ВЕСТНИК ТОМСКОГО ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА

2022 Управление, вычислительная техника и информатика № 61

Tomsk State University Journal of Control and Computer Science

Научная статья
УДК 004.032.26, 004.272
doi: 10.17223/19988605/61/12

Реализация сигмоидной функции активации с помощью концепции перестраиваемых вычислительных сред

Дмитрий Вадимович Шашев[1], Владислав Владимирович Шатравин[2]

[1, 2] Томский государственный университет, Томск, Россия
[1] dshashev@mail.ru
[2] shatravin@stud.tsu.ru

Аннотация. Рассматривается вариант реализации сигмоидной функции активации для ускорителей нейронных сетей, целиком реализованных на перестраиваемых вычислительных средах (ПВС). Показаны преимущества применения подобных ускорителей в маломощных интеллектуальных системах. Предложены модели ускорителей на основе распределенной кусочно-линейной аппроксимации сигмоиды. Приведены результаты временной симуляции Verilog-модулей разработанных моделей.

Ключевые слова: сигмоида; функция активации; нейронные сети; аппроксимация; перестраиваемая вычислительная среда

***Благодарности*:** Исследование выполнено за счет гранта Российского научного фонда № 21-71-00012, https://rscf.ru/project/21-71-00012/.

***Для цитирования*:** Шашев Д.В., Шатравин В.В. Реализация сигмоидной функции активации с помощью концепции перестраиваемых вычислительных сред // Вестник Томского государственного университета. Управление, вычислительная техника и информатика. 2022. № 61. С. 117–127. doi: 10.17223/19988605/61/12

Original article
doi: 10.17223/19988605/61/12

Implementation of the sigmoid activation function using the reconfigurable computing environments

Dmitriy V. Shashev[1], Vladislav V. Shatravin[2]

[1, 2] Tomsk State University, Tomsk, Russian Federation
[1] dshashev@mail.ru
[2] shatravin@stud.tsu.ru

Abstract. In this paper, we consider options for implementation of the sigmoid activation function for hardware accelerators of neural networks implemented entirely on reconfigurable computing environments (RCE). The advantages of using such accelerators in low-power intelligent systems are shown. Accelerator models based on distributed piecewise linear approximation of the sigmoid are proposed. The time simulation results of the developed Verilog-models are presented.

Keywords: sigmoid; activation function; neural networks; approximation; reconfigurable computing environment

***Acknowledgments*:** The research was supported by the Russian Science Foundation, grant No. 21-71-00012, https://rscf.ru/project/21-71-00012/.

***For citation*:** Shashev, D.V., Shatravin, V.V. (2022) Implementation of the sigmoid activation function using the reconfigurable computing environments. *Vestnik Tomskogo gosudarstvennogo universiteta. Upravlenie, vychislitelnaja tehnika i informatika – Tomsk State University Journal of Control and Computer Science*. 61. pp. 117–127. doi: 10.17223/19988605/61/12

© Д.В. Шашев, В.В. Шатравин, 2022

Нейронные сети находят активное применение в разнообразных интеллектуальных системах для задач, решение которых классическими методами является трудоемким или даже невозможным. Среди таких задач – обработка естественного языка, классификация и распознавание образов, предиктивный анализ и многие другие. Одним из ключевых недостатков нейронных сетей является высокая вычислительная сложность. Особенно это характерно для активно применяемых сверточных нейронных сетей, размер которых сейчас достигает сотен миллиардов параметров [1].

Для эффективного применения нейронных сетей в маломощных автономных системах (например, сенсорные сети, мобильные сети) требуются разработка специализированных аппаратных ускорителей и тонкая адаптация применяемых алгоритмов.

Существует множество работ на тему оптимизации аппаратного и алгоритмического обеспечения систем машинного обучения для маломощных устройств. В качестве развития аппаратного обеспечения обычно предлагается замена мощных, но энергозатратных графических процессоров на более эффективные FPGA и ASIC [2–5]. При этом осуществляется адаптация алгоритмов под особенности конкретной архитектуры для эффективного использования вычислительных ресурсов. Зачастую также производится компромиссное ухудшение некоторых характеристик исходной модели для уменьшения вычислительной сложности. К примеру, в [6, 7] предлагается переход к числам меньшей точности, а в [8, 9] – полная бинаризация сигналов. В работе [10] рассматривается уменьшение сложности сети через устранение части внутренних связей – разряжение сети.

Данная работа посвящена аспектам построения перестраиваемых ускорителей для реализации нейронных сетей на основе концепции перестраиваемых вычислительных сред. Данный подход открывает новые возможности для реализации нейросетевых алгоритмов, в том числе с точки зрения повышения качественных характеристик. Одним из ключевых вопросов проектирования ускорителя на перестраиваемых средах является реализация функций активации нейронов. В данной работе предложен вариант реализации на перестраиваемых вычислительных средах одной из наиболее популярных функций активации – сигмоидной.

1. Динамически перестраиваемые ускорители нейронных сетей

Искусственные нейронные сети представляют собой математическую модель вычислительной системы, имитирующей структуру и базовые принципы функционирования биологических нейронных сетей. В классической архитектуре искусственной сети нейроны распределены в несколько слоев (рис. 1). Каждый нейрон выполняет простую функцию: суммирует значения на всех своих входах, добавляет к ним собственное значение смещения и применяет к результату заданную функцию активации (рис. 2). Полученный результат передается нейронам следующего слоя, умножаясь на вес связи [11].

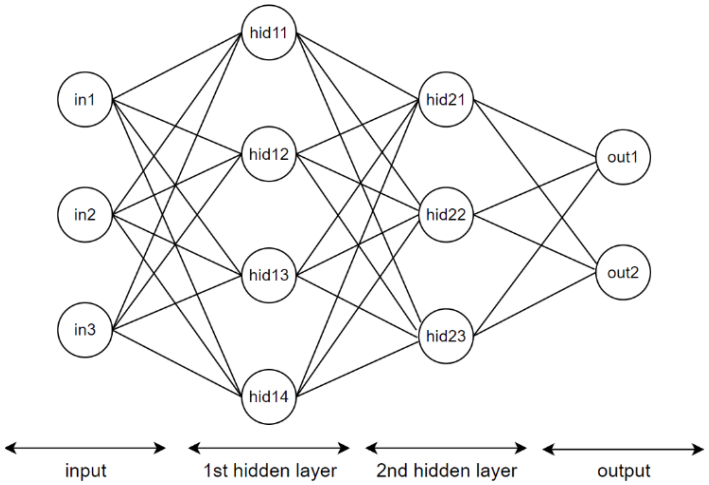

Рис. 1. Полносвязная сеть прямого распространения с двумя скрытыми слоями
Fig. 1. A fully connected direct distribution network with two hidden layers

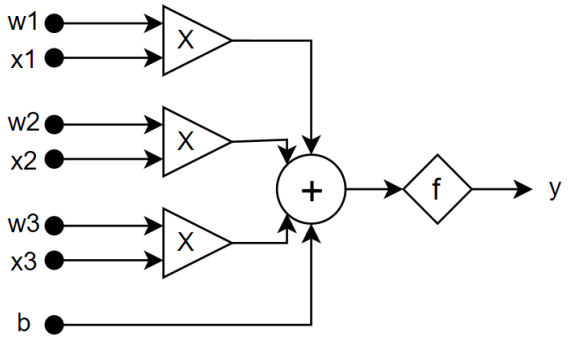

Рис. 2. Структура искусственного нейрона с тремя входами
Fig. 2. The structure of an artificial neuron with three inputs

Функция активации нейрона – нелинейная функция одного аргумента, которая приводит взвешенную сумму входов нейрона к некоторому распределению. На практике применяются различные функции активации в зависимости от назначения слоя и решаемой сетью задачи. В задачах классификации в нейронах скрытых слоев зачастую используется функция активации ReLU, а на выходном слое – сигмоидная функция. Это позволяет получить на выходе сети результат в виде вероятности принадлежности объекта целевому классу. Использование в сигмоидной функции операций деления и возведения в произвольную степень усложняет ее реализацию, в связи с чем применяются различные способы адаптации и упрощения. В данной работе сигмоидная функция будет заменена ее кусочно-линейной аппроксимацией.

Внедрение глубоких нейронных сетей в реальные системы приводит к необходимости применения специальных вычислительных устройств. Большое количество однотипных вычислений даже на современных центральных процессорах требует значительных затрат времени и энергии. В последние годы набирает популярность использование гибридных систем, где применяются одновременно центральный процессор и специализированный под задачи машинного обучения сопроцессор [12]. Благодаря распределению задач между центральным процессором и сопроцессором такие системы достаточно эффективно справляются с широким классом задач. Но при этом сохраняется проблема узкой направленности сопроцессоров. Одним из способов устранения такого ограничения является применение динамически перестраиваемых аппаратных ускорителей.

Отличительной особенностью динамически перестраиваемых ускорителей является способность менять внутреннюю структуру в процессе функционирования для изменения реализуемой модели сети. Иными словами, одно и то же вычислительное устройство в разные моменты времени может реализовывать нейронные сети принципиально разных архитектур. Это дает следующие преимущества:

– способность использовать разные архитектуры нейронных сетей без изменения аппаратной базы;

– поддержку различных моделей сети для различных режимов функционирования (к примеру, использование в режиме ожидания менее точной, но экономной модели);

– поддержку удаленной настройки и модификации реализуемых алгоритмов, что может быть очень полезно, когда отсутствует априорная информация, необходимая для подготовки целевой модели сети, или когда условия функционирования изменчивы;

– возможность восстановления работоспособности устройства посредством изменения структуры вычислителя и перераспределения вычислений по неповрежденным участкам.

На данный момент разрабатываются различные подходы к построению перестраиваемых ускорителей. В [13] предлагается иерархическая «многогранульная» архитектура на основе разделяемой нейронной сети (Bisection Neural Network), допускающая несколько уровней настройки. Работа [14] предлагает реализацию перестраиваемости посредством выделения некоторого количества реконфигурируемых нейронов, которые могут свободно перераспределяться между скрытыми слоями сети. В данной работе авторами предлагается построение динамически перестраиваемых ускорителей на основе концепции перестраиваемых вычислительных сред.

2. Перестраиваемые вычислительные среды

Перестраиваемые вычислительные среды (ПВС) – математическая модель широкого класса вычислительных устройств, основанных на идее организованного совместного функционирования большого количества однотипных, относительно простых вычислительных элементов (ВЭ), расположенных в виде регулярной решетки и попарно соединенных с соседними вычислительными элементами [15, 16]. Каждый ВЭ может быть независимо настроен при помощи внешнего управляющего сигнала на одну операцию из некоторого предопределенного базиса. Организованное функционирование большого количества вычислителей позволяет реализовать на среде сложные алгоритмы обработки. Таким образом, вычислительные возможности перестраиваемой среды ограничиваются количеством элементов и заданным базисом операций.

В общем случае ПВС может иметь ВЭ произвольной формы и быть одно-, двух- и трехмерной. В данной работе будут рассматриваться двумерные среды с квадратными ВЭ. Таким образом, каждый неграничный ВЭ будет соединен с четырьмя соседними элементами (рис. 3).

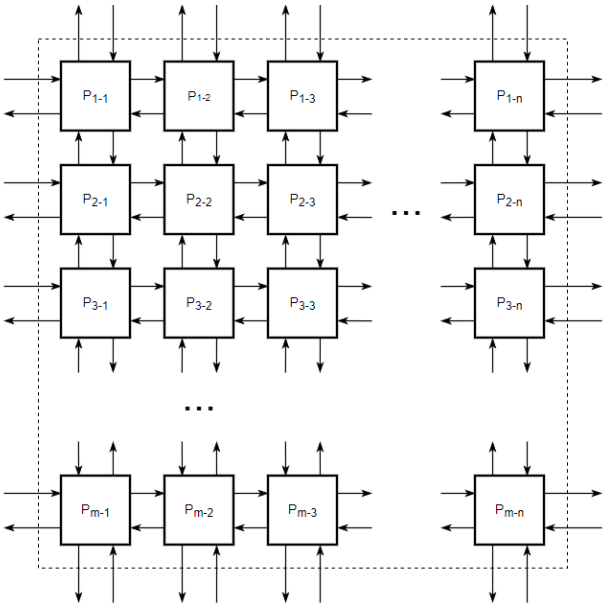

Рис. 3. Перестраиваемая вычислительная среда
Fig. 3. Reconfigurable computing environment

Одной из важных задач при проектировании ускорителей является уменьшение движения данных внутри системы в связи с сильным влиянием на быстродействие и энергопотребление. В целях снижения потока данных их стараются эффективно переиспользовать. Выделяют четыре основные стратегии переиспользования данных: фиксация весов, входов, выходов и строк [17]. При фиксации весов каждый ВЭ хранит собственное значение веса и использует его для каждого поступающего входного сигнала. Фиксация входов подразумевает, что каждый ВЭ хранит фрагмент входного сигнала, а значение весов поступает на вход элемента. Для фиксации выходов характерно пошаговое накопление в каждом ВЭ его промежуточных результатов (полусумм) до получения конечного результата. В ускорителях с фиксацией строк каждый ВЭ хранит строку данных и значение веса и последовательно рассчитывает несколько промежуточных результатов посредством умножения векторов в плавающем окне [18]. Предлагаемая нами модель ускорителя использует гибридную стратегию переиспользования (табл. 1). При обработке входного сигнала большого размера используется фиксация весов, а в остальных сценариях переиспользование не используется. Отсутствие переиспользования компенсируется замыканием всех вычислений внутри среды и конвейеризацией. Благодаря тому, что все вычисления осуществляются внутри среды, отсутствует необходимость ввода / вывода промежуточных результатов, устраняются накладные расходы на обращение к внешней памяти и другим модулям.

Таблица 1

Сравнение архитектур ускорителей на основе однородных сред

Характеристика	TPU systolic array [19]	Eyeriss [18]	Предложенная модель
Вычислительный элемент	8-битовый MAC	16-битовый MAC с модулями памяти и накоплением промежуточных результатов	16-битовый элемент с поддержкой 7 атомарных операций
Память ВЭ	Нет	448KB SRAM + 72KB Reg	21 бит
Размер ВЭ	Очень малый	Крупный	Средний
Количество вычислительных элементов	Очень много: 256 × 256 = 65 536	Мало: 12 × 14 = 168	Много, зависит от задачи
Перестраиваемость ВЭ	Нет	Нет	Да
Задача однородной среды	Матричное умножение	Матричное умножение	Полный цикл вычислений
Стратегия переиспользования данных	Фиксация весов	Фиксация строк	Гибридная (фиксация весов либо отсутствие переиспользования)
Место хранения промежуточных результатов	Буфер	Буфер	ВЭ
Постобработка (активация, субдискретизация)	Отдельный блок	Отдельный блок	Внутри среды

В отличие от некоторых альтернативных решений [13, 18], предлагаемая нами архитектура ускорителя основана на принципе предельной атомарности вычислительных элементов. Это означает, что ВЭ имеют простое строение, а базис операций включает минимально необходимое число операций. Это позволяет добиться высокой степени гибкости среды и допускает тонкую настройку параметров реализуемой модели. Анализ классических сетей прямого распространения позволил выделить следующий базис операций: «источник сигнала», «передача сигнала», «умножение с накоплением», «ReLU», «сигмоида» [20]. С их помощью каждый нейрон сети может быть представлен в виде цепочки из нескольких ВЭ (рис. 4) [21]. Длина цепочки определяет количество входов, что позволяет создавать нейрон любой конфигурации. Разработанная модель ВЭ оперирует 16-битными числами с фиксированной запятой.

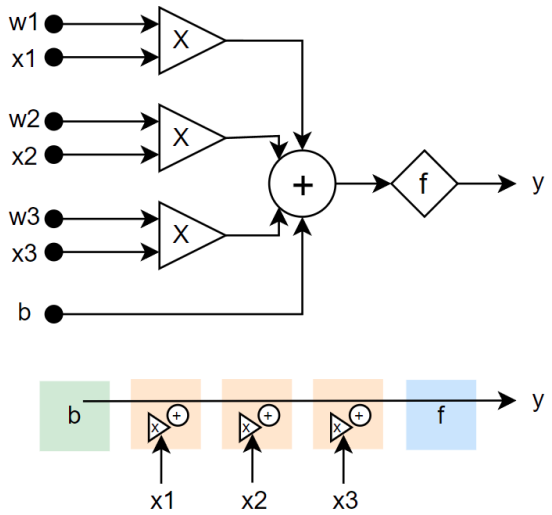

Рис. 4. Реализация нейрона на вычислительных элементах среды
Fig. 4. Implementation of a neuron on computational elements of the environment

Для поддержания реализации глубоких нейронных сетей и обеспечения конвейерной обработки в разрабатываемую вычислительную среду заложен механизм многотактной обработки сигнала посредством внутреннего зацикливания [22]. Он заключается в разбиении среды на несколько сегментов, каждый из которых реализует один из слоев сети. Сигнал, перемещаясь по сегменту, не только проходит необходимую обработку, но и разворачивается в направлении следующего сегмента (рис. 5). По мере движения сигнала сегменты динамически перестраиваются на реализацию последующих

слоев сети. Такая архитектура позволяет реализовать на среде сеть произвольной глубины. Связанные с многотактной обработкой потери производительности могут быть частично устранены при помощи конвейеризации. Для четырехугольной среды с четырьмя сегментами можно на каждом такте перестраивать один сегмент, а на трех других производить обработку трех смещенных друг относительно друга на один такт входных сигналов. Реализация многотактной обработки требует добавления в базис операции «фиксация» для хранения промежуточного результата между сегментами среды. Функция поворота сигнала на 90° заложена в операцию «умножение с накоплением».

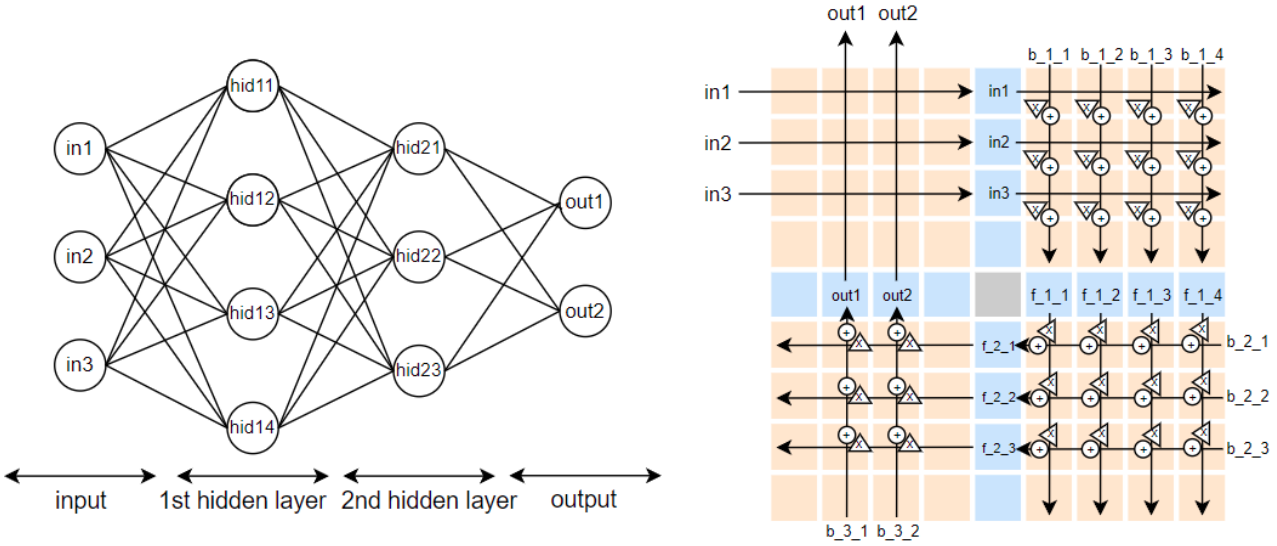

Рис. 5. Реализация многотактной обработки сигнала на сегментированной среде
Fig. 5. Implementation of multi-step signal processing in a segmented environment

Другим недостатком многотактной реализации является ограничение максимально поддерживаемого размера слоя в связи с сегментированием среды. Для решения этой проблемы может быть использован цельный режим функционирования, когда для реализации слоя сети используется вся площадь среды. Данный режим требует применения дополнительной внешней (относительно среды) памяти и контроллера маршрутизации входного сигнала (рис. 6).

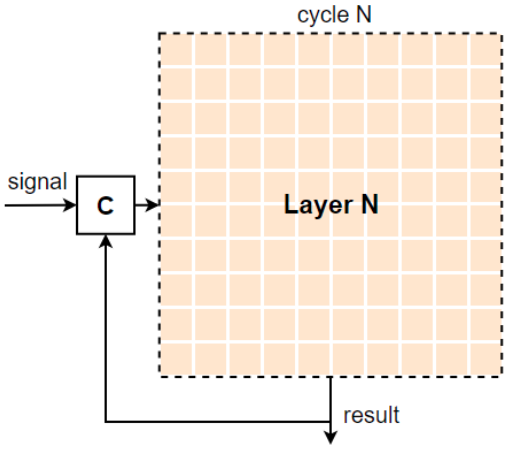

Рис. 6. Структура ускорителя в режиме без сегментирования
Fig. 6. Accelerator structure in non-segmentation mode

Временная симуляция нейронов, выполненных согласно обозначенным принципам, показала приемлемые результаты временной симуляции. Однако одна из базовых операций – сигмоидная функция активации – в изначальном виде является значительно более трудоемкой, что приводит к большой площади каждого вычислительного элемента и их низкому быстродействию.

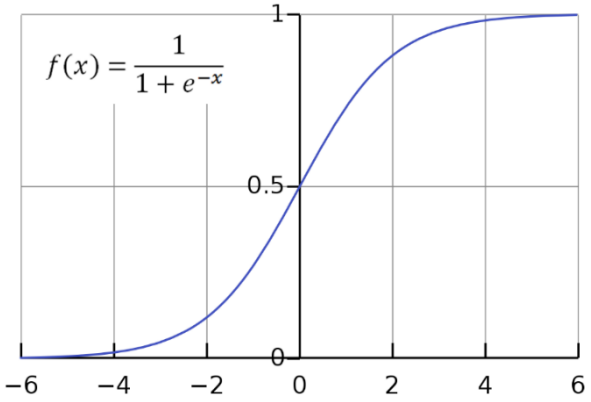

Рис. 7. Сигмоидная функция активации
Fig. 7. Sigmoid activation function

График и математическое выражение сигмоиды (ее также называют логистической функцией) приведены на рис. 7. Применение предложенной в [23] кусочно-линейной аппроксимации отчасти решает проблему, но операция по-прежнему остается самой трудоемкой во всей модели ВЭ. В связи с этим в данной работе рассматривается распределенная реализация сигмоиды на однородных средах.

3. Распределенная реализация сигмоидной функции активации

Распределенная реализация опирается на возможность независимого расчета значений на разных интервалах аппроксимации. Это позволяет распределить вычисления интервалов между разными вычислительными элементами. Блок вычислительных элементов среды, соответствующий распределенной реализации сигмоиды, представлен на рис. 8.

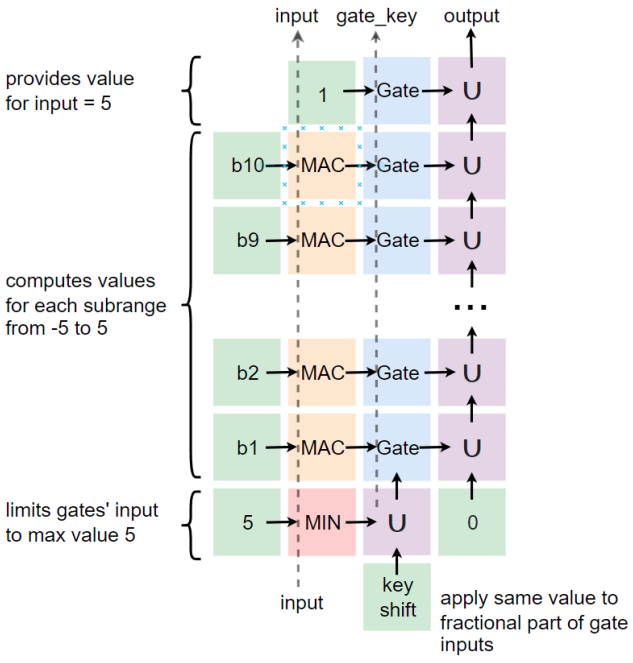

Рис. 8. Распределённая реализация сигмоиды
Fig. 8. Distributed implementation of sigmoid

Для наглядности используется цветовая дифференциация операций: зеленый – «источник сигнала», красный – «минимум двух чисел», желтый – «умножение с накоплением» (первый множитель поступает на нижний вход элемента, второй хранится во внутренней памяти элемента, аккумулирующее значение поступает на левый вход), синий – «затвор», фиолетовый – «объединение». Как видно,

для реализации распределенной сигмоиды было необходимо ввести в базис несколько новых операций – «минимум», «затвор» и «объединение». На данный момент эти операции используются только для сигмоиды, но они могут быть переиспользованы для реализации других функций, поэтому их введение в базис операций целесообразно.

Операция «затвор» используется для ограничения распространения сигнала в зависимости от поступающего на вход-ключ значения. На представленной схеме вход-ключ расположен на нижней грани элемента, а управляемый сигнал поступает на левый вход. Затвор сравнивает поступающее на вход значение ключа и, если он совпадает с заложенным во внутреннюю память элемента значением, пропускает входной сигнал через себя (слева направо). В противном случае на выходе элемента (справа) будет 16-битный ноль. Значение ключа передается вверх другим элементам-затворам независимо от срабатывания текущего затвора. Таким образом, у всех затворов, чей внутренний параметр не совпадает со поступающим значением ключа, выход будет равен нулю.

Как было показано ранее, при сравнении входного сигнала с узлами аппроксимации нас интересует только целая часть числа. В то же время операция «затвор» осуществляет сравнение ключа целиком. Это необходимо для переиспользования операции в других задачах, где сравнения только целой части числа может быть недостаточно. Для разрешения этого противоречия используется нижняя пара вычислительных элементов «key shift» и «объединение». Они осуществляют дополнительное смещение значения ключа – все биты дробной части (младшие 8 бит ключа) устанавливаются равными единице.

Операция объединения заключается в применении побитового ИЛИ к входным сигналам. Она позволяет объединить промежуточные результаты, полученные на разных интервалах аппроксимации (выходные значения элементов-затворов), в конечный результат. Также операция используется для осуществления сдвига ключа.

Минимум используется для ограничения максимального значения ключа. В принятом алгоритме аппроксимации всем значениям аргумента свыше 5 соответствует значение функции 1. Поэтому множество проверок узлов от 5 до 127 можно заменить ограничением ключа значением 5 и одним сравнением с этим значением. Для значений от –128 до –5 дополнительной логики не предусмотрено, так как на этом диапазоне значение функции равно 0, что соответствует нахождению всех затворов в закрытом состоянии.

Таким образом, на реализацию распределенной сигмоиды требуется 48 вычислительных элементов.

4. Симуляция разработанных моделей

Для сравнения предложенной реализации сигмоидной функции была разработана модель вычислительного элемента на языке описания аппаратуры Verilog. Разработанный Verilog-модуль включает весь описанный базис операций ВЭ. Оценивались две характеристики предложенных модулей – размер на среде (в логических элементах, ЛЭ) и быстродействие. Полученные результаты приведены в табл. 2.

Таблица 2

Результаты экспериментов

Реализация	Полный размер, ЛЭ	Размер ячейки, ЛЭ	Наибольшая задержка, нс	Абс. ошибка средняя	Абс ошибка макс.
Распределенная	13 175	296	18,5	$4e^{-3}$	$1e^{-2}$

Для оценки размера был осуществлен синтез модуля в среде Quartus Prime (версия 20.1.0, сборка 711 от 06.05.2020 SJ Edition) для устройства Cyclone V 5CGXFC9E7F35C8. В настройках синтеза отключено использование DSP блоков. Для защиты всех внутренних подмодулей от удаления в процессе оптимизации применялась Verilog-директива synthesis keep.

Для анализа быстродействия использовался встроенный в среду Quartus инструмент Timing Analyzer соответствующей версии. Симуляция происходила в режиме Fast 1000mV 0C посредством

определения наибольшей задержки между входом и выходом исследуемого модуля. Оба модуля были предварительно сконфигурированы, т.е. время перенастройки ячеек не включено в результаты измерения.

Поскольку в структуре однородных сред существуют двусторонние связи между соседними элементами, при временной симуляции распределенной реализации Timing Analyzer обнаруживает петли комбинационной логики. Для устранения этой проблемы на момент измерения быстродействия распределенной реализации были отключены неиспользуемые соединения между отдельными ВЭ.

Измерение ошибки аппроксимации осуществлялось при помощи тестового приложения. Приложение берет случайное число в диапазоне аппроксимации [–5, 5], округляет его до точности 16-битного числа, рассчитывает значение аппроксимации в этой точке и сравнивает со значением натуральной сигмоиды для исходного неокругленного числа. Эксперимент повторяется $1e^6$ раз, в результате чего определяются наибольшая и средняя абсолютные ошибки.

Результаты аналогичных исследований приведены в [24]. Как видно из табл. 2, предложенная реализация сигмоидной функции имеет приемлемую среднюю ошибку и высокое быстродействие. Однако распределенная реализация требует значительно большего количества логических элементов. Это связано с распределением вычислений по большому количеству вычислительных элементов, каждый из которых содержит логику всех базисных операций среды. Также существенную роль играет комбинационная реализация арифметических операций, что увеличивает быстродействие за счет большей площади модулей.

Заключение

В данной работе предложен способ реализации кусочно-линейной аппроксимации сигмоиды на однородной среде. Распределенная реализация имеет меньшую производительность и требует большего количества ВЭ, но строится из простых базовых операций, которые могут быть переиспользованы для других задач и обеспечивают меньший размер отдельного ВЭ. Эксперименты показали высокое быстродействие (18,5 нс) и приемлемую точность (наибольшая и средняя абсолютные ошибки $1e^{-2}$ и $4e^{-3}$ соответственно). Сравнение с аналогичными исследованиями показывает перспективность дальнейшего изучения и применения разработанных моделей.

Список источников

1. Chen J., Li J., Majumder R. Make every feature binary: A 135B parameter sparse neural network for massively improved search relevance. 2021. URL: https://www.microsoft.com/en-us/research/blog/make-every-feature-binary-a-135b-parameter-sparse-neural-network-for-massively-improved-search-relevance/ (accessed: 18.07.2022).
2. Nabavinejad S.M., Reda S., Ebrahimi M. Coordinated Batching and DVFS for DNN Inference on GPU Accelerators // IEEE Transactions on Parallel and Distributed Systems. 2022. 12 p. URL: https://people.kth.se/~mebr/assets/files/TPDS2022-%20Coordinated%20Batching%20and%20DVFS%20for%20DNN%20Inference%20on%20GPU%20Accelerators.pdf (accessed: 18.07.2022).
3. Guo J. et al. AccUDNN: A GPU Memory Efficient Accelerator for Training Ultra-Deep Neural Networks // IEEE 37th International Conference on Computer Design (ICCD). 2019. P. 65–72. doi: 10.1109/ICCD46524.2019.00017
4. Chang K.C., Fan C.P. Cost-Efficient Adaboost-based Face Detection with FPGA Hardware Accelerator // 2019 IEEE International Conference on Consumer Electronics – Taiwan (ICCE-TW). 2019. P. 1–2.
5. Lee J., He J., Wang K. Neural Networks and FPGA Hardware Accelerators for Millimeter-Wave Radio-over-Fiber Systems // 2020 22nd International Conference on Transparent Optical Networks (ICTON). 2020. P. 1–4.
6. Sakai Y. Quantizaiton for Deep Neural Network Training with 8-bit Dynamic Fixed Point // 2020 7th International Conference on Soft Computing & Machine Intelligence (ISCMI). 2020. P. 126–130. doi: 10.1109/ISCMI51676.2020.9311563
7. Trusov A., Limonova E., Slugin D., Nikolaev D., Arlazarov V.V. Fast Implementation of 4-bit Convolutional Neural Networks for Mobile Devices // 2020 25th International Conference on Pattern Recognition (ICPR). 2021. P. 9897–9903. doi: 10.1109/ICPR48806.2021.9412841
8. Liu Z., Zhang H., Su Z., Zhu X. Adaptive Binarization Method for Binary Neural Network // 2021 40th Chinese Control Conference (CCC). 2021. P. 8123–8127. doi: 10.23919/CCC52363.2021.9549344
9. Zhu B., Al-Ars Z., Hofstee H.P. NASB: Neural Architecture Search for Binary Convolutional Neural Networks // 2020 International Joint Conference on Neural Networks (IJCNN). 2020. P. 1–8. doi: 10.1109/IJCNN48605.2020.9207674
10. Tang Z. et al. Automatic Sparse Connectivity Learning for Neural Networks // IEEE Transactions on Neural Networks and Learning Systems. 2022. P. 1–15. doi: 10.1109/TNNLS.2022.3141665

11. Haykin S. Neural Network: A Comprehensive foundation. Prentice Hall International, 1999. 842 p.
12. Chajan E., Schulte-Tigges J., Reke M., Ferrein A., Matheis D., Walter T. GPU based model-predictive path control for self-driving vehicles // 2021 IEEE Intelligent Vehicles Symposium (IV). 2021. P. 1243–1248. doi: 10.1109/IV48863.2021.9575619
13. Kan Y., Wu M., Zhang R., Nakashima Y. A multi-grained reconfigurable accelerator for approximate computing // IEEE Computer Society Annual Symposium on VLSI (ISVLSI). 2020. P. 90–95.
14. Khalil K., Eldash O., Dey D., Kumar A., Bayoumi M. A Novel Reconfigurable Hardware Architecture of Neural Networ // IEEE 62nd International Midwest Symposium on Circuits and Systems (MWSCAS). 2019. P. 618–621.
15. Shashev D.V., Shidlovskiy S.V. Morphological processing of binary images using reconfigurable computing environments // Optoelectronics, Instrumentation and Data Processing. 2015. № 51. P. 227–233. doi: 10.3103/S8756699015030036
16. Евреинов Э.В. Однородные вычислительные системы, структуры и среды. М. : Радио и связь, 1981. 208 с.
17. Ghimire D., Kil D., Kim S-h. A Survey on Efficient Convolutional Neural Networks and Hardware Acceleration // MDPI J. Electronics. 2022. V. 945. P. 1–23.
18. Chen Y., Krishna T., Emer J.S., Sze V. Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks // IEEE J. Solid-State Circuits. 2017. V. 52. P. 127–138.
19. An in-depth look at Google's first Tensor Processing Unit (TPU). 2017. URL: https://cloud.google.com/blog/products/ai-machine-learning/an-in-depth-look-at-googles-first-tensor-processing-unit-tpu (accessed: 18.07.2022).
20. Shatravin V., Shashev D.V. Designing high performance, power-efficient, reconfigurable compute structures for specialized applications // Journal of Physics: Conference Series. 2020. V. 1611. P. 1–6.
21. Shatravin V., Shashev D.V., Shidlovskiy S.V. Applying the Reconfigurable Computing Environment Concept to the Deep Neural Network Accelerators Development // International Conference on Information Technology (ICIT). 2021. P. 842–845. doi: 10.1109/ICIT52682.2021.9491771
22. Shatravin V., Shashev D.V., Shidlovskiy S.V. Developing of models of dynamically reconfigurable neural network accelerators based on homogeneous computing environments // Proc. of the XXIV international scientific conference Distributed Computer and Communication Networks: Control, Computation, Communications (DCCN). 2021. P. 102–107.
23. Faiedh H., Gafsi Z., Besbes K. Digital Hardware Implementation of Sigmoid Function and its Derivative for Artificial Neural Networks // The 13 International Conference on Microelectronics. 2001. P. 189–192.
24. Pan Z., Gu Z., Jiang X., Zhu G., Ma D. A Modular Approximation Methodology for Efficient Fixed-Point Hardware Implementation of the Sigmoid Function // IEEE Transactions on Industrial Electronics. 2022. P. 10694–10703. doi: 10.1109/TIE.2022.3146573

References

1. Chen, J., Li, J. & Majumder, R. (2021) *Make every feature binary: A 135B parameter sparse neural network for massively improved search relevance*. [Online] Available from: https://www.microsoft.com/en-us/research/blog/make-every-feature-binary-a-135b-parameter-sparse-neural-network-for-massively-improved-search-relevance/ (Accessed: 17th July 2022).
2. Nabavinejad, S.M., Reda, S. & Ebrahimi, M. (2022) Coordinated Batching and DVFS for DNN Inference on GPU Accelerators. *IEEE Transactions on Parallel and Distributed Systems*. pp. 1–12.
3. Guo, J. et al. (2019) AccUDNN: A GPU Memory Efficient Accelerator for Training Ultra-Deep Neural Networks. *IEEE 37th International Conference on Computer Design (ICCD)*. pp. 65–72. DOI: 10.1109/ICCD46524.2019.00017
4. Chang, K.C. & Fan, C.P. (2019) Cost-Efficient Adaboost-based Face Detection with FPGA Hardware Accelerator. *2019 IEEE International Conference on Consumer Electronics – Taiwan (ICCE-TW)*. pp. 1–2.
5. Lee, J., He, J. & Wang, K. (2020) Neural Networks and FPGA Hardware Accelerators for Millimeter-Wave Radio-over-Fiber Systems. *22nd International Conference on Transparent Optical Networks (ICTON)*. pp. 1–4.
6. Sakai, Y. (2020) Quantizaiton for Deep Neural Network Training with 8-bit Dynamic Fixed Point. *7th International Conference on Soft Computing & Machine Intelligence (ISCMI)*. pp. 126–130. DOI: 10.1109/ISCMI51676.2020.9311563
7. Trusov, A., Limonova, E., Slugin, D., Nikolaev, D. & Arlazarov, V.V. (2021) Fast Implementation of 4-bit Convolutional Neural Networks for Mobile Devices. *25th International Conference on Pattern Recognition (ICPR)*. pp. 9897–9903. DOI: 10.1109/ICPR48806.2021.9412841
8. Liu, Z., Zhang, H., Su, Z. & Zhu, X. (2021) Adaptive Binarization Method for Binary Neural Network. *40th Chinese Control Conference (CCC)*. pp. 8123–8127. DOI: 10.23919/CCC52363.2021.9549344
9. Zhu, B., Al-Ars, Z. & Hofstee, H.P. (2020) NASB: Neural Architecture Search for Binary Convolutional Neural Networks. *International Joint Conference on Neural Networks (IJCNN)*. pp. 1–8. DOI: 10.1109/IJCNN48605.2020.9207674
10. Tang, Z. et al. (2022) Automatic Sparse Connectivity Learning for Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*. pp. 1–15. DOI: 10.1109/TNNLS.2022.3141665
11. Haykin, S. (1999) *Neural Network: A Comprehensive Foundation*. Prentice Hall International.
12. Chajan, E., Schulte-Tigges, J., Reke, M., Ferrein, A., Matheis, D. & Walter, T. (2021) GPU based model-predictive path control for self-driving vehicles. *IEEE Intelligent Vehicles Symposium (IV)*. pp. 1243–1248. DOI: 10.1109/IV48863.2021.9575619
13. Kan, Y., Wu, M., Zhang, R. & Nakashima, Y. (2020) A multi-grained reconfigurable accelerator for approximate computing. *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*. pp. 90–95.
14. Khalil, K., Eldash, O., Dey, B., Kumar, A. & Bayoumi, M. (2019) A Novel Reconfigurable Hardware Architecture of Neural Network. *IEEE 62nd International Midwest Symposium on Circuits and Systems (MWSCAS)*. pp. 618–621.

15. Shashev, D.V. & Shidlovskiy, S.V. (2015) Morphological processing of binary images using reconfigurable computing environments. *Optoelectronics, Instrumentation and Data Processing*. 51. pp. 227–233. DOI: 10.3103/S8756699015030036.
16. Evreinov, E.V. (1981) *Odnorodnye vychislitel'nye sistemy, struktury i sredy* [Homogeneous computing systems, structures and environments]. Moscow: Radio i svyaz'.
17. Ghimire, D., Kil, D. & Kim, S-h. (2022) A Survey on Efficient Convolutional Neural Networks and Hardware Acceleration. *MDPI J. Electronics*. 945. pp. 1–23.
18. Chen, Y., Krishna, T., Emer, J.S. & Sze, V. (2017) Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks. *IEEE J. Solid-State Circuits.* 52. pp. 127–138.
19. TPU. (2017) *An in-depth look at Google's first Tensor Processing Unit (TPU)*. [Online] Available from: https://cloud.google.com/blog/products/ai-machine-learning/an-in-depth-look-at-googles-first-tensor-processing-unit-tpu (Accessed: 17th July 2022).
20. Shatravin, V. & Shashev, D.V. (2020) Designing high performance, power-efficient, reconfigurable compute structures for specialized applications. *Journal of Physics: Conference Series*. 1611. pp. 1–6.
21. Shatravin, V., Shashev, D.V. & Shidlovskiy, S.V. (2021) Applying the Reconfigurable Computing Environment Concept to the Deep Neural Network Accelerators Development. *International Conference on Information Technology (ICIT)*. pp. 842–845. DOI: 10.1109/ICIT52682.2021.9491771.
22. Shatravin, V., Shashev, D.V. & Shidlovskiy, S.V. (2021) Developing of models of dynamically reconfigurable neural network accelerators based on homogeneous computing environments. *Distributed Computer and Communication Networks: Control, Computation, Communications (DCCN).* Proceedings of the 24th International Conference. pp. 102–107.
23. Faiedh, H., Gafsi, Z. & Besbes, K. (2001) Digital Hardware Implementation of Sigmoid Function and its Derivative for Artificial Neural Networks. *The 13 International Conference on Microelectronics*. pp. 189–192.
24. Pan, Z., Gu, Z., Jiang, X., Zhu G. & Ma, D. (2022) A Modular Approximation Methodology for Efficient Fixed-Point Hardware Implementation of the Sigmoid Function. *IEEE Transactions on Industrial Electronics*. pp. 10694–10703. DOI: 10.1109/TIE.2022.3146573

***Информация об авторах*:**
Шашев Дмитрий Вадимович – кандидат технических наук, доцент кафедры управления качеством факультета инновационных технологий Национального исследовательского Томского государственного университета (Томск, Россия). E-mail: dshashev@mail.ru
Шатравин Владислав Владимирович – аспирант кафедры управления качеством факультета инновационных технологий Национального исследовательского Томского государственного университета (Томск, Россия). E-mail: shatravin@stud.tsu.ru

Вклад авторов: все авторы сделали эквивалентный вклад в подготовку публикации. Авторы заявляют об отсутствии конфликта интересов.

Information about the authors:
Shashev Dmitriy Vadimovish (Candidate of Technical Sciences, Associate Professor, National Research Tomsk State University, Tomsk, Russian Federation). E-mail: dshashev@mail.ru

Shatravin Vladislav (Post-graduate Student, National Research Tomsk State University, Tomsk, Russian Federation). E-mail: shatravin@stud.tsu.ru

Contribution of the authors: the authors contributed equally to this article. The authors declare no conflicts of interests.

Received 06.04.2022; accepted for publication 29.11.2022

Поступила в редакцию 06.04.2022; принята к публикации 29.11.2022