \sum_{j=1}^{d} W(i,j,k)\, U(x+i, y+j, k) \quad . \qquad (1.7)$$

Кількість каналів у вихідному масиві в цьому випадку фіксується до числа каналів у вхідному масиві. Згортка по глибині вимагає C разів менше операцій з плаваючою точкою порівняно зі звичайною згорткою такого ж розміру, але фактичною терміни залежать від ефективності реалізації.

Але на практиці доступні CNN не відповідають обмеженням швидкості Як варіант можливе використання більш потужного обладнання або ж 2. обрання більш ефективної реалізації операції згортки. Також можливе налаштування моделі з використанням швидких наближень.

Перехід на нове апаратне забезпечення (що включає перехід від CPU до GPU) є найпростішим варіантом, оскільки в сучасних фреймворках глибокого навчання це не вимагає будь-яке програмування. Збільшення

швидкості обробки та обсягу пам'яті в сучасних GPU є одним із основних факторів, що розширюють можливості нейронних мереж. Хоча питання апаратного забезпечення та реалізації здебільшого виходять за рамки цієї роботи, деякі аспекти впровадження можуть впливати на приблизне прискорення методів.

Дана реалізація операції згортки (1.4) має кілька вкладених циклів, в загальному випадку їх шість. Малі розміри ядра роблять внутрішні цикли дуже неефективними, оскільки вони часто викликають інструкції JMP. Крім того, кроки прямого та зворотного розповсюдження вимагають доступу як по рядках, так і по стовпцях вводу та ядра, функція, яку неможливо ефективно реалізувати із загальними даними.

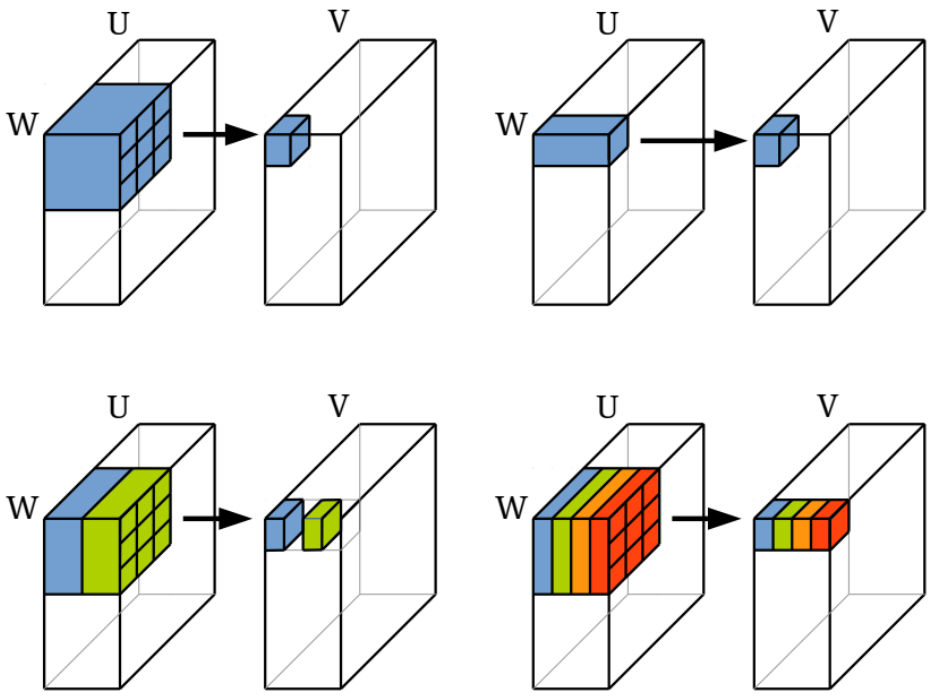

Рисунок 1.1 – Варіанти згортки у сучасних розробках CNN

Варіанти згортки, що використовуються в сучасних конструкціях CNN представлені на рисунку 1.1. Стандартна згортка з фільтрами 3×3. Згортка 1×1, яка переставляє вхідні карти та не фіксує співвідношення сусідніх

пікселів. Вхідні та вихідні карти є поділено на дві групи (позначено синім і зеленим), між якими немає зв'язків їх. У згортці по глибині всі карти обробляються незалежно.

Це так звана розгорнута згортка. Основною ідеєю є зведення згортки до множення двох матриць шляхом дублювання вхідних даних. Зниження дозволяє використовувати високооптимізовані реалізації множення матриці (варіанти BLAS бібліотеки), які розроблялися протягом багатьох років для різних обчислень архітектури, включаючи процесор і графічний процесор. Зменшення показано на рисунку 1.3.

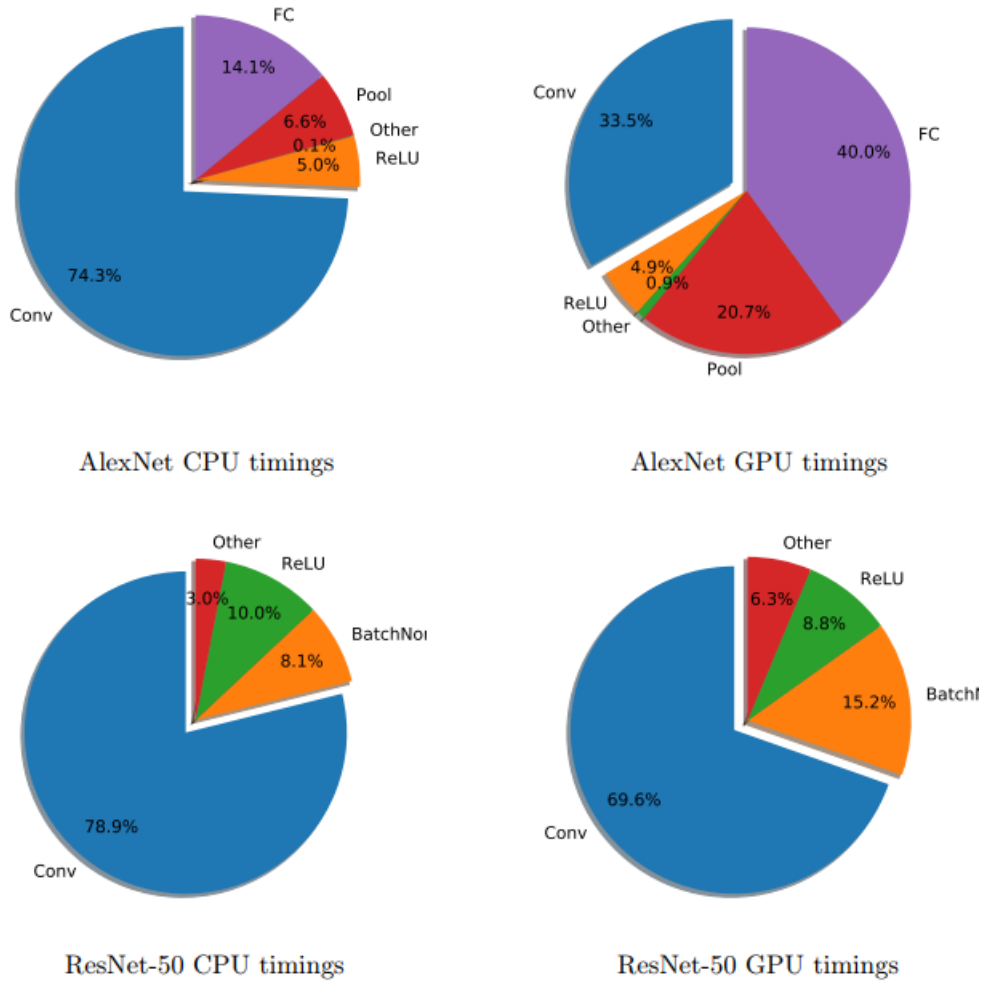

Рисунок 1.2 – Таймінг різних рівнів для архітектур AlexNet і ResNet-50 включені ЦП (Intel Core i7-6800K) і графічний процесор (GeForce GTX 1080), виміряні у рамках Pytorch за допомогою вбудованого профілювача

Сучасна реалізація згортки на GPU настільки ефективна, що для відносно неглибокої архітектури, такої як AlexNet, найбільша частина робочого часу витрачається на повнозв’язані шари. У випадку з процесором також для більш глибоких архітектур основна частина часу споживається згортковими шарами. Наприклад, в ResNet-50, один повністю підключений рівень цієї архітектури займає менше ніж 0,5% від загального часу роботи як на CPU, так і на GPU.

Ще один спосіб створення швидшої реалізації для згортки базується на згортковій теоремі, яка стверджує, що кругові згортки в просторовій області еквівалентні поточковим добуткам в області Фур'є. Позначаючи перетворення Фур’є як F і зворотне перетворення як F−1, ми можемо виразити згортку двох 2D відображає f і g таким чином:

$$f * g = \mathcal{F}^{-1}\left(\mathcal{F}(f)\mathcal{F}(g)\right). \tag{1.8}$$

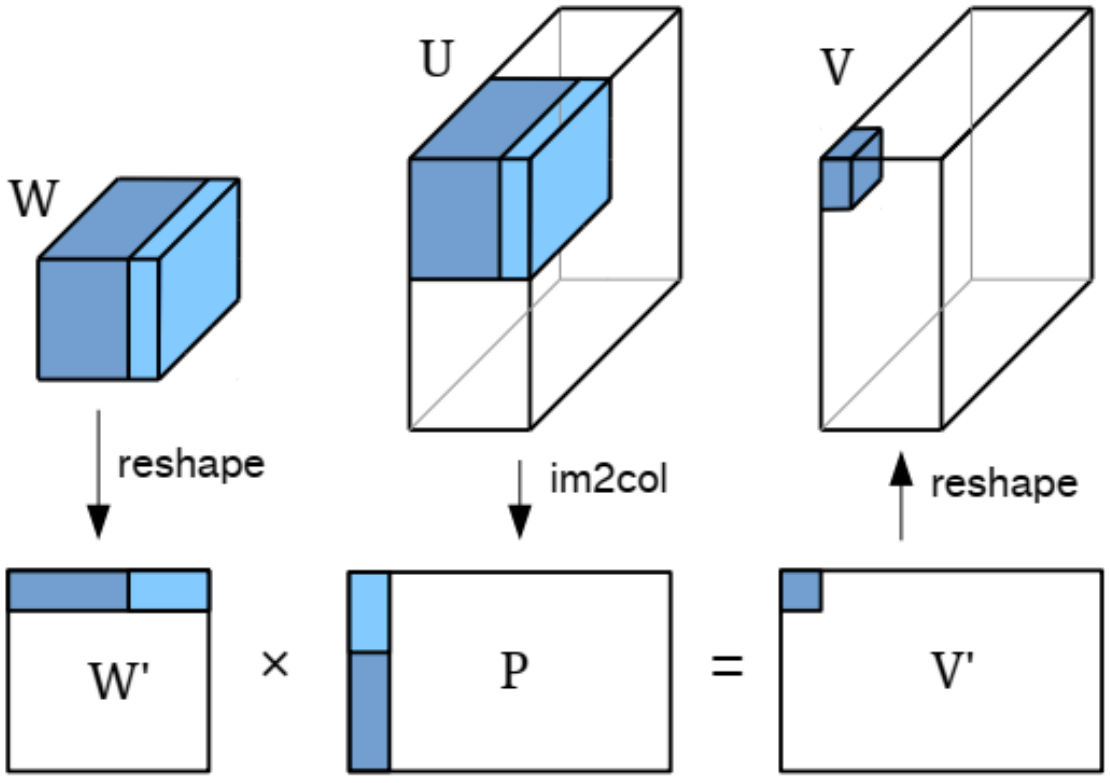

Рисунок 1.3 – Зведення згортки до множення матриці через розгортання

Введений масив U перетворюється на матрицю P за допомогою операції im2col. Стовпці P створено з розгаданих патчів з U, розмір патча визначається розміром фільтрів у ваговому масиві W. Матриця патчів P потім множиться на вагову матрицю W’ (отримано з W шляхом зміни форми), в результаті чого вихідна матриця V’. Остаточний вихідний масив V отримано з V’ іншим переформуванням. Синім виділено один патч у U з відповідним стовпцем P, відфільтроване в W з відповідним рядком W’ і один вихідний піксель, створений цим фільтром у цьому патчі. Представлено не тільки з фільтрів W, оскільки важко візуалізувати чотиривимірний масив.

Основна перевага цього підходу полягає в тому, що його обчислювальна складність не залежить від розміру фільтра, що вигідно для більших фільтрів. Недоліки: більші вимоги до пам'яті для зберігання карт функцій і фільтрів у домені Фур'є, а також можливе уповільнення у випадку малих фільтрів. Оскільки в сучасних архітектурах використовуються фільтри в основному невеликі, цей спосіб не дуже поширений. Інші ефективні підходи до реалізації згорток з малими ядрами включають використання алгоритму швидкого множення матриць Штрассена та алгоритм мінімальної фільтрації Winograd.

1.3 Архітектури CNN

1.3.1 LeNet

LeNet – це проста архітектура CNN, спочатку запропонована і досі часто використовується для набору даних MNIST. Вона складається з двох згорткових, двох об'єднаних і двох повністю з'єднаних шарів. Оригінальна архітектура включала нелінійний тангенс та блоки RBF, які зазвичай замінюються ReLU і звичайним повнозв’язним шаром у сучасній реалізації цієї архітектури.

1.3.2 AlexNet

AlexNet була першою CNN, успішно навченою у великому масштабі набору даних, прорив, який призвів до перемоги на конкурсі ILSVRC2012. AlexNet мав деякі особливості: фільтри різного розміру, включаючи великі фільтри на перших шарах і відносно невеликій глибині. Хоча сучасні CNN перевершують AlexNet у кожному аспекті він все ще часто використовується як загальна базова лінія. AlexNet швидко порівнюється до найглибшої та найточнішої з передових CNN, але не в порівнянні з найшвидшою архітектурою з однаковим рівнем точності.

Ранні архітектури CNN використовували великі згорткові фільтри, такі як фільтри 5×5 в AlexNet і LeNet, і AlexNet навіть мали фільтри 11×11 на першому шарі. Цей великий розмір спостереження за плавністю навчених фільтрів, що вказує на набагато менші потріби для визначення фільтра за допомогою інтерполяції чи іншої процедури. Подібність кількох фільтрів до вертикальних або горизонтальних фільтрів виявлення країв вказує на конкретний метод роздільних фільтрів. Роботи з роздільності та розширення до тензорних розкладів розглядаються в наступному розділі.

1.3.3 VGG

VGG – це сімейство CNN, що визначається двома ключовими характеристиками: значною глибиною та ексклюзивністю використання фільтрів 3×3, що є найменшим розміром для захоплення поняття ліворуч/праворуч, вгору/вниз, центр.

Успіх архітектур VGG поклав початок збільшення глибини сучасних архітектур CNN: чим більше шарів можна скласти, тим краще. Архітектури VGG дуже повільні та навіть важчі за кількістю параметрів, переважно через масивні повністю пов’язані шари. Мережі VGG все ще популярні серед дослідників, оскільки мають просту структуру, і навіть домінують у деяких

застосуваннях, таких як дрібнозерниста класифікація та стилізація зображення.

1.3.4 ResNet

Слід зазначити, що деяка межа глибини CNN все ще існує: коли мережа стає глибшою, її точність насичується і після досягнення деякої межі запуск швидко деградує. Ця проблема спричинена не переобладнанням, а несправністю тренувального процесу. Було запропоновано новий підхід до проекту, щоб полегшити градієнт розповсюдження через мережу і, отже, сприяюнню процесу навчання. Головна ідея полягає в тому, щоб організувати блоки CNN як залишкову функцію де x – вхід, а h(x) – блок згорткових шарів

$$f(x) = h(x) + x \quad . \tag{1.9}$$

Залишкове навчання долає проблему зниження точності та дозволяє ефективно навчати CNN до тисячі шарів, хоча ці надзвичайні розміри можливі лише на наборах даних із невеликими розміри введення, як CIFAR. Члени сімейства моделей ResNet позначаються кількістю шарів у мережі, серед яких ResNet-50 середнього розміру є найпоширенішою моделлю. Спроби покращити оригінальну архітектуру ResNet включають ResNeXt, яка вводить групові згортки до залишкового блоку, і DenseNet, що створює додаткові зв’язки між залишковими блоками.

1.3.5 GoogleNet

Сімейство Inception також відоме як GoogLeNet. Основна ідея полягає в початковій архітектурі, яка заснована на з'ясуванні того, як оптимальна

локальна розріджена структура у мережі згорткового бачення може бути наближена та охоплена легкодоступними щільними компонентами. Цей принцип призвів до побудови блоку Inception, який складається з кількох гілок, кожна з яких має фільтри певного розміру. Цей блок повторюється кілька разів разів, що призводить до дуже глибокої нейронної мережі.

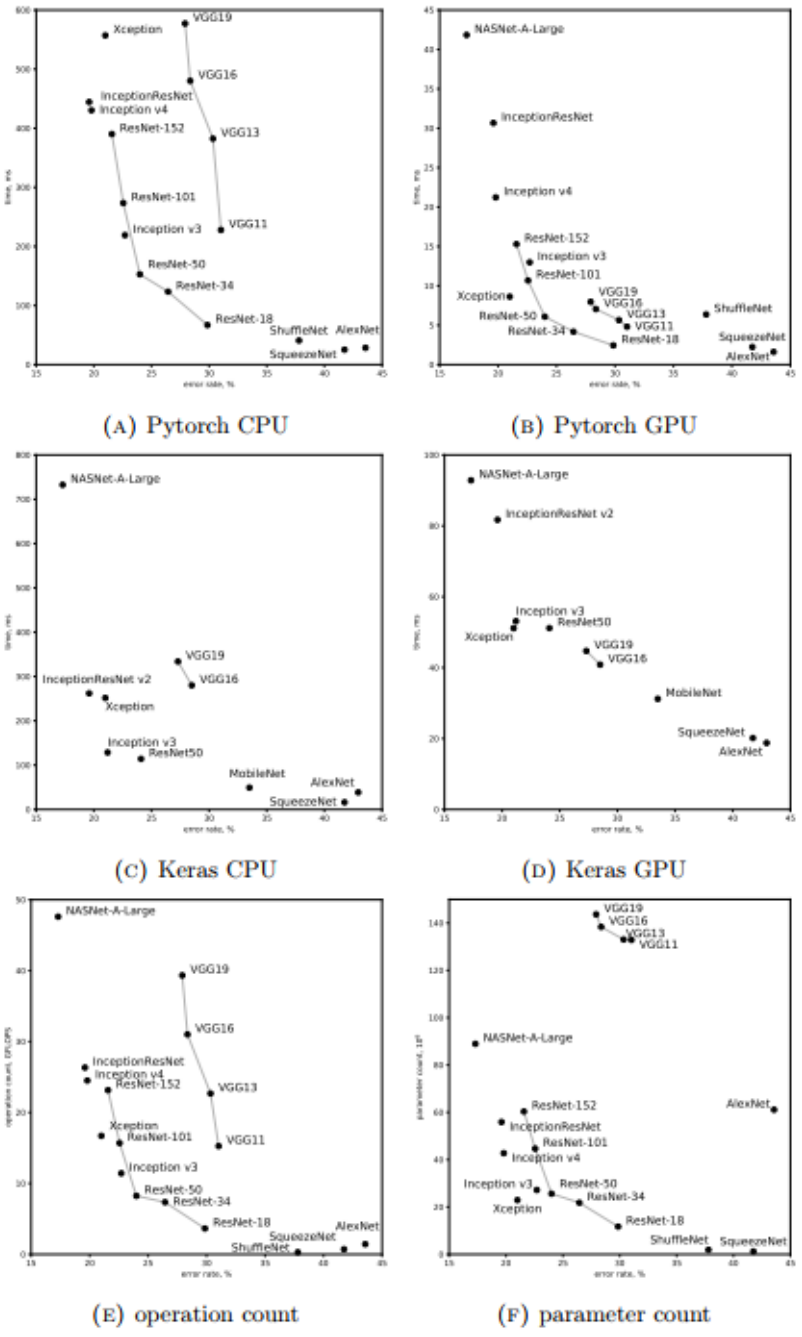

Рисунок 1.4 – Компроміс між часом висновку та помилкою класифікації ILSVRC Top-1 для деяких архітектур CNN

У наш час початкова сім'я включає декілька версії моделі Inception, модель Xception, яка використовує згортки, що розділяються по глибині, і гібридну модель під назвою Inception-ResNet.

Точність має пріоритет над швидкістю та компактністю в основних дослідженнях CNN, які певний час рухаються в напрямку збільшення глибини CNN. Тим не менш, навчання найглибшої з сучасних CNN неможливе без уваги до ефективності спроектованої архітектури.

Принципи ефективного проектування архітектури і CNN, розроблені для максимальної швидкості, детально розглянуті в розділі 2.2. Відповідність між часом висновку та точністю описаних архітектур CNN показано на рисунку 1.4.

Чотири діаграми порівнюють продуктивність CNN на центральному та графічному процесорах два фреймворки, Pytorch і Keras із бекендом Tensorflow і зарахована операція показані на п'ятій діаграмі. Це порівняння виявляє вплив апаратних засобів і деталі програмного забезпечення щодо відносної продуктивності різних архітектур, особливо якщо мережа використовує нестандартний тип згортки, такий як групова згортка або згортка по глибині згортка.

Це явище можна спостерігати за допомогою архітектури Xception, яка різко змінює своє положення відносно сусідніх моделей, напр. ResNet-50. Ці зміни відбуваються не лише з фреймворком і перемикачем CPU/GPU, а й між різними версіями одного фреймворку та різних моделей GPU. Хоча ці фактори надзвичайно важливі на практиці, ми залишаємо більшість деталей обладнання та впровадження програмного забезпечення поза межами цієї роботи та зосередимося на алгоритмах та ідеї наближення.

Хронологічно перша модель, AlexNet, лежить у нижньому правому куті у всіх версіях діаграми, тобто це одна з найшвидших і найменш точних моделей. Наступне покоління моделей, таких як VGG, ResNets і різних моделей з сімейства Inception, займають центральну та ліву частини діаграми.

Оптимальні моделі лягають на нижню частину цієї діаграми. Загальна мета прискорення нейронних мереж полягає в тому, щоб натиснути цей конверт вниз і вліво. В експериментах із фреймворком Pytorch, центральна частина цього конверта містить моделі ResNet і моделі сімейства Inception. Моделі VGG у всіх випадках, за винятком обчислень GPU з Keras, розташовані вище, що означає, що вони не є оптимальними. Ліва верхня частина конверта відповідає до найточніших моделей, які обмінюють велику швидкість на точність. Найповільнішою і найточнішою серед моделей, показаних на цій діаграмі, є NASNet, створена за допомогою автоматичного пошуку архітектури.

1.4 Висновки до розділу 1

Проаналізовано існуючі архітектури згорткових нейронних мереж новий алгоритм прискорення CNN, заснований на CP-розкладі низького рангу згорткових ваг. Досліджений метод використовує структуровану розрідженість, по суті змінюючи форми фільтра зі спеціальними обмеженнями. Потім ми накладається структурована розрідженість на нейронну мережу шляхом навчання за допомогою регуляризатора, що викликає розрідженість, і скорочення. Метод оцінюється та ретельно порівнюється з базовими лініями на MNIST і наборами даних ILSVRC.

2 АНАЛІЗ АЛГОРИТМІВ ПРИСКОРЕННЯ ЗГОРТКОВИХ НЕЙРОННИХ МЕРЕЖ

Огляд існуючих алгоритмяв та методи почнемо з попередньо навченої CNN, а потім перемежовують перетворення згорткових шарів операціями тонкого налаштування. Кожен крок трансформації призводить до прискорення, а також до падіння точності. Подальша операція тонкого налаштування відновлює частину падіння точності. З іншого боку, методи, спрямовані на швидке проектування архітектури (включаючи ті, які роблять це автоматизованим способом), мають на меті розробку архітектури, яку можна навчити з нуля. Нарешті, методи «вчитель-учень» займають середню нишу, оскільки вони зазвичай створюють остаточну архітектуру у двоетапному процесі, який спочатку навчає повільну мережу вчителів, а потім навчає швидку мережу учнів, використовуючи вказівки від вчителя.

2.1 Тензорна декомпозиція

Шари згортки, які відповідають основній частині часу виводу в сучасній CNN базуються на узагальненій операції згортки:

$$V(x,y,k) = \sum_{i=1}^{d} \sum_{j=1}^{d} \sum_{c=1}^{C} W(i,j,c,k)\, U(x+i, y+j, c)\,, \tag{2.1}$$

де U позначає вхідний 3D-масив, що містить C 2D-карти зображень, V позначає вихідний тривимірний масив, що містить N карт двовимірних зображень, а W є чотиривимірним ваговим масивом.

Таким чином, згортковий шар визначається його чотиривимірною вагою масиву W . Ідея методів тензорної декомпозиції полягає в тому, щоб розкласти цей масив на добуток тензорів низької розмірності, що призводить

до кількох швидких згорток із меншою кількістю операцій. Фільтр декомпозиції для прискорення згорток спочатку був розроблений не в контексті CNN. Спочатку розглядали завдання знешумлення та наближені згорткові фільтри за допомогою спільного набору роздільних фільтрів, що призводило до значного прискорення з мінімальною втратою в точності шумозаглушення.

Підхід, заснований на роздільних фільтрах, був потім поширений на згорткові шари CNN. Згорткові ваги з квадратними фільтрами d×d апроксимуються та замінюються добутком двох тензорів із фільтрами 1×d та d×1 і K карт функцій між ними. Параметр K (ранг розкладання) регулює компроміс між швидкістю та точністю: мале значення K призводить до швидких, але неточних моделей, тоді як велике K дозволяє точно відтворити оригінальну згортку, але з великим часом обчислення. Ретельне налаштування K для кожного апроксимованого шару є важливою частиною прискорення нейронної мережі в цьому алгоритмі.

Існує два підходи до оптимізації, залежно від того, чи мінімізується мета декомпозицією. Процедура вимірює похибку реконструкції фільтра або похибку реконструкції активації блоку. З двох підходів останній є більш практичним, наскільки це можливо стає частиною процесу тонкого налаштування CNN, яке оптимізує втрати під час навчання навчання CNN по всіх параметрах мережі.

Кілька методів стиснення вагового тензора на основі кластеризації були запропоновані в науковій літературі. Початкова ідея полягає в тому, щоб згрупувати тензорні зрізи вздовж одного або двох чотиривимірного масиву (так звана бікластеризація), щоб розділити тензор відповідно до кластерів межі, а потім апроксимувати отримані зрізи значеннями центроїда ("монохроматичне наближення"), за допомогою SVD-розкладу або за допомогою канонічного поліадичного CP-розкладу, отриманого жадібним підходом (зовнішнього продукту). CP-розклад є одним із узагальнень SVD-розкладу на тензори вищого порядку. Розщеплення зменшує ранги

розкладання, необхідні для хорошої апроксимації, але це відбувається ціною збільшення кількості тензорів, які необхідно апроксимувати. Недоліком цього підходу є досить велика кількість гіперпараметрів, які стає все важче налаштувати оптимальну продуктивність.

Існує також інший алгоритм, який застосовує CP-розклад до згорткових ваг. В даному випадку, замість кластеризації, робота фокусується над налаштуванням продуктивності CP-декомпозиції, яка виконується в два етапи. Перший етап починається з кращого початкового наближення, отриманого за допомогою нелінійного мінімуму алгоритму квадратів. Другий етап точно налаштовує всю мережу після застосування декомпозиції.

Тонка настройка моделі після декомпозиції нетривіальна, особливо якщо декомпозиція наноситься в кілька шарів. Навіть якщо один шар розкладається, числові нестабільності всередині чотирьох згорткових шарів, введених CP-розкладом, часто призводять до вибуху градієнтів під час тонкого налаштування.

Ключова проблема полягає в тому, що розроблений блок CNN слідує структурі тензорного розкладання та не має нелінійності між ними. Якщо кілька шарів розкладено, точне налаштування можна виконати або один раз після застосування всіх розкладів, або ітераційно після кожного окремого розкладання.

Ітеративний підхід стверджує, що метод тензорної потужності є найбільш прийнятним для отримання початкового розкладання. Навчання з нуля впритул до архітектури пов’язане з таким, отриманим за допомогою CP-розкладу, було реалізовано раніше.

Робота Жанга (2016 р.) пропонує альтернативний спосіб прискорення роботи нейронних мереж шляхом застосування припущення низького рангу до активацій, а не до тензора ваги. Це припущення ділить один згортковий шар на два, а ваги кожного з них на два нових шари отримані шляхом вирішення задач оптимізації. Акцент на активаціях замість ваг має дві переваги: по-перше, можна враховувати нелінійності, по-друге, у випадку

кількох шарів наближення, активація вихідної мережі може бути використана як ціль. Ця ідея, яка називається асиметричною реконструкцією, обмежує накопичення помилок від шару.

Нарешті, як і в попередніх підходах, всю мережу можна точно налаштувати. Усі перераховані вище методи замінюють один згортковий шар на блок менших. Порівняння цих блоків представлено на рисунку 2.1.

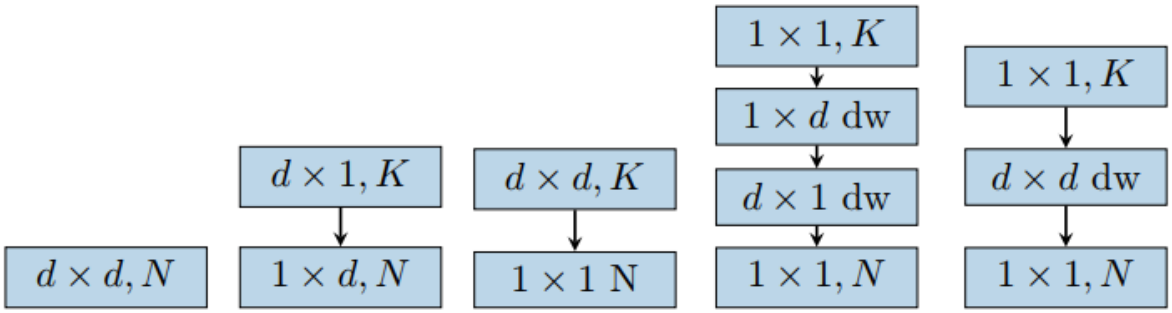

Рисунок 2.1 – Блоки CNN, які використовуються методами тензорної декомпозиції для реплікації одного згорткового шару

Кожен шар тут позначено формою ядра та кількістю фільтрів. Позначка «dw» означає згортку по глибині, у одному з випадків кількість каналів на вході те саме, що й на виході.

Інші тензорні розкладання вищого порядку використовувалися для прискорення CNN. Такі розкладання було застосовано для прискорення та стиснення CNN. Було застосовано декомпозицію тензорного потоку. повнозв’язних шарів згорткових нейронних мереж.

2.2 Швидке проектування архітектури

Дослідження в CNN призвели до появи кількох популярних сімейств архітектур. Історично пошук архітектури був зумовлений бажанням просувати точність класифікації, тоді як швидкість виводу була другорядною проблемою. Підходи тензорної декомпозиції тісно пов’язані із завданням

проектування оптимальної архітектури, що є темою цього розділу. Методи, описані в цьому розділі можуть навчати спроектовані архітектури з нуля, і на вибір проекту часто безпосередньо впливають попередні роботи з тензорної декомпозиції.

Однією з перших помітних спроб побудови архітектури, яка претендувала на ефективність, була архітектура мережі в мережі (NIN). Основна ідея NIN полягає в тому, щоб замінити нелінійності в згортковій мережі більш складною функцією. Багатошаровий (двошаровий) персептрон, який відомий як універсальний апроксиматор, вибирається як заміна. Поділяючи ваги перцептронів через просторові виміри, один закінчується двома згортковими шарами 1×1, які перемежовуються стандартними нелінійностями.

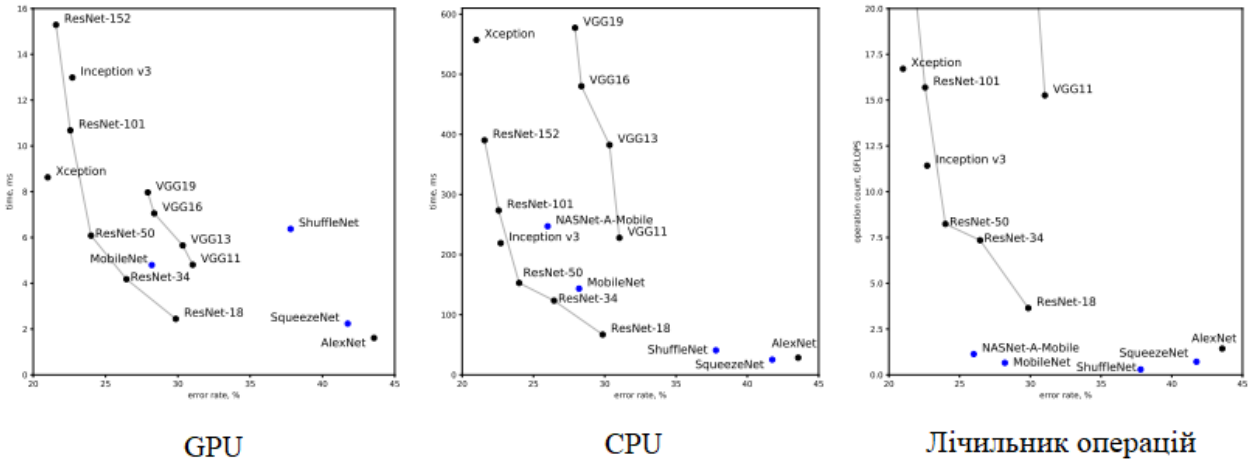

Рисунок 2.2 – Діаграма точності ILSVRC і часу виводу для найшвидшої з доступних архітектур CNN

Таймінги вимірюються як на CPU (Intel Core i7-6800K), так і на GPU GeForce GTX 1080 з Pytorch 0.3. Кількість операцій апаратно-незалежної міри складності моделі також показана праворуч. Ефективні CNN, описані в цьому розділі позначені синіми точками.

NASNet-A-Mobile виключено з лівої діаграми тому що час висновку на GPU занадто високий порівняно з іншими моделями. Ефективними є такі моделі, як MobileNet і ShuffleNet мають бути швидшим порівняно зі звичайними моделями, такими як AlexNet. У цьому експерименті швидкі моделі здаються швидкими лише з точки зору кількості операцій, але не фактично часу виконання. Цей результат, ймовірно, викликаний відносно неефективною реалізацією поглибленої сепарабельної згортки в Pytorch.

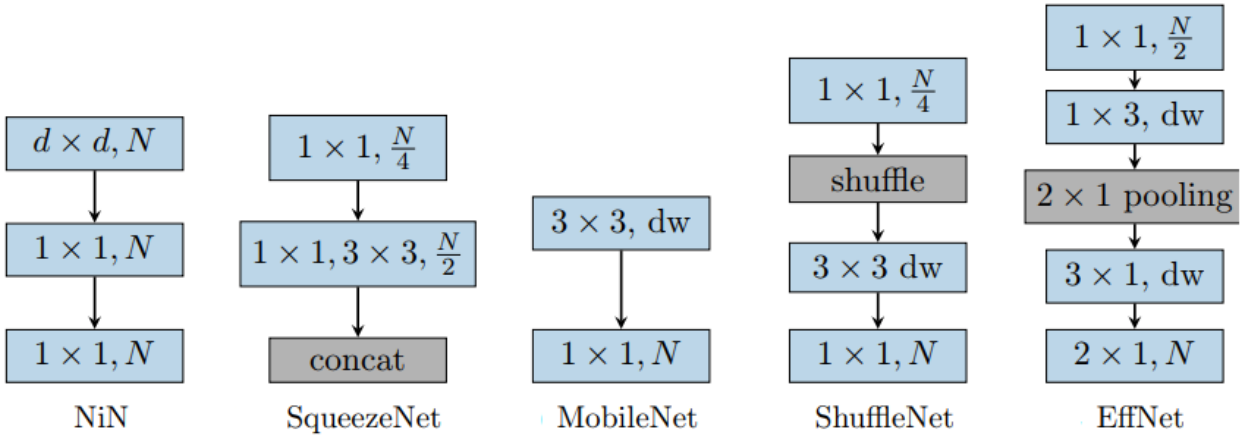

Рисунок 2.3 – Послідовності згорткових шарів, що використовуються для швидких і компактних архітектур

SqueezeNet – це компактна архітектура, яка забезпечує продуктивність AlexNet рівень із у 50 разів меншими параметрами. Ідеологія SqueezeNet заснована за трьома принципами:

- за можливості використовуйте фільтри 1×1 замість фільтрів 3×3. Менші фільтри мають менше параметрів і потрібно менше операцій;

- якщо необхідно застосувати фільтрацію 3×3, мінімізуйте кількість вхідних каналів;

- знижте частоту дискретизації в мережі, щоб дати більшості рівнів можливість працювати з великим карти високої роздільної здатності.

Перший і другий принципи гарантують, що модель одночасно маленька і швидка, а третій принцип конструкції підвищує точність. Усі три

принципи разом дають дуже маленьку модель із трохи меншим часом висновку, ніж AlexNet (яка служить в якості базової моделі).

Архітектура MobileNet є визначною серед архітектур CNN, яка розроблена для оптимального розміру та часу висновку. Основна ідея полягає в тому, щоб розділити фільтрацію та особливість побудови функції згорткового шару на два шари: поглиблену згортку та згортку 1×1 (тобто поточкову згортка). Ця комбінація є називається згорткою, що розділяється по глибині.

MobileNet також використовує прості, але ефективні прийоми для керування продуктивністю архітектури: ширина мережі (кількість каналів) контролюється множником ширини α, а вхідна роздільна здатність контролюється множником роздільної здатності ρ. В загальному вигляді, зміна ширини мережі та вхідної роздільної здатності є простим способом контролювати компроміси між швидкістю та кількістю параметрів з одного боку та точністю з іншого інші.

У порівнянні з SqueezeNet, MobileNet здатна досягти вищої точності, хоча і мають приблизно однаковий розмір моделі та одну десяту операцій мультидодавання. Ця перевага в кількості операцій, однак, важко перевести на фактичне прискорення висновку на GPU, тому що глибинна згортка не так ефективно реалізована на GPU, як звичайна згортка.

Тим не менш, розділення глибинних і внутрішньоканальних згорток стало популярною ідеєю. У сучасних архітектурах ця ідея призводить до домінування згортки 1×1 загальної вартості обчисленної моделі. Єдиний спосіб стиснути згортки 1×1 ще далі – перетворити їх на групові згортки, які можуть лише змішувати канали в межах певних груп каналів. Однак таке групування буде означати, що вся мережа поділена на тонкі стовпчики без зв’язку між собою їх.

ShuffleNets вирішують цю проблему за допомогою перемішування каналів. Перемикання каналів між згортками дозволяє насолоджуватися низькими витратами групових згорток без розбиття мережі на непересічні

частини. ShuffleNet архітектура використовує ще меншу кількість операцій для того самого рівня точності порівняно з MobileNet. Залежно від того, чи вплине ця перевага на фактичний час на ефективність наявних реалізацій поглибленої та групової згорток.

Іншим варіантом блоку MobileNet є архітектура EffNet, мотивована ретельним вивченням. EffNet використовує згортки, що розділяються по глибині і штовхає його ще далі, розбиваючи згортку 3×3 на пару згорток з 1× 3 і 3×1 ядер. Зменшення дискретизації вздовж одного виміру виконується за допомогою крокової згортки, а вздовж інших - через об'єднання 2×1.

Останнім логічним кроком у русі до менших фільтрів у CNN буде повна відмова від згорток з файлами розміром більше 1×1. Проблема з цим видом CNN полягає в тому, що сусідні пікселі не будуть з'єднані, а сприйнятливі поля завжди будуть 1×1.

ShiftNets вирішують цю проблему за допомогою зсувів каналів, що дозволяє суміжним пікселям у різних каналах з’єднуватися через згортки 1× 1. Зсув каналу є дешевою операцією, але ShiftNet часто вимагає збільшення числа каналів у мережі для досягнення однакового рівня продуктивності.

Розглянуті блоки ShiftNet та інших перерахованих вище архітектур показані на рисунку 2.3 (показано час роботи, кількість операцій і точність ILSVRC описаних архітектури). А на рисунку 2.2 показано час роботи, кількість операцій і точність ILSVRC описаних архітектур.

Принципи проектування легкої архітектури також корисні для проектування великих важких архітектур. На практиці пам’ять на GPU та час на експерименти завжди обмежений, тому глибина архітектури CNN така також обмежена. Оскільки продуктивність CNN зазвичай зростає з глибиною, то бажано розширити цей ліміт шляхом ефективного проектування архітектур.

Ефективне проектування архітектури не обмежується завданнями класифікації, оскільки сегментація та виявлення вимагають спеціалізованих архітектур, які тим не менш, можуть мати ті самі принципи.

2.3 Автоматичний пошук архітектури

Зазвичай архітектури нейронної мережі створюються вручну людьми-експертами, які керуються загальними принципами ефективного проектування архітектури. Багато варіантів архітектурної конструкції залишена на інтуїцію та здогади. Ситуація вимагає створення автоматичного алгоритму пошуку архітектури.

Оскільки автоматизований пошук архітектури є галуззю, що швидко розвивається на чолі сучасних досліджень глибокого навчання, цей розділ охоплює лише більшість впливових праць в цій підсфері. Автоматичний пошук архітектури – це, по суті, оптимізація гіперпараметрів, яка є загальною проблемою, яку можна вирішити кількома підходами, такими як пошук по сітці або Байєсовській оптимізації.

Випадок CNN викликає кілька ускладнень. По-перше, оцінка функції стає дуже дорогою. По-друге, кількість гіперпараметрів дуже велика і може змінюватися в просторі оптимізації. Таким чином, гіперпараметрична оптимізація для CNN вимагає спеціальних підходів.

Можливо, першою успішною спробою автоматичного пошуку архітектури є алгоритм пошуку нейронної архітектури (NAS), який використовує навчання з підкріпленням. Архітектура CNN передбачена рекурентною нейронною мережею, яка послідовно виробляє гіперпараметри CNN: розміри файлів, кроки, кількість фільтрів. Можливі розгалуження і з’єднання пропуску моделюються за допомогою механізму уваги, який вирішує, яке з’єднання між поточними та попередніми шарами має бути введено в наступний крок.

NAS потребує надзвичайно великої кількості обчислювальних ресурсів, навіть якщо виконується на невеликих наборах даних. Таким чином, в експериментах із набором даних CIFAR-10 автори повідомляють про тестування 12800 архітектур під час процесу пошуку та використання 800

GPU одночасно. Це унеможливлює архітектуру шукати більші набори даних, такі як ImageNet.

До більших наборів даних можна підійти двома способами: шляхом розробки більш ефективного пошуку алгоритмів або шляхом забезпечення можливості передачі результатів, отриманих на невеликому наборі даних до великих наборів даних. Спочатку, модульна структура накладається на цільову CNN. Таким чином, лише архітектурі потрібно передбачити блок, а не всю мережу, збільшуючи простір для пошуку. По-друге, версії архітектур CIFAR-10 і ImageNet можуть бути виготовлені з однакових блоків з різною кількістю об’єднань (або ступінчастих згорток) між ними. Архітектури, побудовані таким чином, перемагають архітектури, створені людьми на ImageNet. Пошук нейронної архітектури можна додатково прискорити шляхом спільного використання параметрів різних архітектур або шляхом прогнозування остаточного виконання архітектури на основі перших епох на початку навчання процесу. Навчання з підкріпленням (RL) не є єдиним можливим способом пошуку в просторі архітектури CNN. Генетичний алгоритм ретельно порівнювали з базовою лінією RL. Генетичний алгоритм може швидше підібрати RL конвергенцію.

2.4 Процес відсікання частин згорткових ваг

Відсікання частин згорткових ваг є природним способом зменшити складність згорткової операції. Цей підхід застосовується для прискорення згорток у нейронних мережах, є популярною темою досліджень з дуже великою кількістю публікацій. Більшість підходів до скорочення слідують тому самому конвеєру. Починаючи з попередньо підготовленої базової лінії, наступні два кроки застосовуються, можливо, ітераційно. По-перше, важливість нейронів обчислюється за певним критерієм. Найменш важливими нейрони обрізають. Потім мережа точно налаштовується, що призводить до часткового відновлення падіння точності. У випадку

ітераційного процесу під час може бути застосовано регуляризатор, що викликає розрідженість етапу тонкої настройки. Щоб реалізувати цей конвеєр, потрібно зробити три основні варіанти. По-перше, бажане необхідно вибрати структуру розрідженості. По-друге, критерій важливості (обрізання). Має бути обраним. Нарешті, слід обрати регуляризатор, що викликає розрідженість (якщо підхід використовує один).

Оскільки деякі елементи W будуть замінені нулями за допомогою алгоритму скорочення, можна перейти до деякого розрідженого представлення для W. У відсутності структури розрідженості, множення розрідженої матриці несе значні накладних витрат порівняно з множенням щільної матриці. Як показано на рисунку 2.4, розріджена версія стає швидшою, лише якщо щільність W значно нижче 0:1, що є недосяжним у практичних умовах без істотного значення падіння точності. Єдиний спосіб подолати цю проблему – розташувати елементи W у групи та використовувати структуровану розрідженість. Хоча використання im2col не єдиний шлях, щоб реалізувати узагальнену згортку, інші реалізації слідують подібним шаблонам і так само не можуть отримати користь від середніх рівнів неструктурованої розрідженості.

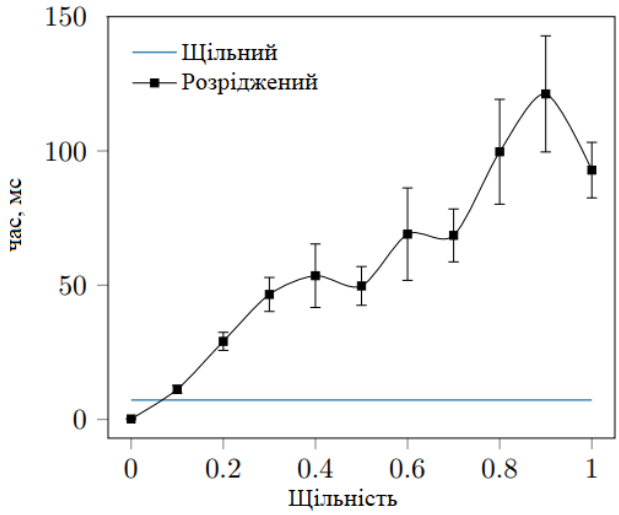

Рисунок 2.4 – Порівняння матриці множення з розрідженою (представлення CSC) і щільність вагових матриць

Розріджена версія стає ефективнішою лише для щільностей менше ніж 0,1. Ці таймінги CPU (Intel Core i7-6600U) обчислюються за допомогою numpy і бібліотеки scipy.sparse для матриць з розмірами, що відповідають згортці 3×3 з 64 каналами та картами 64×64.

Обговорене вище міркування вимагає використання структурованої розрідженості під час обрізки. Дані алгоритми видаляють стовпці з вагової матриці W* і відповідні рядки з матриці патчів U*. Видалення полегшується за звичаєм версії функції im2col, яка пропускає елементи, що відповідають видаленим частинам W* під час побудови матриці патча. З такою структурованою розрідженістю, оригінал множення матриць замінюється множенням менших матриць, які є ще щільними. Це призводить до прискорень, які майже прямо пропорційні щільності. Був запропонований пов’язаний, але ортогональний підхід до структурованої розрідженості, який називається перфорацією. Основна ідея полягає у видаленні стовпців із матриці латок U*. Оскільки стовпці відповідають фрагментам зображення, це означає, що згортки не будуть обчислюється для деяких підмножин точок на зображенні. Вихідне значення в цих точках може інтерполювати або фактично пропускати, якщо наступний рівень виконує операцію об’єднання.

2.5 Навчання з вчителем

Підходи «навчання з вчителем» базуються на ідеї, що модель CNN можна навчити на результати іншої моделі (вчителя), на відміну від звичайного навчання на позначених даних. Такий підхід дозволяє передавати знання з однієї моделі в іншу та включати немарковані або синтетичні дані в процес навчання (як немаркований приклад все ще можна пройти через модель попередньо підготовленого вчителя). Іншою метою підходу «вчитель-учень» була б передача знання від великих, повільних і точних моделей до маленьких і швидких. З цією метою пропонується стиснути ансамбль моделей у єдину нейронну мережу. Спочатку набір класифікаційних моделей

навчається на певному анотований наборі даних. Очікується, що ансамбль буде більш стійким до переобладнання порівняно з окремою моделлю. Цей ансамбль використовується для маркування великої кількості синтетичних даних, згенерованих кількома простими процедурами випадкової вибірки, і, нарешті, одна модель вивчається на отриманому синтетичному наборі даних. Цей підхід може полегшити проблему переобладнання для нейронних мереж без часу та витрати пам'яті на побудову ансамблю. Слід, однак, зазначити, що цю роботу була виконано на невеликих наборах даних із повністю підключеними нейронними мережами та використано генерацію даних методів, які безпосередньо не застосовуються до зображень. З сучасними CNN життєздатність цього підходу обмежується наступними фактами: набори даних вже дуже великі, а моделі занадто великі для створення великих ансамблів.

Рисунок 2.5 – Зразки класу туалетного паперу для перевірки ILSVRC

Справа в тому, що учень тепер не тільки має доступ до виходів викладача, а й отримує інформацію з внутрішнього представлення даних, призводить до швидшої конвергенції та краща продуктивність методу. Підхід «викладач-учень» є потужним інструментом, який можна використовувати для навчання квантованих мереж. У сучасну епоху нейронних мереж більшість робіт зосереджуються на ситуації, коли обидва вчитель і учень – це нейронні мережі. Хоча логічно використовувати найкраще метод, доступний вчителю, потреба в швидшому учневі може призвести до різних видів

моделей. Таким чином, можливе використання м’якого дерева рішень як студентської моделв. Дерево рішень, теоретично, може забезпечити дуже високе прискорення.

Деякі зразки легко класифікувати, тоді як інші заплутані. Потужна модель здатна виробляти правильні мітки для зразків праворуч з надмірністю для зразків ліворуч (рисунок 2.5).

2.6 Адаптивні методи

Доводячи до межі ідею автоматичного налаштування архітектури, ми отримуємо модель, яка обробляє зразки з того самого набору даних по-різному. Така модель може адаптуватися до складності зразку, швидка обробка легких зразків і витрата більше часу на комплексних. Показано приклад діапазону складності в наборі даних ILSVRC на рисунку 2.5.

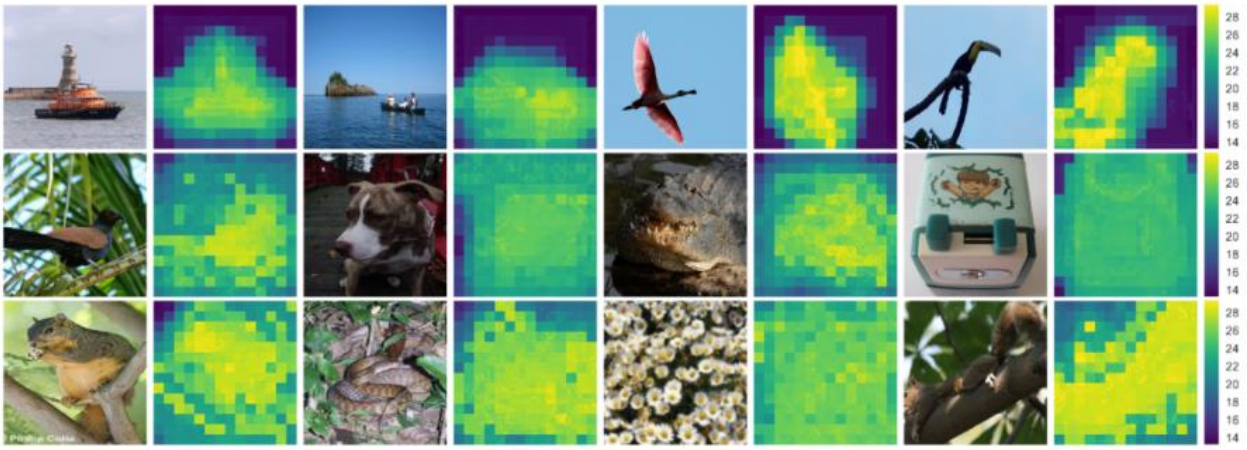

Рисунок 2.6 – Карти вартості для зображень із перевірки ILSVRC

Змінна складність є природньою для алгоритмів виявлення об’єктів із кроком генерації пропозицій: чим більший бюджет часу, тим більше пропозицій має бути згенеровано та оцінено. Для класифікації зображень та інших завдань, які вирішуються за допомогою єдиної мережі CNN структуру

потрібно змінити, щоб змінити час висновку. Природний спосіб досягти змінного часу логічного висновку – змінна глибина, яка може бути архітектурою типу ResNet якщо видалити деякі її блоки. Таким чином, у навчальному часі мережа має різну глибину протягом навчання та велику фіксовану глибину під час тестового етапу.

Архітектуру, яка адаптивно змінює глибину Adaptive Computation Time (ACT), було запропоновано для рекурентних нейронних мереж. Особлива гілка, додається до кожного шару, передбачає оцінку зупинки h, одне значення, що лежить у [0,1] діапазон. Ці оцінки зупинки можна інтерпретувати ймовірнісним або детермінованим способом. У першому випадку зупинка – це просто ймовірність зупинитися на поточному рівні. У другому випадку обчислення зупиняється, коли сума балів зупинки перевищує 1. Щоб уникнути необмеженого збільшення часу обчислення, спеціальний термін втрати, який називається вартістю обміркування, який штрафує за низький рівень введено зупинку балів. Карти розміру вартості демонструють, що надається SACT механізм уваги, як показано на рисунку 2.6. ACT було розширено для обробки зображень як просторово адаптивне обчислення в часі.

SACT застосовує механізм ACT для кожного просторового розташування, тобто зупинка обчислення оцінки та рішення про завершення виконується окремо для кожного місця. Це означає, що обчислення продовжуватимуться для деяких позицій, коли воно закінчено для інших. Такий процес можна реалізувати через перфоровану згортку Підходи, перераховані вище, регулюють глибину шляхом зупинки, що є логічним способом вирішення різниці між легкими та складними зразками однакової природи. В інших випадках може бути бажаною спеціалізація на вищих рівнях мережі.

Навчання SkipNets є складним завданням, оскільки пропускає деякі шари в середині потік даних за своєю суттю є дискретним рішенням, і спроби навчити його за допомогою деяких різновид м’яких наближень призводить

до суттєвого падіння точності під час тестування, коли жорсткі необхідно запровадити порогове обмеження. У результаті складний алгоритм навчання, який використовує м’які наближення, і потрібне навчання з підкріпленням на різних етапах.

2.7 Висновки до розділу 2

Деякі підходи, такі як методи на основі тензорного розкладання, здається, досягли насичення, оскільки було випробувано багато розкладів, і важко придумати нові ідеї. Проектування ефективних архітектур є, ймовірно, найбільш практичним підходом, оскільки це так і не вимагають складних багатоетапних процесів, які підлягають модифікації та тонкому налаштуванню етапів.

3 АЛГОРИТМ ДЕКОМПОЗИЦІЇ ВАГІВ У ЗНМ

Група робіт, розглянута в розділі 2, досягла значного прискорення CNN шляхом застосування різних тензорних розкладів до ваг і активацій згорткових шарів. У цьому розділі зосереджуємося на CP-розкладі низького рангу, і застосуванні його до ваг згорткового шару, щоб отримати послідовність з чотирьох нових згорткових шарів з малими фільтрами. Ці шари працюють швидше, мають менше параметрів і використовують лише існуючі бблоки CNN. Після апроксимації згорткового шару і замінени, можна з легкістю точно налаштувати всю мережу на використання навчальних даних зворотнього поширення. CNN, отримані за допомогою запропонованого методу, дуже точні, враховуючи, що кількість параметрів зменшено в кілька разів порівняно з початковою мережею. Практично це зменшення кількості параметрів означає більшу компактність мережі зі зменшеним обсягом пам’яті, що може бути важливим для архітектур з обмеженою оперативною пам'ятю.

Така економія пам'яті може бути особливо цінною для мережіпрямого зв'язку, які включають згортки з просторово змінними фільтрами (локально з'єднані шари). З теоретичної сторони ці результати підтверджують інтуїцію про те, що сучасні CNN надпараметризовані, тобто величезна кількість параметрів у сучасних CNN не є необхідною для зберігання інформації про задачу класифікації, але, радше, служить для полегшення збіжності до хороших локальних мінімумів функції втрат.

Використання розкладання низького рангу для прискорення згортки було запропоновано Пігамонті у контексті вивчення кодової книги, а потім застосування до CNN. Декомпозиція, запропонована Джедебергом (рисунок 3.1) ефективно апроксимує 4D масив ваг як композицію (продукт) двох 3D тензорів.

Після обчислення декомпозиції виконує локальне тонке налаштування

з втратою L2 на відхилення між повним і апроксимованим виходами згорток на даних навчання. На відміну від Джадеберга, запропонований метод тонко налаштовує всю мережу на основі вихідного дискримінаційного критерію. Таке дискримінаційне тонке налаштування було неефективним для попередньої схеми, але в нашому випадку це працює добре, навіть коли CP-розклад має велику похибку апроксимації. Нижче надається порівняння теоретичної та емпіричної складності порівняння нашої схеми з Джадебергом.

У роботі Дейтона, яка найбільше пов’язана з нашою, запропоновано схему на основі CP-декомпозиції частин згорткових ваг, отриманих шляхом бікластеризації (поряд із різними розкладаннями для першого згорткового шару та повністю з’єднаних шарів). Бікластеризація Дентона окремо розділяє два непросторові розміри на підгрупи та використовує отримане групування для скорочення тензора ваги на групу менших тензорів, щоб зменшити ефективні ранги в CP-розкладі.

CP-розклади згорткових ваг у Дентона були обчислені жадібно, збільшуючи ранг апроксимації на одиницю на кожному кроці без зміни наближення, отриманого на попередньому кроці.

Хоча розкладання дозволяє відокремлюючи всі чотири виміри тензора, Дентон відмовився від поділу просторових вимірів, зазначивши, що результуючий приріст був мінімальним.

3.1 CP-декомпозиція

Тензорні розкладання є природним способом узагальнення підходу низького рангу до багатовимірного випадку. Нагадаємо, що низькоранговий розклад матриці A розміром n×m з ранг R задається

$$A(i,j) = \sum_{r=1}^{R} A_1(i,r)A_2(j,r), \quad i = \overline{1,n}, \quad j = \overline{1,m}, \tag{3.1}$$

і приводить до ідеї поділу змінних. Найпростіший спосіб розділення змінних у випадку багатьох вимірів полягає у використанні канонічної поліадичної декомпозиції (CP decomposition, також звана моделлю CANDECOMP / PARAFAC). Це розкладання, вперше запропонований Хічкоком [1927], пізніше було повторно відкрите кілька разів. Для d-вимірного масиву A розміром $n_1 \times \cdots \times n_d$ а CP-декомпозиція має такий вигляд

$$A(i_1, \ldots, i_d) = \sum_{r=1}^{R} A_1(i_1, r) \ldots A_d(i_d, r) \quad , \qquad (3.2)$$

де мінімально можливе R називається канонічним рангом. Перевага цього розкладання полягає в тому, що замість цілого тензора потрібно зберігати лише $(n_1 + \cdots + n_d)$ R елементів з $n_1 ::: n_d$ елементами.

У двох вимірах апроксимація низького рангу може бути стабільно обчислена за допомогою розкладу за сингулярним значенням (SVD) або, якщо матриця велика, шляхом визначення рангу алгоритма. На жаль, це не так для CP-розкладу, коли $d > 2$, оскільки не існує кінцевого алгоритму для визначення канонічного рангу тензора.

Тому більшість алгоритмів апроксимують тензор з різними значеннями R, поки похибка апроксимації не стане достатньо малою. Це веде до суті до пошуку хорошого CP-розкладу низького рангу і потрібно використовувати певні прийоми. А можна знайти детальний огляд методів обчислення CP-розкладів низького рангу.

Використовувався пакет програм Tensorlab для розрахунку CP-розклад. У різноманітності доступних методів оптимізації було обрано метод нелінійних найменших квадратів (NLS). Цей метод мінімізує Фробеніус норму апроксимаційного залишку (для визначеного користувачем фіксованого R) за допомогою оптимізації Гаусса-Ньютона.

3.2 Апроксимація згорткових вагів

Узагальнена згортка, яка відображає вхідний масив U(·,·,·) розміром X×Y×C у вихідний масив V(·,·,·) розміром (X−d+1)×(Y−d+1)×N за допомогою наступного лінійного відображення:

$$V(x,y,k) = \sum_{i=1}^{d} \sum_{j=1}^{d} \sum_{c=1}^{C} W(i,j,c,k)\, U(x+i, y+j, c)\,, \tag{3.3}$$

Тут W(·,·,·) є 4D масивом згорткових ваг розміром d×d×C×N і перші два виміри відповідають просторовим вимірам, третій вимір відповідає різним вхідним каналам, четвертий вимір відповідає різним вихідні канали. Просторова ширина і висота фільтрів позначаються як d.

CP-розклад рангу R (3.2) 4D згорткових ваг має вигляд:

$$W(i,j,c,k) = \sum_{r=1}^{R} W^x(i,r)\, W^y(j,r)\, W^c(c,r)\, W^k(k,r)\,, \tag{3.4}$$

де $W^x(\cdot,\cdot)$, $W^y(\cdot,\cdot)$, $W^c(\cdot,\cdot)$, $W^k(\cdot,\cdot)$ чотири компоненти складу що представляють 2D тензори (матриці) розмірів d×R, d×R, C×R та N×R відповідно.

Підставляючи (3.4) у (3.3) і виконуючи перестановку та групування доданків дає такий вираз для наближеної оцінки згортки (3.3):

$$V(x,y,k) = \sum_{r=1}^{R} W^k(k,r) \left(\sum_{i=1}^{d} W^x(i,r) \left(\sum_{j=1}^{d} W^y(j,r) \left(\sum_{c=1}^{C} W^c(c,r) U(x+i, y+j, c) \right) \right) \right) \tag{3.5}$$

Базуючись на (3.5), вихідний масив V(·,·,·) може бути обчислений із входів U(·,·,·) через послідовність з чотирьох згорток з меншими фільтрами (рисунок 3.1):

$$U^c(x,y,r) = \sum_{c=1}^{C} W^c(c,r)\, U(x,y,c) \tag{3.6}$$

$$U^{cy}(x,y,r) = \sum_{j=1}^{d} W^y(j,r)\, U^s(i,y+j,r) \tag{3.7}$$

$$U^{cyx}(x,y,r) = \sum_{i=1}^{d} W^x(i,r)\, U^{cy}(x+i,y,r) \tag{3.8}$$

$$V(x,y,k) = \sum_{r=1}^{R} W^k(k,r)\, U^{cyx}(x,y,r)\,, \tag{3.9}$$

де $U^s(\cdot,\cdot,\cdot)$, $U^{cy}(\cdot,\cdot,\cdot)$ і $U^{cyx}(\cdot,\cdot,\cdot)$ є проміжними тензорами (стеками карт).

3.3 Опис експериментів

Для оцінки моделей проводиться кілька вимірювань. Після наближення згорткових вагів з CP-розкладом, точність цього розкладання розраховується, тобто ||W’ – W||/||W||, де W – початкові ваги, а W’ – отримане наближення. Різниця між вихідною вагою та наближеною може порушити розповсюдження даних у CNN, що призведе до зниження точності класифікації. Це падіння вимірюється до і після тонкого налаштування CNN. Крім того, CPU таймінг фіксується для моделей і прискорення порівняно з

таймінгом CPU повідомляється про оригінальну модель (усі таймінг базуються на коді Caffe, що виконується в CPU для пакетів зображень розміром 64). Нарешті, скорочення кількості параметрів обчислюється в результаті апроксимації низького рангу. Усі результати повідомляються для кількості рангів R.

Для експериментів з меншою мережею обрано класифікацію символів CNN як прийнятну базову лінію. Мережа має чотири згорткові шари з максимальними нелінійностями між ними та м'яким максимальним виходом. Це був класифікатор (вже пройшовший навчання) фрагментів зображень розміром 24 × 24 в один із 36 класів (10 цифр плюс 26 символи).

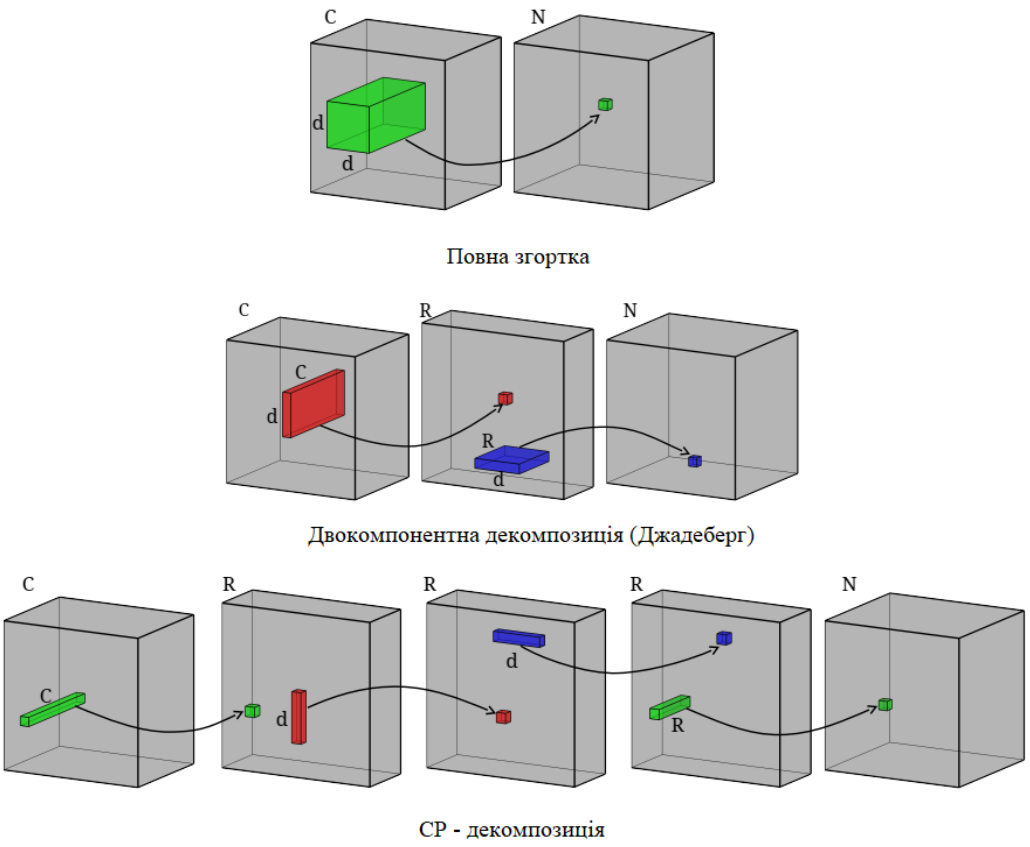

Рисунок 3.1 – Тензорні розкладання для прискорення узагальненої згортки

Наше Caffe портує загальнодоступну попередньо навчену модель (наведену нижче як CharNet) та досягає 91,2% точності на тестовому наборі

(дуже схожий на оригінал). Розглядається другий і третій згортковий шар. Ці шари складають більше 90% часу обробки. Рівень 2 має 48 входів і 128 вихідних каналів і фільтрів розміром 9×9, шар 3 має 64 вхідних і 512 вихідних каналів, розмір фільтра 8×8.

3.4 Результати експериментів

Показано результати окремої апроксимації шарів 2 і 3 на рисунку 3.2 Похибка тензорної апроксимації зменшується зі зростанням рангу апроксимації, і коли ранг апроксимації стає достатньо великим, тоді можна точно визначити вагу. Однак наші експерименти показують, що для належного функціонування мережі не потрібна точна апроксимація, особливо після тонкого налаштування. Наприклад, при апроксимації рівня 3 точність класифікації мережі неефективна, навіть якщо похибка апроксимації становить 78%.

Оскільки декомпозиція значно зменшує кількість параметрів, її також можна розглядати як потужну техніку регуляризації, особливо корисну для надпараметризованих моделей, таких як CharNet. Мережа з декомпозованим шаром менше перебирається і може досягти кращої точності на тестовому наборі після деякого тонкого налаштування.

Розкладання можна застосовувати до кількох шарів нейронної мережі, що підтверджується наступним експериментом. По-перше, був шар 2 апроксимовано з рангом 64. Після цього падіння точності було зменшено шляхом точного налаштування всіх шарів, крім нових. Нарешті, шар 3 був апроксимований рангом 64, і для цього рівня така апроксимація не призводить до значного падіння мережі точності передбачення, тому немає необхідності зайв,ого тонкого налаштування мережі. Мережа, отримана цією процедурою, у 8:5 разів швидша, ніж вихідна модель, тоді як точність класифікації падає на 1% до 90,2%. Порівнюючи з дослідженнями Джадеберга, CP-декомпозиція досягає майже вдвічі більшого прискорення за

тих же втрат точності (1% втрати точності для прискорення 4,2× та 5% втрати точності для прискорення 6×).

Щоб довести життєздатність розглянутого методу для більших мереж і складніших завдань, ми переходимо до класифікації зображень на наборі даних ImageNet. Слідуючи Дентону, перші експерименти з архітектурою AlexNet виконуються на другому згортковому шарі (розмір фільтра 5×5, 96 вхідних і 256 вихідних каналів) і попередньо навчена модель поставляється з Caffe, використовується як базова лінія. Різні властивості мережі для кількох різних рангі наближення показані на рисунку 3.2. Можна помітити, що conv2 розглянутої мережі вимагає набагато більшого рангу (порівняно з експериментом CharNet) для досягнення належної продуктивності.

Загалом, щоб досягти точності 0,5% падіння достатньо взяти 200 компонентів, які також дають чудове прискорення шару (3,6× проти 2×), досягнуте схемою 2. Тривалість роботи conv2 може бути додатково скорочена ціною невеликого збільшення частоти неправильної класифікації: апроксимація рангу 140 призводить до прискорення 4:5× ціною ≈ 1% втрати точності, що перевищує результати Дентона. Поряд зі звичайним тонким налаштуванням повної мережі ми спробували уточнити отриманий тензор апроксимації шляхом застосування підходу реконструкції даних з Джадеберга. На жаль, нам не вдалося знайти хороший рівень навчання SGD: більші значення призвели до вибухових градієнтів, тоді як менші не дозволяли суттєво зменшити втрати від реконструкції. Ймовірною причиною цього ефекту є нестабільність низького рангу CP-декомпозиції. Одним із способів обійти проблему буде чергування навчання компонентів (тобто не оптимізувати їх усі одночасно).

Інший спосіб боротьби з нестабільністю випливає з спостереження, що це не викликано збільшенням глибини CNN (як доведено нещодавнім прогресом у навчанні надзвичайно глибоких мереж), але шляхом укладання згорткових шарів без нелінійності між ними. Введення нелінійностей змінює

структуру декомпозиції, наведення існуючих методів обчислення CP-декомпозиції низького рангу марне.

Сірі коробки відповідають тривимірним тензорам (стекам карт) у CNN, з двома фронтальними сторонами, що відповідають просторовим розмірам. Стрілки показують лінійні відображення та демонструють скалярні значення праворуч (маленькі прямокутники, що відповідають окремим елементам цільового тензору).

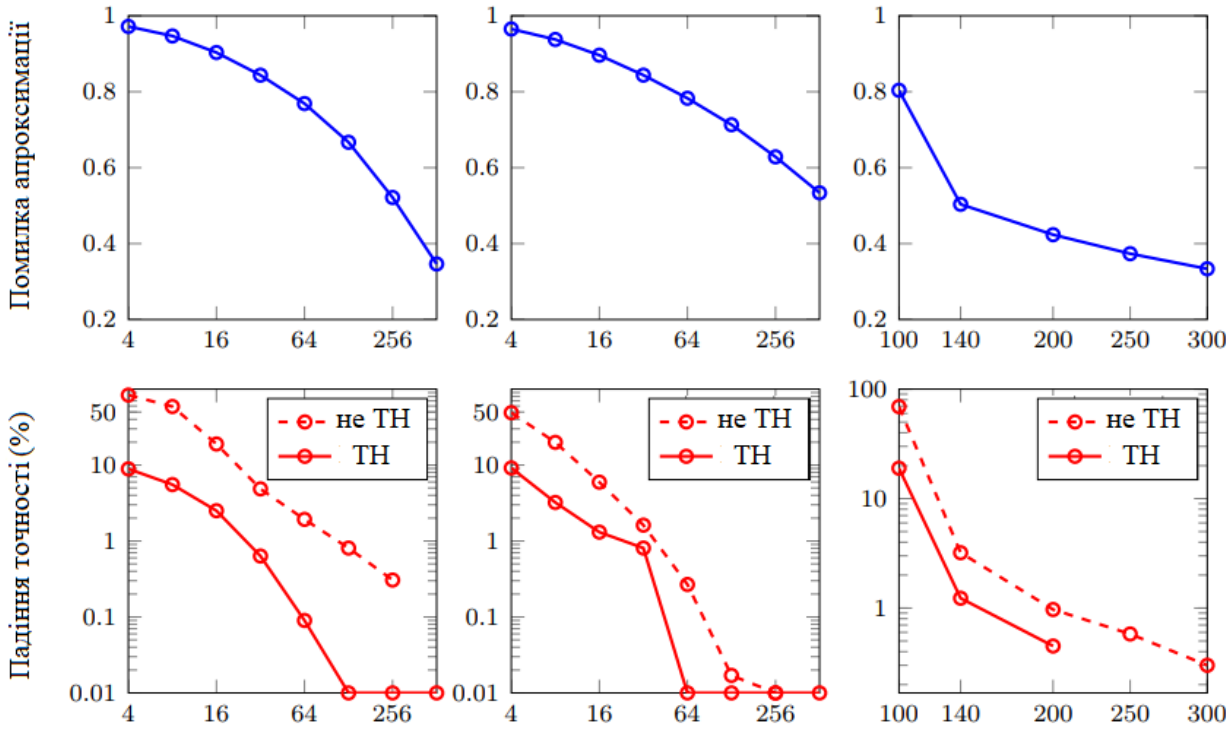

Рисунок 3.2 – Різні властивості та метрики ефективності різних апроксимованих CNN, побудованих як функції рангу апроксимації. Помилка апроксимації вагового масиву та падіння точності класифікації повної моделі з апроксимованими шарами

Початкова повна згортка обчислює кожен елемент вихідного масиву як лінійну комбінацію елементів тривимірного зрізу, що охоплює просторове вікно d×d над усіма вхідними картами. Алгоритм Джадеберга наближено до початкової згортки як композицію двох лінійних відображень із проміжним

стеком карт та мають R карт (де R – ранг розкладання). Кожне з двох відображень обчислює кожне цільове значення на основі просторового вікна розміром 1×d або d×1 загалом вхідної карти. Нарешті, CP-декомпозиція апроксимує згортку як композицію чотирьох згорток з малими фільтрами, щоб цільове значення обчислювалося на основі 1D-масив, який охоплює або один піксель у всіх вхідних картах, або 1D просторове вікно в одній вхідній карті.

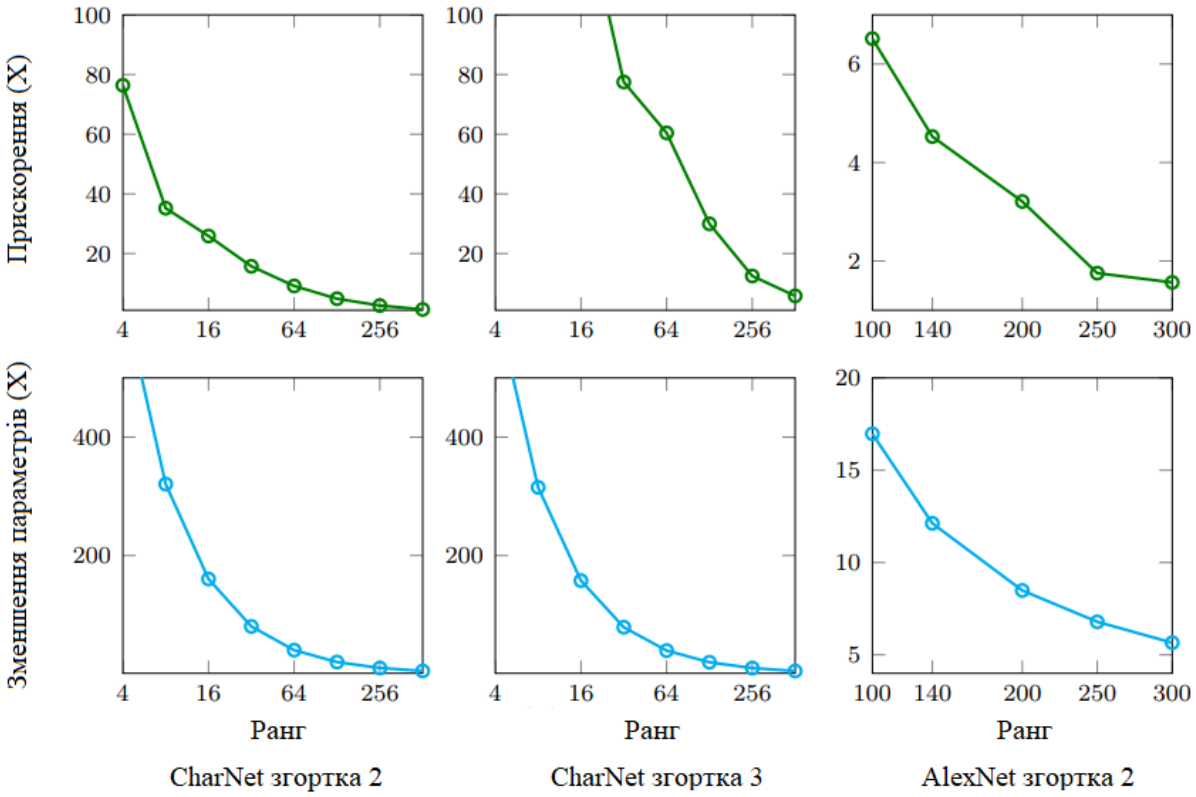

Рисунок 3.3 – Різні властивості та метрики ефективності різних апроксимованих CNN, побудованих як функції рангу апроксимації. Емпіричне прискорення апроксимованого шару. Співвідношення між числами параметрів в вихідному і апроксимованому шарах

Декомпозиція NLS значно вищої точності з тонким налаштуванням або без нього. Перевага є більше для складнішої мережі (AlexNet). По-друге, вихід тонкого налаштування явно залежить від якості апроксимації. Це

спостереження узгоджується з гіпотезою про те, що велика кількість параметрів у CNN необхідна, щоб уникнути поганого локального мінімума. Дійсно, CP-розклад радикально зменшує кількість параметрів в шарі.

3.5 Висновки до розділу 3

Досить просте застосування методу тензорної декомпозиції (CP-декомпозиція низького рангу на основі NLS) у поєднанні з дискримінаційним тонким налаштуванням усієї мережі може досягти значного прискорення з мінімальними втратами точності. Однак тонке налаштування розкладеної моделі ускладнюється нестабільністю розкладання. Ідеальний підхід для прискорення роботи нейронної мережі дозволив би одночасно пришвидшити стільки шарів, скільки потрібно, і точно налаштувати всю мережу з легкістю оригінального тренувального процесу.

ВИСНОВКИ

Проведено аналіз методів та алгоритмів прискорення згорткових нейронних мереж. Вони поділяються на кілька груп: тензорні розклади, квантування, скорочення, навчання з вчителем, пошук ефективних архітектур та адаптивних моделей. Детально розглянуто алгоритми прискорення згортковиху нейронних мережах з низькоранговим CP-розкладом згорткових ваг. Реалізація методів базується на існуючих блоках ЗНМ, що дозволяє легко розгортати, і найголовніше, точніше налаштовувати моделі, хоча нестабільність CP-декомпозиції ускладнює процес тонкого налаштування. Експериментальні результати демонструють значне прискорення з мінімальним падінням точності для кількох архітектур.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Hande Alemdar, Vincent Leroy, Adrien Prost-Boucle, and Fr´ed´eric P´etrot. Ternary neural networks for resource-efficient AI applications. In International Joint Conference on Neural Networks, 2017.

2. Genevera Allen. Sparse higher-order principal components analysis. In International Conference on Artificial Intelligence and Statistics (AISTATS), 2012.

3. Sajid Anwar, Kyuyeon Hwang, and Wonyong Sung. Fixed point optimization of depth convolutional neural networks for object recognition. In Acoustics, Speech, and Signal Processing (ICASSP), International Conference on, 2015.

4. M. Astrid and Seung-Ik Lee. Cp-decomposition with tensor power method for convolutional neural networks compression. In Big Data and Smart Computing, 2017.

5. Hossein Azizpour, Ali Sharif Razavian, Josephine Sullivan, Atsuto Maki, and Stefan Carlsson. From generic to specific deep representations for visual recognition. In Conference on Computer Vision and Pattern Recognition (CVPR), 2015.

6. Artem Babenko and Victor Lempitsky. Aggregating local deep features for image retrieval. In International Conference on Computer Vision (ICCV), 2015.

7. Hessam Bagherinezhad, Mohammad Rastegari, and Ali Farhadi. LCNN: Lookup-based convolutional neural network. Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

8. Дяченко В.О., Міхаль О.П., Пономаренко Р.Д. Алгоритми підвищення ефективності згорткових нейронних мереж// Проблеми інформатизації : десята міжнародна науково-технічна конференція. Черкаси – Баку – Бельсько-Бяла – Харків, 2022, т.2, с.95.