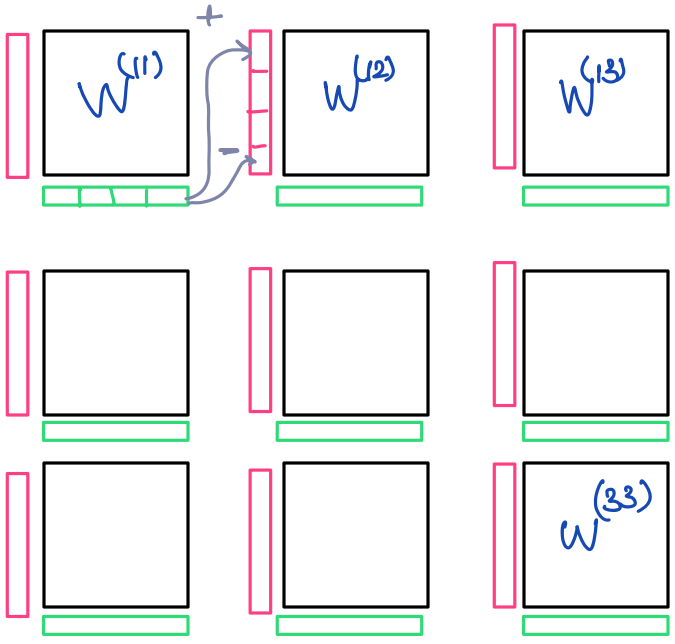
Mathematical setup of training the chip architecture.



256x256

Let $W^{ij} \in \mathbb{R}$ be the weight matrix in a core C_{ij}

Let V^{ijkpq} be the connectivity matrix

V^{ijkpq}

- $ij \rightarrow$ core id of source
- $k \rightarrow$ "chunk" (64) of the output vector $(k = 1,2,3,4)$
- $p,q \rightarrow$ core id of destination $(p == i)$ or $(q == j)$ $(s = 1,2,3,4)$

V^{ijkpq} is NOT trainable!

Make sure that the connections where $(p != i)$ and $(q != j)$ are zero

Let $U_p \longmapsto$ takes $y^{(lv)}$ and constructs $x_p^{(lv)}$ ($y^{(lv)}$ of one output slot is $x^{(lv)}$ of input slot of another core)

$U_N \longmapsto$ takes $y^{(lv)}$ & constructs $x_N^{(lv)}$

U_p & U_N together must produce two vectors:

$$x_p = \begin{bmatrix} x_p^{(1)} \\ \vdots \\ x_p^{(4)} \end{bmatrix}, \quad x_N = \begin{bmatrix} x_N^{(1)} \\ \vdots \\ x_N^{(4)} \end{bmatrix} \quad \text{vstack}$$

the goal is to do $y^{ij} = f\left(W^{ij}\left(x_p^{ij} - x_N^{ij}\right)\right)$

dot product

if some core C_{ab} is designated as output core,
we want to solve:

$$\underset{W^{ij}}{\operatorname{argmin}}\ \mathcal{L} = \text{Loss}\left(\underbrace{\text{targets}}_{\text{for MNIST: } \mathbb{R}^{10} \text{ vector}}, \text{probits}^{ab}\right)$$

At input layer:

Split x into two identical parts x_p, x_n composed of $x_p^{(k)}$ & $x_n^{(k)}$ & scramble the order.

HOW TO IMPLEMENT USING JAX??