УДК 004.048 DOi 10.30837/ bi.2020.1(94).13

Н.С. Кравець[1], А.В. Ховрат[2], Н.С. Сайчишина[3]

[1]Кандидат технічних наук, доцент кафедри Програмної інженерії,
Харківський національний університет радіоелектроніки,
natalia.kravets@nure.ua, ORCID iD: 0000-0002-6753-3333

[2]Студент кафедри Програмної інженерії,
Харківський національний університет радіоелектроніки, artem.khovrat@nure.ua

[3] Студент кафедри Програмної інженерії,
Харківський національний університет радіоелектроніки, nataliia.saichyshyna@nure.ua

АНАЛІЗ ЗАСТОСУВАННЯ ТЕНЗОРНИХ ПРОЦЕСОРІВ В ЗАДАЧАХ МАШИННОГО НАВЧАННЯ НА ПРИКЛАДІ GOOGLE TPU

Проведено детальний аналіз тензорного процесору від компанії Google, розглянуто його математичне підґрунтя, структурні складові та ключові стадії роботи для використання при розв'язанні задач пов'язаних з машинним навчанням. Розглянуто методи прискорення процесу тренування нейронної мережі без втрати якості, реалізовані в TPU: квантування, паралельна обробка, систолічний масив, механізм інкапсуляції обчислень в нейронних мережах. Здійснений розбір обмежень та переваг цього виду процесору загалом та у порівнянні з графічним та центральним процесорами. Розглянуто конкурентні переваги даного тензорного процесору з аналогами, що пропонуються іншими компаніями. Описана взаємодія із хмарною платформою Google та з програмною бібліотекою TensorFlow.

АРХІТЕКТУРА ПРОЦЕСОРА, МАШИННЕ НАВЧАННЯ, ТЕНЗОР, ТЕНЗОРНИЙ ПРОЦЕСОР, ХМАРНІ ТЕХНОЛОГІЇ, ЦЕНТРАЛЬНИЙ ПРОЦЕСОР, ГРАФІЧНИЙ ПРОЦЕСОР, ПАРАЛЕЛЬНІ ОБЧИСЛЕННЯ

Проведен детальный анализ тензорного процессору от компании Google, рассмотрен его математический базис, структурные составляющие и ключевые стадии работы для использования при решении задач, связанных с машинным обучением. Рассмотрены методы ускорения процесса тренировки нейронной сети без потери качества, реализованные в TPU: квантование, параллельная обработка, систолический массив, механизм инкапсуляции вычислений в нейронных сетях. Проведен разбор ограничений и преимуществ этого вида процессору в целом и в сравнении с графическим и центральным процессорами. Рассмотрены конкурентные преимущества данного тензорного процессору с аналогами, предлагаемыми другими компаниями. Описано взаимодействие с облачной платформой Google и с программной библиотекой TensorFlow.

АРХИТЕКТУРА ПРОЦЕССОРА, МАШИННОЕ ОБУЧЕНИЕ, ТЕНЗОР, ТЕНЗОРНЫЙ ПРОЦЕССОР,ОБЛАЧНЫЕ ТЕХНОЛОГИИ, ЦЕНТРАЛЬНЫЙ ПРОЦЕССОР, ГРАФИЧЕСКИЙ ПРОЦЕССОР, ПАРАЛЛЕЛЬНЫЕ ВЫЧИСЛЕНИЯ

A detailed analysis of the tensor processor by Google was carried out, its mathematical basis, structural components, and key work stages for use in solving problems related to machine learning were considered. Methods for speeding up the neural network training process without loss of quality implemented in TPU are considered: quantization, parallel processing, systolic array, and the mechanism for encapsulating calculations in neural networks. The analysis of the limitations and advantages of this type of the processor as a whole and in comparison, with the graphics and central processors. The competitive advantages of this tensor processor with analogues offered by other companies are considered. The interaction with the Google cloud platform and with the TensorFlow software library is described.

ARCHITECTURE OF THE PROCESSOR, MACHINE LEARNING, TENSOR, TENSOR PROCESSOR, CLOUD TECHNOLOGIES, CENTRAL PROCESSOR, GRAPHICS PROCESSOR, PARALLEL COMPUTATION

Вступ

Різновид машинного навчання, відомий як глибоке навчання, використовується Google для обробки пошукових запитів на природній мові та розпізнавання зображень. Реалізація моделей глибинного навчання збільшує точність розпізнавання об'єктів, але потребує обробки дуже великих наборів даних зі складною структурою, що у свою чергу вимагає нових алгоритмів та паралельного обчислювального обладнання. Дослідники з Google випустили свою власну платформу глибокого навчання TensorFlow в якості програмного забезпечення з відкритим кодом.

У 2017 році компанія Google анонсувала TPU (Tensor Processing Unit) [13, 14], тензорний процесор, — кастомний пристрій, що є спрямованою інтегрованою мікросхемою, сотвореною спеціально для машинного навчання. Рік потому TPU були розміщені на хмарній платформі GoogleCloud та зроблені публічними для не комерційного використання. Як і подібні мікросхеми від NVIDIA чи Intel, наприклад CPU[1], центральний процесор, та GPU[2], графічний процесор, цей засіб обробки тензорів застосовується для збільшення швидкості виконання високонавантажених операцій у машинному навчанні, зокрема для апаратного прискорення глибинних нейромереж.

Окрім цього TPU спроектований спеціально для використання із фреймворком машинного навчання

від Google TensorFlow[3], що збільшує його ефективність. У цій статті здійснений розгляд, пропонованої технології як загалом так і у порівнянні з CPU, GPU та аналогами TPU від інших компаній

1. Використання тензорів у машинному навчанні

Тензор — об'єкт лінійної алгебри, лінійно перетворюючий елемент одного лінійного простору в елементи другого. Окремим випадками тензорів є скаляри, вектори, білінійні форми і т. п.[4]

Часто тензори представляють багатовимірну таблицю $d \times d \times \ldots \times d$ заповнену числами — компонентами тензора (де d — розмірність векторного простору, над яким задано тензор, кількість множників співпадає з так званою валентністю чи рангом тензора).

У комп'ютерних науках тензор — це багатовимірна матриця аналогічна NumPy масиву — фундаментальній структурі даних, яка використовується в алгоритмах машинного навчання. Це основна одиниця операцій у фреймворку TensorFlow, який використовує NumPy.

TensorFlow — фреймворк представлений компанією Google в 2015 році [15], який призначений для проектування, створення і вивчення нейромережевих моделей. Він використовується для того, щоб здійснювати обчислення, реалізовані за допомогою графів потоків даних. У цих графах вершини представляють собою математичні операції, в той час як ребра — є даними, які зазвичай подаються у вигляді багатовимірних масивів або тензорів.

Бібліотеки TensorFlow помітно спрощують вбудовування в додатки елементів, які самонавчаються, та функцій штучного інтелекту, призначених для організації роботи комп'ютерного зору, обробки природної мови тощо.

Звичайно, TensorFlow не єдина бібліотека машинного навчання, але, як і пошуковий механізм Google, вона вважається кращою в своєму класі. Альтернативами є програмне забезпечення Torch, створене швейцарськими дослідниками, а також Caffe, розроблена Каліфорнійським університетом у Берклі.

NumPy — це модуль з відкритим кодом для мови програмування Python[5], який надає загальні математичні та числові операції у вигляді попередньо скомпільованих швидких функцій. Вони об'єднуються в високорівневі пакети, що забезпечує функціонал, який можна порівняти з функціоналом MatLab. Причиною його використання у TensorFlow є надання швидких методів для маніпуляції великими масивами та матрицями.

Алгоритми машинного навчання (інакше нейромережевий алгоритм) частіше за все використовують операції додавання та множення наступних об'єктів:

– скаляри;
– вектори;
– матриці.

Для прикладу розглянемо основні етапи роботи абстрактного алгоритму машинного навчання: початковий, класифікація та завершення.

На початковому етапі усі вхідні дані нормуються, наприклад, якщо це зображення, то відбувається уніфікація розмірів, рівня насиченості, тощо.

На основі цих даних ключова стадія будь-якого алгоритму — класифікація. Власне вирішення цієї задачі і пов'язане з тензорами.

Для прикладу приведемо алгоритм класифікації Метод опорних векторів, інакше SVM [6], який наразі часто використовується.

Виходячи з того, що об'єкт, який знаходиться в багатовимірному просторі, відноситься до одного з двох класів, алгоритм будує гіперплощину, що має мірність на 1 меншу, таким чином щоб усі об'єкти опинилися в одній з двох груп. Окрім цієї сепарації, гіперплощина має бути максимально віддаленою від найближчого об'єкта кожної групи. Для цього алгоритм шукає точки на графіку, які розташовані найближче до лінії поділу. Ці точки називаються опорними векторами. Потім, алгоритм обчислює відстань між опорними векторами і розділяє їх площиною. Ця відстань називається зазором. Основна мета алгоритму — максимізувати міру зазору.

Після подібної класифікації вхідних об'єктів на основі отриманого результату формуються прогнозні значення, яких не вистачає для вирішення поставленої задачі — це і є завершальний етап, на якому тренується нейромережа. Він також пов'язаний з тензорною алгеброю.

Для того щоб його пройти кожен нейрон нейромережі здійснює наступні розрахунки:

– вхідні дані x множаться з вагами w, задля обчислення сили сигналу;

– цей результат додається, задля агрегації стану нейрону;

– після цього застосовується активаційна функція $f()$, задля того, щоб спровокувати активність штучного нейрону.

Власне схему входів та нейронів зображено на рис. 1.

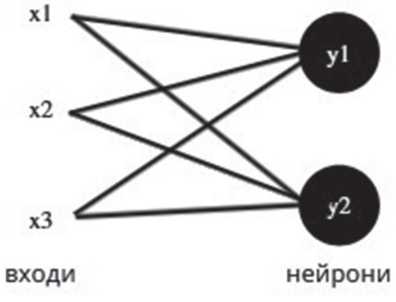

Рис. 1. Схема входів та нейронів

Як можна бачити на Рис. 1, з трьома входами та двома нейронами в повнозв'язній однорівневій нейронній мережі, необхідно виконати 6 операцій множення між вагами та входами, об'єднав множники у дві групи по 3 за допомогою засобів додавання.

Ця послідовність множення та додавання може бути записана у вигляді матричного множення, що з обчислювального боку є найбільш навантаженою частиною даного етапу.

Кожне передбачення вимагає багато кроків множення входів матриці ваги та застосування активаційної функції.

У результаті, множення та зрізи масивів даних або створюють велике навантаження на процесор або вимагають значних обсягів пам'яті для реалізації.

TPU був спроектований таким чином, щоб знизити навантаження на процесор, не вимагаючи значних обсягів додаткової пам'яті.

2. Логіка TPU

Проблему прискорення процесу тренування нейронної мережі без втрат якості TPU вирішує декількома методами [7]:

1. Квантування

Разом з технікою квантування відбувається процес апроксимації випадкового значення між заданим мінімумом та максимумом у 8-бітному вигляді, TPUмістить 65,536 8-бітних множників. У своїй суті ця техніка — це стискання 32-бітних чисел з плаваючою точкою чи 16-бітних у 8-бітний вигляд. Квантування — це перший засіб, який використовує TPU для зниження витрат при виконанні передбачення за допомогою нейромереж, без значних втрат у швидкості.

2. Увага до обчислень

Архітектура TPU інкапсулює сутність обчислення у нейронних мережах, за допомогою наступних обчислювальних ресурсів:

Matrix Multiplier Unit (MMU): 65,536 8-бітних процесорів множення та додавання для матричних операцій;

Unified Buffer (UB): 24 мегабіти статичної оперативної пам'яті з довільним доступом, яка працює в якості регістрів;

Activation Unit (AU): провідні функції активації.

Усе це керується високорівневими інструкціями, які відповідають за основні математичні операції, що потрібні для роботи нейронних мереж. Спеціальний компілятор та програмний стек транслює запити API з графу Tensor Flow у інструкції до TPU.

3. Паралельна обробка

Типові процесори зі скороченим набором команд (RISC-процесори) надають інструкції для простих обчислень таких як множення, обробкою одиничної та скалярної операції кожної інструкції. Як вже було сказано, TPU містить MMU, що спроектований як матричний, а не скалярний процесор. Це дозволяє обробляти сотні тисяч операції за один тік. Існує багато яскравих ілюстрацій цього процесу, наприклад, друк документа у цілому, а не по рядку.

4. Систолічний масив

Основою матричного процесору MMUє систолічний масив (systolic array). У традиційній архітектурі (такій як CPU чи GPU), значення зберігаються у регістрах, програма вказує арифметико-логічному пристрою на ті регістри, які необхідно зчитати, які виконати операції, та у які регістри розмістити результат. Фактично, програма є послідовністю подібних операції.

В MMU перемноження матриць використовує вхідні дані декілька разів для того, щоб отримати вихідний результат. Інакше кажучи, значення зчитуються один раз, але використовуються декілька разів для різних операцій, при цьому без зберігання в регістри. Тобто арифметико-логічний прилад виконує багато операцій множення та додавання, однак за фіксованими шаблонами.

3. TPU як служба хмарної платформи Google

Одним з найважливіших характеристик TPU від Googleє те, що компанія зробила його доступним в якості служби на своїй хмарній платформі. Як результат, даний TPU дозволяє запускати робочі навантаження на прискорювачі Google TPU з використанням Tensor Flow. Згодом це навантаження розподіляється на хмарний сервер, так як це показано на рис.2.

Як зазначають автори: хмарний TPU має на меті допомогти дослідникам та розробникам у створенні обчислювальних кластерів Tensor Flow.

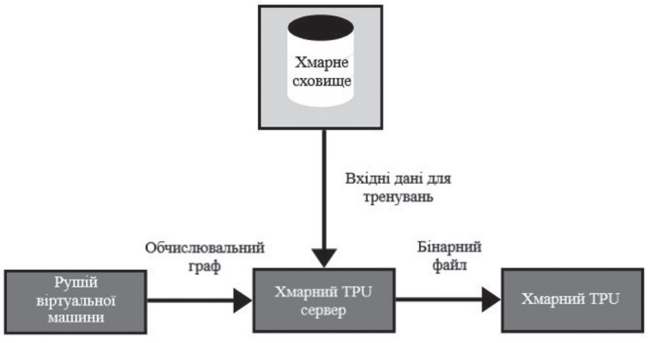

Рис. 2. Розподіл навантаження на сервері

Що стосується самого процесу програмування моделі, то тут варто відмітити, що передача даних між хмарним TPU та пам'яттю хоста відбувається повільніше у порівнянні зі швидкістю обчислень (це наслідок використання PCIe шини). Як наслідок можна спостерігати простоювання TPU. Розв'язання цієї проблеми передбачає наступні особливості:

– усі параметри моделі зберігаються у вбудованій пам'яті з високою пропускною здатністю;

– багато етапів навчання моделі виконуються в циклі, амортизуючи вартість запуску обчислень у хмарному TPU;

– Tensor Flow витягує та попередньо обробляє дані перед передачею на обладнання хмарного TPU;

– ядра хмарного TPU синхронно виконують ідентичну програму, яка зберігається у їх власному відповідному слоті пам'яті з високою пропускною здатністю (HBM).

4. Переваги та обмеження TPU

Використання TPU від Google надає ряд переваг з точки зору підвищення ефективності та швидкості обчислень, включаючи наступні [8]:

– покращення продуктивності лінійно алгебраїчних обчислень, які активно використовуються у машинному навчанні;

– мінімізація часу навчання за фіксованою точності результатів при навчання великих складних моделей нейронних мереж;

– збільшення швидкості отримання вихідних даних за рахунок масштабування операцій на хмарних серверах із TPU.

Варто зазначити, що TPU були спеціально оптимізовані для швидкого виконання великої кількості матричних множень. Відповідно, у ситуаціях, коли матричне множення не є превалюючим, використання TPU не дає бажаного результату. Наприклад, це наступні ситуації [9]:

– програми лінійної алгебри з великою кількістю розгалужень;

– програми, які не часто звертаються до пам'яті;

– навантаження, які вимагають високоточних арифметичних обчислень, зокрема операції з числами з плаваючою точкою;

– навантаження, що містять користувацькі операції Tensor Flow, написані на C++.

5. Обмеження CPU та GPU при роботі з нейронними мережами

Для того аби проаналізувати важливість нового процесору для машинного навчання, розглянемо інші види процесорів, про які згадувалося раніше — центральний (CPU) та графічний (GPU).

1. Центральний процесор

CPU — процесор загального призначення, побудований на архітектурі фон Неймана (зображений на Рис.3)[10, 16].

З цього визначення можна зробити висновок, щодо головної переваги процесору — гнучкості. Завдяки якій є можливість звантажувати різноманітне програмне забезпечення, що вирішує велику кількість завдань користувачів: обробка текстів, класифікація зображень за допомогою нейронних мереж, проведення банківських транзакцій тощо.

Однак, в наслідок подібної гнучкості, порядок операцій не є заздалегідь визначеним, отже існують накладні витрати пов'язані з операцією зчитування даних з програмного забезпечення. З цим також пов'язана необхідність зберігати усі обчислення в регістрах процесору (або у кеші першого рівня), що лише поглиблює недолік архітектури у розрізі роботи алгоритмів машинного навчання. Оскільки у них більшість операцій є заздалегідь визначеними та повторюваними.

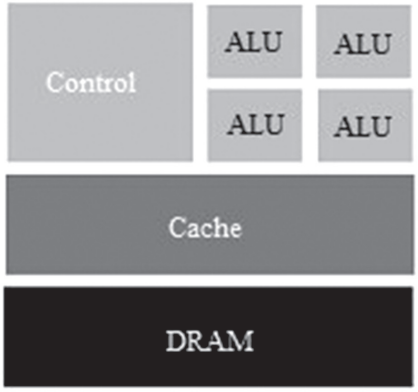

Рис. 3. Архітектура CPU

2. Графічний процесор

Головною відмінністю графічного процесору [11] є те, що за подібної архітектури (Рис. 4), використовується близько 2,500-5,000 арифметико-логічних пристроїв — це дозволяє виконувати тисячі операцій множення та додавання одночасно, надаючи можливість досягнути високого рівня паралелізму, а відповідно, продуктивності.

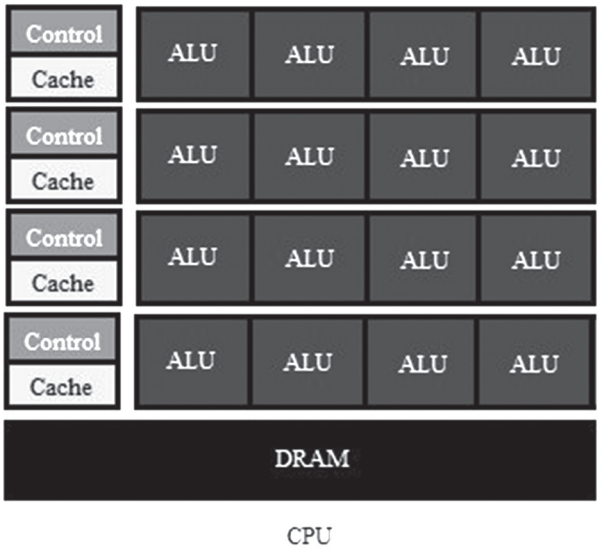

Рис. 4. Архітектура GPU

Зазначена вище особливість зумовила широке використання графічних процесорів при роботі з нейронними мережами. Однак у подібної архітектури

є значний недолік. Оскільки GPU виконує велику кількість паралельних обчислень, то він також використовує пропорційно більше енергії на доступ до пам'яті. Окрім цього, як і CPU цей процесор не є спеціалізованим для роботи з нейромережами, не зважаючи на загальну популярність. І хоча деякі серії обох видів процесорів підтримують роботу з ключовими фреймворками для машинного навчання, вони поступаються у швидкості та точності TPU.

6. Порівняння TPU з CPU та GPU

Розглянувши сутність CPU та GPU, здійснимо порівняння з Google TPU 3-го покоління, на прикладі IntelSkylake CPU [1] та NVIDIA's V100 GPU [2]. Для цього були обрані наступні критерії: пам'ять (обсяг та тип), максимальна продуктивність, максимальне енергоспоживання, шина даних та пропускна здатність при навчанні кумулятивної нейронної мережі.

1. Пам'ять

У випадку центрального процесору від Intel наявна пам'ять обсягом 120 GB типу DDR4SDRAM (синхронний динамічний оперативно запам'ятовуючий пристрій четвертого покоління з подвійною швидкістю передачі даних). Однак ця пам'ять є не настільки швидкою як у випадку GPU та TPU, що містять пам'ять з високою пропускною здатністю (HBM). Для першого — 16 гігабайт, для другого — 32. Подібний стан речей надає тензорному процесору від Google конкурентні переваги.

2. Максимальна продуктивність

Процесор від Intel досягає максимально продуктивності у 32 терафлопси одиничної точності, тоді як NVIDIA's V100 GPU 125 терафлопсів. Як бачимо, графічний процесор все ж потужніший за центральний, однак у випадку Google TPU максимальна продуктивність знаходиться на рівні 180 терафлопсів, що вказує на більшу ефективність тензорного процесору.

3. Максимальне енергоспоживання

У енергоспоживанні TPU також має конкурентні переваги — споживання на рівні 200 ваттів, тоді як у графічного процесору — більше 300 ваттів, а у центрального— 250. Ґрунтуючись на чинному та попередньому критерії, необхідно зауважити, що тензорний процесор має найбільший показник продуктивності на одиницю енергії — 0.9 терафлопсів на ватт. У той же час у CPU — 0.128, а у GPU — 0.42.

4. Пропускна здатність

За цим критерієм тензорний процесор поступається графічному — 128 гігабіт за секунду для TPU та близько 300 гігагбіт за секунду для GPU. Стосовно Skylake, то пропускна здатність цих процесорів досягає 60 гігабіт за секунду.

5. Шина даних

У кожному процесорі застосовується PCIe-шина 3-го покоління, тож чинний критерій не впливає на пропускну здатність.

Таким чином TPU від Google чинного покоління має переваги майже за усіма критеріями, окрім пропускної здатності, що і зумовлює обмеження названі у пункті 7.

7. Конкуренти Google TPU

Зважаючи на високу ефективність тензорних процесорів при вирішенні задач машинного навчання, велика кількість компаній займалася розробкою власних моделей TPU. Найвідомішою з них можна вважати розробку китайської компанії Huawei Ascend 910 [12], яку ті представили в 2019 році. Ascend 910 — це однокристальний процесор з широкими можливостями інтеграції та продуктивністю до 256 терафлопсів, я той час як процесор від Google досягає лише 180 терафлопсів. Крім ядер, які орієнтовані на вирішення задач машинного навчання, процесор містить у собі і звичайний центральний процесор, і планувальник задач з технологією DVPP — віртуальною динамічною системою електропостачання. Це дозволяє процесору самостійно керувати власною роботою, аби максимально використовувати обчислювальну потужність.

Зазначений процесор має працювати з новим фреймворком MindSpore від компанії Huawei, що вийшов у 1-му кварталі 2020 року та поширюється як проект з відкритим кодом в рамках стратегії компанії, спрямованої на розширення впровадження машинного навчання та допомогу розробникам.

Задля порівняння процесорів від Google та Huawei виокремимо кілька критеріїв, виключаючи вже названу продуктивність. До них можна віднести: максимальне енергоспоживання, пам'ять, використовувана шина даних, пропускна здатність цієї шини:

1. Максимальне енергоспоживання

TPU від компанії Google споживає значно менше електроенергії 200 ватт проти 310 ваттів у Huawei, що є певною перевагою для Google. Однак, з огляду на наявність DVPP ця перевага нівелюється.

2. Пам'ять

В обох випадках використовується пам'ять з високою пропускною здатністю (HBM) обсягом 32 гігабайти, однак у Huawei вона працює швидше, оскільки містить подвійний чіп на додаткові 16 гігабайт.

3. Шина даних

Зазначена раніше проблема Google TPU, де використовуються PCIe-шина 3-го покоління, вирішується у процесорі від Huawei, оскільки ті працюють з 4-им, більш швидким, поколінням PCIe-шини.

4. Пропускна здатність

У цій категорії Google TPU також поступається, оскільки має пропускну здатність 128 гігабіт у секунду, в той час як Ascend 910 — 240 гігабіт у секунду.

Як бачимо, тензорний процесор від компанії Google нинішнього покоління є слабшим майже за усіма показниками, проте його реліз відбувся

в середині 2018 року, на противагу кінцю 2019 року для процесору від Huawei. Також Huawei на момент написання чинною статті, не здійснили розміщення власної розробки на хмарній платформі, що не дає можливості для використання у корпоративних цілях, показуючи більшу цінову конкурентоздатність Google TPU.

Висновки

В статті здійснений огляд тензорного процесору від компанії Google, його особливості та переваги при навчанні нейронних мереж за фіксованої точності результатів. Здійснено порівняння технічних характеристик TPU з GPU від NVIDIA та CPU від Intel на прикладі роботи з кумулятивною нейромережею, що показало значну перевагу розробки Google над іншими видами процесорів, зокрема, у одному з найважливіших показників, продуктивності на одиницю енергії. Також зроблено порівняння з TPU від Huawei, виходячи з анонсованих китайською компанією характеристик, які вказують на те, що тензорний процесор від Google поступається конкуренту у потужності, однак надання можливості користування TPU на хмарній платформі, нівелює переваги процесору від Huawei перед користувачами, які не мають корпоративних можливостей купівлі власного TPU.

Список літератури:

[1] *Arafa M. et al.* Cascade lake: Next generation intel xeon scalable processor //IEEE Micro. – 2019. – Т. 39. – №. 2. – С. 29-36.

[2] NVIDIA Tesla V100 GPU architecture [Електронний ресурс] // NVIDIA. – 2017. – Режим доступу до ресурсу: https://images.nvidia.com/content/volta-architecture/pdf/volta-architecture-whitepaper.pdf.

[3] *Goldsborough P.* A tour of tensor flow //arXiv preprint arXiv:1610.01178. – 2016.

[4] *Вильчевская Е. Н.* Тензорная алгебра и тензорный анализ //СПб.: Изд-во Политехнического ун-та. – 2012.

[5] *McKinney W.* Python for data analysis: Data wrangling with Pandas, NumPy, and IPython. – " O'Reilly Media, Inc.", 2012.

[6] *Ray S. et al.* Understanding Support Vector Machine algorithm from examples (along with code) //Analytics Vidhya. – 2017. – Т. 13. – С. 19.

[7] *Hemsoth N.* First In-Depth Look at Google's TPU Architecture //The Next Platform, Apr. – 2017. – Т. 5. – С. 13.

[8] *Kennedy P.* Google Cloud TPU Details Revealed [Електронний ресурс] / Patrick Kennedy // STH. – 2017. – Режим доступу до ресурсу: https://www.servethehome.com/google-cloud-tpu-details-revealed/.

[9] *Dean J., Hölzle U.* Build and train machine learning models on our new google cloud tpus, 2017 //URL https://www.blog. google/topics/google-cloud/google-cloud-offer-tpus-machine-learning. – 2017.

[10] NVIDIA [Електронний ресурс] – Режим доступу до ресурсу: https://www.geforce.com/hardware/desktop-gpus/geforce-gtx-690/specifications.

[11] *Rowe J.* The Continuing Importance of GPUs For More Than Just Pretty Pictures //Haettu. – 2017. – Т. 31. – С. 2019.

[12] Процессор Ascend 910 [Електронний ресурс] . – 2019. – Режим доступу до ресурсу: https://e.huawei.com/ru/products/cloud-computing-dc/atlas/ascend-910.

[13] *Jouppi N. P. et al.* In-datacenter performance analysis of a tensor processing unit //Proceedings of the 44th Annual International Symposium on Computer Architecture. – 2017. – С. 1-12.

[14] *Osborne J.* Google's tensor processing unit explained: this is what the future of computing looks like //TechRadar. Available via http://www. techradar. com/. Accessed. – 2017. – Т. 6.

[15] *Abadi M. et al.* Tensorflow: Large-scale machine learning on heterogeneous distributed systems //arXiv preprint arXiv:1603.04467. – 2016.

[16] CUDA C++ Programming Guide [Електронний ресурс] – Режим доступу до ресурсу: https://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html.

Надійшла до редколегії 25.05.2020