Федеральное государственное учреждение «Федеральный исследовательский центр «Информатика и управление» Российской академии наук»

На правах рукописи

Лимонова Елена Евгеньевна

Биполярная морфологическая аппроксимация нейрона для уменьшения вычислительной сложности глубоких сверточных нейронных сетей

Специальность 1.2.2 —
«Математическое моделирование, численные методы и комплексы программ»

Диссертация на соискание учёной степени
кандидата технических наук

Научный руководитель:
кандидат технических наук
Арлазаров Владимир Викторович

Москва — 2022

Оглавление

Стр.

Стр.

Стр.

Введение

Современные технологии нейросетевого распознавания используются в различных сферах жизнедеятельности человека. Они способны облегчить решение ряда прикладных задач, однако их внедрение ограничивается не только точностью распознавания и скоростью работы, но и соображениями безопасности и конфиденциальности данных пользователей. Именно эти вопросы выходят на первый план при распознавании идентификационных документов, банковских карт и платежных документов, обработке медицинской информации. Один из наиболее эффективных способов обеспечения безопасности пользовательских данных предлагает концепция граничных вычислений, при которой вычисления выполняются в точке, максимально приближенной к конечному пользователю, в идеале — на конечном устройстве, где эти результаты и будут использоваться далее.

Однако конечные устройства чаще всего обладают ограниченной вычислительной мощностью и объемом доступной оперативной памяти. Кроме того, повышенные требования предъявляются к их энергоэффективности, так как часто они работают от аккумулятора (например, смартфоны и различные носимые устройства) или являются составной частью других систем, также ограниченных в энергопотреблении (например, беспилотных транспортных средств или элементов интернета вещей). Также вопрос энергоэффективности нейросетевого распознавания в последнее время привлекает все большее внимание из-за возможного негативного влияния на экологию вследствие затрат энергии на обучение и многократные запуски глубоких нейронных сетей уже после их внедрения.

Таким образом, разработчикам распознающих систем с одной стороны необходимо обеспечить достаточно высокую точность распознавания для успешного решения поставленных задач, которая обычно достигается за счет усложнения нейросетевых моделей, а с другой — выполнить требования по энергоэффективности и скорости работы.

Особенно важной является эта задача в случае распознавания в режиме реального времени, например, при обработке видеопотока: прослеживании траекторий объектов, сегментации меняющейся сцены или извлечении текстовой информации в произвольных условиях.

Для визуального распознавания, как правило, используются модели, имеющие сверточную архитектуру, то есть состоящие из большого количества последовательно расположенных сверточных слоев, между которыми могут включаться слои субдискретизации, нормализации или слои других типов. Основную вычислительную сложность таких сетей составляют именно вычисления в сверточных слоях. Для обеспечения высокой точности распознавания такие модели выполняют несколько миллиардов операций аккумулирующего умножения на запуск. Современные мощные вычислительные устройства имеют частоту в несколько гигагерц и несколько вычислительных ядер, однако даже они могут рассчитать всего несколько таких запусков в секунду. Таким образом, задача исследования вычислительной эффективности сверточных нейросетевых моделей на сегодняшний день крайне актуальна. В разное время ей занимались отечественные и зарубежные ученые, такие как Ю. И. Журавлев, В. Л. Арлазаров, В. А. Сойфер, Ю. В. Визильтер, И. Б. Гуревич, В. Б. Бетелин, Д. П. Николаев, а также H. Wen, M. Rastegari, A. Farhadi, Y. Lecun, Y. Bengio, G. Hinton и другие.

Повышение вычислительной эффективности таких моделей возможно из-за наличия в них неявной вычислительной избыточности. Исследования показывают, что эта избыточность в большей степени связана с несовершенством существующих методов обучения, а не конкретным числом нейронов и способом их организации в слои. Процесс обучения нейросетевых моделей заключается в поиске минимума некоторой функции потерь, которая в общем случае является невыпуклой и имеет множество экстремумов. С теоретической точки зрения такая задача крайне сложна и не имеет общего решения. Вследствие этого поиск методов снижения вычислительной избыточности нейросетевых моделей носит экспериментальный характер. Есть множество методов, снижающих число тех или иных арифметических операций в нейросетевых моделях, таких как тензорные декомпозиции, обрезка моделей, применение дистилляции знаний для создания более простых моделей. Эти методы позволяют в разы или даже на порядки снизить число операций, однако все еще не позволяют достичь желаемой вычислительной эффективности глубоких нейросетевых моделей при сохранении удовлетворительной точности распознавания.

Одним из наиболее перспективных направлений повышения вычислительной эффективности нейросетевых моделей последнего времени является совместная оптимизация архитектуры нейросетевой модели и архитектуры вы-

числительного устройства. Оно требует высокой квалификации специалиста как в области искусственного интеллекта, так и в области проектирования вычислительных устройств, или создания междисциплинарной команды ученых. Ведь разработчики нейросетевых моделей, ограниченные конкретным вычислительным устройством или классом устройств, вынуждены проектировать модели, опираясь на доступный объем вычислительных ресурсов. Как правило, современные нейросетевые модели направлены на исполнение на графических процессорах. С другой стороны, разработчики специализированных устройств выполняют большую работу по низкоуровневому проектированию и обычно рассматривают лишь одну нейросетевую архитектуру, позволяющую получить высокую точность распознавания. Результирующее устройство при этом отличается высокой эффективностью, но может требовать модификации при малейших изменениях модели. В качестве компромисса были созданы специализированные тензорные процессоры (например, Google TPU или Intel VPU), которые могут эффективно исполнять отдельные классы нейросетевых моделей. Однако они также потребуют модификации при создании новых классов распознающих архитектур, например, в настоящее время они не поддерживают исполнение моделей с бинарными или тернарными весами.

В таких условиях особый интерес представляет смена модели вычислений в элементарных логических элементах нейронной сети — отдельных слоях или отдельных нейронах. Такие изменения не затрагивают архитектуру сети и все также позволяют строить и использовать модели разных типов, но могут сделать аппаратную реализацию модели гораздо эффективнее, поскольку различные типы нейронов требуют разных аппаратных и энергетических затрат при реализации и в процессе работы. Поскольку существующие модели слоев и нейронов уже доказали свою эффективность в решении практических задач и позволяют добиться высокой точности распознавания, данная работа посвящена исследованию их аппроксимаций, упрощающих последующее создание вычислителя, но при этом сохраняющих высокую точность работы.

Основные результаты диссертации были получены в процессе выполнения работ по следующим научным грантам РФФИ:

1. 18-07-01384 — «Исследование применимости методов нелинейных аппроксимаций для оптимизации быстродействия искусственных нейронных сетей на современнных микропроцессорных архитектурах»

2. 17-29-03297 — «Исследование возможности создания энергоэффективных аппаратных устройств для мобильных устройств комплексов идентификации и верификации личности в составе систем технического зрения наземных робототехнических комплексов»
3. 17-29-03240 — «Глубокие нейронные сети с вычислительно упрощенной моделью нейрона»

Целью данной работы является разработка и исследование вычислительно-эффективных аппроксимаций нейросетевых моделей, методов их обучения и оптимизации их вычисления на существующих и перспективных вычислителях.

Для достижения этой цели были поставлены следующие **задачи**:

1. Разработать метод аппроксимации вычислительно-интенсивных частей нейросетевых моделей, исследовать его вычислительную эффективность и точность.
2. Оценить вычислительную эффективность на различных платформах.
3. Разработать методы обучения предложенной аппроксимирующей структуры.
4. Провести экспериментальную оценку точности предложенного метода обучения аппроксимированных нейросетевых моделей для различных нейросетевых архитектур.
5. Разработать комплекс программ, позволяющий моделировать аппроксимацию нейросетевых моделей, обучение полученных структур и проверку результирующего качества работы.

Научная новизна:

1. Предложена новая аппроксимация классического нейрона нейроном с морфологической структурой, позволяющая создавать глубокие нейронные сети с морфологическими слоями и обеспечивающая высокую точность распознавания.
2. Предложен новый метод обучения произвольных, в том числе биполярных морфологических и целочисленных, аппроксимаций классических нейросетевых моделей путем послойного преобразования и дообучения, позволяющий повысить их качество.
3. Впервые показано, что для предложенной аппроксимации метод послойного преобразования и дообучения позволяет добиться более высокого качества работы нейросетевой модели, чем прямое обучение с

помощью метода обратного распространения ошибки и градиентных методов оптимизации.

4. Проведено оригинальное исследование точностных характеристик нейросетевых моделей LeNet- и ResNet-подобных архитектур, использующих предложенную морфологическую аппроксимацию.
5. Впервые теоретически показано, что нейросетевая модель с достаточным числом нейронов биполярного морфологического вида может приблизить произвольную непрерывную на компакте функцию с любой заранее заданной точностью.

Практическая значимость. Предложенная аппроксимация позволяют создать нейросетевые модели, подобные по архитектуре классическим глубоким моделям, но в то же время обладающие принципиально новыми теоретическими свойствами. Она снижает вычислительную сложность исходных моделей и потенциально способна повысить их эффективность.

Разработанные в рамках диссертации методы были реализованы в виде программных компонентов и внедрены в программное обеспечение «Smart ID Engine», «Smart Code Engine», «Smart Document Engine», а также «Smart IDReader» компании ООО «Смарт Энджинс Сервис». Данные продукты интегрированы в информационную инфраструктуру и мобильные приложения АО «Тинькофф Банк», а также в ряд информационных решений государственных структур Российской Федерации. Кроме того, полученные оценки и результаты моделирования демонстрируют, что включение специализированных модулей для элементарных арифметических операций при создании устройств для исполнения нейросетевых моделей способно повысить эффективность их работы и используются в АО «МЦСТ» при проектировании новых устройств.

Соответствие диссертации паспорту научной специальности. В соответствии с формулой специальности 1.2.2 «Математическое моделирование, численные методы и комплексы программ» (технические науки) в работе выполнены разработка, исследование и реализация модели вычислительно-эффективного биполярного морфологического нейрона как аппроксимации классического математического нейрона. Работа соответствует следующим пунктам паспорта специальности: п. 2 «Разработка, обоснование и тестирование эффективных вычислительных методов с применением современных компьютерных технологий», п. 3 «Реализация эффективных численных методов и алгоритмов в виде комплексов проблемно-ориентированных программ для

проведения вычислительного эксперимента», п. 7 «Качественные или аналитические методы исследования математических моделей (технические науки)» и п. 9 «Постановка и проведение численных экспериментов, статистический анализ их результатов, в том числе с применением современных компьютерных технологий (технические науки)».

Методология и методы исследования. В диссертационной работе использовались методы математического анализа, линейной алгебры, методы численного моделирования и нелинейной теории оптимизации.

Основные положения, выносимые на защиту:

1. Разработана аппроксимация модели математического нейрона и сверточного слоя: биполярные морфологические нейрон и сверточный слой, не задействующие умножений в своих вычислительно-интенсивных частях.
2. Доказано, что нейронная сеть из биполярных морфологических нейронов может с любой заранее заданной точностью приблизить любую непрерывную на компакте функцию.
3. Предложен метод обучения аппроксимаций классических нейросетевых моделей путем послойного преобразования и дообучения, позволяющий повысить их качество.
4. Экспериментально показано, что предложенный метод послойного преобразования и дообучения позволяет добиться высокого качества работы аппроксимированных нейросетевых моделей на примере линейно квантованных малобитных и биполярных морфологических нейронных сетей.
5. Разработан комплекс программ, реализующий предложенную в работе модель биполярного морфологического нейрона, метод послойного дообучения для этой модели и позволяющий оценивать точностные характеристики результирующих сетей.

Достоверность полученных результатов подтверждается соответствием теоретических и экспериментальных результатов, продемонстрированных в работе, успешной апробацией результатов и внедрением в коммерческие системы распознавания документов.

Апробация работы. Основные результаты работы докладывались и обсуждались на следующих семинарах и конференциях:

1. Междисциплинарной школе-конференции Института проблем передачи информации им. А. А. Харкевича Российской академии наук (ИППИ РАН) «Информационные технологии и системы» (ИТиС) в 2015 году.
2. Международной конференции «International Conference on Machine Vision» (ICMV) в 2016, 2019, 2020 годах.
3. международной конференции «International Conference on Pattern Recognition» (ICPR) в 2020 году.
4. Научном семинаре Лаборатории №11 ИППИ РАН в 2021 году.
5. Международном научно-исследовательском семинаре «Анализ и понимание изображений (Математические, когнитивные и прикладные проблемы анализа изображений и сигналов)» в 2022 году.

Личный вклад. Все основные результаты диссертационной работы получены и обоснованы автором самостоятельно. Постановка задач и обсуждение результатов проводились совместно с научным руководителем. В [1—4] автором предложена аппроксимация классического нейрона морфологической структурой, методы для ее обучения, а также выполнено экспериментальное исследование ее точности для нейросетевых моделей различных архитектур, оценки вычислительной эффективности и выразительной способности предложенной структуры. В [5; 6] автор осуществил анализ вычислительной эффективности рассматриваемых алгоритмов на VLIW-платформах, выполнил их доработку и оценки производительности. В [7; 8] автор предложил методы квантования нейросетевых моделей, метод послойного преобразования и дообучения таких моделей и провел экспериментальную оценку их вычислительной эффективности. В [9] автору принадлежит идея разработанной аппаратной архитектуры и план проведения экспериментов. Исследование аппроксимаций функций активации биполярных морфологических моделей было выполнено и опубликовано в [10] без соавторства.

Публикации. Основные результаты по теме диссертации изложены в 10 печатных изданиях, из которых 1 работа издана в журнале, рекомендованном ВАК, 8 — в научных изданиях, индексируемых Web of Science и Scopus, 1 — в сборнике трудов конференции. Зарегистрировано 2 программы для ЭВМ.

Объем и структура работы. Диссертация состоит из введения, 3 глав, заключения. Полный объём диссертации составляет 138 страниц, включая 29 рисунков и 19 таблиц. Список литературы содержит 135 наименований.

Краткое содержание глав. Первая глава посвящена модели программно-аппаратного нейросетевого распознавания. В ней рассмотрены основные типы нейронов и слоев, используемые в современных нейросетевых моделях, и описаны вычислительные платформы, на которых может выполняться распознавание. Для этих платформ предложены способы оценки аппаратной сложности и вычислительной эффективности. Приведены различных методов повышения вычислительной эффективности нейросетевых моделей, а также сформулированы цель и задачи диссертационного исследования.

Во второй главе предложена аппроксимация классического нейрона: биполярный морфологический нейрон. Показан процесс построения сверточных и полносвязных слоев на основе биполярного морфологического нейрона, приведены оценки их вычислительной сложности для центральных процессоров и программируемых/специализированных логических интегральных схем. Описаны реализации вычислительных модулей для функций активации биполярного морфологического нейрона и выполнены оценки числа вентилей и латентности получающихся сверточных слоев. Рассмотрена модель вычислительного устройства для нейросетевых моделей предложенного типа.

Третья глава посвящена процессу обучения биполярных морфологических моделей. Показано, что прямое преобразование классических моделей к биполярному морфологическому виду, а также обучение стандартными методами не подходят для получения распознающих моделей. Предложен оригинальный метод послойного преобразования и дообучения, позволяющий успешно обучать биполярные морфологические нейросетевые модели и продемонстрирована его эффективность в задачах классификации и семантической сегментации с помощью глубоких нейронных сетей. Показано, что разработанный метод обобщается на широких класс аппроксимаций на примере целочисленных малобитных нейросетевых моделей. Приведено описание программного комплекса для моделирования биполярных морфологических сетей.

Приложение А содержит информацию о зарегистрированных программах для ЭВМ, в которых применяются результаты диссертационной работы.

Приложение Б содержит акты о внедрении результатов диссертационной работы.

Глава 1. Модели программно-аппаратного нейросетевого распознавания

1.1 Модели нейрона в задачах технического зрения

Основной структурной единицей нейросетевых моделей является *искусственный нейрон*. Первые искусственные нейроны имитировали процессы, протекающие в биологических нейронах – нервных клетках, из которых состоит нервная система живых существ и которые способны принимать, обрабатывать и передавать различные сигналы [11]. Математические модели, описывающие нервные клетки, известны еще с начала XX века и преимущественно состоят из систем дифференциальных уравнений, например, модель «интегрировать и сработать» и ее обобщения [12; 13], модель Ходжкина-Хаксли [14] и др.

Начиная с 60-х годов прошлого века исследователи начали активно создавать аппаратные модели нейронов, которые называли нейроморфными устройствами или нейромимами. Например, Л. Хармон [15] и Е. Льюис [16] предложили электрические цепи, позволяющие симулировать нейрон Ходжкина-Хаксли. Однако создание устройств с большим количеством нейронов, решающих осмысленные задачи, было в то время невозможно. Поэтому были созданы упрощенные модели нейрона и нейронных структур, воспроизводящие процессы обработки информации в живых системах. Именно эти модели легли в основу современных нейросетевых методов распознавания. Наиболее известной и используемой на практике моделью нейрона является нейрон Мак-Каллока-Питтса или классический математический нейрон, однако были предложены и другие модели, которые будут рассмотрены далее в этой главе.

1.1.1 Классическая модель математического нейрона

.

В 1943 году Уоррен Мак-Каллок совместно с Уолтером Питтсом опубликовали статью «A logical calculus of the ideas immanent in nervous activity» [17].

Они исследовали нейроны головного мозга с целью построения их математической модели, которая могла бы стать основой искусственного интеллекта. Построенный ими искусственный нейрон был бинарным, т.е. в зависимости от входных сигналов мог оказываться в возбужденном или невозбужденном состояниях. Как и биологические нейроны, он имел тело, соединенное синапсами с несколькими дендритами, по которым поступали входные сигналы, и один аксон, служивший выходом. Синапсы дендритов ослабляют или усиливают входные сигналы путем умножения на весовой коэффициент и передают результат в тело нейрона, которое было представлено в виде сумматора. При превышении этой суммой некоторого порогового значения нейрон переходил в возбужденное состояние. Нейрон Мак-Каллока-Питтса описывается следующим выражением:

$$f(\mathbf{x}) = \theta\left(\sum_{i=1}^{N} w_i x_i + w_0\right), \tag{1.1}$$

где $\mathbf{x}$ – вектор входных сигналов, $\mathbf{w}$ – вектор весов нейрона, θ – пороговая функция активации:

$$\theta(x) = \begin{cases} 1, \, x \geqslant 0, \\ 0, \, x < 0. \end{cases} \tag{1.2}$$

Из таких математических нейронов Мак-Каллок и Питтс построили простейшую нейронную сеть и показали, что она способна вычислять различные математические функции. Кроме того, они предлагали моделировать и явление самообучения, наблюдающееся в реальных нейронных сетях, путем изменения весовых коэффициентов в ответ на определенные последовательности входных сигналов.

Позднее эта модель была обобщена до искусственного нейрона, использующего произвольную функцию активации φ (см. Рис. 1.1). Именно эту модель часто называют классической моделью математического нейрона:

$$f(\mathbf{x}) = \varphi\left(\sum_{i=1}^{N} w_i x_i + w_0\right). \tag{1.3}$$

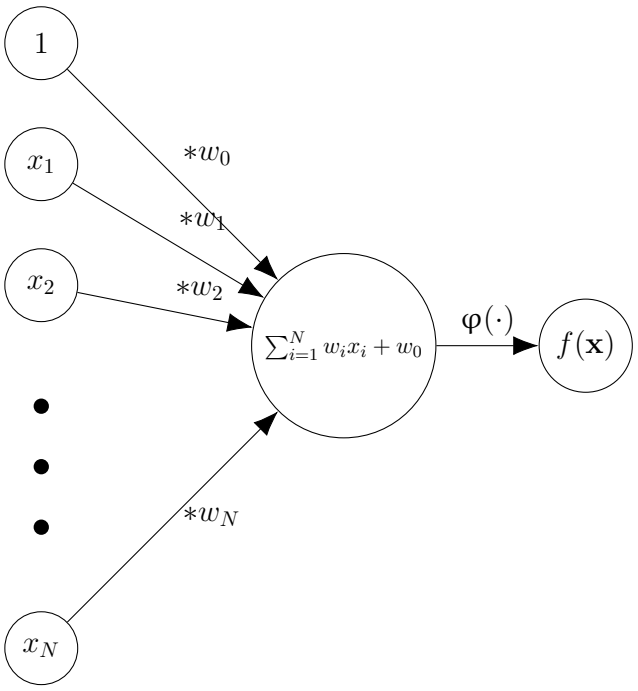


Рисунок 1.1 — Схема классического математического нейрона.

1.1.2 Модель морфологического нейрона

В 1990 году Г. Риттер вместе с коллегами предложил альтернативную модель нейрона и структуры нейронной сети, которые назвал морфологической моделью нейрона и морфологической нейронной сетью соответственно [18]. В морфологической модели тело нейрона выполняет не суммирование, а взятие максимума или минимума. Весовые коэффициенты влияют на входные сигналы не мультипликативно, а аддитивно. Далее найденное значение максимума или минимума сравнивается с порогом, и, таким образом, выход морфологического нейрона является бинарным. Морфологический нейрон можно описать формулой:

$$f(\mathbf{x}) = \theta\left(p \max_{i=1}^{N} r_i(x_i + w_i) + w_0\right), \tag{1.4}$$

где $\mathbf{x}$ — вектор входных сигналов, $\mathbf{w}$ — вектор весов нейрона, θ — пороговая функция активации, $r_i \in \{-1, 1\}$ отвечают за воздействие i-го входного сигнала на нейрон (возбуждение или торможение), а $p \in \{-1, 1\}$ определяет знак выходного сигнала. Морфологический нейрон показан на Рис. 1.2.

Такой нейрон оказывается более вычислительно эффективным, чем классический математический нейрон, поскольку операции сложения, вычитания и

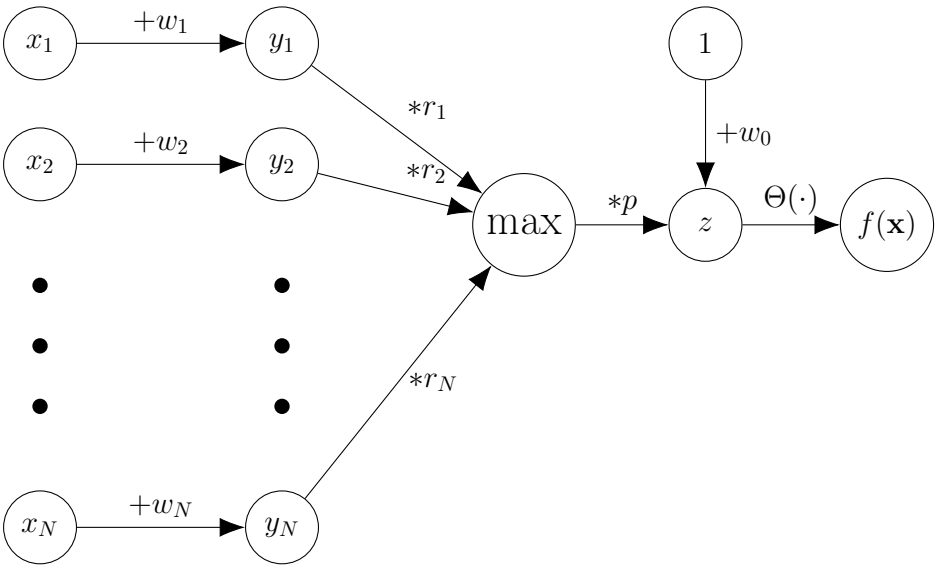


Рисунок 1.2 — Схема морфологического нейрона.

взятия максимума или минимума требуют меньшего числа логических вентилей для реализации, чем операция умножения. Это означает, что он является более энергоэффективным, и требует значительно меньшего времени для вычисления.

Дальнейшим развитием морфологического нейрона стала предложенная Риттером в 2003 году модель морфологического нейрона с дендритами [19]:

$$f(\mathbf{x}) = \theta\left(\min_{k=1}^{K} p_k \left(\min_{i,l}(-1)^{1-l}(x_i + w_{ik}^l)\right) + w_0\right), \tag{1.5}$$

где $\mathbf{x}$ — вектор входных сигналов, $\mathbf{w}$ — вектор весов нейрона, θ — пороговая функция активации, K — число дендритов, $l \in \{0, 1\}$, а $p \in \{-1, 1\}$ отвечает за знак выходного сигнала.

Дендриты позволяли более гибко управлять возбуждением и торможением нейрона, поскольку вычисляли несколько комбинаций входных сигналов с разными знаками и разными весовыми коэффициентами. Особенностью данной модели является то, что необходимое количество дендритов устанавливается во время обучения модели, т.е. дендриты «наращиваются» по мере надобности. Кроме того, в этой работе было доказано, что с помощью морфологических нейронных сетей с дендритами можно решить любую задачу классификации с любой заданной точностью. Однако требуемое количество нейронов при этом может быть крайне большим, что нивелирует преимущества простоты каждого отдельного нейрона.

Кроме того, эти морфологические модели оказались неспособны обеспечить качество распознавания, сравнимое с качеством распознавания более классических моделей. Только недавно исследования морфологических сетей возобновились: разрабатываются новые методы обучения для существующих моделей [20; 21], и рассматриваются способы применения этих моделей в реальных задачах, например, на примере расшифровки электроэнцефалограмм [22]. Также слои из морфологических нейронов с дендритами показывают многообещающие результаты в гибридных моделях, когда морфологические нейроны используются для извлечения признаков и входят в состав отдельных слоев сети [23].

1.1.3 Модель спайкового нейрона

Обе рассмотренные модели нейрона хорошо подходили для симулирования на вычислительных устройствах, однако не воспроизводили одно важное свойство биологического нейрона: импульсный характер его функционирования. При передаче реальных сигналов в нервной ткани, входными и выходными сигналами нейронов являются последовательности импульсов. Первой моделью спайкового нейрона была модель Алана Ходжкина и Эндрю Хаксли, предложенная в 1952 году. Они описали механизмы, лежащие в основе возникновения и распространения сигналов по аксону. Эта модель является одной из самых биологически точных моделей, и поэтому она довольно сложна: она содержит четыре дифференциальных уравнения и достаточно много параметров [24] и непригодна для задач искусственного интеллекта.

Самая известная модель импульсного нейрона была предложена французским физиологом Луи Лапиком в 1907 году. Эта модель называется «интегрировать и сработать» (англ. integrate-and-fire). Идея этой модели заключается в том, что, когда на вход подается некоторый сигнал, выходной сигнал возрастает до тех пор, пока не достигнет порогового значения, а потом резко сбрасывается, и процесс начинается сначала. Недостатком такой модели является то, что при линейном возрастании входного сигнала частота срабатывания нейрона неограниченно растет, что является некоторой идеализацией реального нейрона. Для устранения этого недостатка в модель вводится запрет

на слишком частые срабатывания. Вторым недостатком является то, что если входной сигнал не вызвал срабатывание нейрона, то значение выходного сигнала так и останется промежуточным между максимальным и минимальным. Для устранения последнего недостатка была разработана новая модель «интегрировать-и-сработать» с утечками (англ. leaky integrate-and-fire neuron). В этой модели вводится «сопротивление» нейрона, которое вызывает постепенную утечку, благодаря которой выходной сигнал постепенно сбросится, если он не достиг порога срабатывания [24].

С ростом вычислительной мощности компьютеров и технологий производства микрочипов возобновился интерес к построению устройств, симулирующих поведение реальных нейронов. В 2003 году Евгений Ижикевич разработал новый класс моделей нейронов, а именно спайковые модели. Нейроны в них описываются всего двумя простыми дифференциальными уравнениями. С одной стороны, они достаточно простые для реализации и вычислительно эффективные, а с другой стороны сохраняют биологическую точность модели Ходжкина-Хаксли [25].

Для симуляции такие нейронов и нейронных сетей из них был разработан отдельных класс вычислительных устройств – нейроморфные процессоры. Это процессоры, которые позволяют эффективно реализовать спайковые нейронные сети и отличаются крайне низким энергопотреблением. Например, в 2014 году компания IBM выпустила нейроморфный чип TrueNorth, который состоит из миллиона нейронов, которые могут образовывать 256 миллионов связей. В 2017 году компания Intel представила нейроморфный чип Loihi, который поддерживает обучение нейросетевых моделей на самом устройстве. В 2020 году был создан Akida neural processor с приблизительно 1.2 миллиона нейронов, которые могут образовывать 10 миллиардов связей.

Несмотря на многообразие моделей и активные исследования спайковых нейросетевых моделей, на сегодняшний день нет стабильных методов обучения, которые бы позволяли получить достаточно точные решения современных задач распознавания [26]. При этом работы в этой области активно ведутся и демонстрируют многообещающие результаты [27—29].

1.2 Архитектуры искусственных нейронных сетей

После создания нейрона Мак-Каллока-Питтса следующим принципиальным вопросом стала организация нейронов в единую сеть. Одним из вариантов стал персептрон, в котором отдельные нейроны были организованы в слои и каждый нейрон следующего слоя был связан с некоторыми нейронами предыдущего. Для обучения использовался метод коррекции ошибки (Ф. Розенблатт, [30]), а позднее метод обратного распространения ошибки с использованием различных метрик (Д. Румельхарт, [31]). Несмотря на критику и ограничения подобной модели (например, не-инвариантный относительно пространственного положения анализ образов), эта модель была способна решать ряд простых задач.

Другим вариантом организации нейронной сети стали когнитрон (1975) и неокогнитрон (1980) К. Фукусимы [32]. Неокогнитрон позволял выполнять инвариантный анализ входного изображения. Для этого использовались нейроны двух типов: S- («простые») и C- («сложные»). «Простые» нейроны были связаны с некоторой фиксированной областью входных сенсоров и выделяли некоторый конкретный признак. «Сложные» нейроны принимали на вход выходы «простых» нейронов и анализировали наличие признаков на всем изображении. Неокогнитрон имел иерархическую структуру и мог включать несколько слоев из «простых» и «сложных» нейронов. В результате он обладал способностью распознавать объекты на изображении вне зависимости от их положения и небольших деформаций. Несмотря на то, что приложения неокогнитрона ограничены только задачами распознавания изображений, в этих задачах он демонстрировал впечатляющие результаты.

Именно на основе неокогнитрона и модели классического математического нейрона основаны современные сверточные нейросетевые архитектуры, начиная с LeNet-5 [33] и AlexNet [34]. При этом интерес стали представлять не отдельные нейроны, а целые слои со специальными свойствами, которые затем комбинировались для создания сети. Таким образом, нейронная сеть может быть представлена как направленный граф, вершинами которого являются слои, а ребра обозначают поток данных.

1.2.1 Основные слои нейросетевых моделей

Рассмотрим подробнее устройство отдельных слоев современных нейросетевых моделей. Обычно к ним относят следующие слои:

– сверточный слой;

– полносвязный слой;

– слой субдискретизации;

– нормализующие слои.

Как правило сверточные слои применяются в моделях, которые анализируют визуальную информацию. Такие модели получают на вход изображение. Цифровое изображение представляет собой трехмерную матрицу, где две размерности – пространственные координаты, а третья – каналы, которые чаще всего содержат цветовую информацию. Сверточный слой с входом I размера $L \times M \times C$ и выходом O размера $L \times M \times F$ выполняет следующую операцию:

$$O(l,m,f) = \varphi\left(\sum_{c=1}^{C}\sum_{\Delta l=0}^{K-1}\sum_{\Delta m=0}^{K-1} I(l+\Delta l, m+\Delta m, c)\cdot \right. \\ \left. \cdot\, w(\Delta l, \Delta m, c, f) + b(f)\right),\ f=\overline{1,F}, l=\overline{1,L}, m=\overline{1,M}, \tag{1.6}$$

где F – число фильтров, C – число входных каналов, $K \times K$ – пространственные размерности фильтра, $L \times M \times C$ – размер входного изображения, w – набор сверточных фильтров, b – вектор коэффициентов сдвига. Будем считать, что границы изображения I корректно дополнены так, чтобы результат имел тот же размер. Обычно используются следующие функции активации φ:

– $\mathrm{ReLU}(x) = \max(x, 0)$;

– гиперболический тангенс $\tanh(x) = \dfrac{e^x - e^{-x}}{e^x + e^{-x}}$;

– сигмоидальная $\mathrm{sigmoid}(x) = \dfrac{e^x}{1 + e^x}$.

Иногда применение функций активации выделяют в виде отдельного слоя, чтобы подчеркнуть, что можно использовать разные функции в рамках одной нейронной сети. Кроме того, именно вычисление свертки в сверточных слоях обычно занимают больше всего времени в глубоких нейросетевых моделях, а никак не нелинейное преобразование промежуточных данных.

Полносвязные слои можно описать следующим образом:

$$P(l) = \varphi\left(\sum_m W(l,m)\cdot I(m) + b(m)\right), \tag{1.7}$$

где P — вектор выходных значений слоя, I — вектор входных значений слоя, W — двумерная матрица весовых коэффициентов, b — вектор смещения слоя, φ — функция активации. Для перехода от трехмерной структуры, являющейся выходом сверточных слоев, к одномерному вектору, способному служить входом полносвязного слоя, элементы структуры просто перечисляются в любом заранее заданном порядке. Как правило, используются те же функции активации, что и для сверточных слоев.

Слой субдискретизации с операцией максимума:

$$S(l,m,c) = \max_{\substack{0\leqslant\Delta l\leqslant w_x\\ 0\leqslant\Delta m\leqslant w_y}} I(l\cdot w_x + \Delta l, m\cdot w_y + \Delta m, c), \tag{1.8}$$

где $S(l,m,c)$ — выходные значения слоя субдискретизации, $I(l,m,c)$ — C-канальное входное изображения, w_x, w_y — размеры окна субдискретизации. В результате применения слоя ширина и высота выходного изображения уменьшаются. Кроме операции максимума также может использоваться усреднение (с весами или нет).

Слой нормализации:

$$N(x) = \gamma\frac{x-\mu}{\sigma} + \beta, \tag{1.9}$$

где μ и σ определяются как скользящие среднее и среднеквадратичное отклонение входных данных x, а γ и β – обучаемые параметры.

В задачах классификации изображений на выходе модели обычно располагается функция активации $\mathrm{softmax}(x_i) = \frac{e^{x_i}}{\sum_k e^{x_k}}$, которая вычисляет уверенности модели в каждом из возможных ответов.

1.2.2 LeNet-подобные нейросетевые архитектуры

Первый сверточный нейросетевой классификатор LeNet-5, архитектура которого позволила эффективно решать практические задачи распознавания

был предложен Яном Лекуном [33]. Он использовал два блока из свертки с сигмоидальной функцией активации и субдискретизации с взвешенным усреднением для извлечения признаков изображения. Далее полученный сигнал интерпретировался как одномерный и подавался на вход полносвязного персептрона, состоящего из одного слоя с гиперболическим тангенсом. На выходе сети применялись евклидовы радиально-базисные функции, по одной на каждый класс. Оригинальная архитектура LeNet-5 приведена на рисунке 1.3.

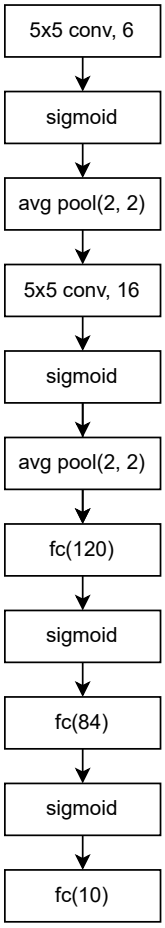


Рисунок 1.3 — Нейросетевая архитектура LeNet-5, где $n \times n$ conv, m — сверточный слой с m фильтрами размера n на n, sigmoid — сигмоидальная функция активации, avg pool(n, n) – слой усредняющей субдискретизации с окном n на n, fc(n) — полносвязный слой с n нейронами.

Такая архитектура сделала извлекаемые признаки основанными на локальной области изображения, они стали инвариантными к сдвигу, а благодаря тому, что в процессе свертки используются одни и те же весовые коэффициенты, общее число коэффициентов модели заметно снизилось. Структура из последовательности сверточных слоев и слоев, снижающих размерность, все

еще широко используется в современные нейросетевых моделях. Более поздние исследователи

- стали использовать для снижения размерности не только субдискретизацию с усреднением, но и субдискретизацию с максимумом или же свертки с нединичным шагом [34];
- стали использовать в качестве функций активации ReLU [34].

Нейросетевые архитектуры с последовательно исполняемыми блоками из сверточных слоев и слоев субдискретизации называют LeNet-подобными архитектурами.

Для повышения качества распознавания число таких блоков все увеличивалось и увеличивалось. Однако это открыло новую проблему: обучение моделей с большим количеством последовательных сверточных слоев с помощью метода обратного распространения ошибки не позволяет получить высокого качества распознавания из-за затухания градиента [35]. Для ее решения был предложен новый класс нейросетевых моделей – остаточные нейронные сети ResNet.

1.2.3 Семейство нейросетевых архитектур ResNet

ResNet – это современная модель, которая широко используется на практике в сложных задачах распознавания. Основной особенностью этой модели является использование остаточных сквозных соединений между блоками слоев, позволяющее решить проблему затухания градиента в глубоких нейронных сетях [36; 37]. Она хорошо масштабируется, то есть может включать в себя разное число остаточных блоков: малое число для простых задач и большее для более сложных.

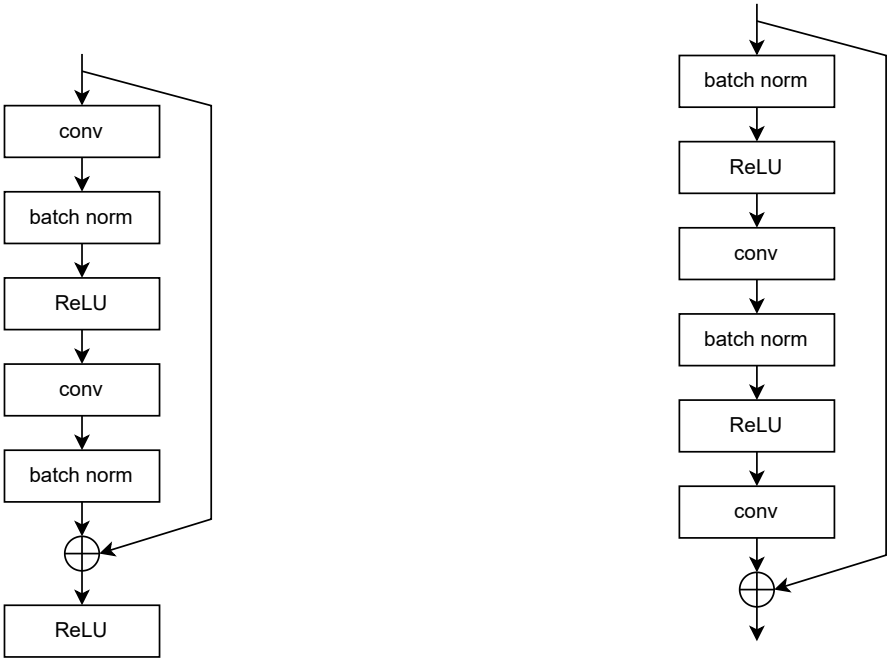
В первой версии архитектуры один остаточный блок состоял из последовательных сверточного слоя, нормализации, функции активации ReLU, еще одного сверточного слоя и нормализации. Далее вход x и выход блока $f_1(x)$ складываются и к ним применяется функция активации ReLU, то есть

$$y = \mathrm{ReLU}(x + f_1(x)). \tag{1.10}$$

Улучшенный остаточный блок $f_2(x)$ состоит из 6 последовательных слоев нормализации, активации ReLU, сверточного слоя, нормализации, активации ReLU и снова сверточного слоя, результат которых складывается с входным сигналом x:

$$y = x + f_2(x). \tag{1.11}$$

Оба остаточных блока проиллюстрированы на рисунке 1.4. При необходимости остаточные соединения могут включать в себя сверточный слой для изменения размерности входного сигнала, чтобы было возможно осуществить сложение с выходным сигналом.



а) б)

Рисунок 1.4 — Остаточные блоки а) первой версии, б) второй версии, где conv — сверточный слой, ReLU — функция активации, batch norm — слой нормализации.

1.2.4 Обучение нейросетевых моделей

Процесс получения параметров нейронной сети называется обучением. В задачах распознавания широко распространены алгоритмы обучения по прецедентам. Такие алгоритмы используют выборку данных (обучающую выборку)

для подбора коэффициентов нейронной сети путем минимизации некоторого функционала потерь $\mathcal{L}$. Будем называть выборкой набор пар $(\mathbf{x}, \mathbf{y})$, где $\mathbf{x}$ — распознаваемый объект, а $\mathbf{y}$ — верный ответ для этого объекта.

Тогда для нейронной сети $\mathcal{A}$ процесс обучения можно записать следующим образом:

$$w : \sum_i \mathcal{L}(\mathcal{A}(\mathbf{x}_i), \mathbf{y}_i) + \mathcal{R}(w) \to \min, \tag{1.12}$$

где $\mathcal{R}$ — регуляризирующий функционал. Регуляризация весов может использоваться при обучении нейросетевых моделей для предотвращения переобучения. Переобученная нейронная сеть хорошо решает задачу распознавания на данных из обучающей выборки, однако демонстрирует неудовлетворительные результаты на других данных. Так происходит потому, что обучение нейросетевой модели — плохо обусловленная оптимизационная задача. Однако регуляризация не решает проблему переобучения полностью и поэтому для контроля переобучения используется валидационная выборка, не пересекающаяся с обучающей. Критерием остановки обучения служит значительный рост значения функционала потерь на валидационной выборке. Финальная оценка качества нейронной сети происходит с помощью тестовой выборки, также не пересекающейся с обучающей и валидационной.

Для оценки качества решения задачи распознавания могут использоваться и дополнительные метрики помимо значения функции потерь. Например, в задачах классификации часто используется точность классификации: доля объектов тестовой выборки, на которых модель выдала верный ответ.

1.3 Модели вычислительного устройства

Одной из основных характеристик нейросетевой модели является ее вычислительная эффективность. Однако понятие эффективности нуждается в формализации, так как современные вычислительные устройства обладают довольно сложной архитектурой. Например, они могут поддерживать конвейеризацию, внеочередное исполнение инструкций, иерархическую систему кеширования данных и т.д. Из-за этого реальное время работы конкретной

нейронной сети будет сложным образом зависеть от числа ее параметров и архитектуры.

Нейросетевые модели могут работать на различных вычислительных платформах:

– на высокопроизводительных серверах, получающих данные с удаленных устройств; в этом случае сложность нейросетевой модели может варьироваться в широких пределах, однако при этом возникают задержки, связанные с каналами связи, а также вопросы к безопасности пересылаемых данных, которые могут содержать персональную, медицинскую или финансовую информацию, которая должна оставаться конфиденциальной;
– непосредственно на конечных устройствах; чаще всего это будут пользовательские рабочие станции, мобильные или встраиваемые устройства.

Особое значение вычислительная эффективность имеет для вычислительных платформ конечных устройств. Поэтому в данном разделе будет рассмотрена оценка вычислительной эффективности в рамках модели специализированной логической интегральной схемы, отвечающей вычислителям встраиваемых устройств, а также модели SIMD-процессора, отвечающей центральным процессорам мобильных устройств или рабочих станций.

1.3.1 Оценка вычислительной эффективности для специализированных логических интегральных схем

Базовая модель специализированной логической интегральной схемы (СЛИС) представляет собой логическую цепь, состоящую из логических вентилей.

Логический вентиль — это физическое устройство, которое реализует булеву функцию. Логическая цепь — это направленный ациклический граф, в котором все вершины, кроме входных, являются логическими вентилями [38]. Это означает, что логическая цепь вычисляет бинарную функцию

$$f: \quad B_n \to B_m, \;\; B \in \{0, 1\}$$

с m выходами от n переменных. С помощью логических цепей можно реализовать программы без циклов и ветвлений.

Основной характеристикой логических цепей являются размер и глубина. Размер логической цепи это число вершин в графе цепи, а глубина определяется как число ребер в наиболее длинном пути от любой из входных вершин к любой из выходных вершин. Размер логической цепи напрямую связан с числом вентилей, которое требуется для ее реализации, и, таким образом, может использоваться для оценки энергоэффективности и сложности получающегося устройства. Глубина цепи характеризует задержку прохождения сигнала, то есть время, необходимое для получения корректного сигнала на выходе при подаче нового входного сигнала. Задержку получения выходного сигнала также называют латентностью и чаще всего измеряют в тактах.

Теперь рассмотрим вычисление некоторой функции f, которая может быть представлена в виде логической цепи. Пусть F — логическая цепь, вычисляющая функцию f, в вершинах которой записаны операции $op \in \mathcal{M}$, где $\mathcal{M}$ — множество операций, реализованных с помощью логических вентилей. Вычислительную эффективность F будем задавать с помощью двух параметров:

1. Общего числа вентилей:

$$V_f = \sum_{op \in \mathcal{M}} N_{op} V_{op}, \tag{1.13}$$

   где N_{op} — число операций op в цепи F, V_{op} — число логических вентилей, необходимое для реализации операции op.

2. Общей латентности:

$$T = \sum_{d \in D(F)} T_d, \tag{1.14}$$

   где $D(F)$ — путь в графе F, обеспечивающий наибольшую суммарную латентность.

1.3.2 Оценка вычислительной эффективности для SIMD-процессора

Современные центральные процессоры являются сложными устройствами с вычислительными модулями, памятью для хранения данных и команд,

а также возможностями для реализации различных типов параллелизма. Для описания модели SIMD-процессора сначала рассмотрим модель конечного автомата и модель машины с произвольным доступом к памяти.

Конечный автомат

Конечный автомат – это вычислитель с памятью, в которой хранится его внутреннее состояние. Любое действие конечного автомата предпринимается с учетом его внутреннего состояния: его можно представить в виде логической цепи, которая принимает в качестве одного из входов внутреннее состояние и выдает целевую функцию и новое внутреннее состояние (см. рисунок 1.5).

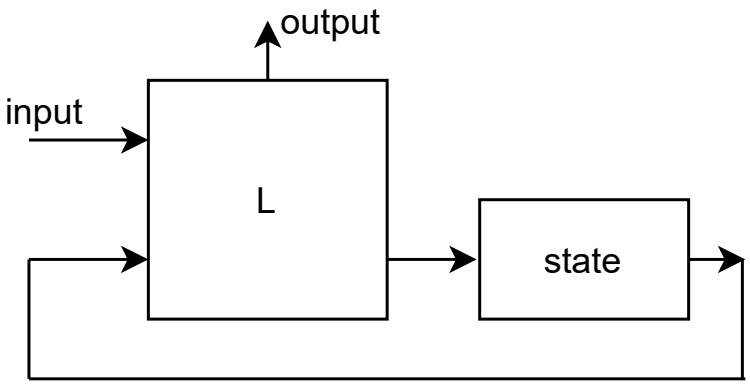


Рисунок 1.5 — Схема конечного автомата `L`, `input` — входные сигналы, `output` — выходные сигналы, `state` — внутреннее состояние.

Машина с произвольным доступом к памяти

Машина с произвольным доступом к памяти (RAM-машина) — модель вычислителя, отвечающая основным свойствам современных центральных процессором. RAM-машина может быть представлена как два синхронных связанных конечных автомата, центральный процессор (ЦП) и память с произвольным доступом (см. рисунок 1.6) [38]. Память с произвольным доступом (ППД) — это большое хранилище данных. Центральный процессор также оборудован собственным хранилищем данных: регистрами. Однако регистры обычно

имеют небольшой размер и их число обычно мало. ЦП может производить операции только над данными, расположенными в регистрах.

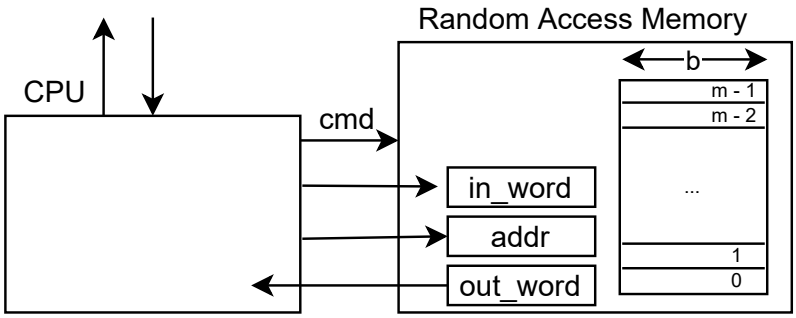


Рисунок 1.6 — Схема RAM-машины.

ЦП циклически декодирует и исполняет команды из памяти с произвольным доступом. Есть пять базовых типов команд:

1. арифметические и логические инструкции,
2. загрузка/выгрузка данных между ППД и регистрами ЦП,
3. инструкции переходов,
4. инструкции ввода/вывода,
5. инструкция остановки.

ППД оперирует словами – элементарными единицами данных. У нее есть три входных слова, в которых хранятся адрес, данные и команда, и одно выходное слово. Возможные варианты команды: чтение, запись, пустая команда. Во время чтения память вернет выходное слово (`out_word`), содержащее данные, хранящиеся по адресу (`addr`). При записи входное слово (`in_word`) будет записано по адресу (`addr`). Данный тип памяти называется памятью с произвольным доступом, потому что предполагает, что время доступа к данным не зависит от их адреса. Объем ППД обычно ограничен некоторой величиной $m = 2^r$, слова имеют фиксированную длину 2^b байт и адресуются r-битными числами.

Модель SIMD-процессора и оценка вычислительной эффективности

При моделировании современных центральных процессоров RAM-модель необходимо дополнить для учета возможностей внутреннего параллелизма. Чаще всего параллелизм на уровне данных поддерживается через SIMD-расширения, которые позволяют выполнить одну и ту же арифметическую операцию

над несколькими элементами данных (вектором данных) одновременно. В данной работе SIMD-инструкции рассматриваются как обычные арифметические операции, с учетом того, что предварительно необходимо обеспечить загрузку и подготовку не одной, а нескольких величин. Другие виды параллельности останутся за рамками данной работы.

Другим отличием ЦП является иерархическая структура памяти. Как правило, при обработке нейронных сетей, коэффициенты, входные данные и промежуточные значения не помещаются в регистры и даже L1-кэш, поэтому доступу в память необходимо уделить внимание при создании оптимальной реализации.

В данной работе оценку эффективности алгоритмов на ЦП предлагается выполнять путем подсчета числа арифметических инструкций каждого типа (с учетом возможности SIMD обработки данных) и последующего учета латентностей этих инструкций для этого семейства ЦП. Несмотря на то, что такая оценка не учитывает особенности доступа к памяти, она может быть использована для сравнения эффективности методов нейросетевой классификации, различающихся преимущественно в вычислительной части.

Рассмотрим вычисление функции f. Пусть F — логическая цепь, вычисляющая функцию f, в вершинах которой записаны операции $op \in \mathcal{M}$, $\mathcal{M}$ — множество SIMD-операций.

Определим вычислительную эффективность подсчета функции f как:

$$T = \frac{1}{S} \sum_{op \in \mathcal{M}} \frac{N_{op} T_{op}}{N_{op}^S}, \tag{1.15}$$

где N_{op} — число операций op в цепи F, S — число элементов, обрабатываемых SIMD-устройством, N_{op}^S — число SIMD-устройств для вычисления операции op, T_{op} — латентность вычисления операции op.

1.4 Методы повышения вычислительной эффективности нейросетевых моделей

Объем использования нейронных сетей на конечных устройствах неуклонно растет. Вычислителями конечных устройств, на которых осуществляется

распознавание, чаще всего являются центральные процессоры общего назначения или программируемые или специализированные логические интегральные схемы. Центральные процессоры имеют фиксированную архитектуру, которая не оптимизировалась под решение задач распознавания, поэтому разработчикам приходится решать задачу повышения эффективности использования доступных ресурсов. Программируемые логические интегральные схемы (ПЛИС) дают разработчику большую свободу в реализации вычислителя для конкретной нейросетевой модели и поиске оптимальной аппаратной архитектуры, однако они ограничены количеством доступных на плате вентилей того или иного типа и модулей памяти. Работы на эту тему обычно направлены на повышение эффективности использования вентилей и реализуют уже существующие нейросетевые архитектуры и методы обучения [39—42].

Когда вычислительная архитектура, оптимальная для некоторой нейросетевой модели или семейства моделей, уже определена, становится возможным создание СЛИС с необходимыми аппаратными характеристиками. Как правило, они имеют заданную архитектуру и не поддаются изменению со стороны разработчика. К таким устройствам относятся Google Tensor Processing Unit [43], Intel Vision Processing Unit [44], Huawei Ascend [45] и др.

Современные методы снижения вычислительной трудоемкости нейросетевых моделей можно разделить на две практически не пересекающиеся группы. Методы первой группы уменьшают число коэффициентов и вычислительных операций в сети, за счет чего она также может работать быстрее независимо от вида вычислителя. К ним можно отнести тензорные разложения сверток, обрезку моделей, дистилляцию знаний. Методы второй группы ориентируются на возможности конкретных вычислителей и заменяют одни вычислительные операции и структуры другими, которые могут быть оптимальнее реализованы и требуют меньше времени для подсчета. Например, малобитные вычисления эффективны как для центральных процессоров, так и для логических интегральных схем, а альтернативные эффективно-вычислимые модели отдельных нейронов/слоев предназначены в основном для логических интегральных схем.

Стоит отметить, что на практике методы из разных групп могут комбинироваться для достижения желаемой эффективности.

1.4.1 Тензорные разложения свертки

Основную вычислительную сложность в сверточных нейросетевых моделях составляет операция свертки входной карты признаков с фильтром в сверточных слоях. Для снижения вычислительной сложности свертки можно использовать тензорное разложение фильтра: представление его в виде композиции нескольких фильтров меньших размерностей. Поскольку обученные стандартными методами сверточные фильтры чаще всего не имеют точного тензорного разложения, для их представления в подобном виде нужны специальные методы [46—48].

Например, Ригамонти и др. [46] выполняют аппроксимацию сверточных фильтров набором композиций одномерных фильтров. Для получения такого набора авторы модифицировали процесс обучения с целью ограничить пространство всех сверточных фильтров только фильтрами, допускающими предлагаемое разложение и применили методы декомпозиции тензоров. Аналогично в [48] были предложены две другие модификации структуры сверточного слоя и методы обучения для них. При этом потери качества распознавания в результате аппроксимации не слишком велики и позволяют использовать полученные модели на практике. В [49] авторы предложили декомпозировать сверточный слой на этапе обучения, что позволяет использовать стандартные методы обучения. В дальнейшем были предложены разные подходы к декомпозиции слоев на основе разложения Такера [50; 51], канонического разложения тензоров [52; 53], разложения тензорного поезда [54; 55] и др. Особое внимание уделяется стабильности подобных структур и методам ее повышения [53].

1.4.2 Обрезка моделей

Одним из распространенных методов снижения избыточности нейросетевых моделей является обрезка – исключение определенных наименее значимых коэффициентов из модели. Чаще всего незначимые коэффициенты инициализируются нулями. Однако стоит отметить, что эта группа методов хорошо подходит для уменьшения числа весовых коэффициентов и снижения объема

занимаемой моделью памяти, но при этом зачастую не влияет на эффективность работы, так как убираемые коэффициенты могут быть расположены в разных частях модели.

Методы обрезки моделей можно разделить на две группы: статические и динамические. Статические методы модифицируют модель до начала ее использования на практике, в то время как динамические методы анализируют вычислительную сложность модели во время ее работы. Статические методы могут ориентироваться на величину весовых коэффициентов и исключать наименьшие коэффициенты, вносящие слабый вклад в работу сети [56; 57]. Другие методы, такие как Optimal Brain Damage [58] и Optimal Brain Surgeon [59], используют оценку информативности коэффициентов на основе гессиана функции потерь. Однако для глубоких нейронных сетей это влечет значительные вычислительные затраты, поэтому были предложены методы, вносящие дополнительный штраф в функцию потерь для того, чтобы большее число коэффициентов приблизилось к нулю и могло быть далее обрезано [60]. Такие методы могут обрезать не только отдельные весовые коэффициенты, но и целые сверточные фильтры, что позволяет повысить вычислительную эффективность модели [61; 62]. Большинство методов статической обрезки удаляют выбранные элементы безвозвратно, что, как правило, влечет за собой снижение качества. В таком случае можно дообучить обрезанную сеть [63], что однако требует доступа к обучающим данным и занимает дополнительное время.

Для того, чтобы не терять содержащуюся в исходной модели информацию, были разработаны методы динамической обрезки. Этот подход предполагает, что выбор слоев, фильтров или отдельных нейронов, которые не будут использоваться при запуске, осуществляется непосредственно во время работы сети. Как правило, в таком случае дообучение сети не требуется. Динамический выбор обрезаемых элементов производит компонент принятия решения. Например, это могут быть дополнительные связи, добавленные к исходной сети, или компоненты, оценивающие и изменяющие связи между слоями [64]. Такие компоненты принятия решения обучаются вместе с основной сетью стандартным методами. Кроме того, иногда используются сторонние нейросетевые модели для принятия решений [65; 66]. Они показывают хорошие результаты, однако обучаются с подкреплением, что требует значительных вычислительных ресурсов.

Динамическая обрезка может осуществляться на разных уровнях: на уровне отдельных каналов [64; 65], слоев [67] или целых блоков сети [68]. Стра-

тегии осуществления обрезки могут быть различными. Некоторые алгоритмы позволяют пропускать при вычислении выбранные слои или блоки слоев [68]. Есть стратегия, при которой в зависимости от входных данных каждый блок сети, который будет использован при запуске, выбирается из нескольких вариантов [66; 69]. В некоторых случаях компонент принятия решения может досрочно остановить вычисления, как на уровне каждого блока [70], так и всей модели в целом [67]. Основным недостатком подхода динамической обрезки является то, что все вычисления, необходимые для выбора обрезаемых элементов, производятся непосредственно во время запуска. Это накладывает дополнительные требования на вычислитель, на котором осуществляется запуск.

1.4.3 Малобитные нейронные сети

Малобитные нейронные сети хранят коэффициенты и используют для вычислений типы данных малой разрядности, часто целые. Таким образом вычисления выполняются быстрее, чем в типе данных с единичной точностью. Процесс преобразования вещественных коэффициентов к целым числам с несколькими нормировочными параметрами называется квантованием. Широко используются нейронные сети с 8-битными коэффициентами, для них разработан ряд библиотек для различных вычислительных платформ [71—73]. При этом разрабатываются как системы, целиком использующие целочисленные вычисления и фиксированную схему квантования [74], так и системы смешанной точности. Например, Кай и Васконселос [75] отмечают, что использование единого способа квантования для всех фильтров сети неоптимально и предлагают выполнить поиск достаточной точности. Они показали, что это позволяет заметно улучшить результаты классификации по сравнению с однородно квантованными сетями.

Кроме того, ведутся исследования по дальнейшему снижению разрядности. Поскольку при стандартных подходах к квантованию коэффициентов получающаяся аппроксимация не дифференцируема, прямая конверсия сопряжена со снижением качества распознавания, первоочередное значение приобретают методы повышения качества. В [76] предложили подход к поиску весов, который позволил улучшить качество для большого набора комбина-

ций разрядностей (1, 2, 4 бита для коэффициентов и 1, 2, 4, 8, 32 бита для промежуточных значений). Чжуан и др. [77] дополнили сеть вспомогательным модулем полной точности на время обучения. Этот метод позволил повысить качество классификации или поиска объектов на выборках ImageNet, CIFAR-100 и COCO. В [78] была предложена модель, использующая троичные веса, а также метод обучения для нее. Она не только позволила избавиться от операции умножения, но и дала возможность контролировать разреженность сети, т.е. количество нулевых коэффициентов.

Особенно популярны модели с бинарными коэффициентами, поскольку они могут быть реализованы с меньшим количеством умножений (или без умножений вообще). В ряде задач их использование практически не приводит к потере качества распознавания по сравнению с моделями с вещественными коэффициентами одинарной точности [79—82]. Подобные решения обеспечивают высокую скорость исполнения нейросетевых моделей в конкретных задачах, однако имеют ограниченную область применения: чаще всего они подходят только для разработки программно-аппаратных платформ, которые могут использовать специализированные вычислители на базе FPGA или ASIC.

При этом есть также работы, посвященные использованию малобитных вещественных типов данных в нейросетевых моделях [83—85]. Основное преимущество в этом случае заключается в повышении точности вычислений за счет того, что вещественное представление ограничивает относительную ошибку. Такие модели позволяют даже выполнять обучение в 8-битном типе данных с плавающей точкой, однако также нуждаются в аппаратной поддержке для практического использования.

Существует и совсем другой подход к вычислению нейросетевых моделей с использованием целых чисел, при котором выполняется квантование и приближенное вычисление матричного произведения [86]. В рамках этого подхода строки левой матрицы подвергаются кодированию, при котором строка разбивается на подстроки и каждой из них ставится в соответствие набор ближайших векторов из нескольких кодовых наборов. На основе столбцов правой матрицы создаётся таблица предподсчета. Далее с помощью функции агрегации, которая чаще всего представляет собой просто суммирование нужных значений из таблицы вычисляется приближенный результат произведения матриц. При этом применяется линейное квантование, чтобы хранить и использовать при вычислениях только целые числа. Функции кодирования могут быть различными,

например, в [87] авторы предложили вычислительно-эффективную обучаемую функцию кодирования, позволяющую легко использовать векторные расширения центральных процессоров. Кроме того, они реализуют вычисления в 8-битном типе данных, используя усреднение вместо суммирования. Однако данный метод накладывает ограничения на размеры матриц: число строк левой матрицы должно значительно превосходить число столбцов левой и правой матриц. Из-за этого область его применения ограничена лишь полносвязными слоями. Кроме того, совместное обучение и весовых коэффициентов нейросетевых моделей, и непосредственно функции кодирования затрудняется тем, что эта функция недифференцируема, поэтому методы для преобразования моделей к подобному виду без снижения точности только предстоит разработать.

1.4.4 Неклассические модели слоев или нейронов

Также активно разрабатываются модификации отдельных слоев нейронных сетей с целью снижения их вычислительной сложности или повышения качества распознавания. Основная идея подобных методов – заменить умножения на менее вычислительно сложные операции. Например, в модели DeepShift авторы обучили сеть использовать битовый сдвиг вместо умножения [88]. Битовый сдвиг равноценен умножению на степень двойки и может быть эффективно реализован вычислителем. В [89] авторы предложили модель MConv сверточного слоя, который вычисляет операцию псевдоэрозии или псевдодилатации в скользящем окне, построенные на основе контрагармонического среднего. Подобные слои были использованы вместо сверточных при распознавании десятичных цифр и не привели к потере качества. В [90] были предложены модели SMorph и LMorph слоев, вычисляющих псевдоморфологические операции в скользящем окне, и показано, что их можно обучить для представления широкого класса признаковых карт.

Другим семейством неклассических моделей слоя стали log-sum-exp модели, в которых в сверточных слоях применяется соответствующая последовательность: логарифмирование, суммирование с фильтром свертки, потенцирование [91]. Эта модель замечательна тем, что позволяет использовать методы выпуклой оптимизации при обучении. Далее для повышения

качества были разработаны двухветочные log-sum-exp сети, в основе каждой ветки которых находился log-sum-exp слой, а выход представлял собой разность двух результатов веток. Для них в [92] было доказано, что нейронная сеть с нейронами такого вида может аппроксимировать любые непрерывные функции над выпуклыми компактными множествами с заданной точностью. При этом, поскольку слой имеет вид разности двух выпуклых функций, и для него доступен математический аппарат теории оптимизации. Однако подобные модели все же уступали в качестве классическим и не нашли широкого практического применения.

Чен и др. в [93] предложили архитектуру AdderNet, в которой модифицировали сверточные слои так, чтобы в них использовалась L_1-норма. Это значительно упрощает вычисления в сверточных слоях, в то время как умножения в других слоях остаются. Для подобных сетей был предложен оригинальный метод обучения, который все еще не позволил добиться качества аналогичных классических сетей. Однако в [94] предложили способ на основе дистилляции знания, который позволил обучить сверточные слои с L_1-нормой так, что результирующая сеть работает практически без потери качества на CIFAR-10, CIFAR-100 и ImageNet, а в [95] предложили подход с постепенной аппроксимацией L_1-нормы в процессе обучения. Также было экспериментально показано, что AdderNet может успешно применяться в задаче построения изображений сверхвысокого разрешения [96], а также в задаче детектирования объектов [97] при некоторой модификации процесса обучения. Несмотря на это, на компактных нейросетевых архитектурах, применяющихся на конечных устройствах, эта модель все еще уступает по качеству классическим, и поэтому может применяться только в некоторых частях сети [98]. Несмотря на это, для AdderNet активно разрабатываются высокопроизводительные ПЛИС и СЛИС имплементации [99; 100]. Позднее Шен и др. [101] модифицировали модель AdderNet, применив аппроксимацию умножения на основе четверть-квадратичного умножителя [102], которая позволила им реализовать слои, использующие только операцию сравнения, взятия модуля и знака величины. Такие модели вдвое более эффективны по числу вентилей, чем оригинальный AdderNet и показали большую стабильность в процессе обучения, однако уступили AdderNet по качеству в задаче классификации на CIFAR10. Другая модель, в которой вместо умножения используется квадрат разности, была предложена в [103]. Она представляет собой некоторый компромисс между вы-

числительно-эффективной моделью не содержащей умножений, но уступающей по качеству классической и требующей специальных методов обучения, и собственно классическими моделями.

1.5 Выводы по главе 1. Задачи диссертационного исследования

В данной главе последовательно рассмотрено устройство нейросетевых моделей, начиная с моделей отдельного нейрона, моделей слоев разных типов и заканчивая нейросетевыми архитектурами, которые могут применяться в практических задачах распознавания. Эти модели состоят из нескольких блоков «сверточный слой-функция активации-слой субдискретизации» в случае LeNet-подобных моделей или нескольких блоков из сверточных слоев, функций активации и слоев нормализации с остаточными соединениями в случае ResNet-подобных архитектур.

На практике распознающие модели чаще всего используются на высокопроизводительных серверах или конечных устройствах, где вычислителем выступают центральные процессоры или специализированные интегральные схемы. Наиболее актуально повышение вычислительной эффективности нейронных сетей именно для конечных устройств, поэтому в работе представлены способы оценки вычислительной эффективности для этих платформ.

Далее был проведен анализ уже существующих методов снижения вычислительной трудоемкости нейросетевых моделей. Их можно разделить на две практически не пересекающиеся группы. Методы первой группы уменьшают число коэффициентов и вычислительных операций в сети, за счет чего она также может работать быстрее независимо от вида вычислителя. К ним можно отнести тензорные разложения сверток, обрезку моделей, дистилляцию знаний. Методы второй группы ориентируются на возможности конкретных вычислителей и заменяют одни вычислительные операции и структуры другими, которые могут быть оптимальнее реализованы и требуют меньше времени для подсчета. Например, малобитные вычисления эффективны как для центральных процессоров, так и для логических интегральных схем, а альтернативные эффективно-вычислимые модели отдельных нейронов/слоев предназначены в

основном для логических интегральных схем. При этом методы из разных групп могут комбинироваться для достижения желаемой эффективности.

Однако при этом все еще нет простого и надежного метода создавать нейросетевые модели для конечных устройств так, чтобы они могли работать в режиме реального времени и не представляли для них значимой вычислительной нагрузки. Существуют достаточно эффективные решения частных задач, однако в общем случае они не позволяют достичь баланса между желаемым качеством распознавания и скоростью работы. Поэтому целью данной работы является разработка и исследование вычислительно-эффективных аппроксимаций существующих нейросетевых моделей, методов их обучения и оптимизации их вычисления на существующих и перспективных вычислителях.

Для достижения этой цели были поставлены следующие задачи:

1. Разработать методы аппроксимации вычислительно-интенсивных частей нейросетевых моделей и исследовать их точность.
2. Оценить вычислительную эффективность на различных платформах.
3. Разработать методы обучения предложенных аппроксимирующих структур.
4. Провести экспериментальную оценку точности предложенных методов для нейросетевых классификаторов различных архитектур.
5. Разработать комплекс программ, позволяющий моделировать аппроксимацию нейросетевых моделей, обучение полученных структур и проверку результирующего качества работы.

Глава 2. Биполярные морфологические нейросетевые модели

В этой главе автор предлагает новый метод снижения вычислительной и аппаратной сложности нейросетевых моделей за счет аппроксимации классического математического нейрона.

Классический математический нейрон является основной вычислительной единицей современных нейронных сетей и задается выражением (1.3):

$$f(\mathbf{x}) = \varphi\left(\sum_{i=1}^{N} w_i x_i + w_0\right),$$

где $\mathbf{x}$ — вектор входных сигналов, $\mathbf{w}$ — вектор весов нейрона, $\varphi(\cdot)$ — функция активации, N — длина входного и весовых векторов.

Такие нейроны используются в сверточных и полносвязных слоях и,как правило, именно на них приходятся наибольшее число операций в нейронной сети. Если предполагать, что нейроны организованы в слои, то число операций активации (вычисления функции $\varphi(\cdot)$) пропорционально длине выходного сигнала, а число операций сложения и умножения — произведению длин входного и выходного сигналов. Поэтому несмотря на то, что активация может быть представлена достаточно сложной для вычисления функцией (например, сигмоидальной), самой трудозатратной операцией классического математического нейрона является умножение за счет кратно большего числа повторений.

Основная идея предложенного автором метода снижения вычислительной сложности заключается в построении аппроксимированных нейронов без операций умножения. На первом этапе аппроксимации отдельно рассматриваются положительные и отрицательные значения коэффициентов и входных сигналов нейрона, формируя отдельные вычислительные пути для каждой комбинации знаков коэффициентов и входов. Данное действие можно интерпретировать как явное моделирование процессов возбуждения и торможения, причем каждый выходной сигнал может вызывать либо возбуждающий, либо тормозящий эффект. Такое поведение напоминает поведение *биполярных нейронов* в биологии. Как правило, такие нейроны отвечают за восприятие и встречаются, например, в сетчатке глаза [104].

На втором этапе аппроксимации операции умножения и сложения на каждом вычислительном пути заменяются на операции сложения и взятия

максимума соответственно. Таким образом, полученный нейрон использует *морфологические* операции [105; 106].

Предложенную аппроксимированную модель нейрона автор назвал *биполярным морфологическим нейроном*. В данной главе автор рассматривает точность и выразительную способность таких нейронов, теоретически показывая, что они могут эффективно использоваться вместо классических, исследует их вычислительную сложность, а также преимущества и недостатки для реализации на различных устройствах. На основе выявленных недостатков автор дорабатывает предложенную модель и демонстрирует финальную версию, которая демонстрирует более высокую эффективность, чем оригинальная.

2.1 Биполярный морфологический нейрон

Впервые идею и модель биполярного морфологического (БМ) нейрона автор представил в докладе «Bipolar morphological neural networks: convolution without multiplication» [4] и далее подробно исследовал в статье «Bipolar Morphological Neural Networks: Gate-Efficient Architecture for Computer Vision» [1]. Рассмотрим его построение.

Основной операцией классического математического нейрона является вычисление взвешенной суммы $\sum_{i=1}^{N} x_i w_i$, где $\mathbf{x}$ — вектор входного сигнала, $\mathbf{w}$ — вектор весов нейрона, а N — длина входа. Рассмотрим случай $x_i \geqslant 0$ и $w_i \geqslant 0$ для $i = 1, \ldots, N$ и введем новые обозначения:

$$\begin{aligned} \sum_{i=1}^{N} x_i w_i = \exp\{\ln \sum_{i=1}^{N} x_i w_i\} = (1+k) \exp \max_i (\ln x_i + \ln w_i) = \\ = (1+k) \exp \max_i (y_i + v_i), \end{aligned} \tag{2.1}$$

где $y_i = \ln x_i$ это новые входные значения, $v_i = \ln w_i$ — новые весовые коэффициенты,

$$k = \frac{\sum\limits_{j=1}^{N} x_j w_j}{M} - 1, \tag{2.2}$$

$$M = \max_j (x_j w_j). \tag{2.3}$$

Величина M обозначает максимальное слагаемое, возникшее в ходе вычислений, а k — степень его доминирования. При $k << 1$ максимальный вклад в выходной сигнал будет вносить именно оно. Введем следующую аппроксимацию:

$$\sum_{i=1}^{N} x_i w_i \approx \exp \max_i (y_i + v_i), \tag{2.4}$$

Теперь рассмотрим взвешенную сумму без ограничения диапазона входного сигнала и весовых коэффициентов:

$$\sum_{i=1}^{N} x_i w_i = \sum_{i=1}^{N} p_i^{++} x_i w_i - \sum_{i=1}^{N} p_i^{+-} x_i |w_i| - \sum_{i=1}^{N} p_i^{-+} |x_i| w_i + \sum_{i=1}^{N} p_i^{--} |x_i| |w_i|, \tag{2.5}$$

где

$$p_i^{++} = \begin{cases} 1, \text{ если } x_i \geqslant 0 \text{ и } w_i \geqslant 0, \\ 0, \text{ иначе,} \end{cases}$$

$$p_i^{-+} = \begin{cases} 1, \text{ если } x_i < 0 \text{ и } w_i \geqslant 0, \\ 0, \text{ иначе,} \end{cases}$$

$$p_i^{+-} = \begin{cases} 1, \text{ если } x_i \geqslant 0 \text{ и } w_i < 0, \\ 0, \text{ иначе,} \end{cases}$$

$$p_i^{--} = \begin{cases} 1, \text{ если } x_i < 0 \text{ и } w_i < 0, \\ 0, \text{ иначе.} \end{cases}$$

Для каждого слагаемого можно считать, что входные значения и весовые коэффициенты неотрицательны и могут быть приближены с помощью аппроксимации (2.4).

Таким образом получим выражение для БМ нейрона:

$$\begin{aligned} f_{BM}(\mathbf{x}, V, v) = \varphi \Bigg(& \exp \max_{j=1}^{N} (\ln x_j^+ + v_j^+) - \exp \max_{j=1}^{N} (\ln x_j^+ + v_j^-) - \\ & - \exp \max_{j=1}^{N} (\ln x_j^- + v_j^+) + \exp \max_{j=1}^{N} (\ln x_j^- + v_j^-) + v_0 \Bigg), \end{aligned} \tag{2.6}$$

$$x_j^+ = \begin{cases} x_j, & x_j \geqslant 0, \\ 0, & x_j < 0, \end{cases}$$
$$x_j^- = \begin{cases} -x_j, & x_j < 0, \\ 0, & x_j \geqslant 0, \end{cases} \tag{2.7}$$

где $\mathbf{x}$ — вектор входных значений длины N, v^+, v^- — векторы весовых коэффициентов длины N, v_0 — смещение, $\varphi(\cdot)$ — нелинейная функция активации. Будем считать, что $\ln 0 = -\infty$ и заменим его достаточно большим отрицательным числом в ходе практических вычислений, а $\exp(-\infty) = 0$ и не будем учитывать такие значения в ходе практических вычислений.

Структура БМ нейрона показана на рисунке 2.1. Функция max позволяет отбрасывать отрицательные значения и сформировать четыре ветки вычислений для различных комбинаций знаков входных значений и весовых коэффициентов. Далее выполняется логарифмирование и основная морфологическая операция внутри слоя. Ее результаты потенцируются и аккумулируются для получения выходного значения.

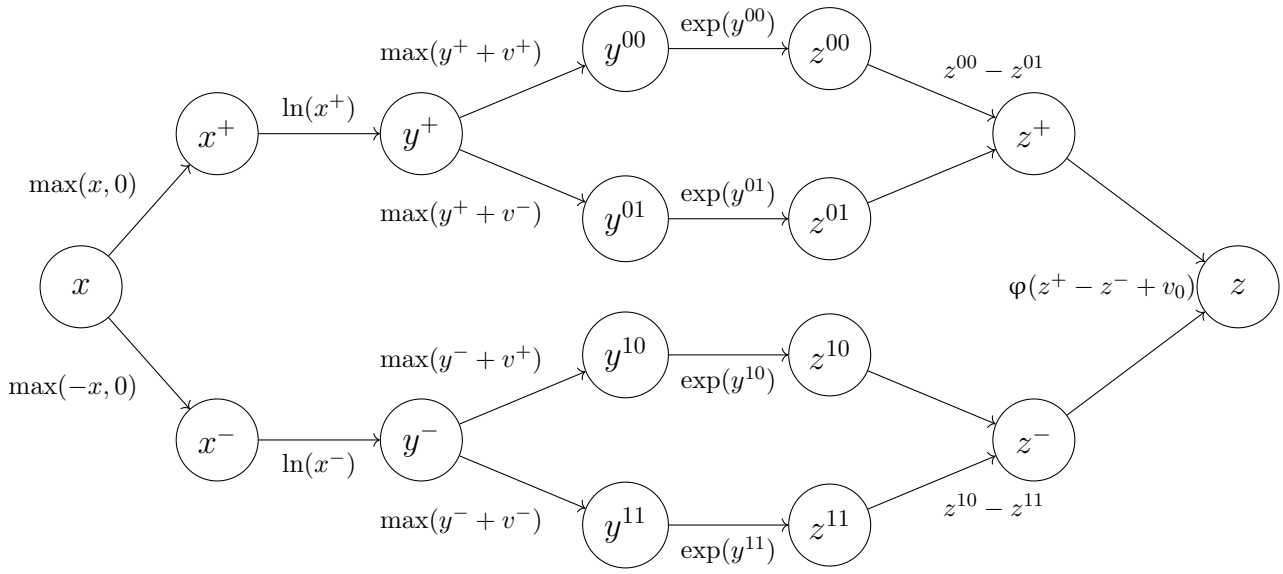


Рисунок 2.1 — Структура БМ нейрона с вектором входных значений x, весовыми коэффициентами v^+, v^-, v_0 и вектором выходных значений z.

2.2 Точность и выразительная способность БМ нейрона

В предыдущем разделе БМ нейрон рассматривался как аппроксимация классического нейрона. Однако приближение (2.4) будет достаточно точным только в случае $k \ll 1$. Для настоящего нейрона k может принимать значения в диапазоне от 0 до $N-1$, где N — число входов нейрона. В лучшем случае в сумме (2.2) будет только одно ненулевое слагаемое и тогда $k = 0$, а в худшем случае все слагаемые будут совпадать с максимальным значением и $k = N - 1$. В этом случае реальное значение выражения (2.4) будет в N больше аппроксимированного. Несмотря на это, в работе «Bipolar Morphological Neural Networks: Gate-Efficient Architecture for Computer Vision» [1] автор показал, что БМ нейронные сети уже имеют такую же выразительную способность, как и классические многослойные персептроны. Приведем формальное доказательство.

Теорема 1. *Любая непрерывная на компакте функция N переменных $f(x_1, ..., x_N)$ может быть равномерно приближена с любой заранее заданной точностью $\varepsilon > 0$ некоторой нейронной сетью, состоящей только из БМ нейронов.*

Прежде чем приступить к доказательству, напомним определения компакта и равномерного приближения.

Определение 1. *Компактом будем называть ограниченное замкнутое множество точек в пространстве $\mathbb{R}^N$.*

Определение 2. *Будем говорить, что функция $g(x)$ равномерно приближает $f(x)$ на компакте C с точностью $\varepsilon > 0$, если*

$$\forall x \in C : |f(x) - g(x)| < \varepsilon. \tag{2.8}$$

Доказательство. Рассмотрим одномерный случай. Пусть функция одного переменного $f(x)$ определена и непрерывна на отрезке $[\alpha, \beta]$.

Построим нейронную сеть из БМ нейронов с одном входом, на которое подается значение x. Пусть на первом слое расположено $2n$ нейронов и первые n из них вычисляют значения

$$\xi_i^+ = x + a_i^+,\ i = 1, \ldots, n, \tag{2.9}$$

где $a_i^+ \in \mathbb{R}$, $i = 1, \ldots, n$ — произвольные действительные числа. Покажем, что это возможно. Поскольку у таких нейронов всего один вход, у них 3 весовых коэффициента: v^+, v^-, v_0. Положим $v^+ = 0$, $v^- = -\infty$, $v_0 = a_i^+$. Тогда согласно (2.6)

$$\xi_i^+ = \varphi(\exp\ln x^+ - \exp\ln x^- + a_j^+) = x^+ - x^- + a_i^+ = x + a_i^+, \tag{2.10}$$

при тождественной функции φ.

Следующие n нейронов будут вычислять

$$\xi_i^- = -x + a_i^-,\ i = 1, \ldots, n, \tag{2.11}$$

где $a_i^- \in \mathbb{R}$, $i = 1, \ldots, n$ — произвольные действительные числа. У этих нейронов также всего один вход и 3 весовых коэффициента: v^+, v^-, v_0. Положим $v^+ = -\infty$, $v^- = 0$, $v_0 = a_i^-$. Тогда согласно (2.6)

$$\xi_i^- = \varphi(-\exp\ln x^+ + \exp\ln x^- + a_j^-) = -x^+ + x^- + a_i^- = -x + a_i^-, \tag{2.12}$$

при тождественной функции φ.

Рассмотрим второй слой сети. На нем будет n нейронов, каждый из которых будет иметь два входа ξ_i^+ и ξ_i^- и вычислять следующее выражение:

$$\eta_i = \theta(\eta_i'), \tag{2.13}$$

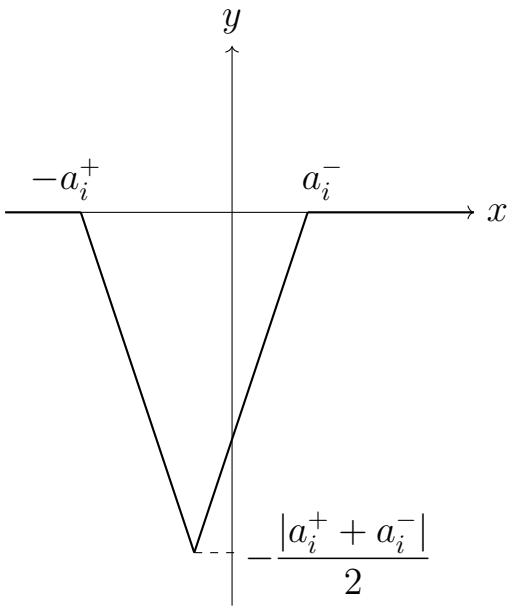
где

$$\begin{aligned}\eta_i' =& \exp\max(\ln\max(\xi_i^+, 0), \ln\max(\xi_i^-, 0)) - \\ &- \exp\max\left(\ln\max(-\xi_i^+, 0), \ln\max(-\xi_i^-, 0)\right) - (a_i^+ + a_i^-) = \\ &= \max(\max(x + a_i^+, 0)), \max(-x + a_i^-, 0) - \\ &- \max(\max(-x - a_i^+, 0), \max(x - a_i^-, 0)) - (a_i^+ + a_i^-).\end{aligned}$$

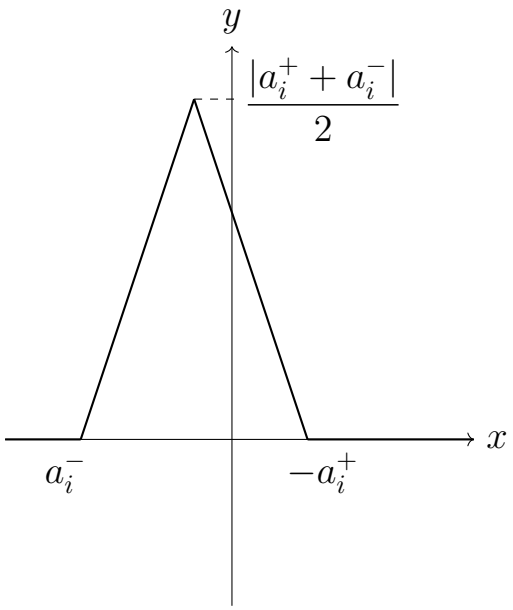
При этом $v_1^+ = v_2^+ = 0$, $v_1^- = v_2^- = -\infty$, $v_0 = -(a_i^+ + a_i^-)$. Результирующее выражение для η_i' описывает отрицательный треугольный импульс с основанием $(-a_i^+, a_i^-)$ при $a_i^- > -a_i^+$ и положительный с основанием $(a_i^-, -a_i^+)$ в противном случае как показано на рисунке 2.2. Высота этого импульса составляет $\dfrac{|a_i^+ + a_i^-|}{2}$.

Определим θ как пороговую функцию активации:

$$\theta(x) = \begin{cases} 1, x > 0, \\ 0, x \leqslant 0. \end{cases} \tag{2.14}$$



а) $a_i^+ > -a_i^-$



б) $a_i^+ < -a_i^-$

Рисунок 2.2 — Треугольные импульсы, описываемые нейронами второго слоя η_i.

Тогда выходом второго слоя η_i будут прямоугольные импульсы. По первой теореме Вейерштрасса $f(x)$ ограничена на $[\alpha, \beta]$ [107]. Так как БМ нейрон способен сдвигать результат на постоянную величину коэффициентом v_0, то при тождественной функции φ без ограничения общности можно считать, что $f(x) > 0$ на всей области определения. Благодаря этому можно рассматривать достаточно рассматривать только положительные импульсы.

Далее на следующем слое можно составить аппроксимацию функции $f(x)$, используя прямоугольные импульсы. Поскольку в качестве области определения функции f рассматривается компакт и функция непрерывна на нем, по теореме Кантора-Гейне она будет равномерно непрерывна на нем:

$$\forall \varepsilon \ \exists \delta(\varepsilon) > 0 : \forall |x_1 - x_2| < \delta \to |f(x_1) - f(x_2)| < \varepsilon. \tag{2.15}$$

Положим $\delta_1 = \delta(\varepsilon)$ и $n = \lceil (\beta - \alpha)/\delta_1 \rceil$. Пусть $\delta = \min((\beta - \alpha)/n, \delta_1)$. Теперь введем последовательность $x_0 = \alpha$, $x_1 = \alpha + \delta, ..., x_n = \beta$. Возьмем $a_i^+ = -x_i$, $a_i^- = x_{i-1}$, где $i = 1, ..., n$. На третьем слое будет один БМ нейрон, вычисляющий:

$$\zeta = \exp \max_{i=1}^{n} (\ln \max(\eta_i, 0) + \ln f(x_i)) = \max_{i=1}^{n} f(x_i)\eta_i. \tag{2.16}$$

Оценим отклонение $\zeta(x)$ от $f(x)$ в произвольной точке $x \in [\alpha, \beta]$. Пусть $x \in [x_k, x_{k+1}]$ для некоторого k (см. рисунок 2.3).

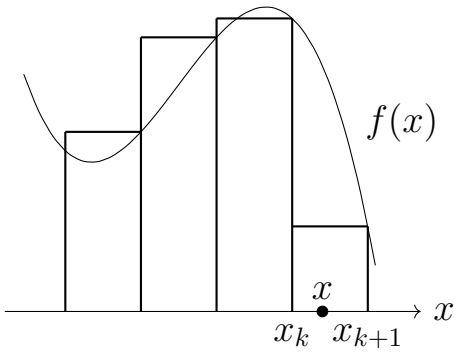


Рисунок 2.3 — Кусочно-постоянная аппроксимация $f(x)$ функцией $\zeta(x)$.

$$|f(x) - \zeta(x)| = |f(x) - f(x_{k+1})| < \varepsilon, \tag{2.17}$$

так как $|x - x_{k+1}| < \delta$. Таким образом, $\zeta(x)$ равномерно приближает $f(x)$ на отрезке $[\alpha, \beta]$.

Доказательство для функции N переменных можно построить аналогичным образом, рассмотрев сеть из $2Nn$ нейронов на первом слое, Nn на втором слое и единственного нейрона на последнем слое. □

2.3 Вычислительная сложность БМ сетей

Теперь сравним число арифметических операций разных типов в БМ и классических слоях нейронных сетей. Впервые такую оценку автор привел в докладе «ResNet-like Architecture with Low Hardware Requirement» [3]. Наибольший интерес представляют сверточные и полносвязные слои, как слои, использующие классические математические нейроны, которые могут быть приближены и заменены на биполярные морфологические.

Рассмотрим классический сверточный слой с входом $I_{L\times M\times C}$ и выходом $O_{L\times M\times F}$. Он выполняет следующую операцию:

$$\begin{aligned} O(l,m,f) =& \varphi\left(\sum_{c=1}^{C}\sum_{\Delta l=0}^{K-1}\sum_{\Delta m=0}^{K-1} I(l+\Delta l, m+\Delta m, c)\cdot\right. \\ &\left.\cdot\, w(\Delta l, \Delta m, c, f) + b(f)\right),\ f=\overline{1,F},\ l=\overline{1,L},\ m=\overline{1,M}, \end{aligned} \tag{2.18}$$

где F — число фильтров, C — число входных каналов, $K \times K$ — пространственные размерности фильтра, $L \times M \times C$ — размер входного изображения,

w — набор сверточных фильтров, b — вектор коэффициентов сдвига. Мы считаем, что границы изображения I корректно дополнены так, чтобы результат имел тот же размер.

Рассмотрим операцию, выполняемую БМ сверточным слоем с таким же входом и выходом:

$$J = \varphi \left(\sum_{\alpha} \sum_{\beta} p^{\alpha} p^{\beta} \exp(\ln I^{\alpha} \odot v^{\beta}) + \mathbf{b} \right),$$

где $\alpha \in \{-, +\}$, $\beta \in \{-, +\}$, $p^{+} = 1$, $p^{-} = -1$, а $\odot$ — операция БМ свертки:

$$(I \odot v)_{n,m,c} = \max_{c=1}^{C} \max_{\Delta n=0}^{K-1} \max_{\Delta m=0}^{K-1} \left(I(n + \Delta n, m + \Delta m, c) + v(\Delta k, \Delta m, c, f) \right),$$

Также введем следующие обозначения для положительной и отрицательной частей входного сигнала I:

$$\begin{aligned} I^{+}(l, m, c) &= \max(I(l, m, c), 0), \\ I^{-}(l, m, c) &= \max(-I(l, m, c), 0), \end{aligned} \tag{2.19}$$

где $l = \overline{1, L}$, $m = \overline{1, M}$, $c = \overline{1, C}$.

Рассмотрим классический полносвязный слой с входом I_P и выходом O_Q:

$$O(q) = \sigma \left(\sum_{p=1}^{P} I(p) \cdot w(p, q) + b(q) \right), \; q = \overline{1, Q}, \tag{2.20}$$

где P — число входных значений, Q — число нейронов в слое, w — матрица весовых коэффициентов полносвязного слоя, b — вектор коэффициентов сдвига.

Для БМ полносвязного слоя с тем же входом и выходом:

$$O(q) = \sigma \left(\sum_{\alpha,\beta} p^{\alpha} p^{\beta} \exp \max_{p=1}^{P} (\ln I^{\alpha}(p) + v^{\beta}(p, q) + b(q) \right), \;\; q = \overline{1, Q}, \tag{2.21}$$

Следует отметить, что, несмотря на то, что обработка положительных и отрицательных значений входного вектора (x^{+} и x^{-} в выражении (2.6)) удваивает его длину, суммарно в них будет лишь N ненулевых значений. Таким образом, вычисление логарифма потребуется только для этих значений. С учетом этого в таблицах 1 и 2 приведено число арифметических операций различных типов для сверточных и полносвязных слоев соответственно. Однако чтобы оценить вычислительную эффективность БМ слоя необходимо

понимать, сколько времени занимает выполнение тех или иных арифметических операций и как эти операции можно исполнять (строго последовательно, с каким-то уровнем параллельности и т.д.), что определяется архитектурой конкретного вычислителя. Поэтому в следующих разделах будет рассмотрена производительность БМ нейронов на центральных процессорах и программируемых логических интегральных схемах как наиболее актуальных платформах, на которых требуется эффективное исполнение нейронных сетей.

Таблица 1 — Число арифметических операций (op) в классическом (conv) и БМ (BM conv) сверточных слоях. F — число фильтров, C — число входных каналов, $K \times K$ — пространственные размеры фильтра, размер входного изображения $L \times M \times C$.

op	conv	BM conv
$\sigma(\cdot)$	FLM	FLM
exp	0	$4FLM$
log	0	CLM
add	FK^2CLM	$2F(K^2C+2)LM$
max	0	$2F(K^2C-1)LM$
mul	FK^2CLM	0

Таблица 2 — Число арифметических операций (op) в классическом (fc) и БМ (BM fc) полносвязных слоях. P — число входных значений, Q — число нейронов в слое.

op	fc	BM fc
$\sigma(\cdot)$	Q	Q
exp	0	$4Q$
log	0	P
add	QP	$2Q(P+2)$
max	0	$2Q(P-1)$
mul	QP	0

2.4 Оценка эффективности БМ нейронных сетей на ЦП

Современные ЦП используют арифметико-логические устройства (АЛУ) для таких вычислений как сложение, умножение и операция максимума. Также на них есть SIMD-расширения с регистрами размера 128-512 бит для векторных данных. В таблице 3 приведены латентности и пропускные способности для этих операций для нескольких современных процессоров архитектур x86_64 и ARM. Можно видеть, что они не различаются для разных операций над вещественными данными, и лишь незначительно отличаются для целочисленных данных.

Это означает, что основные вычисления в БМ слоях, состоящие из операций взятия максимума и сложений, не будут работать быстрее, чем вычисления в классических слоях, при использовании вещественных типов данных.

Кроме того, помимо операций взятия максимума и сложения БМ слои должны также произвести логарифмирование и потенцирование, а также выполнить вычисление нескольких веток БМ структуры (2 или 4 в зависимости от входных данных). Из-за этого БМ слои проигрывают по вычислительной эффективности классическим слоям на ЦП.

Необходимо отметить, что современные процессоры общего назначения используются для ряда стандартных задач, в которых умножение играет ключевую роль. Поэтому их архитектуры хорошо оптимизированы именно для выполнения умножения, а не других арифметических операций. В то же время, число АЛУ общего назначения на ЦП относительно невелико, и они способны обеспечивать всего несколько результатов вычислений за такт. Так происходит потому, что архитектура ЦП предназначена для решения множества задач общего назначения, обеспечения взаимодействия со другими вычислительными и периферийными устройствами и поэтому заметно проигрывает узкоспециализированными вычислителям, например, графическим ускорителям [108; 109] или тензорным процессорам [110].

Это означает, что несмотря на то, что БМ нейронные сети не имеют преимуществ на ЦП относительно классических сетей, они все еще являются перспективными для специализированных вычислителей, которые могут извлечь пользу из использования более простых операций сложения и взятия максимума.

Таблица 3 — Характеристики арифметических операций для скалярных и векторных (SIMD) типов данных на различных устройствах [111; 112] в формате латентость/средняя пропускная способность для каждой операции.

Устройство	add	max	mul	mul+add
Intel Skylake-X, floating-point 128-bit vector	4/0.5	4/0.5	4/0.5	-
Intel Skylake-X, integer 128-bit vector	1/0.33	1/0.5	5/0.5	5/0.5
Intel Skylake-X, single-precision floating-point	3/1	4/0.5	5/1	-
Intel Coffee Lake, floating-point 128-bit vector	4/0.5	4/0.5	4/0.5	-
Intel Coffee Lake, integer 128-bit vector	1/0.33	1/0.5	5/0.5	5/0.5
Intel Coffee Lake, single-precision floating-point	3/1	4/0.5	5/1	-
ARM Cortex-A57, floating-point 128-bit vector	5/2	5/2	5/2	-
ARM Cortex-A57, integer 128-bit vector	3/2	3/2	5/1	5/1
ARM Cortex-A57, single-precision floating-point	5/2	5/2	5/2	-

2.5 Оценка эффективности БМ нейронных сетей на ПЛИС и СЛИС

Наиболее многообещающая платформа для БМ нейронных сетей это ПЛИС и специализированные устройства, поскольку на них можно реализовать отдельные блоки для операций сложения и максимума, которые будут работать быстрее, чем АЛУ общего назначения. Также на них БМ нейроны/слои можно реализовать в виде 4 параллельных наборов устройств параллельных для разных вычислительных веток, чтобы ускорить исполнение. Оценим число вентилей и латентность, требуемую для операций, которые используются

в БМ и классических нейронах. Для этого далее будет рассмотрена реализация вещественных чисел, элементарных арифметических операций (сложения, умножения, взятия максимума), а также логарифма и экспоненты. Несмотря на то, что биполярный морфологический нейрон (2.6) использует операции натурального логарифма и потенцирования, из соображений вычислительной эффективности вместо них рассматривалась пара «двоичный логарифм» и «возведение в степень двойки» эквивалентным образом.

2.5.1 Вещественная арифметика

Вычисления в классических нейросетевых моделях выполняются в вещественных числах. Для представления вещественных чисел в компьютере наиболее широко употребляются форматы данных, определенные стандартом арифметики с плавающей точкой IEEE 754. Этот стандарт был разработан Институт инженеров электротехники и электроники (IEEE) в 1985 году, а затем обновлен в 2008 и 2019 годах. Последняя версия стандарта включает в себя [113]:

1. Форматы двоичных и десятичных вещественных данных.
2. Определения арифметических операций: сложение, вычитание, умножение, деление, комбинированное умножение со сложением, квадратный корень и другие операции.
3. Правила преобразования между целочисленными и вещественными типами данных. Правила преобразования между различными вещественными типами данных.
4. Правила преобразования вещественных данных к внешним представлениям (например, строкам).
5. Форматы и правила обработки вещественных исключений, включая обработку не-числовых данных.

Использование этого стандарта обеспечивает одинаковые результаты вычислений для программных, аппаратных или комбинированных реализаций вещественной арифметики, а также предоставляет единый формат ошибок, не привязанный к конкретной реализации.

Вещественные числа в двоичных форматах из стандарта IEEE 754 состоят из 3 упорядоченных полей:

1. 1-битный знак числа S,
2. w-битовая сдвинутая экспонента $E = e + bias$,
3. $(t = p-1)$-битовая мантисса $T = d_1 d_2 \dots d_{p-1}$, где $d_i \in \{0, 1\}$ – двоичные разряды мантиссы, причем лидирующий разряд d_0 неявным образом закодирован в экспоненте E.

Значение v вещественного числа получается следующим образом:

1. Если $E = 2w - 1$ и $T \neq 0$, тогда v является не-числовым значением (NaN), и значение бита d_1 определяет дальнейшие действия: продолжение вычислений или сигнализирование об ошибке.
2. Если $E = 2w - 1$ и $T = 0$, тогда $v = (-1)^S \cdot (+\infty)$.
3. Если $1 \leqslant E \leqslant 2w - 2$, тогда число считается нормализованным и $v = (-1)^S \cdot 2^{E-bias} \cdot (1 + 2^{1-p}) \cdot T$.
4. Если $E = 0$ и $T \neq 0$, тогда число считается денормализованным и $v = (-1)^S \cdot 2^{emin} \cdot (0 + 2^{1-p}) \cdot T$, где $emin$ — минимальное значение e. Стоит отметить, что денормализованные числа не превышают по абсолютному значению минимальное нормализованное число.
5. Если $E = 0$ и $T = 0$, тогда $v = (-1)^S \cdot (+0)$.

В задачах машинного обучения чаще всего используется двоичный формат $binary32$. Он использует следующий набор параметров: $w = 8$, $p = 24$, $bias = 127$, $emin = -126$.

Естественно полагать, что входные данные нейронной сети являются нормализованными. При правильно заданных параметрах модели, все промежуточные значения также являются конечными числовыми данными. Денормализованные значения при этом могут возникнуть в результате умножения или вычитания, однако в нейросетевых моделях их можно считать нулями и не учитывать. Таким образом, далее в этом разделе рассматриваются только нормализованные вещественные числа.

2.5.2 Элементарные арифметические операции

Оценим число вентилей и латентность, требуемую для арифметических операций, которые используются в БМ нейроне. Для этого было создано описание основных операций (сложение, максимум, умножение) на уровне ре-

гистровых передач на языке Verilog HDL, и вентильная реализация устройств была синтезирована с помощью Synopsys Design Compiler для 16 нм технологических библиотек. Полученные результаты приведены в таблице 4. Далее была произведена оценка эффективности реализации двоичных логарифма и экспоненты. Для этого использовалось число необходимых элементарных операций (в данном случае — сложений, умножений и максимумов) и вычислялась суммарная вентильная сложность и суммарная латентность, как будет подробно показано в двух следующих разделах.

Таблица 4 — Оценка числа элементарных арифметических операций, логических вентилей и латентности для операций в БМ слоях.

Операция	#add	#max	#mul	Вент.	Лат., такты
add	1	0	0	2659	3
max	0	1	0	563	2
mul	0	0	1	3247	4
log2	6	0	5	32189	38
exp2	3	0	3	17718	21

2.5.3 Полиномиальная аппроксимации логарифма

Для реализации операции логарифмирования в докладе «ResNet-like Architecture with Low Hardware Requirement» [3] автор предложил использовать полиномиальную аппроксимацию, демонстрирующую точность до 4 десятичных знаков после запятой, и имеющую меньшую вычислительную сложность, чем стандартная реализация. Покажем, как выглядит эта аппроксимация.

Рассмотрим вещественное число x:

$$x = 2^e \cdot (1 + 2^{-23} d_1 d_2 \dots d_{23}), \tag{2.22}$$

где e — двоичная экспонента, а $d_1, \dots d_{23}$ — двоичные разряды мантиссы этого числа.

Для него

$$\log_2 x = e + \log_2(1 + 2^{-23} d_1 d_2 \dots d_{23}) = e + \log_2(1 + y), \tag{2.23}$$

Это означает, что необходимо приблизить $\log_2(1+y)$ для $y \in [0,1)$.

Полиномиальная аппроксимация 5-го порядка выглядит следующим образом:

$$f(y) = \log_2(1+y) \to \tilde{f}(y) = \sum_{i=0}^{5} C_i y^i. \tag{2.24}$$

Для определения коэффициентов C_i была составлена система линейных уравнений: в трех точках 0, 0.5, 1 значения $f(y)$ были приравнены значениям $\tilde{f}(y)$, а значения $f'(y)$ — значениям $\tilde{f}'(y)$. В результате ее решения были получены значения $C = \{0, 1.44269504, -0.71249131, 0.42046732, -0.1955884, 0.04491735\}$. Максимальная ошибка аппроксимации составила около $7 \cdot 10^{-5}$ на промежутке $[0,1)$.

Эта аппроксимация использует всего 5 умножений и 6 сложений при вычислении с помощью схемы Горнера (включая вычитание, чтобы получить e). При этом считается, что доступ к битам числа для получения значений S, E и T реализован аппаратно и не требует дополнительного времени.

На основе этого числа операций была сделана оценка вентильной сложности и латентности для модуля логарифма:

$$\begin{aligned} V_{log} &= 6V_{add} + 5V_{mul}, \\ L_{log} &= 6L_{add} + 5L_{mul}, \end{aligned} \tag{2.25}$$

где V_{add} и L_{add} — число вентилей и латентность сумматора соответственно, V_{mul} и L_{mul} — число и латентность умножителя соответственно. Оценка, полученная таким образом, не является точной, так как предполагает, что для каждой операции используется отдельный модуль и все эти модули работают последовательно, однако позволяет получить общее представление о сложности логарифмирования. Численные оценки приведены в таблице 4.

2.5.4 Реализация экспоненты

Современные процессоры x86_64 могут включать в себя специальные инструкции для быстрого вычисления экспоненты, как, например, процессоры Intel [114]. При этом используется таблица предподсчитанных значений и полиномиальная аппроксимация второго порядка, а значит ее сложность составляет

3 операции сложения и 3 операции умножения, не считая операции доступа по индексу таблицы. Относительная ошибка такой аппроксимации меньше 2^{-23}, то есть она дает точные результаты при вычислении в типе $binary32$.

На основе этого числа операций была сделана оценка вентильной сложности и латентности для модуля экспоненты:

$$\begin{aligned} V_{log} &= 3V_{add} + 3V_{mul}, \\ L_{log} &= 3L_{add} + 3L_{mul}, \end{aligned} \tag{2.26}$$

где V_{add} и L_{add} — число вентилей и латентность сумматора соответственно, V_{mul} и L_{mul} — число и латентность умножителя соответственно.

Оценка, полученная таким образом, не является точной, так как предполагает, что для каждой операции используется отдельный модуль и все эти модули работают последовательно, однако позволяет получить общее представление о сложности потенцирования. Численные оценки приведены в таблице 4.

2.5.5 Оценка числа вентилей и латентности для сверточного слоя

Приведем оценку числа вентилей и латентности для классического и БМ сверточных слоев, методика которой была предложена автором в докладе «ResNet-like Architecture with Low Hardware Requirement» [3] и далее расширена на 16 нм устройства в статье «Bipolar Morphological Neural Networks: Gate-Efficient Architecture for Computer Vision» [1].

Число вентилей для классического и БМ слоев обозначим как V_{std} и V_{BM} соответственно. Латентности обозначим как L_{std} и L_{BM} соответственно. Для оценок использовалось число операций в слоях (см. таблицу 1) и характеристики этих операций из таблицы 4. Для расчета числа вентилей рассматривалась двухветочная структура слоя, которая соответствует использованию активации ReLU в предыдущем слое, что характерно для ряда современных глубоких нейросетевых моделей, как уже обсуждалось в главе 1. Для оценки латентности использовалась суммарная латентность всех модулей в одной вычислительной ветке, так как вычисления в ветках могут быть организованы параллельным образом. Отношение числа вентилей и латентностей для классического и БМ сверточных слоев с различными параметрами приведено в таблице 5.

Таблица 5 — Оценка отношения числа вентилей V и латентности L для классического (std) и БМ (BM) сверточных слоев для структуры с слоя 2-ветками с F выходными каналами, C входными каналами и размером ядра свертки $K \times K$.

F	C	K	V_{std}/V_{BM}	L_{std}/L_{BM}
16	1	1	0.11	0.22
16	16	1	0.52	0.78
32	1	1	0.11	0.22
32	32	1	0.68	1.00
64	1	1	0.11	0.23
64	64	1	0.77	1.17
128	1	1	0.11	0.23
128	128	1	0.84	1.27
256	1	1	0.12	0.23
256	256	1	0.88	1.33
512	1	1	0.12	0.23
512	512	1	0.90	1.37
16	1	3	0.51	0.87
16	16	3	0.85	1.29
32	1	3	0.51	0.88
32	32	3	0.88	1.34
64	1	3	0.51	0.89
64	64	3	0.90	1.37
128	1	3	0.52	0.90
128	128	3	0.91	1.38
256	1	3	0.52	0.90
256	256	3	0.91	1.39
512	1	3	0.52	0.90
512	512	3	0.92	1.40

Данные в таблице 5 демонстрируют, что для слоев с достаточно большим числом входных и выходных каналов БМ слои используют практически столько же вентилей, сколько и классические слои, однако при этом латентность БМ слоя меньше на 30-40%.

Необходимо отметить, что на реальном устройстве или ПЛИС не будут использоваться отдельные вентили для вычисления каждой операции, поэтому точная оценка будет несколько отличаться от приведенной.

2.6 Моделирование аппаратной реализации БМ сети на ПЛИС

В работе «Hardware Implementation of Classical and Bipolar Morphological Models for Convolutional Neural Network» [9] при участии автора показаны результаты прямого моделирования аппаратной реализации БМ нейронной сети с точными функциями активации.

Рассмотрим эту реализацию подробнее. Аппаратная реализация нейронных сетей включает в себя:

1. Внешнюю память для хранения весовых коэффициентов и результатов промежуточных вычислений.
2. Вычислительное ядро, поэтапно выполняющее вычисления над входным изображением и фильтрами.
3. Обертку, взаимодействующую с внешней памятью (загрузка и выгрузка пакетов данных, необходимых для вычислений).

Рассматривалась реализация только вычислительного ядра для классических и БМ сетей, а обертка для взаимодействия с внешней памятью полагалась универсальной и не вносящей дополнительных временных/аппаратных затрат.

В качестве модели нейронной сети рассматривалась сеть ResNet, описанная в разделе 1.2.3. Для экспериментов была выбрана модель ResNet-22, достаточно компактная для применения на мобильных устройствах и во встраиваемых системах, но в то же время обеспечивающая достаточно высокое качество классификации для решения практических задач. Архитектура сети показана на рисунке 2.4.

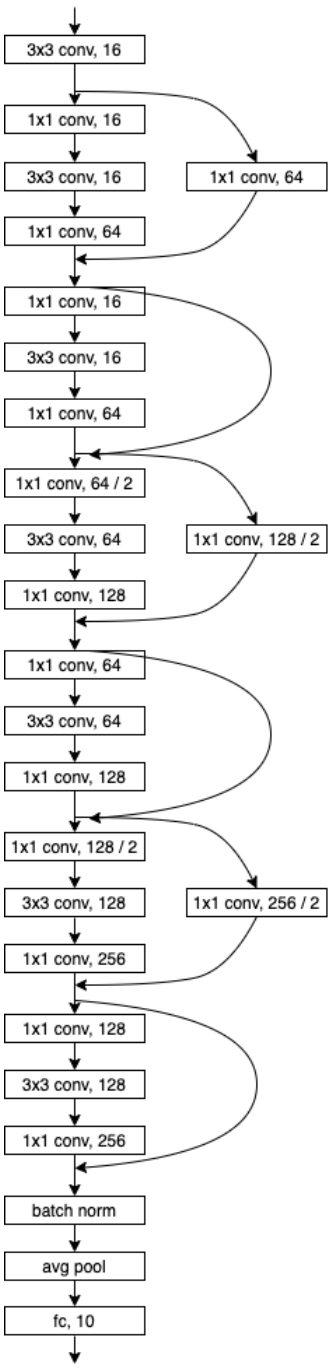


Рисунок 2.4 — Архитектура ResNet-22, $k \times k$ conv, f/s – сверточный слой с f фильтрами размера $k \times k$ и сдвигом s. Если s не указан, предполагается, что он равен 1, batch norm — слой нормализации, avg pool — слой усредняющей субдискретизации, fc, 10 — полносвязный слой с 10 нейронами. Стрелками указано направление потока данных, в случае слияния двух потоков, соответствующие векторы данных складываются. Слои нормализации и активации внутри остаточных блоков опущены для простоты.

2.6.1 Реализация классического сверточного слоя

Для реализации сверточного слоя чаще всего используются специальные устройства, способные умножить два операнда и сложить с третьим операндом, которые называются FMA (Fused Multiply-Add). На основе таких устройств и было реализовано вычислительное ядро, которое принимает на вход:

– канал входного изображения (или часть канала),
– канал фильтра,
– сдвиг и коэффициенты нормализации,
– управляющие сигналы (старт ядра, флаг, обозначающий свертку 3×3, флаг, обозначающий обработку последнего канала изображения).

Вычислительное ядро состоит из FMA модулей и аккумуляторов, в которых суммируются результаты вычислений и из которых далее выгружается результат (см. рисунок 2.5). FMA модуль принимает на вход три операнда типа *binary*32, перемножает первые два между собой, а затем полученное произведение и третий операнд суммируются без промежуточного округления. Работа

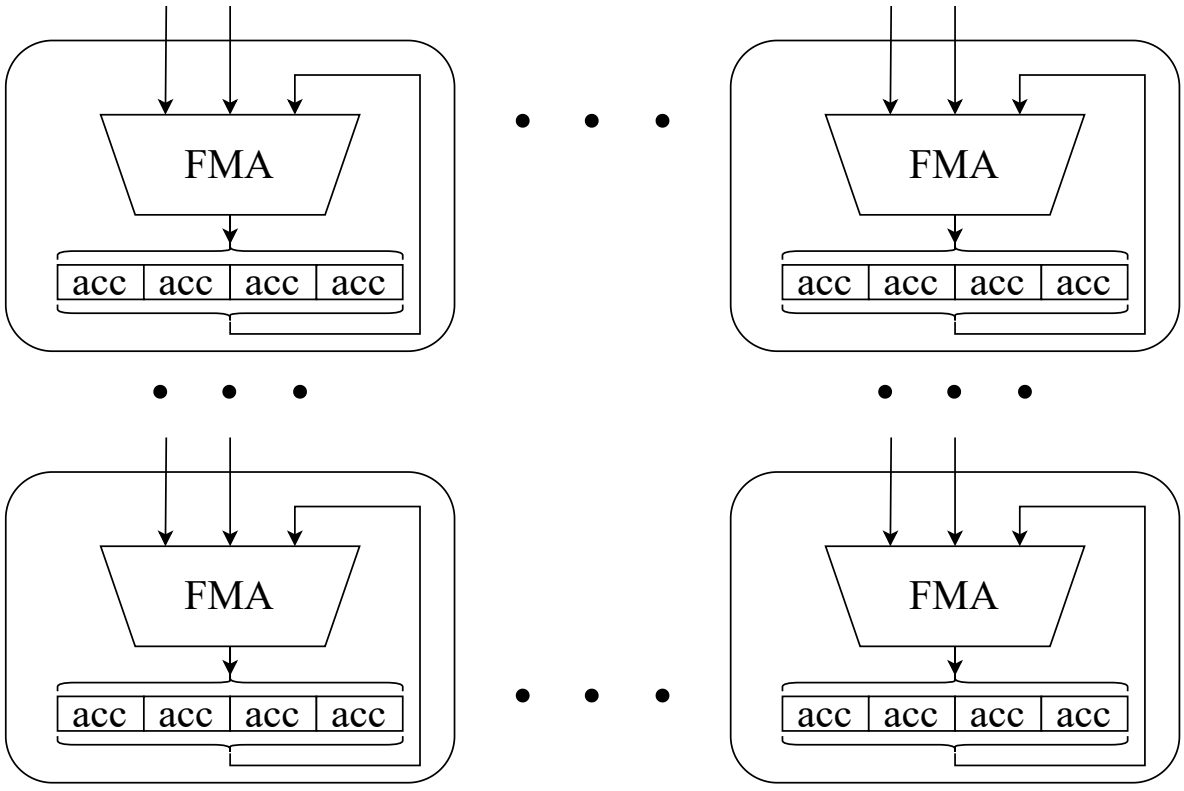


Рисунок 2.5 — Структура вычислительного ядра для классического сверточного слоя. Обозначения: acc — аккумулятор, FMA — модуль FMA.

модуля занимает 4 такта и может быть конвейеризована. Это означает, что при использовании 4-стадийного конвейера, он способен выдавать результат на каждом такте. Поэтому резонно использовать 4 аккумулятора для хранения результатов на каждый FMA модуль. Такой дизайн оптимальнее для аппаратной реализации по сравнению с реализацией с одним аккумулятором на каждый FMA модуль, однако работает несколько дольше, чем четыре 4 FMA, используемые параллельно из-за необходимости заполнить конвейер для достижения максимальной производительности.

Сигнал старта ядра означает начало вычислений на всех FMA устройствах. В течение первых 4 тактов после старта, используются одинаковые элементы фильтра. При этом элементы изображения и значения аккумуляторов меняются каждый такт. Таким образом, разные элементы изображения умножаются на одни и те же элементы фильтра, а затем произведения суммируются с соответствующими аккумуляторами. Например, если ядро вычисляет свертку с фильтром 3×3, то после первых 4 тактов, загружаются следующие элементы фильтра, и операция повторяется. Когда все элементы фильтра обработаны, но канал входного изображения не последний, взводится флаг готовности ядра к новой итерации вычислений и начинается обработка следующего канала. В противном случае начинается этап активации. Во время активации каждый FMA модуль берет коэффициенты нормализации в качестве второго и третьего операндов, а значения аккумуляторов — в качестве первого. То есть, через 4 такта все значения аккумуляторов загружены в конвейер FMA модуля, а еще через 4 такта — обработаны и записаны в аккумуляторы снова. Далее знаковый бит каждого из аккумуляторов используется в качестве маски для реализации ReLU. После этого этап вычисления активации заканчивается и снова взводится флаг готовности к следующей итерации, а обертка внешней памяти выгружает значения аккумуляторов во внешнюю память.

2.6.2 Реализация БМ сверточного слоя

Реализация БМ сверточного слоя должна включать в себя модули для подсчета двоичного логарифма, экспоненты и основного блока, вычисляющего максимум сумм.

Работа модуля основного блока занимает 3 такта и также была конвейеризована. Такой модуль требует в 3 раза меньше аппаратных ресурсов по сравнению с модулем FMA.

Для возведения в степень двойки использовалась реализация, описанная в разделе 2.5.4. Модуль экспоненты был конвейеризован и его работа заняла 5 тактов. Стоит отметить, что модуль работает быстрее, чем предполагалось, так как при создании реального модуля нет нужды выполнять операции (в данном случае — сложения и умножения) последовательно.

Для вычисления двоичного логарифма использовалась полиномиальная аппроксимация, описанная в разделе 2.5.3. Модуль конвейеризован и работает за 16 тактов.

Структура вычислительного ядра БМ слоя аналогична таковой для классического слоя. После каждой операции суммирования выполняется сравнение результата с аккумуляторов, и, в случае превосходства, в аккумулятор сохраняется новый результат.

Однако в таком ядре необходимо иметь 4 набора аккумуляторов, по одному для каждой вычислительной ветки в ($++$, $+-$, $-+$, $--$ в выражении (2.6)). Половина основных модулей в ядре работает с фильтром v^+, а половина — с фильтром v^-. Конвейеры в них работают также как и в классическом ядре: обработка запускается 4 раза с различными элементами входного изображения и одними и теми же элементами фильтра. Стоит отметить, что поскольку лишь одна из величин $\log_2 I^+$ и $\log_2 I^-$ не равна $-\infty$, достаточно вычислить только 2 БМ свертки. Поэтому аккумуляторов используется лишь 8. До обработки последнего канала входного канала обработка происходит аналогично классическому слою. Поскольку модуль вычисления логарифма отличается долгой работой, он помещен в конец вычисления активации. Логарифм входа для первого сверточного слоя вычисляется отдельно.

После обработки последнего канала выполняется вычисление экспонент и их суммирование, прибавление сдвига, активация полученных результатов и логарифмирование. Модули экспоненты и логарифма требуют заметно больше аппаратных ресурсов, чем основные БМ модули, поэтому было использовано 4 модуля экспоненты и 1 модуль логарифма (см. рисунок 2.6).

Когда все значения в аккумуляторах готовы для операции потенцирования, эти значения поступают в модули экспоненты, после чего суммируются и

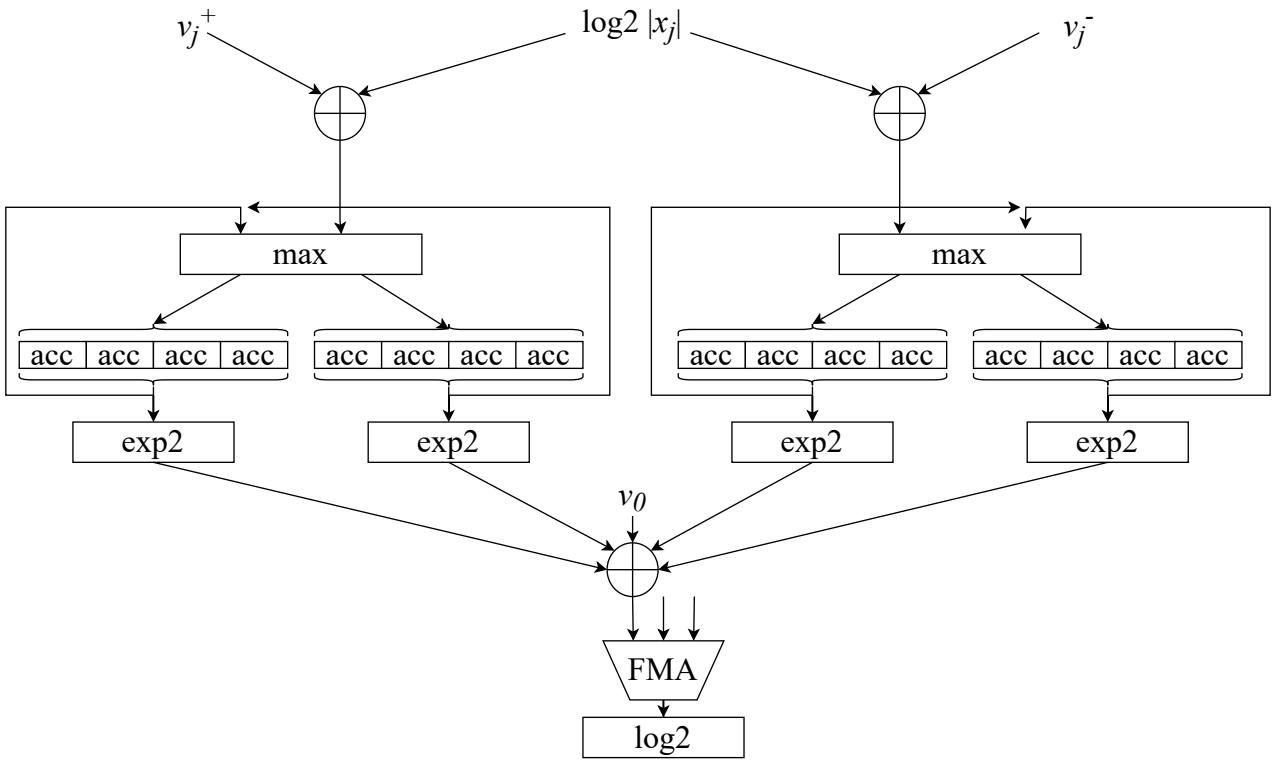


Рисунок 2.6 — Структура одного вычислительного модуля для БМ сверточного слоя. Обозначения: x — вектор входных значения БМ нейрона, v — вектор весовых коэффициентов БМ нейрона, $\oplus$ — модуль для вычисления суммы входов, max — модуль для вычисления максимума входов, exp2 — модуль для вычисления двоичной экспоненты входа, log2 — модуль для вычисления двоичного логарифма входа, acc — аккумулятор, FMA — модуль FMA.

к ним прибавляется сдвиг. Сумматоры, используетмые на этом этапе — те же, что выполняют основные вычисления в слое.

Далее выполняется нормализация, для этого используется один FMA модуль. На выходе знаковый бит результата используется в качестве маски для реализации ReLU, и, наконец, выполняется логарифмирование, чтобы получить вход для следующего сверточного слоя. Стоит отметить, что БМ ядро отличается от классического, в котором результаты оказываются в массиве аккумуляторов и вычисляются параллельно. БМ ядро выдает результаты последовательно, что требует корректной обработки системой взаимодействия с внешней памятью.

Кроме того, использование ReLU делает входные сигналы для большей части слоев неотрицательными. То есть, в выражении (2.6) ненулевыми являются всего два слагаемых из четырех. Поэтому резонной выглядит идея убрать два незадействованных массива аккумуляторов. Слои, имеющие и положитель-

ные, и отрицательные входы, можно обрабатывать последовательно, запуская вычисления дважды, а затем суммируя результаты:

$$\begin{aligned} r_1 &= \exp \max_j(\ln x_j^+ + v_j^+) - \exp \max_j(\ln x_j^+ + v_j^-), \\ r_2 &= \exp \max_j(\ln x_j^- + v_j^+) - \exp \max_j(\ln x_j^- + v_j^-), \\ r &= r_1 - r_2 + v_0. \end{aligned} \tag{2.27}$$

Время работы сверточного слоя, можно вычислить по формуле:

$$T_{layer} = F\frac{M \times N}{m \times n} \cdot \left[C \cdot (4K_xK_y + t_1) + t_2\right], \tag{2.28}$$

где F — число фильтров в слое, C — число входных каналов слоя, $M \times N$ — пространственные размеры входного изображения, $m \times n$ — размеры пространственной части изображения, которая обрабатывается ядром, $K_x \times K_y$ — пространственные размеры фильтра, t_1 — время ожидания обработки входного канала (3 такта для FMA, 2 такта для сумматора), t_2 — время обработки выходного канала (нормализация в классической сети и потенцирование и суммирование в БМ сети)

В таблице 6 показано полное время вычисления сверточных слоев сети ResNet-22 для $m \times n = 8 \times 8$ с частотой устройства 1 ГГц, $M \times N \times C = 32 \times 32 \times 3$ при условии отсутствия задержек, вызванных внешней памятью. Можно видеть, что необходимость второго прохода для некоторых слоев в версии с 2 массивами аккумуляторов демонстрирует на 10% большую латентность, но модель с 4 массивами аккумуляторов незначительно отличается по скорости от классической. Однако версия с 4 массивами аккумуляторов занимает в 1.68 раз большую площадь и требует заметно больше вентилей, в то время как версия с 2 массивами аккумуляторов занимает в 1.21 раз большую площадь и в 1.12 раза больше вентилей.

Таблица 6 — Характеристики вычислительных ядер по результатам синтеза.

	Классическая модель	БМ модель с 4 аккумуляторами	БМ модель с 2 аккумуляторами
T_{layer}, мкс	5488	5616	6095
Площадь, мкм^2	40780	68395	49267
Число вентилей	154615	229529	173743

Таким образом, результаты прямого моделирования для модели ResNet-22 показали, что использование БМ сверточных слоев для полной сети не позволяет получить значительного преимущества относительно классической сети с 2-веточной и даже с 4-веточной структурой.

Этот результат в первую очередь связан с тем, что конвейеризация арифметико-логических устройств заметно снизила относительное преимущество модуля аккумулирующего максимума над модулем аккумулирующего умножения. Это преимущество продолжает наблюдаться для вычислительно-интенсивных слоев сети с большим числом входных и выходных каналов. Однако таких слоев в модели ResNet-22 не так много, поэтому для практического применения выглядит разумным использовать гибридные модели, в которых лишь часть самых ресурсоемких слоев приводится к БМ виду.

Другой причиной ухудшения характеристик модели БМ слоя относительно классического является медленная работа и большой транзисторный бюджет точных функций активаций, а также недостаток специализированных модулей для их реализации на ПЛИС. Эту проблему можно решить путем использования аппроксимированных функций двоичного логарифма и потенцирования, что и будет сделано далее.

2.7 Быстрые аппроксимации функций активации БМ нейрона

Оценка и моделирование БМ слоя показали, что есть значения параметров, при которых он уступает классическому слою. Одно из таких мест — необходимость сложных устройств логарифмирования и потенцирования. Поэтому в работе «Fast and gate-efficient approximated activations for bipolar morphological neural networks» [10] автор предложил использовать куда более простые для реализации аппроксимации Митчелла и Шраудольфа для этих операций. Рассмотрим их подробнее.

2.7.1 Аппроксимация Митчелла

Одной из самых вычислительно-эффективных аппроксимаций двоичного логарифма является аппроксимация, предложенная Дж. Митчеллом [115]. Он предложил использовать первый член ряда Маклорена чтобы аппроксимировать $\log_2(1 + y)$:

$$\log_2(1 + y) \approx y.$$

Тогда

$$\log_2(x) = e + y, \tag{2.29}$$

где e и y можно получить с помощью битовых манипуляций над вещественным числом. Таким образом, вычислительная сложность этой аппроксимации составляет всего одну операцию сложения. Максимальное по абсолютной величине отклонение от точной функции двоичного логарифма можно определить аналитическим путем, оно составляет 0.08639 на интервале (0, 1). Однако, поскольку вычисления в нейронных сетях чаще всего можно приближать в широких пределах, эта аппроксимация заслуживает внимания. Аппроксимация Митчелла проиллюстрирована на Рис. 2.7. Можно видеть, что она представляет собой

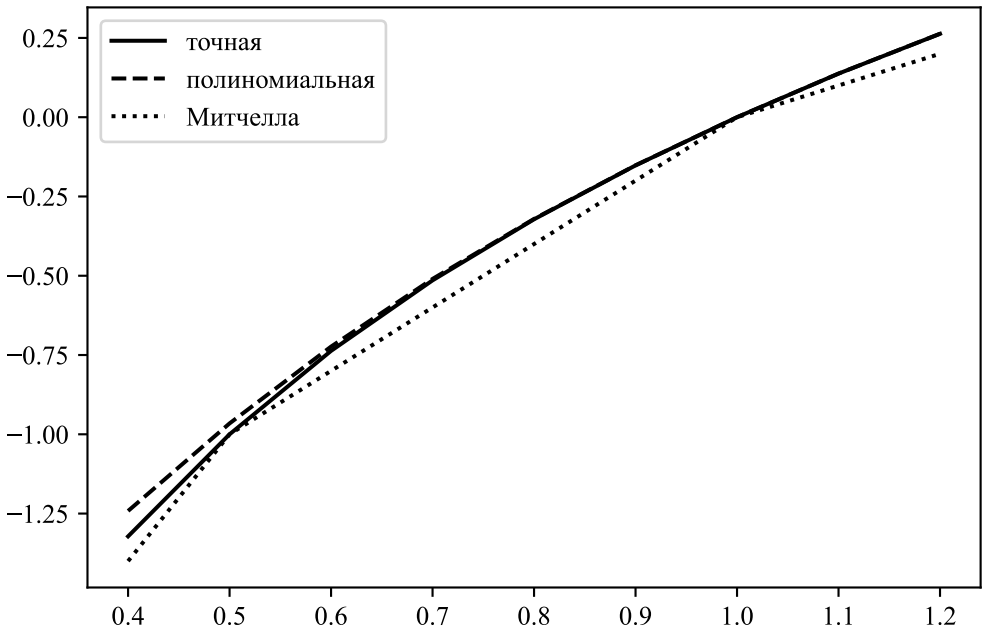


Рисунок 2.7 — Сравнение различных реализаций двоичного логарифма.

кусочно-линейную функцию, причем концы каждого отрезка расположены в точках, равных степеням двойки.

Таким образом, аппроксимация двоичного логарифма по Митчеллу требует в 6 раз меньше сложений и не требует умножений по сравнению с полиномиальной аппроксимацией.

2.7.2 Аппроксимация Шраудольфа

В 1999 году Н. Шраудольф предложил крайне эффективную для вычисления аппроксимацию экспоненциальной функции [116], основанную на структуре двоичных форматов IEEE 754 для представления вещественных чисел. В его работе рассмотрены данные типа $binary64$ или $double$, однако предложенный подход можно легко распространить на $binary32$ данные.

Так как нормализованное вещественное число записывается следующим образом:

$$y = 2^e \cdot (1 + 2^{-23} d_1 d_2 \dots d_{23}), \tag{2.30}$$

где e — двоичная экспонента, а $d_1, \dots d_{23}$ — двоичные разряды мантиссы этого числа, для вычисления 2^x где x — целое, необходимо просто записать $x + bias$ в поле экспоненты, то есть:

$$i = 2^{p-1} \left(x + bias\right),$$

где i — непосредственное представление вещественного числа. Напомним, что $bias$ — это константа, определяемая типом данных и равная 127 для $binary32$. При использовании не целого x, эта операция также изменит старшие биты мантиссы. Согласно [116], при это будет осуществляться линейная интерполяция между соседними экспонентами с целочисленными степенями. Таким образом, аппроксимация Шраудольфа имеет вид:

$$i = ax + \left(b - c\right), \tag{2.31}$$

где i — непосредственное представление вещественного числа, $a = 2^{23}$, $b = 127 \cdot 2^{23}$, а c — поправочный коэффициент, взятый равным 486411.

Эта аппроксимация использует одно целочисленное сложение и одно вещественное умножение. Максимальное абсолютное отклонение от точной

степенной функциии на полуинтервале $[0, 1)$ было определено численно и составило 0.05798 (см. Рис. 2.8).

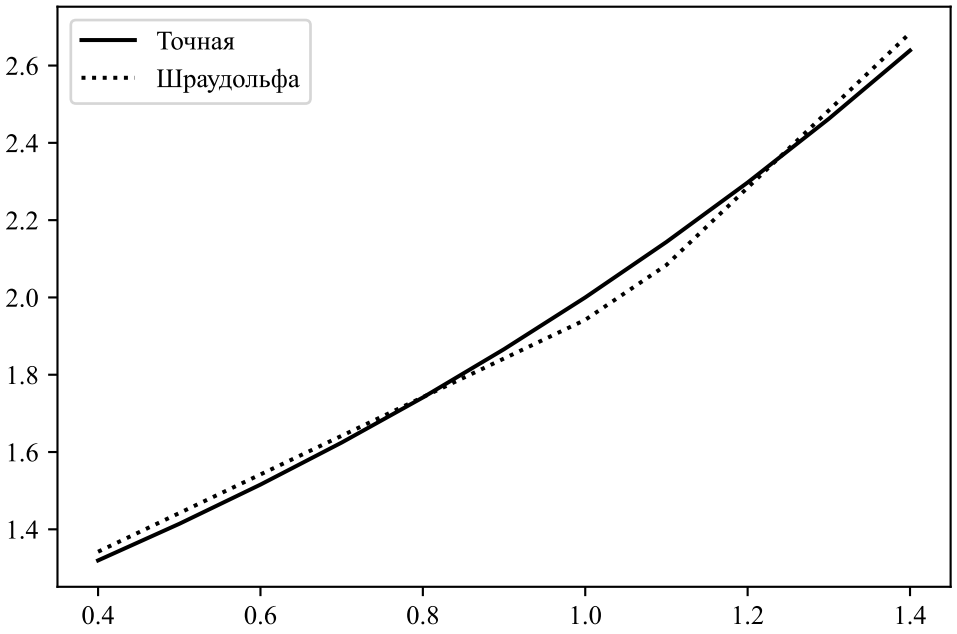


Рисунок 2.8 — Сравнение различных реализаций операции двоичного потенцирования.

2.7.3 Оценка вентильной сложности и латентности

Оценки вентильной сложности аппроксимированных модулей логарифма Митчелла и экспоненты Шраудольфа показаны в таблице 7. Оценки сложности остальных операций также приведены для иллюстрации.

Можно видеть, что аппроксимация Митчелла требует приблизительно в 12 раз меньше логических вентилей и имеет в 12.7 раз меньшую латентность, чем полиномиальная аппроксимация логарифма. Аппроксимация двоичной экспоненты по Шраудольфу использует в 3 раза меньше операций сложения и умножения, чем точная реализация и демонстрирует в 3 раза меньшее число вентилей и латентность.

Приведем оценки числа вентилей и латентности для классического и БМ сверточных слоев с аппроксимированными функциями активации. Число венти-

Таблица 7 — Оценка числа элементарных арифметических операций, логических вентилей и латентности для операций в БМ слоях.

Операция	#add	#max	#mul	Вент.	Лат., такты
add	1	0	0	2659	3
max	0	1	0	563	2
mul	0	0	1	3247	4
log2	6	0	5	32189	38
log2 (Митчелл)	1	0	0	2659	3
exp2	3	0	3	17718	21
exp2 (Шраудольф)	1	0	1	5906	7

лей для данных слоев обозначается как V_{std} и $\tilde{V}_{BM}$ соответственно. Латентности обозначим как L_{std} и $\tilde{L}_{BM}$ соответственно. Оценки выполнялись также как и в разделе 2.5.5, то есть для расчета числа вентилей рассматривалась двухветочная структура слоя, которая соответствует использованию активации ReLU в предыдущем слое, что характерно для ряда современных глубоких нейросетевых моделей. Для оценки латентности использовалась суммарная латентность всех модулей в одной вычислительной ветке, так как вычисления в ветках могут быть организованы параллельным образом. Отношение числа вентилей и латентностей для классического и БМ сверточных слоев с/без аппроксимацией активаций приведено в таблице 8.

Данные в таблице 8 демонстрируют, что для слоев с достаточно большим числом входных и выходных каналов оценки для БМ слои с аппроксимированными функциями активации практически не изменились, то есть функции активации вносят малый вклад как в число вентилей, так и в общую латентность. Однако для малого числа входных каналов и фильтров 3×3 использование аппроксимированных активаций делает латентность БМ слоев на 12-40% меньше, чем классических. Тем не менее, основной вклад во время вычислений вносят именно слои с большим числом входных и выходных каналов. В нейросетевой модели с остаточной архитектурой ResNet-22, конфигурации слоев которой в том числе рассмотрены в таблице 8, есть лишь несколько слоев, которые получают значительное преимущество от аппроксимированных функций активации. Это означает, что повторение эксперимента по прямому

Таблица 8 — Оценка отношения числа вентилей и латентности для классического и БМ сверточных слоев для структуры с слоя 2-ветками.

F	C	K	V_{std}/V_{BM}	L_{std}/L_{BM}	$V_{std}/\tilde{V}_{BM}$	$L_{std}/\tilde{L}_{BM}$
16	1	1	0.11	0.22	0.21	0.43
16	16	1	0.52	0.78	0.74	1.19
32	1	1	0.11	0.22	0.21	0.43
32	32	1	0.68	1.00	0.82	1.29
64	1	1	0.11	0.23	0.21	0.44
64	64	1	0.77	1.17	0.87	1.34
128	1	1	0.11	0.23	0.21	0.44
128	128	1	0.84	1.27	0.89	1.37
256	1	1	0.12	0.23	0.21	0.44
256	256	1	0.88	1.33	0.90	1.38
512	1	1	0.12	0.23	0.21	0.44
512	512	1	0.90	1.37	0.91	1.39
16	1	3	0.51	0.87	0.67	1.12
16	16	3	0.85	1.29	0.89	1.37
32	1	3	0.51	0.88	0.67	1.12
32	32	3	0.88	1.34	0.90	1.39
64	1	3	0.51	0.89	0.67	1.12
64	64	3	0.90	1.37	0.90	1.39
128	1	3	0.52	0.90	0.67	1.12
128	128	3	0.91	1.38	0.91	1.40
256	1	3	0.52	0.90	0.67	1.12
256	256	3	0.91	1.39	0.92	1.40
512	1	3	0.52	0.90	0.67	1.12
512	512	3	0.92	1.40	0.92	1.40

моделированию такой нейронной сети не позволит получить качественно новых результатов.

2.8 Финальная БМ модель

В данном разделе подведем итог проведенному исследованию вычислительной эффективности БМ нейрона и приведем финальную модель, использующую аппроксимацию Митчелла для реализации логарифмирования и аппроксимацию Шраудольфа для реализации потенцирования.

Логарифм, аппроксимированный методом Митчелла, можно задать следующим выражением:

$$\widehat{\log 2}(x) = \lfloor \log_2 x \rfloor + \frac{x - 2^{\lfloor \log_2 x \rfloor}}{2^{\lceil \log_2 x \rceil} - 2^{\lfloor \log_2 x \rfloor}}. \tag{2.32}$$

Экспоненту, аппроксимированную методом Шраудольфа, можно задать следующим выражением:

$$\begin{gathered} \widehat{\exp 2}(x) = float(ax + (b - c)), \\ a = 2^{23},\ b = 127 \cdot 2^{23},\ c = 486411, \end{gathered} \tag{2.33}$$

где функция $float$ — интерпретирует число как $binary32$.

Тогда один БМ нейрон задается выражением:

$$\begin{aligned} f_{BM}(\mathbf{x}, V, v) = \varphi \Bigg(& \widehat{\exp 2}(\max_{j=1}^{N}(\widehat{\log 2}(x_j^+) + v_j^+)) - \widehat{\exp 2}(\max_{j=1}^{N}(\widehat{\log 2}(x_j^+) + v_j^-)) - \\ & - \widehat{\exp 2}(\max_{j=1}^{N}(\widehat{\log 2}(x_j^-) + v_j^+)) + \widehat{\exp 2}(\max_{j=1}^{N}(\widehat{\log 2}(x_j^-) + v_j^-)) + v_0 \Bigg), \end{aligned}$$

$$\begin{aligned} x_j^+ &= \begin{cases} x_j, & x_j \geqslant 0, \\ 0, & x_j < 0, \end{cases} \\ x_j^- &= \begin{cases} -x_j, & x_j < 0, \\ 0, & x_j \geqslant 0, \end{cases} \end{aligned} \tag{2.34}$$

где $\mathbf{x}$ — вектор входных значений длины N, v^+, v^- — векторы весовых коэффициентов длины N, v_0 — смещение, $\varphi(\cdot)$ — нелинейная функция активации.

Сверточный слой из БМ нейронов с входом $I_{L\times M\times C}$ и выходом $O_{L\times M\times F}$:

$$J = \varphi\left(\sum_{\alpha}\sum_{\beta} p^{\alpha}p^{\beta}\widehat{\exp 2}(\widehat{\log 2}I^{\alpha}\odot v^{\beta}) + \mathbf{b}\right),$$

где $\alpha \in \{-,+\}$, $\beta \in \{-,+\}$, $p^{+}=1$, $p^{-}=-1$, а $\odot$ — операция БМ свертки:

$$(I\odot v)_{n,m,c} = \max_{c=1}^{C}\max_{\Delta n=0}^{K-1}\max_{\Delta m=0}^{K-1}\left(I_{n+\Delta n,m+\Delta m,c} + v_{\Delta k,\Delta m,c,f}\right),$$

$$\begin{aligned} I^{+} &= \max(I,0),\\ I^{-} &= \max(-I,0). \end{aligned} \tag{2.35}$$

Полносвязный слой из БМ нейронов с входом I_P и выходом O_Q:

$$O(q) = \sigma\left(\sum_{\alpha,\beta} p^{\alpha}p^{\beta}\widehat{\exp 2}\max_{p=1}^{P}(\widehat{\log 2}I^{\alpha}(p) + v^{\beta}(p,q) + b(q))\right),\ \ q=\overline{1,Q}, \tag{2.36}$$

где $\alpha \in \{-,+\}$, $\beta \in \{-,+\}$, $p^{+}=1$, $p^{-}=-1$.

Заданные таким образом слои можно использовать как составную часть широкого класса нейросетевых моделей, заменив на них классические слои. Полученные модели будут аппроксимировать исходные, но демонстрировать меньшую латентность при реализации на ПЛИС/СЛИС. В следующей главе будет рассмотрено обучение БМ нейронных сетей и приведены экспериментальные результаты в различных практических задачах.

2.9 Выводы по главе 2

В данной главе представлены следующие результаты:

1. Предложена новая аппроксимация классического математического нейрона, имеющая меньшую вычислительную трудоемкость в составе слоев нейронной сети, впервые изложенная автором в докладе «Bipolar morphological neural networks: convolution without multiplication» [4] и подробно описанная в статье «Bipolar Morphological Neural Networks: Gate-Efficient Architecture for Computer Vision» [1].

2. Доказано, что БМ нейронная сеть может приблизить любую непрерывную на компакте функцию с заданной точностью; результат впервые опубликован в статье автора «Bipolar Morphological Neural Networks: Gate-Efficient Architecture for Computer Vision» [1].
3. Приведены оценки вычислительной сложности БМ нейрона в терминах числа арифметических операций для сверточных и полносвязных слоев нейронной сети, впервые опубликованные в докладе автора «ResNet-like Architecture with Low Hardware Requirements» [3].
4. Рассмотрены аппроксимации сложных функций активации БМ нейрона, позволяющие дополнительно снизить вычислительную сложность БМ слоя; приведено количество вентилей и латентности для арифметических операций, использующихся в классическом математическом и аппроксимированном БМ нейронах; результат впервые опубликован в статье автора «Fast and gate-efficient approximated activations for bipolar morphological neural networks» [10].
5. Показано, что для ЦП использование БМ нейронов с вещественными коэффициентами не позволяет повысить эффективность вычислений, однако они могут быть эффективно реализованы для ПЛИС/СЛИС: оценка числа вентилей и латентности ПЛИС-реализации для БМ сверточных слоев по сравнению с классическими сверточными слоями показала, что для слоев с достаточно большим числом входных и выходных каналов БМ слои используют практически столько же вентилей, сколько и классические слои, однако имеют латентность на 30-40% ниже; для слоев с достаточно малым числом входных каналов и размером фильтров 3×3 при использовании аппроксимированных функций активации латентность на 12-40% меньше, чем для классических слоев; результаты опубликованы в докладе автора «ResNet-like Architecture with Low Hardware Requirements» [3] и статьях «Bipolar Morphological Neural Networks: Gate-Efficient Architecture for Computer Vision» [1] и «Fast and gate-efficient approximated activations for bipolar morphological neural networks» [10].

Таким образом, предложенная автором биполярная морфологическая аппроксимация классического математического нейрона:

– переносит основные вычисления на операции взятия максимума и сложения, имеющие более простую аппаратную реализацию и латентность,

чем операции сложения и умножения, используемые в классическом нейроне;

– способна приближать непрерывные функции с заданной точностью также как и классические нейроны;
– может обеспечить меньшую латентность вычислений на программируемых логических интегральных схемах и специализированных вычислителях, что подтверждается выполненными оценками и результатами моделирования, тем самым позволяя повысить производительность нейросетевых моделей.

Глава 3. Обучение биполярных морфологических моделей

В предыдущей главе было введено понятие биполярного морфологического нейрона и рассмотрена выразительная способность и вычислительная сложность нейросетевых моделей, состоящих из таких нейронов, на различных вычислительных платформах. Однако несмотря на то, что по теоретическим оценкам выразительная способность БМ сетей не хуже, чем у классических моделей, необходимо проверить, насколько эффективными существующие архитектуры и подходы к обучению нейронных сетей окажутся для БМ моделей и доработать их. Для обучения классических нейросетевых моделей используется алгоритм обратного распространения ошибки и градиентные методы. Такой подход может применяться и для БМ моделей. Однако из-за использования операции максимума для каждого нейрона имеется лишь один ненулевой элемент градиента, то есть лишь один его вес будет обновляться на каждой итерации обучающего процесса. Поэтому большое количество весов может никогда не обновиться после инициализации случайными значениями. Такие веса не будут нести полезную нагрузку и могут затруднять обучение сети. Кроме того, поскольку БМ нейроны преобразуют пространство входных сигналов прежде чем выполнить вычисления, стандартные случайные распределения, используемые для классических моделей, могут быть неэффективны для них. Поэтому в данной главе экспериментально исследуются и дорабатываются методы обучения БМ сетей, а затем рассматривается возможность их применения в практических задачах распознавания.

3.1 Классификация рукописных цифр MNIST с помощью БМ моделей

Рассмотрим базовую задачу классификации изображений рукописных символов на выборке MNIST [117] с помощью моделей, преобразованных к биполярному морфологическому виду. Эта задача считается стандартной и широко используется для начальной оценки свойств методов распознавания. Примеры изображений из выборки показаны на рисунке 3.1.

Рисунок 3.1 — Примеры изображений из выборки MNIST.

Приведем основные параметры выборки MNIST.

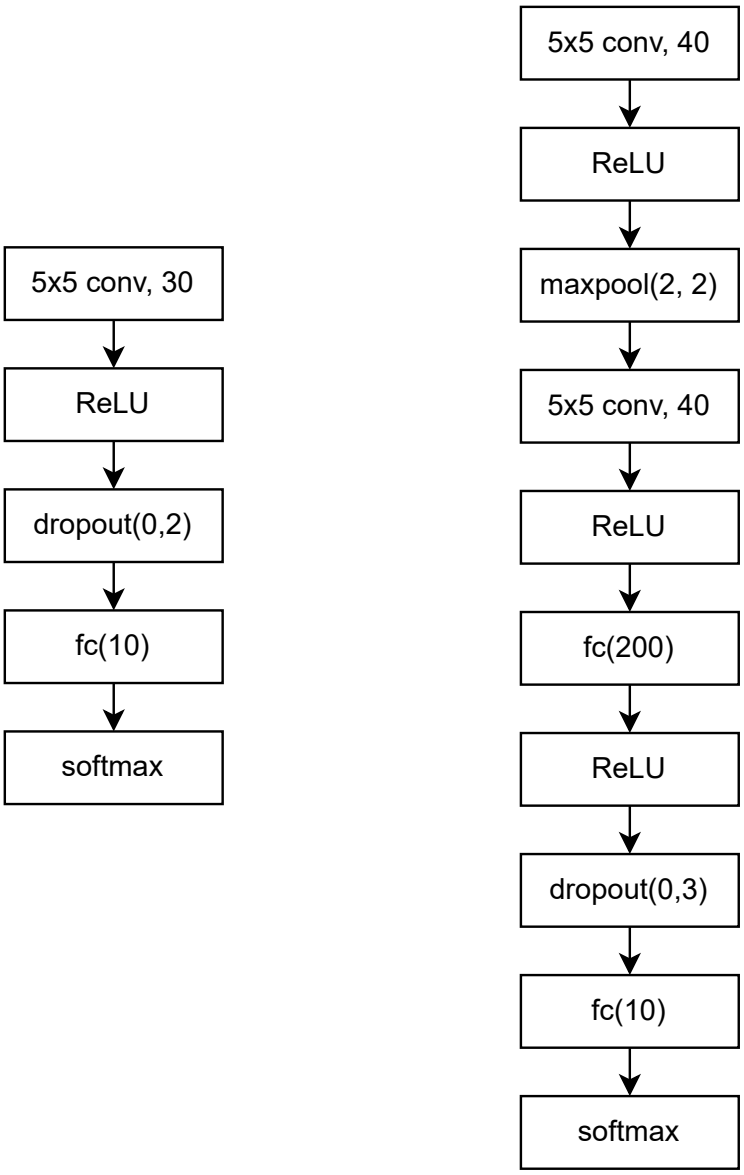
– Число классов: 10.
– Размер изображения: 28×28 пикселей, серое.
– Полный размер выборки: 60000.
– Размер валидационной выборки: 6000.
– Размер тестовой выборки: 10000.

Рассмотрим две LeNet-подобные нейросетевые модели разной сложности с типичной для классификаторов архитектурой: CNN_1 с одним сверточным и одним полносвязным слоями и CNN_2 с двумя сверточными и двумя полносвязными слоями. Обозначения, которые использовались для описания нейросетевых архитектур, приведены в таблице 9, а сами модели продемонстрированы на рисунке 3.2.

Таблица 9 — Условные обозначения для слоев нейронных сетей

Обозначение	Описание
$\mathrm{conv}(n, w_x, w_y)$	сверточный слой с n фильтрами размера $w_x \times w_y$
$\mathrm{fc}(n)$	полносвязный слой с n нейронами
$\mathrm{maxpool}(w_x, w_y)$	слой субдискретизации с операцией максимума и окном размера $w_x \times w_y$
$\mathrm{dropout}(p)$	слой, выполняющий dropout с вероятностью p
relu	функция активации $\mathrm{ReLU}(x) = \max(x, 0)$
softmax	функция активации $\mathrm{softmax}(x_i) = e^{x_i} / \sum_k e^{x_k}$

Для задания БМ нейрона использовалось выражение (2.6) с точными функциями активации. В этом эксперименте к БМ виду приводились только



а) б)

Рисунок 3.2 — Архитектуры нейросетевых моделей для распознавания рукописных цифр, а) CNN_1, б) CNN_2. Стрелками указано направление потока данных.

сверточные слои. Далее было исследовано два подхода к обучению таких моделей:

– с помощью стандартных методов со случайной инициализацией (с равномерным распределением Ксавье [118]);
– с помощью аппроксимации классической модели; согласно выражению (2.4) БМ нейрон с весовыми коэффициентами $\{v^+, v^-, v_0\}_{j=1}^N$ аппроксимирует классический нейрон с весовыми коэффициентами $w_{j=1}^N$

при:

$$v_j^+ = \begin{cases} \ln w_j, \text{ если } w_j > 0, \\ -\infty, \text{ иначе,} \end{cases}$$
$$v_j^- = \begin{cases} \ln |w_j|, \text{ если } w_j < 0, \\ -\infty, \text{ иначе,} \end{cases} \tag{3.1}$$
$$v_0 = w_0.$$

Использованные нейросетевые архитектуры и результаты обучения приведены в таблице 10. Для каждой сети приведены точность классификации классической модели (p_{st}), точность классификации БМ модели, обученной со случайной инициализацией (p_r), а также точность модели с первым БМ слоем, весовые коэффициенты которого получены путем аппроксимации согласно (3.1), а остальные слои — классические (p_a^1). Можно видеть, что для CNN_1 отличие между $p_{st} = 98.7$ и $p_r = 98.5$ невелико, однако $p_a^1 = 42.5$ значительно ниже, чем оба этих значения. Для CNN_2 $p_{st} = 99.5$ заметно выше, чем $p_r = 98.7$, а $p_a^1 = 94.9$ им уступает, также как и для CNN_1. То есть, использование аппроксимированных весов без последующего обучения демонстрирует крайне низкую точность распознавания. Однако и сети обученные со случайной инициализацией заметно проигрывают по точности классическим сетям. Для преодоления этой проблемы автором был предложен метод послойного дообучения, опубликованный в «Bipolar morphological neural networks: convolution without multiplication» [4] и «Fast Integer Approximations In Convolutional Neural Networks Using Layer-By-Layer Training» [8]. Этому методу посвящен следующий раздел.

Таблица 10 — Точность классификации на MNIST: p_{st} — классической сети, p_r — БМ сети, обученной со случайной инициализацией, p_a^1 — сети с первым аппроксимированным БМ слоем и остальными классическими.

Модель	Архитектуры	p_{st}, %	p_r, %	p_a^1, %
CNN_1	conv1(30, 5, 5) - relu1 - dropout1(0,2) - fc1(10) - softmax1	98.7	98.5	42.5
CNN_2	conv1(40, 5, 5) - relu1 - maxpool1(2, 2) - conv2(40, 5, 5) - relu2 - fc1(200) - relu3 - dropout1(0,3) - fc2(10) - softmax1	99.5	98.7	94.9

3.2 Метод послойного преобразования и дообучения

В предыдущем разделе показано, что обучение классическими градиентными методами при помощи алгоритма обратного распространения ошибки оказалось неэффективным для БМ сетей. Это могло произойти по ряду причин:

– градиент БМ слоя является ненулевым только для весовых коэффициентов, которые в данный момент обеспечили максимальные значения у нейронов слоя, то есть всего у одного веса для каждого из нейронов; другие весовые коэффициенты сохранят исходные значения, то есть обновление коэффициентов происходит крайне медленно;
– стандартные распределения, используемые для инициализации начальных значений весов, неэффективны для БМ нейронов.

Для решения этих проблем автор предложил метод, использующий аппроксимационную природу БМ нейрона. Он опирается на два соображения:

1. Классические нейронные сети способны легко адаптироваться к небольшим изменениям входных сигналов при обучении, а значит можно постепенно заменять классические нейроны на БМ и выполнять обучение классической части сети.
2. Весовые коэффициенты классического и БМ нейронов связаны выражением (3.1), а значит эти значения коэффициентов можно использовать в качестве начальных значений.

В качестве части нейронной сети, которая будет приближаться на каждой итерации, предлагается рассматривать элементарный блок современных нейросетевых архитектур: один слой. Рассмотрение отдельных нейронов также возможно, однако представляет собой куда более вычислительно затратную задачу. В то же время, слой сети является достаточно репрезентативным блоком, позволяющим понять преимущества от последовательного преобразования сети.

Таким образом, в предложенном методе предлагается сначала обучить классическую нейронную сеть, а затем выполнить преобразование нейронов каждого слоя к БМ модели, аппроксимируя весовые коэффициенты классического слоя согласно выражению (3.1). Далее полученная нейронная сеть дообучается классическими методами, что позволяет нивелировать падение ка-

чества. Преобразование выполняется послойно от первого к последнему слою. Подробно предложенный подход изложен в Методе 1.

Метод 1: Обучение БМ сети

Входные данные: Обучающая выборка, валидационная выборка.

Выходные данные: БМ нейронная сеть.

1 Обучить классическую нейронную сеть стандартными методами.

2 **для всех** *сверточных и полносвязных слоев* **выполнить**

3 Заменить нейроны типа (1.3) с весовыми коэффициентами w БМ нейронами с весовыми коэффициентами $\{v^+, v^-, v_0\}$, где:

$$v_j^+ = \begin{cases} \ln w_j, \text{ если } w_j > 0, \\ -\infty, \text{ иначе,} \end{cases}$$
$$v_j^- = \begin{cases} \ln |w_j|, \text{ если } w_j < 0, \\ -\infty, \text{ иначе,} \end{cases} \tag{3.2}$$
$$v_0 = w_0.$$

4 Обучить полученную нейронную сеть стандартными методами.

Стоит отметить, что шаги 1-3 Метода 1 можно повторить несколько раз с разной инициализацией классической сети и выбрать наилучший результат. Кроме того, на этапе обучения возможно как обучение всей сети, так и только классической ее части.

3.2.1 Послойное преобразование и дообучение БМ моделей для классификации рукописных цифр MNIST

Для проверки эффективности метода послойного преобразования и дообучения было проведено сравнение с обучением БМ сети стандартными методами со случайной инициализацией. Для этого были рассмотрены те же модели CNN_1 и CNN_2, что и в разделе 3.1.

Они были преобразованы к БМ послойно от первого слоя к последнему предложенным методом и дообучены. Для задания БМ нейрона использова-

лось выражение (2.6) с точными функциями активации, как и в предыдущем эксперименте.

Было рассмотрено два варианта предложенного метода:

1. Преобразованная к БМ виду часть сети фиксировалась и не обучалась при дальнейшем преобразовании.
2. Преобразованная к БМ виду часть сети дообучалась вместе с не преобразованной частью.

Первый вариант позволяет выполнять преобразование и дообучение быстрее, так как зафиксированные весовые коэффициенты не требуют подсчета градиента, однако может демонстрировать меньшее качество распознавания.

Результаты экспериментов приведены в таблице 11. В ней приведены преобразуемый слой модели в обозначениях таблицы 9 и точность классификации символов MNIST сразу после преобразования (p_b) и после дообучения (p_{ft}). В качестве значений p_b «до преобразования» взяты значения точности классификации классических сетей. Для последнего слоя модели точность после дообучения приводится только для случая обучаемых БМ слоев, так как в противном случае все веса модели оказываются зафиксированы.

Таблица 11 — Точность классификации рукописных цифр на разных этапах послойного дообучения; p_b — после преобразования и до дообучения, p_{ft} — после дообучения.

Модель	Преобразованная часть	Точность, %			
		БМ фикс.		БМ не фикс.	
		p_b	p_{ft}	p_b	p_{ft}
CNN_1	до преобразования	**98.7**	-	**98.7**	-
	conv1	42.5	98.5	38.4	**98.8**
	fc1	26.9	-	19.9	94.0
CNN_2	до преобразования	**99.5**	-	**99.5**	-
	conv1	94.9	99.4	94.6	99.4
	conv2	21.3	98.7	36.2	**99.4**
	fc1	10.0	75.0	17.3	99.0
	fc2	12.9	-	48.7	97.9

Можно видеть, что p_b для любого количества преобразованных слоев в обеих моделях крайне низкое по сравнению с исходным качеством, но это сниже-

ние практически нивелируется после дообучения. Кроме того, вариант метода с фиксированием БМ части моделей демонстрирует более низкую точность распознавания, чем у классической модели, а также чем при дообучении БМ части. При этом дообучение всей сети целиком позволило преобразовать сверточные слои (conv1, conv2) без снижения точности относительно классических моделей. Однако оба варианта не смогли успешно обучить полносвязные слои в обеих рассмотренных моделях. Несмотря на это, поскольку именно сверточные слои являются ключевыми в современных нейросетевых моделях, преобразование их к БМ виду, возможное с высокой точностью благодаря предложенному методу, представляет значительный интерес.

Стоит отметить, что применение предложенного метода послойного преобразования и дообучения не ограничивается БМ моделями. Этот метод не накладывает условий на вид и форму аппроксимации, которая используется, и может применяться и в других задачах, где аппроксимируются нейросетевые модели, а обучение стандартными методами не позволяет добиться высоких результатов. Одним из важных его применений является создание нейросетевых моделей с квантованными малобитными коэффициентами. Вычисления в них аппроксимируют вычисления в классических моделях и могут быть приближены, однако прямая аппроксимация не позволяет добиться достаточного качества распознавания. Более подробному рассмотрению этой задачи посвящен следующий раздел.

3.2.2 Метод послойного дообучения целочисленных моделей

Этот раздел посвящен обобщению предложенного метода послойного дообучения и аппроксимации и проверке его работоспособности в другой актуальной задаче. В последние годы широкое распространение получили нейросетевые модели с квантованными коэффициентами малой разрядности [8; 119—121]. Такие коэффициенты можно представить и обрабатывать как целые числа, и таким образом повысить скорость их обработки. Например, в [7] автор показал, что использование 16-битных целых чисел и SIMD-расширений позволяет повысить скорость вычислений на процессорах архитектуры ARM на 40% без потери качества распознавания. Для большего повышения производитель-

ности используется по 1-8 бит на коэффициент, что позволяет им принимать 2-256 дискретных значений. Такие малобитные нейросетевые модели способны обеспечить высокую скорость работы на платформах x86/x86_64 и ARM [122], но не Эльбрус. Процессоры архитектуры Эльбрус имеют архитектуру с очень широким командным словом, которая хорошо подходит для задач с высокой степенью внутреннего параллеллизма, например, для задач распознавания [6]. Однако в этих процессорах число арифметика-логических устройств для вещественных операций превышает число устройств для целочисленных операций (даже с учетом SIMD-параллелизма), что приводит к неэффективности 8-битных нейросетевых моделей, как было показано автором в [5].

Однако преобразование вещественных коэффициентов нейронных сетей к целым числам из настолько ограниченного диапазона приводит к снижению качества распознавания без применения дополнительных методов. Обучение напрямую в малобитных типах данных затруднительно, так как подсчет градиентов и процессы оптимизации определены лишь в непрерывном пространстве коэффициентов.

Существуют несколько популярных схем квантования, сопоставляющих вещественному числу r его квантованную аппроксимацию q. Одна из них – схема с фиксированной точкой, которая выглядит следующим образом:

$$q = \min(\max(\left[2^b r\right], q_{min}), q_{max}), \tag{3.3}$$

где $[\cdot]$ обозначает округление к ближайшему целому, q_{min} и q_{max} – минимальное и максимальное значения в квантованной аппроксимации, b – число знаков в двоичной записи числа r, которое сохраняет данная схема. Обычно это значение выбирается заранее.

Рассмотрим, как вычисляется квантованнное матричное произведение при квантовании с фиксированной точкой. Пусть имеется матрица A размера M на K и матрица B размера K на N. Необходимо вычислить $C = AB$, т.е.

$$c_{ij} = \sum_{k=1}^{K} a_{ik} b_{kj} \tag{3.4}$$

Будем обозначать квантованную аппроксимацию вещественного числа x как $\hat{x}$. Тогда при использовании квантования с фиксированной точкой получим:

$$\begin{aligned} c_{ij} &\approx \frac{1}{2^b}\hat{c}_{ij} \\ \hat{c}_{ij} &= \sum_{k=1}^{K} \frac{\hat{a}_{ik}\hat{b}_{kj}}{2^b}. \end{aligned} \tag{3.5}$$

При этом расчет $\hat{c}_{ij}$ может быть выполнен в целых числах, а операция деления вычислена с помощью битового сдвига на b бит. Нейронная сеть с такой операцией умножения будет работать эффективнее, чем классическая.

Однако несмотря на это, вопрос об обучении квантованных нейронных сетей представляет большой интерес [119; 123]. Квантованная нейронная сеть является аппроксимацией исходной, но содержит много последовательных вычислений, каждое из которых выполнено с некоторой ошибкой, накапливающейся от слоя к слою. Конечный результат работы квантованной сети может сильно отличаться от исходной вещественной сети, что негативно сказывается на качестве распознавания. Классические методы обучения для решения этой проблемы не подходят, поскольку используют методы оптимизации в непрерывном пространстве, а пространство весовых коэффициентов квантованной сети дискретно.

Метод послойного преобразования и дообучения может использоваться решения этой проблемы, как было показано автором в работе «Fast Integer Approximations In Convolutional Neural Networks Using Layer-By-Layer Training» [8]. Для квантованных сетей он сформулирован в Методе 2. Его суть заключается в том, что на вход поступают обучающая и валидационная выборки, далее обучается вещественная нейронная сеть стандартными методами. После этого первый слой сети заменяется своей квантованной аппроксимацией и его весовые коэффициенты фиксируются и больше не участвуют в обучении. Вещественная часть сети дообучается пока не будет удовлетворен некоторый критерий остановки, например, пока не прекратится рост качества на валидационной выборке. Шаги 3-4 повторяются до тех пор, пока не будет квантовано заданное число слоев.

Предложенный метод был опробован экспериментально в задаче классификации печатных символов паспортов Российской Федерации. В выборке было 300000 уникальных символов из 36 классов (букв русского алфавита

Метод 2: Обучение квантованной сети

Входные данные: Обучающая выборка, валидационная выборка.

Выходные данные: Квантованная нейронная сеть.

1 Обучить классическую нейронную сеть стандартными методами.

2 **для всех** *сверточных и полносвязных слоев* **выполнить**

3 Заменить нейроны типа (1.3) с весовыми коэффициентами w_i квантованными нейронами с коэффициентами $\hat{w}_i$:

$$\hat{w}_i = \min(\max(\left[2^b w\right], q_{min}), q_{max}), \tag{3.6}$$

где $[\cdot]$ обозначает округление к ближайшему целому, q_{min} и q_{max} – минимальное и максимальное значения в квантованной аппроксимации, b – параметр аппроксимации. Коэффициенты $\hat{w}_i$ фиксируются.

4 Обучить полученную нейронную сеть стандартными методами.

и специальных символов, таких как знаки точки и тире). Примеры изображений приведены на рисунке 3.3. Символы были разделены на обучающую, валидационную и тестовую выборки из 200000, 30000 и 70000 изображений соответственно. Обучающая выборка была аугментирована также как описано в [124].

Рисунок 3.3 — Примеры изображений символов паспорта РФ.

Для распознавания использовалась 4-слойная нейронная сеть LeNet-подобной архитектуры с 2 сверточными слоями с 16 фильтрами 5×5 и 2 полносвязными слоями, показанная на рисунке 3.4.

Для обучения использовался метод стохастического градиентого спуска с моментом. Применялась L_1-регуляризация для снижения абсолютных значений весов, чтобы уменьшить ошибку аппроксимации при квантовании. Такая вещественная сеть продемонстрировала точность 99.6%. Далее применялась 8-битное квантование с фиксированной точкой со значением параметра

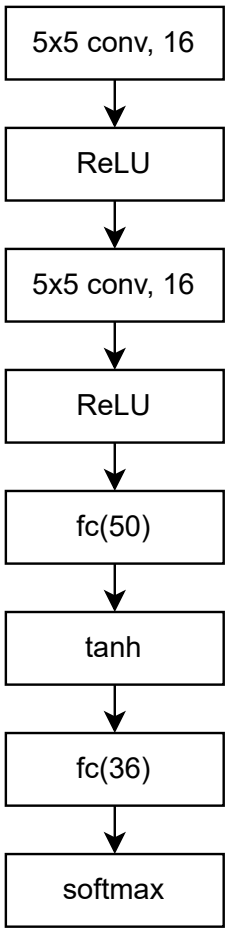


Рисунок 3.4 — Архитектура нейросетевой модели для распознавания символов паспорта РФ. Стрелками указано направление потока данных.

$b = 5$. При квантовании полученных весовых коэффициентов согласно выражению (3.6) точность составила лишь 48.6%, что является неудовлетворительным для практического применения, поэтому был применен метод послойного преобразования и дообучения. Процесс его использования представлен в таблице 12. Не преобразованная часть сети использовала вещественные числа и дообучалась двумя способами:

1. С инициализацией весовыми коэффициентами базовой сети.
2. С инициализацией случайными значениями.

Можно видеть, что преобразование коэффициентов только первого слоя уже заметно снизило точность распознавания до 87.1%. После дообучения она сооставила 99.5% при инициализации вещественными коэффициентами и 99.4% при случайной инициализации, то есть случайная инициализация проигрывает по качеству. Данное соотношение оказалось верным для обоих сверточных слоев нейросетевой модели, однако полносвязные слои лучше обучились при использовании случайной инициализации. После преобразования второго сверточного

Таблица 12 — Точность классификации символов паспорта РФ на разных этапах послойного дообучения; p_b — после квантования и до дообучения, p_f — после дообучения с инициализацией последующих слоев весовыми коэффициентами вещественной сети, p_r — после дообучения с инициализацией последующих слоев случайными коэффициентами.

Преобразованная часть	Точность, %			Ускорение, %
	p_b,	p_f	p_r	
до преобразования	99.6			-
conv1	87.1	99.5	99.4	2
conv2	77.9	98.7	98.5	20
fc1	65.6	97.4	98.2	23
fc2	80.7	97.0	98.0	25

слоя точность составила лишь 77.9%, применение дообучения позволило повысить его до 98.7%. Квантование полносвязных слоев предложенными методом продемонстрировала достаточно высокую максимальную точность в 98.0%.

Кроме того, для каждого этапа послойного преобразования и дообучения помимо точности распознавания было измерено время работы модели на процессоре Samsung Exynos 5422 архитектуры ARM. Умножение 8-битных чисел выполнялось с расширением в 16-битный тип данных, который затем снова преобразовывался к 8 битам с помощью битового сдвига. Сложение результирующих 8-битных произведений выполнялось с насыщением. Результаты также представлены в таблице 12. Можно видеть, что квантование только сверточных слоев дало значительное ускорение, которое составило 20%, а квантование всей модели – 25%.

Таким образом, автор показал, что метод послойного преобразования и дообучения не ограничен БМ моделями и может успешно применяться в задачах распознавания, например, при обучении квантованных нейронных сетей, которое широко используются для повышения скорости работы реальных приложений.

3.3 Апробация БМ моделей в практических задачах

Итак, в предыдущем разделе было показано, что БМ нейросетевые модели могут решать базовую задачу классификации символов и предложен метод для их обучения. Однако на практике многообразие задач компьютерного зрения, в которых применяются нейросетевые методы, куда шире, а архитектуры используемых моделей варьируются в значительных пределах. Поэтому в этом разделе БМ модели апробируются в двух основных категории задач, которые решаются с помощью нейросетевых моделей с помощью сетей актуальных архитектур. Это задачи визуальной классификации и семантической сегментации. В задачах визуальной классификации нейронная сеть получает на вход изображение, принадлежащее к одному из нескольких заранее определенных классов, и определяет к какому классу это изображение относится. В случае семантической сегментации входом модели является изображение, каждый пиксель которого нужно отнести к одному из нескольких заранее заданных классов. Таким образом, выходом сети также является изображение.

3.3.1 Задачи классификации

На практике сложность классифицирующих моделей может значительно варьироваться в зависимости от задачи. Поэтому рассмотрим несколько задач, задействующих модели различной сложности:

1. Задачу классификации символов машиночитаемой зоны паспортов на реальных данных с помощью LeNet-подобных моделей; сложности таких моделей вполне достаточно для обеспечения высокой точности классификации, и в то же время они достаточно достаточно вычислительно-эффективные для использования на мобильных и встраиваемых устройствах.
2. Задачу классификации рукописных цифр выборки MNIST с помощью модели ResNet-22, подробно рассмотренной в разделе 1.2.3; такая модель содержит 22 сверточных слоя и является достаточно глубокой, чтобы проиллюстрировать возможность использования БМ слоев

в глубоких нейронных сетях, но в то же время является достаточно вычислительно-эффективной для использования на мобильных и встраиваемых устройствах. В этой модели 17 из 22 слоев являются достаточно вычислительно-емкими, чтобы получить преимущество от использования БМ слоев согласно таблице 8.

3. Более сложную задачу классификации цветных изображений объектов выборки CIFAR10 также с помощью модели ResNet-22.

Классификация символов машиночитаемой зоны

Машиночитаемая зона паспортов (МЧЗ) и иных документов содержит информацию об основных реквизитах держателя документа и применяется в идентификационных документах различных стран. Она представляет собой две или три строки с символами и ее вид регламентирован стандартом «Doc 9303. Машиносчитываемые проездные документы», изданным Международной организацией гражданской авиации [125].

Для эксперимента использовалась закрытая выборка символов машиночитаемой зоны, извлеченных из реальных изображений идентификационных документов. Примеры изображений показаны на рисунке 3.5.

Приведем параметры этой выборки.

– Число классов: 37.
– Размер изображения: 21×17 пикселей, серые.
– Полный размер выборки: около $3{,}7 \times 10^5$.
– Размер валидационной выборки: $2{,}8 \times 10^4$.
– Размер тестовой выборки: $9{,}4 \times 10^4$.

Для задания БМ нейрона использовалось выражение (2.6) с точными функциями активации. Рассмотренные нейросетевые архитектуры приведены в таблице 13 и проиллюстрированы на рисунке 3.6, а в таблице 14 показаны результаты преобразования и дообучения БМ сети. Для последнего слоя модели точность после дообучения приводится только для случая обучаемых БМ слоев, так как в противном случае все веса модели оказываются зафиксированы.

Приведенные результаты показывают, что фиксирование БМ слоев сразу после преобразования приводит к низкой финальной точности распознавания

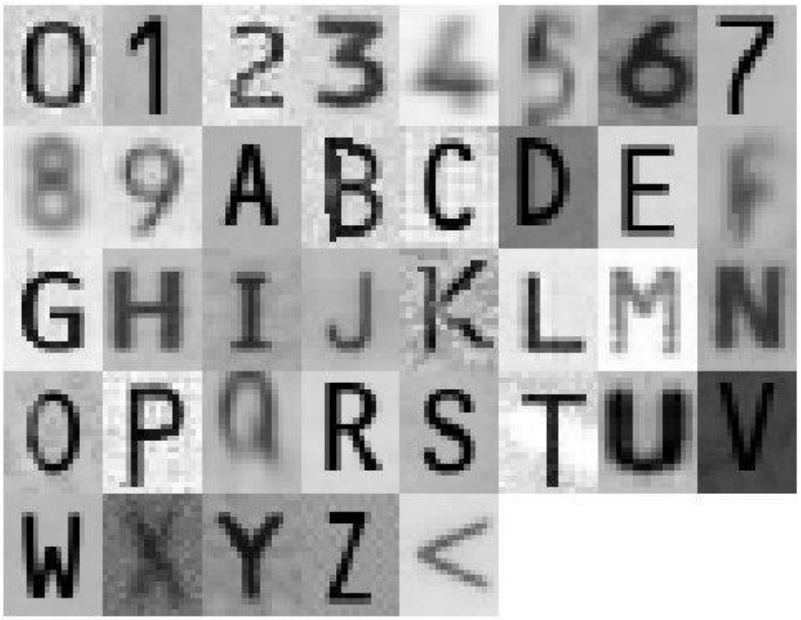
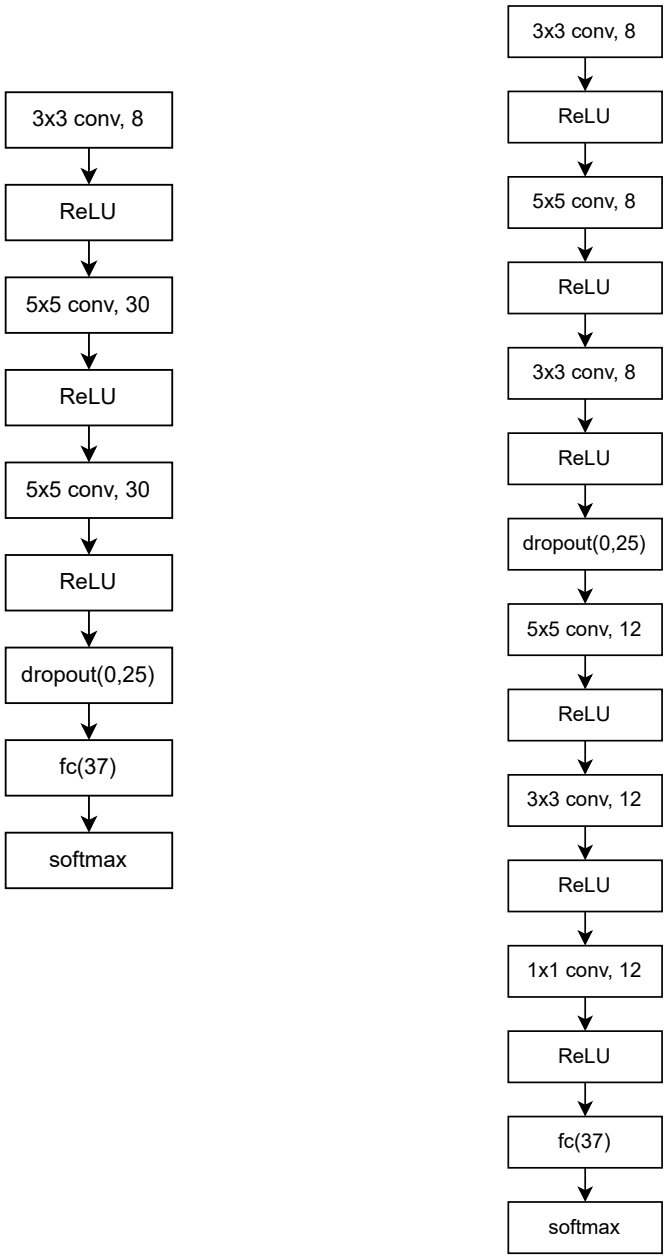


Рисунок 3.5 — Примеры изображений символов машиночитаемой зоны.

Таблица 13 — Архитектуры LeNet-подобных моделей для классификации символов МЧЗ.

Модель	Архитектура
CNN_3	conv1(8, 3, 3) - relu1 - conv2(30, 5, 5) - relu2 - conv3(30, 5, 5) - relu3 - dropout1(0,25) - fc1(37) - softmax1
CNN_4	conv1(8, 3, 3) - relu1 - conv2(8, 5, 5) - relu2 - conv3(8, 3, 3) - relu3 - dropout1(0,25) - conv4(12, 5, 5) - relu4 - conv5(12, 3, 3) - relu5 - conv6(12, 1, 1) - relu6 - fc1(37) - softmax1

на реальных данных. Это означает, что дообучение БМ слоев имеет принципиальное значение для БМ сетей. Использование БМ полносвязных слоев также значительно снижает точность классификации, как и на выборке MNIST, в то время как БМ сверточные слои преобразуются без потери точности для обеих моделей (снижение с 99.7% до 99.6% для CNN_4 можно считать незначительным и близким к значению погрешности). Таким образом, целесообразно всегда дообучать БМ слои, а также преобразовывать лишь сверточные слои. При этом нейронная сеть с БМ сверточными слоями демонстрирует точность классификации, не уступающую точности классических моделей, и может использоваться в практических задачах.



а) б)

Рисунок 3.6 — Архитектуры нейросетевых моделей для распознавания МЧЗ, а) CNN_3, б) CNN_4. Стрелками указано направление потока данных.

Таблица 14 — Точность классификации символов МЧЗ на разных этапах послойного дообучения; p_b — после преобразования и до дообучения, p_{ft} — после дообучения.

Модель	Преобразованная часть	Точность, %			
		БМ фикс.		БМ не фикс.	
		p_b	p_{ft}	p_b	p_{ft}
CNN_3	до преобразования	**99.6**	-	**99.6**	-
	conv1	97.8	99.6	83.1	99.6
	conv1 - relu1 - conv2	8.6	99.5	21.1	99.6
	conv1 - relu1 - conv2 - relu2 - conv3	3.7	98.8	36.9	**99.6**
	conv1 - relu1 - conv2 - relu2 - conv3 - relu3 - dropout1 - fc1	12.6	-	27.8	93.4
CNN_4	до преобразования	**99.7**	-	**99.7**	-
	conv1	91.2	99.7	93.7	99.7
	conv1 - relu1 - conv2	6.1	99.5	73.8	99.7
	conv1 - relu1 - conv2 - relu2 - conv3	23.6	99.4	70.3	99.7
	conv1 - relu1 - conv2 - relu2 - conv3 - relu3 - dropout1 - conv4	29.6	99.0	77.9	99.6
	conv1 - relu1 - conv2 - relu2 - conv3 - relu3 - dropout1 - conv4 - relu4 - conv5	34.2	98.5	17.1	99.6
	conv1 - relu1 - conv2 - relu2 - conv3 - relu3 - dropout1 - conv4 - relu4 - conv5 - relu5 - conv6	5.8	98.0	90.5	**99.6**
	conv1 - relu1 - conv2 - relu2 - conv3 - relu3 - dropout1 - conv4 - relu4 - conv5 - relu5 - conv6 -relu6 - fc1	4.7	-	27.6	95.5

Классификация рукописных цифр MNIST с помощью глубокой нейронной сети

В этом разделе рассмотрено преобразование глубокой нейросетевой модели, решающей задачу классификации рукописных цифр из выборки MNIST, к БМ виду. Использовалась модель архитектуры ResNet (см. раздел 1.2.3). Это современная модель, которая широко используется на практике в сложных задачах классификации. Согласно оценке вычислительной эффективности БМ нейронов наиболее эффективно использовать их на ПЛИС. ПЛИС чаще всего используются во встраиваемых системах с низким энергопотреблением и ограниченной вычислительной мощностью. Поэтому для экспериментов была выбрана модель ResNet-22, достаточно компактная для применения на мобильных устройствах и во встраиваемых системах, но в то же время обеспечивающая достаточно высокое качество классификации для решения практических задач.

Для задания БМ нейрона использовалось выражение (2.6) с точными функциями активации. Модель БМ-ResNet была обучена методом послойного преобразования и дообучения. При дообучении веса БМ нейронов не фиксировались, так как их фиксация продемонстрировала стабильное ухудшение результатов в предыдущих экспериментах.[1] Точность модели в процессе дообучения показана в таблице 15 и проиллюстрирована на рисунках 3.7 и 3.8, где показана точность модели после преобразования и до дообучения и точность после дообучения соответственно. По горизонтальной оси отложено число преобразованных к БМ виду слоев, а по вертикальной — точность такой модели. Пунктирной линией показана точность классической сети 99.3%. Можно видеть, что БМ модель с преобразованными и дообученными слоями 19 слоя не уступает в точности классической, но дальнейшее преобразование снижает ее до 99.1%. Такое снижение точности не слишком велико и полученная сеть вполне пригодна для классификации. Если же оно неприемлемо в конкретной задаче, возможно использовать гибридную, то есть частично преобразованную модель, которая не демонстрируют снижения качества распознавания, но сохраняет преимущества БМ вида для большей части слоев.

[1]Исходный код эксперимента на языке Python3 доступен по ссылке `https://github.com/SmartEngines/bipolar-morphological-resnet`

Таблица 15 — Точность классификации символов MNIST с помощью глубокой нейронной сети на разных этапах послойного дообучения; p_b — после преобразования и до дообучения, p_{ft} — после дообучения.

Преобразовано слоев	Точность, %	
	p_b	p_{ft}
до преобразования	99.3	
1	61.3	99.5
2	99.3	99.4
3	94.8	99.4
4	99.4	99.4
5	14.6	99.3
6	98.7	99.5
7	73.5	99.4
8	98.9	99.3
9	94.3	99.4
10	91.6	99.4
11	99.2	99.4
12	95.9	99.3
13	80.2	99.1
14	67.0	99.3
15	49.5	99,3
16	80.3	99.3
17	11.4	99.2
18	73.2	99.3
19	91.3	99.3
20	59.6	99.1
21	11.4	98.9
22	85.2	99.1
все conv слои	99.1	

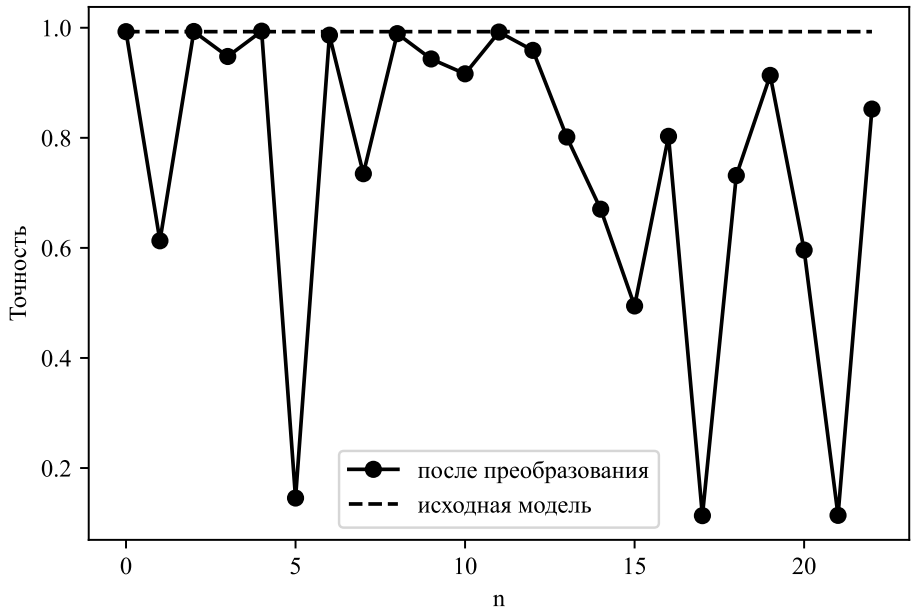
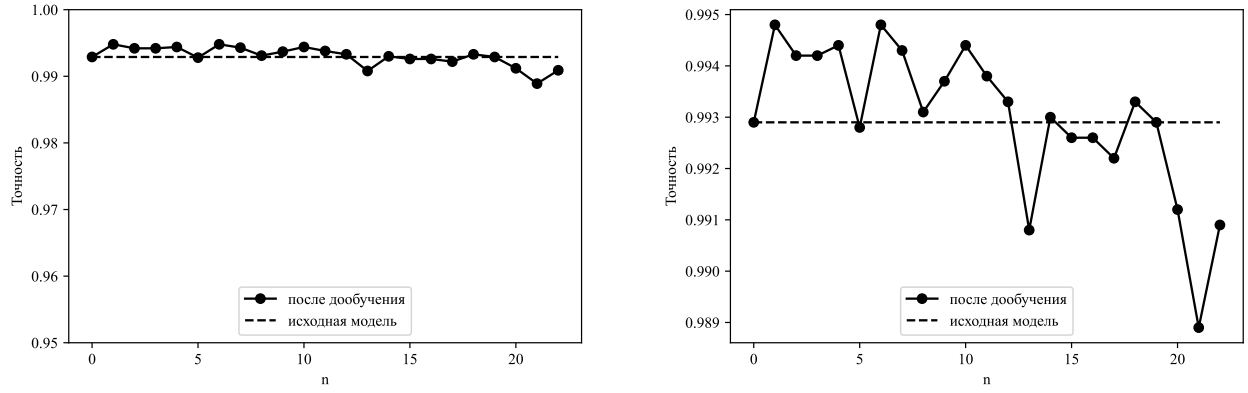


Рисунок 3.7 — Точность классификации БМ ResNet на выборке MNIST после послойного преобразования и до дообучения очередного слоя в зависимости от числа преобразованных слоев n.



а) б)

Рисунок 3.8 — Точность классификации БМ ResNet на выборке MNIST после послойного преобразования и дообучения в зависимости от числа преобразованных слоев n в диапазоне а) 0.95-1.00, б) 0.989-0.995.

БМ модель с аппроксимированными функциями активации

Предыдущие эксперименты были выполнены для БМ нейрона с точными функциями активации, логарифмом и экспонентой. Рассмотрим, как их аппроксимации, а именно аппроксимация Митчелла $\widehat{\log 2}$ и аппроксимация

Шраудольфа $\widehat{\exp 2}$, описанные в разделах 2.7.1 и 2.7.2, повлияют на точность классификации.

В этом разделе рассматривалась выборка MNIST и нейросетевая модель ResNet-22, такая же как и в предыдущем разделе. Далее была применена идея постепенного преобразования модели. В каждом БМ сверточном слое от первого к последнему:

1. Были заменены операции двоичного логарифма на аппроксимацию Митчелла, а операция возведения в степень – на аппроксимацию Шраудольфа.
2. Преобразованный слой был дообучен стандартными методами.
3. Была оценена полученная точность гибридной модели.

Для реализации дообучения с помощью обратного распространения ошибки необходимо вычислить градиенты функций $\widehat{\log 2}$ и $\widehat{\exp 2}$. В качестве производной логарифма Митчелла использовалась кусочно-постоянная функция — точная производная выражения (2.32). Аппроксимация Шраудольфа задается через битовые операции и не является дифференцируемой в таком представлении, поэтому в качестве производной использовалась производная точной функции $\exp 2'(x) = \ln 2 \cdot 2^x$.

Этот эксперимент покажет, как весовые коэффициенты слоев адаптируются к аппроксимированным версиям функций активации, может ли нейросетевая модель восстановить свое исходное качество, и, таким образом, охарактеризует выразительную силу БМ сети с $\widehat{\log 2}$ и $\widehat{\exp 2}$. Точность после послойного преобразования и дообучения показана в таблице 16 и проиллюстрирована на рисунке 3.9. Можно видеть, что точность классификации на тестовой выборке снизилась с 99.1% до 98.9%. Это означает, что доля ошибок возросла с 0.9% до 1.1%, что является 1.2-кратным ростом ошибки. Тем не менее, точность преобразованной модели все еще достаточно высока. Более того, 17-й слой сети все еще сохранил исходную точность, а снижение произошло между 17 и 22 слоями, причем это снижение было постепенно нарастающим в зависимости от числа преобразованных слоев. Это означает, что возможно создание гибридных моделей с высокой точностью и сохраняющих преимущества БМ вида.

Таблица 16 — Точность классификации символов MNIST глубокой нейронной сетью с аппроксимированными функциями активации на разных этапах послойного дообучения; p_b — после аппроксимации функций активации и до дообучения, p_{ft} — после дообучения.

Преобразовано слоев	Точность, %	
	p_b	p_{ft}
до преобразования	99.1	
1	99.1	99.1
2	99.1	99.1
3	99.0	99.1
4	99.0	99.1
5	99.1	99.1
6	99.1	99.1
7	99.1	99.1
8	99.1	99.1
9	99.1	99.1
10	99.1	99.1
11	99.1	99.1
12	99.0	99.1
13	99.0	99.0
14	99.0	99.1
15	99.0	99.1
16	99.1	99.0
17	99.0	99.1
18	99.0	99.0
19	98.9	99.0
20	98.8	99.0
21	98.8	99.0
22	98.9	98.9
все conv слои	98.9	

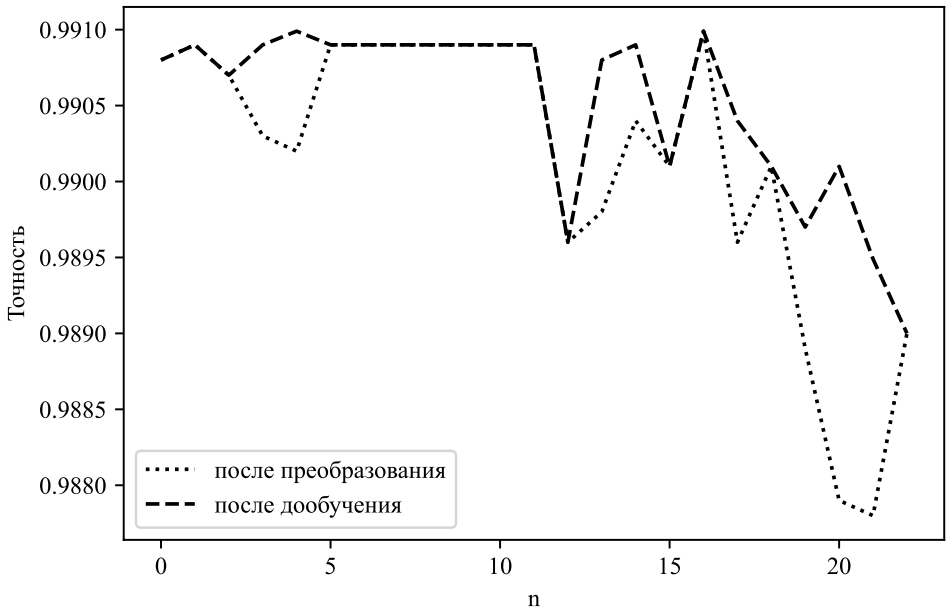


Рисунок 3.9 — Точность классификации изображений из выборки MNIST при послойной замене функций активации аппроксимированными версиями в зависимости от числа преобразованных слоев n.

Классификация объектов CIFAR10 с помощью глубокой нейронной сети

Рассмотрим более сложную задачу классификации объектов, а именно задачу классификации цветных изображений выборки CIFAR10. Выборка CIFAR10 содержит изображения объектов 10 различных классов: самолет, автомобиль, птица, кошка, олень, собака, лягушка, лошадь, корабль, грузовик [126]. Ее характеристики:

– Число классов: 10.
– Размер изображения: 32×32 пикселей, 3-канальное.
– Полный размер выборки: 60000.
– Размер валидационной выборки: 6000.
– Размер тестовой выборки: 10000.

Для решения задачи также использовалась глубокая нейросетевая модель архитектуры ResNet-22 (см. раздел 1.2.3). Для задания БМ нейрона использовалось выражение (2.6) с точными функциями активации. Модель БМ-ResNet

была обучена методом послойного преобразования и дообучения, весовые коэффициенты БМ нейронов при дообучении не фиксировались.[2]

Точность модели после преобразования и до дообучения, а также после дообучения показана в таблице 17 и проиллюстрирована на рисунках 3.10 и 3.11 соответственно. По горизонтальной оси отложено число преобразованных к БМ виду слоев, а по вертикальной — точность такой модели. Пунктирной линией показана точность классической сети 85.3%. Можно видеть, что БМ модель с преобразованными и дообученными слоями 16 слоями не уступает в точности классической, а при дальнейшем преобразовании точность начинает снижаться до 77.7% у полностью преобразованной модели. Это означает, что доля ошибок выросла в 1.5 раза, что представляет собой достаточно большой рост ошибки. Тем не менее, гибридная модель с 16 преобразованными слоями может использоваться на практике без ограничений, поскольку не приводит к снижению качества распознавания.

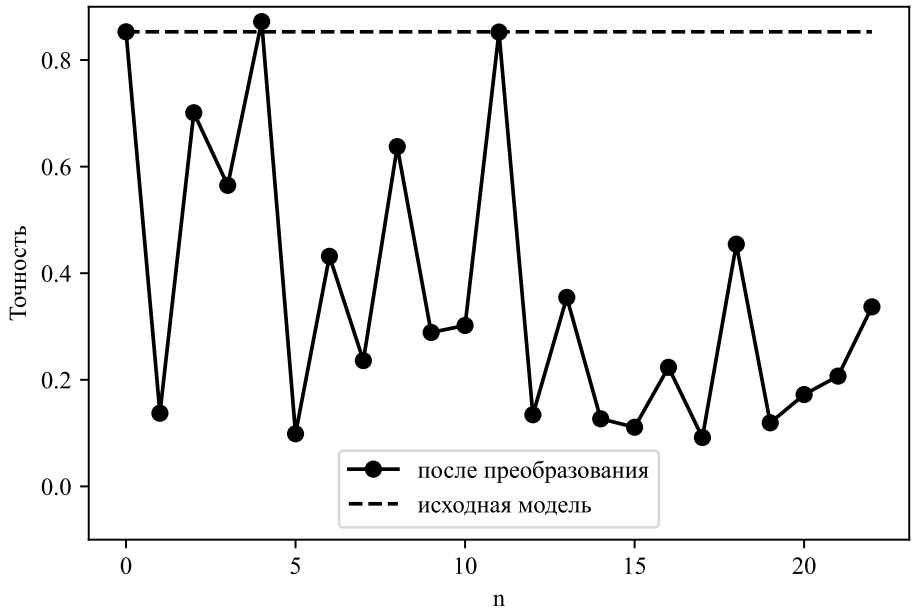


Рисунок 3.10 — Точность классификации БМ ResNet на выборке CIFAR10 после послойного преобразования и до дообучения очередного слоя в зависимости от числа преобразованных слоев n.

Итак, в этом разделе были рассмотрены три задачи классификации с использованием БМ моделей различной сложности. Эксперименты показали, что

[2]Исходный код эксперимента на языке Python3 доступен по ссылке `https://github.com/SmartEngines/bipolar-morphological-resnet`

Таблица 17 — Точность классификации объектов CIFAR10 с помощью глубокой нейронной сети на разных этапах послойного дообучения; p_b — после преобразования и до дообучения, p_{ft} — после дообучения.

Преобразовано слоев	Точность, %	
	p_b	p_{ft}
до преобразования	85.3	
1	13.7	85.4
2	70.1	86.6
3	56.5	88.0
4	87.2	89.0
5	9.9	88.9
6	43.2	89.3
7	23.6	89.1
8	63.8	89.3
9	28.9	86.2
10	30.2	85.5
11	85.3	86.6
12	13.4	86.1
13	35.5	86.6
14	12.7	85.2
15	11.1	85.4
16	22.3	85.1
17	9.2	83.6
18	45.4	83.9
19	11.9	83.1
20	17.2	82.7
21	20.7	79.6
22	33.7	77.7
все conv слои	77.7	

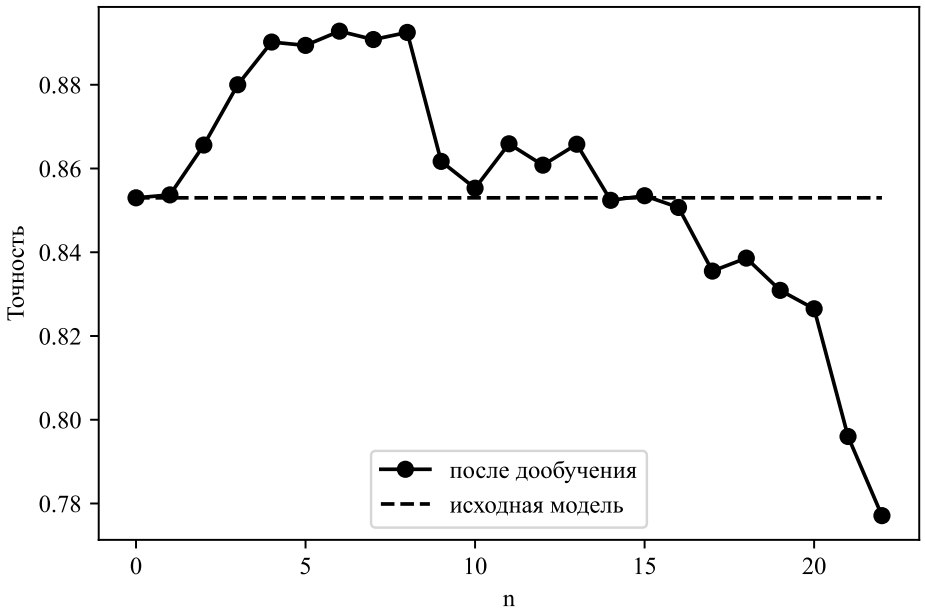


Рисунок 3.11 — Точность классификации БМ ResNet на выборке CIFAR10 после послойного преобразования и дообучения в зависимости от числа преобразованных слоев n.

БМ слои могут успешно применяться при решении задач классификации реальных данных, а также при построении глубоких сверточных моделей. При этом полностью пребразованные к БМ виду модели продемонстрировали некоторое снижение качества распознавания, которое может быть неприемлемо для некоторых задач, однако их частично преобразованные (гибридные) варианты имеют качество сопоставимое с качеством исходных моделей, а значит полностью пригодны для использования в реальных задачах.

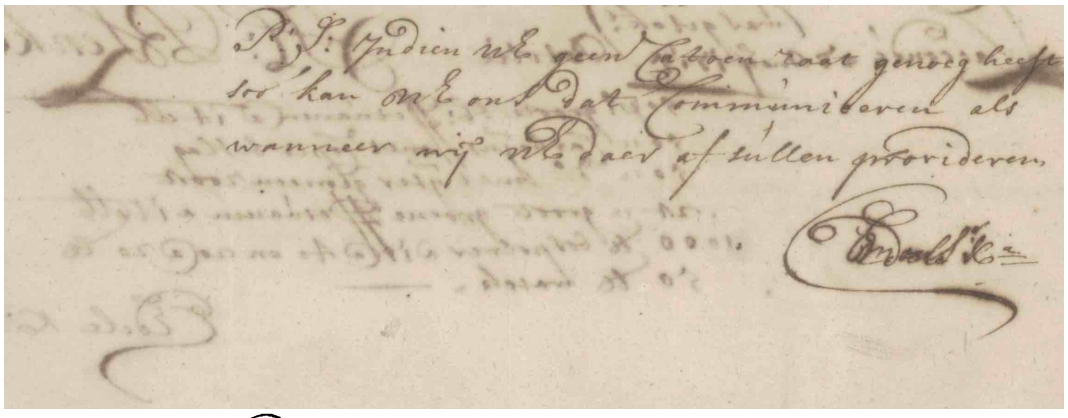
3.3.2 Семантическая сегментация

Задача семантической сегментации – задача классификации пикселей изображения, при решении которой каждый пиксель маркируется по принадлежности к одной из нескольких категорий. Это задача часто появляется при анализе биомедицинских изображений, когда нужно отделить одни типы структур или объектов от других. Одной из основных нейросетевых архитектур, которые применяются для семантической сегментации является сверточная ар-

хитектуры U-Net. Впервые она была предложена для выделения нейронных структур на изображениях, полученных с помощью электронного микроскопа в рамках конкурса IEEE International Symposium on Biomedical Imaging (ISBI) 2012 года и показала наилучший результат. На конкурсе ISBI 2015 года, где участникам предлагалось решить задачу прослеживания клеток, решение с помощью U-Net также заняло первое место [127]. На сегодняшний день эта архитектура активно развивается и применяется на практике [128; 129].

Задача бинаризации является частным случаем задачи сегментации пикселей изображения на два класса: черные и белые. В данном разделе рассмотрена задача бинаризации исторических документов с конкурса Document Image Binarization Competition (DIBCO) 2017 [130]. Организаторы предоставили 86 изображений для обучения и 20 изображений для тестирования с эталонно бинаризованными изображениями (см. пример на рисунке 3.12а, б), а также утилиты для оценки качества бинаризации.

а)



б)

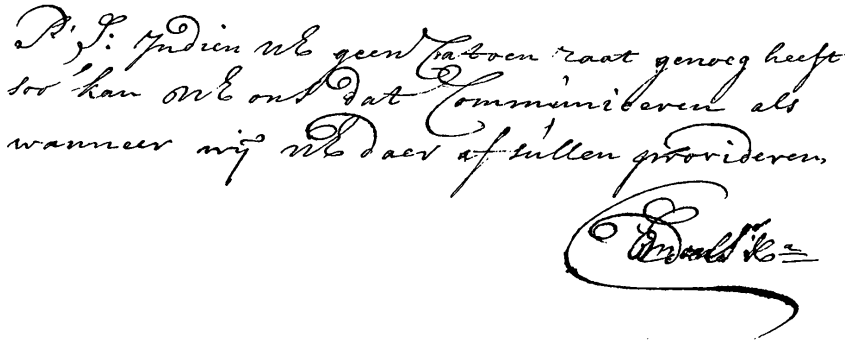


Рисунок 3.12 — Пример бинаризации: а) входное изображение, б) эталонное изображение.

Лучшее решение конкурса использовало сверточную нейронную сеть U-Net [131]. Она состоит из 10 сверточных слоев с размерами ядер 3×3 и функциями активации ReLU, а также выходной свертки с ядром размера

1×1 и сигмоидной функцией активации. Ее архитектура показана на рисунке 3.13. Для получения бинаризованного изображения, выходные величины модели округлялись.

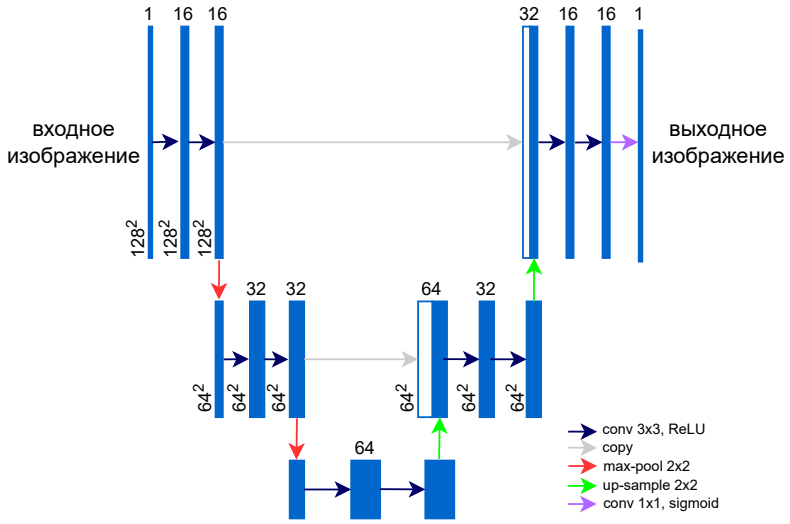


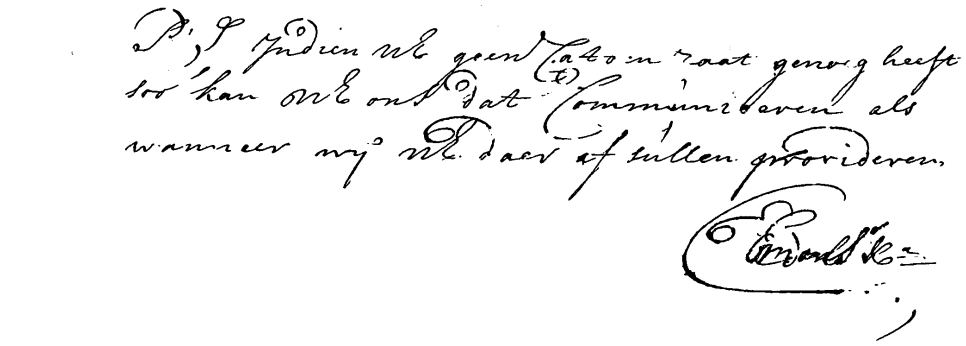
Рисунок 3.13 — Нейросетевая архитектура U-Net. Обозначения: conv — сверточный слой, copy — копирование промежуточных результатов, max-pool — слой субдискретизации с операцией максимума, up-sample — сверточный слой, повышающий размерность, sigmoid — сигмоидальная функция активации.

Далее изображения из обучающей выборки были подвергнуты осерению и разделены на блоки 128×128. Затем эти блоки были разделены на обучающую и валидационную выборки. При этом все блоки из одного изображения помещались либо в обучающую, либо в валидационную выборки. После этого данные были аугментированы с добавлением сдвигов, шума, контрастирования, масштабирования и пересечения случайными прямыми. Нейросетевая модель далее обучалась на этих данных с оптимизатором Adam [132], двоичной кросс-энтропией в качестве функции потерь и средним значением коэффициента Жаккара (mIoU) для оценки качества бинаризации. В данной работе именно это решение было использовано в качестве базового. Пример его работы показан на рисунке 3.14а.

Далее выполнялось послойное преобразование и дообучение сверточных слоев. Качество на каждом шаге преобразования показано в таблице 18. Можно видеть, что для первых двух преобразованных слоев качество практически не снизилось, для 7 слоев – снизилось слабо, но для всей сети суммарный рост ошибки составил 1.9 раза. Однако несмотря на это, БМ U-Net все еще демонстрирует высокое качество работы, пример бинаризации с его помощью показан

на рисунке 3.14б. Можно видеть, что с визуальной точки зрения качество бинаризации на разных участках изображения неоднородно: в некоторых зонах БМ-UNet демонстрирует лучший результат, а в некоторых – несколько проигрывает классической модели.

а)



б)

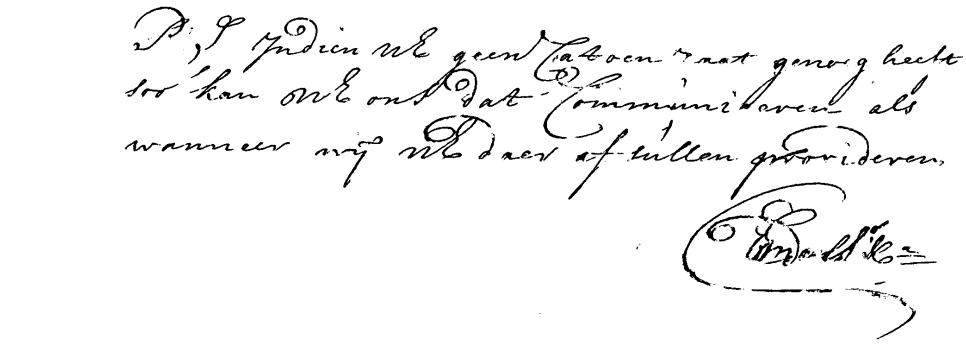


Рисунок 3.14 — Результаты бинаризации: а) с помощью U-Net, б) с помощью БМ U-Net.

Таблица 18 — Точность БМ U-Net для различного числа БМ слоев на валидационной выборке.

БМ слоев	0	1	2	3	4	5
mIoU, %	99.36	99.36	99.33	99.23	99.21	99.07
БМ слоев	6	7	8	9	10	
mIoU, %	99.11	99.08	99.02	98.85	98.79	

Поэтому далее была проведена количественная оценка качества бинаризации. Для этого использовались те же метрики, что в и конкурсе DIBCO 2017, приведенные в [130]:

1. F-мера (FM)

$$FM = \frac{2 \times Recall \times Precision}{Recall + Precision}, \tag{3.7}$$

где

$$Recall = TP + FN,$$

$$Precision = TP + FP,$$

TP – число истинно положительных результатов, FP – число ошибочно позитивных результатов, FN – число ошибочно негативных результатов.

2. псевдо F-мера (Fps)

   Fps считается также как и F-мера, с тем отличием, что используются псевдо-точность и псевдо-полнота, которые отличаются от точности и полноты наличием взвешивания [130].

3. PSNR

$$PSNR = 10 \log \frac{C^2}{MSE}, \tag{3.8}$$

   где

$$MSE = \frac{\sum_{x=1}^{M} \sum_{y=1}^{N} (I(x,y) - I'(x,y))^2}{MN}, \tag{3.9}$$

   а C – разность между первым планом и фоном на изображении. PSNR показывает сходство между изображениями I и I'. Большие значения соотвествуют более близким изображениям.

4. Distance Reciprocal Distortion Metric (DRD)

   Эта метрика используется, чтобы оценить визуальное искажение на бинарных изображениях документов [133]. Она коррелирует с визуальным восприятием человека и оценивает искажения для всех пар пикселей с разными значениям следующим образом:

$$DRD = \frac{\sum_{k=1}^{S} DRD_k}{NUBN}, \tag{3.10}$$

   где S – число пар пикселей с разными значениями, $NUBN$ – число неравномерных (не целиком белых или не целиком черных) блоков 8x8 на эталонном изображении GT, а DRD_k – оценка искажения в k-ой пары пикселей с разными значениями. Она вычисляется с использованием нормализованной матрицы коэффициентов размера 5x5 W_{Nm}, приведенной в [133] и равняется взвешенной сумме разностей пикселей

эталонного (GT) и оцениваемого изображения (B) в блоке 5x5 с центром в пикселе k с координатами (x, y):

$$DRD_k = \sum_{i=-2}^{2} \sum_{j=-2}^{2} |GT_k(i,j) - B_k(x,y)| W_{Nm}(i,j). \qquad (3.11)$$

Количественная оценка качества бинаризации по этим метрикам представлена в таблице 19. Для иллюстрации приведены другие методы бинаризации: классических методы Отсу и Сауволы, а также еще 2 лучших нейросетевых решений конкурса. Методы Отсу [134] и Сауволы [135] не являются нейросетевыми и демонстрируют достаточно посредственное качество бинаризации по всем приведенным метрикам. Решения конкурса кратко описаны в [130]. Метод 17a занял второе место и в нем использовалась полносверточная глубокая нейронная сеть, которая запускалась на преобразованном к оттенкам серого и нормированном изображении. На третьем месте оказался метод под номером 12, в котором использовался ансамбль из 3 глубоких нейросетевых моделей с архитектурами разной сложности.

Можно видеть, что БМ U-Net уступает классической модели U-Net, однако все еще значительно превосходит методы Отсу и Сауволы. Однако при этом в достаточно широком диапазоне параметров он сопоставим по качеству с вторым и третьим решениями конкурса.

Также можно отметить, что качество бинаризации достаточно равномерно снижается с ростом числа преобразованных сверточных слоев. Это означает, что на практике можно гибким образом выбирать между качеством и сложностью нейронной сети и использовать гибридные частично преобразованные модели. Таким образом, полученные результаты показывают, что БМ нейроны могут успешно применяться в задачах семантической сегментации изображений.

Таблица 19 — Сравнение качества бинаризации различными методами.

Метод	FM	Fps	PSNR	DRD
Отсу [131]	77.7	77.9	13.9	15.5
Саувола [131]	77.1	84.1	14.3	8.9
U-Net [130]	91.0	92.9	18.3	3.4
17a (полносверточная глубокая сеть) [130]	89.7	91.0	17.6	4.4
12 (ансамбль из 3 глубоких сетей) [130]	89.4	91.5	17.6	3.6
U-Net (в данной работе)	90.9	92.8	18.2	3.3
БМ-U-Net (10 БМ слоев)	85.8	88.0	17.0	5.1
БМ-U-Net (9 БМ слоев)	87.7	89.5	17.1	4.9
БМ-U-Net (8 БМ слоев)	87.2	89.4	17.3	4.7
БМ-U-Net (7 БМ слоев)	89.0	90.6	17.5	4.2
БМ-U-Net (6 БМ слоев)	88.2	90.2	17.4	4.5
БМ-U-Net (5 БМ слоев)	89.3	91.1	17.5	4.4
БМ-U-Net (4 БМ слоя)	90.5	92.1	18.0	3.6
БМ-U-Net (3 БМ слоя)	90.4	92.5	18.0	3.5
БМ-U-Net (2 БМ слоя)	90.4	92.6	18.0	3.5
БМ-U-Net (1 БМ слой)	90.9	92.4	18.0	3.4

3.4 Программный комплекс для моделирования биполярных морфологических сетей

Изложенные выше экспериментальные результаты были получены с помощью разработанного автором программного комплекса, который будет описан в этом разделе (см. Приложения А).

3.4.1 Общие сведения

Программный комплекс носит название «Программа для обучения сверточных биполярных морфологических нейронных сетей» и написан на языке программирования python3. Он предназначения для работы на пользовательских персональных компьютерах с установленным python 3 со следующим набором пакетов: numpy, os, keras, а также tensorflow. Пакет tensorflow является оптимизированным программным пакетом для обучения нейросетевых моделей и может осуществлять вычисления как на центральном процессора, так и на видеокартах с поддержкой CUDA, обеспечивая более высокую скорость работы.

Результирующее программное обеспечение может работать на операционных системах семейств Windows и Linux.

3.4.2 Функциональность

Программный комплекс обладает следующими возможностями и позволяет:

1. Обучать нейросетевые модели, использующие сверточные БМ слои.
2. Обучать классические нейросетевые модели.
3. Преобразовывать обученные классические сверточные слои к БМ виду.
4. Применять метод послойной аппроксимации и дообучения для преобразования модели к БМ виду.
5. Оценивать точность классических, гибридных и полностью БМ моделей.

3.4.3 Структура и состав программного комплекса

Разработанный автором программный комплекс состоит из следующих модулей и программ:

– модуль для обучения классических нейросетевых моделей;

– модуль для обучения БМ и гибридных нейросетевых моделей;
– модуль для преобразования коэффициентов слоя классической нейросетевой модели к биполярному морфологическому виду;
– модуль, реализующий метод послойного дообучения;
– программа оценки качества биполярной морфологической сети на произвольном наборе данных.

Его структурная схема представлена на рисунке 3.15, где прямоугольниками обозначаются модули программного комплекса, а параллелограммами – их входные данные. Пунктирный прямоугольник обозначает модуль, реализующий метод послойного преобразования и дообучения, а ромб – условие, при выполнении которого данный метод закончит работу. Стрелками обозначается процесс передачи данных на вход модулям или от одного модуля другому. Таким образом, в программном комплексе 5 модулей, взаимодействующих между собой. Опишем эти модули подробнее.

1. Модуль обучения классических нейросетевых моделей. Он реализует стандартный подход к обучению моделей и предусматривает задание архитектуры модели, обучающей и валидационной выборок, вида и параметров алгоритма градиентного спуска, а также сохранение промежуточных результатов на разных шагах обучения. В качестве входа он получает архитектуру модели в формате tensorflow, параметры обучения и обучающую и валидационную выборки, а в качестве выхода модуль выдает обученную модель в формате tensorflow.
2. Модуль для преобразования коэффициентов классического сверточного слоя к БМ виду. Он реализует выражение (3.1) для заданного сверточного слоя сети. При этом выполняется проверка на близость значений коэффициентов к нулю и логарифм таких значений заменяется на достаточно большое отрицательное число. В качестве входа модуль получает идентификатор сверточного слоя, который нужно преобразовать, а в качестве выхода возвращает модель с преобразованным к БМ виду слоем.
3. Модуль, реализующий дообучение модели. Он также реализует стандартный подход к обучению моделей, как и модуль 1, и предусматривает задание обучающей и валидационной выборок, вида и параметров алгоритма градиентного спуска, а также сохранение промежуточных результатов на разных шагах обучения. В качестве входа модуль полу-

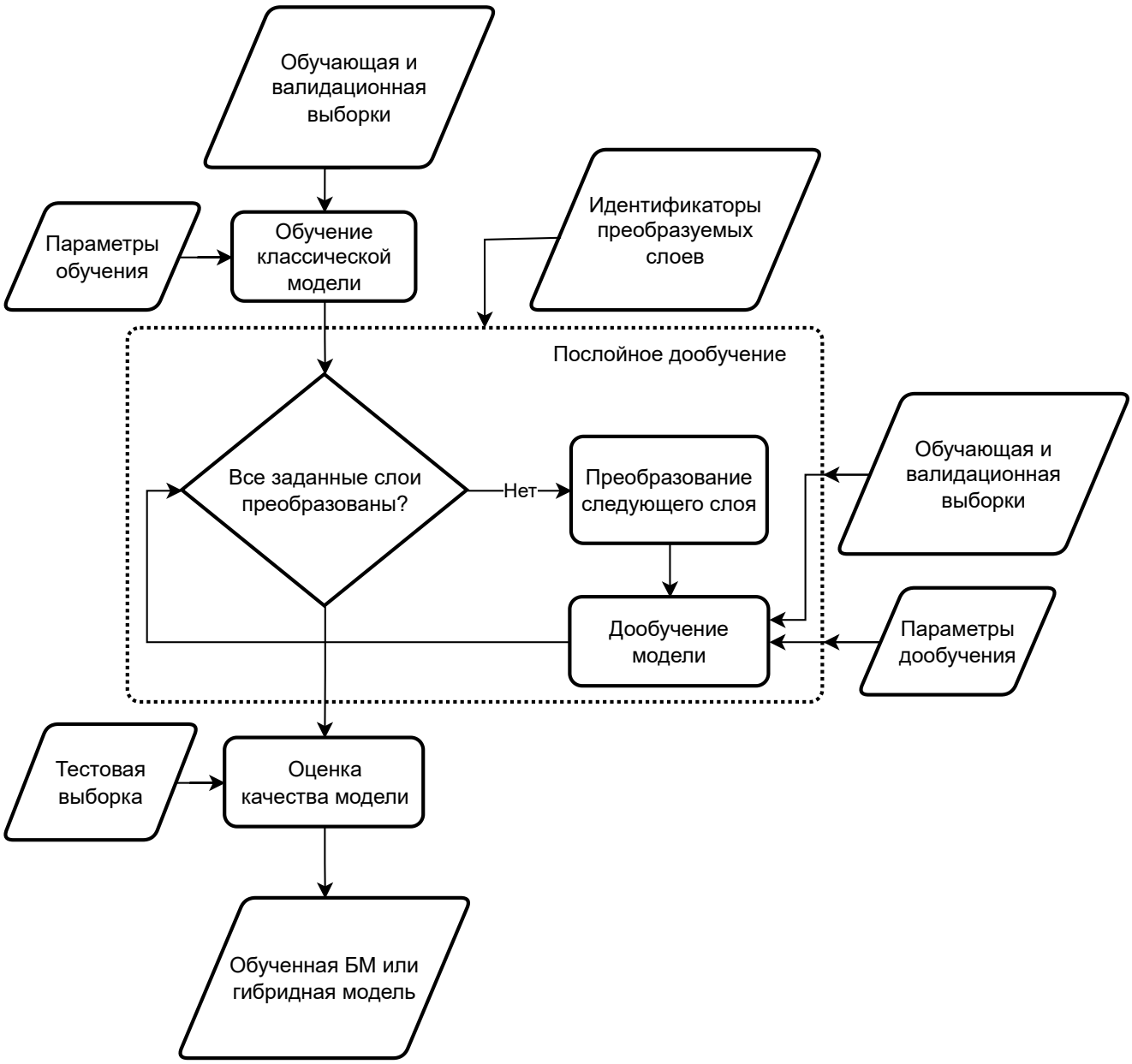


Рисунок 3.15 — Общая схема программного комплекса биполярной морфологической аппроксимации.

чает модель с БМ слоями в формате tensorflow, параметры обучения, обучающую и валидационную выборки, а в качестве выхода возвращает обученную модель в формате tensorflow.

4. Модуль послойного преобразования и дообучения. Он получает на вход идентификаторы слоев для преобразования к БМ виду, параметры обучения, а также обучающую и валидационную выборки. Для каждого преобразуемого слоя он вызывает задействует модуль преобразования коэффициентов, а затем модуль дообучения, пока все необходимые слои не будут преобразованы.

5. Программа оценки качества нейросетевой модели. Она позволяет найти точностные характеристики сети на тестовом наборе данных, который получает в качестве входа вместе с моделью. К этим характеристикам относится доля верно распознанных примеров, точность и полнота результатов классификации для классифицирующих сетей. Для сетей, осуществляющих семантическую сегментацию, она оценивает индекса Жаккара, характеризующий среднее отношение пересечения области найденного объекта с его реальным положением к их объединению.

Для реализации этих модулей также была разработана структура биполярного морфологического слоя в виде класса, совместимого с пакетом tensorflow.

3.4.4 Результаты работы программного комплекса

Используя предложенные модули и программы, можно реализовать полный цикл аппроксимации классических нейросетевых моделей: их обучение, послойное преобразование, дообучение, оценку качества распознавания полученной модели на всех этапах послойного преобразования.

В результате можно выбрать оптимальное число преобразованных слоев для рассматриваемой задачи и далее интегрировать полученную модель в конкретные приложения.

3.5 Выводы по главе 3

Данная глава посвящена экспериментальному исследованию БМ нейронов и слоев, выполненному с помощью разработанного автором программного комплекса. В ней изложен оригинальный метод послойного преобразования и дообучения, опубликованный автором в работе «Fast Integer Approximations In Convolutional Neural Networks Using Layer-By-Layer Training» [8] и «Bipolar morphological neural networks: convolution without multiplication» [4]. С ис-

пользованием этого метода автор показал, что БМ нейроны могут успешно применяться:

1. В задачах классификации реальных данных в LeNet-подобных моделях; результат опубликован в докладе «Bipolar morphological neural networks: convolution without multiplication» [4].
2. В задачах классификации с помощью глубоких нейронных сетей архитектуры ResNet; несмотря на то, что точность классификации падает с ростом числа преобразованных слоев, приводя к росту ошибки до 1.5 раз, использование частично преобразованных моделей, позволяет выбирать желаемый баланс между сложностью сети и ее точностью; результат изложен автором в работах «ResNet-like Architecture with Low Hardware Requirement» [3] и «Bipolar Morphological Neural Networks: Gate-Efficient Architecture for Computer Vision» [1].
3. В задачах семантической сегментации с помощью нейронных сетей архитектуры U-Net; несмотря на то, что точность сегментации несколько снижается с ростом числа преобразованных слоев, полученная модель все еще превосходит не-нейросетевые методы и сопоставима по качеству с другими нейросетевыми методами; результат впервые продемонстрирован автором в докладе «Bipolar Morphological U-Net for Document Binarization» [2].

Таким образом, автор продемонстрировал, что БМ нейронные сети могут успешно решать задачи визуальной классификации и семантической сегментации, которые на сегодняшний день составляют большую долю задач компьютерного зрения, решаемых нейросетевыми методами.

Заключение

Основные результаты работы заключаются в следующем.

1. Разработана аппроксимация модели математического нейрона – биполярный морфологический нейрон, – которая может применяться в сверточных и полносвязных слоях нейросетевых моделей для упрощения их внутренней структуры. При такой аппроксимации в вычислительно-интенсивных частях слоя остаются лишь операции взятия максимума и сложения, однако слой дополняется функциями активации на основе операций потенцирования и логарифмирования.
2. Аналитическими методами доказано, что нейросетевая модель с достаточным числом нейронов биполярного морфологического вида может приблизить произвольную непрерывную на компакте функцию с любой заранее заданной точностью. Это означает, что биполярные морфологические нейронные сети имеют ту же выразительную способность, что и классические модели.
3. Вычислительно-емкие сверточные биполярные морфологические слои могут быть эффективно реализованы для ПЛИС/СЛИС. Оценка числа вентилей и латентности ПЛИС-реализации для БМ сверточных слоев по сравнению с классическими сверточными слоями показала, что для слоев с достаточно большим числом входных и выходных каналов БМ слои используют практически столько же вентилей, сколько и классические слои, однако имеют латентность на 30-40% ниже; для слоев с достаточно малым числом входных каналов и размером фильтров 3×3 при использовании аппроксимированных функций активации латентность на 12-40% меньше, чем для классических слоев;
4. Для обучения аппроксимированных нейросетевых моделей предложен оригинальный метод послойного дообучения, позволивший получить лучшее качество по сравнению с обучением стандартными методами для биполярных морфологических нейросетевых моделей и квантованных 8-битных нейросетевых моделей по результатам численных экспериментов.
5. Вычислительным экспериментом показано, что биполярная морфологическая аппроксимация сверточных слоев позволяет снизить вычис-

лительную избыточность глубоких нейросетевых моделей в задачах классификации изображений и семантической сегментации без снижения качества распознавания для гибридных моделей и ряда полностью преобразованных моделей.

6. Разработан комплекс программ, позволяющий выполнить послойную и обучение аппроксимацию классической модели: обучить классическую модель, выполнить послойное преобразование к БМ виду, провести дообучение и оценить результирующее качество.

   Разработанные в рамках диссертации методы были реализованы в виде программных компонентов и внедрены в программное обеспечение «Smart ID Engine», «Smart Code Engine», «Smart Document Engine», а также «Smart IDReader» компании ООО «Смарт Энджинс Сервис». Данные продукты интегрированы в информационную инфраструктуру и мобильные приложения АО «Тинькофф Банк», а также в ряд информационных решений государственных структур Российской Федерации. Кроме того, полученные оценки и результаты моделирования демонстрируют, что включение специализированных модулей для элементарных арифметических операций при создании устройств для исполнения нейросетевых моделей способно повысить эффективность их работы и используются в АО «МЦСТ» при проектировании новых устройств.

Список литературы

1. Bipolar Morphological Neural Networks: Gate-Efficient Architecture for Computer Vision / E. E. Limonova [и др.] // IEEE Access. — 2021. — Т. 9. — С. 97569—97581.
2. *Limonova E.*, *Nikolaev D.*, *Arlazarov V. V.* Bipolar Morphological U-Net for Document Binarization // ICMV 2020. Т. 11605. — International Society for Optics, Photonics, 2021. — С. 1—9.
3. ResNet-like Architecture with Low Hardware Requirements / E. E. Limonova [и др.] // ICPR 2020. — IEEE. 2021. — С. 6204—6211.
4. Bipolar morphological neural networks: convolution without multiplication / E. Limonova [и др.] // ICMV 2019. Т. 11433. — International Society for Optics, Photonics, 2020. — С. 1—8.
5. *Limonova E. E.*, *Neyman-Zade M. I.-O.*, *Arlazarov V. L.* Special aspects of matrix operation implementations for low-precision neural network model on the Elbrus platform // Bulletin of the South Ural State University, Series: Mathematical Modelling, Programming and Computer Software. — 2020. — Т. 13, № 1. — С. 118—128.
6. Performance Evaluation of a Recognition System on the VLIW Architecture by the Example of the Elbrus Platform / E. E. Limonova [и др.] // Programming and Computer Software. — 2019. — Т. 45, № 1. — С. 12—17.
7. *Николаев Д.*, *Лимонова Е.*, *Ильин Д.* Ускорение нейросетевого распознавания образов на SIMD архитектурах // 39-я междисциплинарная школа-конференция ИТиС 2015. — ИППИ РАН, 2015. — С. 472—483.
8. Fast Integer Approximations In Convolutional Neural Networks Using Layer-By-Layer Training / D. Ilin, E. Limonova, V. Arlazarov, D. Nikolaev // ICMV 2016. Т. 10341. — International Society for Optics, Photonics, 2017. — С. 1—5.
9. *Tsoy M. O.*, *Alfonso D. M.*, *Limonova E. E.* Hardware Implementation of Classical and Bipolar Morphological Models for Convolutional Neural Network // En&T-2021. — IEEE. 2022. — С. 1—5.

10. *Limonova E. E.* Fast and gate-efficient approximated activations for bipolar morphological neural networks // Информационные технологии и вычислительные системы. — 2022. — № 2. — С. 3—10.

11. *Purves D.* Neuroscience. — Oxford University Press, 2012. — С. 759.

12. *Lapique L.* Recherches quantitatives sur l'excitation electrique des nerfs traitee comme une polarization. // Journal of Physiology and Pathololgy. — 1907. — Т. 9. — С. 620—635.

13. *Abbott L. F.* Lapicque's introduction of the integrate-and-fire model neuron (1907) // Brain research bulletin. — 1999. — Т. 50, № 5/6. — С. 303—304.

14. *Hodgkin A. L.*, *Huxley A. F.* A quantitative description of membrane current and its application to conduction and excitation in nerve // The Journal of physiology. — 1952. — Т. 117, № 4. — С. 500.

15. *Harmon L. D.* Studies with artificial neurons, I: properties and functions of an artificial neuron // Kybern. — 1961. — Т. 1, № 3. — С. 89—101.

16. *Lewis E. R.* The Locus Concept and Its Application to Neural Analogs // IEEE Transactions on Bio-medical Electronics. — 1963. — Т. 10, № 4. — С. 130—137.

17. *McCulloch W. S.*, *Pitts W.* A logical calculus of the ideas immanent in nervous activity // The bulletin of mathematical biophysics. — 1943. — Т. 5, № 4. — С. 115—133.

18. *Ritter G.*, *Sussner P.* An introduction to morphological neural networks // ICPR 1996. — 1996. — Т. 4. — С. 709—717.

19. *Ritter G. X.*, *Iancu L.*, *Urcid G.* Morphological perceptrons with dendritic structure // FUZZ 2003. Т. 2. — IEEE. 2003. — С. 1296—1301.

20. Differential evolution training algorithm for dendrite morphological neural networks / F. Arce [и др.] // Applied Soft Computing. — 2018. — Т. 68. — С. 303—313.

21. *Dimitriadis N.*, *Maragos P.* Advances in the training, pruning and enforcement of shape constraints of Morphological Neural Networks using Tropical Algebra // ICASSP 2021. — IEEE. 2021. — С. 3825—3829.

22. Dendrite morphological neural networks for motor task recognition from electroencephalographic signals / J. M. Antelis [и др.] // Biomedical Signal Processing and Control. — 2018. — Т. 44. — С. 12—24.

23. Hybrid neural networks for big data classification / G. Hernández [и др.] // Neurocomputing. — 2020. — Т. 390. — С. 327—340.

24. *Gerstner W.*, *Kistler W. M.* Spiking neuron models: Single neurons, populations, plasticity. — Cambridge university press, 2002.

25. *Izhikevich E. M.* Simple model of spiking neurons // IEEE Transactions on neural networks. — 2003. — Т. 14, № 6. — С. 1569—1572.

26. Deep learning in spiking neural networks / A. Tavanaei [и др.] // Neural Networks. — 2019. — Т. 111. — С. 47—63.

27. Direct training for spiking neural networks: Faster, larger, better / Y. Wu [и др.] // Proceedings of the AAAI Conference on Artificial Intelligence. Т. 33. — AAAI. 2019. — С. 1311—1318.

28. *Lu S.*, *Sengupta A.* Exploring the connection between binary and spiking neural networks // Frontiers in Neuroscience. — 2020. — Т. 14. — С. 535.

29. *Wang X.*, *Lin X.*, *Dang X.* Supervised learning in spiking neural networks: A review of algorithms and evaluations // Neural Networks. — 2020. — Т. 125. — С. 258—280.

30. *Rosenblatt F.* The perceptron: a probabilistic model for information storage and organization in the brain. // Psychological review. — 1958. — Т. 65, № 6. — С. 386—408.

31. *Rumelhart*, *E. D.*, *Mcclelland J.* Parallel distributed processing: explorations in the microstructure of cognition. Volume 1. Foundations. — MIT Press, 1986. — С. 567.

32. *Fukushima K.*, *Miyake S.*, *Ito T.* Neocognitron: A neural network model for a mechanism of visual pattern recognition // IEEE transactions on systems, man, and cybernetics. — 1983. — № 5. — С. 826—834.

33. Gradient-based learning applied to document recognition / Y. LeCun [и др.] // Proceedings of the IEEE. — 1998. — Т. 86, № 11. — С. 2278—2324.

34. *Krizhevsky A.*, *Sutskever I.*, *Hinton G. E.* Imagenet classification with deep convolutional neural networks // Advances in neural information processing systems. — 2012. — Т. 25. — С. 1097—1105.

35. *Bengio Y.*, *Simard P.*, *Frasconi P.* Learning long-term dependencies with gradient descent is difficult // IEEE transactions on neural networks. — 1994. — Т. 5, № 2. — С. 157—166.

36. Deep residual learning for image recognition / K. He [и др.] // CVPR 2016. — IEEE. 2016. — С. 770—778.

37. Identity mappings in deep residual networks / K. He [и др.] // ECCV 2016. — Springer. 2016. — С. 630—645.

38. *Savage J. E.* Models of Computation: Exploring the Power of Computing. — Addison-Wesley Longman Publishing Co., Inc., 1997. — С. 672.

39. Architecture of A Novel Low-Cost Hardware Neural Network / K. Khalil [и др.] // MWSCAS 2020. — IEEE. 2020. — С. 1060—1063.

40. DSP-Efficient Hardware Acceleration of Convolutional Neural Network Inference on FPGAs / D. Wang [и др.] // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. — 2020. — Т. 39, № 12. — С. 4867—4880.

41. *Shawahna A.*, *Sait S. M.*, *El-Maleh A.* FPGA-based accelerators of deep learning networks for learning and classification: A review // IEEE Access. — 2018. — Т. 7. — С. 7823—7859.

42. A survey of FPGA-based neural network inference accelerators / K. Guo [и др.] // ACM Transactions on Reconfigurable Technology and Systems. — 2019. — Т. 12, № 1. — С. 1—26.

43. Motivation for and Evaluation of the First Tensor Processing Unit / N. Jouppi [и др.] // IEEE Micro. — 2018. — Т. 38, № 3. — С. 10—19.

44. Intel Movidius Vision Processing Units [Эл. Ресурс], https://www.intel.com/content/www/us/en/products/processors/movidius-vpu.html (дата обращения: 12.11.2022).

45. Technology advancement and growth: A case study of Huawei / C. Yeo [и др.] // Journal of the Community Development in Asia. — 2020. — Т. 3, № 1. — С. 82—91.

46. Learning Separable Filters / R. Rigamonti [и др.] // CVPR 2013. — IEEE. 2013. — С. 2754—2761.

47. Exploiting Linear Structure Within Convolutional Networks for Efficient Evaluation / E. L. Denton [и др.] // Advances in Neural Information Processing Systems. Т. 27. — Curran Associates, Inc., 2014. — С. 1269—1277.

48. *Jaderberg M.*, *Vedaldi A.*, *Zisserman A.* Speeding up Convolutional Neural Networks with Low Rank Expansions // Proceedings of the British Machine Vision Conference. — BMVA Press. 2014. — С. 1—13.

49. *Jin J.*, *Dundar A.*, *Culurciello E.* Flattened convolutional neural networks for feedforward acceleration // arXiv preprint arXiv:1412.5474. — 2014.

50. Automated Multi-Stage Compression of Neural Networks / J. Gusak [и др.] // ICCVW 2019. — IEEE. 2019. — С. 2501—2508.

51. Hybrid tensor decomposition in neural network compression / B. Wu [и др.] // Neural Networks. — 2020. — Т. 132. — С. 309—320.

52. *Astrid M.*, *Lee S.-I.* Cp-decomposition with tensor power method for convolutional neural networks compression // BigComp 2017. — IEEE. 2017. — С. 115—118.

53. Stable low-rank tensor decomposition for compression of convolutional neural network / A.-H. Phan [и др.] // ECCV 2020. — Springer. 2020. — С. 522—539.

54. Tensorizing Neural Networks / A. Novikov [и др.] // Advances in Neural Information Processing Systems. Т. 28. — Curran Associates, Inc., 2015. — С. 442—450.

55. Compressing 3DCNNs based on tensor train decomposition / D. Wang [и др.] // Neural Networks. — 2020. — Т. 131. — С. 215—230.

56. Learning both Weights and Connections for Efficient Neural Network / S. Han [и др.] // Advances in Neural Information Processing Systems. Т. 28. — Curran Associates, Inc., 2015. — С. 1135—1143.

57. Exploring sparsity in recurrent neural networks / S. Narang [и др.] // arXiv preprint arXiv:1704.05119. — 2017.

58. *LeCun Y.*, *Denker J.*, *Solla S.* Optimal Brain Damage // Advances in Neural Information Processing Systems. Т. 2. — Morgan-Kaufmann, 1989. — С. 598—605.

59. *Hassibi B.*, *Stork D. G.*, *Wolff G. J.* Optimal brain surgeon and general network pruning // IEEE international conference on neural networks 1993. — IEEE. 1993. — С. 293—299.

60. *Muthukrishnan R.*, *Rohini R.* LASSO: A feature selection technique in predictive modeling for machine learning // ICACA 2016. — IEEE. 2016. — С. 18—20.

61. Sparse convolutional neural networks / B. Liu [и др.] // CVPR 2015. — IEEE. 2015. — С. 806—814.

62. Learning Structured Sparsity in Deep Neural Networks / W. Wen [и др.] // Advances in Neural Information Processing Systems. Т. 29. — Curran Associates, Inc., 2016. — С. 2074—2082.

63. *Han S.*, *Mao H.*, *Dally W. J.* Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding // arXiv preprint arXiv:1510.00149. — 2015.

64. Dynamic channel pruning: Feature boosting and suppression / X. Gao [и др.] // arXiv preprint arXiv:1810.05331. — 2018.

65. Runtime Neural Pruning / J. Lin [и др.] // Advances in Neural Information Processing Systems. Т. 30. — Curran Associates, Inc., 2017. — С. 2178—2188.

66. *Odena A.*, *Lawson D.*, *Olah C.* Changing model behavior at test-time using reinforcement learning // arXiv preprint arXiv:1702.07780. — 2017.

67. The cascading neural network: building the Internet of Smart Things / S. Leroux [и др.] // Knowledge and Information Systems. — 2017. — Т. 52, № 3. — С. 791—814.

68. Blockdrop: Dynamic inference paths in residual networks / Z. Wu [и др.] // CVPR 2018. — IEEE. 2018. — С. 8817—8826.

69. Slimmable neural networks / J. Yu [и др.] // arXiv preprint arXiv:1812.08928. — 2018.

70. Spatially adaptive computation time for residual networks / M. Figurnov [и др.] // CVPR 2017. — IEEE. 2017. — С. 1039—1048.

71. Gemmlowp: a small self-contained low-precision GEMM library [Эл. ресурс], https://github.com/google/gemmlowp (дата обращения 12.11.2022) / B. Jacob [и др.]. — 2017.

72. The ruy matrix multiplication library [Эл. ресурс], https://github.com/google/ruy (дата обращения 12.11.2022). — 2020.

73. *Dukhan M.*, *Wu Y.*, *Lu H.* QNNPACK: open source library for optimized mobile deep learning [Эл. ресурс], https://github.com/pytorch/QNNPACK (дата обращения 12.11.2022). — 2018.

74. Efficient implementation of convolutional neural networks with end to end integer-only dataflow / Y. Yao [и др.] // ICME 2019. — IEEE. 2019. — С. 1780—1785.

75. *Cai Z.*, *Vasconcelos N.* Rethinking differentiable search for mixed-precision neural networks // CVPR 2020. — IEEE. 2020. — С. 2349—2358.

76. Searching for Low-Bit Weights in Quantized Neural Networks / Z. Yang [и др.] // Advances in Neural Information Processing Systems. Т. 33. — Curran Associates, Inc., 2020. — С. 4091—4102.

77. Training quantized neural networks with a full-precision auxiliary module / B. Zhuang [и др.] // CVPR 2020. — IEEE. 2020. — С. 1488—1497.

78. *Deng X.*, *Zhang Z.* An Embarrassingly Simple Approach to Training Ternary Weight Networks // arXiv preprint arXiv:2011.00580. — 2020.

79. Simulate-the-hardware: Training accurate binarized neural networks for low-precision neural accelerators / J. Li [и др.] // ASPDAC 2019. — Association for Computing Machinery, New York, United States, 2019. — С. 323—328.

80. Single-bit-per-weight deep convolutional neural networks without batch-normalization layers for embedded systems / M. D. McDonnell [и др.] // ACIRS 2019. — IEEE. 2019. — С. 197—204.

81. Forward and backward information retention for accurate binary neural networks / H. Qin [и др.] // CVPR 2020. — IEEE. 2020. — С. 2250—2259.

82. *Li Y.*, *Bao Y.*, *Chen W.* Fixed-Sign Binary Neural Network: An Efficient Design of Neural Network for Internet-of-Things Devices // IEEE Access. — 2020. — Т. 8. — С. 164858—164863.

83. Shifted and squeezed 8-bit floating point format for low-precision training of deep neural networks / L. Cambier [и др.] // arXiv preprint arXiv:2001.05674. — 2020.

84. *Johnson J.* Rethinking floating point for deep learning // arXiv preprint arXiv:1811.01721. — 2018.

85. Hybrid 8-bit Floating Point (HFP8) Training and Inference for Deep Neural Networks / X. Sun [и др.] // Advances in Neural Information Processing Systems. Т. 32. — Curran Associates, Inc., 2019. — С. 4900—4909.

86. *Jegou H.*, *Douze M.*, *Schmid C.* Product quantization for nearest neighbor search // IEEE transactions on pattern analysis and machine intelligence. — 2010. — Т. 33, № 1. — С. 117—128.

87. *Blalock D.*, *Guttag J.* Multiplying matrices without multiplying // ICML 2021. — PMLR. 2021. — С. 992—1004.

88. Deepshift: Towards multiplication-less neural networks / M. Elhoushi [и др.] // CVPR 2021. — IEEE. 2021. — С. 2359—2368.

89. Morphological Convolutional Neural Network Architecture for Digit Recognition / D. Mellouli [и др.] // IEEE Transactions on Neural Networks and Learning Systems. — 2019. — Т. 30, № 9. — С. 2876—2885.

90. Going beyond p-convolutions to learn grayscale morphological operators / A. Kirszenberg [и др.] // DGMM 2021. — Springer. 2021. — С. 470—482.

91. *Calafiore G. C.*, *Gaubert S.*, *Possieri C.* Log-sum-exp neural networks and posynomial models for convex and log-log-convex data // IEEE transactions on neural networks and learning systems. — 2019. — Т. 31, № 3. — С. 827—838.

92. *Calafiore G. C.*, *Gaubert S.*, *Possieri C.* A universal approximation result for difference of log-sum-exp neural networks // IEEE transactions on neural networks and learning systems. — 2020. — Т. 31, № 12. — С. 5603—5612.

93. AdderNet: Do we really need multiplications in deep learning? / H. Chen [и др.] // CVPR 2020. — IEEE. 2020. — С. 1468—1477.

94. Kernel Based Progressive Distillation for Adder Neural Networks / Y. Xu [и др.] // Advances in Neural Information Processing Systems. Т. 33. — Curran Associates, Inc., 2020. — С. 12322—12333.

95. Universal Adder Neural Networks / H. Chen [и др.] // arXiv preprint arXiv:2105.14202. — 2021.

96. AdderSR: Towards Energy Efficient Image Super-Resolution / D. Song [и др.] // CVPR 2021. — IEEE. 2021. — С. 15643—15652.

97. An Empirical Study of Adder Neural Networks for Object Detection / X. Chen [и др.] // Advances in Neural Information Processing Systems. Т. 34. — Curran Associates, Inc., 2021. — С. 6894—6905.

98. Searching for Energy-Efficient Hybrid Adder-Convolution Neural Networks / W. Li [и др.] // CVPR 2022. — IEEE. 2022. — С. 1943—1952.

99. Winograd Algorithm for AdderNet / W. Li [и др.] // ICML 2021. — PMLR. 2021. — С. 6307—6315.

100. *Zhu S.*, *Li S.*, *Liu W.* iMAD: An In-Memory Accelerator for AdderNet with Efficient 8-bit Addition and Subtraction Operations // GLSVLSI 2022. — Association for Computing Machinery, 2022. — С. 65—70.

101. Conjugate Adder Net (CAddNet)-A Space-Efficient Approximate CNN / L. Shen [и др.] // CVPR 2022. — IEEE. 2022. — С. 2793—2797.

102. *Foster G. C.* The Method of Quarter-Squares // Nature. — 1889. — Т. 40. — С. 593—593.

103. EuclidNets: Combining Hardware and Architecture Design for Efficient Training and Inference / M. Prazeres [и др.] // ICPRAM 2022. — SciTePress, 2022. — С. 141—151.

104. *Carlson B. M.* Chapter 7 - Special Senses—Vision and Hearing // The Human Body. — Academic Press, 2019. — С. 177—207.

105. *Serra J.* Introduction to mathematical morphology // Computer vision, graphics, and image processing. — 1986. — Т. 35, № 3. — С. 283—305.

106. *Davidson J. L.*, *Ritter G. X.* Theory of morphological neural networks // Digital Optical Computing II. Т. 1215. — International Society for Optics, Photonics. 1990. — С. 378—388.

107. *Ильин В. А.*, *Позняк Э. Г.* Основы математического анализа. Часть 1. — Москва : Физматлит, 2004. — С. 646.

108. CPU versus GPU: which can perform matrix computation faster—performance comparison for basic linear algebra subprograms / F. Li [и др.] // Neural Computing and Applications. — 2018. — Т. 31. — С. 4353—4365.

109. *Buber E.*, *Banu D.* Performance analysis and CPU vs GPU comparison for deep learning // CEIT 2018. — IEEE. 2018. — С. 1—6.

110. *Wang Y.*, *Wei G.-Y.*, *Brooks D.* A systematic methodology for analysis of deep learning hardware and software platforms // Proceedings of Machine Learning and Systems. — 2020. — Т. 2. — С. 30—43.

111. *Fog A.* Instruction tables: Lists of instruction latencies, throughputs and micro-operation breakdowns for Intel, AMD and VIA CPUs [Эл. ресурс], https://www.agner.org/optimize/microarchitecture.pdf (дата обращения 12.11.2022). — 2017.

112. Cortex-A57 Software Optimization Guide [Эл. ресурс], https://developer.arm.com/documentation/uan0015/b (дата обращения 12.11.2022).

113. IEEE Standard for Floating-Point Arithmetic // IEEE Std 754-2019 (Revision of IEEE 754-2008). — 2019. — С. 1—84.

114. Reference Implementations for Intel® Architecture Approximation Instructions VRCP14, VRSQRT14, VRCP28, VRSQRT28, and VEXP2 [Эл. ресурс], https://www.intel.com/content/www/us/en/developer/articles/code-sample/reference-implementations-for-ia-approximation-instructions-vrcp14-vrsqrt14-vrcp28-vrsqrt28-vexp2.html (дата обращения 12.11.2022).

115. *Mitchell J. N.* Computer Multiplication and Division Using Binary Logarithms // IRE Transactions on Electronic Computers. — 1962. — Т. EC—11, № 4. — С. 512—517.

116. *Schraudolph N. N.* A Fast, Compact Approximation of the Exponential Function // Neural Computation. — 1999. — Т. 11. — С. 853—862.

117. THE MNIST DATABASE of handwritten digits, http://yann.lecun.com/exdb/mni

118. *Glorot X.*, *Bengio Y.* Understanding the difficulty of training deep feedforward neural networks // AISTATS 2010. Т. 9. — PMLR, 2010. — С. 249—256.

119. Incremental network quantization: Towards lossless cnns with low-precision weights / A. Zhou [и др.] // arXiv preprint arXiv:1702.03044. — 2017.

120. Low-power Computer Vision: Improve the Efficiency of Artificial Intelligence / G. K. Thiruvathukal [и др.]. — CRC Computer Vision, 2022. — С. 416.

121. *Ghimire D.*, *Kil D.*, *Kim S.-h.* A Survey on Efficient Convolutional Neural Networks and Hardware Acceleration // Electronics. — 2022. — Т. 11, № 6.

122. *Zhu S.*, *Duong L. H. K.*, *Liu W.* TAB: Unified and Optimized Ternary, Binary, and Mixed-Precision Neural Network Inference on the Edge // ACM Transactions on Embedded Computing Systems. — 2022. — Т. 21, № 5. — С. 1—26.

123. HAWQ: Hessian aware quantization of neural networks with mixed-precision / Z. Dong [и др.] // CVPR 2019. — IEEE. 2019. — С. 293—302.

124. *Ilin D.*, *Krivtsov V.* Creating training datasets for OCR in mobile device video stream // ECMS 2015. — European Council for Modelling, Simulation, 2015. — С. 516—520.

125. Doc 9303, Machine Readable Travel Documents, Eighth Edition 2021 [Эл. ресурс], https://www.icao.int/publications/Documents/9303_p3_cons_en.pdf (дата обращения 12.11.2022).

126. Learning multiple layers of features from tiny images. Technical Report TR-2009 / A. Krizhevsky, G. Hinton [и др.]. — 2009.

127. *Ronneberger O.*, *Fischer P.*, *Brox T.* U-Net: Convolutional Networks for Biomedical Image Segmentation // MICCAI 2015. — Cham : Springer International Publishing, 2015. — С. 234—241.

128. FF-UNet: a U-Shaped Deep Convolutional Neural Network for Multimodal Biomedical Image Segmentation / A. Iqbal [и др.] // Cognitive Computation. — 2022. — Т. 14, № 4. — С. 1287—1302.

129. Half-UNet: A Simplified U-Net Architecture for Medical Image Segmentation / H. Lu [и др.] // Frontiers in Neuroinformatics. — 2022. — Т. 16.

130. ICDAR2017 Competition on Document Image Binarization (DIBCO 2017) / I. Pratikakis [и др.] // ICDAR 2017. Т. 01. — 2017. — С. 1395—1403.

131. *Bezmaternykh P. V.*, *Ilin D. A.*, *Nikolaev D. P.* U-Net-bin: hacking the document image binarization contest // Computer optics. — 2019. — Т. 43, № 5. — С. 825—832.

132. *Kingma D. P.*, *Ba J.* Adam: A method for stochastic optimization // arXiv preprint arXiv:1412.6980. — 2014.

133. *Haiping Lu*, *Kot A. C.*, *Shi Y. Q.* Distance-reciprocal distortion measure for binary document images // IEEE Signal Processing Letters. — 2004. — Т. 11, № 2. — С. 228—231.

134. *Otsu N.* A threshold selection method from gray-level histograms // IEEE transactions on systems, man, and cybernetics. — 1979. — Т. 9, № 1. — С. 62—66.

135. *Sauvola J.*, *Pietikäinen M.* Adaptive document image binarization // Pattern recognition. — 2000. — Т. 33, № 2. — С. 225—236.

Список рисунков

Список таблиц

Приложение А

Свидетельства о государственной регистрации программ для ЭВМ

Программа для обучения сверточных биполярных морфологических нейронных сетей

Программа предназначена для обучения сверточных биполярных морфологических сетей методом послойной аппроксимации и дообучения. Программа принимает на вход обучающую, валидационную и тестовую выборки, состоящие из растровых изображений, параметры нейросетевой архитектуры, а также конфигурационный файл с параметрами метода дообучения. Результатом работы программы является набор обученных нейронных сетей, в которых сверточные слои исходной модели последовательно преобразованы к биполярному морфологическому виду и качество работы этих моделей на тестовой выборке. Основной функцией программы является автоматическое обучение сверточных биполярных морфологических сетей. Ключевыми отличительными особенностями программы являются: использование метода послойной аппроксимации и дообучения, который позволяет достичь качества распознавания, сопоставимого с качеством исходной модели; возможность оценки качества распознавания сверточной биполярной морфологической сети.

Тип ЭВМ: IBM PC совмест. ПК;

ОС: Linux, Mac OS X, Windows.

Язык программирования: Python 3

Объем программы для ЭВМ: 200 Кб

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2022660269

«Программа для обучения сверточных биполярных морфологических нейронных сетей»

Правообладатель: ***Общество с ограниченной ответственностью «Смарт Энджинс Сервис» (RU)***

Автор(ы): ***Лимонова Елена Евгеньевна (RU)***

Заявка № **2022618573**
Дата поступления **05 мая 2022 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***01 июня 2022 г.***

Руководитель Федеральной службы по интеллектуальной собственности

Ю.С. Зубов

Программа для распознавания идентификационных карт личности «Smart IDReader»

Программа предназначена для распознавания текстовых данных на сканах, фотографиях и в видеопоследовательностях идентификационных карт личности. Особенностью библиотеки Smart IDReader является выполнение всех функций на вычислительном устройстве без подключения к Интернету. Основными функциями библиотеки Smart IDReader являются: детектирование наличие образа документа на изображениях; локализация границ образов документов на изображениях; идентификация образа документа; выделение на образе документа информационных зон; распознавание текстовой информации в информационных зонах; объединение результатов распознавания полей в нескольких образах одной и той же идентификационной карты личности в видеопоследовательности.

Тип реализующей ЭВМ: процессоры ARMv7-v8 (AArch32 и AArch64), x86 and x86_64, Эльбрус

Язык программирования: C++

Вид и версия операционной системы: iOS, Android, Windows, Windows Phone, Linux, Mac OS

Объем программы для ЭВМ: 2,2 Мб

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2016616961

Программа для распознавания идентификационных карт личности "Smart IDReader"

Правообладатель: ***Общество с ограниченной ответственностью «Смарт Энджинс Сервис» (RU)***

Авторы: ***Арлазаров Владимир Викторович (RU), Николаев Дмитрий Петрович (RU), Усилин Сергей Александрович (RU), Булатов Константин Булатович (RU), Чернов Тимофей Сергеевич (RU), Слугин Дмитрий Геннадьевич (RU), Ильин Дмитрий Алексеевич (RU), Безматерных Павел Владимирович (RU), Муковозов Арсений Александрович (RU), Лимонова Елена Евгеньевна (RU)***

Заявка № **2016612014**

Дата поступления **10 марта 2016 г.**

Дата государственной регистрации в Реестре программ для ЭВМ ***22 июня 2016 г.***

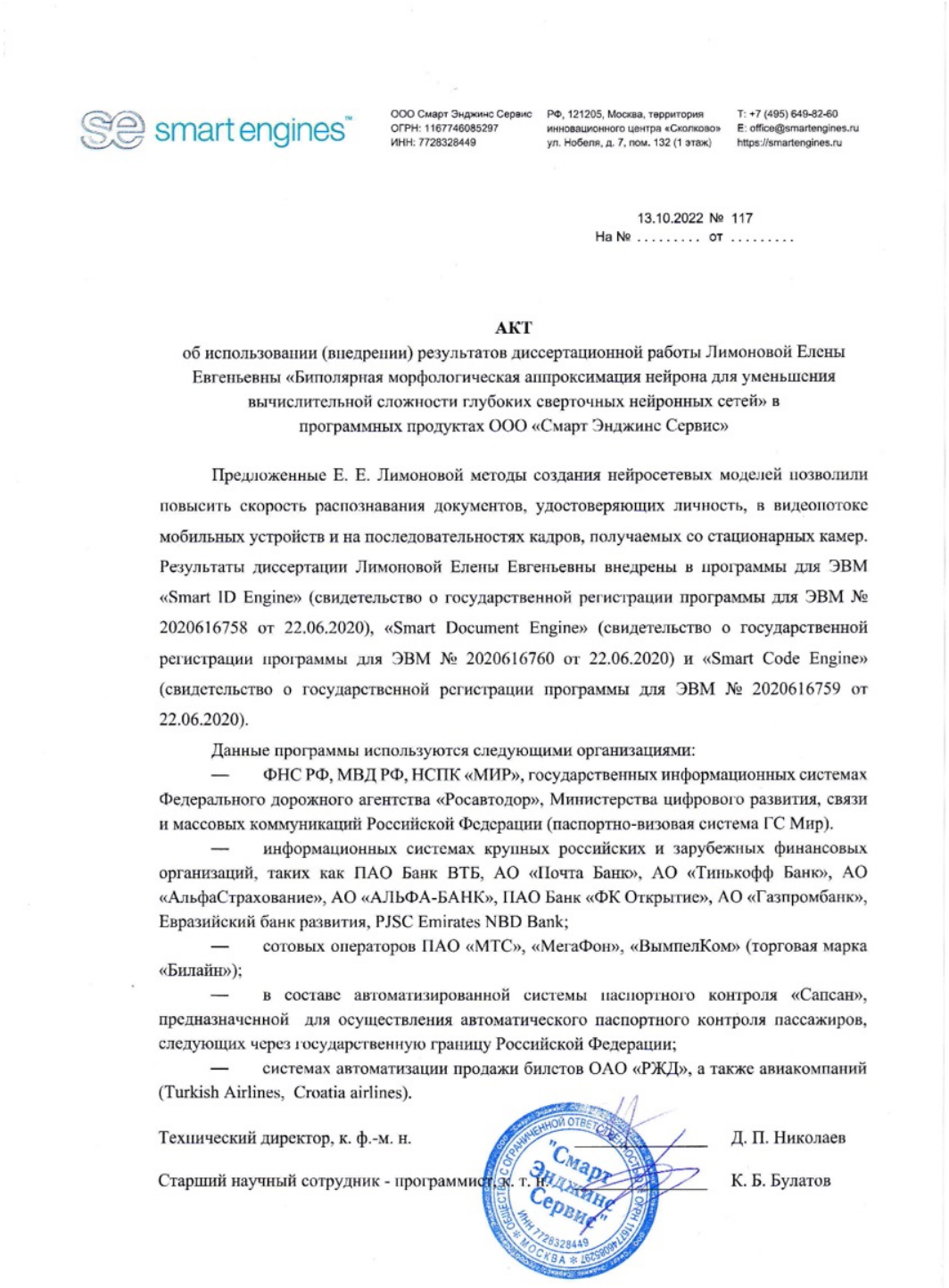
Руководитель Федеральной службы по интеллектуальной собственности

Г.П. Ивлиев

Приложение Б

Акты о внедрении

Акт о внедрении результатов диссертационной работы в деятельность ООО «Смарт Энджинс Сервис».



smart engines

ООО Смарт Энджинс Сервис
ОГРН: 1167746085297
ИНН: 7728328449

РФ, 121205, Москва, территория инновационного центра «Сколково» ул. Нобеля, д. 7, пом. 132 (1 этаж)

T: +7 (495) 649-82-60
E: office@smartengines.ru
https://smartengines.ru

13.10.2022 № 117
На № ..... от .....

АКТ

об использовании (внедрении) результатов диссертационной работы Лимоновой Елены Евгеньевны «Биполярная морфологическая аппроксимация нейрона для уменьшения вычислительной сложности глубоких сверточных нейронных сетей» в программных продуктах ООО «Смарт Энджинс Сервис»

Предложенные Е. Е. Лимоновой методы создания нейросетевых моделей позволили повысить скорость распознавания документов, удостоверяющих личность, в видеопотоке мобильных устройств и на последовательностях кадров, получаемых со стационарных камер. Результаты диссертации Лимоновой Елены Евгеньевны внедрены в программы для ЭВМ «Smart ID Engine» (свидетельство о государственной регистрации программы для ЭВМ № 2020616758 от 22.06.2020), «Smart Document Engine» (свидетельство о государственной регистрации программы для ЭВМ № 2020616760 от 22.06.2020) и «Smart Code Engine» (свидетельство о государственной регистрации программы для ЭВМ № 2020616759 от 22.06.2020).

Данные программы используются следующими организациями:

— ФНС РФ, МВД РФ, НСПК «МИР», государственных информационных системах Федерального дорожного агентства «Росавтодор», Министерства цифрового развития, связи и массовых коммуникаций Российской Федерации (паспортно-визовая система ГС Мир).

— информационных системах крупных российских и зарубежных финансовых организаций, таких как ПАО Банк ВТБ, АО «Почта Банк», АО «Тинькофф Банк», АО «АльфаСтрахование», АО «АЛЬФА-БАНК», ПАО Банк «ФК Открытие», АО «Газпромбанк», Евразийский банк развития, PJSC Emirates NBD Bank;

— сотовых операторов ПАО «МТС», «МегаФон», «ВымпелКом» (торговая марка «Билайн»);

— в составе автоматизированной системы паспортного контроля «Сапсан», предназначенной для осуществления автоматического паспортного контроля пассажиров, следующих через государственную границу Российской Федерации;

— системах автоматизации продажи билетов ОАО «РЖД», а также авиакомпаний (Turkish Airlines, Croatia airlines).

Технический директор, к. ф.-м. н. Д. П. Николаев

Старший научный сотрудник - программист, к. т. н. К. Б. Булатов

"Смарт Энджинс Сервис" ИНН 7728328449 ОГРН 1167746085297 МОСКВА

Акт о внедрении результатов диссертационной работы в деятельность АО «МЦСТ».

эльбрус

АО «МЦСТ»
ОГРН 1027739148469 | ул. Профсоюзная, д.108, Москва, 117437
ИНН 7736053886 | тел: (495) 363-96-65 факс: (495) 363-95-99
КПП 773601001 | http://www.mcst.ru e-mail: mcst@mcst.ru

_____ № _____
На № _____ от _____

АКТ

об использовании (внедрении) результатов диссертационной работы Лимоновой Елены Евгеньевны «Биполярная морфологическая аппроксимация нейрона для уменьшения вычислительной сложности глубоких сверточных нейронных сетей» в АО «МЦСТ»

В своей диссертационной работе Е. Е. Лимонова представила оценки сложности и результаты моделирования характеристик аппаратной реализации нейросетевых моделей классического и биполярного морфологического вида и показала, что они могут использоваться на практике в ряде задач распознавания. Эти результаты демонстрируют, что включение специализированных модулей для элементарных арифметических при создании устройств для исполнения нейросетевых моделей способно повысить эффективность их работы. Результаты используются в АО «МЦСТ» в процессе принятия решений при проектировании новых устройств.

Зам. Генерального директора АО МЦСТ Волконский В.Ю.

Нач. отделения АО МЦСТ Слесарев М.В.

Акт о внедрении результатов диссертационной работы в информационные системы и мобильные приложения АО «Тинькофф Банк».

АКЦИОНЕРНОЕ ОБЩЕСТВО «ТИНЬКОФФ БАНК»
РОССИЯ, 127287, МОСКВА, УЛ. 2-Я ХУТОРСКАЯ, Д. 38А, СТР. 26
ТЕЛ.: +7 495 648-10-00, TINKOFF.RU

АКТ
об использовании (внедрении) результатов диссертационной работы Лимоновой Елены Евгеньевны «Биполярная морфологическая аппроксимация нейрона для уменьшения вычислительной сложности глубоких сверточных нейронных сетей» в АО «Тинькофф Банк»

Результаты диссертационной работы «Биполярная морфологическая аппроксимация нейрона для уменьшения вычислительной сложности глубоких сверточных нейронных сетей» обладают высокой актуальностью и представляют практический интерес для решения задачи распознавания банковских карт и идентификационных документов на мобильных устройствах.

Технологии распознавания в видеопотоке на мобильных устройствах, использующие разработанные Е. Е. Лимоновой методы, позволяют повысить скорость и безопасность распознавания документов и, таким образом, улучшить качество и эффективность обслуживания клиентов в банковской сфере. Данные технологии в составе программных продуктов ООО «Смарт Энджинс Сервис» внедрены и используются в информационных системах и мобильных приложениях АО «Тинькофф Банк».

«Тинькофф Банк» был признан лучшим розничным онлайн-банком в мире в 2020 и 2018 гг. по версии Global Finance. В 2020 г. банк также стал победителем в категории «Лучший розничный европейский банк» международной банковской премии Retail Banker International Awards. Мобильное приложение банка регулярно признается лучшим на рынке российскими и международными независимыми экспертами (Deloitte в 2013, 2014, 2015 и 2016 гг., Global Finance в 2018 г.).

Директор по информационным технологиям
Вице-Президент
Заместитель Председателя Правления _____ **Цыганов В.В.**