What is...machine learning in mathematics - part 2?

Or: AI and matrix multiplication

Standard matrix multiplication is not optimal!

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix} \times \begin{pmatrix} E & F \\ G & H \end{pmatrix} = \begin{pmatrix} AE+BG & AF+BH \\ CE+DG & CF+DH \end{pmatrix}$$

Trick:

$P_1 = A \cdot (F-H)$

$P_2 = (A+B) \cdot H$

$P_3 = (C+D) \cdot E$

$P_4 = D \cdot (G-E)$

$P_5 = (A+D) \cdot (E+H)$

$P_6 = (B-D) \cdot (G+H)$

$P_7 = (A-C) \cdot (E+F)$

$AE+BG = P_5 + P_4 - P_2 + P_6$

$AF+BH = P_1 + P_2$

$CE+DG = P_3 + P_4$

$CF+DH = P_5 + P_1 - P_3 - P_7$

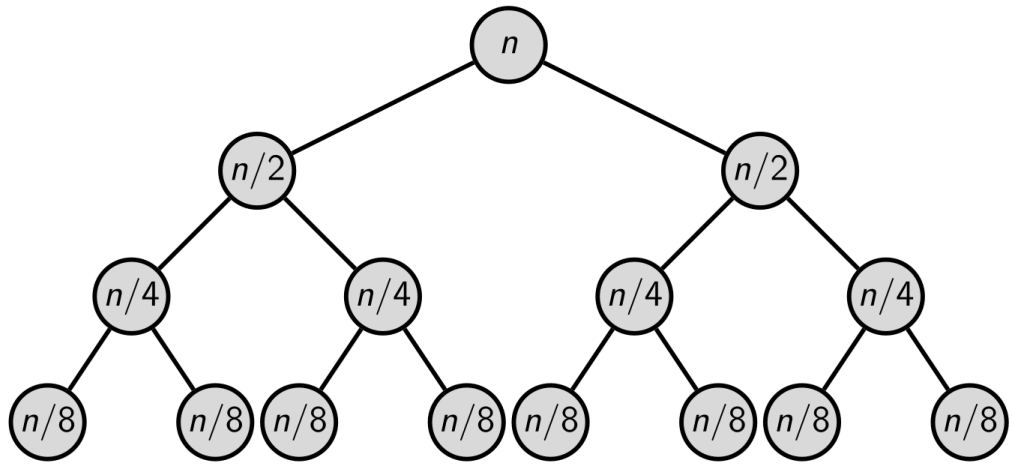
- Standard 2-by-2 matrix multiplication needs eight multiplications
- Strassen We only need seven
- Observation The seven multiplications are not trivial to find

Divide and conquer

- Break the matrices in blocks of size $n/2 \times n/2$ **Divide!**

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix} = \begin{pmatrix} S1 + S4 - S5 + S7 & S3 + S5 \\ S2 + S4 & S1 - S2 + S3 + S6 \end{pmatrix}$$

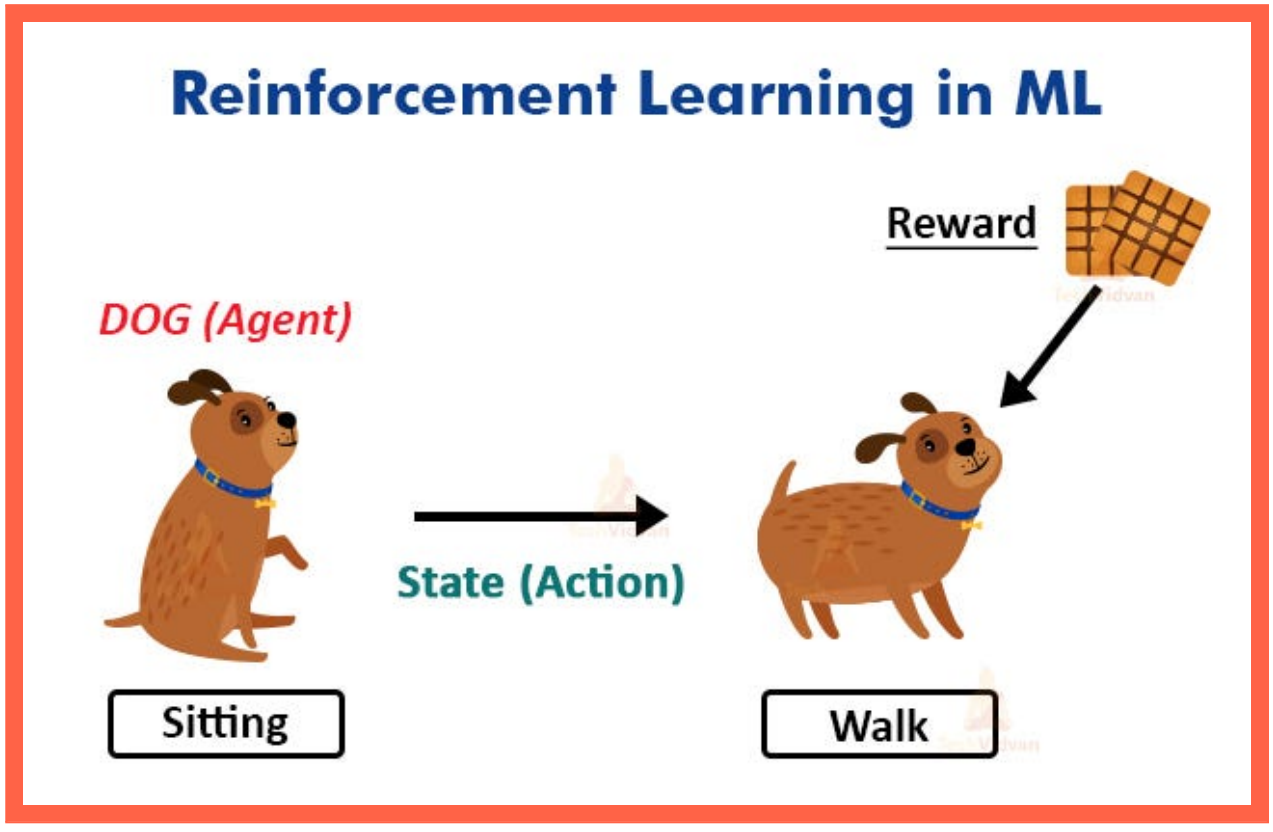
- Needs 7 calls on $n/2 \times n/2$ matrices
- Repeat recursively **Conquer!**



Strassen then needs $\approx n^{\log_2(7)}$ operations

- **Recursion** Safe one operation each step while multiplying 2^n-by-2^n matrices
- **Standard multiplication** Needs n^3 operations ($3 = \log_2 8$)
- **Strassen multiplication** Needs $n^{2.81}$ operations ($2.81 \approx \log_2 7$)

Why not k-by-k matrices?



- Idea Why don't we use large matrices for the recursion (difficult – remember how weird Strassen's formulas look!)
- Example If for 3-by-3 matrices we safe 7 operations, then we are down to $2.73 \approx \log_3 20$
- Idea Use reinforcement learning, let AI play a game and reward saving operations

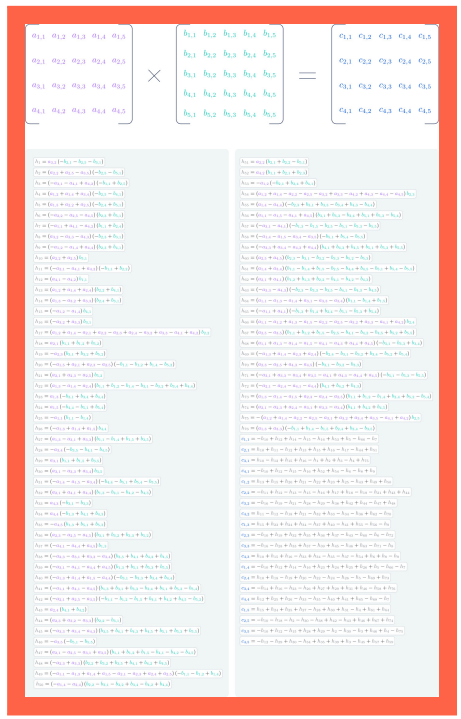
Enter, the theorem

Deepmind's AlphaTensor found a 5-by-5 algorithm with 76 operations

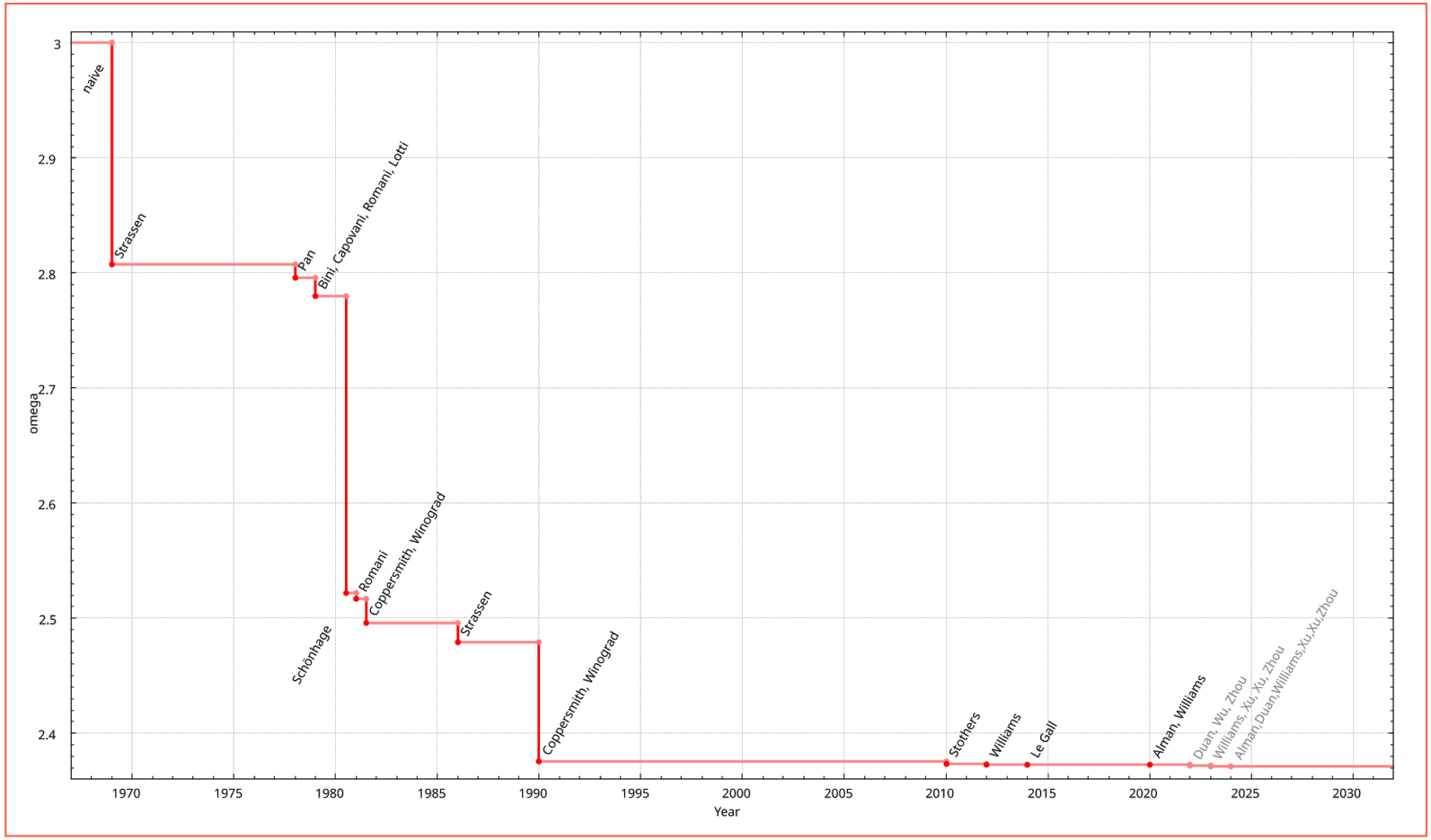
$$\log_5 76 \approx 2.69$$

The neutral network used is not a traditional one but used reinforcement learning

- They actually discovered several other algorithms as well
- Here is the list Don't look ☺



Actually...



- There are actually faster algorithms
- Most of the newer ones use different methods
- Problem Most of them are not of practical importance

Thank you for your attention!

I hope that was of some help.