УДК 004.032.26, 004.272, 622.006 DOI: 10.14529/cmse230401

ОБЗОР ПРИМЕНЕНИЯ ГЛУБОКИХ НЕЙРОННЫХ СЕТЕЙ И ПАРАЛЛЕЛЬНЫХ АРХИТЕКТУР В ЗАДАЧАХ ФРАГМЕНТАЦИИ ГОРНЫХ ПОРОД

© 2023 М.В. Ронкин[1], Е.Н. Акимова[1,2], В.Е. Мисилов[1,2], К.И. Решетников[1]

[1] *Уральский федеральный университет*
(620002 Екатеринбург, ул. Мира, д. 19),
[2] *Институт математики и механики им. Н.Н. Красовского УрО РАН*
(620108 Екатеринбург, ул. С. Ковалевской, д. 16)
E-mail: m.v.ronkin@urfu.ru, aen15@yandex.ru, v.e.misilov@urfu.ru, ki.reshetnikov@urfu.ru
Поступила в редакцию: 14.07.2023

Оценка производительности добычи полезных ресурсов, в том числе определение геометрических размеров объектов горной породы в открытом карьере, является одной из наиболее важных задач в горнодобывающей промышленности. Задача фрагментации горных пород решается с помощью методов компьютерного зрения, таких как экземплярная сегментация или семантическая сегментация. В настоящее время для решения таких задач для цифровых изображений используются нейронные сети глубокого обучения. Нейронные сети требуют больших вычислительных мощностей для обработки цифровых изображений высокого разрешения и больших наборов данных. Для решения этой проблемы в литературе предлагается использование облегченных архитектур нейронных сетей, а также методов оптимизации производительности, таких как параллельные вычисления с помощью центральных, графических и специализированных процессоров. В обзоре рассматриваются последние достижения в области нейронных сетей глубокого обучения для решения задач компьютерного зрения применительно к фрагментации горных пород и вопросы повышения производительности реализаций нейронных сетей на различных параллельных архитектурах.

Ключевые слова: компьютерное зрение, сверточные нейронные сети, глубокое обучение, экземплярная сегментация, семантическая сегментация, обнаружение объектов, параллельные вычисления, задачи горнодобывающей промышленности, фрагментация горных пород.

ОБРАЗЕЦ ЦИТИРОВАНИЯ

Ронкин М.В., Акимова Е.Н., Мисилов В.Е., Решетников К.И. Обзор применения глубоких нейронных сетей и параллельных архитектур в задачах фрагментации горных пород // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2023. Т. 12, № 4. С. 5–54. DOI: 10.14529/cmse230401.

Введение

Цифровизация горнодобывающей промышленности относится к области так называемой Индустрии 4.0. В настоящее время многие горнодобывающие процессы автоматизируются с использованием подходов на основе нейронных сетей (НС) глубокого обучения (Deep learning, DL). Примерами таких процессов являются: бурение, взрывные работы, транспортировка полезных ископаемых и их переработка. Несмотря на широкое использование нейронных сетей в горнодобывающей промышленности данный подход остается не достаточно исследованным по сравнению с другими видами промышленности. Более того, текущее состояние научной области не позволяет полностью автоматизировать процессы и на большинстве этапов обработки горных работ по-прежнему требуется работа оператора.

Предварительный анализ использования DL в горнодобывающей промышленности показывает, что в большинство случаев задачи сводятся к использованию сверточных нейронных сетей (СНС) в системах компьютерного зрения [1]. Примерами такого подхода яв-

ляются следующие задачи: классификация типов руды [2] классификация обогащенных пород [3]; классификация типов минералов [4]; оценка распределения размеров частиц [5]; анализ результатов бурения [6, 7]; анализ спутниковых снимков карьера [8, 9]; исследование карьера [10]; распознавание типов местности [11]; автономное управление транспортными средствами [12]; оценка качества взрывных работ на карьерах [13], и многие другие.

Результаты исследований авторов в области использования компьютерного зрения для оценки фрагментации горных пород демонстрируют перспективность данного подхода. Предложенные методы напрямую связаны с задачей определения асбестовых прожилок на кусках породы при обработке в условиях конвейерной ленты [14] и в условиях открытого карьера [15]. Проблему оценки фрагментации горных пород можно рассматривать как самостоятельную задачу или как задачу, связанную с оценкой производительности карьера [16].

В настоящее время оценка производительности карьера обычно проводится либо визуально, либо в лаборатории. Специалисты геологической службы могут проводить оперативный визуальный контроль практически в режиме реального времени, но с относительно низкой точностью. Стационарный лабораторный контроль обеспечивает высокую точность, но является трудоемким [17]. В результате лабораторный контроль может дать только среднюю оценку по всем рабочим площадкам.

С другой стороны, оценки, сделанные геологической службой (путем визуального анализа), могут быть очень субъективными. Как правило, специалисты геологической службы не могут формально описать свои алгоритмы или критерии, которые они используют для оценки производительности карьера. Более того, оценки, сделанные разными специалистами, могут существенно отличаться между собой [15]. В научной литературе обсуждаются методы автоматизации для решения таких специфических задач с использованием различных систем компьютерного зрения.

Специфику использования компьютерного зрения можно объяснить следующим образом. Типичные проблемы компьютерного зрения требуют большого количества извлекаемых признаков, которые не могут быть формально описаны. Глубокое обучение позволяет автоматически извлекать полезное признаковое описание даже из необработанных наборов данных. С другой стороны, эта задача требует довольно больших баз данных и вычислительных ресурсов, что приводит к соответствующим проблемам. Наиболее критичными проблемами являются требования к производительности вычислительных систем и требования к пространству для хранения моделей и данных [18].

Анализ литературы в области использования нейронных сетей в задачах фрагментации горных пород показывает, что в большинстве случаев используются только базовые сверточные архитектуры нейросетей без какой-либо оптимизации. Важным аспектом использования сверточных нейронных сетей является уменьшение объема памяти и оптимизация вычислений для низкопроизводительных устройств.

Целью данной статьи является обзор современных тенденций в области оптимизации глубокого обучения нейронных сетей для решения задач компьютерного зрения в задачах фрагментации горных пород и других подобных задач. Данная работа является продолжением и расширением обзорной статьи [19]. В настоящей работе рассматриваются последние достижения в области нейронных сетей глубокого обучения для решения задач компьютерного зрения и вопросы повышения производительности реализаций нейронных сетей на различных параллельных архитектурах.

В рамках достижения указанной цели мы выделяем следующие подзадачи:

- Использование современных архитектур кодировщиков признаков, специально разработанных для работы в реальном времени.
- Использование современных архитектур головных подсетей моделей, соответствующих задачам компьютерного зренрия, в частности, экземплярной сегментации, семантической сегментации и других.
- Техники оптимизации сети для использования на конечных вычислительных устройствах.

Новизна исследований в статье заключается в обзоре методов оптимизации и ускорения решения задач фрагментации в горнодобывающей промышленности. Мы ограничиваем обзор тенденциями в подходах глубокого обучения в компьютерном зрении, современными архитектурами кодировщиков признаков, а также оптимизацией их обучения и работы. Улучшения должны быть направлены на повышение вычислительной производительности при сохранении или повышении точности. Этого можно достичь путем сочетания параллельных вычислений при оптимизации нейронных сетей и использованием современных кодировщиков признаков.

Настоящая статья имеет следующую структуру. В разделе 1 кратко описана методология нашего исследования. В разделе 2 мы обсуждаем проблемы компьютерного зрения, возникающие при оценке фрагментации горных пород и аналогичные задачи в горнодобывающей промышленности. В разделе 3 мы рассматриваем современное состояние и тенденции в глубоком обучении компьютерного зрения применительно к фрагментации горных пород и аналогичных задач в горнодобывающей промышленности. В разделе 4 мы проводим обзор реализации нейронных сетей на различных типах параллельных вычислительных устройств. Основное внимание уделяется графическим процессорам, особенности использования многопроцессорных вычислений. В заключении мы формулируем основные тенденции, описанные в литературе, и наши выводы относительно перспективных методов и подходов к рассмотренной теме.

1. Методология

В нашем исследовании мы сформулировали и ответили на следующие вопросы.

- Вопрос 1. Какие методы решения задач компьютерного зрения применяются при оценке фрагментации горных пород и в смежных областях горнодобывающей промышленности?
- Вопрос 2. Какие подходы к решению соотвествующих задач компьютерного зрения являются наиболее современными?
- Вопрос 3. Какие параллельные архитектуры и методы оптимизации используются при реализации сверточных нейронных сетей?

Для сбора релевантных статей по каждому вопросу мы сформулировали запросы для поисковых систем Google Scholar и Scopus, объединив соответствующие ключевые слова.

- Запрос 1: rock fragmentation, blast quality estimation, open-pit mining, mining productivity estimation, mining industry problem, computer vision.
- Запрос 2: computer vision, deep learning, computer vision neural network architectures, feature extractors for computer vision, semantic segmentation, instance segmentation, real-time instance segmentation.

- Запрос 3: deep learning, convolutional neural networks, parallel computing, high performance, real-time performance.

При отборе работ мы использовали следующие критерии:

- Статьи должны описывать специфику использования компьютерного зрения в задачах, связанных с фрагментацией горных пород. Мы рассматриваем подходы на основе глубокого обучения нейронных сетей и классические системы компьютерного зрения.
- Статьи должны описывать современное состояние, тенденции и трудности применения глубокого обучения в компьютерном зрении относительно рассматриваемых задач. Сюда входит современное состояние т.н. кодировщика признаков (feature extractor, backbone) сверточных нейронных сетей. Также сюда входит современное состояние головных подсетей для решения задач, связанных с сегментацией в реальном времени.
- В работах должно быть описано использование параллельных архитектур для реализации нейронных сетей или методов оптимизации, связанных с нейронными сетями. Это включает специфику различных архитектур, а именно многоядерных CPU, GPGPU и специализированных ускорителей, стратегии распараллеливания и оптимизации математических операций.
- Объем работы не должен быть менее шести страниц.
- Мы рассматриваем востребованные работы, опубликованные после 2017 года или имеющие не менее 10 цитирований.

2. Современное состояние компьютерного зрения в задаче фрагментации горных пород

Анализ и оценка эффективности и производительности добычи горных пород, особенно в открытом карьере, является одной из приоритетных проблем горной промышленности. Эта задача напрямую связана с оценкой фрагментации горных пород [13, 14, 17, 20–22].

Добыча в открытом карьере предполагает сбор и переработку фрагментов горных пород, полученных в результате взрывных работ на ряде относительно небольших рабочих площадках открытого карьера. На рис. 1 показан пример рабочей площадки. Снимок сделан на Баженовском асбестовом месторождении, Россия. Полученные куски породы доставляются на обогатительную фабрику для дальнейшей переработки. В результате необходимый выход продукции для всего карьера получается из комбинации всех оценок производительности рабочих площадок внутри карьера. Качественная оценка и контроль текущей производительности на каждой рабочей площадке карьера необходимы для успешного управления карьером и фабрикой.

Пример на рис. 1 соответствует исследованию, проведенному авторами [15, 16]. Работа [16] посвящена проблеме оценки фрагментации горных пород. Работа [15] посвящена оценке производительности карьера с использованием систем компьютерного зрения путем сегментации кусков породы и асбестовых прожилок внутри них. Оба исследования демонстрируют потребность в быстрых и точных методах глубокого обучения нейронных сетей в горной промышленности. Особенно это важно при использовании мобильных устройств, а также при измерениях в реальном времени. Приведенные примеры являются типичными проблемами горнодобывающей промышленности, рассматриваемыми в данном обзоре. Схема алгоритма оценки выхода продукции асбестового волокна, который был предложен в работе [15], показана на рис. 2.

Задача, показанная на рис. 2, заключается в оценке содержания асбеста в кусках породы в открытом карьере [15]. Для каждого выбранного куска породы было сделано изображение высокого разрешения, на котором были сегментированы асбестовые прожилки. Среднее отношение площади асбестовых прожилок к площади соответствующего куска породы на изображении принимается за меру производительности карьера.

Наборы данных с размеченными изображениями для задачи на рис. 2 приводятся в работе [23] для разрабатываемых участков карьеров и в работе [24] для кусков породы с асбестовыми прожилками.

Рис. 1. Изображение рабочей площадки полученной после проведения взрывных работ

Методы оценки фрагментации горных пород при помощи систем компьютерного зрения относятся к «непрямым» методам [25]. Анализ работ в области таких систем показал, что этот подход известен в литературе как один из наиболее точных. Большинство работ можно разделить на два класса. Первый основан на применении классических методов компьютерного зрения, каких как Watershed (алгоритм водораздела). Второй подход включает использование современных методов глубокого обучения нейронных сетей. В рамках этого подхода задача фрагментации горных пород может быть решена как задача сегментации контура объекта с использованием подхода семантической сегментации (см. [20, 21, 26]) либо как задача семантической сегментации [13]. Также в ряде случаев могут быть рассмотрены подходы на основе обнаружения объектов [16]. Различия этих подходов проиллюстрированы на рис. 3.

В литературе авторы отмечают, что точность для классического подхода зависит от времени суток, погодных, сезонных и других условий [27–29]. Показано, что классический подход хорошо работает только на изображениях высокого разрешения и не дает достаточно точных результатов для больших масштабов изображений. Тем не менее, такие алгоритмы могут быть менее ресурсоемкими, чем подходы на основе глубокого обучения нейронных

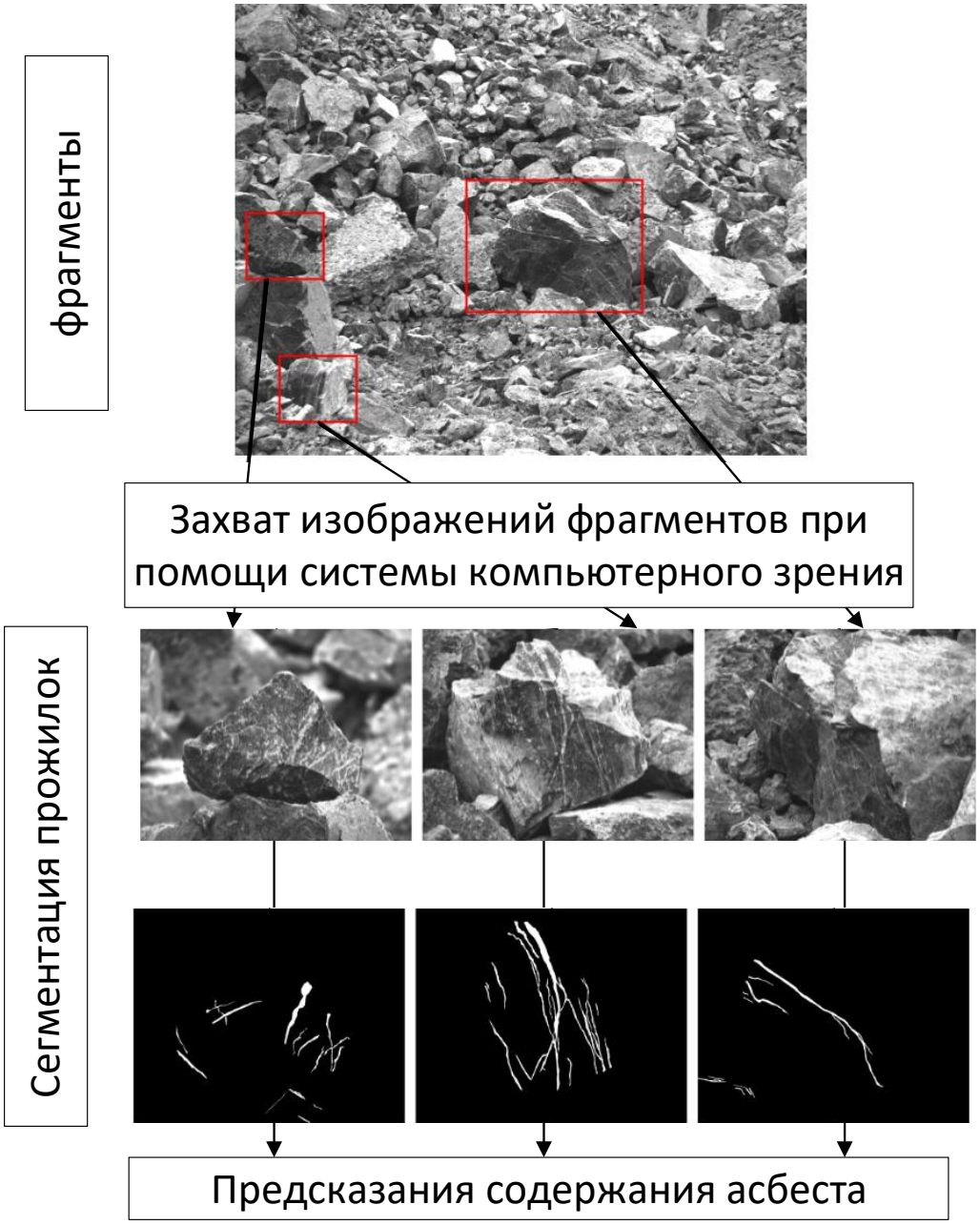
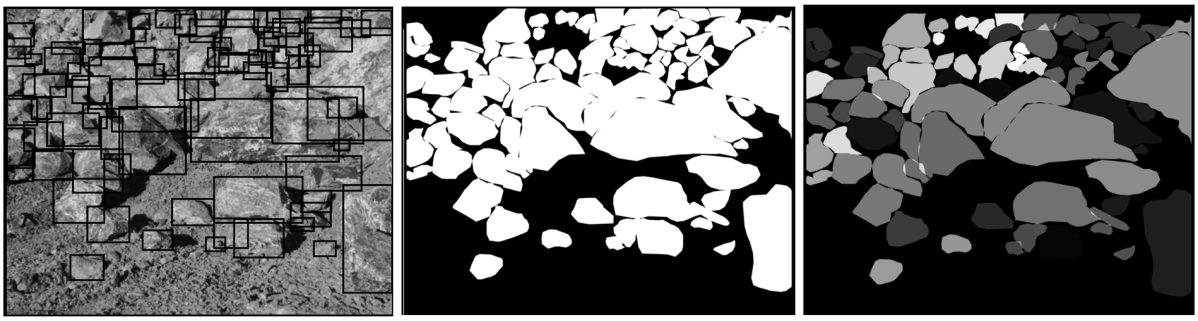


Рис. 2. Схема оценки продуктивности асбестовой породы внутри открытого карьера

сетей, применяемые в настоящее время в этой области [13, 25]. Отметим, что большинство классических решений на основе компьютерного зрения уже существует в коммерческом программном обеспечении, например, в пакетах Gold Size, Wipfrag и других. При этом данное программное обеспечение не специализировано для обработки изображений рабочих участков открытых карьеров [13, 25, 30–32]. В классическом подходе распределение размера фрагментов оценивается по значению площади круга сопоставимого размера [32]. Это значение «эквивалентного размера» может быть некорректной мерой и не отражает напрямую максимальный размер фрагмента породы, который наиболее важен с практической точки зрения в рассматриваемых задачах [13, 25]. При этом современные архитектуры нейронных сетей для задач экземплярной сегментации (instance segmentation) и семантической сегмен-

тации (semantic segmentation) способны достигать сравнимой производительности, а также более высокой точности и стабильности результатов при различных внешних условий. Подходы на основе нейронных сетей позволяют преодолеть основные недостатки классических подходов компьютерного зрения [27–29]. По этой причине далее в работе будут рассмотрены подходы на основе нейронных сетей глубокого обучения.



а) обнаружение объектов б) семантическая сегментация в) экземплярная сегментация

Рис. 3. Иллюстрация примера разметки изображения в наборе данных для задач

Задача семантической сегментации предполагает попиксельную классификацию. Выход нейронной сети должен иметь те же пространственные размеры, что и вход, а количество каналов должно быть равно количеству классов. Каждый пиксель для всех каналов взвешивается (с помощью Softmax) для определения класса, к которому он принадлежит. Среди всех сетей семантической сегментации наиболее популярным выбором является архитектура U-Net [33]. Будучи предложенной в 2015 году, эта архитектура остается сегодня одной из наиболее популярных. Рисунок 4 иллюстрирует архитектуру U-Net и принцип попиксельной классификации с применением функции Softmax. Архитектура U-Net реализует архитектуру вида кодировщик-декодировщик. Ключевой особенностью архитектуры является использование карт признаков из кодировщика в декодировщике. С 2015 года было предложено множество изменений в архитектуре U-Net. Большинство из них рассматривались для медицинских приложений [34, 35], а некоторые — для анализа микроскопических изображений [36]. Среди работ, в которых используются сети семантической сегментации, можно выделить следующие примеры. В работе [21] предложена классификация угля и пустых пород на конвейерной ленте с использованием пользовательской сверточной нейронной сети. Авторы собрали набор данных из 300 изображений, содержащих только уголь, смесь камней и угольную породу. Этого было достаточно для достижения точности распознавания 90%. Авторы [17] предлагают использовать архитектуру U-Net на смешанных изображениях для сегментации пустых пород в целях последующей сортировки.

В работе [20] рассматривается проблема определения размера руды как проблема сегментации контуров (задача семантической сегментации). Авторы предлагают использование модифицированной архитектуры Res-U-Net для определения размера руды как в условиях конвейерной ленты, так и в условиях открытого карьера. В обоих случаях получена точность около 90%. Кроме того, авторы протестировали классический алгоритм водораздела (watershed), который показывает сопоставимую точность для крупномасштабных изображений фрагментов скальной породы на конвейерной ленте, но имеет гораздо меньшую точность для изображений открытого карьера.

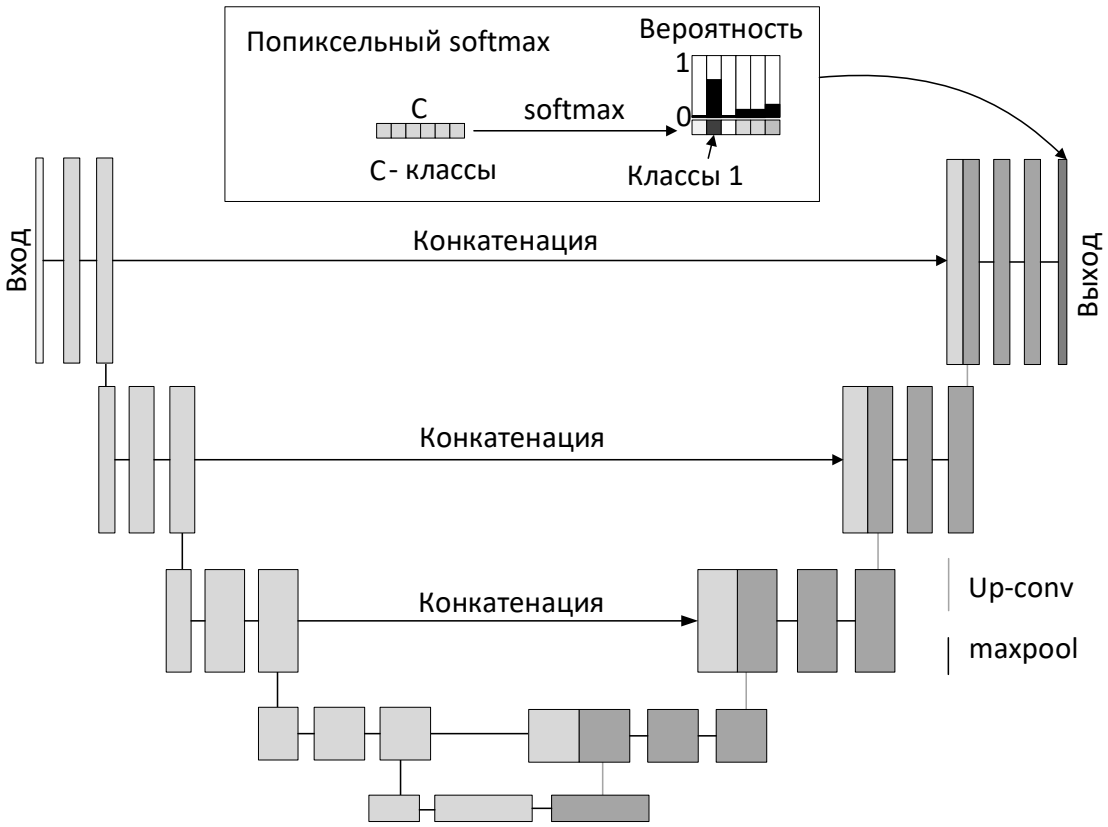


Рис. 4. Иллюстрация архитектуры U-Net и попиксельной классификации softmax

В работе [26] предложено объединить задачу сегментации при помощи U-Net с алгоритмом водораздела [37] для группировки результатов попиксельной сегментации в отдельные объекты. В статье [38] предложено использовать модифицированный алгоритм водораздела для сегментации в условиях карьера.

В статье [14] используется архитектура U-Net с модифицированными блоками для вычисления размера кусков породы и асбестовых прожилок внутри них на заводской конвейерной ленте.

Задача экземплярной сегментации в применении к глубоким нейронным сетям часто рассматривается как задача обнаружения объектов (заключение объектов в ограничивающие рамки) с последующей семантической сегментацией экземпляров в каждой ограничивающей рамке [39]. Существуют другие подходы, например, основанные на идее разделения результатов семантической сегментации на отдельные объекты [40]. Наиболее распространенной архитектурой для экземплярной сегментации является Mask R-CNN [41]. Эта архитектура является многоэтапной. Mask R-CNN состоит из общего кодировщика признаков (например, ResNet [42]); подсети предварительного предложения регионов-кандидатов (Region Proposal Network, RPN) и головной подсети. Сеть RPN предназначена для предварительного обнаружения областей изображения, содержащих объекты потенциального интереса. Для каждой области определяются габаритные размеры, координаты центра и объектность (вероятность того, что область не является фоном). Среди таких областей в головную подсеть подаются только области с площадью пересечения меньше заданного порога и объектностью выше заданного порога. Алгоритм отбора называется немаксималь-

ным подавлением (Non-Maximum supression, NMS). Результаты предварительного отбора «вырезаются» из карт признаков на выходе кодировщика и приводятся к единому размеру при помощи билинейной интерполяции. Головная подсеть состоит из подсети классификации объектов и подсети регрессии параметров ограничивающих рамок, а также из подсети семантической сегментации для каждого региона-кандидата. После работы сети алгоритм NMS применяется к окончательным регионам-кандидатам для отбора результатов [41]. Иллюстрация архитектуры Mask R-CNN показана на рис. 5. Впервые подход Mask R-CNN был предложен в 2017 году, но до сих пор остается одним из самых популярных решений для экземплярной сегментации [43].

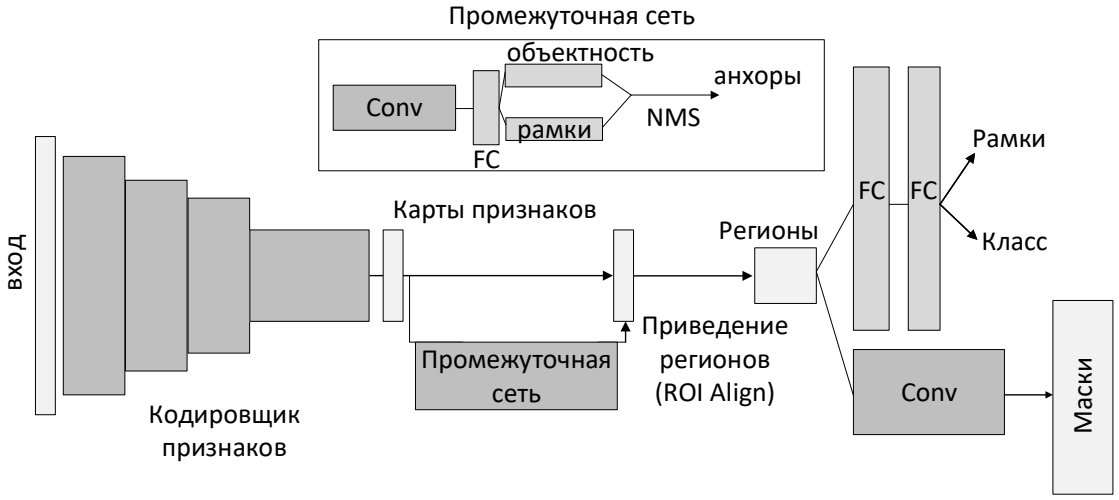


Рис. 5. Иллюстрация архитектуры Mask R-CNN

Способ использования Mask R-CNN в задачах фрагментации был предложен в работе [13], где авторы достигли точности около 90% для рабочих площадок внутри карьерах. Кроме того, было показано, что алгоритмы глубокого обучения работают лучше, чем обычные подходы компьютерного зрения для изображений высокого разрешения. Также в работе отмечена зависимость точности классических алгоритмов от времени суток, освещенности, сухой погоды и других внешних условий.

Аналогичные результаты получены в работе [44]. В статье сравниваются традиционный подход и подход на основе глубокого обучения для крупномасштабных изображений рабочих площадок внутри карьера. Авторы заявили о преимуществах Mask R-CNN для крупномасштабных изображений. Многие современные авторы утверждают, что традиционные подходы теряют отчность от таких факторов, как большой разброс размеров фрагментов, перекрывающихся экземпляров и качества фотографий, см. [13, 28, 44, 45].

В работе [15] предложен алгоритм обнаружения фрагментов породы на разрабатываемых участках карьера при помощи Mask R-CNN, а также предложено использование модифицированной архитектуры U-Net для сегментации прожилок асбестового волокна для изображения каждого из выделенных фрагментов породы.

Алгоритм состоит из четырех этапов.

1. Выявление кусков породы на полном изображении карьера.
2. Получение изображений каждого выбранного куска породы.
3. Сегментация асбестовых прожилок для каждого изображения фрагмента.

4. Получение средней оценки производительности (содержания) асбеста в карьере.

В статье [16] предлагается использовать архитектуру YOLOv5 [46] для обнаружения кусков породы и оценки их размера. Показано, что можно рассматривать задачу фрагментации как проблему обнаружения объектов. Отмечается, что использование архитектуры YOLOv5 позволяет в 10 раз ускорить решение проблемы фрагментации изображений карьеров без существенной потери точности.

Таблица 1. Современное состояние методов компьютерного зрения в задачах фрагментации горных пород

Ссылка	**Комментарий**
[33]	Архитектура U-Net, оригинальаня публикация
[34–36]	Обзор архитектуры U-Net
[47–52]	Обзор семантической сегментации
[14, 17, 20, 21, 38, 40]	Семантическая сегментация для задачи фрагментации
[41]	Архитектура Mask R-CNN, оригинальная публикация
[13, 15, 16, 44]	Mask R-CNN в задаче фрагментации
[39, 40, 43, 49, 53–55]	Обзор сегментации экземпляров

Результаты анализа литературы по применению компьютерного зрения для решения задач фрагментации горных пород и аналогичных задач в горнодобывающей промышленности показывают, что наиболее часто применяемые методы глубокого обучения нейронных сетей основаны на хорошо зарекомендовавших себя архитектурах U-Net и Mask R-CNN. Авторы не уделяют большого внимания вычислительной оптимизации предлагаемых ими алгоритмов (моделей). Хорошо известно, что эти подходы являются ресурсоемкими и неэффективно работают на низкопроизводительных конечных устройствах и в задачах, требующих реального масштаба времени вычислений [56].

Проведенный анализ расмотренных задач сегментации показывает, что такие архитектуры состоят из подсети кодировщика признаков (энкодер, feature extractor, backbone), промежуточной подсети (neck part) и головной подсети (решающая подсеть, head part, decision part). При этом часто именно кодировщик признаков оказывает наибольшее влияние на точность и вычислительную сложность модели. Заметим, что в упомянутых работах не уделяется внимание проблеме выбора кодировщика признаков и оптимизации его работы.

Основное решение соответствующих проблем состоит в следующем: [47–50, 57, 58]:

- использование современных, специально разработанных для реального масштаба времени архитектур кодировщиков признаков;
- обоснованный с точки зрения компромисса «точность — скорость работы» выбор общей архитектуры решения задачи;
- техника оптимизации модели нейронной сети для конечных вычислительных устройств.

Обзор наиболее важных работ по этому разделу представлен в табл. 1.

В следующем разделе описывается современное состояние в области архитектур нейронных сетей глубокого обучения для решения вышеописанных задач компьютерного зрения.

Все рассмотренные в следующем разделе архитектурные решения могут быть использованы в оптимизации задачи фрагментации горных пород.

3. Методы глубокого обучения

3.1. Архитектура сверточных нейроннных сетей

В разделе представлен обзор тенденций в области глубокого обучения нейронных сетей в задачах компьютерного зрения, релевантных исследуемой тематике. В частности, рассмотрены современные архитектуры кодировщиков признаков с точки зрения их исторического развития. Отметим, что основным методом исследования архитектур кодировщиков признаков является экспериментальное тестирование для решения типичных задач классификации. Поэтому данный тип архитектур рассмотрен через призму именно этой задачи. После решения задачи классификации такой кодировщик признаков может быть перенесен на другие задачи компьютерного зрения.

Идея использования сверточной архитектуры (сверточная нейронная сеть, СНС, Convolutional Neural Network, CNN) в компьютерном зрении была предложена в 1989 году для распознавания почтовых индексов [59]. Современный подход к построению архитектур сверточных нейронных сетей был предложен в 1998 году в модели LeNet [60]. Иллюстрация LeNet показана на рис. 6. С этой архитектуры начинается исследование подхода глубоких нейронных сетей, в котором производится автоматическое извлечение признаков из «сырых» данных.

Автоматизированное извлечение признаков является основным преимуществом использования глубокого обучения в компьютерном зрении. Хорошо известно, что большинство задач компьютерного зрения трудно описать формально. Такое формальное описание можно заменить экстраполяцией результатов для набора примеров (обучающей выборки). Для этого на практике необходим хорошо подобранный и достаточно большой набор данных. Для решения задачи в такой постановке глубокая нейронная сеть включает два основных компонента: кодировщик признаков и подсеть, принимающая решения. Кодировщик признаков СНС состоит из последовательных комбинаций сверточных и вспомогательных слоев. Отметим, что применительно к компьютерному зрению операция свертки имеет несколько интуитивных преимуществ: инвариантность по масштабу, расположению и вращению признаков, а также меньшее количество параметров по сравнению с другими подходами [61, 62]. В настоящее время эти преимущества делают сверточные нейронные сети основным способом решения задач компьютерного зрения.

Современное развитие глубокого обучения СНС началось со сверточной архитектуры AlexNet (2012) [63, 64]. AlexNet состоит из 7 слоев. Сеть включает в себя такие особенности, как функция активации Rectified Linear Unit (ReLU); слой регуляризации DropOut; набор встроенных техник расширения (augmentation) изображений; слой max pooling для уменьшения размерности и другие приемы оптимизации модели [63]. Это была первая попытка интегрировать все современные на тот момент подходы как к построению архитектуры сети, так и к процедуре ее обучения.

После AlexNet следующие несколько лет прогресса глубокого обучения сверточных нейронных сетей были посвящены тому, чтобы сделать архитектуру более глубокой. Идея заключалась в увеличении так называемого рецептивного поля [61].

Предложены следующие модификации, которые сейчас являются распространенной практикой:

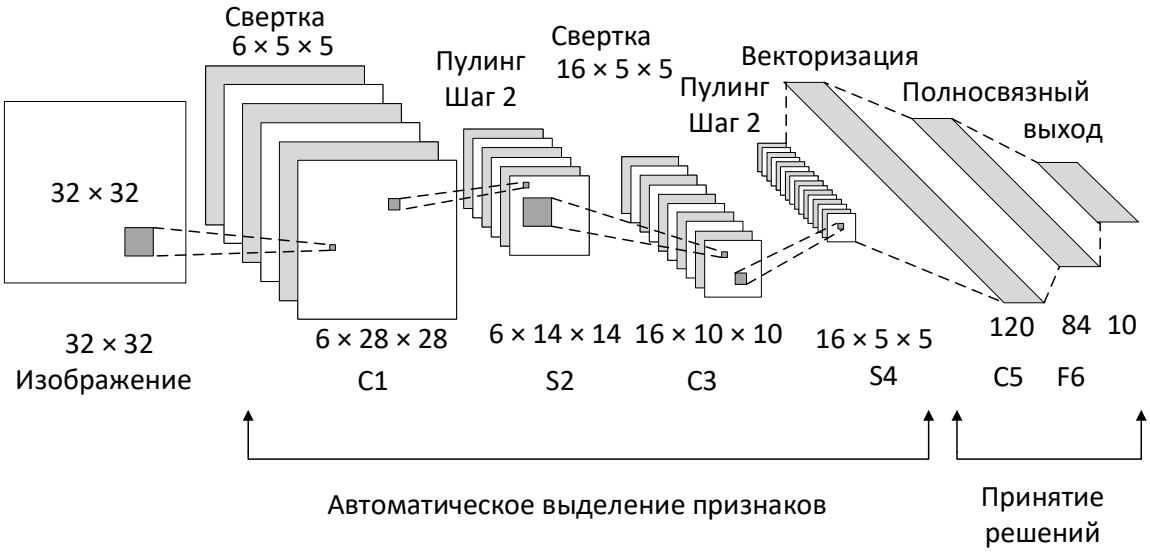
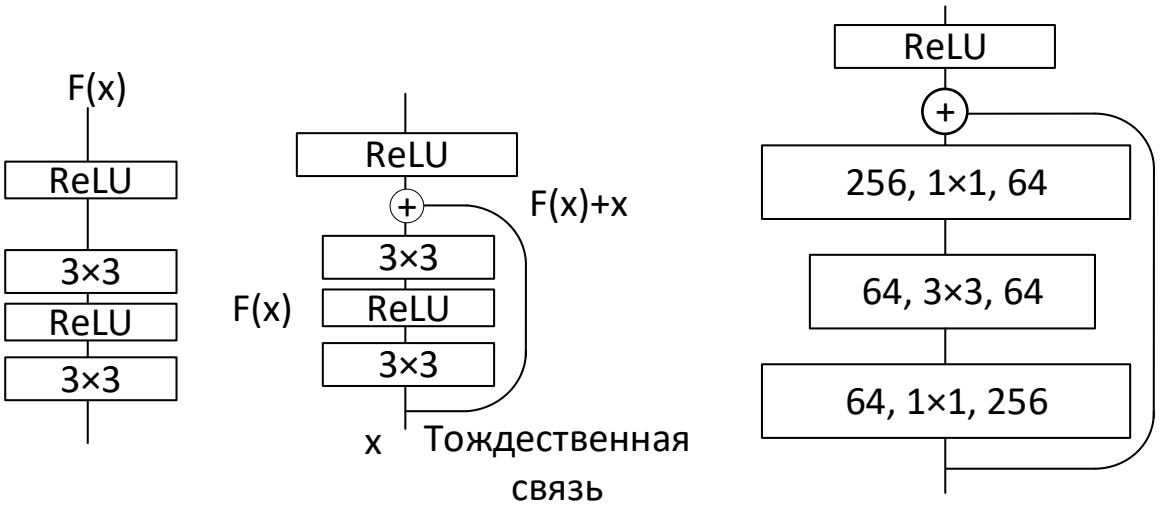


Рис. 6. Иллюстрация архитектуры сверточной нейронной сети LeNet

- оптимизация сверточного слоя с использованием различных подходов: каскадная свертка [65]; пространственно разделяемая свертка [66]; точечная свертка [67]; свертка по глубине [68];
- блок слоев с различным рецептивным полем [69];
- блоки слоев с параллельным соединением [42];
- пакетная нормализация как основной способ регуляризации [70];
- метод глобального пулинга [67];
- адаптивная оптимизация статистических моментов низшего порядка на основе градиента (ADAM) [71].

«Гонка за глубиной» закончилась в 2016 году, когда была создана модифицированная версия ResNet, включающая более 1000 слоев [72]. Было отмечено, что количество слоев больше 150 не имеет существенного эффекта. Идея остаточных слоев (residual connection, skip connection) стали доминирующими в проектировании архитектур сверточной нейронной сети. Показано, что этот прием устраняет практически все негативные эффекты, которые могут возникнуть в процессе обучения сверточной нейронной сети. Иллюстрация схемы блоков ResNet показана на рис. 7. Архитектура ResNet остается самой популярной архитектурой СНС в литературе [73]. В последние годы было предложено множество модификаций ResNet [74]. Блок ResNeXt [75] предполагает разделение основной ветви блока ResNet на несколько для повышения различия извлекаемых признаков. Блок Xception [68] расширяет результаты ResNeXt, используя отдельную свертку для каждой карты признаков. Блок DenseNet [76] предлагает объединить несколько слоев с каскадом остаточных связей для увеличения обмена информацией внутри каждого такого блока. Блок ResNet-D [77] улучшает архитектуру блока ResNet и предлагает множество практических приемов для ее обучения. Подход BiT [78] предлагает приемы для эффективного «переноса обучения» весовых параметров ResNet-подобных архитектур, а также правила «тонкой» настройки и способы избежать недостатков пакетной нормализации. Подход RegNet [79], предлагает несколько эффективных модификаций блоков ResNet, полученных методами автоматического поиска конфигураций блоков.



а) сверточный блок б) остаточный блок в) блок узкого горла

Рис. 7. Иллюстрация различных блоков остаточного соединения

Другой важный этап развития глубокого обучения СНС заключался в оптимизации архитектуры для малопроизводительных конечных устройств. Ключевое достижение 2018 года было реализовано в архитектуре MobileNetv2 [80]. Эта архитектура состоит из 17 слоев, объединенных в блоки с оптимизированными структурами. В 2018–2020 годах прогресс в оптимизации архитектур осуществлялся с помощью методов автоматизированного поиска архитектуры, известных как Neural Architecture Search (NAS) [81, 82]. Кроме того, оптимизация архитектур выполнялась с использованием механизма внимания. Среди реализаций механизма внимания основным стал подход Squeeze and Excitation (SE). Основная идея слоя SE-слоя заключается в перевзвешивании каналов выхода сверточного блока со множителями, вычисленными на основе дополнительных полносвязных слоев [83]. Наиболее успешным результатом использования NAS и SE-слоев стала архитектура EfficientNet [84]. Базовый блок EfficientNet стал базовым для многих современных исследований в области СНС в различных задачах компьютерного зрения. Схема блока EfficientNet показана на рис. 8.

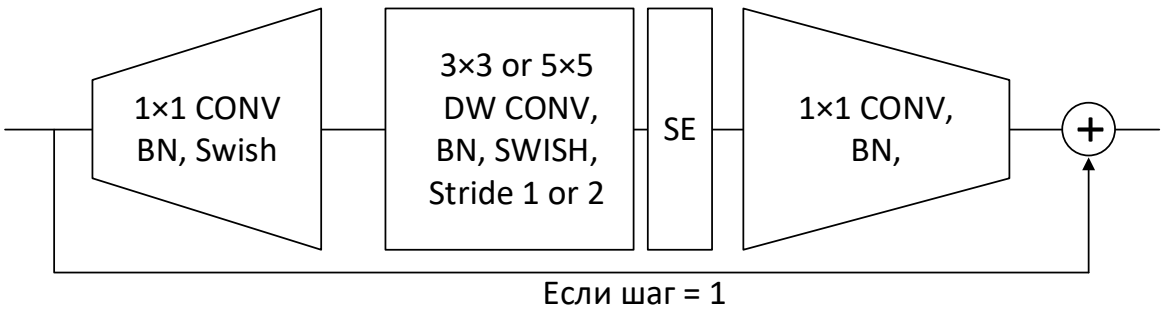


Рис. 8. Схема блочной архитектуры EfficientNet V1

3.2. Архитектуры нейронных сетей: трансформеры

Современные тенденции исследования систем компьютерного зрения включают не только сверточные блоки, но и использование блоков трансформеров [85, 86]. Архитектура блоков трансформеров предполагает только полносвязные слои. Входное изображение для трансформера делиться на сетку небольших участков изображения (patches). Основной блок трансформера состоит из многоголового блока внимания (multi-head self-attention) [87] со слоем нормализации [88] и полносвязного слоя, также с нормализацией. Первым успешным примером реализации сети, построенной только из трансформеров, стала архитектура ViT в 2020 году [85]. Отметим, что ViT имеет весьма много параметров для рассматриваемых приложений и требует большого набора данных для обучения. Замечено, что блоки трансформера легко обучаются «глобальным» признакам изображений, но с трудом обучаются «локальным» и относительно небольшим признакам. Следующие шаги по модификации сети были посвящены поиску решения этой проблемы.

Современные решения включают:

- вычисление информации о внимании только для некоторых окон вместо всех участков изображения; окна могут различаться между слоями (SWIN) [58];
- использование последовательной комбинации сверточных слоев и слоев трансформеров (CoAtNet) [89];
- комбинация в каждом блоке операций свертки и операций блока трансформера (MaxVit) [90];
- поиск наиболее эффективных с вычислительной точки зрения модификаций блока трансформера (MobileVit) [91];
- поиск путей замены механизма внимания на другую нелинейную операцию для глобального извлечения признаков в блоке подобном трансформеру (MLP-Mixer) [92];
- использование мета-обучения, в т. ч. «дистилляции знаний» для конкретных задач; DeiT [93]; улучшение качества обучения целевых моделей путем подбора меток с помощью дополнительных моделей [94]; использование кросс-дата аугментации (аугментация меток и данных) [95].

Отметим, что лучшие на сегодня архитектуры-трансформеры имеют более одного миллиарда параметров и проходят предварительное обучение на сверхбольших наборах данных, имеющих порядка сотен миллионов изображний [96]. Иллюстрация архитектуры VIT показана на рис. 9.

3.3. Решающая подсеть

Как показано в обзоре, в настоящее время основными подходами к использованию компьютерного зрения в горнодобывающей промышленности являются архитекура семантической сегменатции U-Net [33] и архитектура экземплярной сегментации Mask R-CNN [41]. Эти архитектуры являются наиболее популярными в соответствующих областях и могут рассматриваться как базовые модели. Обе сети содержат кодировщики признаков, основанные в основном на базовом блоке ResNet и специфичные для задач головные подсети. Важно отметить, что в исходном виде эти архитектуры не обеспечивают высокой производительности и точности по сравнению с более современными подходами (см. например [97–99], а также сравнение современного состояния на веб-сайте `https://paperswithcode.com/sota`).

В работе [19] отмечается, что среди современных достижений в области быстрых методов решения задач экземплярной сегментации (в реальном времени) выделяются такие ар-

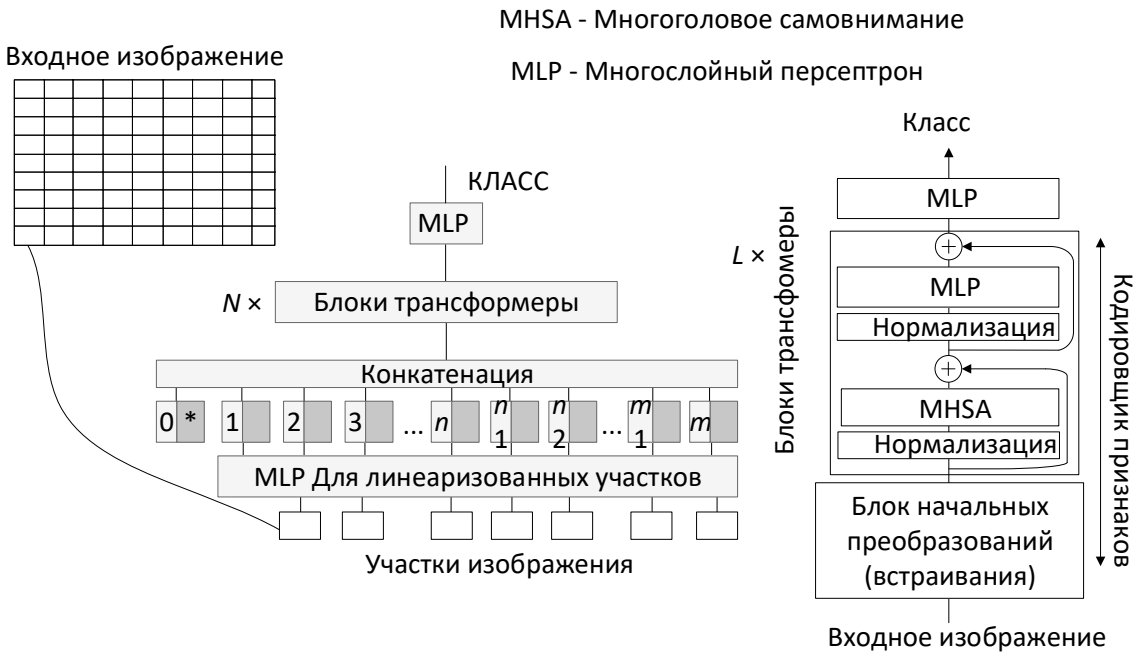


Рис. 9. Иллюстрация архитектуры ViT

хитектуры, как YOLACT [97] и ее модификации, такие как YOLACT++ [100], SOLOv2 [101] и SparseInst [98]. Отметим, что данные методы используют подход на основе поиска центральных позиций объектов и их сегментации, в отличие от сегментации на основе поиска регионов (как в Mask R-CNN). Модель обучается различать пиксели, принадлежащие либо одному и тому же, либо разным объектам. Например, подход YOLACT генерирует маски-прототипы по всему изображению, предсказывая наборы параметорв для каждого экземпляра. Окончательные маски строятся после процедуры NMS. Архитектура YOLACT++ улучшает результаты базовых моделей за счет модификаций кодировщика признаков и добавления т. н. деформируемых сверток к головной подсети. Подход SOLO [102] основан на предположении, что экземпляры можно разделить по их центральному положению и размерам. Положения центров вычисляются путем разделения изображения на ячейки и вычисления положения центра внутри каждой из них. Размеры экземпляров определяются на основе признаков кодировщика с т. н. пирамидальной структурой. Архитектура SOLO v2 улучшает предыдущие результаты, генерируя динамически весовые параметры для каждого потенциального экземпляра. Эта концепция называется динамическим ядром. Для каждого экземпляра генерируется своя маска (размером с изображение). Подход SparseInst аналогичен SOLOv2, но имеет две головных подсети, работающие с такими масками. Первая используется для создания масок с учетом классов, а вторая — для оценок положения и размеров объектов в каждой маске. Маски умножаются на предсказанные маски для создания масок сегментации.

Среди достойных внимания методов быстрого обнаружения объектов следует отметить архитектуры YOLOv5–YOLOv8. Эти архитектуры можно использовать как в режиме обнаружения объектов, так и в режиме экземплярной сегментации. Также в некоторых случаях в данных архитектурах выделяют отдельный подход — ориентированный поиск объектов, когда выделяют не только ограничивающие рамки для объектов, но угол наклона рамки (такой, чтобы рамка имела наименьшую площадь) [46, 99, 103–106]. Указанные архитекту-

ры наследуют единый подход YOLO, в котором предполагается, что в входное изображение делится на одинаковые ячейки. Каждый объект определяется как принадлежащий одной из этих ячеек. Каждая ячейка соответствует одному вектору на выходе модели. Вектор может включать в себя один или несколько наборов с предсказаниями размеров, класса и координат объекта. Результат объектов определяется с помощью алгоритма NMS [107]. Семейство архитектур YOLO разрабатывается с использованием большого количества экспериментов с архитектурой и методами обучения, которые называются «Bag of Specials» и «Bag of Fribies», соответственно [105, 106, 108]. В основе этих экспериментов лежит подход архитектуры YOLOv4, иллюстрация которой показана на рис. 10.

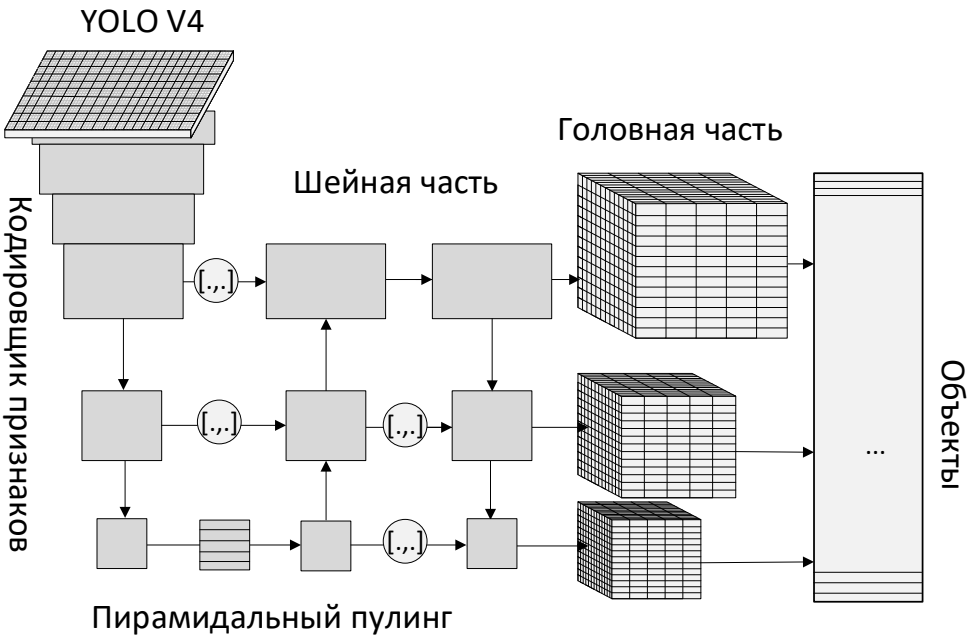


Рис. 10. Иллюстрация архитектуры YOLOv4

3.4. Тренды компьютерного зрения

Последние тенденции в компьютерном зрении можно обобщить следующим образом.

Оптимизация операции свертки. Например, для EfficientNet V2 [109] было замечено, что т. н. глубокая разделенная свертка (DepthWise Separable Convolution) становится вычислительно эффективной только для больших слоев, несмотря на меньшее количество параметров. Таким образом, в начальных блоках было предложено заменить ее на обычную свертку.

Оптимизация операции нормализации. Пакетная нормализация (BatchNorm) может снизить точность в случае предварительного обучения на большом наборе данных, при изменении размера пакета и в других случаях. Одним из способов решения этой проблемы является использование альтернатвы — послойной нормализации (LayerNorm) [88]. Другой способ предполагает замену пакетной нормализации набором отдельных операций. Этот прием был реализован в NFNet [110]. Как правило замена пакетной нормализации сопровождается дополнительными операциями.

Автоматический поиск наиболее подходящей архитектуры. Подход поиска архитектуры нейросети (NAS) имеет множество преимуществ и недостатков. Основным недостатком

является зависимость пространства поиска от опыта исследователя. Также отбор архитектур из пространства поиска ограничен доступными вычислительными ресурсами и опытом (знаниями) исследователей, которые занимаются этой работой [81, 82].

Компромисс между блоками трансформерами, полносвязными и сверточными слоями. Как описано выше, этот вопрос остается открытым. Основными проблемами являются поиск наименьшей архитектуры (с точки зрения параметров), избежание необходимости работы с большими наборами данных для предварительного обучения и ускорение процедуры обучения.

Необходимость оптимизации головных подсетей для решения конкретных задач. Как показано выше, выбор архитектуры головной подсети позволяет достичь определенного компромисса между точностью и скоростью работы. Наиболее производительными подходами на текущий момент являются архитектуры семейства YOLO в задачах обнаружения объекта и архитектуры SOLO в задачах экземплярной сегментации [55, 111]. Наиболее точными архитектурами для решения соотвествующих классов задач являются архитектуры типа Cascade-Mask-R-CNN и Hybrid task cascade for instance segmentation (HTC) [112].

Сравнение современных кодировщиков признаков на некоторых репрезентативных тестовых наборах можно найти в работе [96].

Основное обобщение ключевых публикаций представлено в табл. 2 и 3.

3.5. Архитектуры компьютерного зрения для фрагментации горных пород

В результате рассмотренных тенденций можно сделать следующий вывод относительно компромисса между ростом точности и низкой вычислительной производительностью. В компьютерном зрении современные архитектуры CHC используют как блоки свертки, так и трансформеры в виде ResNet-подобных блоков. Сверточный слой может быть реализован как в традиционном виде (для начальных слоев), так и в виде DepthWise-сверток для остальных блоков. Метод свертки DepthWise имеет преимущества в количестве обучаемых параметров и, вероятно, обладает большей потенциальной репрезентативной способностью. Блоки трансформеры в целом являются ресурсоемкими, но некоторые модификации, такие как MobileVit или SWIN, оптимизированы для достижения наилучшего компромисса между точностью и скоростью работы.

Наиболее эффективная комбинация блоков кодировщика признаков для конкретной задачи может быть определена только экспериментально. Тем не менее, в задачах фрагментации горных пород архитектуры Mask R-CNN и U-Net могут быть использованы в качестве базовых решений для соответствующих задач. Подходы, основанные на архитектур SOLO и YOLO, можно рассматривать как основу для перспективных подходов для планируемых исследований.

При выборе архитектур нейронных сетей важно учитывать специфику подходов к переносу обучения. В частности, требуются модификации операций пакетной нормализации (BatchNorm) и работы с большими наборами данных на стадии предварительного обучения. Также итоговая модель может быть обучена с использованием таких техник, как дистилляция знаний.

Таблица 2. Тренды компьютерного зрения: кодировщики признаков

Ссылка	Архитектура	Комментарии
[60]	LeNet	Первая сверточная нейронная сеть; SGD с импульсом; инициализация весов;
[63]	AlexNet	Первая из эпохи современного глубокого обучения сверточная нейронная сеть; аугментация данных; дропаут
[65]	VGGNet	Каскадная сверточная сеть
[71]		Адаптивная оптимизация статистических моментов низшего порядка на основе градиента (ADAM)
[67]	NIN	Концепция блока; точечная свертка; глобальное среднее объединение
[69]	Inception	Блок с изменением рецептивного поля
[70]		Пакетная нормализация
[66]	Inception V3	Пространственно разделимая свертка
[42]	ResNet	Остаточная связь; блок «узкого горлышка»
[75]	ResNeXt	Многоканальный блок «узкого горлышка»
[68]	Xception	DepthWise глубокая разделяемая свертка
[76]	DenseNet	Блок с несколькими тождественными связми
[80]	MobileNet V2	MobileNet архитектура; Обратные остаточные блоки с пространственно разделяемыми свертками
[83]	SE Net	Слой сжатия и возбуждения (Squeeze and Excitation layer)
[84, 109]	EfficientNet	Оптимизированный блок, полученный с помощью NAS
[77]	ResNet-D	Улучшенный ResNet
[78]	BiT	Эффективный перенос обучения
[79]	RegNet	Оптимизация блока ResNet
[110]	NFNet	Альтернатива пакетной нормализации
[85]	ViT	Первая эффективная архитектура визуального трансформера
[93]	DeiT	Дистилляция знаний, алгоритм мета-обучения для архитектур-трансформеров
[58]	SWIN	Вычисление информации о внимании только для некоторых окон, а не для всех
[89]	CoAtNet	Комбинация сверточного блока и блока-трансформера
[91]	MobileVit	Эффективный мобильный трансформер
[92]	MLP-Mixer	Блок типа трансформер без внимания

Таблица 3. Тренды компьютерного зрения: решающие подсети

Ссылка	Архитектура	Комментарии
[41]	Mask R-CNN	Базовая архитектура для экземплярной сегментации
[97, 100]	YOLACT, ++	Архитектура экземплярной сегментации для низкопроизводительных устройств
[101, 102]	SOLO V1,V2	Архитектура экземплярной сегментация и динамическая свертка
[98]	SparseInst	Архитектура быстрой сегментации в режиме реального времени с улучшением результатов SOLO
[33]	U-Net	Базовая архитектура семантической сегментации
[113]	DeepLab v3+	Современная архитектура семантической сегментации
[108]	YOLOv4	Архитектура YOLO 4
[46, 99, 103, 104]	YOLOv5–v8	Современные архитектуры YOLO
[81, 82]		Обзор метода поиска архитектуры нейронной сети (NAS)
[64]		Исторический обзор архитектур
[85, 86]		Обзор архитектур трансформеров
[94]		Мета-обучение
[95]		Аугментация данных и кросс-дата аугментация
[105, 106]		Обзор архитектур решения задач обнаружения объектов в реальном времени
[96]		Результаты для бенчмарка ImageNet
[55]		Результаты тестов для экземплярной сегментации в реальном времени
[111]		Результаты тестов для задачи обнаружения объектов в реальном времени

Актуальны и другие вопросы: ускорение времени обучения и работы СНС, а также снижение требований к объему памяти для модели. Особенно это важно для низкопроизводительных систем. В следующем разделе обсуждаются параллельные архитектуры и методы оптимизации для эффективной реализации нейронных сетей.

4. Реализация на параллельных архитектурах

4.1. Специфика центральных процессоров

Центральные процессоры (CPU) можно рассматривать как многоядерные и многопоточные архитектуры, оптимизированные для векторных и матричных операций. Например, настольные процессоры могут иметь до 64 ядер и 128 потоков (с точки зрения аппаратной многопоточности) и до 72 ядер и 288 потоков в серверных процессорах. Такие процессоры

можно использовать как основу для обучения и работы нейронных сетей. Преимущества данного подхода заключаются в следующем:

- отсутствие необходимости в дополнительном оборудовании;
- простота работы с любыми фреймворками (большинство современных фреймворков поддерживает архитектуру x86-64);
- наличие дополнительных инструментов оптимизации производительности (OpenVINO [114], NNPack [115], oneDNN [116], CcT [117]), а также инструментов для оптимизации и распараллеливания нейронных сетей, например, Intel BigDL [118];
- оптимизированные наборы SIMD-инструкций (AMX, AVX-512VNNI и другие), поддерживающие форматы с ограниченной точностью BF16 и INT8 [119];
- возможность использования не только для машинного обучения, но и для вспомогательных задач.

Одним из передовых направлений в развитии процессоров являются гетерогенные архитектуры, включающие как CPU, так и GPU, FPGA и другие типы ускорителей, использующие унифицированную кэш-память и SoC. Примером такого подхода является ускорители нейронных сетей Intel на базе процессора Miryard. Ожидается, что такой подход позволит достичь максимальной эффективности при низкой стоимости конечного устройства и ограниченном энергопотреблении.

4.2. Специфика графических процессоров

Графические процессоры (GPU) — это тип вычислительных устройств, построенных по принципу матричных архитектур с синхронным параллелизмом на уровне инструкций (так называемые SIMT-архитектуры, «одиночный поток команд, множественный поток данных»). Этот тип архитектуры ориентирован на обработку больших массивов регулярных данных (включая изображения). Первоначально задачами, для которых разрабатывались такие процессоры, были специальные операции обработки изображений, которые выполнялись на аппаратном уровне. Большинство таких операций включает умножение матриц и реализуются с помощью операций сложения и умножения с плавающей точкой (Fused Multiply-Add, FMA). Поэтому ряд современных графических процессоров специализируется именно на этом типе операций. Такие ускорители называются General Purpose Graphics Processor Unit (GPGPU) [120].

В настоящее время GPGPU представляют собой многопроцессорные системы, состоящие из набора потоковых мультипроцессоров (SMT). Мультипроцессоры также могут быть объединены в группы, кластеры (или срезы). Каждый мультипроцессор SIMT выполняет блок (или поток) инструкций SIMD с помощью набора функциональных модулей.

Графические ускорители стали популярным инструментом для обучения и выполнения алгоритмов машинного обучения. В первую очередь это связано с тем, что алгоритмы машинного обучения (включая глубокие нейронные сети) хорошо поддаются распараллеливанию. Более того, алгоритмы обучения и выполнения нейронных сетей включают набор последовательных операций, которые выполняются одна за другой над одним набором данных. Большинство вычислительных операций нейронной сети сводится к FMA. Среди преимуществ графических ускорителей в таких задачах можно выделить способность к распараллеливанию и обеспечение высокой вычислительной мощности (десятки TFLOPs в формате FP32 и сотни TFLOPs в формате FP16 с использованием тензорных ядер), а

также продвинутые технологии для организации GPU-серверов. Для процесса обучения нейронных сетей графические ускорители, как правило, обеспечивают наилучшую производительность по отношению к стоимости. Кроме того, GPU часто являются единственным оборудованием, которое способно масштабироваться с вычислительными требованиями определенных алгоритмов и моделей глубокого обучения.

В настоящее время наиболее популярными в задачах машинного обучения являются графические ускорители компании NVIDIA [121]. Такая популярность обусловлена как поддержкой со стороны NVIDIA, так и использованием технологии CUDA большинством фреймворков для обучения нейронных сетей. Использование в ускорителях технологии разреженных матриц и форматов INT8/INT4 позволяет ускорить выполнение до 10 раз. Реальная производительность графических ускорителей, как правило, ниже пиковой и зависит от оптимизации кода. Такая оптимизация в ручном режиме часто игнорируется, так как является очень трудоемкой. Возможно использование автоматической оптимизации с помощью утилит, таких как NVIDIA TensorRT [122].

4.2.1. Использование пониженной точности на GPU

Одной из ключевых проблем при реализации нейронных сетей является большой объем оперативной памяти, необходимый для хранения модели. В настоящее время наиболее популярными методами сжатия моделей являются квантование с использованием разреженности весов и огрубление функций активации. Квантование может быть линейным или нелинейным [123]. Одной из основных тенденций в нейронных сетях и других алгоритмах машинного обучения является использование арифметических операций с пониженной и смешанной точностью. Операции со смешанной точностью предполагают, что хотя бы один из операндов имеет пониженную точность. Примерами таких форматов являются TensorFloat-32 (TF32) [124], Brain Float (BF32 и BF16) [125] и другие.

Одним из частных случаев является так называемая бинаризованная нейронная сеть (BNN) [126]. Ее преимущество заключается в максимальном использовании пропускной способности памяти и минимальном размере модели [127, 128]. Также интерес представляют архитектуры, основанные на формате INT2 $\{-1, 0, 1\}$ — они относятся к классу Ternary Neural Networks (TNN) [129]. Главная особенность TNN заключается в автоматическом округлении малых значений весов до нуля во время обучения. При этом потеря точности может быть снижена до 1% по сравнению с традиционным форматом FP32 [130].

Ряд современных работ показывают, что наиболее перспективным способом является обучение сети в форматах с плавающей запятой с последующей тонкой настройкой или переобучением в целочисленных форматах. Цель такого подхода — уменьшить размер обученной модели и времени выполнения при минимальной потери точности [131]. Следует отметить, что в этом случае нет необходимости оптимизировать все слои нейронной сети. Можно использовать различные степени квантования для параметров нейронной сети, чтобы уменьшить потери в точности. Верхние слои глубоких нейронных сетей, а также входной слой, требуют большего диапазона весов и значений функций активации, чем нижние слои. Также отмечается, что при квантовании требуется процесс переобучения (или дообучения) нейронной сети [132]. Например, в сверточных нейронных сетях наибольшая вычислительная нагрузка приходится на многомерные свертки в верхних слоях кодировщика признаков. Таким образом, они должны быть приоритетными для оптимизации [133].

Современные GPGPU позволяют выполнять операции FMA с несколькими вариантами глубины квантования [120]. В частности, поддерживаются операции со смешанной точностью, например, операции FMA вида $D = A \cdot B + C$, где операнды A и B имеют половинную точности [134]. Использование операндов с пониженной точностью уменьшает объем ОЗУ, занимаемый параметрами нейронной сети.

Для ускорения обработки вычислений в глубоких нейронных сетях в ряде архитектур графических ускорителей используются так называемые модули тензорных вычислений. В зависимости от архитектуры GPU, эти модули могут отличаться как по количеству, так и по поддерживаемой ими функционалу. Например, в процессорах архитектуры NVIDIA Ampere [135] в каждом блоке используется тензорное ядро, поддерживающее операции смешанной точности FP16/FP32, FP16, BF16, TF32, FP64, INT8, INT4, бинарные операции, а также операции над разреженными матрицами в соотношении 2 нуля на 4 параметра. Более того, каждый тензорный модуль представляет собой массив, например, $4 \times 4 \times 4$, предназначенный для выполнения операций FMA $D = A \cdot B + C$ в цикле, где A, B, C и D — матрицы размера 4×4 [134].

Современные фреймворки для нейронных сетей предоставляют инструменты для автоматического и адаптивного применения методов смешанной точности [136–138]. Они облегчают задачу квантования модели и балансировки ее производительности, эффективности и точности.

4.2.2. Использование разреженных матриц на GPU

На практике обученная нейронная сеть имеет большинство весовых коэффициентов, равных или близких к нулю. Такие веса могут быть удалены или обрезаны [139, 140]. Отмечается, что в сверточных сетях разреженность может достигать 90% без существенных потерь в точности. Также отмечается, что в TNN разряженность может достигать до 50% без значительных потерь в точности. В экспериментах [130] авторы получили 1% потери точности при разреженности 65% для архитектуры GoogLeNet. Также отметим, что в современных графических ускорителях используются технологии оптимизации вычислений операций над разреженными матрицами [141]. Эта техника сокращает время работы нейронных сетей.

Отметим, что разреженность нейронных сетей может достигаться автоматически в ходе обучения модели. Это связано с частым использованием полулинейного модуля (ReLU) в качестве функции активации в CHC. Это может ускорить процедуру обучения нейронной сети. Например, в архитектуре NVIDIA Ampere [135], используется предположение, что операнды инструкций для разреженных матриц — это матрицы, где есть два ненулевых значения в каждом векторе с четырьмя входными значениями (разреженность строк 2:4). Благодаря такой структуре матрицы ее можно сжимать, уменьшая требуемый объем памяти и пропускную способность почти вдвое (разреженное тензорное ядро).

4.2.3. Оптимизация доступа к памяти

Скорость доступа к памяти может ограничивать производительность нейронных сетей. Обычно для оценки производительности системы в отношении вычислительной производительности и пропускной способности памяти используется модель Roofline [142]. Эта модель включает предельные технические характеристики системы и может быть использована

для оценки производительности реальной программы. Анализ модели Roofline показывает, что для достижения более высокой производительности следует применять одновременно несколько методов оптимизации. Повышение производительности может быть достигнуто, например, за счет оптимизации вычислительных модулей, использования адаптированных форматов данных и оптимизации вычислительных алгоритмов. Также значительное повышение производительности может быть получено за счет оптимизации циклов. Для этого могут применяться различные методы, такие как переупорядочивание циклов, конвейеризация циклов, разворачивание циклов и другие. Переупорядочивание циклов уменьшает количество обращений к памяти между итерациями цикла. При разворачивании цикла несколько итераций выполняются одновременно параллельно. Метод конвейеризации циклов выполняет итерации цикла с перекрытием таким образом, что следующая итерация начинается до завершения предыдущей. Оптимальное количество не перекрывающихся итераций может варьироваться в зависимости от алгоритма вычислений. Этот параметр также влияет на количество обращений к памяти и должен зависеть от архитектуры аппаратного обеспечения. Оптимизация цикла может быть выполнена путем использования двойных буферов для хранения результатов [143]. В некоторых работах [144] предлагается разворачивание ядра для CHC. Такая оптимизация подразумевает замену одного сверточного ядра, например, размера 5×5, несколькими ядрами размером 3×3. Оптимизация цикла также может быть достигнута за счет использования регулярной структуры нейронных сетей [133].

В дополнение к стратегиям, описанным выше, для оптимизации циклов могут использоваться методы более высокого уровня. К таким методам относятся блокирование (тайлинг) циклов и чередование циклов. Блокирование циклов — это разбиение циклов на более мелкие компоненты, блоки (тайлы). Все входные данные для одного блока хранятся в специальном буфере или кэш-памяти. Цикл выполняется с каждым блоком по очереди. Во время чередования циклов компоненты цикла переставляются таким образом, чтобы соседние компоненты использовали одни и те же данные. Таким образом, не нужно перезагружать данные из памяти для отдельных компонентов. Эти методы позволяют контролировать размер и порядок блоков. Таким образом, можно оптимизировать циклы для различных слоев нейронной сети, например, CHC [133].

Для оптимизации операции свертки [145, 146], можно использовать несколько типов алгоритмов: пространственная свертка (CONV), векторизованная свертка (например, im2col), быстрая свертка Винограда (Winograd) и частотная свертка. Алгоритм Winograd является самым быстрым, но при этом он требует отдельной реализации для различных размеров сверточных ядер, что не всегда выполнимо. Частотная свертка имеет умеренную скорость, но может быть реализована только для размеров ядра, равных степеням двойки. Пространственная свертка является самой медленной, но она имеет самые низкие требования к объему кэш-памяти и пропускной способности шины доступа к памяти.

4.3. Использование нескольких графических процессоров

Достичь высокого параллелизма нейронных сетей можно путем повышения эффективности вычислительных операций и распределение вычисления всей сети по нескольким измерениям. Например, разбив данные на мини-пакеты послойно или тензорно, можно разделить прямой проход и обратное распространение между параллельными процессорами. Существует несколько основных стратегий распределения рабочей нагрузки [147]:

- распределение входных данных — параллелизм данных;
- распределение структуры нейросети — параллелизм модели (тензорная нарезка);
- конвейерная обработка — параллелизм модели (послойный);
- комбинированная стратегия — гибридный параллелизм.

4.3.1. Параллелизм данных

Среди методов обучения нейронной сети популярным является стохастический градиентный спуск (Stochastic Gradient Descent, SGD). При его применении количество обновлений весов уменьшается за счет вычисления потерь выборки в мини-пакетах, а затем усреднения градиента по мини-пакетам [148]. Данные обрабатываются за N выборок. Мини-пакетные методы используются как компромисс между традиционным SGD, который использует всю выборку за один раз, и пакетными методами, использующими весь набор данных на каждой итерации. В [149] доказано, что SGD с мини-пакетами обеспечивает сходимость, аналогичную традиционному SGD. Самый простой подход к распараллеливанию — разделить обработку мини-пакетов на несколько вычислительных устройств, поскольку большинство операций не зависят от количества элементов мини-пакета. Этот подход известен как шаблонный параллелизм (Pattern Parallelism) [150].

Параллелизм данных имеет основной недостаток, связанный с чрезмерными коммуникационными затратами между вычислительными узлами, поскольку веса должны быть синхронизированы между каждым вычислительным узлом. Такие коммуникационные затраты увеличиваются с ростом размера модели, что значительно затрудняет масштабируемость параллелизма данных [150].

Большинство фреймворков глубокого обучения сегодня поддерживают параллелизм данных для одного GPU, нескольких GPU или кластера из нескольких GPU. Масштаб параллелизма данных естественным образом определяется размером мини-пакета. За исключением пакетной нормализации, которая обычно применяется между слоями и функциями активации, все операции обрабатывают одну выборку за один раз. Это означает, что процедуры прямого прохождения через СНС и обратного распространения ошибки почти полностью независимы. На этапе обновления весов распределенные результаты усредняются, чтобы получить градиент для всего пакета данных. Поскольку все параметры глубокой нейронной сети должны быть доступны всем устройствам, они дублируются.

Существует несколько факторов, которые влияют на производительность синхронного SGD с параллелизмом данных. Чтобы сохранить точность, размеры мини-пакетов обычно ограничены. Если мы увеличиваем размер выше определенного порога, то мы теряем точность [151]. В то же время при масштабировании системы до нескольких рабочих узлов в кластере размер обучающей партии увеличивается линейно. Таким образом, в обучающей системе с большим количеством узлов размер набора данных для каждого узла системы должен быть небольшим. Малый размер выборки данных на вычислительном узле способствует сокращению времени вычислений и высоким затратам на синхронизацию, что делает алгоритмы передачи данных важным фактором производительности в больших обучающих системах [152].

Применяя различные модификации [153, 154] в процессе обучения, можно увеличить размер мини-пакета без значительной потери точности. Хотя проблема обобщающей способности сети все еще существует, она не настолько серьезна, как это было заявлено в предыдущих работах. Узким местом (ограничивающим масштабируемость) является опера-

ция пакетной нормализации, для выполнения которой требуется синхронизация. Поскольку пакетная нормализация применяется многократно в некоторых архитектурах глубоких нейронных сетей, она становится слишком затратной. Для смягчения этой проблемы популярные подходы к пакетной нормализации используют небольшие подмножества из мини-пакета (например, 32 изображения) для независимой нормализации. Если для каждого процессора запланировано хотя бы 32 изображения, то синхронизация становится локальной, что, в свою очередь, увеличивает потенциал масштабирования.

В работе [155] предлагается использовать весовую нормализацию для отделения нормы параметра от его направления путем перепараметризации. Это уменьшает вычислительную сложность с $O(\log N)$ до $O(1)$ за счет устранения внутренних зависимостей внутри мини-пакета. По мнению авторов этого метода, весовая нормализация снижает необходимость пакетной нормализации и обеспечивает сопоставимую точность при использовании упрощенной версии пакетной нормализации без коррекции дисперсии.

В литературе были предложены и другие подходы к параллелизму данных. Например, в ParallelSGD [156] алгоритм SGD выполняется с помощью мини-пакетов, несколько раз параллельно, распределяя набор данных между процессорами. После того, как все экземпляры SGD сходятся, полученные веса объединяются и усредняются, что обеспечивает параллелизм данных.

ParallelSGD был разработан с использованием парадигмы программирования MapReduce [157, 158]. Используя MapReduce, можно планировать параллельные задачи для выполнения на мультипроцессорах, распределенных средах и нескольких GPU. Ранее потенциал масштабирования MapReduce был протестирован в различных задачах машинного обучения, включая нейронные сети [159–161], что способствовало необходимости перехода от однопроцессорного обучения к системам с распределенной памятью. Отметим, что его высокая универсальность затрудняет создание реализаций, специально предназначенных для нейронных сетей глубокого обучения.

Современные реализации используют высокопроизводительные коммуникационные интерфейсы (такие как MPI) [162] для достижения тонкого параллелизма. Это уменьшает задержку за счет асинхронного выполнения кода, конвейеризации, непостоянных коммуникаций и использования параллелизма внутри вычислительных устройств [163–166]. В последнем случае мини-пакеты разбиваются на микро-пакеты, которые распределяются для параллельной или последовательной обработки. Это уменьшает объем памяти, позволяя использовать более быстрые методы.

Фреймворк PyTorch реализует автоматический параллелизм данных [167] путем разделения входных данных по указанным устройствам с помощью разбиения пакета на части. Модуль реплицируется на каждой машине и каждом устройстве, и каждая такая реплика обрабатывает часть входных данных. Во время обратного прохода градиенты от каждого узла усредняются.

4.3.2. Параллелизм модели

Второй стратегией, используемой для распараллеливания глубоких нейронных сетей, является параллелизм модели, также известный как параллелизм сети [168, 169].

Эта стратегия представляет собой распределение частей глубокой нейронной сети между компьютерами. Каждое вычислительное устройство выполняет обновление весов внутри назначенного ему подмножества модели. В результате интенсивность обмена данными зна-

чительно снижается по сравнению с параллелизмом данных. Более того, такой подход позволяет обучать более крупные модели и обходить ограничения на размер, накладываемые памятью, доступной одному процессору [149].

Здесь вычисления распределяются по нейронам в каждом слое или по измерениям в четырехмерном тензоре. В этом случае мини-пакет копируется на все процессоры, и разные части глубокой нейронной сети вычисляются на разных процессорах. Это снижает потребление памяти (поскольку вся сеть больше не хранится в одном месте), но добавляет дополнительные коммуникации между уровнями.

Поскольку при этом размер мини-пакетов не меняется, то компромисс между использованием данных и коммуникациями отсутствует. Архитектура глубоких нейронных сетей создает внутренние зависимости между слоями, которые, в свою очередь, генерируют коммуникации, определяющие общую производительность. Например, полносвязные слои подразумевают коммуникации «все со всеми» (в отличие от стратегии параллелизма данных, которая позволяет избежать этих коммуникаций), поскольку нейроны одного слоя связаны со всеми нейронами следующего слоя.

Метод сокращения числа коммуникаций между полносвязными слоями заключается в использовании алгоритма матричного умножения Кэннона, модифицированного для использования в нейронных сетях [170]. Сообщается, что такой алгоритм обеспечивает более высокую эффективность и скорость по сравнению с простым разделением на многослойные сети меньшего масштаба.

Для CHC применение модельного параллелизма имеет ограниченную эффективность [171]. Если выборки разделены между процессорами по частям (или каналам), то для вычисления результата каждой свертки придется получать результаты от всех остальных процессоров, поскольку эта операция суммирует результаты. Для частичного решения этой проблемы были предложены локально связанные сети (Locally Connected Network, LCN) [172–175]. Они также выполняют свертки, но для каждой области они применяют несколько локальных фильтров, обеспечивающих разделение по измерениям, устраняя необходимость в коммуникациях «все со всеми».

Использование локально связанных сетей дает возможность запустить CHC на 5000 узлах CPU с трехузловом кластере с несколькими GPU [176]. Отсутствие обмена весами приводит к тому, что обучение не зависит от коммуникаций, что дает значительный потенциал для масштабирования. Успешное применение тех же методов в CHC требует тонкого контроля параллелизма. Обмен весами является неотъемлемой частью работы CHC. Он помогает уменьшить объем памяти и улучшить коммуникации. Таким образом, стандартные операции свертки используются более широко, чем локально связанные сети.

4.3.3. Конвейеризация

Под конвейеризацией глубокого обучения можно понимать либо перекрытие вычислений между различными слоями по мере поступления данных, либо разделение глубокой нейронной сети (Deep Neural Network, DNN) по глубине с закреплением уровней за конкретными процессорами. Конвейер можно рассматривать как форму параллелизма данных (элементы или выборки обрабатываются параллельно), а также как параллелизм модели (длина конвейера определяется структурой DNN).

Первая форма конвейеризации может быть использована для перекрытия этапов прямого прохождения, обратного распространения и обновления весов [177, 178]. Эта схема

широко используется на практике и повышает эффективность использования ресурсов за счет сокращения времени простоя процессора. На более высоком уровне детализации архитектуры нейронных сетей могут быть разработаны по принципу вычислений перекрывающихся слоев, как это происходит в сетях с глубоким суммированием (Deep Stacking Network, DSN) [179]. В DSN каждый полносвязный слой вычисляется на отдельном шаге. В конечном итоге результаты всех предыдущих шагов передаются для входа следующего слоя. Благодаря свободной зависимости данных, этот метод позволяет частично вычислять каждый слой параллельно.

Применительно к послойному разделению [180, 181], многопроцессорный конвейер имеет ряд преимуществ как перед параллелизмом данных, так и перед параллелизмом моделей. Во-первых, нет необходимости хранить все параметры на всех процессорах во время прямого прохода и обратного распространения ошибки (как в случае с модельным параллелизмом). Во-вторых, существует фиксированное число конечных точек связи между процессорами на границах уровней, поскольку исходный и целевой процессоры всегда известны. Более того, поскольку процессоры всегда вычисляют одни и те же слои, веса можно хранить в кэше, чтобы уменьшить накладные расходы на доступ к памяти. Недостатки конвейеризации заключаются в том, что данные (выборки) должны поступать с определенной скоростью для полного использования системы, а задержка пропорциональна количеству процессоров.

При простой реализации модельного параллелизма одновременно активен только один GPU. PipeDream [182] предлагает решение проблемы с помощью конвейеризации, но конвейерный модельный параллелизм вводит проблемы стабильности и согласованности обновления весов. Поскольку в конвейере одновременно обрабатывается несколько мини-пакетов, последующий мини-пакет может начать обучение до того, как предыдущая обновит веса. Проблема застоя/согласованности приводит к нестабильному обучению и снижает точность модели. Gpipe [183] предлагает метод конвейера, отличный от PipeDream, чтобы полностью избежать этих проблем. Он разбивает каждый мини-пакет на несколько микро-пакетов, а затем конвейеризует выполнение каждого набора микро-пакетов по ячейкам. Веса обновляются синхронно для всех накопленных градиентов после завершения обработки мини-пакета. Несмотря на то, что этот подход может увеличить использование GPU по сравнению с неконвейерным параллелизмом моделей, он снижает пропускную способность GPU. Для ее повышения был предложен механизм SpecTrain, который предсказывает будущие веса на ранних стадиях конвейера [171].

4.3.4. Комбинированный параллелизм

Комбинация нескольких стратегий параллелизма может преодолеть недостатки каждой стратегии. Существует несколько успешных подходов к реализации гибридных методов.

Например, AlexNet [64, 184] — сверточная нейронная сеть, где большинство вычислений выполняется в сверточных слоях, но большая часть параметров принадлежит полносвязным слоям. В работе [185] авторы использовали несколько стратегий параллелизма и коммуникационных оптимизаций. Для обучения сети они добились ускорения в 400 раз, используя 512 графических процессоров.

AMPNet [186] — асинхронная реализация обучения DNN на процессорах, которая использует промежуточное представление для реализации параллелизма модели. В частности, параллельные подзадачи внутри и между слоями формируются и планируются асин-

хронно. Кроме того, асинхронное выполнение динамического потока управления позволяет конвейерно выполнять прямой проход, обратное распространение и обновление весов. Основным преимуществом AMPNet является древовидная рекуррентная нейронная сеть, которая использует переменную длину выборки и динамический поток управления (в отличие от однородных СНС).

Наконец, распределенная система глубокого обучения DistBelief [187] сочетает в себе все три стратегии параллелизма. В этой реализации обучение проводится одновременно на нескольких экземплярах модели, причем каждый экземпляр обучается на разных выборках (параллелизм данных). Внутри каждого экземпляра DNN происходит распределение как по нейронам в одном слое (параллелизм модели), так и по разным слоям (конвейеризация). Таким образом, в DistBelief конвейеризация не ограничивается различными ядрами CPU на одном узле.

4.3.5. Особенности параллельных сверточных нейронных сетей

В этом разделе мы рассмотрим различные подходы к распараллеливанию СНС. Обучение глубоких СНС на больших наборах данных затратно и требует много времени.

Архитектура сверточных нейронных сетей позволяет сократить время обучения сети, поскольку первые четыре слоя не является полностью связанными между собой. Эта часть сети может быть распараллелена, что позволяет сократить время обучения. Существует множество методик распараллеливания алгоритмов нейронных сетей, например, распараллеливание операций на нижних слоях, распределение их по разным прямым и обратным потокам и т.д. Для выбора оптимального подхода необходимо учитывать характеристики сети и особенности архитектуры компьютера. Разработка параллельных алгоритмов для нейронных сетей осуществляется при наличии как минимум двух физических процессоров.

Существуют две проблемы при разработке масштабируемых параллельных сверточных нейронных сетей в вычислительных средах с распределенной памятью. Одна из них — высокая степень зависимости данных, которая проявляется при обновлении параметров модели между двумя соседними мини-пакетами, а другая — большой объем данных, которые необходимо передавать по каналам связи. Для их решения используются различные стратегии, включая перекрытие межпроцессных взаимодействий [188].

В работе [188] предложен подход для совмещения передачи данных и вычислений. В алгоритме обратного распространения ошибки градиент функции стоимости вычисляются над всеми параметрами сети. Поскольку нет зависимости данных между градиентами на разных слоях модели, то обмен данными для обмена градиентами между всеми вычислительными узлами может осуществляться одновременно с расчетами для других слоев. Необходимо дублировать вычисления и передачу данных для полного использования аппаратных ресурсов и достижения высокого ускорения. В научной литературе есть работы, посвященные эффективному распределению рабочей нагрузки для снижения затрат на коммуникации между узлами. Работа [188] демонстрирует влияние дублирования на масштабируемость обучения СНС, когда архитектура СНС распараллеливается с помощью параллелизма данных. Модели обучаются с помощью синхронного SGD, поэтому результаты параллельного и последовательного обучения одинаковы. Более оптимальная стратегия распараллеливания основана на двух ключевых приемах, направленных на максимизацию перекрытия между вычислениями и коммуникациями узлов. На начальном этапе все градиенты параметров объединяются в два больших блока с последующим их уменьшением

по всем узлам с использованием асинхронной связи. Далее, исходя из анализа зависимости данных, необходимо максимизировать перекрытие путем выбора оптимального размера каждого фрагмента градиента. Кроме того, вычисления градиента повторяются в нескольких полносвязных слоях. В зависимости от архитектуры модели пересчет градиента может значительно снизить коммуникационные затраты. Это также позволяет перекрыть время связи с временем прямого распространения ошибки для следующего мини-пакета. Данный метод позволил добиться ускорения обучения модели VGG-A до 77.97 раз при использовании 128 узлов.

4.4. Специфика FPGA и ASIC

Многие исследования и коммерческие проекты предлагают разработку специализированных аппаратных ускорителей для обучения и исполнения нейронных сетей [144, 189–191]. Наиболее важной причиной для разработки специализированных архитектур является максимальное использование параллелизма, присущего моделям машинного обучения и нейронным сетям, особенно глубоким нейронным сетям. Как правило, CPU, GPU и другие ускорители общего назначения не достигают оптимальной производительности в конкретных задачах. В то же время высокая универсальность таких процессоров означает большее энергопотребление и высокую цену. Даже если процессоры имеют специальные модули для обработки нейронных сетей, они лишь дополняют основную архитектуру [192].

В настоящее время как в литературе, так и в коммерческих реализациях предложено большое количество специализированных архитектур вычислительных ускорителей. Часто специализированные ускорители основаны на технологии Field-Programmable Gate Array (FPGA) или Application Specific Integrated Circuit (ASIC). Основным преимуществом FPGA является возможность организации достаточно гибкой параллельной вычислительной системы с заданной точностью. Современные программируемые логические интегральные схемы (ПЛИС) могут включать несколько различных вариантов вычислений, таких как таблицы соответствия (LUT), цифровая обработка сигналов (DSP) и двоичная логика (flip-flop). Вычисления могут выполняться на любом из этих компонентов. Во многих случаях ПЛИС реализуются как гетерогенные системы с контроллером или процессором (технология System-on-Chip). Это позволяет разработчикам обеспечить наиболее гибкий подход к аппаратной и программной реконфигурации блока исполнения.

Технология FPGA — популярная технология для реализации аппаратных ускорителей нейронных сетей. Также ускорители могут быть изготовлены на основе технологии ASIC [193]. Следует отметить, что обычно специализированные архитектуры FPGA основаны на принципах SIMD CPU или SIMT GPGPU. Помимо анализа аппаратных ускорителей, в некоторых работах рассматриваются программно-аппаратные архитектуры с возможностью реконфигурации [194]. Существуют следующие преимущества специализированных ускорителей нейронных сетей, помимо энергоэффективности, цены и размера:

- оптимизированный набор инструкций (Instructure Set Architecture, ISA);
- более высокая степень параллелизации, степень повторного использования данных и буферизации (с использованием встроенных буферов First In, First Out, FIFO);
- возможность обработки специализированных форматов данных (задаваемая пользователем битовая глубина), а также разреженных вычислений.

В некоторых реализациях вместо ISA вычисления нейронной сети выполняются с помощью машины конечных состояний [195].

Таблица 4. Параллельные архитектуры, используемые для нейронных сетей

Тип оборудования	Ссылки	Преимущества	Недостатки
CPU	[114–119]	Широкая доступность, широкая поддержка	Ограниченная производительность, высокое энергопотребление
GPGPU	[120–122]	Лучшая производительность по цене, хорошая масштабируемость	Высокое энергопотребление, высокая цена
FPGA/ASIC	[119, 130, 144, 189–193, 196, 197]	Низкое энергопотребление, низкая цена, хорошая масштабируемость	Ограниченная производительность

Отмечается, например, [130], что ускорители FPGA лучше подходят для CHC, чем GPU и CPU, поскольку они могут создавать произвольные конфигурации вычислительных модулей. Эти модули обычно формируются в виде элемента обработки (Processing Element, PE). Блок PE представляет собой элемент скалярных, векторных или матричных вычислений. В литературе рассматриваются архитектуры элемента обработки с SIMD (обычно для сопроцессоров CPU) или SIMT (для GPGPU). Эти конфигурации представлены в классе темпоральных архитектур [119]. Специализированные конфигурации PE для ускорителей DNN включают архитектуры Very Long Instruction Word (VLIW) и Decoupled Access/Execute, а также систолические массивы [196]

Таблица 5. Методы ускорения реализации нейронных сетей

Техника	Ссылки	Описание
Сокращенная и смешанная точность	[123–125, 130, 136–138, 199–202]	Уменьшает размер моделей, ускоряет обучение и выполнение
Разреженные матрицы	[130, 139, 140]	Сокращает время выполнения
Оптимизация доступа к памяти	[133, 143–146]	Оптимизация циклов, оптимизация свертки матрицы
Параллелизм данных	[148–150, 152, 156, 167, 171]	Распределение входных данных
Параллелизм модели	[168, 169, 172–175, 203]	Распределение структуры сети
Конвейеризация	[171, 177–183]	Распределение слоев сети или этапов обучения
Комбинированный параллелизм	[185, 186]	Комбинация нескольких стратегий параллелизма

Архитектуры типа систолических массивов реализуют принцип обработки потока данных и относятся к классу пространственных ускорителей [119]. Систолические массивы могут быть как статическими, так и реконфигурируемыми. Большинство ускорителей DNN используют статические массивы, построенные по технологии ASIC [197]. В настоящее время наиболее распространенным типом систолического массива является Tensor Processor Unit (TPU) [193].

Систолические массивы обладают такими преимуществами, как высокая степень повторного использования данных, низкие требования к памяти и низкое энергопотребление. Целью использования этих типов архитектуры является снижение влияния задержки памяти вне кристалла за счет оптимизации структуры процессора под архитектуру нейронных сетей [198]. Систолические тензорные массивы также могут быть оптимизированы для разряженных матричных умножений [197]. Многочисленные источники (см. обзор [119]) говорят о том, что помимо TPU перспективными могут быть и некоторые другие типы пространственных архитектур ускорителей нейронных сетей. Эти системы предназначены для ускорения операций обобщенного матричного умножения и свертки с использованием множества элементов с регулярной организацией.

В табл. 4 и 5 представлен обобщенный обзор литературы для этого раздела.

Заключение

Проведен обзор применения методов глубокого обучения в компьютерном зрении для решения задач фрагментации горных пород и задач в горнодобывающей промышленности. Результаты анализа показали, что большинство современных работ сосредоточено на проведении оценок параметров фрагментов горных пород с использованием систем компьтерного зрения на базе сверточных нейронных сетей глубого обучения. При этом рассматриваются подходы семантической сегментации в сочетании с дополнительными операциями или подходы на основе экземплярной сегментации. Для реализации обучения и работы нейронных сетей используются параллельные вычислительные архитектуры.

Перечислим основные тенденции, представленные в литературе. Тенденции разделены в соответствии с методологией исследования и поставленными выше вопросами.

Вопрос 1. Какие методы решения задач компьютерного зрения применяются при оценке фрагментации горных пород и в смежных областях горнодобывающей промышленности?

- Сверточные нейронные сети являются одним из самых популярных подходов к решению проблемы оценки фрагментации горных пород.
- Оригинальные архитектуры U-Net и Mask R-CNN могут применяться в качестве базовых решений.
- Задачи требуют быстрых вычислений при сохранении высокой точности. Эти требования могут быть выполнены с помощью современных облегченных архитектур нейронных сетей, с одной стороны, и методов оптимизации вычислений для глубокого обучения, с другой стороны.

Вопрос 2. Какие подходы к решению соотвествующих задач компьютерного зрения являются наиболее современными?

- Использование современных архитектур кодировщиков признаков, достигающих наилучшей точности при условии ориентации на работу в реальном времени.
- Поиск наиболее эффективных комбинаций сверток и других операций, в том числе блоков трансформеров и операций внимания.
- Стратегия обучения и регуляризации модели может включать различные приемы: кросс-аугментация данных, пакетная нормализации и мета-обучение для снижения вычислительных затрат при сохранении высокой точности.
- Быстрые модели решения задач обнаружения объектов и экземплярной сегментации (модели реального времени) могут быть модифицированы с помощью различных кодировщиков, обеспечивающих снижение вычислительной сложности при минимальных потерях точности.

Вопрос 3. Какие параллельные архитектуры и методы оптимизации используются при реализации сверточных нейронных сетей?

- Наиболее популярным средством для реализации искусственных нейронных сетей в научных и промышленных приложениях являются графические процессоры.
- Для реализации архитектур и процессов их обучения большинство производителей вычислительных устройств и разработчиков программного обеспечения вводят возможности вычислений со смешанной и пониженной точностью, а также работы с разреженными матрицами. Данная тенденция связана с попытками сократить время выполнения вычислительных операций, а также уменьшить объем данных в памяти. Это позволяет увеличить размер пакетов при обучении нейронных сетей, а также обеспечить работу сетей на низкопроизводительных устройствах.

Сформулируем основные результаты нашего обзора.

- Обзор посвящен проблемам использования систем компьютерного зрения для решения задач оценки фрагментации горных пород и других аналогичных задач горнодобывающей промышленности. Как показывают рассмотренные публикации, в настоящее время иследователи в этой области продолжают использовать базовые устаревшие подходы компьютерного зрения как с нейронными сетями глубокого обучения, так и без них. Большинство этих подходов относится к работам, опубликованным до 2017 года.
- Мы хотим обратить внимание читателей и исследователей на последние достижения в области компьютерного зрения (с 2017 года по настоящее время) и на использование этих результатов в задачах фрагментации горных пород.
- Результаты обзора показывают, что в исследуемой области наиболее целесообразно использовать подход глубокого обучения нейронных сетей для задач экземплярной сегментации и обнаружения объектов в реальном времени с различными вариантами кодировщиков признаков и решаюших подсетей. Выбор конкретной архитектуры, плюсы и минусы различных подходов, а также рекомендации по их использованию являются предметом отдельного исследования.

Подводя итоги, можно сделать вывод, что в области оценки фрагментации горных пород и связанных с ней задач в горнодобывающей промышленности возможен значитель-

ный прорыв по отношению к текущему состоянию области исследований. Перспективными являются вопросы достижения компромисса между современным состоянием архитектур компьютерного зрения с глубоким обучением и оптимизацией их работы для вычислительных устройств с помощью рассмотренных методов.

Исследование выполнено за счет совместного гранта Российского научного фонда и Правительства Свердловской области №22-21-20051, https://rscf.ru/project/22-21-20051/.

Литература

1. Fu Y., Aldrich C. Deep learning in mining and mineral processing operations: a review // IFAC-PapersOnLine. 2020. Vol. 53, no. 2. P. 11920–11925. DOI: 10.1016/j.ifacol.2020.12.712.

2. Zhou W., Wang H., Wan Z. Ore Image Classification Based on Improved CNN // Computers and Electrical Engineering. 2022. Vol. 99. P. 107819. DOI: 10.1016/j.compeleceng.2022.107819.

3. Liu X., Wang H., Jing H., *et al.* Research on intelligent identification of rock types based on faster R-CNN method // IEEE Access. 2020. Vol. 8. P. 21804–21812. DOI: 10.1109/ACCESS.2020.2968515.

4. Amiripallia S.S., Rao G.N., Beharaa J., Sanjay K. Mineral Rock Classification Using Convolutional Neural Network // First International Conference on Recent Trends in Computing (ICRTC 2021), Virtual, Kopargaon, India, May 21–22, 2021. Advances in Parallel Computing. Vol. 39 / ed. by M. Rajesh, K. Vengatesan, M. Gnanasekar, *et al.* Amsterdam, The Netherlands: IOS Press, 2021. P. 499–505. DOI: 10.3233/APC210235.

5. Karimpouli S., Tahmasebi P. Segmentation of digital rock images using deep convolutional autoencoder networks // Computers & Geosciences. 2019. Vol. 126. P. 142–150. DOI: 10.1016/j.cageo.2019.02.003.

6. He M., Zhang Z., Ren J., *et al.* Deep convolutional neural network for fast determination of the rock strength parameters using drilling data // International Journal of Rock Mechanics and Mining Sciences. 2019. Vol. 123. P. 104084. DOI: 10.1016/j.ijrmms.2019.104084.

7. Alzubaidi F., Mostaghimi P., Swietojanski P., *et al.* Automated lithology classification from drill core images using convolutional neural networks // Journal of Petroleum Science and Engineering. 2021. Vol. 197. P. 107933. DOI: 10.1016/j.petrol.2020.107933.

8. Chen T., Hu N., Niu R., *et al.* Object-Oriented Open-Pit Mine Mapping Using Gaofen-2 Satellite Image and Convolutional Neural Network, for the Yuzhou City, China // Remote Sensing. 2020. Vol. 12, no. 23. P. 3895. DOI: 10.3390/rs12233895.

9. Baek J., Choi Y. Deep neural network for predicting ore production by truck-haulage systems in open-pit mines // Applied Sciences. 2020. Vol. 10, no. 5. P. 1657. DOI: 10.3390/app10051657.

10. Williams J., Singh J., Kumral M., Ramirez Ruiseco J. Exploring deep learning for dig-limit optimization in open-pit mines // Natural Resources Research. 2021. Vol. 30, no. 3. P. 2085–2101. DOI: 10.1007/s11053-021-09864-y.

11. Somua-Gyimah G., Frimpong S., Nyaaba W., Gbadam E. A computer vision system for terrain recognition and object detection tasks in mining and construction environments //

2019 SME Annual Conference and Expo and CMA 121st National Western Mining Conference, Denver, CO, USA, February 24–27, 2019. Society for Mining, Metallurgy and Exploration (SME), 2019.

12. Zeng F., Jacobson A., Smith D., *et al.* Lookup: Vision-only real-time precise underground localisation for autonomous mining vehicles // 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, May 20–24, 2019. IEEE, 2019. P. 1444–1450. DOI: 10.1109/ICRA.2019.8794453.

13. Vu T., Bao T., Hoang Q.V., *et al.* Measuring blast fragmentation at Nui Phao open-pit mine, Vietnam using the Mask R-CNN deep learning model // Mining Technology. 2021. Vol. 130, no. 4. P. 232–243. DOI: 10.1080/25726668.2021.1944458.

14. Zyuzin V., Ronkin M., Porshnev S., Kalmykov A. Computer vision system for the automatic asbestos content control in stones // Big Data and AI Conference 2020, Moscow, Russian Federation, September 17–18, 2020. IOP Publishing: Journal of Physics: Conference Series. Vol. 1727, 2021. P. 012014. DOI: 10.1088/1742-6596/1727/1/012014.

15. Zyuzin V., Ronkin M., Porshnev S., Kalmykov A. Automatic Asbestos Control Using Deep Learning Based Computer Vision System // Applied Sciences. 2021. Vol. 11, no. 22. P. 10532. DOI: 10.3390/app112210532.

16. Ronkin M., Kalmykov A., Reshetnikov K., Zyuzin V. Investigation of Object Detection Based Method for Open-Pit Blast Quality Estimation // 2022 Ural-Siberian Conference on Biomedical Engineering, Radioelectronics and Information Technology (USBEREIT), Yekaterinburg, Russian Federation, September 19–21, 2022. IEEE, 2022. P. 248–251. DOI: 10.1109/USBEREIT56278.2022.9923353.

17. Gao R., Sun Z., Li W., *et al.* Automatic coal and gangue segmentation using U-Net based fully convolutional networks // Energies. 2020. Vol. 13, no. 4. P. 829. DOI: 10.3390/en13040829.

18. Sangaiah A.K. Deep learning and parallel computing environment for bioengineering systems. St. Louis, MO, USA: Academic Press, 2019. 280 p.

19. Ronkin M. V., Akimova E. N., Misilov V. E. Review of deep learning approaches in solving rock fragmentation problems // AIMS Mathematics. 2023. Vol. 8, no. 10. P. 23900–23940. DOI: 10.3934/math.20231219.

20. Liu X., Zhang Y., Jing H., *et al.* Ore image segmentation method using U-Net and Res_Unet convolutional networks // RSC advances. 2020. Vol. 10, no. 16. P. 9396–9406. DOI: 10.1039/C9RA05877J.

21. Si L., Xiong X., Wang Z., Tan C. A deep convolutional neural network model for intelligent discrimination between coal and rocks in coal mining face // Mathematical Problems in Engineering. 2020. Vol. 2020. P. 2616510. DOI: 10.1155/2020/2616510.

22. Su C., Xu S.-j., Zhu K.-y., Zhang X.-c. Rock classification in petrographic thin section images based on concatenated convolutional neural networks // Earth Science Informatics. 2020. Vol. 13, no. 4. P. 1477–1484. DOI: 10.1007/s12145-020-00505-1.

23. Ronkin M., Reshetnikov K., Zyuzin V. Open-Pits asbestos. 2022. DOI: 10.17632/pfdbfpfygh. (accessed: 16.12.2022).

24. Ronkin M., Reshetnikov K., Zyuzin V., *et al.* Asbest veins in the open pit conditions. 2022. DOI: 10.17632/y2jfk63tpd. (accessed: 16.12.2022).

25. Babaeian M., Ataei M., Sereshki F., *et al.* A new framework for evaluation of rock fragmentation in open pit mines // Journal of Rock Mechanics and Geotechnical Engineering. 2019. Vol. 11, no. 2. P. 325–336. DOI: 10.1016/j.jrmge.2018.11.006.

26. Li H., Pan C., Chen Z.and Wulamu A., Yang A. Ore image segmentation method based on U-Net and watershed // Comput. Mater. Contin. 2020. Vol. 65. P. 563–578. DOI: 10.32604/cmc.2020.09806.

27. Mkwelo S., Nicolls V., De Jager G. Watershed-based segmentation of rock scenes and proximity-based classification of watershed regions under uncontrolled lighting // SAIEE Africa Research Journal. 2005. Vol. 96, no. 1. P. 28–34. DOI: 10.23919/SAIEE.2005.9488146.

28. Bamford T., Esmaeili K., Schoellig A.P. A deep learning approach for rock fragmentation analysis // International Journal of Rock Mechanics and Mining Sciences. 2021. Vol. 145. P. 104839. DOI: 10.1016/j.ijrmms.2021.104839.

29. Jung D., Choi Y. Systematic review of machine learning applications in mining: Exploration, exploitation, and reclamation // Minerals. 2021. Vol. 11, no. 2. P. 148. DOI: 10.3390/min11020148.

30. Franklin J.A., Katsabanis T. Measurement of blast fragmentation. Rotterdam, the Netherlands: A. A. Balkema, 1996. 324 p.

31. Tosun A. A modified Wipfrag program for determining muckpile fragmentation // Journal of the Southern African Institute of Mining and Metallurgy. 2018. Vol. 118, no. 10. P. 1113–1199. DOI: 10.17159/2411-9717/2018/v118n10a13.

32. Latham J.-P., Kemeny J., Maerz N., *et al.* A blind comparison between results of four image analysis systems using a photo-library of piles of sieved fragments // Fragblast. 2003. Vol. 7, no. 2. P. 105–132. DOI: 10.1076/frag.7.2.105.15899.

33. Ronneberger O., Fischer P., Brox T. U-Net: Convolutional networks for biomedical image segmentation // International Conference on Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, Munich, Germany, October 5–9, 2015. Proceedings, Part III. Vol. 9351 / ed. by N. Navab, J. Hornegger, W. Wells, A. Frangi. Springer, 2015. P. 234–241. Lecture Notes in Computer Science. DOI: 10.1007/978-3-319-24574-4_28.

34. Siddique N., Paheding S., Elkin C.P., Devabhaktuni V. U-net and its variants for medical image segmentation: A review of theory and applications // IEEE Access. 2021. Vol. 9. P. 82031–82057. DOI: 10.1109/ACCESS.2021.3086020.

35. Yin X.-X., Sun L., Fu Y., *et al.* U-Net-Based Medical Image Segmentation // Journal of Healthcare Engineering. 2022. Vol. 2022. P. 4189781 DOI: 10.1155/2022/4189781.

36. Wu J., Liu W., Li C., *et al.* A State-of-the-art Survey of U-Net in Microscopic Image Analysis: from Simple Usage to Structure Mortification // CoRR. 2022. Vol. abs/2202.06465. arXiv: 2202.06465. URL: https://arxiv.org/abs/2202.06465.

37. Beucher S. Use of watersheds in contour detection // International Workshop on Image Processing: Real-time Edge and Motion detection/estimation, Rennes, France, September 17–21, 1979. CCETT, 1979.

38. Guo Q., Wang Y., Yang S., Xiang Z. A method of blasted rock image segmentation based on improved watershed algorithm // Scientific Reports. 2022. Vol. 12, no. 1. P. 1–21. DOI: 10.1038/s41598-022-11351-0.

39. Gu W., Bai S., Kong L. A review on 2D instance segmentation based on deep neural networks // Image and Vision Computing. 2022. Vol. 120. P. 104401. DOI: 10.1016/j.imavis.2022.104401.

40. Hafiz A.M., Bhat G.M. A survey on instance segmentation: state of the art // International journal of multimedia information retrieval. 2020. Vol. 9, no. 3. P. 171–189. DOI: 10.1007/s13735-020-00195-x.

41. He K., Gkioxari G., Dollár P., Girshick R. Mask R-CNN // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2020. Vol. 42, no. 2. P. 386–397. DOI: 10.1109/TPAMI.2018.2844175.

42. He K., Zhang X., Ren S., Sun J. Deep residual learning for image recognition // 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, June 27–30, 2016. IEEE, 2016. P. 770–778. DOI: 10.1109/CVPR.2016.90.

43. Ramesh C.S., *et al.* A Review on Instance Segmentation Using Mask R-CNN // Proceedings of the International Conference on Systems, Energy & Environment (ICSEE) 2021, Kerala, India, January 22–23, 2021. SSRN, 2021. P. 183–186. DOI: 10.2139/ssrn.3794272.

44. Schenk F., Tscharf A., Mayer G., Fraundorfer F. Automatic muck pile characterization from UAV images // ISPRS Geospatial Week 2019, Enschede, The Netherlands, June 10–14, 2019. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences. 2019. Vol. IV-2/W5. P. 163–170. DOI: 10.5194/isprs-annals-IV-2-W5-163-2019.

45. Maitre J., Bouchard K., Bédard L.P. Mineral grains recognition using computer vision and machine learning // Computers & Geosciences. 2019. Vol. 130. P. 84–93. DOI: 10.1016/j.cageo.2019.05.009.

46. Jocher G., Chaurasia A., Stoken A., *et al.* ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation. DOI: 10.5281/zenodo.7347926.

47. Zaidi S.S.A., Ansari M.S., Aslam A., *et al.* A survey of modern deep learning based object detection models // Digital Signal Processing. 2022. P. 103514. DOI: 10.1016/j.dsp.2022.103514.

48. Mo Y., Wu Y., Yang X., *et al.* Review the state-of-the-art technologies of semantic segmentation based on deep learning // Neurocomputing. 2022. Vol. 493. P. 626–646. DOI: 10.1016/j.neucom.2022.01.005.

49. Minaee S., Boykov Y.Y., Porikli F., *et al.* Image segmentation using deep learning: A survey // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2021. Vol. 44, no. 7. P. 3523–3542. DOI: 10.1109/TPAMI.2021.3059968.

50. Yuan X., Shi J., Gu L. A review of deep learning methods for semantic segmentation of remote sensing imagery // Expert Systems with Applications. 2021. Vol. 169. P. 114417. DOI: 10.1016/j.eswa.2020.114417.

51. PapersWithCode.com. Semantic segmentation benchmarks. URL: https://paperswithcode.com/task/semantic-segmentation (accessed: 25.11.2022).

52. PapersWithCode.com. Real-time semantic segmentation benchmarks. URL: https://paperswithcode.com/task/real-time-semantic-segmentation/latest (accessed: 25.11.2022).

53. Carvalho O.L.F.d., Carvalho Junior O.A. de, Albuquerque A.O.d., *et al.* Instance segmentation for large, multi-channel remote sensing imagery using Mask-RCNN and a mosaicking approach // Remote Sensing. 2020. Vol. 13, no. 1. P. 39. DOI: 10.3390/rs13010039.

54. PapersWithCode.com. Instance segmentation benchmarks. URL: https://paperswithcode.com/task/instance-segmentation (accessed: 25.11.2022).

55. PapersWithCode.com. Real-time Instance Segmentation on MSCOCO. URL: https://paperswithcode.com/sota/real-time-instance-segmentation-on-mscoco (accessed: 25.11.2022).

56. Hossain S., Lee D.-j. Deep learning-based real-time multiple-object detection and tracking from aerial imagery via a flying robot with GPU-based embedded devices // Sensors. 2019. Vol. 19, no. 15. P. 3371. DOI: 10.3390/s19153371.

57. Strudel R., Garcia R., Laptev I., Schmid C. Segmenter: Transformer for semantic segmentation // 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, October 10–17, 2021. IEEE, 2022. P. 7262–7272. DOI: 10.1109/ICCV48922.2021.00717.

58. Liu Z., Lin Y., Cao Y., *et al.* Swin transformer: Hierarchical vision transformer using shifted windows // 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, October 10–17, 2021. IEEE, 2022. P. 10012–10022. DOI: 10.1109/ICCV48922.2021.00986.

59. LeCun Y., Boser B., Denker J.S., *et al.* Backpropagation applied to handwritten zip code recognition // Neural computation. 1989. Vol. 1, no. 4. P. 541–551. DOI: 10.1162/neco.1989.1.4.541.

60. LeCun Y., Bottou L., Bengio Y., Haffner P. Gradient-based learning applied to document recognition // Proceedings of the IEEE. 1998. Vol. 86, no. 11. P. 2278–2324. DOI: 10.1109/5.726791.

61. Goodfellow I., Bengio Y., Courville A. Deep Learning. Cambridge, MA, USA: MIT Press, 2016. 800 p. URL: http://www.deeplearningbook.org.

62. Zhang A., Lipton Z.C., Li M., Smola A.J. Dive into deep learning // Cambridge, UK: Cambridge University Press, 2023. URL: https://D2L.ai

63. Krizhevsky A., Sutskever I., Hinton G.E. ImageNet classification with deep convolutional neural networks // Communications of the ACM. 2017. Vol. 60, no. 6. P. 84–90. DOI: 10.1145/3065386.

64. Alom M.Z., Taha T.M., Yakopcic C., *et al.* The history began from AlexNet: A comprehensive survey on deep learning approaches // CoRR. 2018. Vol. abs/1803.01164. arXiv: 1803.01164. URL: https://arxiv.org/abs/1803.01164.

65. Simonyan K., Zisserman A. Very deep convolutional networks for large-scale image recognition // 3rd International Conference on Learning Representations, ICLR 2015, San

Diego, CA, USA, May 7-9, 2015. Conference Track Proceedings // ed. by Y. Bengio, Y. LeCun. URL: https://arxiv.org/abs/1409.1556.

66. Szegedy C., Vanhoucke V., Ioffe S., *et al.* Rethinking the inception architecture for computer vision // 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, June 27–30, 2016. IEEE, 2016. P. 2818–2826. DOI: 10.1109/CVPR.2016.308.

67. Lin M., Chen Q., Yan S. Network in network // 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014. Conference Track Proceedings / ed. by Y. Bengio, Y. LeCun. URL: https://arxiv.org/abs/1312.4400.

68. Chollet F. Xception: Deep learning with depthwise separable convolutions // 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, July 21–26, 2017. IEEE, 2017. P. 1251–1258. DOI: 10.1109/CVPR.2017.195.

69. Szegedy C., Liu W., Jia Y., *et al.* Going deeper with convolutions // 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, June 7–12, 2015. IEEE, 2015. P. 1–9. DOI: 10.1109/CVPR.2015.7298594.

70. Ioffe S., Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift // Proceedings of the 32nd International Conference on Machine Learning, ICML'15, Lille, France, July 7–9, 2015. Proceedings of Machine Learning Research. Vol. 37 / ed. by F. Bach, D. Blei. PMLR, 2015. P. 448–456. URL: https://proceedings.mlr.press/v37/ioffe15.html.

71. Kingma D.P., Ba J. Adam: A method for stochastic optimization // 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7–9, 2015. Conference Track Proceedings / ed. by Y. Bengio, Y. LeCun. URL: https://arxiv.org/abs/1412.6980.

72. He K., Zhang X., Ren S., Sun J. Identity mappings in deep residual networks // European conference on computer vision – ECCV 2016, Amsterdam, The Netherlands, October 11–14, 2016. Proceedings, Part IV. Vol. 9908 / ed. by B. Leibe, J. Matas, N. Sebe, M. Welling. Springer, 2016. P. 630–645. Lecture Notes in Computer Science. DOI: 10.1007/978-3-319-46493-0_38.

73. PapersWithCode.com. Convolutional neural networks. URL: https://paperswithcode.com/methods/category/convolutional-neural-networks (accessed: 25.11.2022).

74. PapersWithCode.com. Most popular image models. URL: https://paperswithcode.com/methods/category/image-models (accessed: 25.11.2022).

75. Xie S., Girshick R., Dollár P., *et al.* Aggregated residual transformations for deep neural networks // 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, July 21–26, 2017. IEEE, 2017. P. 5987–5995. DOI: 10.1109/CVPR.2017.634.

76. Huang G., Liu Z., Van Der Maaten L., Weinberger K.Q. Densely connected convolutional networks // 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, July 21–26, 2017. IEEE, 2017. P. 2261–2269. DOI: 10.1109/CVPR.2017.243.

77. He T., Zhang Z., Zhang H., *et al.* Bag of tricks for image classification with convolutional neural networks // 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition

(CVPR), Long Beach, CA, USA, June 15–20, 2019. IEEE, 2020. P. 558–567. DOI: 10.1109/CVPR.2019.00065.

78. Kolesnikov A., Beyer L., Zhai X., *et al.* Big transfer (bit): General visual representation learning // European conference on computer vision – ECCV 2020, Glasgow, UK, August 23–28, 2020. Proceedings, Part V. Vol. 12350 / ed. by A. Vedaldi, H. Bischof, T. Brox, JM. Frahm. Springer, 2020. P. 491–507. Lecture Notes in Computer Science. DOI: 10.1007/978-3-030-58558-7_29.

79. Radosavovic I., Kosaraju R.P., Girshick R., *et al.* Designing network design spaces // 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, June 13–19, 2020. IEEE, 2020. P. 10425–10433. DOI: 10.1109/CVPR42600.2020.01044.

80. Sandler M., Howard A., Zhu M., *et al.* MobileNetV2: Inverted Residuals and Linear Bottlenecks // 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, June 18–23, 2018. IEEE, 2018. P. 4510–4520. DOI: 10.1109/CVPR.2018.00474.

81. Kyriakides G., Margaritis K. An introduction to neural architecture search for convolutional networks // CoRR. 2020. Vol. abs/2005.11074. arXiv: 2005.11074. URL: https://arxiv.org/abs/2005.11074.

82. He X., Zhao K., Chu X. AutoML: A survey of the state-of-the-art // Knowledge-Based Systems. 2021. Vol. 212. P. 106622. DOI: 10.1016/j.knosys.2020.106622.

83. Hu J., Shen L., Sun G. Squeeze-and-excitation networks // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2020. Vol. 42, no. 8. P. 2011–2023. DOI: 10.1109/TPAMI.2019.2913372.

84. Tan M., Le Q. EfficientNet: Rethinking model scaling for convolutional neural networks // Proceedings of the 36th International Conference on Machine Learning, Long Beach, CA, USA, June 9–15, 2019. Proceedings of Machine Learning Research. Vol. 97 / ed. by K. Chaudhuri, R. Salakhutdinov. PMLR, 2019. P. 6105–6114. URL: https://proceedings.mlr.press/v97/tan19a.html.

85. Dosovitskiy A., Beyer L., Kolesnikov A., *et al.* An image is worth 16x16 words: Transformers for image recognition at scale // 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3–7, 2021. URL: https://arxiv.org/abs/2010.11929.

86. Khan S., Naseer M., Hayat M., *et al.* Transformers in vision: A survey // ACM Computing Surveys. 2021. Vol. 54, no. 10s. P. 1–41. DOI: 10.1145/3505244.

87. Vaswani A., Shazeer N., Parmar N., *et al.* Attention is all you need // 31st Annual Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA, December 4–9, 2017. Advances in Neural Information Processing Systems. Vol. 30 / ed. by I. Guyon, U. von Luxburg, S. Bengio, *et al.* Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

88. Ba J.L., Kiros J.R., Hinton G.E. Layer normalization // CoRR. 2016. Vol. abs/1607.06450. arXiv: 1607.06450. URL: https://arxiv.org/abs/1607.06450.

89. Dai Z., Liu H., Le Q.V., Tan M. Coatnet: Marrying convolution and attention for all data sizes // 35th Conference on Neural Information Processing Systems (NeurIPS

2021), Virtual, Online, December 6–14, 2021. Advances in Neural Information Processing Systems. Vol. 34 / ed. by M. Ranzato, A. Beygelzimer, Y. Dauphin, *et al.* Curran Associates, Inc., 2021. P. 3965–3977. URL: https://proceedings.neurips.cc/paper/2021/hash/20568692db622456cc42a2e853ca21f8-Abstract.html.

90. Tu Z., Talebi H., Zhang H., *et al.* MaxViT: Multi-axis Vision Transformer // 17th European conference on computer vision – ECCV 2022, Tel Aviv, Israel, October 23–27, 2022. Proceedings, Part XXIV. Vol. 13684 / ed. by S. Avidan, G. Brostow, M. Cissé, *et al.* Springer, 2022. P. 459–479. Lecture Notes in Computer Science. DOI: 10.1007/978-3-031-20053-3_27.

91. Mehta S., Rastegari M. MobileViT: Light-weight, General-purpose, and Mobile-friendly Vision Transformer // The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25–29, 2022. URL: https://arxiv.org/abs/2110.02178.

92. Tolstikhin I.O., Houlsby N., Kolesnikov A., *et al.* MLP-Mixer: An all-MLP Architecture for Vision // 35th Conference on Neural Information Processing Systems (NeurIPS 2021), Virtual, Online, December 6–14, 2021. Advances in Neural Information Processing Systems. Vol. 34 / ed. by M. Ranzato, A. Beygelzimer, Y. Dauphin, *et al.* Curran Associates, Inc., 2021. P. 24261–24272. URL: https://proceedings.neurips.cc/paper/2021/hash/cba0a4ee5ccd02fda0fe3f9a3e7b89fe-Abstract.html.

93. Touvron H., Cord M., Douze M., *et al.* Training data-efficient image transformers & distillation through attention // Proceedings of the 38th International Conference on Machine Learning, Virtual, July 18–24, 2021. Proceedings of Machine Learning Research. Vol. 139 / ed. by M. Meila, T. Zhang. PMLR, 2021. P. 10347–10357. URL: https://proceedings.mlr.press/v139/touvron21a.

94. Hospedales T.M., Antoniou A., Micaelli P., Storkey A.J. Meta-learning in neural networks: A survey // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2021. Vol. 44, no. 9. P. 5149–5169. DOI: 10.1109/TPAMI.2021.3079209.

95. Naveed H. Survey: Image mixing and deleting for data augmentation // CoRR. 2023. Vol. abs/2106.07085. arXiv: 2106.07085. URL: https://arxiv.org/abs/2106.07085.

96. PapersWithCode.com. Image Classification on ImageNet. URL: https://paperswithcode.com/sota/image-classification-on-imagenet (accessed: 25.11.2022).

97. Bolya D., Zhou C., Xiao F., Lee Y.J. Yolact: Real-time instance segmentation // 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea (South), October 27 – November 2, 2019. IEEE, 2020. P. 9157–9166. DOI: 10.1109/ICCV.2019.00925.

98. Cheng T., Wang X., Chen S., *et al.* Sparse Instance Activation for Real-Time Instance Segmentation // 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, June 18–24, 2022. IEEE, 2022. P. 4433–4442. DOI: 10.1109/CVPR52688.2022.00439.

99. Wang C.-Y., Bochkovskiy A., Liao H.-Y.M. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors // 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, Canada, June 20–22, 2023. IEEE, 2023. P. 7464–7475.

100. Bolya D., Zhou C., Xiao F., Lee Y.J. Yolact++: Better real-time instance segmentation // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2022. Vol. 44, no. 2. P. 1108–1121. DOI: 10.1109/TPAMI.2020.3014297.

101. Wang X., Zhang R., Kong T., *et al.* Solov2: Dynamic and fast instance segmentation // Annual Conference on Neural Information Processing Systems (NeurIPS 2020), Virtual, December 6–12, 2020. Advances in Neural Information Processing Systems. Vol. 33 / ed. by H. Larochelle, and M. Ranzato, R. Hadsell, *et al.* Curran Associates, Inc., 2020. P. 17721–17732. URL: https://proceedings.neurips.cc/paper/2020/hash/cd3afef9b8b89558cd56638c3631868a-Abstract.html.

102. Wang X., Kong T., Shen C., *et al.* Solo: Segmenting objects by locations // European conference on computer vision – ECCV 2020, Glasgow, UK, August 23–28, 2020. Proceedings, Part V. Vol. 12350 / ed. by A. Vedaldi, H. Bischof, T. Brox, J.M. Frahm. Springer, 2020. P. 649–665. Lecture Notes in Computer Science. DOI: 10.1007/978-3-030-58523-5_38.

103. Li C., Li L., Jiang H., *et al.* YOLOv6: A single-stage object detection framework for industrial applications // CoRR. 2022. Vol. abs/2209.02976. arXiv: 2209.02976. URL: https://arxiv.org/abs/2209.02976.

104. Jocher G. Ultralytics YOLOv8. URL: https://github.com/ultralytics/ultralytics (accessed: 20.04.2023).

105. Diwan T., Anirudh G., Tembhurne J.V. Object detection using YOLO: Challenges, architectural successors, datasets and applications // Multimedia Tools and Applications. 2023. Vol. 82, P. 9243–9275. DOI: 10.1007/s11042-022-13644-y.

106. Jiang P., Ergu D., Liu F., *et al.* A Review of Yolo algorithm developments // Procedia Computer Science. 2022. Vol. 199. P. 1066–1073. DOI: 10.1016/j.procs.2022.01.135.

107. Redmon J., Divvala S., Girshick R., Farhadi A. You only look once: Unified, real-time object detection // 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, June 27–30, 2016. IEEE, 2016. P. 779–788. DOI: 10.1109/CVPR.2016.91.

108. Bochkovskiy A., Wang C.-Y., Liao H.-Y.M. YOLOv4: Optimal speed and accuracy of object detection // CoRR. 2020. Vol. abs/2004.10934. arXiv: 2004.10934. URL: https://arxiv.org/abs/2004.10934.

109. Tan M., Le Q. EfficientNetv2: Smaller models and faster training // Proceedings of the 38th International Conference on Machine Learning, Virtual, July 18–24, 2021. Proceedings of Machine Learning Research. Vol. 139 / ed. by M. Meila, T. Zhang. PMLR, 2021. P. 10096–10106. URL: http://proceedings.mlr.press/v139/tan21a.html.

110. Brock A., De S., Smith S.L., Simonyan K. High-performance large-scale image recognition without normalization // Proceedings of the 38th International Conference on Machine Learning, Virtual, July 18–24, 2021. Proceedings of Machine Learning Research. Vol. 139 / ed. by M. Meila, T. Zhang. PMLR, 2021. P. 1059–1071. URL: https://proceedings.mlr.press/v139/brock21a.html.

111. PapersWithCode.com. Real-Time Object Detection. URL: https://paperswithcode.com/task/real-time-object-detection (accessed: 25.11.2022).

112. Chen K., Pang J., Wang J., *et al.* Hybrid Task Cascade for Instance Segmentation // 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, June 15–20, 2019. IEEE, 2020. P. 4974–4983. DOI: 10.1109/CVPR.2019.00511.

113. Chen L.-C., Zhu Y., Papandreou G., *et al.* Encoder-decoder with atrous separable convolution for semantic image segmentation // European conference on computer vision – ECCV 2018, Munich, Germany, September 8–14, 2018. Proceedings, Part VII. Vol. 11211 / ed. by V. Ferrari, M. Hebert, C. Sminchisescu, Y. Weiss. Springer, 2018. P. 801–818. Lecture Notes in Computer Science. DOI: 10.1007/978-3-030-01234-2_49.

114. Intel. Open Vino. 2023. URL: https://docs.openvino.ai/2023.0/home.html (accessed: 23.08.2023).

115. Georgia Tech and Facebook Artificial Intelligence Research. NNpack acceleration package for neural networks on multi-core CPUs. 2022. URL: https://github.com/Maratyszcza/NNPACK (accessed: 01.10.2022).

116. Intel. oneDNN Intel math kernel library for deep neural networks (Intel MKL-DNN) and deep neural network library (DNNL). 2022. URL: https://github.com/oneapi-src/oneDNN (accessed: 01.10.2022).

117. Hadjis S., Abuzaid F., Zhang C., Ré C. Caffe con Troll: Shallow Ideas to Speed Up Deep Learning // DanaC'15: Proceedings of the Fourth Workshop on Data analytics in the Cloud, Melbourne, Australia, May 31–June 4, 2015 / ed. by A. Katsifodimos. New York: ACM, 2015. P. 1–4. DOI: 10.1145/2799562.2799641.

118. Dai, J. J., Ding, D., Shi, D., *et al.* BigDL 2.0: Seamless Scaling of AI Pipelines from Laptops to Distributed Cluster // 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, June 18–24, 2022. IEEE, 2022. P. 21407–21414. DOI: 10.1109/CVPR52688.2022.02076.

119. Capra M., Bussolino B., Marchisio A., *et al.* An Updated Survey of Efficient Hardware Architectures for Accelerating Deep Convolutional Neural Networks // Future Internet. 2020. Vol. 12, no. 7. P. 113. DOI: 10.3390/fi12070113.

120. Dumas II J.D. Computer architecture: Fundamentals and principles of computer design. CRC Press, 2017. 447 p.

121. NVIDIA Corporation. Artificial Neural Network. 2022. URL: https://developer.nvidia.com/discover/artificial-neural-network (accessed: 01.10.2022).

122. NVIDIA Corporation. TensorRT SDK. 2023. URL: https://developer.nvidia.com/tensorrt (accessed: 08.23.2023).

123. Gholami A., Kim S., Dong Z., *et al.* A Survey of Quantization Methods for Efficient Neural Network Inference. Low-Power Computer Vision. New York: Chapman and Hall/CRC, 2022, P. 291–326. DOI: 10.1201/9781003162810-13.

124. Kikuchi Y., Fujita K., Ichimura T., *et al.* Calculation of Cross-correlation Function Accelerated by Tensor Cores with TensorFloat-32 Precision on Ampere GPU // 22nd International Conference – ICCS 2022, London, UK, June 11–23, 2022. Proceedings, Part II. Vol. 13351 / ed. by D. Groen, C. de Mulatier, M. Paszynski, *et al.* Springer, 2022. P. 277–290. Lecture Notes in Computer Science. DOI: 10.1007/978-3-031-08754-7_37.

125. Burel S., Evans A., Anghel L. Zero-Overhead Protection for CNN Weights // 2021 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT), Athens, Greece, October 6–8, 2021. IEEE, 2021. P. 1–6. DOI: 10.1109/DFT52944.2021.9568363.

126. Simons T., Lee D.-J. A Review of Binarized Neural Networks // Electronics. 2019. Vol. 8, no. 6. P. 661. DOI: 10.3390/electronics8060661.

127. Wang Y., Feng B., Ding Y. QGTC: Accelerating Quantized Graph Neural Networks via GPU Tensor Core // PPoPP'22: Proceedings of the 27th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, Seoul, Republic of Korea, April 2–6, 2022. New York: ACM, 2022. P. 107–119. DOI: 10.1145/3503221.3508408.

128. Feng B., Wang Y., Geng T., *et al.* APNN-TC: Accelerating Arbitrary Precision Neural Networks on Ampere GPU Tensor Cores // SC '21: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, St. Louis, MO, USA, November 14–19, 2021. New York: ACM, 2021. P. 1–12. DOI: 10.1145/3458817.3476157.

129. Alemdar H., Leroy V., Prost-Boucle A., Pétrot F. Ternary neural networks for resource-efficient AI applications // 2017 International Joint Conference on Neural Networks (IJCNN), Anchorage, AK, USA, May 14–19, 2017. IEEE, 2017. P. 2547–2554. DOI: 10.1109/IJCNN.2017.7966166.

130. Nurvitadhi E., Venkatesh G., Sim J., *et al.* Can FPGAs Beat GPUs in Accelerating Next-Generation Deep Neural Networks? // FPGA'17: Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, Monterey, CA, USA, February 22–24, 2017. New York: ACM, 2017. P. 5–14. DOI: 10.1145/3020078.3021740.

131. Li Z., Wallace E., Shen S., *et al.* Train Big, Then Compress: Rethinking Model Size for Efficient Training and Inference of Transformers // Proceedings of the International Conference on Machine Learning, Virtual, July 13–18, 2020. Proceedings of Machine Learning Research. Vol. 119 / ed. by H. Daumé III, A. Singh. PMLR, 2020. P. 5958–5968. URL: https://proceedings.mlr.press/v119/li20m.html.

132. Qiu J., Wang J., Yao S., *et al.* Going Deeper with Embedded FPGA Platform for Convolutional Neural Network // FPGA'16: Proceedings of the 2016 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, Monterey, CA, USA, February 21–23, 2016. New York: ACM, 2016. P. 26–35. DOI: 10.1145/2847263.2847265.

133. Li C., Yang Y., Feng M., *et al.* Optimizing Memory Efficiency for Deep Convolutional Neural Networks on GPUs // SC '16: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, Salt Lake City, UT, USA, November 13–18, 2016. IEEE, 2016. P. 633–644. DOI: 10.1109/SC.2016.53.

134. NVIDIA Corporation. Mixed-Precision Programming with CUDA 8. 2016. URL: https://developer.nvidia.com/blog/mixed-precision-programming-cuda-8/ (accessed: 01.10.2022).

135. Anzt H., Tsai Y.M., Abdelfattah A., *et al.* Evaluating the Performance of NVIDIA's A100 Ampere GPU for Sparse and Batched Computations // 2020 IEEE/ACM Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems (PMBS), Virtual, November 12, 2020. IEEE, 2021. P. 26–38. DOI: 10.1109/PMBS51919.2020.00009.

136. Tian R., Zhao Z., Liu W., *et al.* SAMP: A Toolkit for Model Inference with Self-Adaptive Mixed-Precision // CoRR. 2022. Vol. abs/2209.09130. arXiv: 2209.09130. URL: https://arxiv.org/abs/2209.09130.

137. Linux Foundation. Automatic Mixed Precision package - torch.amp. 2022. URL: https://pytorch.org/docs/stable/amp.html (accessed: 01.10.2022).

138. Honka T. Automatic Mixed Precision Quantization of Neural Networks using Iterative Correlation Coefficient Adaptation: PhD thesis / Honka Tapio. Tampere University, Finland, 2021. URL: https://trepo.tuni.fi/handle/10024/135952.

139. Liang T., Glossner J., Wang L., *et al.* Pruning and quantization for deep neural network acceleration: A survey // Neurocomputing. 2021. Vol. 461. P. 370–403. DOI: 10.1016/j.neucom.2021.07.045.

140. Wimmer P., Mehnert J., Condurache A.P. Dimensionality Reduced Training by Pruning and Freezing Parts of a Deep Neural Network, a Survey // Artificial Intelligence Review. 2023. DOI: 10.1007/s10462-023-10489-1.

141. Sun W., Li A., Geng T., *et al.* Dissecting Tensor Cores via Microbenchmarks: Latency, Throughput and Numerical Behaviors // IEEE Transactions on Parallel and Distributed Systems. 2023. Vol. 34, no. 1. P. 246–261. DOI: 10.1109/TPDS.2022.3217824.

142. Wang Y., Yang C., Farrell S., *et al.* Time-Based Roofline for Deep Learning Performance Analysis // 2020 IEEE/ACM Fourth Workshop on Deep Learning on Supercomputers (DLS), Atlanta, GA, USA, November 11, 2020. IEEE, 2020. P. 10–19. DOI: 10.1109/DLS51937.2020.00007.

143. Li Y., Liu Z., Xu K., *et al.* A GPU-outperforming FPGA accelerator architecture for binary convolutional neural networks // ACM Journal on Emerging Technologies in Computing Systems. 2018. Vol. 14, no. 2. P. 1–16. DOI: 10.1145/3154839.

144. Wu R., Guo X., Du J., Li J. Accelerating Neural Network Inference on FPGA-Based Platforms – A Survey // Electronics. 2021. Vol. 10, no. 9. P. 1025. DOI: 10.3390/electronics10091025.

145. Habib G., Qureshi S. Optimization and acceleration of convolutional neural networks: A survey // Journal of King Saud University — Computer and Information Sciences. 2022. Vol. 34, no. 7. P. 4244–4268. DOI: 10.1016/j.jksuci.2020.10.004.

146. Mittal S. A Survey on optimized implementation of deep learning models on the NVIDIA Jetson platform // Journal of Systems Architecture. 2019. Vol. 97. P. 428–442. DOI: 10.1016/j.sysarc.2019.01.011.

147. Xu W., Zhang Y., Tang X. Parallelizing DNN Training on GPUs: Challenges and Opportunities // WWW'21: Companion Proceedings of the Web Conference 2021, Ljubljana, Slovenia, April 19–23, 2021. New York: ACM, 2021. P. 174–178. DOI: 10.1145/3442442.3452055.

148. Le Q.V., Ngiam J., Coates A., *et al.* On optimization methods for deep learning // ICML'11: Proceedings of the 28th International Conference on International Conference on Machine Learning, Bellevue, WA, USA, June 28 – July 2, 2011. Madison, WI, USA: Omnipress, 2011. P. 265–272. DOI: 10.5555/3104482.3104516.

149. Shamir O. Without-Replacement Sampling for Stochastic Gradient Methods // 30th Annual Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, December 5–10, 2016. Advances in Neural Information Processing Systems. Vol. 29 / ed. by D. Lee, M. Sugiyama, U. Luxburg, *et al.* Curran Associates, Inc., 2016. URL: https://proceedings.neurips.cc/paper/2016/hash/c74d97b01eae257e44aa9d5bade97baf-Abstract.html.

150. Wei J., Zhang X., Ji Z., *et al.* Deploying and scaling distributed parallel deep neural networks on the Tianhe-3 prototype system // Scientific Reports. 2021. Vol. 11. P. 20244. DOI: 10.1038/s41598-021-98794-z.

151. Smith S.L., Le Q.V. A Bayesian Perspective on Generalization and Stochastic Gradient Descent // 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 – May 3, 2018. Conference Track Proceedings. URL: https://arxiv.org/abs/1710.06451.

152. Que C., Zhang X. Efficient Scheduling in Training Deep Convolutional Networks at Large Scale // IEEE Access. 2018. Vol. 6. P. 61452–61456. DOI: 10.1109/ACCESS.2018.2875407.

153. Xiang S., Li H. On the Effects of Batch and Weight Normalization in Generative Adversarial Networks // CoRR. 2017. Vol. abs/2005.11074. arXiv: 1704.03971. URL: https://arxiv.org/abs/1704.03971.

154. Gitman I., Ginsburg B. Comparison of Batch Normalization and Weight Normalization Algorithms for the Large-scale Image Classification // CoRR. 2017. Vol. abs/1709.08145. arXiv: 1709.08145. URL: https://arxiv.org/abs/1709.08145.

155. Dukler Y., Gu Q., Montufar G. Optimization Theory for ReLU Neural Networks Trained with Normalization Layers // Proceedings of the International Conference on Machine Learning, Virtual, July 13–18, 2020. Proceedings of Machine Learning Research. Vol. 119 / ed. by H. Daumé III, A. Singh. PMLR, 2020. P. 2751–2760. URL: https://proceedings.mlr.press/v119/dukler20a.html.

156. Yu H., Yang S., Zhu S. Parallel Restarted SGD with Faster Convergence and Less Communication: Demystifying Why Model Averaging Works for Deep Learning // Proceedings of the AAAI Conference on Artificial Intelligence AAAI-19, Honolulu, HI, USA, January 27 – February 1, 2019. Vol. 33, no. 1. Palo Alto, CA, USA: AAAI Press, 2019. P. 5693–5700. DOI: 10.1609/aaai.v33i01.33015693.

157. Xu J., Wang J., Qi Q., *et al.* Effective Scheduler for Distributed DNN Training Based on MapReduce and GPU Cluster // Journal of Grid Computing. 2021. Vol. 19. P. 8. DOI: 10.1007/s10723-021-09550-6.

158. Si T.N., Van Hung T., Ngoc D.V., Le Q.N. Using Stochastic Gradient Descent On Parallel Recommender System with Stream Data // 2022 IEEE/ACIS 7th International Conference on Big Data, Cloud Computing, and Data Science (BCD), Danang, Vietnam, August 4–6, 2022. IEEE, 2022. P. 88–93. DOI: 10.1109/BCD54882.2022.9900664.

159. Sukanya J., Gandhi K.R., Palanisamy V. An assessment of machine learning algorithms for healthcare analysis based on improved MapReduce // Advances in Engineering Software. 2022. Vol. 173. P. 103285. DOI: 10.1016/j.advengsoft.2022.103285.

160. Asadianfam S., Shamsi M., Kenari A.R. TVD-MRDL: traffic violation detection system using MapReduce-based deep learning for large-scale data // Multimedia Tools and Applications. 2021. Vol. 80, no. 2. P. 2489–2516. DOI: 10.1007/s11042-020-09714-8.

161. Kul S., Sayar A. Sentiment Analysis Using Machine Learning and Deep Learning on Covid 19 Vaccine Twitter Data with Hadoop MapReduce // Proceedings of the 6th International Conference on Smart City Applications (SCA2021), Virtual Safranbolu, Turkey, October 27–29, 2021. Innovations in Smart Cities Applications Volume 5. Vol. 393 / ed. by M. Ben Ahmed, A.A. Boudhir, I.R. Karaș, *et al.* Springer, 2022. P. 859–868. Lecture Notes in Computer Science. DOI: 10.1007/978-3-030-94191-8_69.

162. Snir M., Otto S. W., Huss-Lederman S., *et al.* MPI: The complete reference: The MPI core. Cambridge, MA, USA: MIT Press, 1998. 427 p.

163. Thao Nguyen T., Wahib M., Takano R. Hierarchical Distributed-Memory Multi-Leader MPI-Allreduce for Deep Learning Workloads // 2018 Sixth International Symposium on Computing and Networking Workshops (CANDARW), Takayama, Japan, November 27–30, 2018. IEEE, 2018. P. 216–222. DOI: 10.1109/CANDARW.2018.00048.

164. Awan A.A., Bédorf J., Chu C.-H., *et al.* Scalable Distributed DNN Training using TensorFlow and CUDA-Aware MPI: Characterization, Designs, and Performance Evaluation // 2019 19th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID), Larnaca, Cyprus, May 14–17, 2019. IEEE, 2019. P. 498–507. DOI: 10.1109/CCGRID.2019.00064.

165. Bhagirath, Mittal N., Kumar S. Machine Learning Computation on Multiple GPU's using CUDA and Message Passing Interface // 2019 2nd International Conference on Power Energy, Environment and Intelligent Control (PEEIC), Greater Noida, India, October 18–19, 2019. IEEE, 2020. P. 18–22. DOI: 10.1109/PEEIC47157.2019.8976714.

166. Ghazimirsaeed S.M., Anthony Q., Shafi A., *et al.* Accelerating GPU-based Machine Learning in Python using MPI Library: A Case Study with MVAPICH2-GDR // 2020 IEEE/ACM Workshop on Machine Learning in High Performance Computing Environments (MLHPC) and Workshop on Artificial Intelligence and Machine Learning for Scientific Applications (AI4S), Virtual, November 9–19, 2020. IEEE, 2021. P. 1–12. DOI: 10.1109/MLHPCAI4S51975.2020.00010.

167. Linux Foundation. Distributed Data Parallel - torch.nn. 2022. URL: https://pytorch.org/docs/stable/generated/torch.nn.parallel.DistributedDataParallel.html (accessed: 01.10.2022).

168. Jia Z., Zaharia M., Aiken A. Beyond Data and Model Parallelism for Deep Neural Networks // Machine Learning and Systems (MLSys 2019), Stanford, CA, USA, March 31 – April 2, 2019. Proceedings of Machine Learning and Systems, Vol. 1 / ed. by A. Talwalkar, V. Smith, M. Zaharia. MLSYS, 2019. P. 1–13. URL: https://proceedings.mlsys.org/paper_files/paper/2019/hash/b422680f3db0986ddd7f8f126baaf0fa-Abstract.html.

169. Xu A., Huo Z., Huang H. On the Acceleration of Deep Learning Model Parallelism With Staleness // 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, June 13–19, 2020. IEEE, 2020. P. 2085–2094. DOI: 10.1109/CVPR42600.2020.00216.

170. Ericson L., Mbuvha R. On the Performance of Network Parallel Training in Artificial Neural Networks // CoRR. 2017. Vol. abs/1701.05130. arXiv: 1701.05130. URL: https://arxiv.org/abs/1701.05130.

171. Chen C.-C., Yang C.-L., Cheng H.-Y. Efficient and Robust Parallel DNN Training through Model Parallelism on Multi-GPU Platform // CoRR. 2018. Vol. abs/1809.02839. arXiv: 1809.02839. URL: https://arxiv.org/abs/1809.02839.

172. Bruna J., Zaremba W., Szlam A., LeCun Y. Spectral Networks and Locally Connected Networks on Graphs // 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14–16, 2014. Conference Track Proceedings / ed. by Y. Bengio, Y. LeCun. URL: http://arxiv.org/abs/1312.6203.

173. Chen Y.-h., Moreno I.L., Sainath T., *et al.* Locally-connected and convolutional neural networks for small footprint speaker recognition // Proceedings of 16th Annual Conference of the International Speech Communication Association (INTERSPEECH-2015), Dresden, Germany, September 6–10, 2015. ISCA, 2015. P. 1136–1140. DOI: 10.21437/Interspeech.2015-297.

174. Wadekar S.N. Locally connected neural networks for image recognition: PhD thesis / Wadekar Shakti Nagnath. Purdue University Graduate School, West Lafayette, IN, USA, 2019. URL: https://hammer.purdue.edu/articles/thesis/LOCALLY_CONNECTED_NEURAL_NETWORKS_FOR_IMAGE_RECOGNITION/11328404/1.

175. Ankile L.L., Heggland M.F., Krange K. Deep Convolutional Neural Networks: A survey of the foundations, selected improvements, and some current applications // CoRR. 2020. Vol. abs/2011.12960. arXiv: 2011.12960. URL: https://arxiv.org/abs/2011.12960.

176. Coates A., Huval B., Wang T., *et al.* Deep learning with COTS HPC systems // Proceedings of the 30th International Conference on Machine Learning, Atlanta, GA, USA, June 17–19, 2013. Proceedings of Machine Learning Research. Vol. 28, no. 3 / ed. by S. Dasgupta, D. McAllester. PMLR, 2013. P. 1337–1345. URL: https://proceedings.mlr.press/v28/coates13.html.

177. Gironés R.G., Salcedo A.M. Forward-backward parallelism in on-line backpropagation // International Work-Conference on Artificial and Natural Neural Networks, IWANN'99, Alicante, Spain, June 2–4, 1999. Proceedings, Volume II. Vol. 1607 / ed. by J. Mira, J.V. Sánchez-Andrés. Springer, 1999. P. 157–165. Lecture Notes in Computer Science. DOI: 10.1007/BFb0100482.

178. Xiao D., Yang C., Wu W. Efficient DNN training based on backpropagation parallelization // Computing. 2022. Vol. 104. P. 2431–2451. DOI: 10.1007/s00607-022-01094-1.

179. Zhang H., Dai Y., Li H., Koniusz P. Deep Stacked Hierarchical Multi-Patch Network for Image Deblurring // 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, June 15–20, 2019. IEEE, 2020. P. 5971–5979. DOI: 10.1109/CVPR.2019.00613.

180. Xiang Y., Kim H. Pipelined data-parallel CPU/GPU scheduling for multi-DNN real-time inference // 2019 IEEE Real-Time Systems Symposium (RTSS), Hong Kong, China, December 3–6, 2019. IEEE, 2020. P. 392–405. DOI: 10.1109/RTSS46320.2019.00042.

181. Shi S., Tang Z., Wang Q., *et al.* Layer-wise Adaptive Gradient Sparsification for Distributed Deep Learning with Convergence Guarantees // 24th European Conference on Artificial Intelligence (ECAI 2020), Santiago de Compostela, Spain, August 29 – September 8, 2020. Frontiers in Artificial Intelligence and Applications. Vol. 325 / ed. by G. De Giacomo, A. Catala, B. Dilkina, *et al.* Amsterdam, The Netherlands: IOS Press, 2020. P. 1467–1474. DOI: 10.3233/FAIA200253.

182. Harlap A., Narayanan D., Phanishayee A., *et al.* PipeDream: Fast and Efficient Pipeline Parallel DNN Training // CoRR. 2018. Vol. abs/1806.03377. arXiv: 1806.03377. URL: https://arxiv.org/abs/1806.03377.

183. Huang Y., Cheng Y., Bapna A., *et al.* GPipe: Efficient Training of Giant Neural Networks using Pipeline Parallelism // Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada, December 8–14, 2019. Advances in Neural Information Processing Systems. Vol. 32 / ed. by H. Wallach, H. Larochelle, A. Beygelzimer, *et al.* Curran Associates, Inc., 2019. URL: https://proceedings.neurips.cc/paper/2019/hash/093f65e080a295f8076b1c5722a46aa2-Abstract.html.

184. Krizhevsky A., Sutskever I., Hinton G.E. ImageNet Classification with Deep Convolutional Neural Networks // Communications of the ACM. 2017. Vol. 60, no. 6. P. 84–90. DOI: 10.1145/3065386.

185. Sun P., Feng W., Han R., *et al.* Optimizing Network Performance for Distributed DNN Training on GPU Clusters: ImageNet/AlexNet Training in 1.5 Minutes // CoRR. 2019. Vol. abs/1902.06855. arXiv: 1902.06855. URL: https://arxiv.org/abs/1902.06855.

186. Gaunt A.L., Johnson M.A., Riechert M., *et al.* AMPNet: Asynchronous Model-Parallel Training for Dynamic Neural Networks // CoRR. 2017. Vol. abs/1705.09786. arXiv: 1705.09786. URL: https://arxiv.org/abs/1705.09786.

187. Dean J., Corrado G., Monga R., *et al.* Large Scale Distributed Deep Networks // Conference on Neural Information Processing Systems (NIPS 2012), Lake Tahoe, NV, USA, December 3–8, 2012. Advances in Neural Information Processing Systems. Vol. 25 / ed. by F. Pereira, C. Burges, L. Bottou, K. Weinberger. Curran Associates, Inc., 2012. URL: https://proceedings.neurips.cc/paper/2012/hash/6aca97005c68f1206823815f66102863-Abstract.html.

188. Lee S., Jha D., Agrawal A., *et al.* Parallel Deep Convolutional Neural Network Training by Exploiting the Overlapping of Computation and Communication // 2017 IEEE 24th International Conference on High Performance Computing (HiPC), Jaipur, India, December 18–21, 2017. IEEE, 2018. P. 183–192. DOI: 10.1109/HiPC.2017.00030.

189. Hu Y., Liu Y., Liu Z. A Survey on Convolutional Neural Network Accelerators: GPU, FPGA and ASIC // 2022 14th International Conference on Computer Research and Development (ICCRD), Shenzhen, China, January 7–9, 2022. IEEE, 2022. P. 100–107. DOI: 10.1109/ICCRD54409.2022.9730377.

190. Kim J.-Y. Chapter Five - FPGA based neural network accelerators // Advances in Computers (Vol. 122): Hardware Accelerator Systems for Artificial Intelligence and Machine Learning. Elsevier, 2021. P. 135–165. DOI: 10.1016/bs.adcom.2020.11.002.

191. Mittal S., Vibhu. A survey of accelerator architectures for 3D convolution neural networks // Journal of Systems Architecture. 2021. Vol. 115. P. 102041. DOI: 10.1016/j.sysarc.2021.102041.

192. Omondi A.R., Rajapakse J.C. FPGA implementations of neural networks. Dordrecht, The Netherlands: Springer, 2006. 365 p.

193. Reagen B., Adolf R., Whatmough P., *et al.* Deep learning for computer architects. Synthesis Lectures on Computer Architecture. Morgan & Claypool Publishers, 2017. 123 p. DOI: 10.2200/S00783ED1V01Y201706CAC041.

194. Genc H., Kim S., Amid A., *et al.* Gemmini: Enabling Systematic Deep-Learning Architecture Evaluation via Full-Stack Integration // 2021 58th ACM/IEEE Design Automation Conference (DAC), San Francisco, CA, USA, December 5–9, 2021. IEEE, 2021. P. 769–774. DOI: 10.1109/DAC18074.2021.9586216.

195. Ding W., Huang Z., Huang Z., *et al.* Designing efficient accelerator of depthwise separable convolutional neural network on FPGA // Journal of Systems Architecture. 2019. Vol. 97. P. 278–286. DOI: 10.1016/j.sysarc.2018.12.008.

196. Hu Y.H., Kung S.-Y. Systolic Arrays // Handbook of Signal Processing Systems / ed. by S.S. Bhattacharyya, E.F. Deprettere, R. Leupers, J. Takala. Cham: Springer International Publishing, 2019. P. 939–977. DOI: 10.1007/978-3-319-91734-4_26.

197. Liu Z.-G., Whatmough P.N., Mattina M. Systolic Tensor Array: An Efficient Structured-Sparse GEMM Accelerator for Mobile CNN Inference // IEEE Computer Architecture Letters. 2020. Vol. 19, no. 1. P. 34–37. DOI: 10.1109/LCA.2020.2979965.

198. Chen K.-C., Ebrahimi M., Wang T.-Y., Yang Y.-C. NoC-Based DNN Accelerator: A Future Design Paradigm // NOCS '19: Proceedings of the 13th IEEE/ACM International Symposium on Networks-on-Chip, New York, NY, USA, October 17–18, 2019. New York: ACM, 2019. P. 1–8. DOI: 10.1145/3313231.3352376.

199. Sun X., Choi J., Chen C.-Y., *et al.* Hybrid 8-bit floating point (HFP8) training and inference for deep neural networks // Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada, December 8–14, 2019. Advances in Neural Information Processing Systems. Vol. 32 / ed. by H. Wallach, H. Larochelle, A. Beygelzimer, *et al.* Curran Associates, Inc., 2019. P. 4900–4909. URL: https://proceedings.neurips.cc/paper/2019/hash/65fc9fb4897a89789352e211ca2d398f-Abstract.html.

200. Jia X., Song S., He W., *et al.* Highly Scalable Deep Learning Training System with Mixed-Precision: Training ImageNet in Four Minutes // CoRR. 2018. Vol. abs/1807.11205. arXiv: 1807.11205. URL: https://arxiv.org/abs/1807.11205.

201. Yang J.A., Huang J., Park J., *et al.* Mixed-Precision Embedding Using a Cache // CoRR. 2020. Vol. abs/2010.11305. arXiv: 2010.11305. URL: https://arxiv.org/abs/2010.11305.

202. Courbariaux M., Hubara I., Soudry D., *et al.* Binarized Neural Networks: Training Deep Neural Networks with Weights and Activations Constrained to +1 or -1 // CoRR. 2016. Vol. abs/1602.02830. arXiv: 1602.02830. URL: https://arxiv.org/abs/1602.02830.

203. Nour B., Cherkaoui S. How Far Can We Go in Compute-less Networking: Computation Correctness and Accuracy // IEEE Network. 2022. Vol. 36, no. 4. P. 197–202. DOI: 10.1109/MNET.012.2100157.

Ронкин Михаил Владимирович, к.т.н., доцент, кафедра информационных технологий и систем управления, Уральский федеральный университет (Екатеринбург, Российская Федерация)

Акимова Елена Николаевна, д.ф.-м.н., профессор, кафедра информационных технологий и систем управления, Уральский федеральный университет; в.н.с., отдел некорректных задач анализа и приложений, Институт математики и механики им. Н.Н. Красовского Уральского отделения Российской академии наук (Екатеринбург, Российская Федерация)

Мисилов Владимир Евгеньевич, к.ф.-м.н., доцент, кафедра информационных технологий и систем управления, Уральский федеральный университет; н.с., отдел некорректных задач анализа и приложений, Институт математики и механики им. Н.Н. Красовского Уральского отделения Российской академии наук (Екатеринбург, Российская Федерация)

Решетников Кирилл Игоревич, аспирант, кафедра информационных технологий и систем управления, Уральский федеральный университет (Екатеринбург, Российская Федерация)

DOI: 10.14529/cmse230401

REVIEW ON APPLICATION OF DEEP NEURAL NETWORKS AND PARALLEL ARCHITECTURES FOR ROCK FRAGMENTATION PROBLEMS

© 2023 M.V. Ronkin[1], E.N. Akimova[1,2], V.E. Misilov[1,2], K.I. Reshetnikov[1]

[1] *Ural Federal University (Mira Street 19, Ekaterinburg, 620002 Russia),*

[2] *Krasovskii Institute of Mathematics and Mechanics, Ural Branch of RAS (S. Kovalevskaya Street 16, Ekaterinburg, 620108 Russia)*

E-mail: m.v.ronkin@urfu.ru, aen15@yandex.ru, v.e.misilov@urfu.ru, ki.reshetnikov@urfu.ru

Received: 14.07.2023

Evaluation of mining productivity, including the determination of the geometric dimensions of rock objects in an open pit, is one of the most critical tasks in the mining industry. The problem of rock fragmentation is usually solved using computer vision methods such as instance segmentation or semantic segmentation. Today, deep learning neural networks are used to solve such problems for digital images. Neural networks require a lot of computing power to process high-resolution digital images and large datasets. To address this issue, in literature, lightweight architectural neural networks are proposed, as well as parallel computing using CPU, GPU, and specialized accelerators. The review discusses the latest advances in the field of deep learning neural networks for solving computer vision problems in relation to rock fragmentation and aspects of improving the performance of neural network implementations on various parallel architectures.

Keywords: computer vision, convolutional neural networks, deep learning, instance segmentation, semantic segmentation, object detection, parallel computing, mining industry problems, rock fragmentation.

FOR CITATION

Ronkin M.V., Akimova E.N., Misilov V.E., Reshetnikov K.I. Review on Application of Deep Neural Networks and Parallel Architectures for Rock Fragmentation Problems. Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering. 2023. Vol. 12, no. 4. P. 5–54. (in Russian) DOI: 10.14529/cmse230401.