Аппаратное обеспечение для неиросетей

Д. А. Иванов[1]

Современный успех в области нейронных сетей во многом обусловлен наличием достаточного количества аппартных ресурсов. В работе проведен анализ основных современных аппаратных решений для ИИ (CPU, GPU, TPU), рассмотрены их преимущества и недостатки.

Ключевые слова: графический процессор, тензорный процессор, аппаратное обеспечение, нейронные сети

1. Введение

Большие наборы данных и достаточные аппартные ресурсы определили со- временный успех в области нейронных сетей. Именно отсутствие двух данных вещей не позволило появившимся в конце 80-х/начале 90-х годов архитектурам нейронных сетей завоевать популярность. Как хоршоо показано в работе [3] именно наличие подходящего железа выводит определенную технологию на передний план, а не только качество самой технлологии/идеи.

На сегодняшний день появляется огромное количество специализированных аппаратных решений для ИИ. Во многом это обуславливается ослаблением работы закона Мура и прекращение работы закона Деннарда, что в итоге лишает возможности опираться на постоянный рост вычислительных мощностей и дает большое поле для появления специализированных архитектур [2].

Данные DSP (Domain Specific Processors) решения могут быть сосредоточены как на решении только задачи inference сети, так и на задачах обучения, также они могут быть специально адаптрированы под определенные типы нейросетей или нацелены на решения проблема энергоэффективности. Далее мы сосредоточимся на трех архитектруах (CPU, GPU, TPU), которые могут быть использованы как для inference так и для обучения. Первые две являются повсеместно распространненными и являются основными вычислителями для работы с нейросетями.

[1] *Иванов Дмитрии Александрович* — аспирант кафедры суперкомпьютеров и квантовой информатики ф-та ВМК МГУ; эксперт группы неироморфных вычислений, ЧУ Цифрум, Росатом, e-mail: rudimiv@gmail.com

Ivanov Dmitry Alexandrovich — graduate student, Lomonosov Moscow State University, Faculty of CMC, Chair of Supercomputers and Quantum Informatics; Expert of the Neuromorphic Computing Group, PI Cifrum, Rosatom.

2. Нейронные сети с точки зрения архитектруы фон Неймана

Одним из принципов архитектры фон Неймана является разделение памяти и вычислений. И если данный подход себя оправдывал на заре компьютерной эры, то на сегодняшний день он привел к такому явлению как фон Неймановское бутылочное горлышко. В самом деле если посомтреть на энергозатраты и скорость доступа к памяти в регистрах/кэшах/RAM, то можно увидеть различия на порядки (см рис. 1).

При этом основной операцией в нейронных сетях является операция перемножения матрицы на вектор. $y = Wx$, $o = f(y)$. Для проведения данной операции требуется вначале получить данные из памяти, а именно $n2$ весов W и n значений вектора x. Причем, $n2$ весов будет использоваться единожды за эту операцию, в то время как значения из вектора x будут переиспользоваться. Ситуация отчасти облегчается, если используюся батчи, что однако не всегда возможно. Также ситуация может быть чуть проще, если происходит работа со сверточными операциями, в которых веса регулярно переиспользуются при расчете новой свертки.

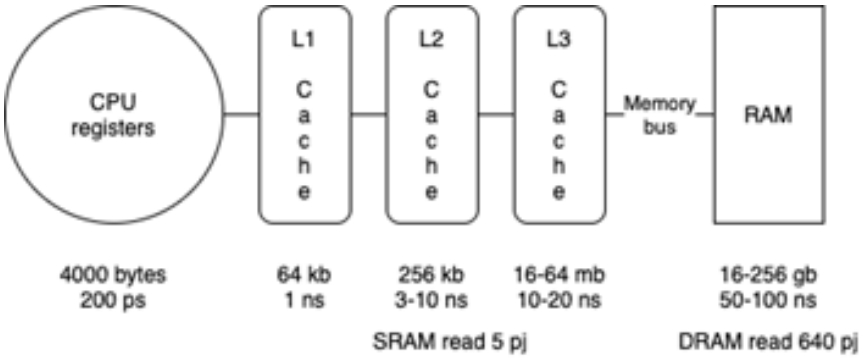

Рис. 1. Иерархия памяти.

3. CPU

Классически, проблема в скоростях доступа к памяти решалась в CPU с помощью сложной многоуровненовй системы кешей. Так в современных процессорах размер кешей может составлять 40 процентов площади чипа, обеспечивая десятки мегабайт сверхбыстрой памяти. Однако размер нейронных сетей и особенности их вычислений не позволяет уместить все веса в кэши.

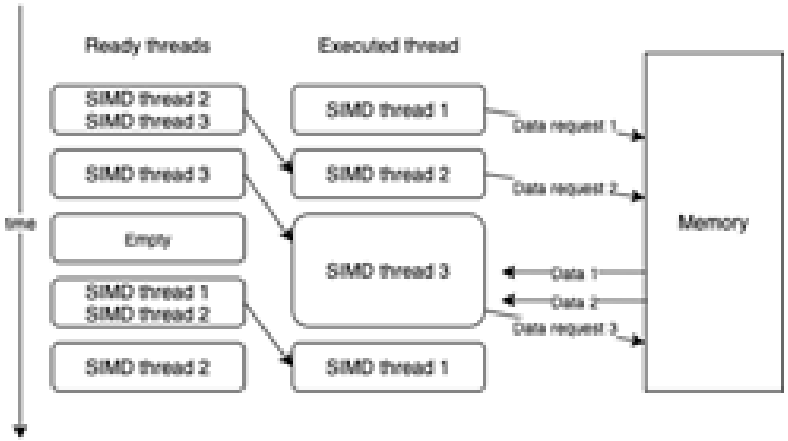

Рис. 2. Переключение между потоками в GPU

Другим традиционным подходом оптимизации процессоров были спекулятивные вычисления, предсказание ветвлений и тд. Однако перемножение матриц является процессом в котором порядок вычислений заранее известен и не требует таких сложных подходов к вычислениям, что делает данные механизмы бесполезными.

Таким образом CPU может подходить только для вычисления маленьких нейронных сетей, но никак не современных больших архитектру размером в сотни мегабайт.

4. GPU

В GPU используется несколько стратегий для борьбы с задержками памяти. Основная из них заключается в том, что на каждом вычислительном ядре (Streaming Multiprocessor) имеется большой file register который позволяет сохранять контекст выполнения для многих потоков и быстро между ними переключаться (см. рис. 2). Планировщик вычислений использует данную возможность и когда в каком-то из потоков инструкций (варпов) (SIMD thread 1) исполняется команда с большим latency, к примеру получение данных из памяти, то сразу же происходит переключение на другой поток инструкций (SIMD thread 2), при наступлении и в нем медленной команды, начинается выполнение нового готового потока инструкций (SIMD thread 3). Через какое-то время приходят данные для первого потока и он тоже становится готовым к исполнению. Таким образом происходит сокрытие latency памяти [1] (см. рис. 2). Причем таких Streaming Multiprocessorces в GPU могут быть десятки и даже больше сотни, которые делят между собой нагрузку.

Однако помимо latency важна также пропускная способность памяти, то есть количество данных, которое мы можем в пределе получать из памяти. Именно для оптимизации данной проблемы в видеокартах начиная с P100 (2016) Nvidia начала добавлять HBM (High Bandwidth Memory) память, что резко увеличило их производительность по сравнению с предыдущими поколениями. В архитектруах Volta и Turing Nvidia продолжала наращивать пропускную способность памяти, доведя в архитектуре A100 ее до 1.5 тб/с.

5. TPU

Первая версия процессора на данной архитектуре была анонсирована компанией Google в 2016 году и носит название TPUv1 [4]. Основным способом борьбы с задержками и низкой пропускной спообностью памяти является использование вместо кэшей так называемых систоллических матриц и программно управляемой памяти.

Идея систоллических вычислений заключается в создании большой матрицы (для tpuv1 — 256x256) вычислительных блоков, каждый из которых хранит внутри себя вес и выполняет следующие две операции: число x, которое пришло к нему от блока сверху, он умножает на вес и прибавляет полученное значение к числу, которое пришло к нему слева. Затем он отправляет число x, полученное сверху, своему соседу снизу, а полученную сумму отправляет соседу справа. Таким образом TPU выполняет перемножения матриц в конвейере. При достаточно большом размере батча ему не придется постоянно обращаться за весами в память, так как веса находятся в самих вычислительных блоках. При этом, при большом размере батча (больше ширины систоллического массива), TPU сможет каждый такт выдавать по одному результату перемножения матрицы 256x256 на вектор длины 256.

6. Заключение

Как мы видим одной из основных вычислительных проблем для систем ИИ является проблема доступа к памяти, с которой все предложенные варианты архитектур борятся с той или иной степенью успешности. Именно бутылочное горлышко фон Неймана дает нам проблемы задержек доступа к памяти (latency), пропускной способности памяти (bandwidth) и энергопотоpеблении. Развитие вычислительных мощностей для ИИ будет во многом определяться именно решением данной проблемы.

Список литературы

[1] Patterson D. A., Hennessy J. L., *Computer Organization and Design: The Hardware/Software Interface, ARM Edition*, Morgan Kaufmann, 2016, 720 pp.

[2] Hennessy J. L., Patterson D. A.., "A New Golden Age for Computer Architecture", *Communications of the ACM*, **62**:2 (2019), 48–60

[3] Hooker S., *The hardware lottery*, 2020, arXiv: 2009.06489.

[4] Jouppi N. P. et al., "A domain-specific architecture for deep neural networks", *Communications of the ACM*, **61**:9 (2018), 50–51.

Hardware for neural networks

Ivanov D.A.

The current success in the field of neural networks is largely due to the availability of a sufficient amount of hardware resources. The paper analyzes the main modern hardware solutions for AI (CPU, GPU, TPU), considers their advantages and disadvantages.

Keywords: GPU, TPU, hardware, neural networks

References

[1] Patterson D. A., Hennessy J. L., *Computer Organization and Design: The Hardware/Software Interface, ARM Edition*, Morgan Kaufmann, 2016, 720 pp.

[2] Hennessy J. L., Patterson D. A.., "A New Golden Age for Computer Architecture", *Communications of the ACM*, **62**:2 (2019), 48–60

[3] Hooker S., *The hardware lottery*, 2020, arXiv: 2009.06489.

[4] Jouppi N. P. et al., "A domain-specific architecture for deep neural networks", *Communications of the ACM*, **61**:9 (2018), 50–51.