Tensor Methods for Enhanced Computational Efficiency

James Nguyen
University of California, Irvine

November 21, 2024

Table of Contents (Summary)

What is a tensor?

Most simply, arrays generalized to higher dimensions.

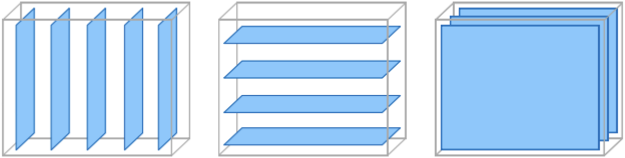

Figure 1: Order-3 tensor $\mathcal{X}_3 \in \mathbb{R}^{I_1 \times I_2 \times I_3}$

What is a tensor?

Most simply, arrays generalized to higher dimensions.

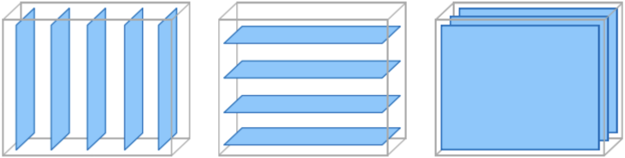

Figure 1: Order-3 tensor $\mathcal{X}_3 \in \mathbb{R}^{I_1 \times I_2 \times I_3}$

Scalars $c \in \mathbb{R}$, vectors $x \in \mathbb{R}^{I_1}$, and matrices $X \in \mathbb{R}^{I_1 \times I_2}$ are tensors of orders 0, 1, and 2, respectively.

Transforming Large Data

Consider square matrix multiplication. Let A, B be order-n square matrices.

Consider square matrix multiplication. Let A, B be order-n square matrices.

Case (1): n = 10

$$\begin{bmatrix} a_{1\,1} & \cdots & a_{1\,10} \\ \vdots & \ddots & \vdots \\ a_{10\,1} & \cdots & a_{10\,10} \end{bmatrix} \begin{bmatrix} b_{1\,1} & \cdots & b_{1\,10} \\ \vdots & \ddots & \vdots \\ b_{10\,1} & \cdots & b_{10\,10} \end{bmatrix}$$

Takes t seconds to compute.

Transforming Large Data

Consider square matrix multiplication. Let A, B be order-n square matrices.

Case (1): $n = 10$

$$\begin{bmatrix} a_{1\,1} & \cdots & a_{1\,10} \\ \vdots & \ddots & \vdots \\ a_{10\,1} & \cdots & a_{10\,10} \end{bmatrix} \begin{bmatrix} b_{1\,1} & \cdots & b_{1\,10} \\ \vdots & \ddots & \vdots \\ b_{10\,1} & \cdots & b_{10\,10} \end{bmatrix}$$

Takes t seconds to compute.

Case (2): $n = 100$

$$\begin{bmatrix} a_{1\,1} & \cdots & a_{1\,100} \\ \vdots & \ddots & \vdots \\ a_{100\,1} & \cdots & a_{100\,100} \end{bmatrix} \begin{bmatrix} b_{1\,1} & \cdots & b_{1\,100} \\ \vdots & \ddots & \vdots \\ b_{100\,1} & \cdots & b_{100\,100} \end{bmatrix}$$

Takes $100t$ seconds to compute.

Efficient Matrix Multiplication

The product of square order-2 matrices can be computed with just 7 multiplications.

Efficient Matrix Multiplication

The product of square order-2 matrices can be computed with just 7 multiplications.

$$
\begin{aligned}
I &= (a_{11} + a_{22})(b_{11} + b_{22}), \\
II &= (a_{21} + a_{22})b_{11}, \\
III &= a_{11}(b_{12} - b_{22}), \\
IV &= a_{22}(-b_{11} + b_{21}),
\end{aligned}
$$

$$
\begin{aligned}
V &= (a_{11} + a_{12})b_{22}, \\
VI &= (-a_{11} + a_{21})(b_{11} + b_{12}), \\
VII &= (a_{12} - a_{22})(b_{21} + b_{22})
\end{aligned}
$$

Efficient Matrix Multiplication

The product of square order-2 matrices can be computed with just 7 multiplications.

$$\begin{aligned} I &= (a_{11} + a_{22})(b_{11} + b_{22}), \\ II &= (a_{21} + a_{22})b_{11}, \\ III &= a_{11}(b_{12} - b_{22}), \\ IV &= a_{22}(-b_{11} + b_{21}), \end{aligned}$$

$$\begin{aligned} V &= (a_{11} + a_{12})b_{22}, \\ VI &= (-a_{11} + a_{21})(b_{11} + b_{12}), \\ VII &= (a_{12} - a_{22})(b_{21} + b_{22}) \end{aligned}$$

$$AB = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} = \begin{bmatrix} (I + IV - V + VII) & (II + IV) \\ (III + IV) & (I + III - II + VI) \end{bmatrix}$$

Idea

By sub-matrix multiplication, Strassen's Algorithm scales for order-$m2^k$ matrices.

$$AB = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$$

Idea

By sub-matrix multiplication, Strassen's Algorithm scales for order-$m2^k$ matrices.

$$AB = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$$

$$A = \begin{bmatrix} \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} & \dots & \begin{bmatrix} a_{1\ n-1} & a_{1\ n} \\ a_{2\ n-1} & a_{2\ n} \end{bmatrix} \\ \vdots & \ddots & \vdots \\ \begin{bmatrix} a_{n-1\ 1} & a_{n-1\ 2} \\ a_{n\ 1} & a_{n\ 2} \end{bmatrix} & \dots & \begin{bmatrix} a_{n-1\ n-1} & a_{n-1\ n} \\ a_{n\ n-1} & a_{n\ n} \end{bmatrix} \end{bmatrix}$$

Idea

By sub-matrix multiplication, Strassen's Algorithm scales for order-$m2^k$ matrices.

$$AB = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$$

Theorem (Master Theorem for Divide and Conquer Recursive Algorithms)

If combining sub-problems costs $f(n) \leq O(n^{\log_b a})$, for reduction factor b and number of sub-problems a, then total complexity $T(n) = O(n^{\log_b a})$.

New Limits

Previously, square matrix multiplication: $T(n) = O(n^3)$

Now, Strassen's Algorithm: $T(n) = O(n^{\log_2 7}) \approx O(n^{2.81})$

New Limits

Previously, square matrix multiplication: $T(n) = O(n^3)$

Now, Strassen's Algorithm: $T(n) = O(n^{\log_2 7}) \approx O(n^{2.81})$

Shmuel Winograd's Lower Bound

Order-2 matrix multiplication requires at least 7 multiplications.

What about order-2^n matrices?

Interaction

Definition (Tensor Product)

Let U, V be vector spaces. Then the tensor product $U \otimes V$ is, itself, a vector space where $\mathcal{X} \in U \otimes V$ is a tensor. Additionally, $\exists\ \pi : U \times V \mapsto U \otimes V$.

Theorem (Universal Property of Tensor Products)

Let $f : U \times V \mapsto W$ be a bilinear map. Then there exists a unique homomorphism $\phi : U \otimes V \mapsto W$ such that $f = \phi \circ \pi$.

$$\begin{tikzcd} U \times V \arrow[r, "f"] \arrow[d, "\pi"'] & W \\ U \otimes V \arrow[ur, "\phi"'] & \end{tikzcd}$$

Deepmind: AlphaTensor

Note: Matrix multiplication is a bilinear map

$$\implies \exists\, \pi : \mathbb{R}^{m\times n} \times \mathbb{R}^{n\times p} \mapsto \mathbb{R}^{m\times n\times p},\ \phi : \mathbb{R}^{m\times n\times p} \mapsto \mathbb{R}^{m\times p}.$$

Then every algorithm can be represented by some order-3 tensor $\mathcal{T}_{m,n,p}$ such that $\phi \circ \pi(A, B) = \phi(\mathcal{T}_{m,n,p}) = C$, where

$$\mathcal{T}_{m,n,p} = \sum_{r=1}^{R} \mathbf{u}^{(r)} \otimes \mathbf{v}^{(r)} \otimes \mathbf{w}^{(r)}$$

and R denotes $rank(\mathcal{T}_{m,n,p})$ and number of multiplications.

Breakthrough in Efficient Operations

$$
\begin{aligned}
I &= {\color{red}(a_1 + a_4)}{\color{blue}(b_1 + b_4)} \\
II &= (a_3 + a_4)b_1 \\
III &= a_1(b_2 - b_4) \\
IV &= a_4(b_3 - b_1) \\
V &= (a_1 + a_2)b_4 \\
VI &= (a_3 - a_1)(b_1 + b_2) \\
VII &= (a_2 - a_4)(b_3 + b_4) \\
C_{11} &= {\color{purple}I + IV - V + VII} \\
C_{12} &= I + III \\
C_{21} &= II + IV \\
C_{22} &= I - II + III + VI
\end{aligned}
$$

$$
A = \begin{bmatrix} {\color{red}1} & 0 & 1 & 0 & 1 & -1 & 0 \\ {\color{red}0} & 0 & 0 & 0 & 1 & 0 & 1 \\ {\color{red}0} & 1 & 0 & 0 & 0 & 1 & 0 \\ {\color{red}1} & 1 & 0 & 1 & 0 & 0 & -1 \end{bmatrix},
$$

$$
B = \begin{bmatrix} {\color{blue}1} & 1 & 0 & -1 & 0 & 1 & 0 \\ {\color{blue}0} & 0 & 1 & 0 & 0 & 1 & 0 \\ {\color{blue}0} & 0 & 0 & 1 & 0 & 0 & 1 \\ {\color{blue}1} & 0 & -1 & 0 & 1 & 0 & 1 \end{bmatrix},
$$

$$
C = \begin{bmatrix} {\color{purple}1} & {\color{purple}0} & {\color{purple}0} & {\color{purple}1} & {\color{purple}-1} & {\color{purple}0} & {\color{purple}1} \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & -1 & 1 & 0 & 0 & 1 & 0 \end{bmatrix}.
$$

Tensor Factor Representation of Strassen's Order-2 Matrix Multiplication Algorithm

Discovered order-4 algorithm for only 47 multiplications.

$\implies f(n) = O(n^{\log_4 47}) \approx O(n^{2.77}) < O(n^{\log_2 7})$.