You could've invented Strassen's Algorithm

Zeke

July 2025

Matrix Multiplication

Given two 2×2 matrices A, B

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}, \qquad B = \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix}$$

Matrix Multiplication

Given two 2×2 matrices A, B

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}, \qquad B = \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix}$$

Their product $C = AB$ is

$$C = \begin{pmatrix} a_{11}b_{11} + a_{12}b_{21} & a_{11}b_{12} + a_{12}b_{22} \\ a_{21}b_{11} + a_{22}b_{21} & a_{21}b_{12} + a_{22}b_{22} \end{pmatrix}$$

Matrix Multiplication

Given two 2×2 matrices A, B

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}, \qquad B = \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix}$$

Their product $C = AB$ is

$$C = \begin{pmatrix} a_{11}b_{11} + a_{12}b_{21} & a_{11}b_{12} + a_{12}b_{22} \\ a_{21}b_{11} + a_{22}b_{21} & a_{21}b_{12} + a_{22}b_{22} \end{pmatrix}$$

But computing C requires 8 multiplications. Can we do better?

Strassen's Algorithm

Remarkably, yes. C can be computed with 7 multiplications.

$$\begin{aligned}
P_1 &= (a_{11} + a_{22})(b_{11} + b_{22}) \\
P_2 &= (a_{21} + a_{22})b_{11} \\
P_3 &= a_{11}(b_{12} - b_{22}) \\
P_4 &= a_{22}(b_{21} - b_{11}) \\
P_5 &= (a_{11} + a_{12})b_{22} \\
P_6 &= (a_{21} - a_{11})(b_{11} + b_{12}) \\
P_7 &= (a_{12} - a_{22})(b_{21} + b_{22})
\end{aligned}$$

$$C = \begin{pmatrix} P_1 + P_4 - P_5 + P_7 & P_3 + P_5 \\ P_2 + P_4 & P_1 - P_2 + P_3 + P_6 \end{pmatrix}$$

My Hackathon Project

- But this is arcane...

$$C = \begin{pmatrix} P_1 + P_4 - P_5 + P_7 & P_3 + P_5 \\ P_2 + P_4 & P_1 - P_2 + P_3 + P_6 \end{pmatrix}$$

My Hackathon Project

- But this is arcane...

$$C = \begin{pmatrix} P_1 + P_4 - P_5 + P_7 & P_3 + P_5 \\ P_2 + P_4 & P_1 - P_2 + P_3 + P_6 \end{pmatrix}$$

- What if we could automatically discover this with a computer program?

My Hackathon Project

- But this is arcane...

$$C = \begin{pmatrix} P_1 + P_4 - P_5 + P_7 & P_3 + P_5 \\ P_2 + P_4 & P_1 - P_2 + P_3 + P_6 \end{pmatrix}$$

- What if we could automatically discover this with a computer program?
- Would we learn something about optimizing machine learning workloads along the way?

An algebra problem

1. Let $\bar{a}$ be $(a_{11}, a_{12}, a_{21}, a_{22})$ and $\bar{b}$, $\bar{c}$ be likewise.
2. Neither $a_{ij}a_{kl}$ nor $b_{ij}b_{kl}$ appears in C. So, the kth product will be of the form

$$P_k = (\sum_i \bar{a}_i \mu_i^{(k)})(\sum_j \bar{b}_j \varphi_j^{(k)})$$

where $\mu_i^{(k)}, \varphi_i^{(k)} \in \{0, 1\}$ are 1 if and only if $\bar{a}_i/\bar{b}_j$ appear in the kth product.

An algebra problem

1. $\bar{c}_l$ will be a combination of P_ks.

An algebra problem

1. $\bar{c}_l$ will be a combination of P_ks.
2. Letting $w_l^{(k)} \in \{-1, 0, 1\}$ denote P_k's coefficient in the expression for $\bar{c}_l$,

$$\bar{c}_l = \sum_k w_l^{(k)} P_k$$

Putting it together

$$
\begin{aligned}
\bar{c}_l &= \sum_k w_l^{(k)} P_k \\
&= \sum_k w_l^{(k)} (\sum_i \bar{a}_i \mu_i^{(k)})(\sum_j \bar{b}_j \varphi_j^{(k)}) \\
&= \sum_k w_l^{(k)} \sum_i \sum_j \bar{a}_i \bar{b}_j \mu_i^{(k)} \varphi_j^{(k)} \\
&= \sum_i \sum_j \bar{a}_i \bar{b}_j \sum_k w_l^{(k)} \mu_i^{(k)} \varphi_j^{(k)}
\end{aligned}
$$

8 multiply equivalence

1. To make this an algebra problem, we need a formula for $\bar{c}_I$.

8 multiply equivalence

1. To make this an algebra problem, we need a formula for $\bar{c}_l$.
2. Recall, $\bar{c}$ is a flattened version of

$$C = \begin{pmatrix} \bar{a}_1\bar{b}_1 + \bar{a}_2\bar{b}_3 & \bar{a}_1\bar{b}_2 + \bar{a}_2\bar{b}_4 \\ \bar{a}_3\bar{b}_1 + \bar{a}_4\bar{b}_3 & \bar{a}_3\bar{b}_2 + \bar{a}_4\bar{b}_4 \end{pmatrix}$$

8 multiply equivalence

1. To make this an algebra problem, we need a formula for $\bar{c}_l$.
2. Recall, $\bar{c}$ is a flattened version of

$$C = \begin{pmatrix} \bar{a}_1 \bar{b}_1 + \bar{a}_2 \bar{b}_3 & \bar{a}_1 \bar{b}_2 + \bar{a}_2 \bar{b}_4 \\ \bar{a}_3 \bar{b}_1 + \bar{a}_4 \bar{b}_3 & \bar{a}_3 \bar{b}_2 + \bar{a}_4 \bar{b}_4 \end{pmatrix}$$

3. Leting T_{ijl} be 1 if $\bar{a}_i \bar{b}_j$ appears in the above expression for $\bar{c}_l$ and 0 otherwise gives a formula for $\bar{c}_l$ as desired.

$$\bar{c}_l = \sum_i \sum_j \bar{a}_i \bar{b}_j T_{ijl}$$

Putting it together

Because

$$\bar{c}_l = \sum_i \sum_j \bar{a}_i \bar{b}_j \sum_k w_l^{(k)} \mu_i^{(k)} \varphi_j^{(k)}$$

and

$$\bar{c}_l = \sum_i \sum_j \bar{a}_i \bar{b}_j T_{ijl}$$

we have

$$T_{ijl} = \sum_k w_l^{(k)} \mu_i^{(k)} \varphi_j^{(k)}$$

Putting it together

Because

$$\bar{c}_l = \sum_i \sum_j \bar{a}_i \bar{b}_j \sum_k w_l^{(k)} \mu_i^{(k)} \varphi_j^{(k)}$$

and

$$\bar{c}_l = \sum_i \sum_j \bar{a}_i \bar{b}_j T_{ijl}$$

we have

$$T_{ijl} = \sum_k w_l^{(k)} \mu_i^{(k)} \varphi_j^{(k)}$$

an algebra problem, the solution of which corresponds to a way to compute $C = AB$ using k multiplies.

Conclusion

1. If you put this into a computer program for solving algebra problems, it finds Strassen's algorithm. :)
2. Have we learned something deep about optimizing machine learning workloads?

Conclusion

1. If you put this into a computer program for solving algebra problems, it finds Strassen's algorithm. :)
2. Have we learned something deep about optimizing machine learning workloads? ¯_(ツ)_/¯

Conclusion

1. If you put this into a computer program for solving algebra problems, it finds Strassen's algorithm. :)
2. Have we learned something deep about optimizing machine learning workloads? ¯_(ツ)_/¯
3. Thanks for having me!