ОПТИМИЗАЦИЯ ВЫЧИСЛИТЕЛЬНЫХ ЗАДАЧ МАШИННОГО ОБУЧЕНИЯ НА CPU

Соколов А. П.
YADRO
Москва, Российская Федерация
E-mail: andrey.sokolov@yadro.com

В работе рассматривается вопрос оптимизации задач машинного обучения и искусственного интеллекта на процессорах общего назначения (CPU). Перечислены некоторые причины, почему оптимизация данных задач актуальна для CPU. Показаны основные пути оптимизации: оптимизация математических алгоритмов, оптимизация моделей машинного обучения и, наконец, усиление вычислительных возможностей CPU за счет введения в их состав специализированных инструкций. Для каждого из путей показаны конкретные успешные примеры того, как удалось существенно повысить производительность задач машинного обучения на CPU.

Введение

В настоящее время задачи машинного обучения и искусственного интеллекта требуют все большего объема вычислительных ресурсов. Часто для решения этих задач используют специализированные вычислители. Например, графические карты (GPU). При этом процессоры общего назначения зачастую незаслуженно не рассматриваются. В тоже время есть ряд причин, почему CPU стоит рассматривать как платформу выбора для ряда задач машинного обучения.

Во-первых в CPU используется стандартная модель программирования. Это существенно снижает затраты на разработку и адаптацию алгоритмов под изменяющиеся требования. Во-вторых существует широчайший спектр CPU в смысле энергопотребления (от 10 мВт до 100 Вт при частоте 1 ГГц). Это позволяет более точно подобрать процессор под ту или иную задачу. В-третьих, CPU гораздо более доступны по сравнению со специализированными ускорителями, благодаря тому, что существует множество независимых разработчиков и производителей и архитектур (x86, ARM, RISC-V и др.). В-четвертых, на CPU-системах доступен гораздо больший объем оперативной памяти, чем на специализированных ускорителях. Например, на типовых серверных платформах доступно до 16 ТБ RAM, в то время как на GPU – максимум 128 ГБ. В ряде задач машинного обучения это требование является принципиальным.

Стандартные реализации задач машинного обучения могут обладать неудовлетворительными характеристиками производительности на CPU. Из общих соображений возможны следующие пути оптимизации их быстродействия:

- оптимизация математических алгоритмов, лежащих в основе задач машинного обучения;
- оптимизация используемых моделей;
- усиление возможностей самих CPU, например, за счет специализированных инструкций.

Далее, мы кратко рассмотрим некоторые успешные примеры оптимизации задач машинного обучения в каждом из этих направлений.

I. Оптимизация математических алгоритмов

Одним из ярких примеров того, как появление нового математического алгоритма позволяет значительно увеличить скорость решения практически важных задач машинного обучения, является применение метода Винограда для быстрого вычисления сверточных нейросетевых моделей (CNN).

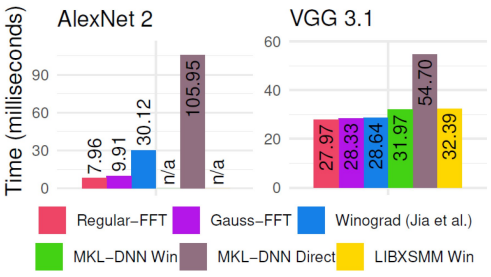

Рис. 1 – Время вычисления CNN моделей при помощи различных алгоритмов [1]

Другим примером эффективности оптимизации математических алгоритмов является фреймворк CatBoost. До появления этого фреймворка для обработки табличных данных в машинном обучении обычно использовались модели градиентного бустинга (GBDT), состоящие из решающих деревьев общего вида. Авторы фреймворка CatBoost предложили использовать симметричные решающие деревья [2]. В итоге модели CatBoost могут быть вычислены значительно быстрее, чем модели на решающих деревьях общего вида. В том числе на CPU.

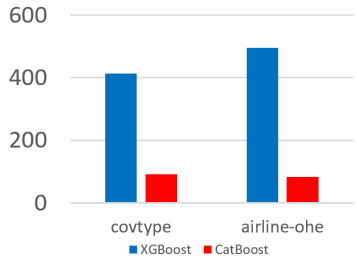

Рис. 2 – Время вычисления GBDT-ансамбля, обеспечивающего точность 95%, мс
Платформа: Intel(R) Xeon(R) Gold 6248R

II. Оптимизация моделей

Известно, что существует значительная избыточность информации в параметрах глубоких моделей. Вещественные веса нейросетей можно квантизовать в числа с низкой точностью (8 бит и менее) или вообще «занулять» (прунинг). При этом, итоговое качество модели практически не снижается. Это явление называется «перепараметризованностью» моделей машинного обучения. Сочетание прунинга и квантизации позволяет «сжимать» глубокие модели в десятки раз практически без потери качества.

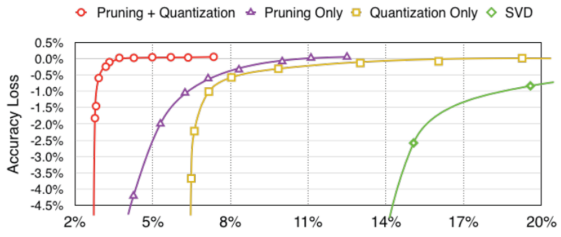

Рис. 3 – Потеря точности глубоких модели в зависимости от метода и степени сжатия [3]

Любопытно, что подобные возможности по существенному сжатию моделей имеются и в семействе моделей градиентного бустинга.

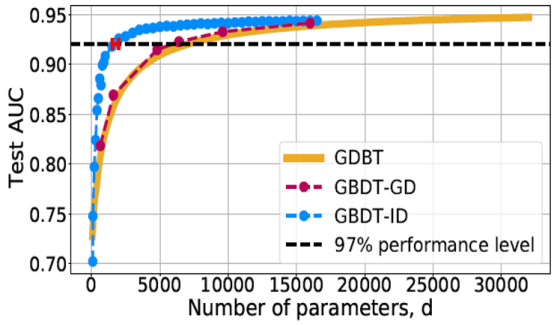

Рис. 4 – Внутренняя размерность GBDT ансамбля (датасет - Epsilon) [4]

III. Специализированные инструкции

Введение в состав CPU аппаратных блоков, реализующих дополнительные инструкции, является мощным инструментом, позволяющим существенно повысить производительность при решении определенных задач.

Так, например, расширение AVX-512 Core Advanced Matrix Extensions (AMX) вводит в состав процессора специализированный матричный ускоритель и набор инструкций для работы с ним. Данный ускоритель предназначен для более быстрого вычисления операции перемножения матриц, которая является очень востребованной в моделях машинного обучения.

Ниже на рисунке показан пример того, как соотносятся производительности процессоров Intel(R) на задаче детектирования лиц. При этом рассматриваются различные наборы специализированных инструкций, начиная от самых старых (AVX-512 Core) и заканчивая самыми современными (AVX-512 Core AMX).

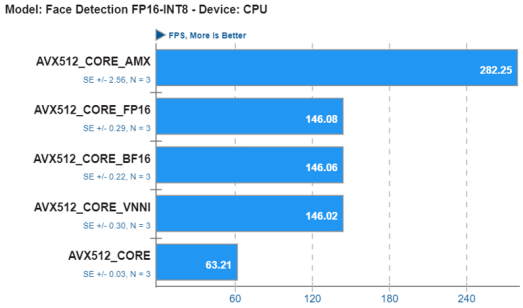

Рис. 5 – Сравнение производительности CPU Intel(R) с различными наборами специализированных инструкций [5]

IV. Заключение

Системы на базе CPU доступны, легко программируемы и могут быть быстро адаптированы под новые вычисления. Видно, что современные серверные процессоры успешно адаптируется к возрастающим запросам со стороны задач машинного обучения. Это позволяет им занимать существенную долю серверного сегмента. Оптимизировать задачи машинного обучения на CPU можно разными способами: искать новые и оптимизировать существующие алгоритмы обучения и исполнения моделей, оптимизировать сами модели, а также адаптировать возможности CPU за счет специальных расширений. Все это позволяет значительно сократить разрыв между CPU и специализированными ускорителями во многих практически-важных случаях.

1. Zlateski A., Zhen J., Kai L. and Fredo D., "The anatomy of efficient FFT and winograd convolutions on modern CPUs. Proceedings of the ACM International Conference on Supercomputing, 2019, pp. 414-424. doi:10.1145/3330345.3330382
2. Prokhorenkova L., Gusev G., Vorobev A., Dorogush A.V., Gulin A., "CatBoost: unbiased boosting with categorical features NeurIPS, 2018.
3. Han S., Mao H., Dally W.J., "Deep Compression: Compressing Deep Neural Network with Pruning, Trained Quantization and Huffman Coding ICLR, 2016.
4. Ivanov S., Litovchenko L., Prokhorenkova L., Sokolov A.P., "Tree Ensembles with Gradient Descent: Loss Landscape, Intrinsic Dimension and Split Permutation Invariance OpenTalks.AI, 2023.
5. `https://www.phoronix.com/review/intel-xeon-amx`