Подбор гиперпараметров для синхронного метода распределенного глубокого обучения при решении задач классификации изображений

Кирилл Блинов [1], Илья Курочкин[2]

[1] *Национальный исследовательский технологический университет "МИСИС", Ленинский проспект 4, Москва, 119049, Россия*
[2] *Институт проблем передачи информации имени А. А. Харкевича РАН, Большой Каретный переулок 19 стр. 1, Москва, 127051, Россия*

Аннотация

Для решения задач с помощью глубоких нейронных сетей в случае большого количества данных может применяться распределенное глубокое обучение. Для некоторых распределенных систем количество вычислительных узлов может быть не определено или изменяться динамически. Необходимо найти такие гиперпараметры, для которых при разном количестве узлов метрики качества моделей не сильно изменялись. В статье рассматривается распределенное глубокое обучение с параметрическим сервером и синхронной агрегацией результатов. С использованием архитектуры ResNet-18 на наборе данных CIFAR-10 была проведена серия экспериментов, в которых варьировались число клиентов, скорость обучения и размер батча. Применялась стратегия частой агрегации, при которой обновления моделей синхронизировались после обработки каждого локального батча данных. На основе полученных результатов было подтверждено подбора такого набора гиперпараметров распределенного обучения, при котором значения точности классификации слабо зависят от количества узлов. На основе анализа полученных результатов продемонстрировано, что время распределенного обучения для некоторых наборах гиперпараметров может быть меньше, чем локальное глубокое обучение. А значение размера батча сильно влияет не только на точность классификации, но и на время распределенного обучения.

Ключевые слова

Глубокое обучение, Распределенное обучение, Распределенная вычислительная система, Метод синхронной агрегации результатов, Классификация изображений, Набор гиперпараметров

1. Введение

В последние годы возрастающие количество данных в сочетании с ограничениями по памяти и вычислительными мощностями стимулировали разработку новых подходов к машинному обучению [1]. Одним из решений в этой области стало распределенное обучение (Distributed Learning) — парадигма, предполагающая разделение вычислительной нагрузки по обучению модели между несколькими взаимосвязанными вычислительными узлами [2].

В рамках распределенного обучения существует множество стратегий и архитектур, каждая из которых обладает своими особенностями. Одним из таких подходов является локальное обучение (Local Training), где каждый узел выполняет несколько итераций обучения на своих локальных данных, прежде чем синхронизировать результаты с остальной системой [3].

7th International Workshop on Information, Computation, and Control Systems for Distributed Environments (ICCS-DE 2025), July 7–11, 2025, Irkutsk, Russia
EMAIL: skips0skips@mail.ru (A. 1); kurochkin@iitp.ru (A. 2)
ORCID: 0009-0004-8759-1719 (A. 1); 0000-0002-0399-6208 (A. 2)

ICCS-DE 2025 Workshop Proceedings
DOI: 10.47350/ICCS-DE.2025.06

В рамках одного раунда обучения узлы (клиенты) обновляют локальные копии глобальной модели, после чего отправляют результаты (например, градиенты или веса) на сервер для агрегации. Обновление глобальной модели может происходить как синхронно, так и асинхронно [4]. Данная работа фокусируется на синхронном подходе, при котором сервер ожидает обновлений от всех выбранных участников раунда, прежде чем вычислить новую версию глобальной модели и разослать ее обратно клиентам [5]. В противоположность этому, асинхронный подход позволяет узлам обновлять глобальную модель независимо друг от друга. Эффективность и скорость сходимости такой системы зависят от множества факторов, однако одним из наиболее критичных является стратегия отбора клиентов для участия в каждом раунде обучения [6].

Проблема отбора клиентов является многоаспектной. Проблему можно разделить на два взаимосвязанных направления: качественное, направленное на выбор наиболее «полезных» клиентов с точки зрения их вклада в модель, и количественное, связанное с определением оптимального числа участников в раунде [7]. Многочисленные исследования подтверждают, что увеличение числа клиентов, участвующих в агрегации, как правило, положительно сказывается на скорости сходимости [8]. Тем не менее, этот эффект нелинеен и подчиняется закону убывающей предельной полезности: прирост производительности значительно снижается после достижения определенного порога участников, как было продемонстрировано в основополагающих работах в этой области [9]. Таким образом, определение оптимального количества клиентов является нетривиальной задачей, поскольку оно зависит от характеристик набора данных, архитектуры и гиперпараметров модели [10].

Основная цель настоящей работы — подбор гиперпараметров для различного количества клиентов в распределенной системе с синхронной агрегацией. Для оценки и сравнения результатов была выбрана эталонная задача классификации изображений с использованием архитектуры ResNet-18 [11]. Чтобы обеспечить высокую скорость итераций исследования использовался набор данных CIFAR-10 [12].

Проведя серию контролируемых экспериментов с варьированием числа клиентов и ключевых гиперпараметров, данное исследование решает следующие задачи: 1) оценить зависимость скорости сходимости от числа активных клиентов; 2) предоставить практические рекомендации гиперпараметров распределенного обучения для распределенных систем с различным числом вычислительных узлов.

2. Материалы и методы

В качестве глубокой нейронной сети использовалась архитектура ResNet-18, которая до сих пор используется в некоторых задачах классификации изображений [11]. В качестве метода распределенного обучения был выбран подход с разделением по данным и с параметрическим сервером (глобальной моделью), а также с синхронной агрегацией результатов. Локальные модели инициализировались путем копирования весов глобальной модели. Использовался датасет CIFAR-10 (60 тыс. цветных изображений 32x32), разделенный на обучающий (80%, 40 тыс. изображений), валидационный (10%, 5 тыс. изображений) и тестовый (10%, 5 тыс. изображений) наборы. Разделение датасета на наборы происходило с сохранением балансов классов [12]. Применялась стандартная нормализация по 3 каналам RGB-изображения.

Ключевой особенностью метода распределенного обучения является динамическое распределение данных: в начале каждой эпохи обучения для каждого клиента случайным образом выбирался локальный набор данных размером N/K (N – общее количество изображений в обучающем наборе, K – количество клиентов) [5]. Это позволило обеспечивать изменение состава данных у каждого клиента от одной эпохи распределенного обучения к другой.

Распределенное обучение проводилось в течение 100 эпох. Каждая эпоха включала распределение данных, локальное обучение (прямой проход, вычисление потерь, обратный проход, шаг оптимизатора) на батчах данных, оценку на валидации и синхронную агрегацию обновлений локальных моделей.

2.1. Локальная синхронная агрегация

В исследовании применялась методология распределенного обучения с интенсивным информационным обменом, при которой агрегация глобальной модели производилась после каждого цикла обучения локальных моделей на одном батче данных. Это обеспечивало быструю адаптацию глобальной модели к изменениям на локальных клиентах. Использовался метод усреднения состояний (State Dict Averaging) [5]. Центральный сервер собирал обновленные веса от клиентов, вычислял поэлементное среднее арифметическое параметров и обновлял глобальную модель по формуле:

$$W_i^{(t+1)} = \frac{1}{N}\sum W_{k,i}^{(t+1)}, \quad (1)$$

где $W_{k,i}^{(t+1)}$ – i-й параметр локальной модели клиента k после раунда обучения t, N – количество клиентов, $W_i^{(t+1)}$ – соответствующий параметр обновленной глобальной модели.

2.2. Определение эпохи обучения и метрики оценки

"Общая эпоха" определялась как завершение каждой локальной моделью полного прохода по всему выделенному ей локальному набору данных. Оценка производительности глобальной модели производилась по завершении каждой "общей эпохи".

Отслеживалась точность (Accuracy) как (Количество верных предсказаний) / (Общее количество предсказаний). Использовалась функция потерь Cross-Entropy Loss [13].

Метрики:

- Global accuracy (точность на валидации в конце каждой эпохи обучения глобальной модели).
- Test accuracy (итоговая точность на тесте глобальной модели).

Также измерялось время выполнения одной полной глобальной эпохи.

3. Эксперименты и результаты

Эксперименты по распределенному обучению для многоклассовой классификации изображений проводились с использованием PyTorch на устройстве с CUDA: процессор Intel Core i9-13900K, 32 ГБ DDR5 RAM, NVIDIA GeForce RTX 3090 Ti, 512 ГБ SSD. Схема эксперимента включала параметрический сервер (глобальная модель) и несколько клиентов-узлов (локальные модели).

Для исследования влияния гиперпараметров распределенного обучения проводилась серия экспериментов (Таблица 1). Варьировались:

- NUM_CLIENTS (K): 1, 2, 4, 6, 8, 10, 15, 20, 30, 40. Влияет на объем данных на клиенте и количество локальных обновлений для агрегации.
- LEARNING_RATE (LR): 0.1, 0.01. Коэффициент обучения контролирует размер шага при обновлении весов.
- BATCH_SIZE: 1024, 256, 64. Влияет на точность оценки градиента и использование памяти.

Изменение этих параметров позволило исследовать динамику распределенного обучения с синхронной агрегацией и влияние на качество обучения модели.

Основным показателем качества обучения является итоговая точность (Test Accuracy) глобальной модели, измеренная на отложенном тестовом наборе данных после завершения 100 эпох обучения. Результаты для всех исследованных комбинаций гиперпараметров сведены в таблицу 1.

Таблица 1

Точность (Test accuracy) глобальной модели на тестовом наборе данных в зависимости от количества клиентов (K), скорости обучения (LR) и размера батча (B)

Кол-во клиентов (K)	LR = 0.1, B = 64	LR = 0.1, B = 256	LR = 0.1, B = 1024	LR = 0.001, B = 64	LR = 0.001, B = 256	LR = 0.001, B = 1024
1	0.7252	0.7280	0.7050	0.6636	0.5474	0.4958
2	0.7288	0.7400	0.7040	0.6204	0.5278	0.4818
4	0.7226	0.7180	0.6752	0.5886	0.5102	0.4742
6	0.7288	0.7122	0.6602	0.5724	0.4972	0.4600
8	0.7218	0.7058	0.6432	0.5560	0.4848	0.4636
10	0.7180	0.6778	0.6610	0.5450	0.5022	0.4472
15	0.7152	0.6788	0.5818	0.5350	0.5042	0.4548
20	0.7090	0.6688	0.5444	0.5330	0.4806	0.3910
30	0.6970	0.6486	0.5912	0.5398	0.4840	0.3830
40	0.6814	0.6324	0.0998	0.5286	0.4624	0.1022
Мат. ожидание	0.7148	0.6910	0.5866	0.5682	0.5001	0.4154
Дисперсия	0.0002	0.0014	0.0305	0.0016	0.0006	0.0152

Эксперименты проводились с различным числом узлов (клиентов) в распределенной системе. Одной из целей данной серии экспериментов являлась проверка возможности подбора таких гиперпараметров, чтобы точность (accuracy) на валидационной и тестовой выборке не сильно зависела от количества узлов и была на достаточно высоком уровне.

Были построены графики динамики точности на валидационном наборе для различных наборов гиперпараметров (Рисунок 1).

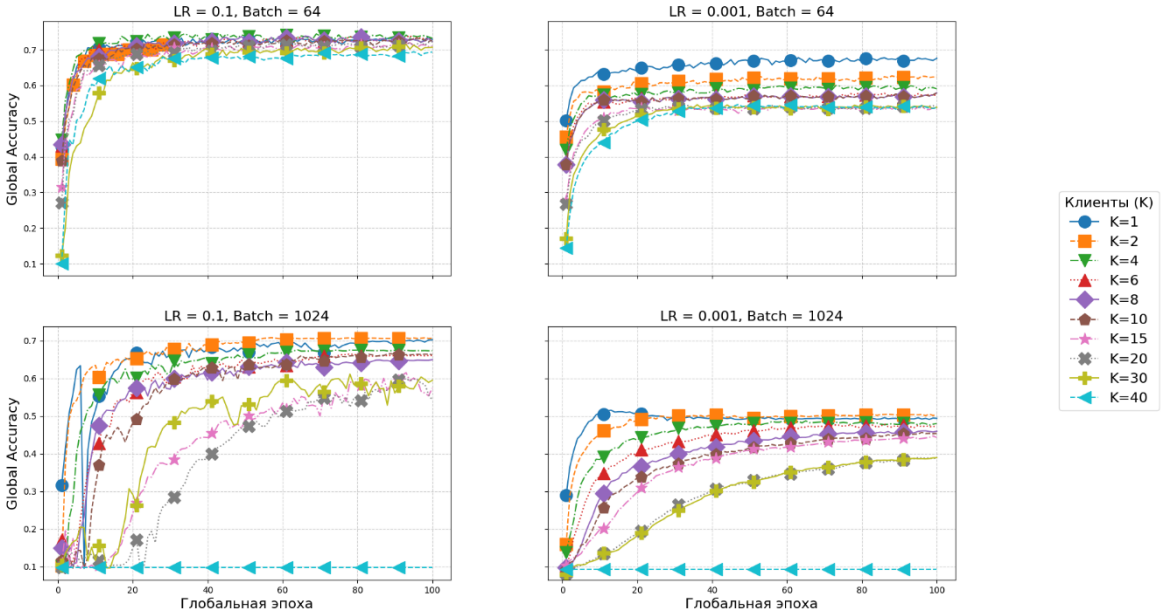

Рисунок 1: Динамика точности на валидационном наборе данных (%) после 100 эпох обучения в зависимости от числа клиентов (K), скорости обучения (LR) и размера пакета (Batch Size)

В таблице 1 и на рисунке 1видно, что при использовании набора гиперпараметров LR=0.1 и Batch Size=64 точность (accuracy) на валидационном наборе данных достаточно высока и значения для разных количеств узлов не сильно различаются.

Уменьшение параметра скорости обучения LR ухудшает результаты для всех вариантов комбинаций значений параметров Batch Size и количества узлов K. При этом следует заметить,

что для больших распределенных систем (с большим количеством узлов) точность падает сильнее.

Несмотря на то, что лучшие результаты по точности accuracy=0.74 (Таблица 1) были достигнуты на другом наборе параметров LR=0.1, Batch Size=256, K=2, дисперсия значений по точности в зависимости от числа узлов была выше, чем для набора LR=0.1 и Batch Size=64.

При этом математическое ожидание точности для набора LR=0.1 и Batch Size=64 выше, чем математическое ожидание для набора LR=0.1, Batch Size=256.

Следовательно, следует заметить, что с ростом числа узлов K объем локального набора данных (N/K) у каждого клиента пропорционально уменьшается. Это приводит к тому, что локальные обновления, вычисленные на малых и потенциально нерепрезентативных выборках, становятся более «шумными». Однако данный недостаток будет нивелирован в случае фиксированного размера локального набора данных, а также при больших датасетах.

На основе анализа динамики точности можно сделать вывод, что для распределенной системы с большим количеством узлов и/или с непостоянным количеством узлов нужно использовать высокие значения скорости обучения и небольшой размер Batch Size.

Помимо точности, важным аспектом производительности распределенной системы является общее время, необходимое для обучения модели. Было измерено полное время выполнения 100 эпох для различных комбинаций гиперпараметров (Рисунок 2).

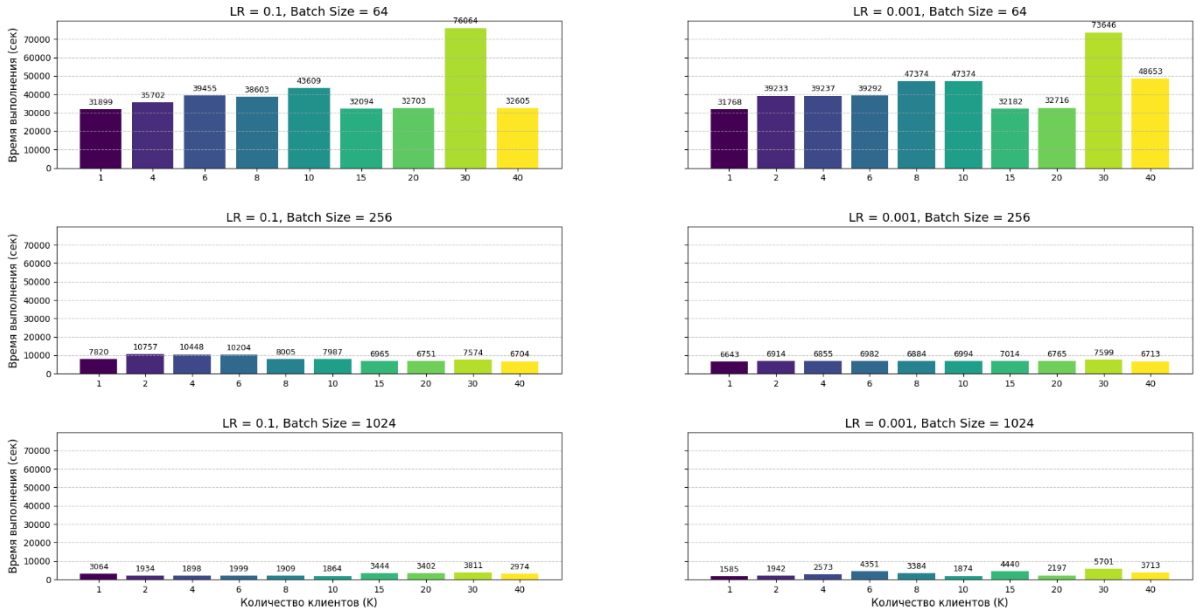

Рисунок 2: Общее время обучения (в секундах) за 100 эпох в зависимости от количества клиентов (K), скорости обучения (LR) и Batch Size

Совместный анализ времени и точности показывает, что не существует единственного набора гиперпараметров. Выбор должен быть компромиссом между скоростью и качеством:

- Использование малых батчей (B=64) является вычислительно неэффективным в исследуемой архитектуре, так как увеличение K приводит к росту временных затрат без гарантированного улучшения точности.
- Использование средних батчей (B=256) демонстрирует наиболее сбалансированное поведение, позволяя найти компромисс между ускорением и итоговой точностью при K в диапазоне 4–15.
- Использование больших батчей (B=1024) обеспечивает максимальное ускорение, однако сопряжен с риском деградации точности и нестабильности обучения при большом числе клиентов.

Стоит отметить, что время обучения для некоторых наборов гиперпараметров меньше для 15-20 узлов, чем для 1 узла. Здесь можно говорить о достаточно эффективном разбиении датасета CIFAR на части между узлами.

Анализ результатов по измерению времени распределенного обучения позволяет сделать вывод о значимости выбора гиперпараметра Batch Size и размера локального набора данных для уменьшения времени распределенного обучения.

4. Заключение

В работе проведено исследование влияния гиперпараметров скорости обучения и размера батча для распределенной системы с различным количеством узлов(клиентов). Был найден набор гиперпараметров, при котором значения точности достаточно высокие для разного количества узлов в распределенной системе. Подтверждено предположение, что время распределенного обучения может быть меньше, чем при локальном глубоком обучении даже на достаточно небольшом датасете. При этом размер батча сильно влияет как на точность распределенного обучения, так и на его скорость. Перспективным направлением дальнейших исследований являются методы распределенного обучения с ограничениями на распространение данных и несбалансированными (non-IID) данными.

Финансовая поддержка: Работа выполнена в рамках государственного задания ИППИ РАН, утвержденного Минобрнауки России.

Список литературы

[1] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (2015) 436–444. doi:10.1038/nature14539.
[2] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, Q. V. Le, M. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang, A. Y. Ng, Large scale distributed deep networks, Advances in Neural Information Processing Systems (NeurIPS) 25 (2012). URL: https://proceedings.neurips.cc/paper/2012/hash/6aca97005c68f1206823815f66102863-Abstract.html.
[3] T. Ben-Nun, T. Hoefler, Demystifying parallel and distributed deep learning: An in-depth concurrency analysis, ACM Computing Surveys (CSUR) 52 (2019) 1–43. doi:10.1145/3320060.
[4] C. Xie, S. Koyejo, I. Gupta, Asynchronous Federated Optimization, in: Proceedings of the 3rd MLSys Conference, Austin, TX, 2020. URL: https://arxiv.org/abs/1903.03934.
[5] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, B. Agüera y Arcas, Communication-efficient learning of deep networks from decentralized data, in: Artificial intelligence and statistics, PMLR, 2017, pp. 1273–1282.
[6] S. Zawad, F. Yan, Hyperparameter tuning for federated learning–systems and practices, in: Q. Yang, L. Wang (Eds.), Federated Learning, Academic Press, 2024, pp. 219–235.
[7] Y. Zhou, P. Ram, T. Salonidis, N. Baracaldo, H. Samulowitz, H. Ludwig, FLoRA: Single-shot hyper-parameter optimization for federated learning, 2021. URL: https://arxiv.org/abs/2112.08524.
[8] S. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečný, S. Kumar, H. B. McMahan, Adaptive federated optimization, 2020. URL: https://arxiv.org/abs/2003.00295.
[9] H. Zhang, Z. Wu, W.-L. Tam, R. Li, Y.-C. Chang, S. Wang, K. Eykholt, Fedtune: Automatic tuning of federated learning hyper-parameters from system perspective, in: Proceedings of the 2022 IEEE Military Communications Conference (MILCOM), IEEE, 2022, pp. 478–483.
[10] M. Kundroo, T. Kim, Demystifying impact of key hyper-parameters in federated learning: A case study on CIFAR-10 and FashionMNIST, IEEE Access 12 (2024) 23811–23824. doi:10.1109/ACCESS.2024.3364239.
[11] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), IEEE Computer Society, 2016, pp. 770–778.
[12] Krizhevsky, Learning multiple layers of features from tiny images, Technical Report, University of Toronto, Toronto, ON, 2009.
[13] Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, Cambridge, MA, 2016.