Министерство науки и высшего образования Российской Федерации
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ (НИ ТГУ)
Радиофизический факультет
Кафедра информационных технологий в исследовании дискретных структур

ДОПУСТИТЬ К ЗАЩИТЕ В ГЭК
Руководитель ООП
д-р физ.-мат. наук, профессор

_____ В.П. Гермогенов
подпись
«19» 06 2021 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА

АППАРАТНАЯ РЕАЛИЗАЦИЯ ИСКУССТВЕННОЙ НЕЙРОННОЙ СЕТИ НА ПЛИС

по направлению подготовки 03.03.03 Радиофизика
направленность (профиль) «Радиофизика, электроника и информационные системы»

Берзин Артем Константинович

Руководитель ВКР
канд. физ.-мат. наук, доцент

_____ М.Л. Громов
подпись
«14» июня 2021 г.

Автор работы
студент группы № 776

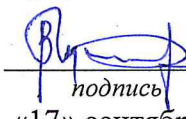
_____ А.К. Берзин
подпись
«14» июня 2021 г.

Томск – 2021

Задания на выполнение ВКР

Министерство науки и высшего образования Российской Федерации.
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ (НИ ТГУ)
Радиофизический факультет
Кафедра информационных технологий в исследовании дискретных структур

УТВЕРЖДАЮ
Руководитель ООП
д-р физ.-мат. наук, профессор


подпись В.П. Гермогенов

«17» сентября 2020 г.

ЗАДАНИЕ

на выполнение выпускной квалификационной работы бакалавра обучающемуся
Берзину Артему Константиновичу
Фамилия Имя Отчество обучающегося
по направлению подготовки 03.03.03 Радиофизика, направленность (профиль) «Радиофизика, электроника и информационные системы»

1 Тема выпускной квалификационной работы
Аппаратная реализация искусственной нейронной сети на ПЛИС

2 Срок сдачи обучающимся выполненной выпускной квалификационной работы:
а) на кафедру – 14.06.2021 б) в ГЭК – 18.06.2021

3 Исходные данные к работе:

Объект исследования –	аппаратная реализация искусственных нейронных сетей на программируемых логических интегральных схемах (ПЛИС)
Предмет исследования –	быстродействие аппаратной реализации искусственных нейронных сетей на ПЛИС по сравнению с существующими распространенными программными реализациями
Цель исследования –	разработка аппаратной реализации полносвязного и сверточного слоев искусственных нейронных сетей на ПЛИС; сравнение реализации с существующими альтернативами

Задачи:
Изучение принципов построения и обучения искусственных нейронных сетей
Изучение подходов и инструментов реализации искусственных нейронных сетей
Изучение инструментов для работы с ПЛИС
Изучение аппаратных алгоритмов организации вычислений на ПЛИС; выбор алгоритмов, подходящих для реализации искусственных нейронных сетей на ПЛИС
Разработка полносвязного слоя искусственных нейронных сетей
Разработка сверточного слоя искусственных нейронных сетей
Реализация полносвязного и сверточного слоев с применением альтернативных подходов (на CPU, на GPU)
Организация и проведение сравнительных экспериментов

Методы исследования:
Экспериментальные исследования, статистическая обработка результатов

Организация или отрасль, по тематике которой выполняется работа, –

4 Краткое содержание работы

В процессе работы будут разработаны аппаратные реализации полносвязного и сверточного слоев искусственных нейронных сетей на программируемых логических интегральных схемах (ПЛИС). Производительность разработанных реализаций будет сравнена с существующими распространенными программными реализациями искусственных нейронных сетей.

Руководитель выпускной квалификационной работы

Доцент, НИ ТГУ		М.Л. Громов
должность, место работы	*подпись*	*И.О. Фамилия*

Задание принял к исполнению

Студент, НИ ТГУ		А.К. Берзин
должность, место работы	*подпись*	*И.О. Фамилия*

АННОТАЦИЯ

Выпускная квалификационная работа бакалавра содержит: 43 страницы, 2 главы, 15 рисунков, 6 таблиц.

ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ, ПЛИС, АППАРАТНАЯ РЕАЛИЗАЦИЯ, ЯЗЫКИ ОПИСАНИЯ АППАРАТУРЫ, СВЕРТКА МАТРИЦ, УМНОЖЕНИЕ МАТРИЦ, ВСТРАИВАЕМЫЕ СИСТЕМЫ.

Выпускная работа посвящена разработке аппаратной реализации полносвязного и сверточного слоев искусственных нейронных сетей на программируемых логических интегральных схемах (ПЛИС).

В ходе работы проведено изучение существующих распространенных методов и инструментов программной реализации искусственных нейронных сетей, методы и инструменты реализации операций, требуемых для вычисления искусственных нейронных сетей, на ПЛИС. Разработаны аппаратные реализации полносвязного и сверточного слоев искусственных нейронных сетей, выполнено сравнение разработанной реализации с распространенными подходами программной реализации искусственных нейронных сетей.

Разработанная аппаратная реализация отличается высокой производительностью в сравнении с существующими подходами ускорения искусственных нейронных сетей, применяемых во встраиваемых системах.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Глубокое обучение – раздел машинного обучения, сосредоточенный на использовании искусственных нейронных сетей – является одной из ключевых и самых быстро развивающихся областей науки и техники двадцать первого века. Решения, основанные на применении искусственных нейронных сетей, опережают традиционные алгоритмы компьютерного зрения [1] [2], распознавания речи, обработки естественного языка [3] и многих других областей.

С развитием Интернета вещей (IoT) возрос интерес к использованию искусственных нейронных сетей во встраиваемых системах (embedded systems) [4]. Стандартные решения на основе центральных процессоров (CPU) и графических процессоров (GPU), обычно используемые для вычисления искусственных нейронных сетей, не применимы в случае встраиваемых систем из-за жестких ограничений по потребляемой и выделяемой мощности и естественных ограничений на габариты используемых компонентов. Часто применяемые во встраиваемых системах 32-битные процессоры архитектуры ARM, как правило, не могут предоставить достаточно вычислительной мощности для использования искусственных нейронных сетей, что вынуждает изыскивать иные решения проблемы: полностью отказываться от использования нейронных сетей, использовать простые, но менее точные архитектуры нейронных сетей (например, SqueezeNet [5]), применять более производительные вычислительные устройства.

Структура искусственных нейронных сетей предполагает легкое распараллеливание вычислений. Этот факт обусловлен тем, что нейронные сети представляют собой композицию матричных умножений, матричных сверток и поэлементных нелинейных функций. Известно, что каждый элемент в результирующей матрице может быть вычислен отдельно от других, потому до некоторого предела с ростом числа потоков, в которых

производится вычисления, время вычисления результата матричной операции сокращается. Программируемые логические интегральные схемы (ПЛИС), в силу своей внутренней структуры, могут быть эффективно применены при распараллеливании вычислений. Также эти интегральные микросхемы как правило имеют сравнительно небольшое энергопотребление и маленький размер, благодаря чему часто применяются во встраиваемых системах, в том числе и в составе систем на кристалле (SoC). В отличие от процессоров общего назначения и графических процессоров у ПЛИС нет вычислительных ядер, а потому максимальное число потоков, в которых производятся вычисления, ограничено лишь количеством логических элементов на кристалле. Эти факты спровоцировали рост интереса к применению ПЛИС для ускорения нейронных сетей.

Следует отметить, что глубокое обучение является сравнительно новой областью науки. К примеру, активное применение нейронных сетей для решения реальных задач науки и промышленности, связанных с компьютерным зрением, приходится на вторую декаду XXI века, и связано с работами, в которых были предложены архитектуры AlexNet [1], ResNet [6] и другие. Помимо этого, глубокое обучение является динамичной и постоянно развивающейся прикладной наукой: разработанные подходы, методы и архитектуры непрерывно появляются, совершенствуются и сменяются новыми. Указанные две особенности области приводят к тому, что в глубоком обучении, как и в области ускорения искусственных нейронных сетей, в настоящий момент не существует классических и общепризнанных учебников и монографий. Исследователи и разработчики, связанные с данной областью, в своей работе опираются главным образом на новые научные публикации, подкрепленные воспроизводимыми примерами кода.

Разработка аппаратной реализации искусственных нейронных сетей является сложной задачей, требующей комплексного подхода. Настоящая работа посвящена разработке аппаратной реализации полносвязного и

сверточного слоев искусственных нейронных сетей на ПЛИС; сравнению реализации с существующими альтернативами.

1 Искусственные нейронные сети

1.1 Устройство искусственных нейронных сетей

Искусственные нейронные сети (ИНС), также называемые просто нейронными сетями, представляют собой вычислительные системы, вдохновленные биологическими нейронными сетями, составляющими нервную систему животных.

Искусственные нейронные сети могут быть представлены в виде композиции искусственных нейронов. Искусственный нейрон – это математическая модель биологического нейрона, задаваемая следующей формулой:

$$y = \varphi\left(\sum_j w_j x_j + b\right), \quad (1)$$

где y – значение на выходе нейрона;

φ – в общем случае произвольная нелинейная функция, называемая функцией активации;

w_j – весовой коэффициент соответствующего входа нейрона;

x_j – сигнал на соответствующем входе нейрона;

b – параметр смещения.

Внутри искусственных нейронных сетей искусственные нейроны могут объединяться в различные типовые конфигурации с обобщенными входами, называемые слоями. Выделяют, например, полносвязные и сверточные слои нейронных сетей.

Существует множество различных архитектур искусственных нейронных сетей: полносвязные, сверточные, рекуррентные, трансформеры и другие.

Полносвязная нейронная сеть представляют собой композицию полносвязных слоев, а сверточная нейронная сеть – композицию, главным

образом, сверточных слоев. Помимо сверточных слоев, выполняющих основные вычисления, в сверточных нейронных сетях также могут присутствовать слои подвыборки (в новых архитектурах зачастую отсутствуют), слои нормализации (могут совмещаться со сверточными слоями), полносвязные слои. Сверточные нейронные сети, не содержащие полносвязных слоев, называют полносверточными. Благодаря отсутствию полносвязных слоев, полносверточные нейронные сети могут принимать на вход данные произвольных размеров (с некоторыми ограничениями). В частности, такие нейронные сети широко используются в задачах улучшения качества изображений, обработки сигналов и других. Из указанных утверждений следует, что главными функциональными блоками полносвязных и сверточных нейронных сетей являются соответственно полносвязные слои и сверточные слои.

Устройство рекуррентных нейронных сетей [7], а также трансформеров [8], во многом опирается на устройство полносвязных нейронных сетей, однако является более сложным с архитектурной точки зрения. Так, в случае рекуррентных нейронных сетей, сигнал с выхода нейронной сети на следующей итерации подается обратно на вход, а элементарный повторяющийся элемент в трансформерах представляет собой композицию нескольких матричных умножений и полносвязных слоев со сложной маршрутизацией сигналов.

Исходя из вышеуказанных фактов, было решено в настоящей работе обозреть устройство и рассмотреть вопросы аппаратного ускорения лишь полносвязных и сверточных нейронных сетей.

Полносвязный слой. Пусть у группы искусственных нейронов, описываемых формулой (1), будут общие входы. Такая группа искусственных нейронов называется полносвязным слоем. Значения на выходе полносвязного слоя будут описываться следующей формулой:

$$y_i = \varphi\left(\sum_j w_{ij} x_j + b_i\right), \quad (2)$$

где y_i – значение на i-том выходе полносвязного слоя;

φ – нелинейная функция, называемая функцией активации слоя;

w_{ij} – весовой коэффициент j-го входа i-го нейрона слоя;

x_j – сигнал на j-том входе слоя;

b_i – параметр смещения i-го нейрона слоя.

Нетрудно заметить, что формула (2), описывающая полносвязный слой, может быть записана в следующей эквивалентной форме с применением матричных операций:

$$\boldsymbol{y} = \varphi(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}), \quad (3)$$

где $\boldsymbol{y}$ – вектор значений на выходе слоя (выходных значений);

φ – нелинейная скалярная функция скалярного аргумента, называемая функцией активации слоя;

$\boldsymbol{W}$ – матрица весовых коэффициентов слоя;

$\boldsymbol{x}$ – вектор значений на входе слоя (входных значений);

$\boldsymbol{b}$ – вектор смещения слоя.

Здесь и далее запись вида $\boldsymbol{a} = \varphi(\boldsymbol{b})$ означает, что i-я компонента a_i вектора $\boldsymbol{a}$ равна значению функции φ, аргументом которой является i-я компонента b_i вектора $\boldsymbol{b}$. Эта запись аналогичным образом обобщается на случай матриц.

Матрица весовых коэффициентов слоя $\boldsymbol{W}$ и вектор смещения слоя $\boldsymbol{b}$ вместе называются параметрами полносвязного слоя.

Из формулы (3) следует, что вычисление полносвязного слоя нейронной сети сводится к операциям умножения матрицы на вектор, сложения двух векторов, поэлементному взятию нелинейной функции.

Сверточный слой. Для решения задач компьютерного зрения, распознавания речи и обработки иных сигналов с большим успехом применяют сверточные нейронные сети, главным элементом которых являются сверточные слои, описываемые сверткой матриц:

$$\boldsymbol{Y} = \varphi(\boldsymbol{X} * \boldsymbol{K} + \boldsymbol{B}), \tag{4}$$

где $\boldsymbol{Y}$ – матрица на выходе слоя;

φ – нелинейная скалярная функция скалярного аргумента, называемая функцией активации слоя;

$\boldsymbol{X}$ – матрица на входе слоя;

$\boldsymbol{K}$ – матрица, называемая ядром свертки;

$\boldsymbol{B}$ – матрица смещения слоя.

Ядро свертки $\boldsymbol{K}$ и матрица смещения $\boldsymbol{B}$ вместе называются параметрами сверточного слоя.

Из формулы (4) видно, что сверточный слой представляет собой композицию свертки двух матриц, сложения матриц и вычисления значения нелинейной функции для всех элементов результирующей матрицы.

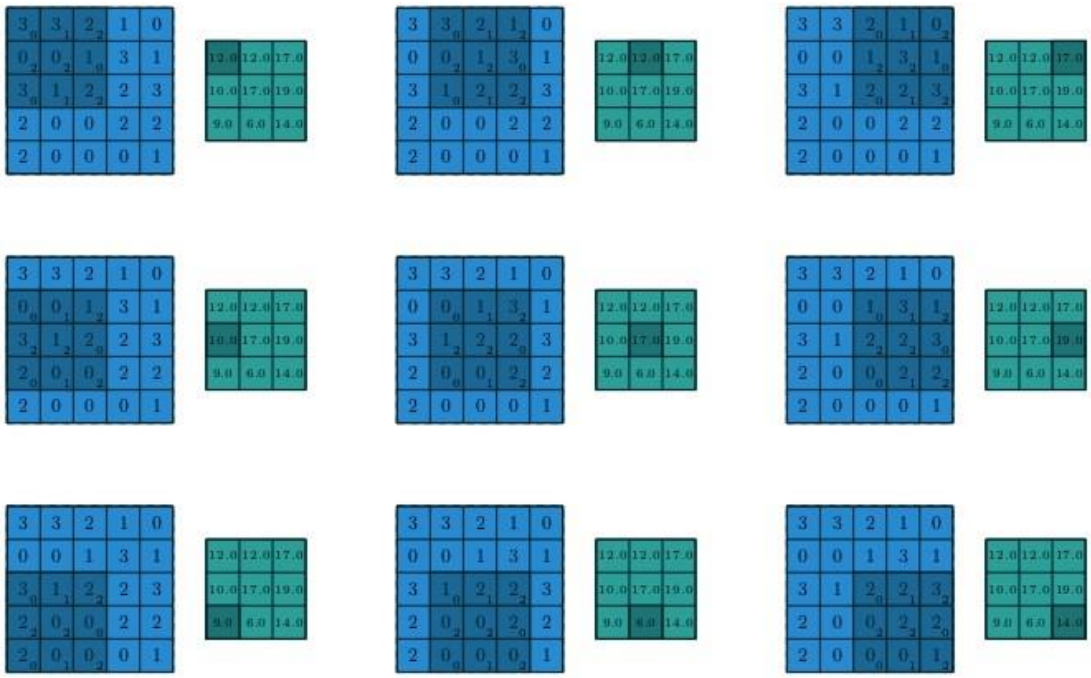


Рисунок 1 – Свертка матриц [9]

Пусть матрица $\boldsymbol{A}$ имеет размер (n, m), а матрица $\boldsymbol{B}$ – размер (k, l). Результатом свертки матрицы $\boldsymbol{A}$ с матрицей (ядром) $\boldsymbol{B}$ будет являться матрица $\boldsymbol{C}$ размером $(n - k + 1, m - l + 1)$, элементы которой могут быть вычислены по следующей формуле [4]:

$$C_{i,j} = \sum_{p=0}^{k-1} \sum_{q=0}^{l-1} A_{i+p,j+q} B_{p,q}. \quad (5)$$

Процесс свертки матриц, описываемый указанной выше формулой, продемонстрирован на рисунке 1. Синим цветом обозначена входная матрица, зеленым – результат свертки. Элементы матрицы, участвующие в вычислениях на очередной итерации, затемнены. Элементы ядра свертки на каждой итерации подписаны снизу справа от соответствующих элементов входной матрицы.

Таким образом, и полносвязный слой, и сверточный слой сводятся к матричным операциям. Известно, что операции над матрицами, включая умножение матриц и свертку матриц, а также поэлементное взятие

произвольных функций, хорошо распараллеливаются. Данная особенность матричных операций может быть использована при разработке специализированных аппаратных реализаций искусственных нейронных сетей и их отдельных слоев в частности.

Функция активации. Можно заметить, что в формулах (3) и (4) в качестве аргумента функции активации выступает результат линейного преобразования, выполняемого над сигналом, поступающим на вход слоя. Очевидно, что если бы функция активации была линейной, то и весь слой являлся бы линейной функцией. Тогда и вся нейронная сеть – композиция нескольких слоев – также являлась бы линейной функцией и не могла бы аппроксимировать более сложные функциональные зависимости. Данная проблема может быть решена выбором в качестве функции активации нелинейной функции.

Исторически в качестве функции активации повсеместно использовалась пороговая функция (известная в физике как функция Хевисайда), этот выбор был обусловлен аналогией с процессами, протекающими в биологических нейронах.

Позже пороговую функцию сменила ее непрерывная разновидность – логистическая функция (сигмоида):

$$\mathrm{Sigmoid}(x) = \frac{1}{1+e^{-x}}\ . \qquad (6)$$

Необходимость использования сигмоиды вместо пороговой функции обусловлена основным методом обучения искусственных нейронных сетей – стохастическим градиентным спуском – который требует, чтобы все преобразования в составе нейронной сети были дифференцируемы.

Позднее было показано, что использование более простой нелинейной функции ReLU облегчает процесс обучения искусственных нейронных сетей, а также ускоряет вычисления за счет отсутствия необходимости вычисления экспоненты [10]. Указанная функция задается следующей формулой:

$$\mathrm{ReLU}(x) = \begin{cases} 0, x < 0, \\ x, x \geq 0. \end{cases} \quad (7)$$

Отметим, что в глубоких архитектурах нейронных сетей, насчитывающих большое количество слоев и использующих функцию ReLU, процесс обучения нейронных сетей затрудняется в связи с затуханием градиентов в области отрицательных значений аргумента функции. В таких случаях используют функцию LeakyReLU:

$$\mathrm{LeakyReLU}(x) = \begin{cases} ax, x < 0, \\ x, x \geq 0, \end{cases} \quad (8)$$

где $a \in (0, 1)$ – вещественный параметр, характеризующий наклон прямой в области отрицательных значений аргумента функции.

Однако аппаратная реализация такой функции активации потребует использования дополнительного умножителя, что, в зависимости от конкретного подхода к реализации слоев нейронных сетей, может значительно увеличить использование ресурсов программируемых логических интегральных схем (ПЛИС), на которых будет происходить работа аппаратной реализации. С учетом указанного факта, в настоящей работе в качестве функции активации будет использована функция ReLU, входящая в большинство современных архитектур сверточных и полносвязных нейронных сетей.

1.2 Применение искусственных нейронных сетей

Начало практике активного применения искусственных нейронных сетей для решения задач компьютерного зрения положила работа [1], описывающая архитектуру глубокой сверточной нейронной сети AlexNet. Работа показала, что сверточные нейронные сети решают задачу определения

классов, к которым принадлежат изображений лучше, чем классические алгоритмы. С тех пор было предложено много новых архитектур, основанных на свертках, имеющих большую точность [6] [5] [2].

Помимо решения задачи классификации изображений, сверточные нейронные сети также используются для сегментации медицинских изображений, например, с целью полуавтоматического или автоматического нахождения туберкулеза, новообразований, метастазов.

С помощью сверточных нейронных сетей осуществляют распознавание объектов беспилотные автомобили, например, автомобили компании Tesla.

Рендеринг изображений в высоком разрешении является вычислительно сложной задачей для современных графических ускорителей в силу конечного объема видеопамяти и конечного числа графических ядер. В случае, когда нужно получить изображение высокого разрешения, компания Nvidia предлагает сначала осуществить рендеринг изображения в низком разрешении, а затем увеличить его с помощью сверточных нейронных сетей. Такая технология получила название deep learning super sampling (DLSS) и позволяет увеличить число кадров в секунду (FPS) в видеоиграх, запускаемых на графических ускорителях компании [11].

Некоторые из указанных выше задач предъявляют жесткие требования к энергопотреблению, производительности и скорости отклика системы, выполняющей вычисления искусственных нейронных сетей. В таком случае, вычисления искусственных нейронных сетей могут быть перенесены на более быстрые вычислительные устройства. Настоящая работа посвящена разработке аппаратной реализации полносвязного и сверточного слоев на ПЛИС, которая в будущем может быть использована при разработке производительного вычислительного устройства.

Для решения указанных выше задач применяются, как правило, сверточные нейронные сети, в связи с чем к ускорению сверточных нейронных сетей наблюдается повышенный интерес. Однако недавно авторами полносвязной нейронной сети MLP-Mixer было показано, что

полносвязные нейронные сети также можно успешно использовать при решении задач компьютерного зрения, что повышает актуальность и важность реализации полносвязного слоя нейронных сетей на ПЛИС, проводимой в данной работе.

1.3 Вычисление искусственных нейронных сетей

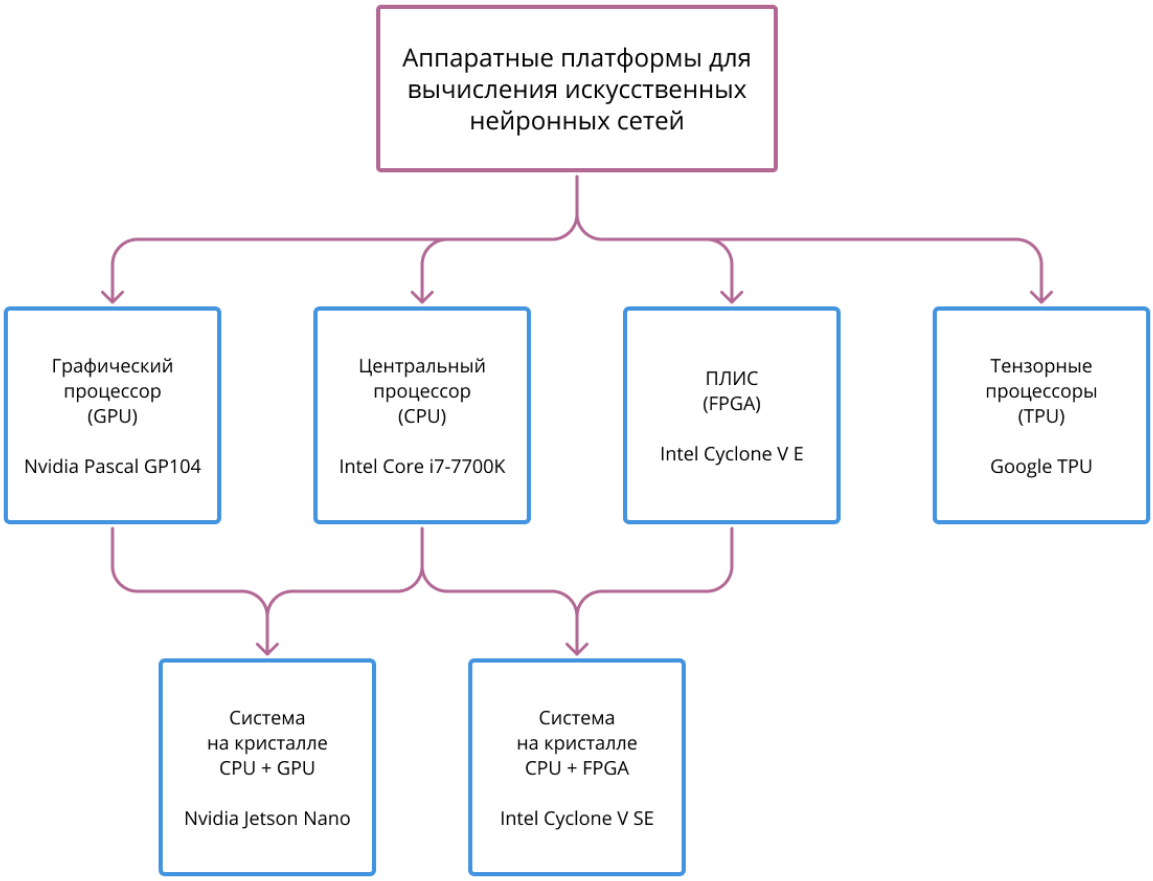


Рисунок 2 – Аппаратные платформы для вычисления искусственных нейронных сетей

В современном глубоком обучении искусственные нейронные сети представляются в виде композиции операций над матрицами и многомерными массивами: умножений, сверток, поэлементных умножений, вычисления нелинейных функций и так далее. Для работы с искусственными нейронными сетями созданы программные пакеты, в частности, PyTorch и TensorFlow, которые позволяют исследователю не заниматься реализацией

указанных операций вручную, а лишь составлять нейронные сети из реализованных авторами программных пакетов слоев нейронных сетей. В свою очередь, эти программные пакеты опираются на низкоуровневые программные реализации матричных операций и отдельных слоев нейронных сетей, например, на LAPACK или cuDNN. В частности, библиотека LAPACK используется для произведения требуемых вычислений на центральных процессорах (CPU), а библиотека cuDNN – для вычислений на графических процессорах и графических ускорителях (GPU).

Первоначально, для вычисления искусственных нейронных сетей использовались центральные процессоры. Однако число потоков, в которых можно параллельно выполнять вычисления, у центральных процессоров невелико, и, за редкими исключениями, на момент написания работы не превышает 128. Операции над большими матрицами и многомерными массивами, как правило, могут выполняться в большем числе потоков. Поэтому для вычисления искусственных нейронных сетей стали также применять графические процессоры и графические ускорители. Например, процессор, установленный в графическом ускорителе Nvidia GeForce GTX 1080, содержит 2560 графических ядер (что эквивалентно 2560 потокам), и способен выполнять операции над большими матрицами существенно быстрее.

Вскоре обнаружилось, что использование схем специального назначения может снизить энергозатратность и увеличить скорость вычислений искусственных нейронных сетей, что особенно важно для проведения масштабных исследований в области машинного обучения. Компанией Google были разработаны тензорные процессоры Google TPU, предназначенные специально для ускорения вычисления искусственных нейронных сетей. Другие компании, например, IBM, также ведут работу над собственными микросхемами специального назначения, однако, их разработки на данный момент не доступны для широкого применения.

Разработка схем специального назначения является трудоемкой и дорогостоящей задачей, которую могут позволить себе только крупные компании. Однако, подобную гибкость в организации вычислений могут предоставить программируемые логические интегральные схемы (ПЛИС) – цифровые интегральные схемы, логика работы которых задается не на этапе изготовления, а путем программирования (загрузки встраиваемого программного обеспечения).

1.4 Оптимизация вычисления искусственных нейронных сетей

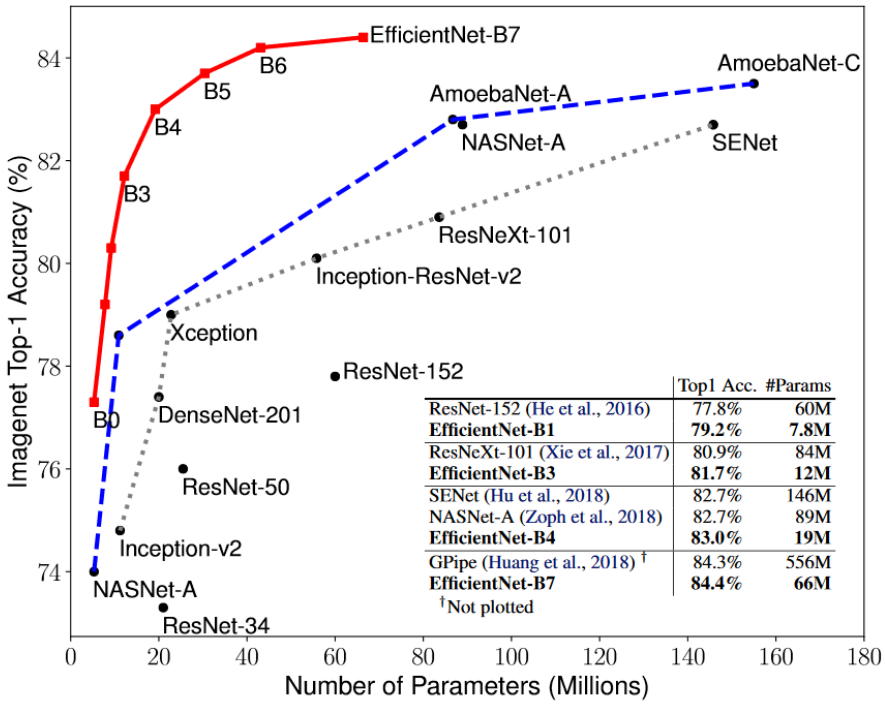


Рисунок 3 – Зависимость точности различных архитектур сверточных нейронных сетей от числа параметров [2]

Вычисление глубоких нейронных сетей является сложной задачей: суммарное число параметров даже у самых эффективных современных архитектур (в частности, у EfficientNet), как продемонстрировано на рисунке 3, очень велико. У сверточной нейронной сети EfficientNet-B1 насчитывается 7,8 миллионов параметров, представленных в виде 32-битных чисел с плавающей запятой. Значительный процент от этого числа приходится на

вещественные параметры сверточных слоев (а именно ядер сверток). Операции над числами с плавающей запятой реализуются (и на программном, и на аппаратном уровне) сложнее и медленнее, чем операции над целыми числами. Оказалось, что возможно квантовать параметры произвольной искусственной нейронной сети без значительного уменьшения целевой метрики решаемой задачи. Квантование параметров сети позволяет снизить время вычисления выходного тензора за счет меньших требований к пропускной способности шин и меньшего времени выполнения целочисленных операций. Квантование параметров также позволяет использовать меньший объем памяти (как внешней, так и расположенной внутри самого вычислительного устройства) для хранения весов нейронной сети и для хранения промежуточных результатов различных вычислений [4].

Из всех возможных преобразований из 32-битных чисел с плавающей запятой (далее float32) в 8-битные целые числа (далее int8), как правило, выбирают следующие:

$$x_{int8} = \text{clip}\left(\left[\frac{x_{float32}}{s}\right] + z, -127, 128\right), \quad (9)$$

$$\text{clip}(x, a, b) = \min(\max(x, a), b), \quad (10)$$

где s – вещественный масштабирующий параметр (scale);
z – целочисленный параметр смещения (zero point, bias).

В зависимости от алгоритма, применяемого для квантования весов нейронной сети, параметры квантования вычисляются либо на этапе обучения нейронной сети (quantization-aware training), либо на этапе калибровки сети с квантованными весами [4].

Квантование весов часто применяется перед выводом искусственных нейронных сетей в режим эксплуатации, в особенности если планируется использовать для вычисления сети процессоры ARM, которые часто встречаются на мобильных устройствах. Наиболее популярные программные платформы глубокого обучения, такие как PyTorch и TensorFlow, позволяют

произвести квантование весов произвольной искусственной нейронной сети без значительной потери точности.

Резюмируя вышеизложенное, искусственные нейронные сети, в частности полносвязные и сверточные архитектуры, востребованы в задачах компьютерного зрения, требовательных к скорости выполнения вычислений. Основными функциональными элементами полносвязных и сверточных нейронных сетей являются соответственно полносвязные и сверточные слои. Такие слои сводятся к композициям хорошо распараллеливаемых матричных операций. Разработке аппаратной реализации полносвязных и сверточных слоев, учитывающих и применяющих эту особенность, посвящена следующая глава настоящей работы. Существуют работы [12], в которых для реквантования результатов арифметических операций с целыми числами используются битовые сдвиги:

$$x_{int8} = x_{int16} \gg 8, \tag{11}$$

где x_{int8} – результат умножения или сложения после реквантования;

x_{int16} – результат умножения или сложения.

Отметим, что реквантование, выполненное по формуле (11), является частным случаем реквантования по формуле (9) при $s = 2^8 = 256,\ z = 0$, однако реализация битовых сдвигов на ПЛИС не требует использования DSP блоков.

2 Аппаратная реализация искусственных нейронных сетей

2.1 Особенности аппаратной реализации вычислительных устройств

В ряде принятых в настоящий момент подходов нейронные сети создаются и обучаются с использованием языков программирования, таких как Python, C++, C. Эти языки программирования позволяют задействовать центральные процессоры, графические процессоры и некоторые более экзотические вычислительные устройства (например, тензорные процессоры) с максимально достижимой эффективностью для проведения необходимых вычислений.

Настоящая работа, однако, ставит своей целью разработку аппаратной реализации слоев искусственных нейронных сетей. Для достижения заявленной цели указанные ранее языки программирования не подходят, так как позволяют лишь описывать алгоритм действий для существующих, заранее разработанных вычислительных устройств (центральных процессоров, графических ускорителей и так далее), но не задавать и не менять их внутреннее устройство. Описывать же структуру новых вычислительных устройств позволяют языки описания аппаратуры (hardware description languages, HDL), например, VHDL, Verilog, SystemVerilog.

Описание структуры вычислительных устройств, выполненное на языках описания аппаратуры, может в дальнейшем использоваться для синтеза и верификации цифровых электронных схем. В частности, по такому описанию может быть выполнен синтез встроенного программного обеспечения (прошивки) для программируемых логических интегральных схем (ПЛИС) – интегральных схем, предназначенных для настройки (программирования) пользователем после изготовления.

Использование реконфигурируемых ПЛИС позволяет проверять гипотезы, возникающие в ходе разработки новых вычислительных устройств, минуя этап разработки и изготовления прототипов интегральных схем. В

связи с этим, этапы настоящей работы, связанные непосредственно с разработкой аппаратных реализаций, будут выполнены на языке описания аппаратуры SystemVerilog. По разработанным описаниям аппаратных реализаций с помощью программного обеспечения Intel Quartus будет синтезировано соответствующее встроенное программное обеспечение, которое будет использовано для программирования ПЛИС для последующего проведения сравнительных экспериментов.

2.2 Квантование параметров

Как было показано в параграфе 1.4, квантование параметров нейронной сети позволяет повысить скорость вычислений и понизить требования к пропускной способности шин вычислительных устройств. Эти утверждения справедливы в том числе и для программируемых логических интегральных схем (ПЛИС), потому предложенные далее аппаратные реализации слоев нейронных сетей будут оперировать 8-битными целыми числами.

Отметим, что результат умножения двух 8-битных целых чисел имеют большую разрядность: в общем случае, для его хранения потребуется 16 бит. Для хранения результата умножения двух 16-битных чисел потребуется уже 32 бита. Объем памяти, требуемой для хранения точного результата операции, будет расти экспоненциально. Схожие рассуждения справедливы и для сложения. Очевидно, что при решении реальных задач объем доступной памяти ограничен. Требуется найти способ понижать разрядность результатов операций над целыми числами: провести реквантование.

В существующих программных реализациях искусственных нейронных сетей, для реквантования используются преобразования, аналогичные формуле (9), в которых результаты операций, имеющие большую разрядность, сначала приводятся к типу с плавающей запятой, а затем повторно квантуются. Такие преобразования приемлемы для существующих вычислительных устройств, ядра которых способны быстро производить

операции с вещественными числами. Нужно отметить, что DSP блоки ПЛИС также позволяют выполнять арифметические операции с числами с плавающей запятой, однако отказ от таких операций в пользу целочисленных позволит увеличить рабочую частоту и снизить потребление ресурсов ПЛИС, а освобожденные в результате такой оптимизации ресурсы использовать для синтеза большего числа вычислительных элементов.

2.3 Реализация полносвязного слоя

Как было показано в параграфе 1.1, вычисление результата полносвязного слоя требует умножения матрицы весов слоя на входной вектор с последующим прибавлением вектора смещения. Рассмотрим существующие публикации, описывающие способы организации умножения матриц или умножения матрицы на вектор на ПЛИС.

Работа [12] описывает способ умножения матрицы на вектор, при котором за один такт работы вычислительного блока умножение затрагивает лишь один элемент матрицы. Таким образом умножение всей матрицы на вектор займет $O(n \cdot m)$ тактов, где n и m – размеры матрицы. Подобный подход к реализации умножения матрицы на вектор не использует возможности ПЛИС к распараллеливанию вычислений. Работа [13] следует схожей идеологии, однако позволяет конвейеризовать вычисления.

Работа [14], в свою очередь, предлагает умножать матрицу на вектор, подавая каждый такт одновременно все элементы вектора и очередной строки (столбца) матрицы на вычислительные элементы схемы, выполняющие умножение с накоплением (multiply and accumulate, MAC). Такая схема позволяет полностью задействовать вычислительные ресурсы ПЛИС, выполняя умножение матрицы размером (n, m) на вектор длиной n за $O(m)$ тактов. Однако такая схема требовательна к пропускной способности шин, через которые осуществляется подача одновременно n элементов

вектора и матрицы на вычислительные элементы, и никак не использует факт физической локальности вычислительных элементов.

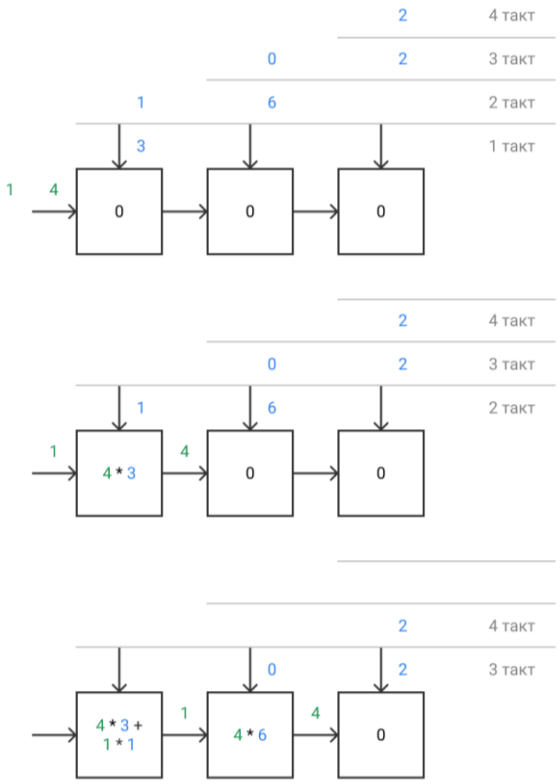


Рисунок 4 – Первые такты процесса умножения матрицы на вектор на систолической структуре

В настоящей работе предлагается использовать двумерную систолическую структуру из вычислительных элементов, осуществляющих умножение с накоплением. Первые такты процесса работы такой систолической структуры продемонстрированы на рисунке 4. Зеленым обозначены компоненты вектора, синим – элементы матрицы. В конце процесса результат умножения матрицы на вектор извлекается из вычислительных элементов структуры. Кроме того, инициализацией регистров, используемых для хранения результатов вычисления, можно одновременно выполнить сложение результата с вектором смещения полносвязного слоя.

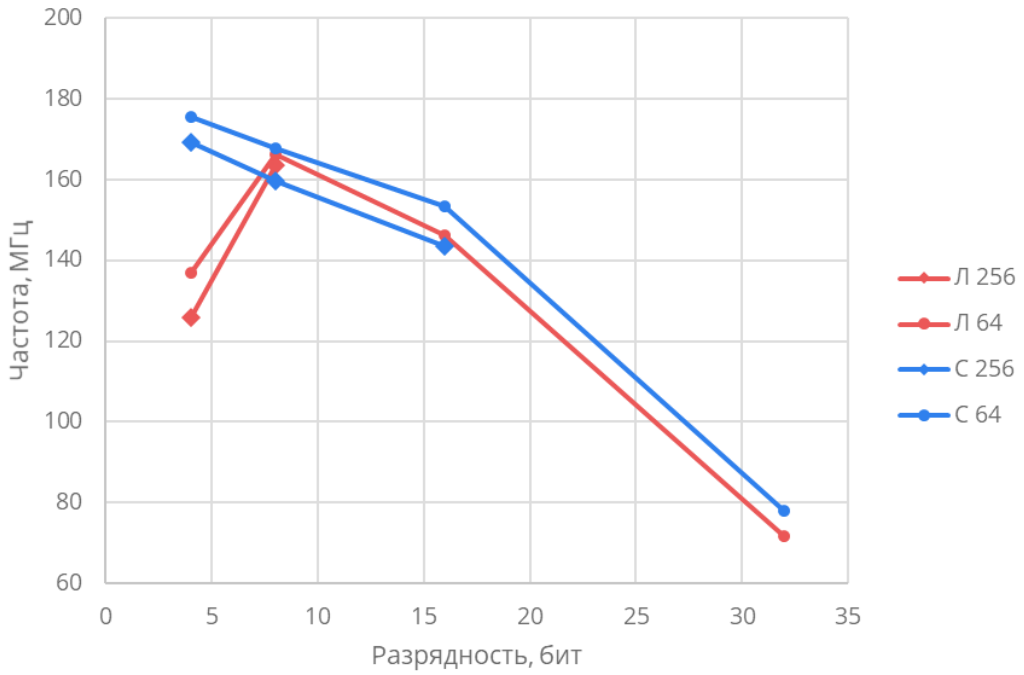


Рисунок 5 – Сравнение рабочих частот предложенного подхода (С) с существующим (Л)

Сравним производительность предложенного подхода с собственной реализацией подхода, предложенного в работе [14], по рабочим частотам. Чем выше рабочая частота устройства, тем больше операций в единицу времени оно способно осуществить. Результаты сравнения двух подходов к умножению матрицы на вектор для ПЛИС Intel Cyclone 10CL120ZF780I8G продемонстрированы на рисунке 5 и в таблицах 1 и 2. Видно, что производительность предложенного подхода, основанного на систолической структуре, сравнима или превосходит производительность подхода, основанного на параллельном вычислении скалярных произведений вектора и столбцов (строк) матрицы. Такой результат следует связывать с особенностями внутренней структуры ПЛИС и с тем, как организована передача данных между логическими блоками по внутренним шинам. Кроме того, значительным аргументом в сторону выбора умножителя, основанного на систолических структурах, является его крайняя эффективность при реализации в составе интегральной схемы специального назначения (ASIC), обеспечиваемая физической близостью элементов систолической структуры

на кристалле. Разработанное описание умножителя на языке описания аппаратуры в будущем может быть использовано для синтеза такой схемы. Исходя из этого, для дальнейшей работы по созданию аппаратной реализации полносвязного слоя искусственной нейронной сети был выбран умножитель матрицы на вектор, основанный на систолических структурах.

Таблица 1 – Результаты тестирования подхода, основанного на параллельном вычислении скалярных произведений

Ширина и высота матрицы, элементы	Разрядность чисел, биты	Частота, МГц		Потребление, штуки		
		При 40 °C	При 100 °C	Логические ячейки	DSP 18x18	DSP 9x9
512	4	175,93	167,81	48076	0	0
	8	168,78	158,30	54434	0	512
256	4	177,43	169,12	23623	0	0
	8	167,11	159,59	26794	0	256
	16	150,63	143,41	42920	256	0
128	4	182,72	175,07	11756	0	0
	8	177,75	162,84	13201	0	128
	16	169,98	154,30	21447	256	0
64	4	183,12	175,62	5886	0	0
	8	183,96	167,73	6540	0	64
	16	169,15	153,37	10793	64	0
	32	85,85	78,02	23552	256	0

Таблица 2 – Результаты тестирования подхода, основанного на систолических структурах

Ширина и высота матрицы, элементы	Разрядность чисел, биты	Частота, МГц		Потребление, штуки		
		При -40 °C	При 100 °C	Логические ячейки	DSP 18x18	DSP 9x9
512	4	134,19	126,76	44354	0	0
	8	165,54	154,70	46753	0	512
256	4	131,86	125,80	22033	0	0
	8	173,82	163,53	23201	0	256
128	4	145,16	137,46	10910	0	0
	8	179,05	163,75	11523	0	128
64	4	144,78	136,93	5421	0	0
	8	182,15	166,39	5857	0	64

Следует отметить, что в случае, если один из размеров матрицы превышает размер умножителя, матрица все еще может быть умножена на вектор. Для этого достаточно разбить ее на блоки, не превышающие размера умножителя (количества умножающих элементов) вдоль большего измерения, умножить поэлементно на вектор, а затем конкатенировать результат. Таким образом может быть реализован умножитель, оперирующий с числами с разрядностью 32 бита.

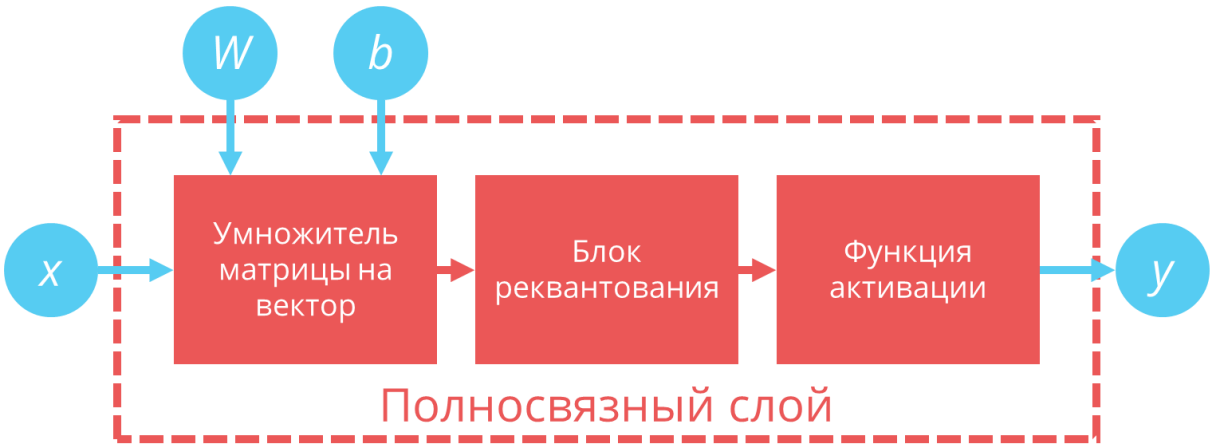


Рисунок 6 – Схема полносвязного слоя искусственной нейронной сети

Как видно из рисунка 6, помимо разработанного умножителя матрицы на вектор, необходимо также реализовать блок реквантования (для снижения размерности результата) и функцию активации.

Блок реквантования был реализован с помощью битового сдвига, как в формуле (11). В ПЛИС операция такого битового сдвига выполняется без использования вычислительных элементов, только путем подключения шины данных к регистрам, в которых хранятся 8 старших битов целого числа.

В качестве функции активации полносвязного слоя была выбрана функция ReLU, определяемая формулой (7). Реализация такой функции активации включает в себя проверку старшего бита целого числа, который определяет знак числа. В зависимости от значения старшего бита, в выходную шину блока, реализующего функцию активации, подается либо нули, либо биты входного числа.

Композиция перечисленных блоков представляет собой аппаратную реализацию полносвязного слоя нейронной сети.

2.4 Сравнение реализации полносвязного слоя

Результаты измерения производительности предложенной реализации полносвязного слоя искусственной нейронной сети для ПЛИС Intel Cyclone

10 10CL120ZF780I8G, характеристики которой указаны в таблице 3, представлены в таблице 4 и на рисунке 7.

Таблица 3 – Характеристики ПЛИС Intel Cyclone 10 10CL120ZF780I8G

Число логических элементов	120000
Блоки DSP	288
Тип DSP блоков	Умножители 18 x 18
Внутренняя память	3,888 Мбит

Таблица 4 – Результаты измерения производительности предложенной реализации полносвязного слоя искусственной нейронной сети

Разрядность чисел	Время вычисления для матрицы с заданной шириной и высотой, мкс			
	512	256	128	64
4	9,14	4,53	2,18	1,08
8	9,69	4,80	2,35	1,14
16	13,22	6,61	3,31	1,65
32	26,04	13,02	6,51	3,26

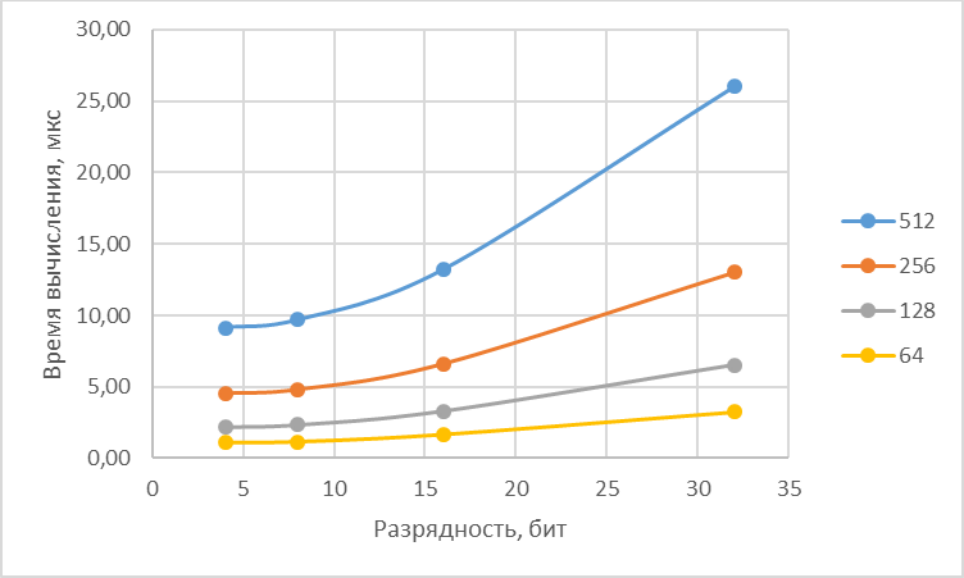


Рисунок 7 – Время вычисления полносвязного слоя нейронной сети в зависимости от разрядности используемых чисел и ширины и высоты матрицы весов

На протяжении экспериментов прогнозируемая потребляемая мощность кристалла не превышала 1 Вт, что указывает на применимость разработанного решения во встраиваемых системах. Максимальная разрядность входных данных составила 32 бита, а максимальное количество одновременно вычисляемых нейронов превысила 256.

Сравним теперь производительность разработанной аппаратной реализации полносвязного слоя искусственной нейронной сети на ПЛИС с производительностью широко распространенной программной реализации нейронных сетей – программным пакетом (фреймворком) PyTorch. Для этого замерим время вычисления полносвязных слоев одинаковой конфигурации на ПЛИС с применением разработанного подхода, на процессоре Intel Core i7-7700K и на графическом ускорителе Nvidia GeForce GTX 1080, характеристики которых представлены в таблице 5. Последние два устройства были выбраны для сравнения, так как используются в практике исследователей, занимающихся машинным обучением и работающих с искусственными нейронными сетями.

Таблица 5 – Характеристики центрального процессора и графического ускорителя, использованных в сравнительных экспериментах

	Intel Core i7-7700K	Nvidia GeForce GTX 1080
Частота работы, ГГц	4,20	1,7
Тип используемой памяти	DDR4	GDDR5X
Размер используемой памяти, ГБ	16	8
Число ядер	4	2560
Тип используемых ядер	Kaby Lake	CUDA
Расчетная мощность, Вт	91	180

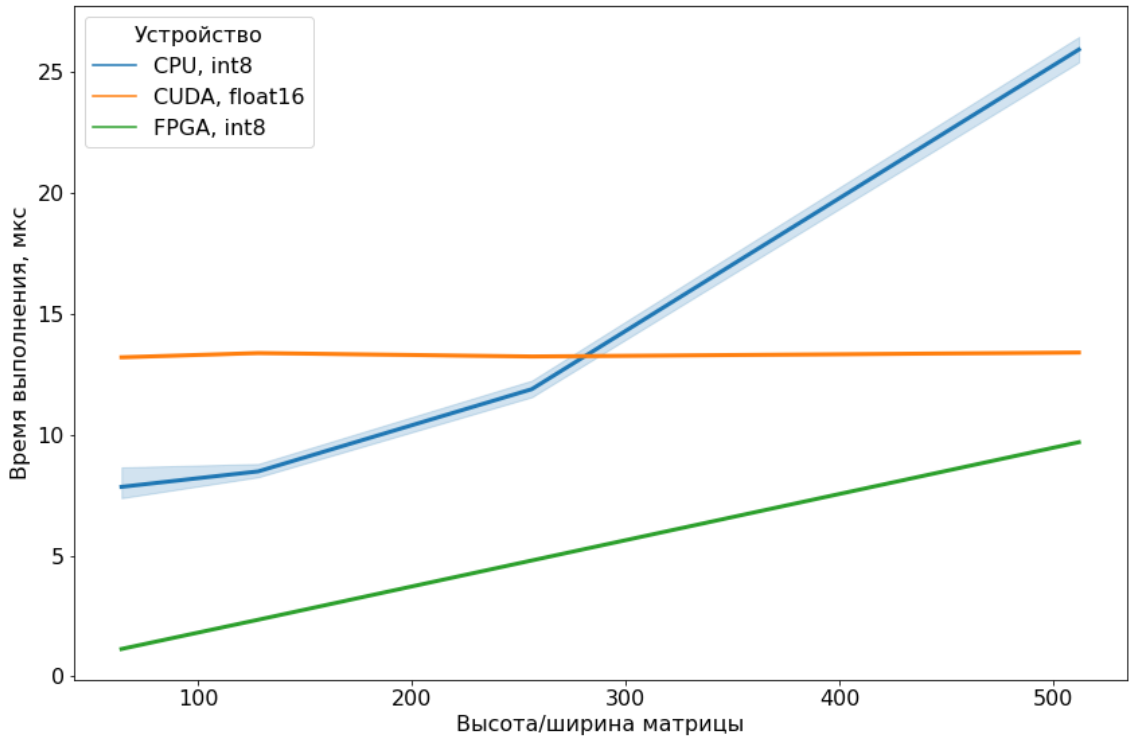


Рисунок 8 – Сравнение производительности разработанной аппаратной реализации полносвязного слоя (зеленая линия) с программной реализацией, выполняемой на центральном процессоре Intel Core i7-7700K (синяя линия) и на графическом ускорителе Nvidia GeForce GTX 1080 (оранжевая линия)

Результаты сравнения производительностей отражены на рисунке 8. Разработанная аппаратная реализация на ПЛИС работает заметно быстрее широко используемых программных реализаций на центральных процессорах и графических ускорителях для матриц весов, не превышающих по высоте и ширине 512. Отметим, что на графике также отложен доверительный интервал, в который попадают 95% проведенных измерений. Доверительный интервал хорошо заметен у измерений, проведенных на центральном процессоре, так как в отличие от графического ускорителя, центральный процессор также активно используется операционной системой для собственных нужд, отчего часть потоков, в которых происходят вычисления, случайным образом приостанавливаются, внося шумы в измерения длительности выполнения вычислений.

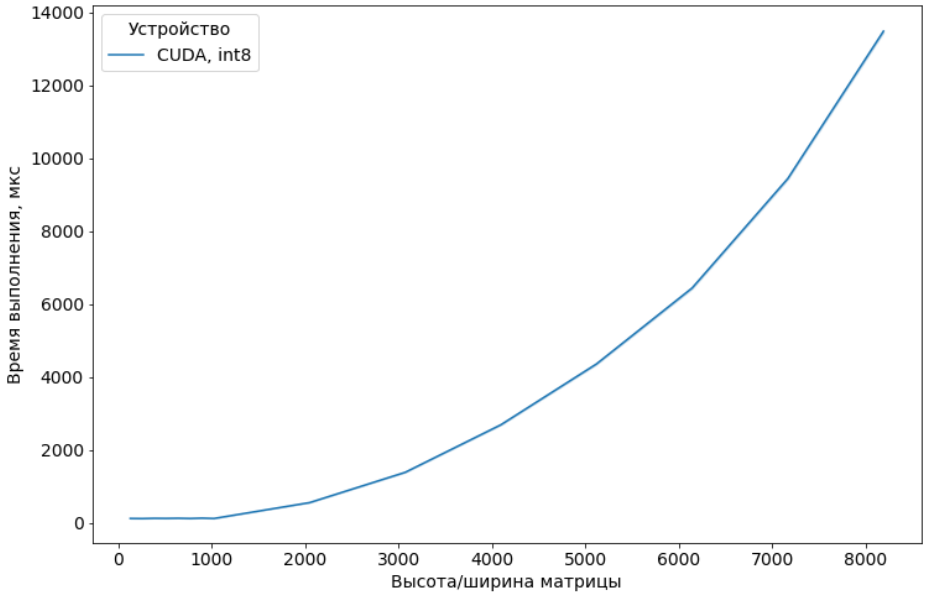


Рисунок 9 – Зависимость времени вычисления на графическом ускорителе полносвязного слоя искусственной нейронной сети от высоты и ширины матрицы весов слоя

Кроме того, время вычисления полносвязного слоя в разработанной аппаратной реализации растет линейно с увеличением высоты и ширины матрицы, в отличие от существующих программных реализаций. Может показаться, что время вычисления полносвязного слоя на графическом ускорителе не зависит от размера матрицы весов. На самом деле, это наблюдение справедливо только для матриц не слишком большого размера, что демонстрирует рисунок 9. Такой эффект связан со сравнительно большим числом ядер (вычислительных элементов) в графическом ускорителе.

Проведем аналогичные сравнения разработанной аппаратной реализацией с той же самой программной реализацией из программного пакета PyTorch, запущенной на системе на кристалле Nvidia Jetson Nano, состоящей из центрального процессора ARM Cortex-A57 и 128 графических ядер Nvidia Maxwell, применяемой во встраиваемых системах.

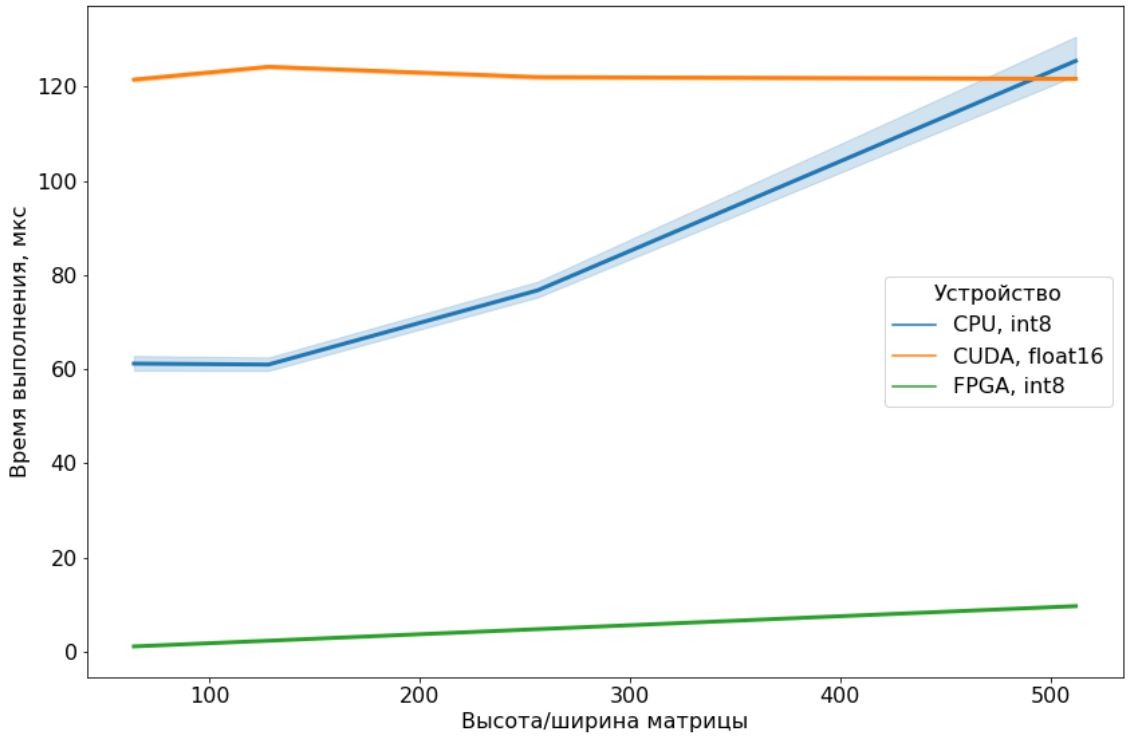


Рисунок 10 – Сравнение производительности разработанной аппаратной реализации полносвязного слоя (зеленая линия) с программной реализацией, выполняемой на центральном процессоре (синяя линия) и на графических ядрах (оранжевая линия) системы на кристалле Jetson Nano

Результаты сравнения разработанной реализации с эталонной во многом напоминают выполненное выше сравнение с полноразмерными центральным процессором и графическим ускорителем, однако, как и следовало ожидать, скорость работы предложенной реализации теперь уже существенно превосходит скорость работы реализации, с которой выполняется сравнение, так как Nvidia Jetson Nano обладает существенно более скромными техническими характеристиками, чем указанные ранее устройства.

2.5 Реализация сверточного слоя

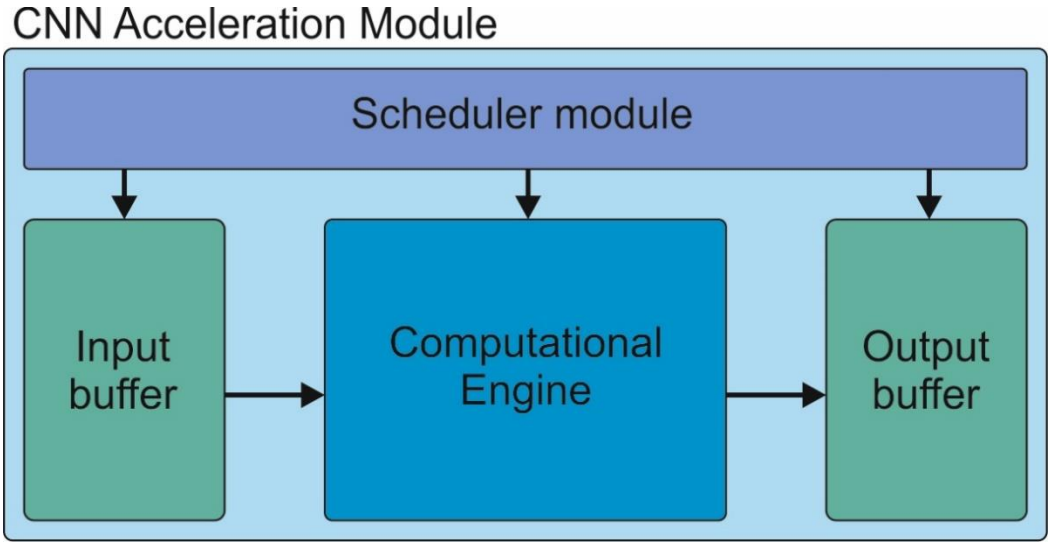


Рисунок 11 – Схема аппаратной реализации сверточного слоя

Как было показано в параграфе 1.1, вычисление сверточного слоя искусственной нейронной сети сводится к последовательному выполнению следующих операций: свертка входной матрицы с ядром, сложение с матрицей смещения, поэлементное взятие функции активации. В настоящей работе предлагается реализовать сверточный слой в виде композиции четырех функциональных блоков, отраженных на рисунке 11.

На рисунке 1, демонстрирующем процесс свертки матриц, видно, что при смещении ядра свертки при вычислении следующего элемента выходной матрицы некоторое число значений может быть буферизовано и переиспользовано на следующем шаге вычисления. Такое решение позволит снизить число обращений к внешней по отношению к ПЛИС памяти.

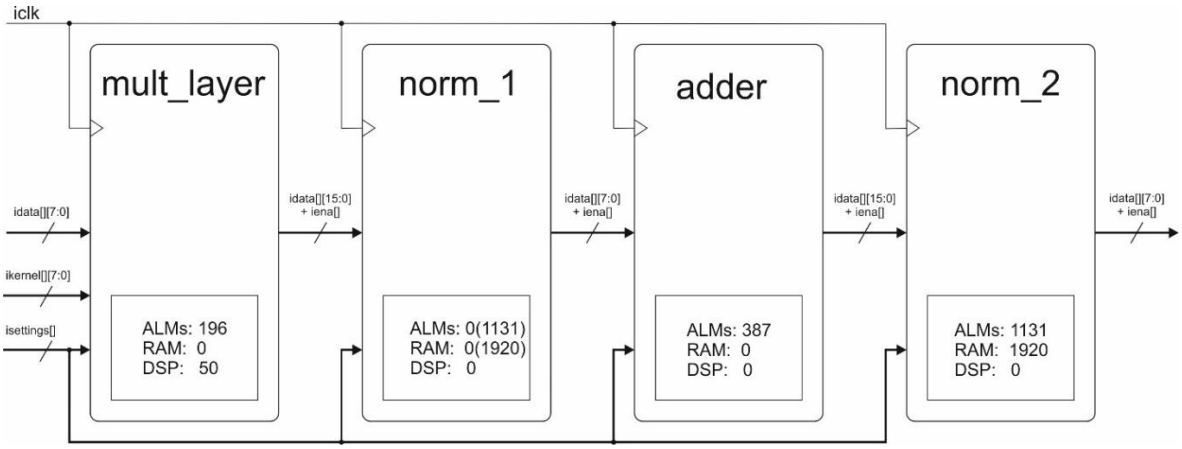


Рисунок 12 – Схема блока свертки матриц

Схема реализации сверточного слоя нейронной сети на ПЛИС представлена на рисунке 12. На схеме изображены четыре модуля: модуль умножения, промежуточный модуль реквантования, модуль сложения и выходной модуль реквантования.

Модуль умножения производит поэлементное умножения входного массива данных на ядро свертки. Результат операции поэлементного умножения представляется массивом (последовательностью) 16-битных целых беззнаковых чисел. Модуль использует DSP блоки, что позволяет эффективно выполнять умножение, не затрачивая логических элементов ПЛИС. DSP блоки присутствуют в большинстве современных ПЛИС.

Промежуточный модуль реквантования может быть использован для понижения разрядности поэлементного произведения, выполненного предыдущим модулем, однако, в большинстве схем (методов) квантования параметров нейронных сетей он опускается с целью сохранения точности промежуточных вычислений. Реквантование производится по формуле (11). Использование такого способа реквантования вместо классического варианта, используемого в программных реализациях, позволяет избежать использования арифметики над числами с плавающей точкой, реализация которой потребовала бы использования большего числа DSP блоков.

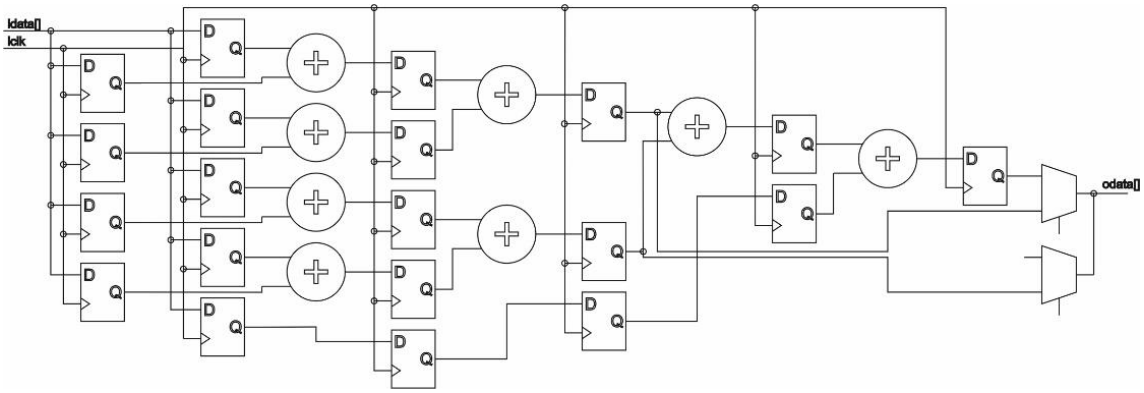


Рисунок 13 – Дерево сумматоров

Модуль сложения представляет собой дерево сумматоров, изображенное на рисунке 13. Временную сложность такой схемы сложения можно оценить как $O(\log(n))$, где n – число элементов ядра (параметров сверточного слоя). Следует отметить, что несмотря на высокую временную эффективность такой схемы суммирования (суммирование с использованием аккумулятора выполняется за время, которое зависит линейно от числа элементов ядра), дерево сумматоров использует большее число логических элементов и DSP блоков ПЛИС.

Так как количество логических элементов и DSP блоков в ПЛИС ограничено, дерево сумматоров может быть спроектировано таким образом, чтобы обеспечить возможность использования его для сверток с ядрами различных размеров, например, для ядер свертки (2, 2) и (3, 3). Схожий подход может быть применен и для иных размеров ядер и позволяет повторно использовать уже синтезированный слой.

Выходной модуль реквантования понижает разрядность элементов выходной матрицы до 8 бит и реализован таким же образом, как и модуль промежуточного реквантования.

Для оценки производительности разработанной реализации сверточного слоя на ПЛИС был использован SoC Intel Cyclone V 5CSEBA6U23I7N. Характеристики данного устройства представлены в таблице 6. Сравнение производилось с помощью программного пакета глубокого обучения PyTorch. Для сравнения свертки одинакового набора

входных матриц и ядер были вычислены на ПЛИС с применением разработанной архитектуры, на процессоре Intel Core i7-7700K и на видеокарте Nvidia GeForce GTX 1080, характеристики которых представлены в таблице 5.

Таблица 6 – Характеристики системы на кристалле Intel Cyclone V 5CSEBA6U23I7N

Число логических элементов	110000
Процессор	2 ядра ARM Cortex-A9
Поддержка DDR	DDR2/DDR3
Внутренняя память	13,59 Мбит

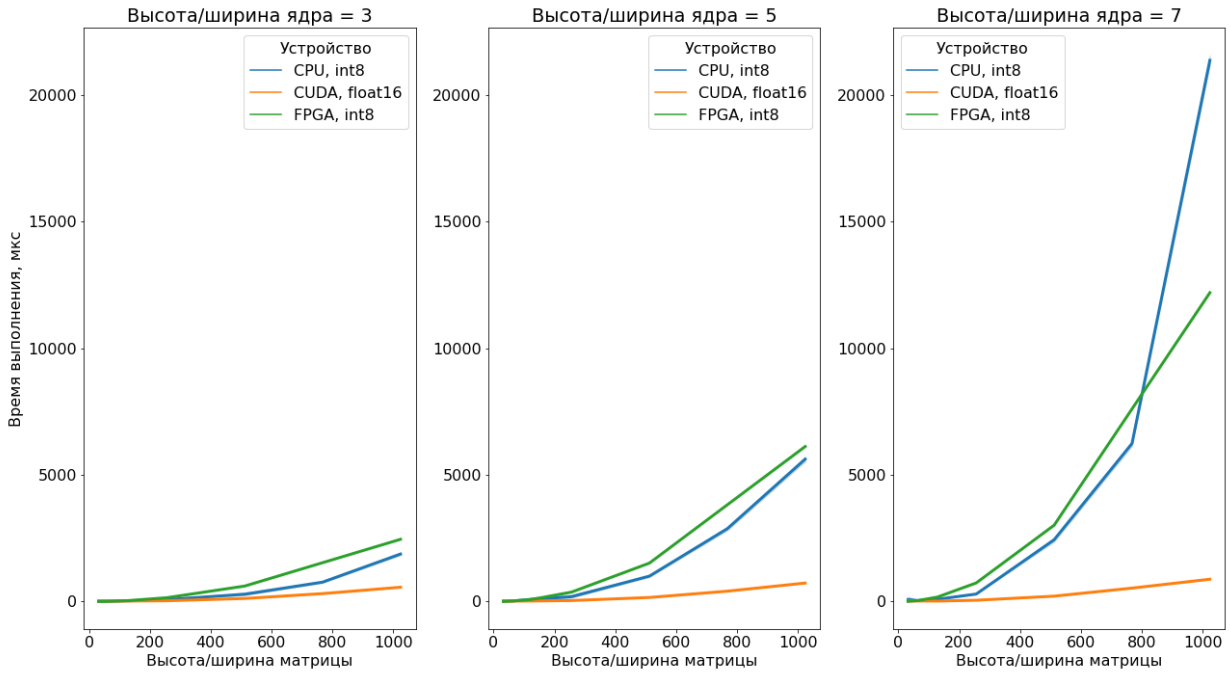


Рисунок 14 – Сравнение производительности разработанной аппаратной реализации сверточного слоя (зеленая линия) с программной реализацией, выполняемой на центральном процессоре Intel Core i7-7700K (синяя линия) и на графическом ускорителе Nvidia GeForce GTX 1080 (оранжевая линия)

На рисунке 14 представлены графики времени вычисления сверток в микросекундах для центрального процессора, графического ускорителя и для

разработанной реализации сверточного слоя на ПЛИС. График демонстрирует, что разработанное решение сравнимо по эффективности с эталонной реализацией на центральных процессорах и графических ускорителях в случае входных матриц малого размера несмотря на существенно более низкую рабочую частоту и потребляемую мощность.

Таким образом, производительность разработанной архитектуры сверточного слоя нейронной сети сопоставима с производительностью решений на базе процессоров общего назначения и графических процессоров. Следует учесть, что разработанное решение позиционируется как часть ускорителя искусственных нейронных сетей для встраиваемых систем. Можно заключить, что использование ПЛИС и разработанной архитектуры позволяет достигнуть приемлемой производительности при существенно меньшем энергопотреблении, характерном для ПЛИС, а потому подходит для встраиваемых систем.

Отметим также, что сравнение разрабатываемой аппаратной реализации с существующим программным аналогом, результаты которого продемонстрированы на рисунке 14, производилось в ходе выполнения настоящей работы неоднократно. В более ранних версиях программного пакета PyTorch вычисление сверточного слоя на идентичном центральном процессоре выполнялось дольше, чем в актуальной версии (1.8.1). Это указывает на то, что программные реализации также продолжают совершенствоваться, а результаты проведенных сравнительных экспериментов в будущем могут потерять свою актуальность.

Выполним также сравнения времени производительности предложенной аппаратной реализации на ПЛИС с программной реализацией на системе на кристалле Nvidia Jetson Nano.

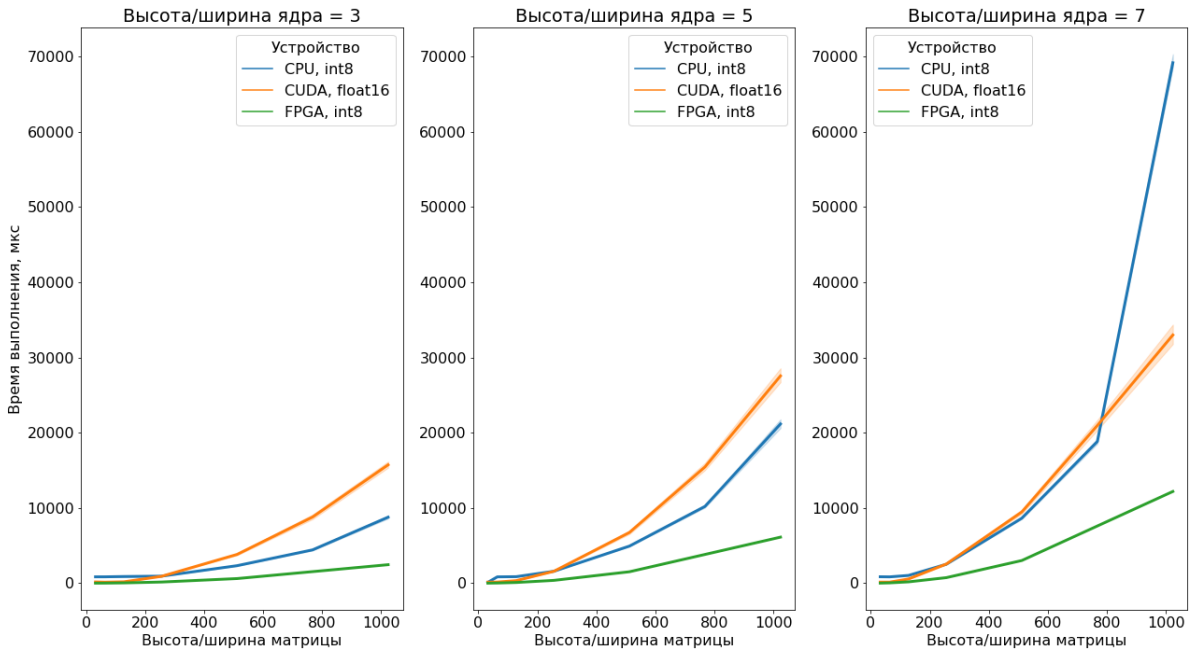


Рисунок 15 – Сравнение производительности аппаратной реализации сверточного слоя (зеленая линия) с программной реализацией, выполняемой на центральном процессоре (синяя линия) и на графических ядрах (оранжевая линия) системы на кристалле Jetson Nano

Результаты сравнения, приведенные на рисунке 15, наглядно показывают, что разработанная аппаратная реализация сверточного слоя на ПЛИС производит вычисления быстрее эталонной программной реализации, запущенной на системе на кристалле Nvidia Jetson Nano, используемой во встраиваемых системах. Этот факт указывает на то, что предложенная реализация также может быть использована во встраиваемых системах, в частности, для решения задач компьютерного зрения с помощью сверточных нейронных сетей.

ЗАКЛЮЧЕНИЕ

В ходе выполнения работы было исследовано устройство полносвязных и сверточных нейронных сетей и способы ускорения их вычисления, изучено квантование параметров искусственных нейронных сетей, разработаны и реализованы полносвязный и сверточный слои искусственных нейронных сетей на базе программируемых логических интегральных схем.

Сравнение производительности разработанного полносвязного слоя с широко используемой программной реализацией из программного пакета PyTorch, запущенной на системе на кристалле Nvidia Jetson Nano, показало, что разработанная аппаратная реализация превосходит на порядок по скорости вычислений указанную систему на кристалле и может использоваться при ускорении искусственных нейронных сетей, применяемых во встраиваемых системах.

Сравнение производительности разработанного сверточного слоя с широко используемой программной реализацией из программного пакета PyTorch, запущенной на системе на кристалле Nvidia Jetson Nano, показало, что разработанная архитектура позволяет достичь сопоставимого времени выполнения свертки для входных матриц, высота и ширина которых не превышает 1024. Данный факт подтверждает наличие потенциала у исследуемого подхода и, с учетом соответствия ПЛИС всем требованиям для использования во встраиваемых системах, возможность его использования для встраиваемых решений.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ И ЛИТЕРАТУРЫ

1. Krizhevsky A., Sutskever I., Hinton G. ImageNet Classification with Deep Convolutional Neural Networks // Proceedings of the 25th International Conference on Neural Information Processing Systems. New York. 2012. Vol. 1. pp. 1097-1105.

2. Tan M., Le Q.V. EfficientNetV2: Smaller Models and Faster Training. 2021.

3. Devlin J., Chang M.W., Lee K., Toutanova K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. 2019.

4. Roth W., Schindler G., Zöhrer M., Pfeifenberger L., Peharz R., Tschiatschek S., Fröning H., Pernkopf F., Ghahramani Z. Resource-Efficient Neural Networks for Embedded Systems. Электронный ресурс.

5. Iandola F., Han S., Moskewicz M., Ashraf K., Dally W., Keutzer K. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and 0.5MB model size. 2016.

6. He K., Zhang X., Ren S., Sun J. Deep Residual Learning for Image Recognition // Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016. pp. 770-778.

7. Hochreiter S., Schmidhuber J. Long Short-term Memory // Neural computation, Vol. 9, Dec 1997. pp. 1735-1780.

8. Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A., Kaiser L., Polosukhin I. Attention Is All You Need // 31st Conference on Neural Information Processing Systems (NIPS 2017). Long Beach, CA. 2017.

9. Dumoulin V., Visin F. A guide to convolution arithmetic for deep learning. 2018.

10. Glorot X., Bordes A., Bengio Y. Deep Sparse Rectifier Neural Networks // Journal of Machine Learning Research. 2010. Vol. 15.

11. Nvidia Turing GPU Architecture. URL: https://images.nvidia.com/aem-dam/en-zz/Solutions/design-visualization/technologies/turing-

architecture/NVIDIA-Turing-Architecture-Whitepaper.pdf (дата обращения: 26.05.2021).

12. Qasim S.M., Telba A.A., AlMazroo A. FPGA Design and Implementation of Matrix Multiplier Architectures for Image and Signal Processing Applications // International Journal of Computer Science and Network Security, Vol. 10, No. 2, Feb 2010. pp. 168-176.

13. Jovanović Ž., Milutinović V. FPGA accelerator for floating-point matrix multiplication // IET Computers & Digital Techniques, Vol. 6, No. 4, Jul 2012. pp. 249-256.

14. Qasim S.M., Abbasi S., AlMashary B. Hardware Realization of Matrix Multiplication using Field Programmable Gate Array // Masaum Journal of Computing, Vol. 1, Aug 2009. pp. 21-25.

15. Sanaullah A., Yang C., Alexeev Y., Yoshii K., Herbordt M. Real-Time Data Analysis for Medical Diagnosis Using FPGA-Accelerated Neural Networks // BMC Bioinformatics, Vol. 19, Dec 2018.

Отчет о проверке на заимствования №1

Автор: Берзин Артем Константинович
Проверяющий: (flerm.steam@ya.ru / ID: 9319938)

Отчет предоставлен сервисом «Антиплагиат» - users.antiplagiat.ru

ИНФОРМАЦИЯ О ДОКУМЕНТЕ

№ документа: 1
Начало загрузки: 21.06.2021 08:45:24
Длительность загрузки: 00:00:01
Имя исходного файла: Берзин А. К. Выпускная квалификационная работа бакалавра.pdf
Название документа: Берзин А. К. Выпускная квалификационная работа бакалавра
Размер текста: 48 кБ
Тип документа: Выпускная квалификационная работа
Символов в тексте: 49387
Слов в тексте: 5720
Число предложений: 278

ИНФОРМАЦИЯ ОБ ОТЧЕТЕ

Начало проверки: 21.06.2021 08:45:26
Длительность проверки: 00:00:18
Корректировка от 21.06.2021 08:53:44
Комментарии: не указано
Модули поиска: Интернет

ЗАИМСТВОВАНИЯ	САМОЦИТИРОВАНИЯ	ЦИТИРОВАНИЯ	ОРИГИНАЛЬНОСТЬ
1,9%	0%	0%	98,1%

Заимствования — доля всех найденных текстовых пересечений, за исключением тех, которые система отнесла к цитированиям, по отношению к общему объему документа.
Самоцитирования — доля фрагментов текста проверяемого документа, совпадающий или почти совпадающий с фрагментом текста источника, автором или соавтором которого является автор проверяемого документа, по отношению к общему объему документа.
Цитирования — доля текстовых пересечений, которые не являются авторскими, но система посчитала их использование корректным, по отношению к общему объему документа. Сюда относятся оформленные по ГОСТу цитаты; общеупотребительные выражения; фрагменты текста, найденные в источниках из коллекций нормативно-правовой документации.
Текстовое пересечение — фрагмент текста проверяемого документа, совпадающий или почти совпадающий с фрагментом текста источника.
Источник — документ, проиндексированный в системе и содержащийся в модуле поиска, по которому проводится проверка.
Оригинальность — доля фрагментов текста проверяемого документа, не обнаруженных ни в одном источнике, по которым шла проверка, по отношению к общему объему документа.
Заимствования, самоцитирования, цитирования и оригинальность являются отдельными показателями и в сумме дают 100%, что соответствует всему тексту проверяемого документа.
Обращаем Ваше внимание, что система находит текстовые пересечения проверяемого документа с проиндексированными в системе текстовыми источниками. При этом система является вспомогательным инструментом, определение корректности и правомерности заимствований или цитирований, а также авторства текстовых фрагментов проверяемого документа остается в компетенции проверяющего.

№	Доля в отчете	Источник	Актуален на	Модуль поиска
[01]	0,92%	Processamento de linguagem natural em Portugu\^es e aprendizagem profunda para o dom\'inio de \'Oleo e G\'as http://arxiv.org	19 Мар 2020	Интернет
[02]	0,56%	Обзор методов обучения глубоких нейронных сетей https://cyberleninka.ru	03 Мая 2020	Интернет
[03]	0,41%	On the Development of Novel Encryption Methods for Conventional and Biometric Images https://core.ac.uk	02 Ноя 2020	Интернет

Руководитель ВКР Громов М.Л. / Громов М.Л./