Федеральное государственное автономное образовательное учреждение
высшего образования «Национальный исследовательский университет
«Высшая школа экономики»

На правах рукописи

Рябинин Максим Константинович

МЕТОДЫ И ПРОБЛЕМЫ ДЕЦЕНТРАЛИЗОВАННОГО ГЛУБИННОГО ОБУЧЕНИЯ

РЕЗЮМЕ
диссертации на соискание ученой степени
кандидата компьютерных наук

Москва – 2023

Диссертационная работа выполнена в федеральном государственном автономном образовательном учреждении высшего образования «Национальный исследовательский университет «Высшая школа экономики».

Научный руководитель: Бабенко Артем Валерьевич, к.ф.-м.н., Национальный исследовательский университет «Высшая школа экономики».

1 Введение

Тема диссертации

За последнее десятилетие глубинное обучение продемонстрировало заметные результаты, превзойдя другие методы машинного обучения в различных задачах и областях. Также в последние годы наблюдается резкий рост размеров нейронных сетей из-за значительного влияния масштаба модели на ее итоговые возможности [45; 46]. Этот рост является проблемой с точки зрения широкого научного сообщества: по мере того, как ресурсы, необходимые для воспроизведения результатов или улучшения самых современных моделей, продолжают расти, исследования в области глубинного обучения становятся все менее доступными для всех организаций, кроме имеющих наибольшее финансирование. В данной работе мы предлагаем *децентрализацию* как потенциальное решение этой проблемы: вместо обучения на одном централизованном кластере мы можем использовать свободные ресурсы добровольцев, которые потенциально распределены по всему миру. Вдохновляясь успехами добровольных вычислений в других научных областях [7; 20; 48], мы предлагаем методы глубинного обучения, которые применимы для широкого класса моделей и задач и при этом учитывают уникальные проблемы добровольных вычислений.

Более конкретно, в данной работе мы представляем слой *децентрализованной смеси экспертов* — разреженную нейросетевую архитектуру, которая отвечает вышеуказанным задачам и естественным образом справляется как с отказами узлов, так и с большим количеством нерегулярно участвующих в обучении узлов. Далее мы рассматриваем вопрос обучения с волонтерами в условиях параллелизма по данным: такая постановка требует метод, который может быстро агрегировать параметры или градиенты модели в условиях сбоев в сети. Для этого мы разработали *Moshpit All-Reduce* — эффективный отказоустойчивый метод усреднения параметров. Используя этот метод, мы предлагаем *Moshpit SGD* — распределенный алгоритм обучения, который может быть применен к сетям из разнородных и ненадежных устройств. Наконец, мы предлагаем *Distributed Deep Learning in Open Collaborations* — практичный и масштабируемый подход к совместному предобучению нейросетей. Этот подход сочетает в себе адаптивную стратегию усреднения, глобальную аккумуляцию градиентов и проектирование специализированной системы с учетом постановки, что позволяет проводить распределенное обучение с узлами, у которых значительно варьируются параметры подключения сети, вычислительная производительность и время участия в обучении.

Актуальность работы

Рост размеров моделей лежит в основе многих последних достижений в глубоком обучении. Последние модели, достигающие наивысшего качества, регулярно достигают масштабов в десятки и сотни миллиардов параметров [27; 35]: эти разработки поддерживаются исследованиями [46], которые демонстрируют растущее качество или даже новые свойства [18] нейронных сетей при увеличении их размеров. Похожим образом растет и размер обучающих выборок: как показывают последние работы [54], количество примеров может быть столь же важным, как и размер модели при обучении нейронной сети с фиксированным бюджетом на вычисления. Оба эти направления роста требуют огромного количества вычислительных ресурсов: все современные модели обучаются на суперкомпьютерах с сотнями или даже тысячами специализированных ускорителей и сетевыми соединениями с высокой скоростью.

Ожидаемо, получение доступа к вычислительным ресурсам для обучения таких больших моделей является сложной задачей для рядового исследователя. Аренда даже одного ускорителя обучения нейросетей за месяц может стоить тысячи долларов, а построение кластера выходит за бюджетные рамки для организаций с небольшим финансированием. Это явление ограничивает доступность самых современных исследований тем кругом лабораторий, которые могут позволить себе проводить масштабные эксперименты с нейронными сетями миллиардных размеров. В свою очередь, это приводит к снижению возможностей по воспроизведению или адаптации последних результатов к новым данным, невозможности проанализировать или улучшить процесс обучения больших моделей, а также к общим трудностям в содействии дальнейшему научному прогрессу в области глубинного обучения.

В настоящей работе мы исследуем альтернативный подход к крупномасштабному глубинному обучению, не требующиий использования дорогостоящих кластеров. Мы черпали вдохновение в успешном применении ресурсов добровольцев в других науках, таких как вычислительная биология [20] или астрофизика [17]. Наиболее известным примером таких проектов является платформа Berkeley Open Infrastructure for Network Computing (BOINC) [7], которая стала первым «суперкомпьютером», достигшим производительности в экзафлоп [19]. В то же время, прямое применение существующих методов распределенного глубинного обучения в подобных условиях затруднено из-за многочисленных проблем инфраструктурного характера.

В частности, наиболее популярные методы эффективного распределенного обучения [22; 40; 47] не рассчитаны на работу в условиях отказов серверов или соединений между ними: в отдельных случаях даже один отключившийся участник может прервать всю процедуру обучения или значительно замедлить ее ход. В то же время, узлы в си-

стеме добровольных вычислений обладают гораздо большей степенью неоднородности: каждый персональный компьютер может иметь уникальную аппаратную и сетевую конфигурацию, и это разнообразие необходимо учитывать при разработке систем децентрализованного обучения. Наконец, скорость передачи данных между узлами кластера может быть в разы выше, чем скорость стандартных интернет-соединений участников коллаборативного обучения, что также влияет на проектирование итоговой системы. Таким образом, мы разрабатываем методы, направленные на максимизацию эффективности распределенного обучения в условиях, описанных выше.

Первая работа, описанная в данной диссертации, посвящена обучению моделей, превосходящих объем памяти отдельного участника в децентрализованном контексте. Жертвуя частью общности подхода в обмен на производительность, эта работа представляет *децентрализованную смесь экспертов* (DMoE) — специализированный слой, разработанный для разделения по компьютерам добровольцев. Подобно стандартным моделям смесей экспертов [2], слой DMoE состоит из независимых подслоёв, называемых *экспертами* и назначаемых для конкретного входа на основе результатов работы *гейтинг-функции*. Мы предлагаем естественное расширение этой архитектуры для отказоустойчивого обучения и показываем, что DMoE не чувствителен к сетевым задержкам. Другое важное отличие заключается в том, что эксперты DMoE находятся другими узлами сети с помощью распределенных хэш-таблиц (DHT) — отказоустойчивого децентрализованного ассоциативного массива. Это снижает необходимость в централизованной структуре данных для хранения доступных экспертов, что может быть невыполнимо в больших коллаборациях без значительных затрат. Для эффективного поиска наиболее релевантных экспертов для заданных входных данных мы предлагаем *структурированную гейтинг-функцию*, которая факторизует множество экспертов в виде заранее определенной многомерной сетки.

В следующей части диссертационной работы рассматривается проблема параллельного по данным обучения с использованием ресурсов добровольцев. Мы рассматриваем данную проблему по двум причинам: во-первых, даже при использовании смеси экспертов в каждом слое модели нам все равно необходимо иметь согласованные по всей коллаборации параметры гейтинг-функции и слоя представлений входных данных. Во-вторых, с помощью эффективных по памяти методов обучения (таких как пониженная численная точность [32] или переиспользование параметров в разных слоях [3]) можно обучать модели, которые могут быть вмещены в потребительские GPU, но при этом все еще требуют больших объемов вычислений для достижения наилучшего качества.

Во второй работе, рассматриваемой в данной диссертации, мы изучаем методы эффективной агрегации градиентов модели для распределенного обучения. Семейство коммуникационно-оптимальных методов, известное как All-Reduce [37], не является от-

казоустойчивым без модификаций и поэтому не подходит для наших целей. С другой стороны, более надежные методы децентрализованного обучения, такие как Gossip [49; 52], требуют много итераций передачи данных для достижения согласованности в сети. Мы предлагаем *Moshpit All-Reduce* — итеративный алгоритм усреднения, который сочетает в себе отказоустойчивость Gossip и эффективность All-Reduce. Он объединяет участников в независимые группы и гарантирует, что сверстники в одной группе будут назначены в разные группы на следующем шаге. *Moshpit SGD* (распределенный алгоритм оптимизации, основанный на Moshpit All-Reduce) имеет скорость сходимости, эквивалентную стандартному распределенному стохастическому градиентному спуску (конкретнее, Local-SGD [51]), но при этом демонстрирует в наших экспериментах гораздо более высокую производительность при обучении в медленных сетях с отказами узлов.

Наконец, третья работа представляет *Distributed Deep Learning in Open Collaborations* (DeDLOC) — подход, учитывающий неоднородность узлов и снижающий влияние низкой скорости связи в распределенном обучении с волонтерами. В частности, мы предлагаем адаптивную стратегию усреднения, которая распределяет задачи обучения и агрегации градиентов между узлами на основе их производительности, чтобы минимизировать общее время усреднения — основной фазы коммуникации в параллельном по данным обучении. Мы также разработали децентрализованный механизм отслеживания общего размера накопленного минибатча, который необходим для серверов с динамическим участием. Помимо контрольных экспериментов, в статье представлены результаты первого коллаборативного эксперимента по предварительному обучению языковых моделей: усилия, организованные авторами работы вместе с сообществом добровольцев, привели к созданию *sahajBERT*, маскированной языковой модели для бенгальского языка, показывающей сопоставимые с одноязычными и многоязычными [25; 56] аналогами результаты.

Отметим, что разработанные нами методы могут применяться не только в сценарии добровольных вычислений. В частности, провайдеры облачных вычислений часто предлагают *вытесняемые* узлы по цене, которая может быть в 2–3 раза ниже, чем стоимость стандартных серверов [5; 21]. Однако вытесняемые узлы имеют недостаток в виде негарантированной доступности: если спрос на серверы с такой же аппаратной конфигурацией увеличивается, некоторые из этих узлов могут стать недоступными, пока спрос не снизится. Такая особенность делает использование традиционных систем распределенного обучения невозможным. Обычно эффективное обучение зависит от надежного времени работы и высокой скорости соединения, чего трудно достичь в описанной выше постановке стандартными методами. Тем не менее, целевой сценарий данной диссертации учитывает большинство проблем, возникающих при использовании вытесняемых узлов. Как будет показано в экспериментальной части далее, предложенные методы могут быть

применены как к неоднородному оборудованию добровольцев, так и к более однородным, но все еще нестабильным вытесняемым облачным серверам.

Целью данной работы является разработка практически применимых методов распределенного обучения для сетей с низкой скоростью соединения, состоящих из разнородных и ненадежных узлов.

2 Основные результаты и выводы

Вклад работы может быть резюмирован следующим образом:

1. Предложен механизм *децентрализованной смеси экспертов* (DMoE) — слой нейронных сетей, разработанный для обучения больших моделей в условиях добровольных вычислений. Чтобы справиться с отказами узлов и низкой скоростью соединения в единой конструкции, мы также разработали *Learning@home* — систему для крупномасштабного обучения в условиях добровольных вычислений. Мы эмпирически подтвердили производительность Learning@home в условиях сетевой задержки и сходимость моделей DMoE при наличии сбоев узлов к тем же результатам, что и у эквивалентных неразреженных нейросетей.

2. Предложен *Moshpit All-Reduce*, эффективный децентрализованный метод усреднения градиентов, а также *Moshpit SGD*, распределенный алгоритм оптимизации, который использует Moshpit All-Reduce, имеет эквивалентную Local-SGD скорость сходимости и подходит для обучения на ненадежных устройствах. В крупномасштабных экспериментах по обучению нейросетевых моделей Moshpit SGD превосходит существующие методы (включая предыдущие децентрализованные подходы) более чем на 30% по времени сходимости к целевому значению функции потерь.

3. Предложен *Distributed Deep Learning in Open Collaborations* — практический метод совместного параллельного по данным глубинного обучения. Этот метод устойчив к подключению и отключению узлов во время обучения и учитывает разнообразие аппаратных и сетевых условий. Мы также провели эксперимент с совместным обучением в реальных условиях, получив *sahajBERT*, нейронную сеть для представлений бенгальского языка с результатами, сравнимыми с обычными моделями, обученными на кластерах, стоящими в ≈ 3 раза больше.

Теоретическая и практическая значимость. В диссертации предлагается набор методов, которые могут быть применены для масштабируемого обучения нейронных сетей в децентрализованной постановке с нестабильными связями между разнородными и периодически доступными узлами. Такие условия могут возникать в двух случаях: первый из

них — совместное обучение (среди нескольких организаций или просто добровольцев), второй — эффективное по стоимости обучение на вытесняемых узлах. Исследование, представленное в данной работе, направлено на то, чтобы сделать эти две экономически эффективных альтернативы кластерам более применимыми и распространенными. Все представленные методы имеют публично доступные реализации с открытым исходным кодом на PyTorch [41], одной из самых популярных библиотек для глубинного обучения на момент написания работы. Это позволяет любому специалисту в области глубинного обучения применять описанные методы и организовывать эксперименты с децентрализованным обучением.

Результаты, выносимые на защиту.

1. Слой *распределенной смеси экспертов* для обучения больших нейронных сетей с использованием ресурсов добровольцев.

2. Протокол *Moshpit All-Reduce* и алгоритм *Moshpit SGD* для эффективного по коммуникации обучения на нестабильных устройствах.

3. *DeDLOC*, подход для параллельного по данным предобучения в больших коллаборациях, состоящих из узлов с варьирующимися мощностями.

Личный вклад в результаты, выносимые на защиту.

В работе «Towards Crowdsourced Training of Large Neural Networks using Decentralized Mixture-of-Experts» автором диссертации разработана основная идея подхода, реализованы ключевые компоненты Learning@home, проведены все эксперименты и написана большая часть статьи (за исключением описания распределенной хэш-таблицы).

В работе «Moshpit SGD: Communication-Efficient Decentralized Training on Heterogeneous Unreliable Devices» автор предложил метод Moshpit All-Reduce, провел большую часть экспериментов по усреднению в разделе 4.1 и все эксперименты по обучению нейронных сетей, доказал теорему C.1 и предложил подход к балансированию нагрузки в приложении G (обобщенный в следующей статье Михаилом Дискиным).

В работе «Distributed Deep Learning in Open Collaborations» автор разработал ключевую идею исследования и схему проведения научного проекта, реализовал изначальный код для предварительного обучения модели ALBERT, провел эксперимент по совместному обучению sahajBERT и написал большую часть статьи.

Публикации и апробация работы

* обозначает равный вклад соавторов

Публикации повышенного уровня

1. **Максим Рябинин***, Антон Гусев. Краудсорсинговое обучение больших нейронных сетей с использованием децентрализованных смесей экспертов (Towards Crowdsourced Training of Large Neural Networks using Decentralized Mixture-of-Experts). В материалах конференции Neural Information Processing Systems, 2020 (NeurIPS 2020). Стр. 3659–3672. Конференция ранга A* по рейтингу CORE.

2. **Максим Рябинин***, Эдуард Горбунов*, Всеволод Плохотнюк, Геннадий Пехименко. Moshpit SGD: коммуникационно-эффективное децентрализованное обучение на гетерогенных и ненадёжных устройствах (Moshpit SGD: Communication-Efficient Decentralized Training on Heterogeneous Unreliable Devices). В материалах конференции Neural Information Processing Systems, 2021 (NeurIPS 2021). Стр. 18195–18211. Конференция ранга A* по рейтингу CORE.

3. Михаил Дискин*, Алексей Бухтияров*, **Максим Рябинин***, Люсиль Солнье, Кентан Лоэст, Антон Синицин, Дмитрий Попов, Дмитрий Пыркин, Максим Каширин, Александр Борзунов, Альберт Вилланова дел Морал, Денис Мазур, Илья Кобелев, Ясин Жернит, Томас Вулф, Геннадий Пехименко. Распределенное глубинное обучение в открытых коллаборациях (Distributed Deep Learning In Open Collaborations). В материалах конференции Neural Information Processing Systems, 2021 (NeurIPS 2021). Стр. 7879–7897. Конференция ранга A* по рейтингу CORE.

Доклады на конференциях и семинарах

1. Приглашенный доклад на тему «Learning@home: Краудсорсинговое обучение больших нейронных сетей с использованием децентрализованных смесей экспертов». Семинар по обучению с подкреплением в Техническом Университете Эйндховена (проводился дистанционно), 30 апреля 2020 года.

2. Постерный доклад на тему «Краудсорсинговое обучение больших нейронных сетей с использованием децентрализованных смесей экспертов». Конференция по нейронным системам обработки информации (NeurIPS, проводилась дистанционно), 6 декабря 2020 года.

3. Семинар факультета компьютерных наук НИУ ВШЭ в Вороново. Тема: «Децентрализованное глубинное обучение». 29 мая 2021 года.

4. Приглашенный доклад на тему «Децентрализованное глубинное обучение: совместное обучение больших нейронных сетей». Второй симпозиум по распределенному машинному обучению (DistributedML, проводился дистанционно), 7 декабря 2021 года.

5. Постерный доклад на тему «Moshpit SGD: коммуникационно-эффективное децентрализованное обучение на гетерогенных и ненадёжных устройствах». Конференция по нейронным системам обработки информации (NeurIPS, проводилась дистанционно), 10 декабря 2021 года.

6. Постерный доклад на тему «Распределенное глубинное обучение в открытых коллаборациях». Конференция по нейронным системам обработки информации (NeurIPS, проводилась дистанционно), 10 декабря 2021 года.

7. Семинар Vector Institute «Ключевые моменты конференции NeurIPS» (дистанционно). Тема «Moshpit SGD: коммуникационно-эффективное децентрализованное обучение на гетерогенных и ненадёжных устройствах». 16 февраля 2022 г.

8. Приглашенный доклад на тему «Децентрализованное глубинное обучение: обучение и применение больших моделей по интернету» (проводился дистанционно). DeepMind, 12 декабря 2022 года.

9. Приглашенный доклад на тему «Децентрализованное глубинное обучение: обучение и применение больших моделей по интернету». Швейцарская высшая техническая школа Цюриха, 20 декабря 2022 года.

10. Приглашенный доклад на тему «Децентрализованное глубинное обучение: обучение и применение больших моделей по интернету». Naver Labs Europe, 7 февраля 2023 года.

11. Приглашенный доклад на тему «Децентрализованное глубинное обучение: обучение и применение больших моделей по интернету». Институт науки и технологий Австрии, 21 марта 2023 года.

Объем и структура работы. Диссертация содержит введение, содержание публикаций и заключение. Полный объем диссертации составляет 132 страницы.

Автор также внес свой вклад в следующие публикации:

1. Александр Борзунов*, **Максим Рябинин***, Тим Деттмерс*, Кентан Лоэст*, Люсиль Солнье*, Михаил Дискин, Ясин Жернит, Томас Вулф. Совместное обучение нейросетей «Трансформер» (Training Transformers Together). В материалах программы

Competitions and Demonstrations Track конференции Neural Information Processing Systems, 2021. Стр. 335–342.

2. Эдуард Горбунов*, Александр Борзунов*, Михаил Дискин, **Максим Рябинин**. Безопасное распределенное обучение на больших масштабах (Secure Distributed Training at Scale). В материалах конференции International Conference on Machine Learning, 2022 (ICML 2022). Стр. 7679–7739.

3 Содержание работы

3.1 Краудсорсинговое обучение больших нейронных сетей с использованием децентрализованных смесей экспертов

Децентрализованная смесь экспертов

Для обучения больших моделей в условиях добровольных вычислений мы предлагаем децентрализованную смесь экспертов (DMoE) — прямое расширение стандартных слоев смесей экспертов (MoE) [2; 34] для децентрализованного глубинного обучения. Каждый слой DMoE содержит несколько *экспертов* — независимых параллельных блоков, которые имеют одинаковую архитектуру, но разные параметры. На прямом проходе через модель каждый входной пример направляется к наиболее релевантным экспертам с помощью *гейтинг-функции*, которая является обучаемым классификатором, определяющим приоритет эксперта для входных активаций. Как и обычная смесь экспертов, DMoE может обрабатывать входные данные произвольного типа при использовании соответствующих типов слоев в качестве экспертов. Ключевое отличие децентрализованной смеси экспертов заключается в том, что эксперты распределяются по сети компьютеров добровольцев в соответствии с ограничениями памяти каждого устройства.

Для обмена необходимыми метаданными по сети мы используем распределенные хэш-таблицы (DHT, [11]) — распределенную структуру данных «ключ-значение», которая не требует централизованной координации, устойчива к отказам серверов и имеет логарифмическую сложность операций относительно числа узлов. Эти свойства сделали DHT популярным выбором для надежного хранения данных в одноранговых (peer-to-peer) приложениях; мы используем Kademlia [31] как один из самых популярных протоколов DHT с публичными реализациями. Для децентрализованной смеси экспертов DHT хранит метаданные экспертов, такие как статус соответствующего эксперту узла и его местоположение в сети. Пример прямого и обратного прохода DMoE приведен на рис. 1.

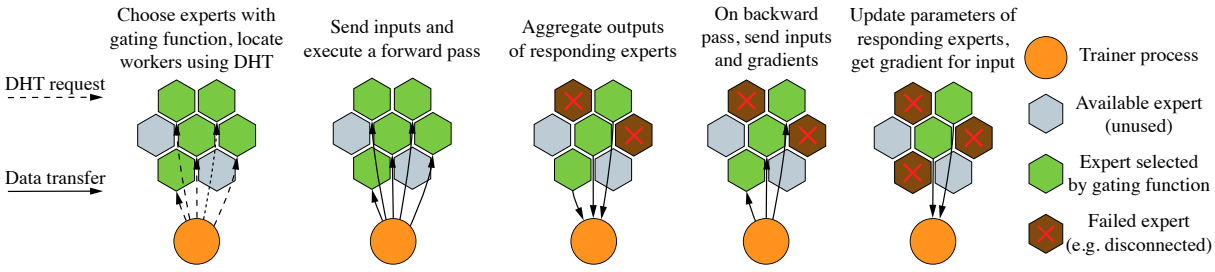


Рис. 1: Обработка одного примера слоем DMoE с отказами нескольких узлов.

Заметим, что использование слоев DMoE позволяет решить ряд проблем, возникающих в постановке добровольных вычислений. Во-первых, если запрошенный эксперт не отвечает, DMoE сможет исключить его из усреднения путем перенормировки весов остальных экспертов. Во-вторых, серверы с разным объемом памяти могут вмещать разное количество экспертов. Помимо этого, увеличение разнообразия между экспертами для их специализации [34] служит естественным способом балансировки нагрузки между различными серверами. Наконец, разреженная природа моделей смеси экспертов делает их менее чувствительными к устаревшим градиентам, возникающим при асинхронном обучении, что позволяет увеличить пропускную способность обучения в случае высокой задержки сети. Строго говоря, если веса нейронной сети были изменены при шаге оптимизации, то градиенты ее параметров для предыдущих шагов, вычисленные на других узлах, становятся некорректными. Наличие независимо обновляемых наборов весов для каждого эксперта уменьшает степень пересечения множеств изменяемых параметров.

Для эффективного поиска наиболее приоритетных экспертов в большой сети добровольцев мы предлагаем структурированную гейтинг-функцию, схожую по своей структуре со слоями факторизованных ключей [29]. Эта функция работает с набором экспертов, организованных в сетку из $d \times M$ элементов: поскольку мы ожидаем, что со временем к ней будут присоединяться новые участники, она должна быть достаточно большой и иметь пустые позиции для распределения будущих экспертов. Важно, что эта структура сетки связывает каждого эксперта f с его уникальным идентификатором, позволяющим кодировать M^d экспертов в Md элементах:

$$\text{uid}(f) = (u_0, u_1, \ldots, u_{d-1}), u_i \in [0, M). \tag{1}$$

При расположении экспертов таким образом гейтинг-функция $g(x, f)$ должна предсказать только Md значений для приоритетов экспертов. Более конкретно, эта функция состоит из d линейных слоев, предсказывающих M значений, а приоритет каждого эксперта вычисляется как сумма значений в сетке, соответствующих его идентификатору:

$$g(x, f) = \sum_{i=1}^{d} g_i(x)_{u_i},\ u_i \in \text{uid}(f). \tag{2}$$

В результате мы можем выбрать k наилучших экспертов за $O(dk \log N)$ времени (N — общее число пиров) приблизительным образом, используя алгоритм лучевого поиска. Сначала мы предсказываем приоритеты для всех измерений сетки, а затем проходим по ней, увеличивая i от 1 до d. На каждой итерации сохраняется список из k префиксов идентификаторов с наивысшим приоритетом, что дает нам ответ после окончания обхода. Для каждого измерения мы сначала расширяем список кандидатов, находя всех экспер-

тов с заданными префиксами, существующих в DHT, а затем сохраняем k продолжений с наивысшим приоритетом в качестве списка кандидатов.

Когда наиболее релевантные эксперты найдены, мы находим соответствующие им серверы с помощью DHT и отправляем им вектор входных активаций. После того как запрошенные эксперты возвращают выходы для этих векторов, мы усредняем их ответы с весами, определяемыми гейтинг-функцией:

$$\text{DMoE}(x) = \sum_{f' \in \text{TopK}(x)} f(x) \frac{\exp\left(g(x, f)\right)}{\sum_{f' \in \text{TopK}(x)} \exp\left(g(x, f')\right)}, \text{ TopK}(x) \text{ — } k \text{ лучших экспертов с т.з. } g. \tag{3}$$

Learning@home

Для поддержки обучения больших моделей на оборудовании волонтеров мы также разработали Learning@home — инфраструктуру, позволяющую эффективно производить обучение в условиях сбоев узлов и низкой скорости сети. Она состоит из трех основных компонентов:

- **Trainer** генерирует микробатчи данных и выполняет прямые и обратные проходы через нейронную сеть.
- **Runtime** обеспечивает доступ к экспертам и обрабатывает входящие запросы на прямой и обратный проходы (включая шаг оптимизации в случае обратного прохода). Чтобы максимизировать пропускную способность, Runtime группирует входящие запросы одного типа.
- **DHT**-узел сообщает о местоположении экспертов и обменивается их метаданными с другими узлами.

Эмпирические результаты

В первом наборе экспериментов мы оцениваем пропускную способность (количество примеров, обрабатываемых в секунду) Learning@home в различных сетевых условиях. Мы эмулируем распределенную среду обучения, расположив большие полносвязные слои и блоки энкодеров с Transformer-архитектурой [10] на 4 GPU и имитируя сетевые задержки различной степени. Нашим базовым уровнем производительности является неасинхронное параллельное обучение модели [22]; в качестве верхней границы производительности мы используем результат обучения без сетевых задержек. Как видно из рис. 2, производительность нашего подхода остается неизменной даже при высокой задержке,

что подтверждает его полезность в условиях добровольных вычислений. Мы также провели сравнение в более реалистичной ситуации с тремя облачными GPU-серверами в разных регионах: в таблице 1 показаны результаты, схожие с результатами симуляций, что подтверждает наши выводы.

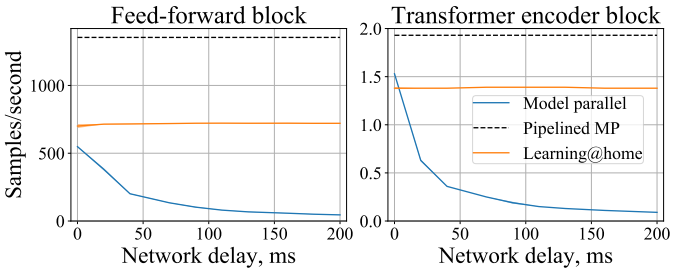


Рис. 2: Пропускная способность обучения в зависимости от сетевой задержки.

Метод	Полносвязная	Transformer
Model-parallel	7.23 ± 0.06	0.01 ± 0.0
Learning@home	300.8 ± 15.9	0.68 ± 0.0

Таблица 1: Пропускная способность обучения для 3 серверов в разных регионах.

Вторая серия экспериментов проверяет устойчивость DMoE к асинхронным обновлениям и устареванию градиентов. Мы сравниваем DMoE с последовательностью слоев, имеющих те же вычислительные затраты на прямой проход, что и подмножество из 4 экспертов. Для оценки различных сетевых условий мы рассматриваем три сценария: низкая задержка (100 мс задержки для каждого запроса), высокая задержка (1000 мс задержки) и высокая задержка с отказами (вероятность отсутствия ответа на запрос 0.1).

Мы обучаем модели на двух наборах данных: последовательность полносвязных слоев на MNIST [23] и Transformer-XL [55] на WikiText-2 [50]. В случае с WikiText-2 мы рассматриваем только постановку высокой задержки с отказами. Результаты показаны на рис. 3 и рис. 4: как видно из графиков, устаревание градиентов, вносимое асинхронным обучением в условиях высокой задержки, влияет на DMoE в меньшей степени по сравнению с обычными моделями.

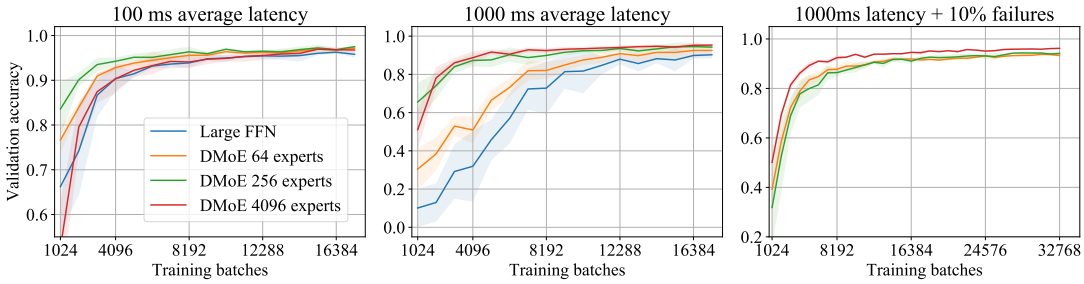


Рис. 3: Сходимость полносвязных моделей, обученных на MNIST с различными задержками сети и частотой отказов. Светлые области показывают стандартное отклонение по 5 запускам.

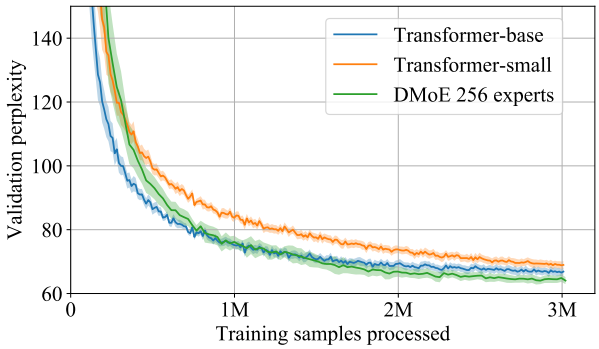


Рис. 4: Сходимость языковых моделей Transformer, обученных на наборе данных WikiText-2. Светлые области показывают стандартное отклонение по 5 запускам.

3.2 Moshpit SGD: коммуникационно-эффективное децентрализованное обучение на гетерогенных и ненадёжных устройствах

Для проведения *параллельного по данным* распределенного обучения (когда все участники обрабатывают разные примеры) необходимо иметь алгоритм для агрегации обновлений по сети. Например, в случае децентрализованной смеси экспертов, параметры слоя входных представлений и гейтинг-функции по-прежнему являются общими между участниками. В простейшей форме обучение нейронной сети сводится к стохастической оптимизационной задаче минимизации ожидаемой функции потерь $f(x, \theta)$, зависящей от входов x и параметров модели θ, по θ:

$$\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} f(x_i, \theta) \tag{4}$$

По этой причине основным шагом коммуникации в параллельном по данным глубинном обучении является усреднение градиентов $\nabla_\theta f(x_i, \theta)$ по узлам сети. Следовательно, для обучения с ненадежными участниками требуется эффективный с точки зрения передачи данных алгоритм, который был бы устойчив к сбоям узлов. Однако большинство широко используемых методов обладают только одним из этих свойств: All-Reduce теоретически оптимален с точки зрения передачи данных по сети [37], но не отказоустойчив, а подходы на основе подхода Gossip [13; 43; 52] к децентрализованному обучению реализуют обмен параметрами по разреженному графу, избегая необходимости одновременной синхронизации по всей сети, но передавая данные менее эффективно. Мы предлагаем Moshpit All-Reduce — метод усреднения для децентрализованного глубокого обучения, который является высокоэффективным и при этом отказоустойчивым.

Moshpit All-Reduce

Выполнение алгоритму Moshpit All-Reduce сводится к нескольким итерациям усреднения в небольших динамически изменяющихся группах, которые не пересекаются в рамках одной итерации. В этом случае отказ одного участника приводит к ошибке только в небольшой подгруппе сети. Каждая группа состоит из узлов, имеющих одну и ту же координату в виртуальной d-мерной сетке с M элементами в каждом измерении. На практике M и d выбираются так, чтобы вместить всех участников с относительно высоким заполнением сетки.

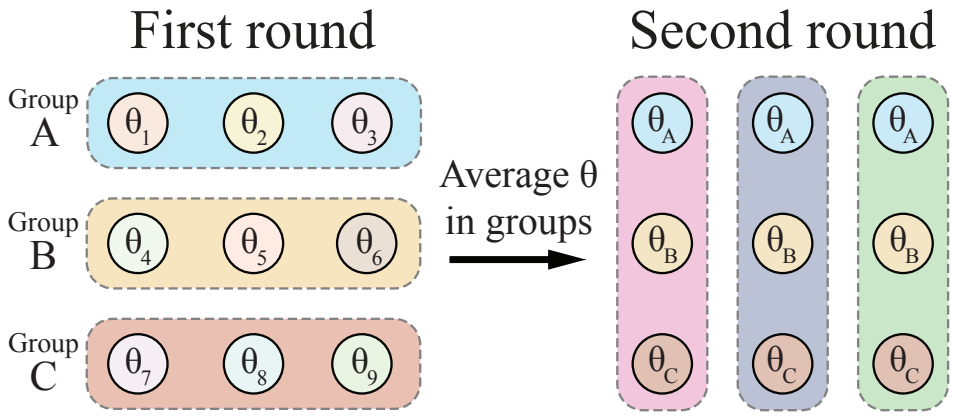


Рис. 5: Пример выполнения шагов Moshpit All-Reduce для 9 участников в 2 итерациях.

Метод проиллюстрирован на рис. 5 и формально описан в алгоритме 1. Участники находят следующую группу для усреднения, используя *индексы сетки* C, которые формируются как упорядоченный набор целых чисел. Эти индексы инициализируются с помощью функции `get_initial_index`, которая выбирает индексы для каждого измерения из равномерного распределения. На каждой итерации процедуры узлы обмениваются информацией с помощью DHT, сообщая свои сетевые адреса и текущие индексы сетки. Затем они объединяются в группы и получают список ближайших участников, также используя DHT. На следующем этапе участники внутри одной группы обмениваются данными, используя метод Butterfly All-Reduce [42]: в этой процедуре, показанной на рис. 6, каждый узел агрегирует отдельную часть усредненного вектора. Наконец, участники обновляют свои индексы сетки на основе индексов частей в Butterfly All-Reduce, за которые они отвечали: это обеспечивает новый набор соседей для следующего шага, поскольку в рамках одного вызова All-Reduce все индексы уникальны.

Если сетка полностью занята ($N \equiv M^d$) и каждый вызов All-Reduce был успешен, алгоритм 1 становится обобщением Torus All-Reduce [44] и вычисляет среднее значений со всех участников после d итераций, что мы доказываем в теореме ниже:

Algorithm 1 Moshpit All-Reduce для участника i

Вход: параметры $\{\theta_j\}_{j=1}^{N}$, число участников N, d, M, число итераций T, индекс i
$\theta_i^0 := \theta_i$
$C_i^0 := \texttt{get\_initial\_index(i)}$
for $t \in 1 \dots T$ **do**
  $\texttt{DHT}[C_i^{t-1}, t].\texttt{add}(\texttt{address}_i)$
  $\texttt{Matchmaking()}$
  $\texttt{peers}_t := \texttt{DHT.get}([C_i^{t-1}, t])$
  $\theta_i^t, c_i^t := \texttt{AllReduce}(\theta_i^{t-1}, \texttt{peers}_t)$
  $C_i^t := (C_i^{t-1}\texttt{[1:]}, c_i^t)$
end for
Выход: θ_i^T

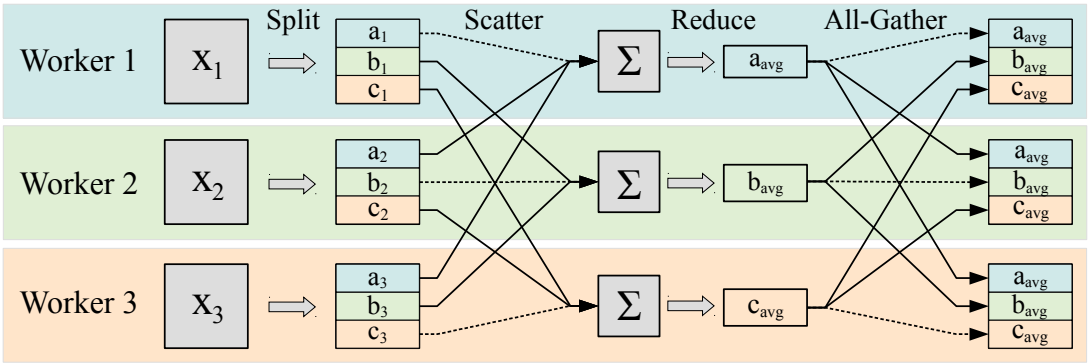


Рис. 6: Алгоритм Butterfly All-Reduce. На первом шаге усредняемый вектор разделяется на равные сегменты. Затем каждый участник агрегирует присвоенный ему сегмент со всех узлов. Наконец, участники отправляют усредненные части остальным узлам, чтобы получить результат.

Теорема 1. *Пусть M^d участников расположены в d-мерном гиперкубе с M позициями в каждом измерении. Также пусть каждый участник успешно участвует в каждом шаге усреднения и группы для каждой итерации усреднения определяются на основе координат гиперкуба. Тогда при запуске d итераций Moshpit All-Reduce без повторения групп (т.е. усреднение по каждому измерению проводится ровно один раз) результатом для каждого участника будет среднее значение θ по всем M^d узлам.*

На практике эти предположения не будут выполняться в течение всего периода обучения. В частности, добровольцы подключаются и отключаются во время эксперимента (возможно, во время фазы All-Reduce); кроме того, может быть сложно обеспечить абсолютное заполнение сетки. Однако, как мы показываем в статье, Moshpit All-Reduce может

асимптотически уменьшить невязку между значениями на узлах даже в условиях непостоянного участия и неравномерных групп.

Наконец, если участники в одной группе имеют неодинаковую пропускную способность сети, назначение равных частей вектора в Butterfly All-Reduce приведет к дисбалансу во времени коммуникации и ненужным задержкам в процедуре обучения. Чтобы предотвратить это, мы можем динамически регулировать нагрузку по передаче данных каждого узла. В частности, мы формулируем оптимизационную задачу, которая распределяет различные доли w_i усредняемого вектора между M участниками с различной пропускной способностью сети b_i. Мы минимизируем общее время усреднения, которое может быть смоделировано как максимальное время на всех узлах для получения, агрегации и отправки соответствующей части вектора и отправки (и получения) остальной части вектора для агрегации остальными узлами.

Результирующая задача оптимизации приведена ниже:

$$\begin{aligned} \min_{w} \quad & \max_{i} \left(1 - w_i + (M-1)w_i\right) \cdot \tfrac{1}{b_i} \\ \text{subject to} \quad & \textstyle\sum_{i=1}^{M} w_i = 1, \\ & w_i \geq 0 \qquad\qquad \forall i \end{aligned} \tag{5}$$

Как мы показываем в статье, эту задачу можно свести к задаче линейного программирования, что позволяет нам использовать эффективные алгоритмы [6] и находить оптимальную стратегию усреднения за малое время.

Moshpit SGD

Для проведения распределенного обучения в сетях, состоящих из ненадежных участников, мы формулируем Moshpit SGD — модификацию стандартного стохастического градиентного спуска с локальными шагами [36]. Во время фазы передачи данных узлы используют Moshpit All-Reduce для усреднения параметров вместо использования стандартного All-Reduce или сервера параметров [30]. Важно отметить, что при ряде стандартных предположений мы можем получить оценки сходимости для Moshpit SGD, сравнимые с лучшими результатами для менее эффективных методов [1; 14] на момент публикации.

Эмпирические результаты

В первом наборе экспериментов мы проверяем эффективность и отказоустойчивость Moshpit All-Reduce, которые мы стремились получить при разработке этого метода. Для этого мы проводим симулированные эксперименты по усреднению, используя выборку из стандартного нормального распределения в качестве входных данных для каждого узла. Мы рассматриваем несколько вариантов: 1024 участника без сбоев, 1024 участника с

долей сбоев на каждом шаге 0.005 и 768 участников с частотой сбоев 0.005 (этот вариант нужен для демонстрации влияния заполнения сетки). В качестве целевой метрики мы используем среднеквадратичное отклонение данных на всех узлах (или, эквивалентно, дисперсию значений между узлами): успешная процедура точного усреднения приводит к отклонению равному нулю.

В качестве базовых подходов мы используем несколько алгоритмов из предыдущих работ: список включает All-Reduce (с перезапусками после сбоев), Gossip [13] и PushSum [53]. Кроме того, мы исследуем упрощение Moshpit All-Reduce, которое усредняет параметры в небольших *случайных* группах.

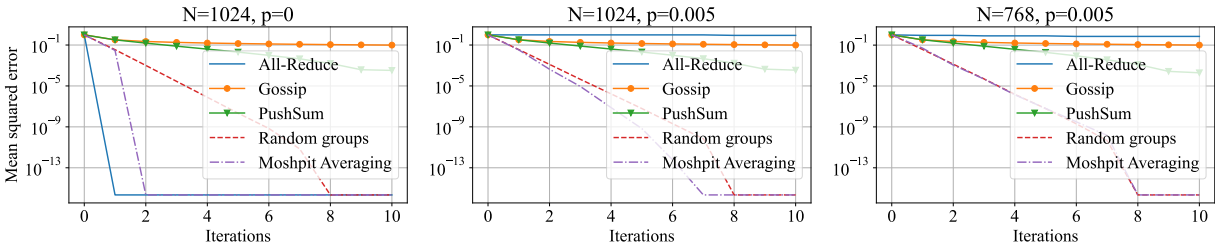


Рис. 7: Средняя квадратичная ошибка методов усреднения в зависимости от числа итераций.

Основные результаты этих экспериментов показаны на рис. 7. Ожидаемо, стандартный All-Reduce вычисляет среднее значение за один шаг при отсутствии сетевых сбоев. Однако он не может усреднить значения с узлов в менее оптимистичных случаях; Gossip и PushSum также требуют значительного числа итераций для сходимости. С другой стороны, Moshpit SGD вычисляет точное среднее за 2 итерации в случае отсутствия сбоев и сходится быстрее других методов в случае ненулевой вероятности сбоя. Усреднение в случайных группах имеет результаты, схожие с Moshpit All-Reduce при сниженной загрузке сетки, но уступает нашему методу при более полных сетках.

Кроме того, мы тестируем Moshpit SGD в нескольких экспериментах по распределенному обучению, сравнивая его производительность с несколькими популярными алгоритмами для крупномасштабного глубинного обучения. Мы рассматриваем два сценария, охватывающих различные области применения: обучение классификатора изображений ResNet-50 [16] на ImageNet [24] и предварительное обучение модели представления языка ALBERT-large [3] на BookCorpus [4]. Наряду с Moshpit SGD мы оцениваем несколько базовых подходов, включая стандартный All-Reduce SGD, а также два децентрализованных метода: AD-PSGD [9] и Stochastic Gradient Push [52]. Сценарии обучения охватывают как случай однородной инфраструктуры (один сервер с несколькими GPU), так и гетерогенного обучения в различных окружениях.

Результаты этих экспериментов можно увидеть на рис. 8: как мы показываем, Moshpit SGD превосходит все базовые решения по реальному времени до сходимости, включая

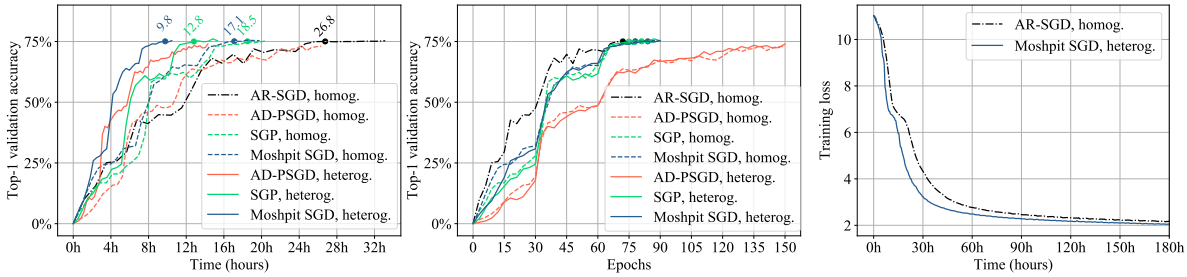


Рис. 8: **(Слева, посередине)** Доля правильных ответов ResNet-50 на валидационной выборке ImageNet в зависимости от реального времени (слева) и числа эпох обучения (посередине). **(Справа)** Сходимость функции потерь при обучении ALBERT-large.

лучшие децентрализованные подходы. Важно отметить, что это *не означает*, что время сходимости в терминах итераций также наименьшее: All-Reduce SGD (AR-SGD) сходится за меньшее количество эпох благодаря более точным оценкам градиента на каждом шаге. Для обучения ALBERT в гетерогенной постановке используются менее мощные вытесняемые узлы, что делает ее более эффективной по стоимости, чем однородную постановку. Однако стоит заметить, что нестабильность узлов делает невозможным обучение с помощью All-Reduce в подобном окружении. С другой стороны, отказоустойчивость Moshpit SGD позволяет достичь того же значения функции потерь за меньшее время.

3.3 Распределенное глубинное обучение в открытых коллаборациях

Хотя Moshpit SGD и может быть применен сам по себе для обучения с добровольцами, работа с большим количеством участников эффективным, доступным и надежным образом требует решения нескольких дополнительных проблем. В частности, в условиях совместного обучения можно ожидать, что состав участников, а следовательно, и набор аппаратных и сетевых условий узлов будет изменяться с течением времени. Кроме того, динамическое участие серверов может сделать нетривиальной задачу обеспечения того, чтобы результаты обучения были сравнимы (в идеале эквивалентны) с результатами, полученными в стандартных условиях кластера. Для решения этих проблем мы разработали Distributed Deep Learning in Open Collaborations (DeDLOC) — комплексный подход для децентрализованный параллельного по данным обучения с участием добровольцев.

Описание метода

Относительно более легкой задачей является достижение консистентных результатов обучения. Поскольку большинство лучших моделей сейчас обучаются на больших подвыборках для каждого шага оптимизации [27; 28; 39; 46; 58], мы используем тот же под-

ход, накапливая градиенты для обработанных примеров по всей коллаборации, прежде чем сделать синхронный шаг алгоритма оптимизации на всех узлах. Важно отметить, что это позволяет естественным образом поддержать динамическое участие узлов: если один из участников покидает эксперимент во время обучения, другие компенсируют это, обрабатывая больше примеров, что задержит, но не отменит следующий шаг обучения. И наоборот, если к эксперименту присоединяется новый узел, скорость вычислений коллаборации возрастает, а время между последующими шагами уменьшается. Иллюстрацию этого свойства обучения с большими батчами можно увидеть на рис. 9.

Важно, что здесь предполагается, что все участники обрабатывают данные, поступающие из одного и того же распределения: в противном случае остающиеся в сети узлы не смогли бы компенсировать результаты работы отключенных. Тем не менее, в экспериментах с крупномасштабным предварительным обучением данные чаще всего получают из открытых источников, а объем их составляет сотни гигабайт; таким образом, это предположение вполне реалистично для рассматриваемой нами постановки задачи.

Данный подход позволяет нам иметь такие же же обновления весов и, следовательно, ту же динамику обучения, что и в стандартном параллельном по данным обучении с алгоритмической точки зрения. Следовательно, если шаг обучения и процедура агрегации градиентов такие же, как и в обычной централизованной постановке, DeDLOC достигнет результатов, схожих с результатами стандартных методов вплоть до численной точности (и других источников нестабильности, часто встречающихся в глубоком обучении). На практике это ограничение ослабляется: мы можем использовать методы группового усреднения, такие как [12] или Moshpit All-Reduce для повышения отказоустойчивости, а также отложенные обновления параметров (DPU [59], также известные как отложенный на один шаг стохастический градиентный спуск [8]) для параллельного выполнения дорогостоящей фазы коммуникации с вычислением градиентов для следующего шага. Важно,

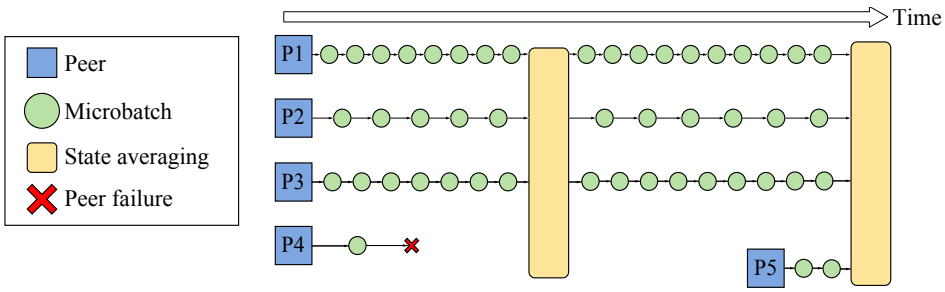


Рис. 9: Пример двух шагов обучения DeDLOC с присоединяющимися и отключающимися узлами. Сначала отключается участник 4, поэтому другим нужно обработать больше примеров, чтобы достичь того же размера подвыборки. На следующей итерации присоединяется участник 5, и время до накопления целевого размера батча становится меньше.

что и анализ, и эмпирические результаты в литературе показывают, что эти модификации не оказывают большого влияния на сходимость обучения.

Далее в работе мы предлагаем адаптивный алгоритм для параллельного по данным обучения, который может динамически корректировать стратегию агрегации с учетом текущих сетевых и аппаратных условий коллаборации. Это необходимо для достижения оптимальной производительности в гетерогенной среде с динамическим участием узлов: стратегия с равномерной нагрузкой будет субоптимальной в присутствии участников с сильно различающейся производительностью. Более того, узлы с быстрым интернет-соединением могут иметь относительно низкую скорость вычислений и наоборот.

Для решения этой проблемы мы формулируем оптимизационную задачу, максимизирующую пропускную способность обучения. В случае DPU пропускная способность может рассматриваться как минимум из времени обработки B примеров и времени агрегации градиента для нейронной сети с P параметрами. Нашими основными ограничениями являются вычислительная производительность каждого участника i (s_i градиентов для примеров, вычисляемых в секунду), скорость загрузки и отдачи данных (d_i и u_i соответственно), а также попарные ограничения на скорость коммуникации (t_{ij} и t_{ji} для участников i и j). Мы оптимизируем стратегию усреднения относительно вычисления градиентов узлами (индикаторные переменные c_i), скорости отправки локальных градиентов для агрегирования a_{ij} и скорости отправки агрегированных градиентов обратно g_{ij}.

Итоговая задача оптимизации приведена ниже:

$$
\begin{aligned}
\max_{a,g,c} \quad & \min\left(\frac{\sum_{i=1}^{n} s_i \cdot c_i}{B},\ \frac{\min_i \sum_j g_{ji}}{P}\right) & & \\
\text{subject to} \quad & g_{ij} \le \min_{k \in \{r : c_r = 1\}} a_{ki} & & \forall i, j \\
& \textstyle\sum_{j \neq i} (a_{ji} + g_{ji}) \le d_i & & \forall i \\
& \textstyle\sum_{j \neq i} (a_{ij} + g_{ij}) \le u_i & & \forall i \\
& a_{ij} + g_{ij} \le t_{ij} & & \forall i, j \\
& a_{ij},\ g_{ij} \ge 0, c_i \in \{0, 1\} & & \forall i, j
\end{aligned}
\tag{6}
$$

Примером допустимого решения для этой задачи является $c_i = 1$ только для узла с наибольшим s_i и $a, g \equiv 0$, что соответствует отсутствию коммуникации. Однако, чтобы найти наиболее эффективную стратегию обучения для заданного состава коллаборации, нам необходимо находить оптимальное решение для каждой итерации обучения. Таким образом, мы переформулируем задачу оптимизации в виде линейной программы, что позволяет нам находить решение за время, меньшее одной секунды, с помощью существующих методов [6].

Заметим, что в частных случаях (например, при однородных сетевых и аппаратных условиях) оптимальное решение соответствует известным методам распределенного обу-

чения, так как описанная нами стратегия обобщает эти подходы. На рис. 10 показана иллюстрация нескольких стратегий, которые будут являться оптимальными в различных условиях.

Эмпирические результаты

В первой группе наших экспериментов мы проверяем адаптивные свойства DeDLOC, запуская его в наборе окружений с различной степенью неоднородности узлов. Нашим фокусом являются задачи машинного обучения с предобучением без учителя: они требуют значительных ресурсов для обучения и могут применяться к множеству практических задач, что потенциально делает их привлекательными для обучения силами волонтеров.

Нашим первым шагом является измерение эффективности обучения модели универсальных представлений изображений Swapping Assignments between multiple Views (SwAV) [57] на выборке ImageNet. Мы рассматриваем три сценария: *Server* с 8 узлами, имеющими по одной GPU V100 и сеть со скоростью 1 Гб/с, *Workstation* с 16 рабочими узлами, имеющими по одной GPU 1080 Ti и сетью 200 Мб/с, а также *Hybrid*, объединяющую эти две постановки. Для измерения скорости усреднения мы также включили четвертую постановку, которая добавляет один узел с высокой пропускной способностью сети. Нашими ключевыми показателями эффективности являются производительность обучения как функция времени, а также скорость усреднения градиентов. Результаты этих экспериментов представлены на рис. 11 и в таблице 2. Ключевой результат заключается в наблюдении, что гибридная стратегия усреднения позволяет получить ускорение до 92% в гетерогенных сетях по сравнению с базовыми методами All-Reduce и Parameter Server, и при этом близка к оптимальному подходу во всех протестированных постановках.

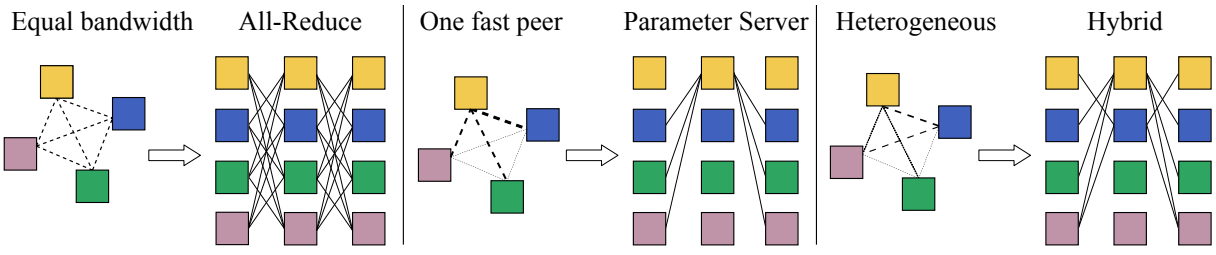


Рис. 10: Пример аппаратных и сетевых условий для участников сети с соответствующими оптимальными стратегиями усреднения. Жирные линии соответствуют более быстрым соединениям.

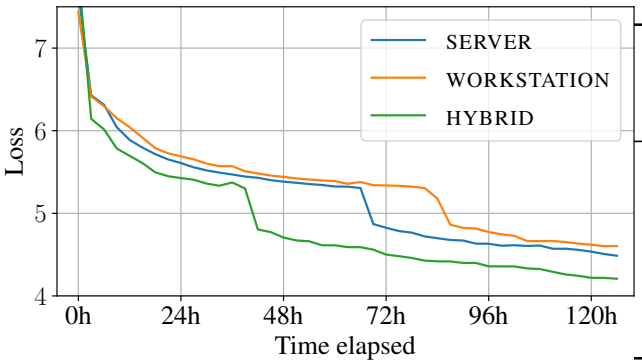


Рис. 11: Сходимость метода SwAV в различных постановках.

Постановка	Алгоритм		
	AR	PS	Адаптивная
A: 8x1Gb/s	**1.19**	4.73	1.20
B: 16x0.2Gb/s	**5.3**	39.6	**5.3**
C: A + B	5.69	14.1	**2.96**
D: B + 1x2.5Gb/s	5.3	3.22	**3.18**

Таблица 2: Производительность усреднения градиентов SWaV.

Далее мы оцениваем эффективность предварительного обучения ALBERT-large [3] на WikiText-103 [38]. Мы сравниваем сходимость функции потерь при обучении в пяти сценариях: *High-bandwidth* имеет 16 узлов с GPU T4 и пропускной способностью 25 Гб/с, *Heterogeneous* имеет те же GPU с сетью 4x 200 Мб/с, 8x 100 Мб/с и 4x 50 Мб/с; в *Heterogeneous + load balancing* добавляется балансировка нагрузки с помощью алгоритма адаптивного усреднения, а постановки *CPU-only* и *Time-varying* добавляют узлы с быстрой сетевой связностью, но без GPU и узлы с нестабильным участием соответственно.

Результаты этого сравнения показаны на рис. 12: наиболее важным наблюдением является то, что использование стандартного алгоритма усреднения в гетерогенных условиях значительно замедляет обучение, а адаптивная стратегия приближает производительность к результатам обучения на типичном кластере. Наличие узлов, которые участвуют только в усреднении или участвуют в обучении часть времени, помогает еще больше сократить время обучения.

Наконец, мы провели эксперимент с коллаборативным обучением в реальных условиях, выполнив обучение версии ALBERT на бенгальском языке на соответствующей части выборки OSCAR [33]. К эксперименту присоединились 40 добровольцев, участвуя как со

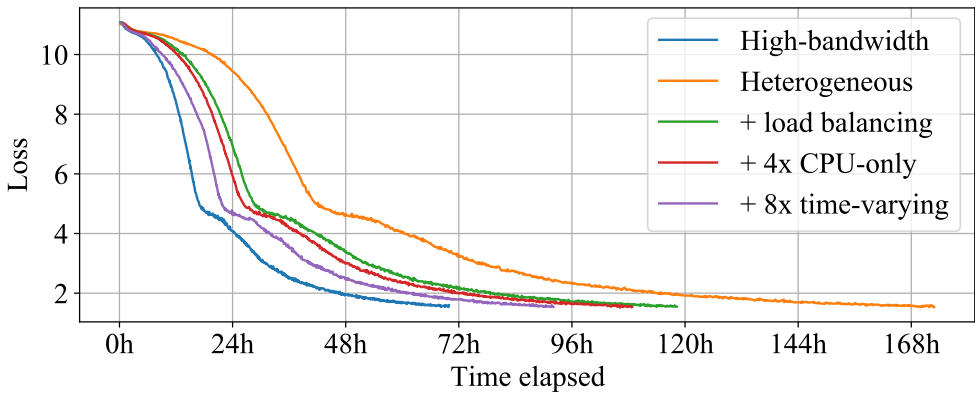
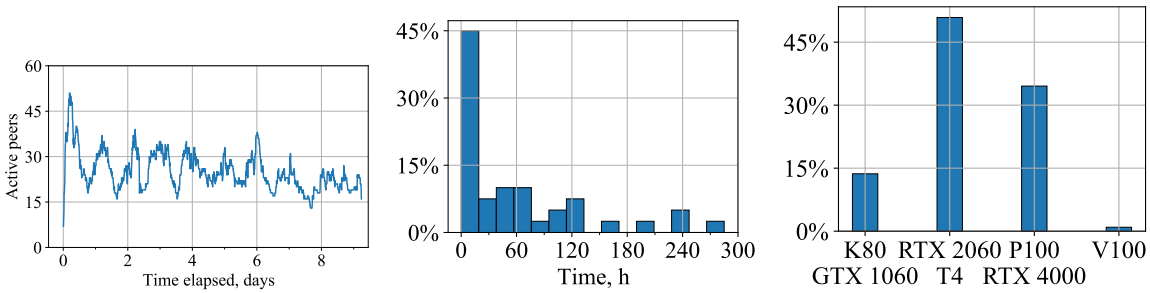


Рис. 12: Обучение модели ALBERT в различных постановках.



(a) Число участников с течением времени.

(b) Гистограмма времени участия.

(c) Аппаратные ускорители.

Рис. 13: Результаты коллаборативного эксперимента.

своих собственных компьютеров с GPU, так и с бесплатными облачных серверов; по итогам голосования полученная модель была названа sahajBERT. Суммарно участники эксперимента подключались к обучению с 91 различных серверов, из которых в среднем 15-35 были в сети единовременно. Среднее время участия составило около 36 часов; дополнительная статистика по волонтерам, собранная в ходе эксперимента, доступна на рис. 13. Обучение модели заняло около 8 дней, что оказалось в 1.8 раз быстрее, чем у базовой системы с несколькими GPU в одном сервере, которую мы использовали для проверки эквивалентности результатов (см. рис. 14).

Мы также сравнили результаты дообучения sahajBERT на двух выборках на бенгальском языке с результатами аналогичных моделей. В качестве данных рассматривались WikiANN [15] и News Category Classification из IndicGLUE [26]; из моделей сравнивались одноязычная bnRoBERTa [25], а также многоязычные IndicBERT [26] и XLM-RoBERTa [56]. Результаты сравнения приведены в таблице 3: несмотря на то, что для sahajBERT использовались нестабильные ресурсы добровольцев, а не выделенный суперкомпьютер, модель достигла сравнимых и даже лучших результатов, чем у нейронных сетей, обученных на суперкомпьютерах.

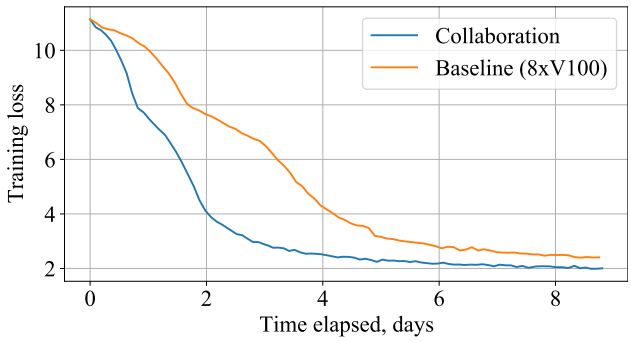


Рис. 14: Динамика обучения sahajBERT.

Модель	WikiANN	NCC
bnRoBERTa	82.32 ± 0.67	80.94 ± 0.45
IndicBERT	92.52 ± 0.45	74.46 ± 1.91
XLM-R	**96.48** ± **0.22**	90.05 ± 0.38
sahajBERT	95.45 ± 0.53	**91.97** ± **0.47**

Таблица 3: Сравнение моделей на конечных задачах.

4 Заключение

В этом разделе мы резюмируем основной научный вклад этой работы. Результатом работы является набор методов для крупномасштабного децентрализованного глубинного обучения на нестабильных гетерогенных узлах с медленным соединением (в частности, на компьютерах добровольцев).

1. Предложена *децентрализованная смесь экспертов* (DMoE) — нейросетевой слой, разработанный в целях отказоустойчивости и обучения со значительным числом участников, имеющих различный объем памяти. Также разработана *Learning@home* — система для глубинного обучения в условиях добровольных вычислений. Она использует сохранение промежуточных градиентов и распределенные хэш-таблицы для отказоустойчивости и асинхронную очередь запросов для максимизации пропускной способности обучения. Эмпирически подтверждено, что Learning@home поддерживает высокую пропускную способность обучения как при эмулированной сетевой задержке, так и в реалистичных условиях межрегионального обучения. Наконец, на двух задачах машинного обучения показано, что нейронные сети со слоями DMoE сходятся к тому же качеству, что и обычные модели с сопоставимым количеством параметров в условиях отказов узлов и высокой задержки.

2. Предложен *Moshpit All-Reduce* — децентрализованный итеративный метод для усреднения градиентов или параметров модели, который опирается на эффективные коммуникационные примитивы и при этом может быть использован в неустойчивых сетях. Данный метод может быть объединен со стандартными алгоритмами стохастической оптимизации для получения *Moshpit SGD* — метода распределенного обучения, подходящего для обучения с ненадежными устройствами. В экспериментах на синтетических данных Moshpit All-Reduce достигает наименьшей невязки между узлами за заданное число итераций по сравнению с популярными методами агрегации, используемыми в предыдущих работах. Moshpit SGD в 1,5 раза превосходит базовые решения на двух крупномасштабных задачах глубинного обучения в терминах реального времени до сходимости. Мы также описали подход по *динамической балансировке нагрузки*, решающий задачу линейного программирования для распределения нагрузки между участниками с различной пропускной способностью сети.

3. Предложен *Distributed Deep Learning in Open Collaborations*, масштабируемый метод для совместного глубинного обучения. Он использует *адаптивную стратегию усреднения* для учета неравномерных вычислительных ресурсов и сетевых соедине-

ний узлов, а также обучение с большими батчами и *глобальной аккумуляцией градиентов* по всей сети для учета динамического участия серверов и присутствия узлов с задержками. Мы также провели эксперимент по совместному предварительному обучению в реальных условиях с сообществом добровольцев и получили *sahajBERT* — модель представлений бенгальского языка на основе архитектуры ALBERT. На момент обучения эта модель достигала качества, схожего с лучшими аналогами, несмотря на то, что обучение проводилось на оборудовании добровольцев, а не на кластере вычислительных ускорителей.

Список литературы

1. A unified theory of decentralized SGD with changing topology and local updates / A. Koloskova [и др.] // International Conference on Machine Learning. — PMLR. 2020. — С. 5381—5393.
2. Adaptive Mixtures of Local Experts / R. A. Jacobs [и др.] // Neural Computation. — Cambridge, MA, USA, 1991. — Март. — Т. 3, № 1. — С. 79—87. — ISSN 0899-7667. — DOI: 10.1162/neco.1991.3.1.79. — URL: https://doi.org/10.1162/neco.1991.3.1.79.
3. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations / Z.-Z. Lan [и др.] // ArXiv. — 2019. — Т. abs/1909.11942.
4. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books / Y. Zhu [и др.] // Proceedings of the IEEE international conference on computer vision. — 2015. — С. 19—27.
5. Amazon EC2 Spot Instances. — URL: https://aws.amazon.com/ec2/spot/.
6. *Andersen E. D.*, *Andersen K. D.* The Mosek Interior Point Optimizer for Linear Programming: An Implementation of the Homogeneous Algorithm // Applied Optimization. — Springer US, 2000. — С. 197—232. — DOI: 10.1007/978-1-4757-3216-0_8. — URL: https://doi.org/10.1007%2F978-1-4757-3216-0_8.
7. *Anderson D. P.* Boinc: A system for public-resource computing and storage // Fifth IEEE/ACM international workshop on grid computing. — IEEE. 2004. — С. 4—10.
8. *Arjevani Y.*, *Shamir O.*, *Srebro N.* A tight convergence analysis for stochastic gradient descent with delayed updates // Algorithmic Learning Theory. — PMLR. 2020. — С. 111—132.
9. Asynchronous Decentralized Parallel Stochastic Gradient Descent / X. Lian [и др.] // Proceedings of the 35th International Conference on Machine Learning. Т. 80 / под ред. J. Dy, A. Krause. — PMLR, 10–15 Jul.2018. — С. 3043—3052. — (Proceedings of Machine Learning Research). — URL: https://proceedings.mlr.press/v80/lian18a.html.
10. Attention is All you Need / A. Vaswani [и др.] // Advances in Neural Information Processing Systems 30 / под ред. I. Guyon [и др.]. — Curran Associates, Inc., 2017. — С. 5998—6008. — URL: http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.

11. Beyond hierarchies: Design considerations for distributed caching on the internet : тех. отч. / R. Tewari [и др.] ; Citeseer.

12. Breaking (Global) Barriers in Parallel Stochastic Optimization with Wait-Avoiding Group Averaging / S. Li [и др.] // IEEE Transactions on Parallel and Distributed Systems. — 2020. — С. 1—1. — ISSN 2161-9883. — DOI: 10.1109/tpds.2020.3040606. — URL: http://dx.doi.org/10.1109/TPDS.2020.3040606.

13. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent / X. Lian [и др.] // Advances in Neural Information Processing Systems. — 2017. — С. 5330—5340.

14. Communication efficient decentralized training with multiple local updates / X. Li [и др.] // arXiv preprint arXiv:1910.09126. — 2019. — Т. 5.

15. Cross-lingual Name Tagging and Linking for 282 Languages / X. Pan [и др.] // Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics. — 07.2017. — С. 1946—1958. — DOI: 10.18653/v1/P17-1178. — URL: https://www.aclweb.org/anthology/P17-1178.

16. Deep Residual Learning for Image Recognition / K. He [и др.] // 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). — 2015. — С. 770—778.

17. Einstein@Home All-sky Search for Continuous Gravitational Waves in LIGO O2 Public Data / B. Steltner [и др.] // The Astrophysical Journal. — 2021. — Март. — Т. 909, № 1. — С. 79. — ISSN 1538-4357. — DOI: 10.3847/1538-4357/abc7c9. — URL: http://dx.doi.org/10.3847/1538-4357/abc7c9.

18. Emergent abilities of large language models / J. Wei [и др.] // arXiv preprint arXiv:2206.07682. — 2022.

19. Folding@home gets 1.5+ Exaflops to Fight COVID-19. — Accessed: 2021-05-20. https://blogs.nvidia.com/blog/2020/04/01/foldingathome-exaflop-coronavirus/.

20. Folding@home update on SARS-CoV-2 (10 MAR 2020). — https://foldingathome.org/covid19/(accessed on June 4, 2020).

21. Google Cloud Spot VMs pricing. — URL: https://cloud.google.com/compute/docs/instances/spot#pricing.

22. Gpipe: Efficient training of giant neural networks using pipeline parallelism / Y. Huang [и др.] // Advances in Neural Information Processing Systems. — 2019. — С. 103—112.

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

24. ImageNet: A Large-Scale Hierarchical Image Database / J. Deng [и др.] // CVPR09. — 06.2009. — С. 248—255. — DOI: 10.1109/CVPR.2009.5206848.

25. Indic-Transformers: An Analysis of Transformer Language Models for Indian Languages / K. Jain [и др.] // arXiv preprint arXiv:2011.02323. — 2020.

26. IndicNLPSuite: Monolingual Corpora, Evaluation Benchmarks and Pre-trained Multilingual Language Models for Indian Languages / D. Kakwani [и др.] // Findings of the Association for Computational Linguistics: EMNLP 2020. — 11.2020. — С. 4948—4961. — DOI: 10.18653/v1/2020.findings-emnlp.445. — URL: https://www.aclweb.org/anthology/2020.findings-emnlp.445.

27. Language Models are Few-Shot Learners / T. B. Brown [и др.] // arXiv preprint arXiv:2005.14165. — 2020. — Т. 33. — С. 1877—1901. — URL: https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

28. Large Batch Optimization for Deep Learning: Training BERT in 76 minutes / Y. You [и др.] // International Conference on Learning Representations. — 2020. — URL: https://openreview.net/forum?id=Syx4wnEtvH.

29. Large Memory Layers with Product Keys / G. Lample [и др.] // Advances in Neural Information Processing Systems 32 / под ред. H. Wallach [и др.]. — Curran Associates, Inc., 2019. — С. 8546—8557. — URL: http://papers.nips.cc/paper/9061-large-memory-layers-with-product-keys.pdf.

30. *Li M.* Scaling Distributed Machine Learning with the Parameter Server // Proceedings of the 2014 International Conference on Big Data Science and Computing. — Beijing, China : Association for Computing Machinery, 2014. — (BigDataScience '14). — ISBN 9781450328913. — DOI: 10.1145/2640087.2644155. — URL: https://doi.org/10.1145/2640087.2644155.

31. *Maymounkov P.*, *Mazieres D.* Kademlia: A peer-to-peer information system based on the xor metric // International Workshop on Peer-to-Peer Systems. — Springer. 2002. — С. 53—65.

32. Mixed Precision Training / P. Micikevicius [и др.] // International Conference on Learning Representations. — 2018. — URL: https://openreview.net/forum?id=r1gs9JgRZ.

33. *Ortiz Suárez P. J.*, *Romary L.*, *Sagot B.* A Monolingual Approach to Contextualized Word Embeddings for Mid-Resource Languages // Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. — 07.2020. — С. 1703—1714. — URL: https://www.aclweb.org/anthology/2020.acl-main.156.

34. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer / N. Shazeer [и др.] // arXiv preprint arXiv:1701.06538. — 2017.

35. PaLM: Scaling Language Modeling with Pathways / A. Chowdhery [и др.]. — 2022. — arXiv: 2204.02311 [cs.CL].

36. Parallelized Stochastic Gradient Descent / M. Zinkevich [и др.] // Advances in Neural Information Processing Systems. Т. 23 / под ред. J. Lafferty [и др.]. — Curran Associates, Inc., 2010. — С. 2595—2603. — URL: https://proceedings.neurips.cc/paper/2010/file/abea47ba24142ed16b7d8fbf2c740e0d-Paper.pdf.

37. *Patarasuk P.*, *Yuan X.* Bandwidth Optimal All-Reduce Algorithms for Clusters of Workstations // J. Parallel Distrib. Comput. — USA, 2009. — Февр. — Т. 69, № 2. — С. 117—124. — ISSN 0743-7315. — DOI: 10.1016/j.jpdc.2008.09.002. — URL: https://doi.org/10.1016/j.jpdc.2008.09.002.

38. Pointer Sentinel Mixture Models / S. Merity [и др.] // arXiv preprint arXiv:1609.07843. — 2017.

39. *Popel M.*, *Bojar O.* Training Tips for the Transformer Model // The Prague Bulletin of Mathematical Linguistics. — 2018. — Март. — Т. 110. — DOI: 10.2478/pralin-2018-0002.

40. PyTorch Distributed: Experiences on Accelerating Data Parallel Training / S. Li [и др.] // Proc. VLDB Endow. — 2020. — Авг. — Т. 13, № 12. — С. 3005—3018. — ISSN 2150-8097. — DOI: 10.14778/3415478.3415530. — URL: https://doi.org/10.14778/3415478.3415530.

41. PyTorch: An imperative style, high-performance deep learning library / A. Paszke [и др.] // Advances in Neural Information Processing Systems. — 2019. — С. 8024—8035.

42. *Rabenseifner R.* Optimization of Collective Reduction Operations // Computational Science - ICCS 2004 / под ред. M. Bubak [и др.]. — Berlin, Heidelberg : Springer Berlin Heidelberg, 2004. — С. 1—9. — ISBN 978-3-540-24685-5.

43. Randomized gossip algorithms / S. Boyd [и др.] // IEEE transactions on information theory. — 2006. — Т. 52, № 6. — С. 2508—2530.

44. *Sack P.*, *Gropp W.* Collective Algorithms for Multiported Torus Networks // ACM Trans. Parallel Comput. — New York, NY, USA, 2015. — Февр. — Т. 1, № 2. — ISSN 2329-4949. — DOI: 10.1145/2686882. — URL: https://doi.org/10.1145/2686882.

45. Scaling Laws for Autoregressive Generative Modeling / T. Henighan [и др.]. — 2020. — arXiv: 2010.14701 [cs.LG].

46. Scaling Laws for Neural Language Models / J. Kaplan [и др.]. — 2020. — arXiv: 2001.08361 [cs.LG].

47. *Sergeev A.*, *Balso M. D.* Horovod: fast and easy distributed deep learning in TensorFlow // arXiv preprint arXiv:1802.05799. — 2018.

48. SETI@home: An Experiment in Public-Resource Computing / D. Anderson [и др.] // Commun. ACM. — 2002. — Нояб. — Т. 45. — С. 56—61. — DOI: 10.1145/581571.581573.

49. SlowMo: Improving Communication-Efficient Distributed SGD with Slow Momentum / J. Wang [и др.] // International Conference on Learning Representations. — 2020. — URL: https://openreview.net/forum?id=SkxJ8REYPH.

50. *Stephen Merity et al. 2.* Wikitext-2. —. — URL: https://arxiv.org/abs/1609.07843.

51. *Stich S. U.* Local SGD Converges Fast and Communicates Little // International Conference on Learning Representations (ICLR). — 2019. — arXiv:1805.09767. — URL: https://arxiv.org/abs/1805.09767.

52. Stochastic Gradient Push for Distributed Deep Learning / M. Assran [и др.] // Proceedings of the 36th International Conference on Machine Learning. Т. 97. — 06.2019. — С. 344—353. — (Proceedings of Machine Learning Research).

53. Stochastic Gradient Push for Distributed Deep Learning / M. Assran [и др.] // Proceedings of the 36th International Conference on Machine Learning. Т. 97 / под ред. K. Chaudhuri, R. Salakhutdinov. — PMLR, 09–15 Jun.2019. — С. 344—353. — (Proceedings of Machine Learning Research). — URL: http://proceedings.mlr.press/v97/assran19a.html.

54. Training Compute-Optimal Large Language Models / J. Hoffmann [и др.]. — 2022. — arXiv: 2203.15556 [cs.CL].

55. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context / Z. Dai [и др.] // Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. — 2019. — С. 2978—2988.

56. Unsupervised Cross-lingual Representation Learning at Scale / A. Conneau [и др.] // Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. — 07.2020. — С. 8440—8451. — DOI: 10.18653/v1/2020.acl-main.747. — URL: https://www.aclweb.org/anthology/2020.acl-main.747.

57. Unsupervised Learning of Visual Features by Contrasting Cluster Assignments / M. Caron [и др.] // Advances in Neural Information Processing Systems. Т. 33. — 2020. — С. 9912—9924.

58. *You Y.*, *Gitman I.*, *Ginsburg B.* Large Batch Training of Convolutional Networks // arXiv preprint arXiv:1708.03888. — 2017.

59. ZeRO-Offload: Democratizing Billion-Scale Model Training / J. Ren [и др.] // arXiv preprint arXiv:2101.06840. — 2021.