ЧИСЛЕННЫЕ МЕТОДЫ И АНАЛИЗ ДАННЫХ

Повышение энергоэффективности нейросетевых вычислений с использованием NVDLA на ПЛИС

Е.С. Носкова [1], И.Е. Захаров [1], Ю.Н. Шкандыбин [1], С.Г. Рыкованов [1]
[1] Сколковский институт наук и технологий,
121205, Россия, г. Москва, ул. Большой бульвар, д. 30, стр.1

Аннотация

На сегодняшний день актуальна проблема создания высокопроизводительных и энергоэффективных аппаратных платформ для решения задач искусственного интеллекта. Популярным решением этой проблемы является использование ускорителей глубокого обучения для запуска нейросетей, таких как графические процессорные устройства и тензорные процессорные устройства. Компания NVIDIA предлагает программный комплекс NVDLA, позволяющий конструировать нейросетевые ускорители на базе открытого исходного кода. Данная статья описывает полный цикл создания прототипа ускорителя NVDLA на ПЛИС, а также тестирование полученного решения путем запуска на нем нейронной сети resnet-50. В завершение предоставляется оценка производительности и энергопотребления прототипа NVDLA ускорителя относительно GPU и CPU, результаты которой показывают превосходство NVDLA по многим характеристикам.

Ключевые слова: NVDLA, ПЛИС, inference, нейросетевые ускорители.

Цитирование: Носкова, Е.С. Повышение энергоэффективности нейросетевых вычислений с использованием NVDLA на ПЛИС/ Е.С. Носкова, И.Е. Захаров, Ю.Н. Шкандыбин, С.Г. Рыкованов // Компьютерная оптика. – 2022. – Т. 46, № 1. – С. 160-166. – DOI: 10.18287/2412-6179-CO-914.

Citation: Noskova ES, Zakharov IE, Shkandybin YN, Rykovanov SG. Towards energy-efficient neural network calculations. Computer Optics 2022; 46(1): 160-166. DOI: 10.18287/2412-6179-CO-914.

Введение

Задача внедрения систем, использующих «искусственный интеллект» (ИИ), делится на два этапа.

На первом этапе строится классификационная модель, обычно с помощью нейронных сетей и процесса глубокого машинного обучения (МО). Результатом МО является: а) структура модели, т.е. функциональные блоки, необходимые для обработки данных и б) мультипликаторы (веса), определяющие вероятности получения классификационных категорий из набора данных, на которых тренируется конкретная модель [1]. Данные технологии требуют огромной вычислительной мощности и поэтому обычно выполняются на высокопроизводительных серверах и суперкомпьютерах. К примеру, суперкомпьютер «Жорес» Сколковского института науки и технологий, предназначенный для решения задач искусственного интеллекта, состоит более чем из 100 мощных графических карт суммарной вычислительной мощностью около половины петафлопса [2], или суперкомпьютер «Anton», установленный в Питтсбургском суперкомпьютерном центре и предназначенный для моделирования молекул, состоит из 512 специализованных процессоров и обладает высокой параллельностью [3].

На втором этапе полученная модель применяется на новых данных, и это задача вывода (inference). Англоязычный термин inference переводится в русскоязычной литературе как «вывод» или «исполнение». В данной статье наиболее естественным будет термин «применение» (модели) как синоним вывода или исполнения. Если есть вероятность разночтения, то в скобках будет добавляться (inference).

Применение модели ставит новые задачи. Для МО главным является точность классификации, и вычислительные ресурсы играют подчиненную роль. После того, как модель определена, применение модели может многократно превосходить МО по количеству запусков и, следовательно, по объему совокупных требуемых ресурсов. В своих материалах Интел к 2020-му году оценивает отношение процессорных циклов на МО и применение модели как 1:5 [4], и ожидается, что это отношение будет только расти, умножая циклы на применения моделей.

Применение модели возможно на тех же вычислительных мощностях МО, но явно расточительно по отношению к ресурсам. С функциональной точки зрения применение модели отличается от обучения тем, что в применении модели отсутствует вычислительный процесс обратного распространения ошибки для уточнения параметров модели, и поэтому применение

модели требует меньше вычислительных ресурсов, однако дополнительно требует учета скорости потока данных, к которым будет применяться модель.

Поэтому при применении модели главным является расчет требуемых ресурсов: по вычислительной производительности, расходу электроэнергии, стоимости применяемого программно-аппаратного комплекса – отнесенных к характеристикам потока данных. При этом какой из вышеуказанных параметров будет основным, определяется областью применения модели. Так, для мобильных систем расход электроэнергии может быть основным критерием применимости модели, в то время как для интернета вещей стоимость может играть главную роль в связи с повсеместным внедрением систем ИИ на периферии (on the edge) [5].

При этом важно помнить, что скорость обработки данных является функциональным критерием применимости решения. Например, для мобильных систем обработки сенсорной информации высокого разрешения для автоматизации управления автомобилем применяются карты не ниже уровня GTX. Поэтому в дальнейшем мы сравниваем энергопотребление предложенного решения NVDLA именно с решениями такого типа [29].

Задача уменьшения ресурсов программно-аппаратного комплекса рассматривается и для центров обработки данных [6], в том числе для уменьшения энергопотребления и связанных с этим оперативных расходов. В этом контексте исследования идут по линии уменьшения разрядности вычислений при применении моделей для упрощения функциональных блоков процессоров и оптимизации доступа к данным в памяти. Так, рассматриваются низкопрецизионные вычисления, вплоть до 2-битных и 1-битных [7], для которых могут применяться аппаратные реализации, использующие программируемые логические матрицы (ПЛИС). В связи с этим главная трудность – это не уже доказанная возможность уменьшения разрядности вычислений, а необходимость гибкой (быстрой) адаптации новых вариантов моделей МО в возникающую аппаратно-ускоренную инфраструктуру применения (inference). Оптимизация аппаратной платформы под существующие модели МО является барьером на пути гибкого применения новых моделей МО, и учет возможной эволюции моделей в проектах аппаратного ускорения применения (inference) играет важную роль.

В настоящее время существует ряд аппаратных решений, которые используются как в системах для применения моделей (inference), так и для ускорения МО [8]. Ярким примером ускорителя для применения моделей можно назвать Google TPU [9]. Главным недостатком подобных ускорителей является их фиксированная структура, что ограничивает гибкость применения различных моделей МО, но обеспечивает лучшие параметры производительности и энергоэффективности. Более гибкий подход обеспечивается внедрением ускорителей на ПЛИС, т.к. программируемые логические матрицы можно перепрограммировать под новые модели при учете ограничений на размер используемых ПЛИС (ов) и менее оптимальные характеристики производительности и энергоэффективности по сравнению с ускорителями с фиксированной структурой [10]. При этом нужно учитывать трудность программирования ПЛИС по сравнению с программированием на CPU или GPU.

Задача программирования на ПЛИС облегчается тем, что для вычислений в нейронных сетях используются всего несколько стандартных операций, таких как матричное умножение и конволюция (свертка) [1]. Поэтому мы рассматриваем процесс создания прототипа ускорителя применения моделей (inference) из укрупненных функциональных блоков, предложенный фирмой Nvidia – NVDLA [11]. В результате шагов программирования по схеме NVDLA можно получить реализацию нейронной сети на ПЛИС непосредственно из описания модели МО в известных схемах (таких как Caffe, например [12]), что существенно сокращает процесс создания прототипа ускорителя. Таким образом, возможно применение ускорителя нейронных сетей на ПЛИС, которое гибко адаптируется под изменяющиеся модели МО.

NVDLA – архитектура с открытым исходным кодом. Исходный код NVDLA состоит из двух частей: аппаратная часть, которая используется для создания архитектуры разрабатываемого ускорителя на ПЛИС, и программная часть, позволяющая компилировать архитектурную модель нейронной сети из описания верхнего уровня и организующая поток данных между CPU и ПЛИС для применения модели.

Помимо этого, репозиторий NVDLA предоставляет доступ к виртуальной платформе – программному симулятору. С его помощью можно запускать и отлаживать нейросеть, используя только CPU компьютера без аппаратной реализации на ПЛИС.

Данная работа ставит своей целью построение модели NVDLA для популярного бенчмарка обработки изображений Resnet и получение на ней производительности, сравнимой с современными CPU, при значительно меньшем энергопотреблении. Хотя используются стандартные блоки NVDLA, в статье рассматриваются детали реализации модели на ПЛИС в облаке для исследования производительности и энергоэффективности такого подхода.

В первой части статьи будет кратко изложена функциональная база NVDLA, затем будут подробно описаны подходы к разработке ускорителя в зависимости от задач применения МО. В завершение будут представлены результаты по оценке производительности и энергопотребления NVDLA ускорителя по сравнению с CPU и GPU.

1. Функциональное описание NVDLA

NVDLA – открытая архитектура, состоящая из функциональных блоков, используемых для ускорения

основных этапов нейросетевых вычислений, таких как конволюция, нормализация и многие другие. NVDLA является масштабируемой и легко настраиваемой архитектурой, что упрощает ее реализацию и использование для широкого спектра встроенных устройств. Существуют различные конфигурации NVDLA (nv_large, nv_small, nv_medium), которые отличаются между собой количеством вычислительных ресурсов, а также наличием дополнительных оптимизаций в аппаратуре. Выбор конфигурации зависит от задачи, для которой будет использоваться аппаратура.

Аппаратная часть NVDLA представляет собой набор конфигурируемых пользователем модулей, написанных на языке Verilog и отвечающих за различные функциональные блоки, которые в сумме представляют собой Verilog-описание NVDLA. Для полноценной работы ускорителя готовый модуль NVDLA соединяют с ядром микропроцессора, благодаря которому организуется обмен информацией между программной частью и NVDLA-модулем.

Программная часть состоит из двух компонентов: инструменты компиляции и среда исполнения. Под инструментами компиляции подразумевается специальный компилятор, превращающий нейронную сеть в формате Caffe в список «аппаратных слоев», где каждый из слоев использует отдельный функциональный блок. Среда исполнения содержит драйвер ядра Linux, поддерживающий обработку NVDLA-специфичных запросов от аппаратуры, а также программную обертку, позволяющую запускать скомпилированную нейронную сеть на построенном ускорителе.

2. Подходы к разработке

Для работы с NVDLA можно использовать различные платформы, а именно: аппаратный ускоритель, программный симулятор в виде программы на высокоуровневом языке программирования, запускаемой на любом процессоре, или построить симулятор с частичным или полным аппаратным ускорением на базе ПЛИС. Каждый из подходов имеет свои положительные и отрицательные стороны и используется для определенных целей [13].

Для исследования программного комплекса NVDLA и проверки работоспособности NVDLA-ускорителя удобно использовать виртуальную платформу, которая является программным симулятором и представляет собой SystemC-модель NVDLA, соединенную с SystemC-моделью процессорного ядра с архитектурой ARM [17]. Такая виртуальная платформа позволяет запускать скомпилированную нейросеть в отсутствие аппаратного ускорителя. Недостатком виртуальной платформы является низкая скорость эмуляции, к тому же виртуальная платформа не является потактовой моделью, исключает возможность оценки эффективности решения.

В случае, когда ускоритель необходим для промышленного использования в реальных встроенных платформах, изготавливают аппаратный ускоритель в виде ASIC - интегральной схемы специального назначения. Ключевыми особенностями данного подхода является специализация под задачу и, в результате, высокая эффективность, выражающаяся в возможности получения высокой тактовой частоты (~ 1 – 2 ГГц) при низком энергопотреблении (~ 3 Вт) (см. например, BeagleV проект [14]). Недостатком такого подхода является высокая стоимость разработки [15] и невозможность перепрограммирования схемы, что делает ASIC непригодным для исследовательских целей.

Более подходящим вариантом для проведения исследования является аппаратный ускоритель на базе ПЛИС (~ 100 МГц) со встроенным CPU-ядром (~ 1 ГГц) [16]. Его характеристики можно менять за счет многократного перепрограммирования и благодаря этому снимать различные показатели поразному сконфигурированного ускорителя.

В данном исследовании использовалась арендованная на AWS платформа с ПЛИС – Xilinx Virtex UltraScale+ VU9P. Эта микросхема является высокопроизводительной, но, к сожалению, не имеет встроенного CPU-ядра, поэтому потребовалось частично эмулировать эту часть функционала. Такая эмуляция возможна в двух конфигурациях:

а) гибридная симуляция, при которой процессорное ядро эмулируется программно, а модель NVDLA зашивается в ПЛИС;
б) полностью аппаратно ускоренная симуляция, при которой оба компонента (процессорное ядро и модель NVDLA) являются частью ПЛИС.

Важно отметить, что конструирование как гибридного, так и аппаратно ускоренного симулятора включает аппаратный синтез модуля NVDLA, являющийся основной частью данной работы. Дадим краткое описание этому процессу.

Прежде всего выбирается конфигурация ускорителя, наиболее подходящая для верхнеуровневой модели нейронной сети (Caffe), затем генерируется ее Verilog-описание с помощью компилятора NVDLA. Затем стандартными средствами Verilog данное описание синтезируется под конкретную модель ПЛИС, и на конечном этапе схема прошивается [18]. На этапе синтеза модели собираются такие данные, как мощность полученной схемы и количество использованных ресурсов ПЛИС. Схему гибридного симулятора, который был использован в данной работе, можно видеть на рис. 1.

Гибридный симулятор существенно ускоряет исполнение задач по сравнению с полностью программным, но, к сожалению, остается потактово неточным и вследствие этого не позволяет замерять производительность полученной системы, что является его главным недостатком. Полностью аппаратно ускоренный симулятор лишен этой проблемы и поэтому может быть использован в научных исследованиях.

В качестве примера рассмотрим аппаратно ускоренный симулятор, представленный в работе [19], использующий в роли (модели) процессорного ядра ядро с открытой архитектурой RISC-V, которая на сегодняшний день приобретает большую популярность. Данный симулятор представляет собой модуль NVDLA, подсоединенный по каналу памяти и каналу управления к Rocket-ядру, на котором исполняется программная часть NVDLA.

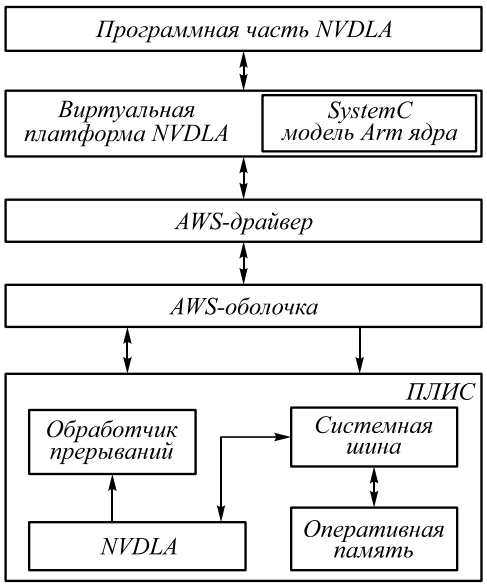

Рис. 1. Схема гибридного симулятора NVDLA-ускорителя на платформе AWS ПЛИС

Rocket-ядро – микропроцессорное ядро на базе архитектуры RISC-V. Схематическое устройство описанного симулятора можно видеть на рис. 2, а в табл. 1 содержится информация об использовании ресурсов облачной ПЛИС при его изготовлении (отчет получен в Vivado design suite). Согласно отчету, ресурсы Xilinx Virtex VU9P ПЛИС, которая на данный момент является одной из самых крупных у Xilinx, были использованы практически полностью, что оправдывает использование представленной модели для построения прототипа.

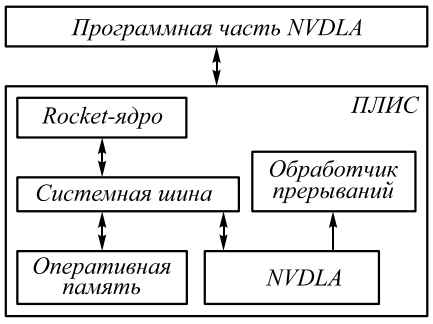

Рис. 2. Схема полностью аппаратно ускоренного NVDLA-ускорителя на платформе AWS ПЛИС

Важно отметить, что аппаратно ускоренный симулятор также не является идеальным с точки зрения оценки производительности, так как в нем не соблюдено соотношение между частотами ядра процессора и ПЛИС, которое наблюдается на физической плате с реализованным аппаратным ядром. Из-за этого итоговое время работы модели нейросети на ПЛИС может в какой-то степени искажаться, так как время обработки нейросетевых операций в модуле NVDLA и в процессорном ядре будут зависеть от их частоты.

В данной работе была оценена энергоэффективность работы прототипа NVDLA в предположении, что основную нагрузку несет NVDLA-модуль (т.е. велик процент времени, занимаемого работой NVDLA, в общем времени работы задачи, а задержка на подготовку данных в ядре мала). Этот факт является важным, так как на физической плате частота процессорного ядра увеличится в ~20 раз, а частота работы модуля NVDLA всего лишь в ~1,5 раза, и за счет этого время работы ядра сократится более существенно, чем время работы NVDLA.

Табл. 1. Использование ресурсов ПЛИС Xilinx Virtex UltraScale+ VU9P при изготовлении полностью аппаратно ускоренного симулятора NVDLA на базе ядра RISC-V

	Использованные ресурсы	Доступные ресурсы	% использования
LUT	785069	895200	87,7
CLB reg	531142	1790400	29,67
CLB	111357	111900	99,51
BRAM	563	1680	33,51
DSP	320	5640	5,67

В данной работе на первоначальных этапах была использована виртуальная платформа NVDLA для проверки работоспособности ускорителя. В дальнейшем мы использовали как программный, так и аппаратно ускоренные симуляторы и прошли все описанные этапы разработки и тестирования прототипа ускорителя NVDLA на ПЛИСе. Таким образом была получена оценка по энергопотреблению и производительности такого решения.

3. Результаты и их обсуждение

Итогом данного исследования стала сравнительная характеристика прототипа ускорителя NVDLA-конфигурации nv_large по отношению к стандартным аппаратным решениям для запуска нейронных сетей, а именно CPU и GPU. Сравнение выполнялось по двум основным параметрам: производительность и энергоэффективность. В качестве CPU были использованы Intel Xeon Gold 5217 с частотой 3 ГГц и Intel Xeon Silver 4114 с частотой 2,2 ГГц [24], а в качестве GPU – Nvidia GeForce GTX 1080Ti [22] и Nvidia GeForce RTX 2080Ti [23]. Для измерения производительности была использована Caffe-модель нейросети resnet-50 (для запуска на NVDLA resnet-50 была пропущена через NVDLA-компилятор для снижения разряности весов до 8 бит) [20].

График на рис. 3 иллюстрирует различия в скорости обработки кадров для CPU, GPU и прототипа NVDLA (nv_large)-ускорителя. Результаты для CPU и

GPU были получены за счет запуска и замеров времени прохождения resnet-50 на узлах суперкомпьютера «Жорес» Сколковского института науки и технологии. Для получения результатов работы NVDLA был развернут полностью аппаратно-ускоренный симулятор с RISC-V ядром на базе Xilinx Virtex UltraScale+ VU9P [21], описанный в предыдущем параграфе, на котором и было снято время прохождения resnet-50.

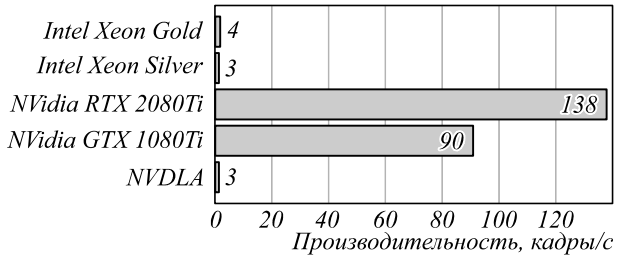

Рис. 3. Производительность полностью аппаратно ускоренного симулятора NVDLA(nv_large) на resnet-50 по сравнению с CPU и GPU

В табл. 2 представлено энегопотребление для CPU, GPU на запуске тестовой задачи. Для CPU актуальное энергопотребление было измерено при помощи утилиты likwid-perfctr [25]. Для GPU в силу особеностей тестовых моделей, вместо энергопотребления, был измерен процент загрузки оборудования (GPU) во время прохождения задачи, а затем номинальные значения мощности (принятые за номинальные 250 Ватт [26, 27]) для представленных моделей были отмасштабированы с учетом полученной загрузки. В данных расчетах предполагаем, что энергопотребление GPU линейно зависит от его загрузки. Для Nvidia RTX 2080Ti коэффициент загрузки – 87 %, а для Nvidia GTX 1080 – 90 %. При этом стабильность загрузки, измеренная программой мониторинга, на Жоресе [28] варьируется в диапазоне ±3 %. Измерения энергопотребления при помощи утилиты likwid-perfctr для CPU, а также с использованием значения загрузки для GPU в общем случае несут оценочный характер, так как предоставляют результаты потребления целой системы, не выделяя процесс «исполнения» нейросети. Однако в силу того, что система эксплуатировалась в однопользовательском режиме для этих измерений, данная оценка является весьма точной.

Также в таблице представлена номинальная мощность аппаратно ускоренного симулятора NVDLA (nv_large). Мощность NVDLA на ПЛИС Xilinx Virtex UltraScale+ VU9P была оценена в программе Vivado design suite [18] на этапе синтеза сгенерированного Verilog-модуля.

Табл. 2. Мощность аппаратно ускоренного NVDLA (nv_large)-симулятора на ПЛИС по сравнению с GPU и CPU, Вatт. Точность измерений – около 6 %

Intel Xeon Gold	Intel Xeon Silver	Nvidia GTX 2080Ti	Nvidia GTX 1080Ti	NVDLA
136	80	217	225	5,5

Объединив результаты по производительности на рис. 3 со значениями из табл. 1, получим производительность на единицу энергопотребления, что и показано на рис. 4.

Согласно полученным результатам, можно сказать, что прототип NVDLA ускорителя отличается низким энергопотреблением: 5,5 Вт по сравнению с CPU и GPU. Можно отметить, что производительность RTX / GTX GPU-решений в 30 – 40 раз выше, чем модели реализованной NVDLA, но производительность NVDLA сравнима с производительностью на CPU при значительно меньшем энергопотреблении (те же ~ 30 – 40 раз). Таким образом, энергоэффективность ускорителя нейросети на ПЛИСе в программной реализации NVDLA можно считать доказанной, даже принимая номинальную мощность NVDLA из ресурсной оценки.

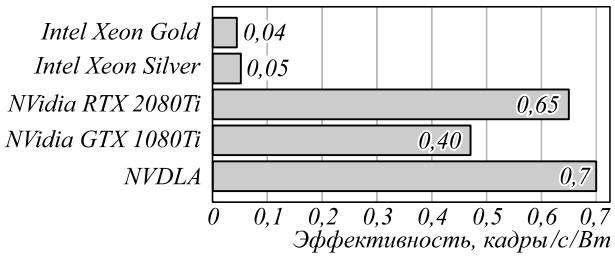

Рис. 4. Производительность полностью аппаратно ускоренного симулятора NVDLA (nv_large) на единицу энергопотребления по сравнению с CPU и GPU

Необходимо подчеркнуть, что проведенные эксперименты демонстрируют производительность аппаратно ускоренного симулятора NVDLA, который работает на частоте 75 МГц (столь низкая частота объясняется сложностью прошиваемой логики и, вследствие этого, большой длиной критического пути). Увеличение частоты работы NVDLA на ПЛИС можно достичь за счет оптимизации Verilog-описания под ПЛИС, однако для прототипа данный процесс производить бессмысленно, так как прототип не используется для решения реальных задач. Производительность NVDLA на ПЛИС со встроенным ядром будет выше в связи с более высокой частотой ядра (1,5 ГГц) и более высокой частотой работы модуля NVDLA на ПЛИС (до 100 МГц). Таким образом, мы делаем вывод, что программный комплекс NVDLA с реализацией на ПЛИС со встроенным ядром является пригодным для решения реальных задач.

К сожалению, речь идет о качественной оценке, количественную оценку преимущества такого решения невозможно сделать на основе проведенных опытов. Даже зная рабочие частоты для физической платы со встроенным ядром, результаты для симулятора нельзя экстраполировать до этих значений, так как у физической платы соотношение частот работы ядра и модуля NVDLA гораздо выше, чем у аппаратно ускоренного симулятора. Для оценки времени прохождения тестовой задачи на физической плате со встроен-

ным ядром, зная время ее прохождения на симуляторе, необходимо знать вклад работы ядра и модуля NVDLA в суммарное время прохождения задачи. Получение такой информации достаточно трудоемко и требует построения системы мониторинга выполнения инструкций и простоя конвейера, что может являться целью будущей работы.

Заключение

В данной статье были описаны подходы к созданию прототипа ускорителя применения модели (inference) глубокого обучения на базе открытого проекта NVDLA для бенчмарка Resnet. Проделанная работа показала, что NVDLA является рабочим решением для создания нейросетевых ускорителей, и одновременно помогла выявить места, требующие особого внимания в процессе разработки, а именно: выбор платформы для реализации ускорителя, учитывая поставленные цели, оценка финансовых вложений (что не акцентировалось в статье), требуемых для получения готового продукта, и понимание, что открытый продукт (в данном случае NVDLA) еще требует существенной доработки для реализации поставленных целей.

Однако наиболее значимыми результатами стали измерения производительности и энергопотребления, которые доказали, что программный комплекс NVDLA с аппаратной реализацией на ПЛИС даже в симуляторной версии имеет производительность, сравнимую с CPU, и меньшее энергопотребление, что обеспечивает высокую производительность на единицу энергопотребления.

References

[1] Goodfellow I, Bengio Y, Courville A. Deep learning. Cambridge: The MIT Press; 2016.

[2] Zacharov I, Arslanov R, Gunin M, Stefonishin D, Pavlov S, Panarin O, Maliutin A, Rykovanov SG, Fedorov M. "Zhores" – Petaflops supercomputer for data-driven modeling, machine learning and artificial intelligence installed in Skolkovo Institute of Science and Technology. Open Eng 2019; 9(1): 512-520.

[3] Shaw DE, Deneroff MM, Dror RO, et al. Anton, a special-purpose machine for molecular dynamics simulation. Commun ACM 2008; 51(7): 91-97.

[4] Singer G. Deep Learning is coming of age. 2018. Source: ⟨https://www.nextplatform.com/2018/10/18/deep-learning-is-coming-of-age/⟩.

[5] Merenda M, Porcaro C, Iero D. Machine learning for AI-enabled IoT devices: a review. Sensors 2020; 20(9): 2533.

[6] Park J, Naumov M, Basu P, et al. Deep learning inference in facebook data centers: Characterization, performance optimizations and hardware implications. arXiv preprint arXiv:1811.09886. 2018. Source: ⟨https://arxiv.org/abs/1811.09886⟩.

[7] Mishra A, Nurvitadhi E, Cook J. Marr D. WRPN: Wide reduced-precision networks. ICLR (Poster) 2018.

[8] Chen Y, Xie Y, Song L, Chen F, Tang T. A survey of accelerator architectures for deep neural networks. Engineering 2020; 6(3): 264-274.

[9] Jouppi NP, Young C, Patil N, et al. In-datacenter performance analysis of a tensor processing unit. Proc 44th Annual Int Symposium on Computer Architecture 2017: 1-12.

[10] Guo K, Zeng S, Yu J, Wang Y, Yang H. A survey of FPGA-based neural network accelerator. arXiv preprint arXiv:1712.08934. 2017. Source: ⟨https://arxiv.org/abs/1712.08934⟩.

[11] NVDLA. ⟨Source: http://nvdla.org/⟩.

[12] Jia Y, Shelhamer E, Donahue J, Karayev S, Long J, Girshick RB, Guadarrama S, Darrell T. Caffe: Convolutional architecture for fast feature embedding. Proc 22nd ACM Int Conf on Multimedia 2014: 675-678.

[13] Tan Z, Waterman A, Cook H, Bird S, Asanovic K, Patterson D. A case for FAME: FPGA architecture model execution. ACM SIGARCH Computer Architecture News 2010; 38(3): 290-301.

[14] BeagleV Forum. Source: ⟨https://beagleboard.org/beaglev⟩.

[15] The economics of ASICs: At what point does a custom SoC become viable? Source: ⟨https://www.electronicdesign.com/technologies/embedded-revolution/article/21808278/the-economics-of-asics-at-what-point-does-a-custom-soc-become-viable⟩.

[16] Xilinx Zynq UltraScale+ MPSoCZCU104 evaluation kit Source: ⟨https://www.electronicdesign.com/technologies/embedded-revolution/article/21808278/the-economics-of-asics-at-what-point-does-a-custom-soc-become-viable⟩.

[17] Delbergue G, Burton M, Konrad F, Le Gal B, Jego C. QBox: An industrial solution for virtual platform simulation using QEMU and SystemC TLM-2.0. 8th European Congress on Embedded Real Time Software and Systems (ERTS 2016) 2016: hal-01292317.

[18] The Xilinx Vivado. Source: ⟨https://www.xilinx.com/products/design-tools/vivado.html⟩.

[19] Farshchi F, Huang Q, Yun H. Integrating NVIDIA deep learning accelerator (NVDLA) with RISC-V SoC on FireSim. 2019 2nd Workshop on Energy Efficient Machine Learning and Cognitive Computing for Embedded Applications (EMC2) 2019: 21-25.

[20] He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. 2016 IEEE Conf on Computer Vision and Pattern Recognition (CVPR) 2016: 770-778.

[21] UltraScale+ FPGA product tables and product selection guide. Source: ⟨https://www.xilinx.com/support/documentation/selection-guides/ultrascale-plus-fpga-product-selection-guide.pdf⟩.

[22] GeForce GTX 1080 Ti. Source: ⟨https://www.nvidia.com/en-sg/geforce/products/10series/geforce-gtx-1080-ti/⟩.

[23] GeForce RTX 2080 Ti. Source: ⟨https://www.nvidia.com/ru-ru/geforce/graphics-cards/rtx-2080-ti/⟩.

[24] Second Generation Intel Xeon scalable processors datasheet. Source: ⟨https://www.intel.ru/content/www/ru/ru/products/docs/processors/xeon/2nd-gen-xeon-scalable-datasheet-vol-1.html⟩.

[25] Likwid perfctr. Source: ⟨https://github.com/RRZE-HPC/likwid/wiki/likwid-perfctr⟩.

[26] TechPowerUp. NVIDIA GeForce RTX 2080 Ti. Source: ⟨https://www.techpowerup.com/gpu-specs/geforce-rtx-2080-ti.c3305⟩.

[27] TechPowerUp. NVIDIA GeForce GTX 1080 Ti. Source: ⟨https://www.techpowerup.com/gpu-specs/geforce-gtx-1080-ti.c2877⟩.

[28] Zakharov IE, Panarin OA, Rykovanov SG, Zagidullin RR, Malyutin AK, Shkandybin YuN, Ermekova AE. Monitor-

ing applications on the ZHORES cluster at Skoltech. Program Systems: Theory and Applications 2021; 12(2:49): 73-103.

[29] Panarin OA, Zacharov IE. Monitoring mobile information processing systems. Russian Digital Libraries Journal 2020; 23(4): 835-847.

Сведения об авторах

Носкова Елизавета Сергеевна, бакалавр, в 2019 году получила степень бакалавра МФТИ по специальности 09.03.01 «Информатика и вычислительная техника», на данный момент является студентом магистратуры Сколковского института наук и технологий по специальности «Information Science and Technology», а также студентом магистратуры МФТИ по специальности 09.03.01 «Информатика и вычислительная техника», работает программистом в АО «МЦСТ». Область научных интересов: нейросетевые ускорители, архитектура процессоров, бинарная трансляция.
E-mail: *elizaveta.noskova@skoltech.ru* .

Захаров Игорь Евгеньевич, PhD, работает старшим научным сотрудником Сколковского института науки и технологий. Область научных интересов: высокопроизводительные компьютерные системы, системное программирование. E-mail: *i.zacharov@skoltech.ru* .

Шкандыбин Юрий Николаевич, системный архитектор суперкомпьютера «Жорес» Сколковского института наук и технологий. E-mail: *y.shkandybin@skoltech.ru* .

Рыкованов Сергей Георгиевич, PhD, окончил Московский государственный университет. Работает доцентом Сколковского института науки и технологий в Центре вычислительной науки и техники. Область научных интересов: высокопроизводительные компьютерные системы, лазерная плазма, источники рентгеновского излучения, ускорение частиц, вычислительная физика. E-mail: *s.rykovanov@skoltech.ru* .

ГРНТИ: 50.33.04
Поступила в редакцию 24 апреля 2021 г. Окончательный вариант – 4 сентября 2021 г.

Towards energy-efficient neural network calculations

E.S. Noskova [1], I.E. Zakharov [1], Y.N. Shkandybin [1], S.G. Rykovanov [1]
[1] Skolkovo Institute of Science and Technology,
121205, Moscow, Russia, Bolshoi boulevard, 30, building 1

Abstract

Nowadays, the problem of creating high-performance and energy-efficient hardware for Artificial Intelligence tasks is very acute. The most popular solution to this problem is the use of Deep Learning Accelerators, such as GPUs and Tensor Processing Units to run neural networks. Recently, NVIDIA has announced the NVDLA project, which allows one to design neural network accelerators based on an open-source code. This work describes a full cycle of creating a prototype NVDLA accelerator, as well as testing the resulting solution by running the resnet-50 neural network on it. Finally, an assessment of the performance and power efficiency of the prototype NVDLA accelerator when compared to the GPU and CPU is provided, the results of which show the superiority of NVDLA in many characteristics.

Keywords: NVDLA, FPGA, inference, deep learning accelerators.

Citation: Noskova ES, Zakharov IE, Shkandybin YN, Rykovanov SG. Towards energy-efficient neural network calculations. Computer Optics 2022; 46(1): 160-166. DOI: 10.18287/2412-6179-CO-914.

Authors' information

Elizaveta Sergeevna Noskova, bachelor, get bachelor degree at Moscow Institute of Science and Technology in 2019, majoring in Informatics and Computer Engineering. Currently is getting a master degree at Skolkovo Institute of Science and Technology, majoring in Information Science and Technology and at Moscow Institute of Science and Technology in 2019, majoring in Informatics and Computer Engineering and works as a programmer the MCST. Research interests are deep learning accelerators, computer architecture, binary translation.
E-mail: *elizaveta.noskova@skoltech.ru* .

Igor Evgenievich Zacharov, PhD, works as a senior researcher at the Skolkovo Institute of Science and Technology, specialist in the field of high-performance computer systems and system programming.
E-mail: *i.zacharov@skoltech.ru* .

Yuri Nikolaevich Shkandybin, system architect of "Zhores" supercomputer of Skolkovo Institute of Science and Technology. E-mail: *y.shkandybin@skoltech.ru* .

Sergey Georgievich Rykovanov, PhD, studied at the Moscow State University. Works as Asscociate Professor at Skolkovo Institute of Science and Technology in the Center for Computational and Data-Intensive Sciene and Engineering. Research interests: high-performance computer systems, laser plasma, X-ray sources, particle acceleration, computational physics. E-mail: *s.rykovanov@skoltech.ru* .

Received April 24, 2021. The final version – September 4, 2021.