This is a personal TLDR of the following paper
Discovering faster matrix multiplication algorithms with RL - Deepmind

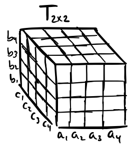
The crux of the paper is a perspective shift on matrix multiplications

Ex. $\begin{pmatrix} a_1 & a_2 \\ a_3 & a_4 \end{pmatrix}\begin{pmatrix} b_1 & b_2 \\ b_3 & b_4 \end{pmatrix} = \begin{pmatrix} c_1 & c_2 \\ c_3 & c_4 \end{pmatrix}$

now clearly $c_1 = a_1 b_1 + a_2 b_3$ (1)
$\vdots$
$c_4 = a_3 b_2 + a_4 b_4$

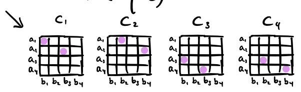
The authors represent the eqs (1) as a binary tensor T_n

$T_{2\times2}$



where the cell at (a_i, b_i, c_i) is 1 if $c_i = a_i b_i + \ldots$
0 otherwis

For example, in the 2x2 case above,



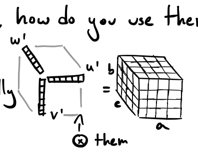
Now the cool part of their algo is through finding decompositions of T_n, they implicitly find diff procedures to perform the mat mul.

$T_n = \sum_{r=1}^{R} u^{(r)} \otimes v^{(r)} \otimes w^{(r)}$, $\otimes$ is outer product

↳ this is similar to SVD of $A^{n\times n}$ but here we are doing it for $A^{n\times n\times n}$.

So imagine $U, V, W \in \{-1,0,1\}^{n^2 \times R}$ are given, how do you use them?

$T_n = u^{(1)} \otimes v^{(1)} \otimes w^{(1)} + \ldots$ geometrically



therefore, us correspond to a
vs correspond to b
ws correspond to c

Thus the algo to actually perform mat mul given U, V, W is

Algo 1

for $r=1$ to R do
  $m_r \leftarrow (u_1^{(r)} a_1 + \ldots + u_{n^2}^{(r)} a_{n^2})(v_1^{(r)} b_1 + \ldots + v_{n^2}^{(r)} b_{n^2})$

for $i = 1, \ldots, n^2$ do
  $c_i \leftarrow w_i^{(1)} m_1 + \ldots + w_i^{(R)} m_R$

return C

The RL agent has to find u, v, w and the authors construct a game, which is

Let $S_0 = T_n$, the agent outputs (u', v', w'), thus $S_1 = S_0 - u^{(1)} \otimes v^{(1)} \otimes w^{(1)}$ and so on until $S_t = 0$. The agent is penalized for each step to encourage it to find shortest path (minimize R). The RL algo is based on AlphaZero where they do MCTS with a policy and value network.