Семинар **Математические основы искусственного интеллекта**

Материалы к докладу 14.02.2024
А.В. Гасников "Федеративное обучение. Фундаментальные аспекты. Открытые задачи"

Обзор по федеративному обучению (federated learning, FL).

Kairouz P. et al. *Advances and open problems in federated learning*, Foundations and Trends® in Machine Learning, 14:1–2 (2021), P. 1-210, arXiv:1912.04977.

Комментарии и задачи.

Простейшая постановка Woodworth et al. (2021) [1]. Пусть

$F : \mathcal{B} \to \mathbb{R}$ – выпуклая гладкая функция на единичном замкнутом шаре $\mathcal{B} \subset \mathbb{R}^p$ такая, что $F = \mathsf{E} f(\xi, \cdot)$ для некоторой гладкой $f : \mathbb{R}^d \times \mathcal{B} \to \mathbb{R}$ и случайного вектора ξ в $\mathbb{R}^d$,

F и ее градиент ∇F неизвестны, а функция ∇f, называемая *стохградиентом*, известна (∇ относительно x), и при этом $\nabla F(x) = \mathsf{E} \nabla f(\xi, x)$ для всякого x.

Требуется приближенно решить задачу

$$F(x) \to \min_x,$$

имея достаточное число независимых реализаций случайного вектора ξ.

Классическое решение основано на методе стохастического градиентного спуска (SGD)

$$x_{t+1} = x_t - \gamma_t \hat{\nabla} F_t(x_t)$$

с подходящим образом подобранными положительными параметрами γ_t и оценкой $\hat{\nabla} F_t$ градиента ∇F на шаге t, например,

$$\hat{\nabla} F_t(\cdot) = \frac{1}{n} \sum_{i=1}^{n} \nabla f(\xi_i^{(t)}, \cdot),$$

где $\xi_i^{(t)}$ – независимые копии (реализации) ξ.

Очевидно, вычисление $\hat{\nabla} F_t(\cdot)$ можно проводить параллельно. Представим, что для этого у нас есть m вычислительных узлов (машин), причем $m \ll n$. Конечно, возникает необходимость в коммуникации узлов, что может быть затратно. Но раз уж коммуникации

не избежать, то можно ли дополнительно ускорить вычисления? Естественная идея, по-видимому, впервые пришедшая Ю.Е. Нестерову около 20 лет назад, была описана в 2016 г. и получила название federated learning/федеративное обучение. Суть идеи в том, чтобы на каждом узле делать $K(=n/m)$ локальных шагов SGD[1], и далее, предполагая, что все узлы коммуницируют, вычислять следующее x_{t+1} как среднее арифметическое значений по результатам этих K локальных шагов. Потом процедура повторяется, стартуя с общей для всех точки x_{t+1}. Таким образом, батч размера $n = mK$ (набор значений стохградиентов) вычисляется не в одной точке, а как бы с локальным "заглядыванием" в будущее. Надежда на то, что такое "заглядывание" может ускорить скорость сходимости. В действительности, теория говорит следующее, что для гладкой выпуклой задачи

$$\mathsf{E}F(x_T) - F^* \leqslant c\Big(\frac{1}{T} + \frac{1}{\sqrt{mKT}}\Big) \qquad (*)$$

при всех T для некоторого $c > 0$, не зависящего от T, где F^* – минимум F (см. формулы (11)–(13) в [1]).

Можно показать, что точно такой же результат будет иметь место, если все m узлов не делают локальных шагов, а вычисляют K независимых стохградиентов в текущей точке x_t (общей для всех узлов), потом путем коммуникации вычисляют среднее арифметическое всех стохградиентов, и в заключении делают шаг SGD.

То есть получается, что по теории локальные шаги ничего не дают? На самом деле мы несколько огрубили настоящий результат для федеративного SGD (первое слагаемое в (*) правильнее писать $1/(K^{1/3}T^{2/3})$). Но общий вывод остается тем же. В общем случае, локальные шаги в теории и правда ничего не дают.

Аналогичный результат переносится и на ускоренные версии SGD (см. (10) в [1]), для которых немного улучшается первое слагаемое в правой части

$$\mathsf{E}F(x_T) - F^* \leqslant c\Big(\frac{1}{T^2} + \frac{1}{\sqrt{mKT}}\Big). \qquad (**)$$

Однако, если функция F квадратичная, то оценка улучшается очень существенно:

$$\mathsf{E}F(x_T) - F^* \leqslant c\Big(\frac{1}{(KT)^2} + \frac{1}{\sqrt{mKT}}\Big). \qquad (***)$$

По сути последний результат означает, что можно вообще не коммуницировать, а про-

[1]Используя независимые от других узлов реализации ξ и только один стохградиент (как при $n = 1$) при вычислении ∇F на каждом локальном шаге.

сто в самом конце посчитать среднее арифметическое по всем m траекториям. Качество будет такое же, как если бы коммуникации были постоянны.

Собственно, тут мы имеем интересное отличие общего гладкого выпуклого случая от квадратичного выпуклого случая. Интересно тут то, что не для распределенных процедур эти классы задач оптимизации с точки зрения сложности практически одинаковы. Поскольку локально любая разумная выпуклая функция хорошо аппроксимируется квадратичной, то мы вправе рассчитывать, что на практике эффект от локальных шагов всё-таки будет. Так оно и есть на самом деле!

Открытые задачи.

(1) Перенести результаты Woodworth et al. (2021) в части оптимальных алгоритмов на седловые задачи и вариационные неравенства на базе метода Juditsky et al. (2011) и его пробатченного варианта (в частности, с помощью идей из работы Безносикова и др. (2022)).

(2) Получить нижнюю оценку в архитектуре FL для вариационных неравенств, следуя идеям Woodworth et al. (2021). В других архитектурах подобные результаты получены Безносиковым и др. (2022).

(3) Получить аналог результатов раздела 3 работы Woodworth et al. (2020) в случае седловых билинейных задач и более общего класса вариационных неравенств с линейным векторным полем.

(4) Woodworth и Srebro (2021) впервые показали как ускоренные методы оптимизации уживаются с условиями интерполяции и процедурой батчинга (см. также Ilandarideva et al. (2023)). Однако, этого пока не сделано в архитектуре FL. Без условий интерполяции оптимальные методы для задач гладкой выпуклой оптимизации в архитектуре FL получены Woodworth et al. (2021). Задача состоит в том, чтобы получить аналогичные оптимальные методы в условиях интерполяции (см. также Qin et al. (2022).)

Некоторые элементраные упражнения, иллюстрирующие другие идеи из доклада.

Упражнение. Каждый из n узлов (машин) вычисляет некоторую величину из $[0,1]$ и передает её значение x_i на сервер, который считает среднее $n^{-1}\sum x_i$.

(1) Пусть все числа представлены в двоичном виде, напр., $0,11001\ldots$, а узлы передают только биты 0 или 1 (в одинаковом количестве). Сколько бит суммарно со всех узлов следует передать на сервер, чтобы посчитать среднее с точностью до 1/100 (одной сотой)?

(2) Предположим, что каждый узел вместо передачи x_i будет генерировать случайную бернуллиевскую величину X_i с $\mathsf{P}(X_i = 1) = x_i$, $\mathsf{P}(X_i = 0) = 1 - x_i$ и передавать её

значение X_i, в частности, сервер будет получать всего n бит. Покажите, что вероятность ошибиться более чем на 1/100 при оценке $n^{-1}\sum x_i$ средним $n^{-1}\sum X_i$ на сервере не превосходит $2\exp\{-n^2/5000\}$.

Указание: Воспользуйтесь неравенством Хёфдинга.

Замечание 1: Процедура (2), примененная в обучении со стохградиентами (при некоторой их нормировке/компрессии), позволяет в несколько раз снизить затраты на вычисления. При обучении ChatGPT удалось сэкономить десятки и даже сотни миллионов долларов на электроэнергии с помощью данного метода.

Замечание 2: В процедуре (2) индивидуальные значения X_i сами по себе практически не несут никакой информации о x_i, иначе говоря, в постановке с *перехватчиком* их нет смысла подслушивать.