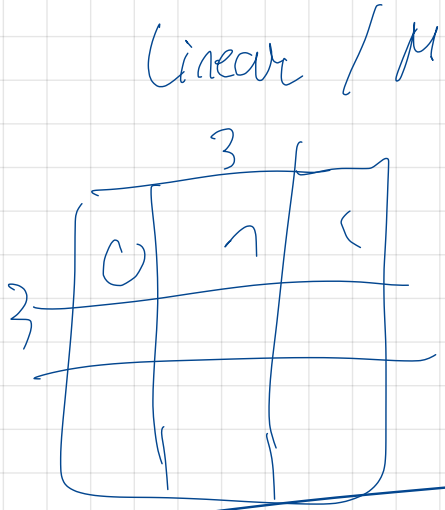
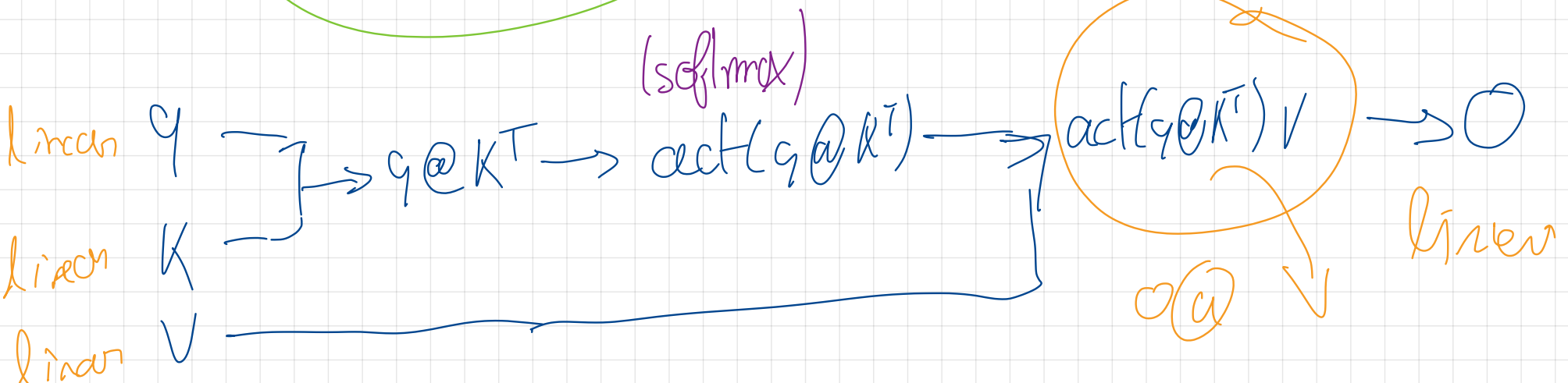
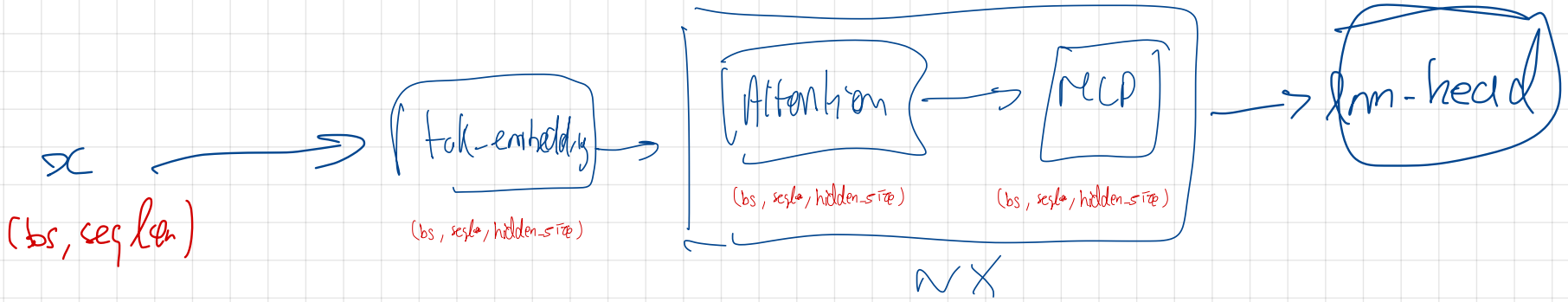
COURS 1


Linear / MLP
3
3

MP-proj
3x3
x
LINEAR
act
LINEAR
3x3
Attention
MLP
Nx
decoder


(softmax)
linear Q
linear K
linear V
q@K^T
act(q@K^T)
act(q@K^T)V
@
@V
linear



seq. len

tokenizer()

Embeddig

[Jaime, les, Rois] → [72, 4096, 8]

[Je, suis, Rendeo] [10, 96, 2]

(2, 3)

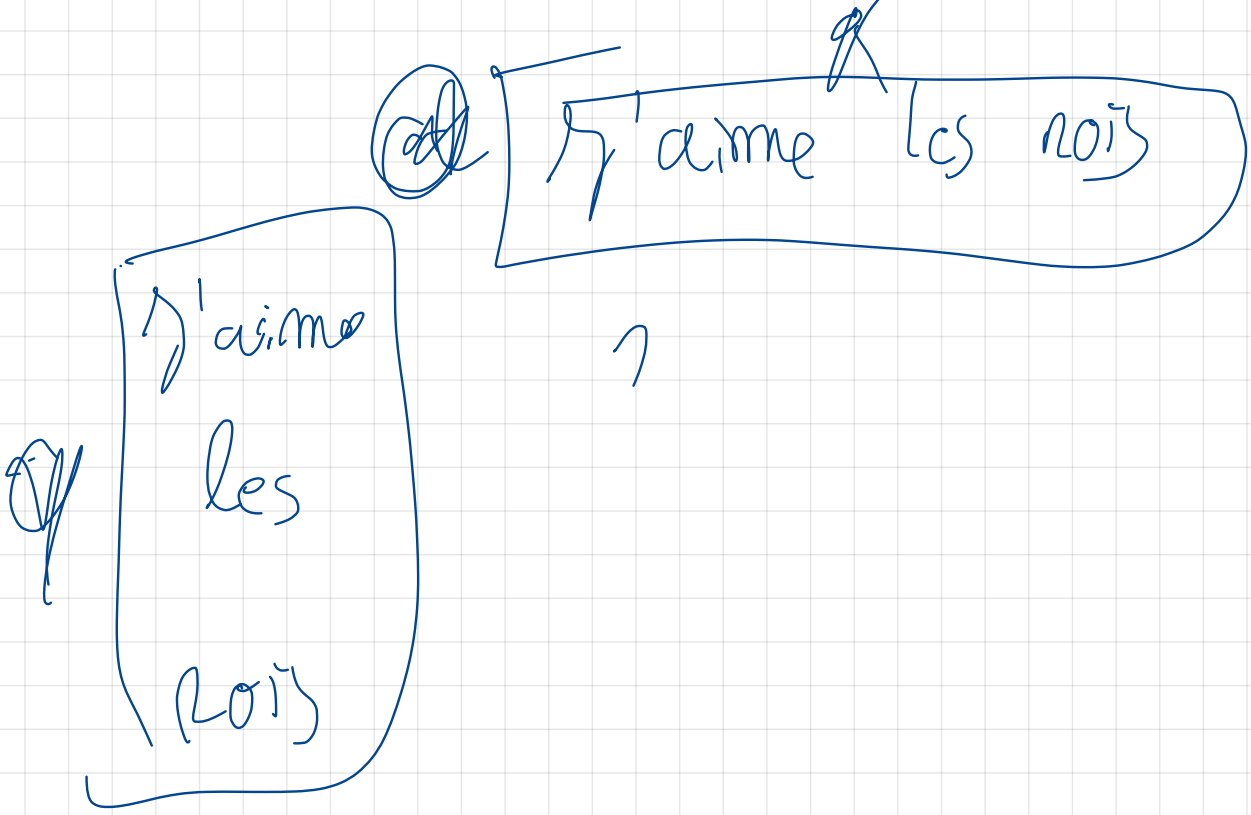
[2, 3, 4096]

0 2 :

1 2 :

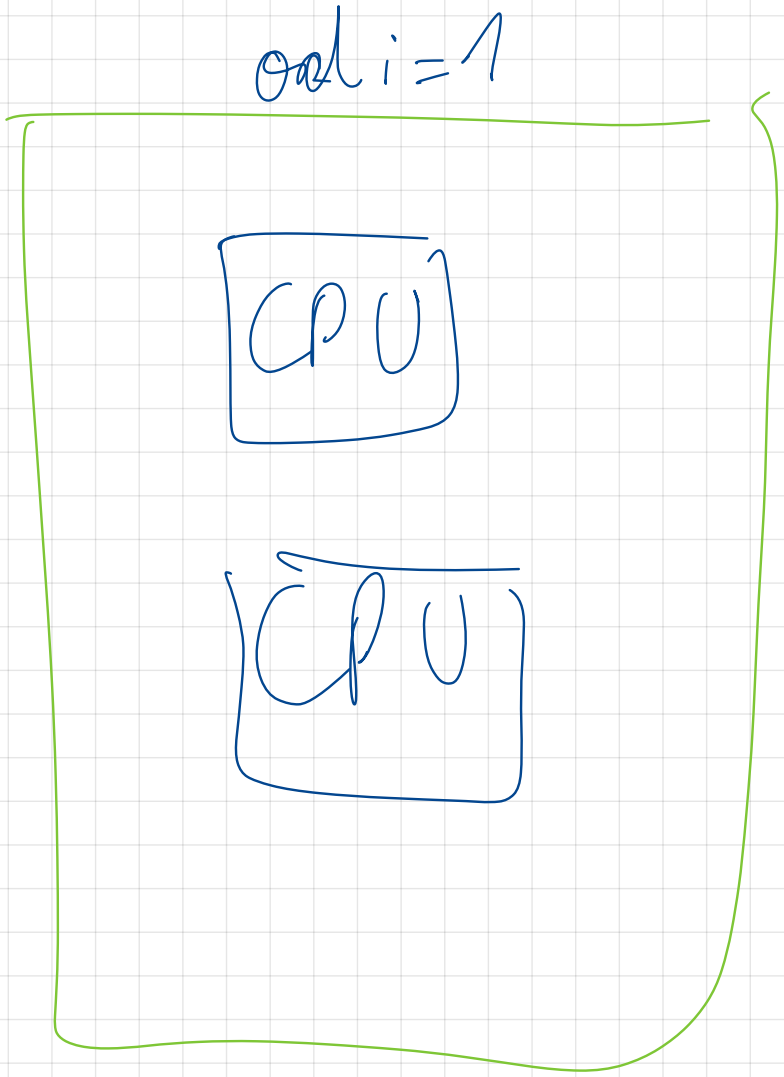
batch size = 2

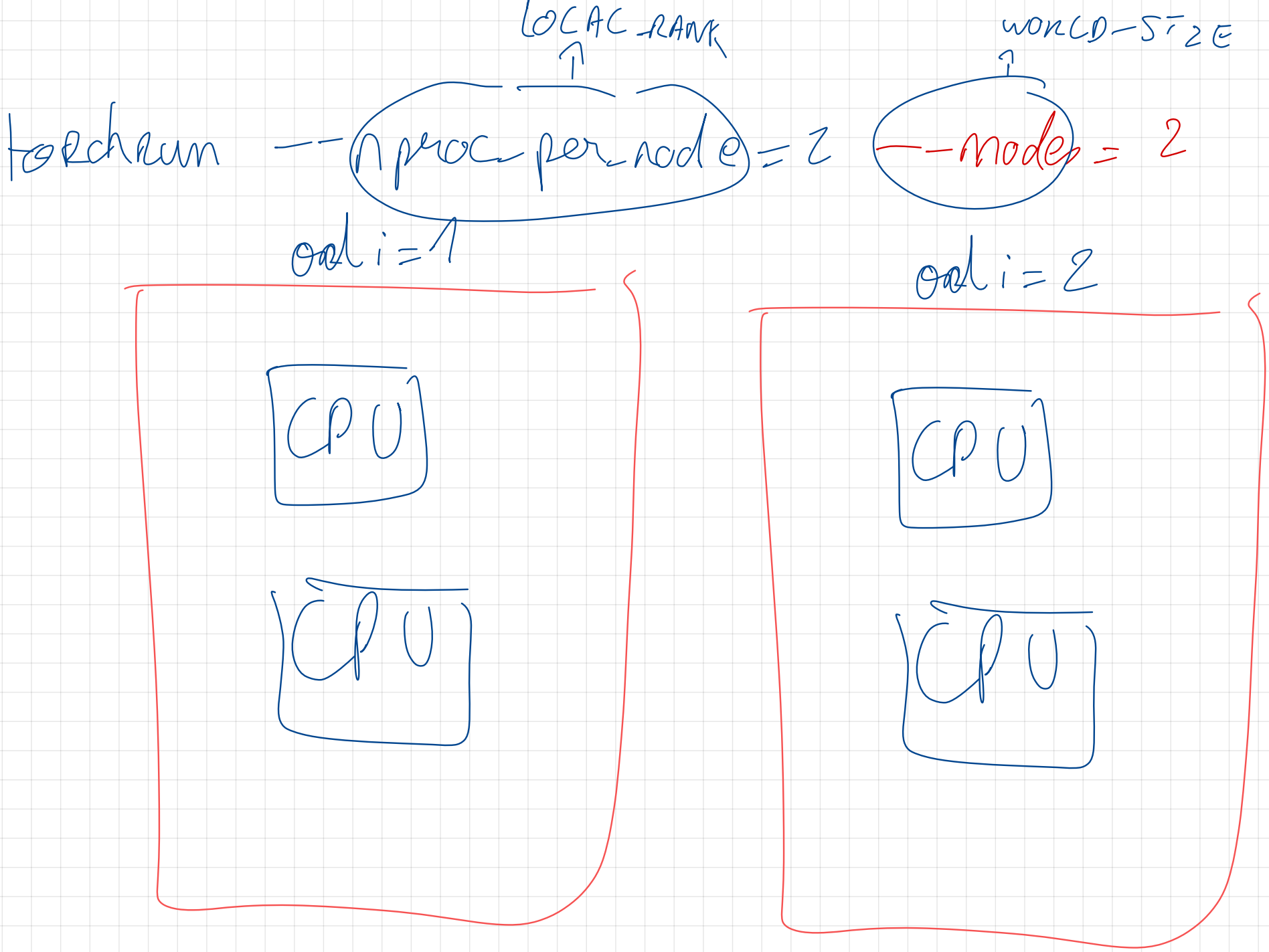
seq len = 3


J'aime les rois
J'aime les rois

Cours 2 :

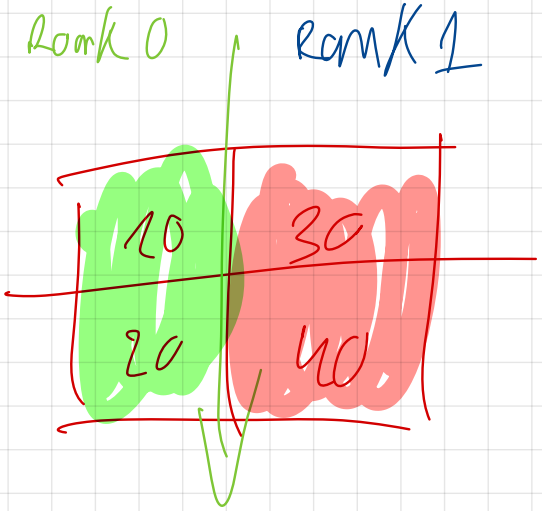
torchrun --nproc_per_node=2 --nodes=1

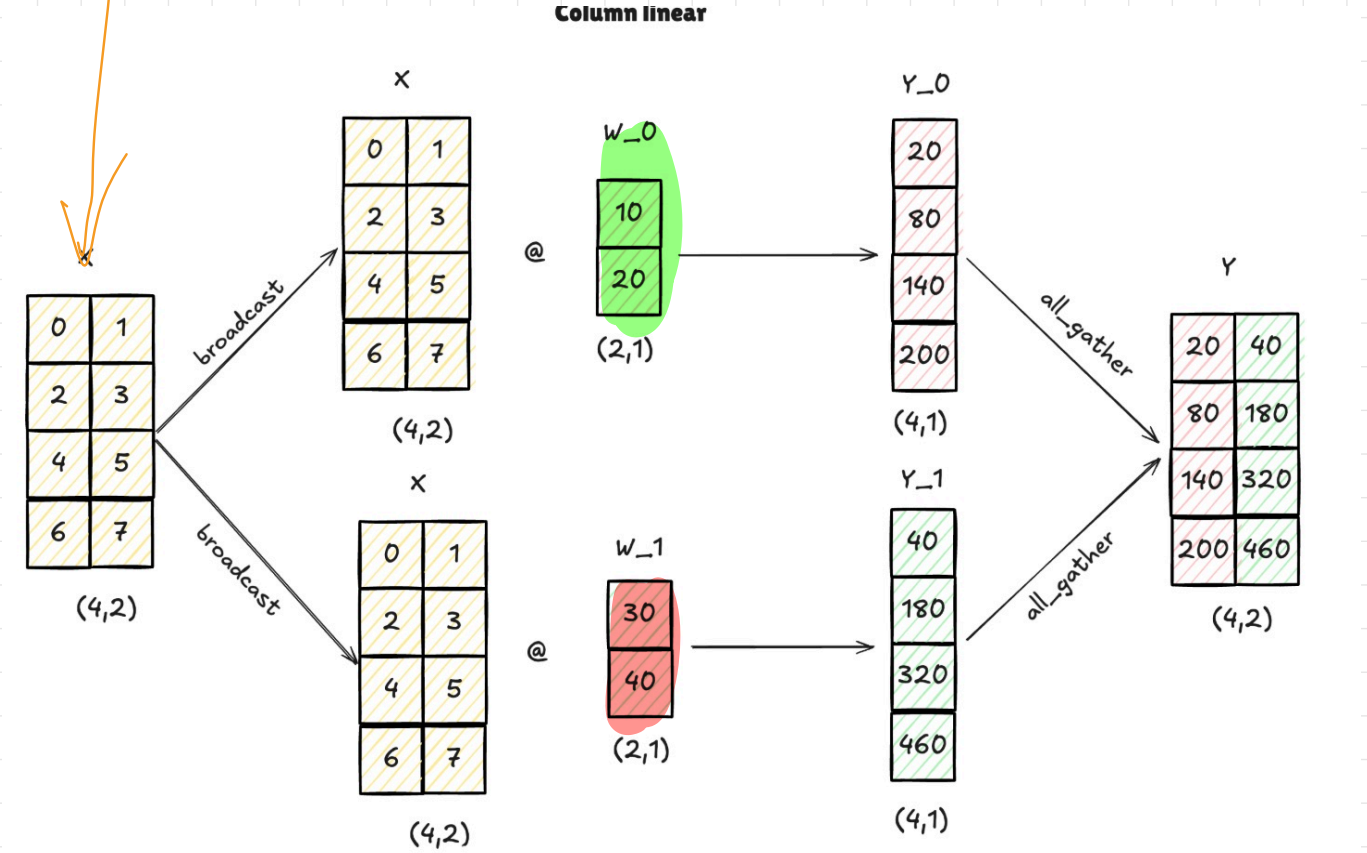



LOCAL_RANK
WORLD_SIZE
torchrun
--nproc_per_node=2
--nodes = 2
odi = 1
odi = 2
CPU
CPU
CPU
CPU

Cours 3: Tensor parallel

$$f(x) = XW^T + b$$





$$y = x @ w$$

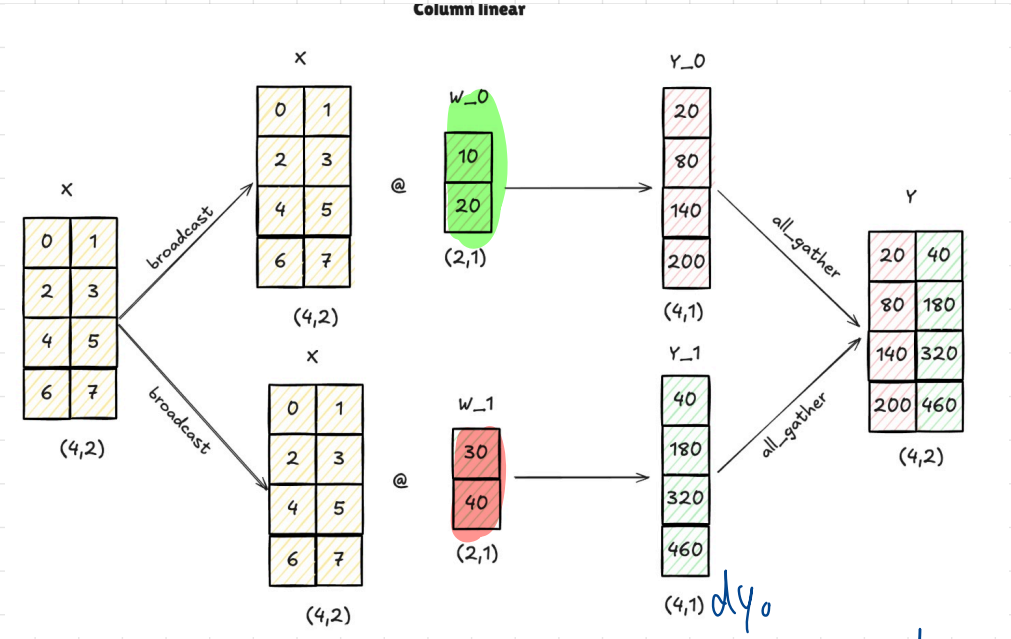
$$\frac{dy}{dx} = w$$

$$\frac{dy}{dw} = x$$

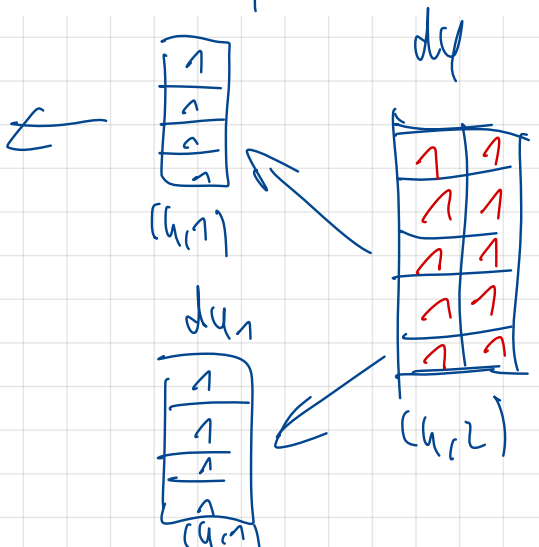
$$\frac{dL}{dx} = \frac{dL}{dy}\left(\frac{dy}{dx}\right) = \frac{dL}{dy}(w)$$

$$\frac{dL}{dw} = \frac{dL}{dy} \times \left(\frac{dy}{dw}\right) = \frac{dL}{dy} \times (x)$$

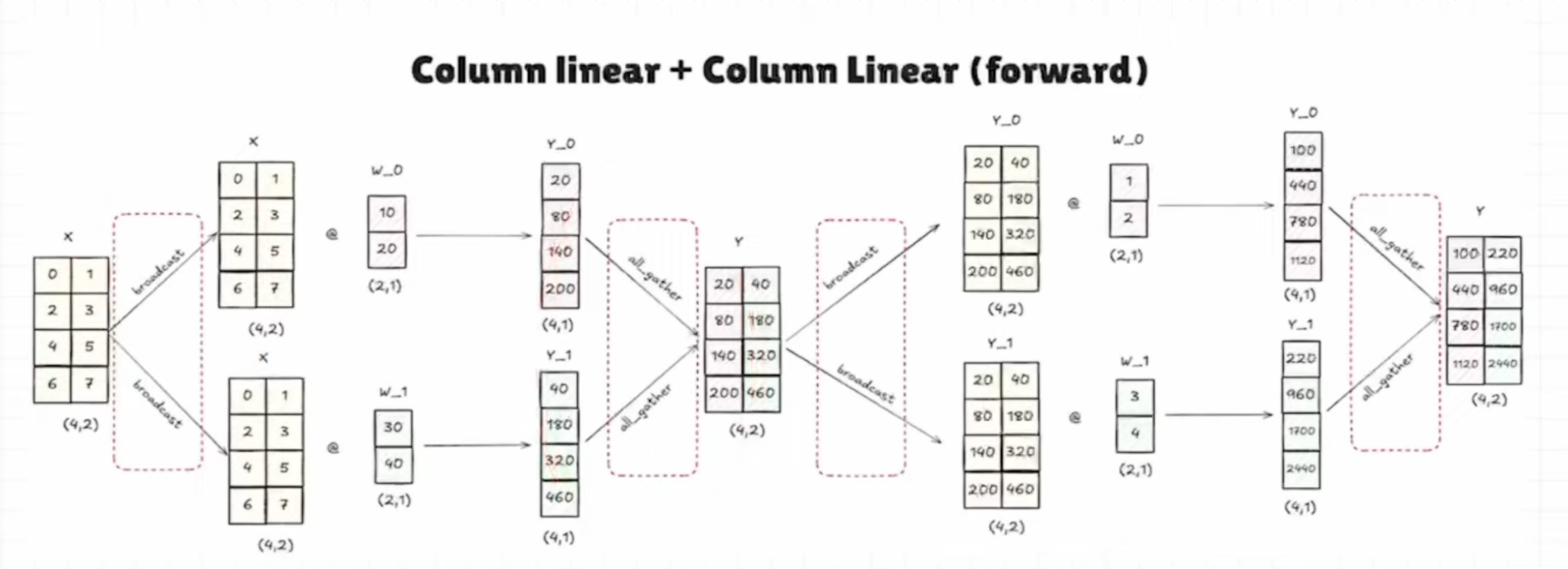
Backward column linear



$$\frac{dL}{dX} = \frac{dL}{dy} \times W$$



Week 3 + 4


Column linear + Column Linear (forward)
X
Y_0
W_0
Y
Y_1
W_1
broadcast
all_gather
(4,2)
(2,1)
(4,1)

COL + COL

COL + ROW

ROW + COL

ROW + ROW

Linear 1 → $f(\cdot)$ → LINEAR 2

$cat[\text{Gelu}(x@W_0), \text{Gelu}(x@W_1)] \Leftarrow //$

$=$

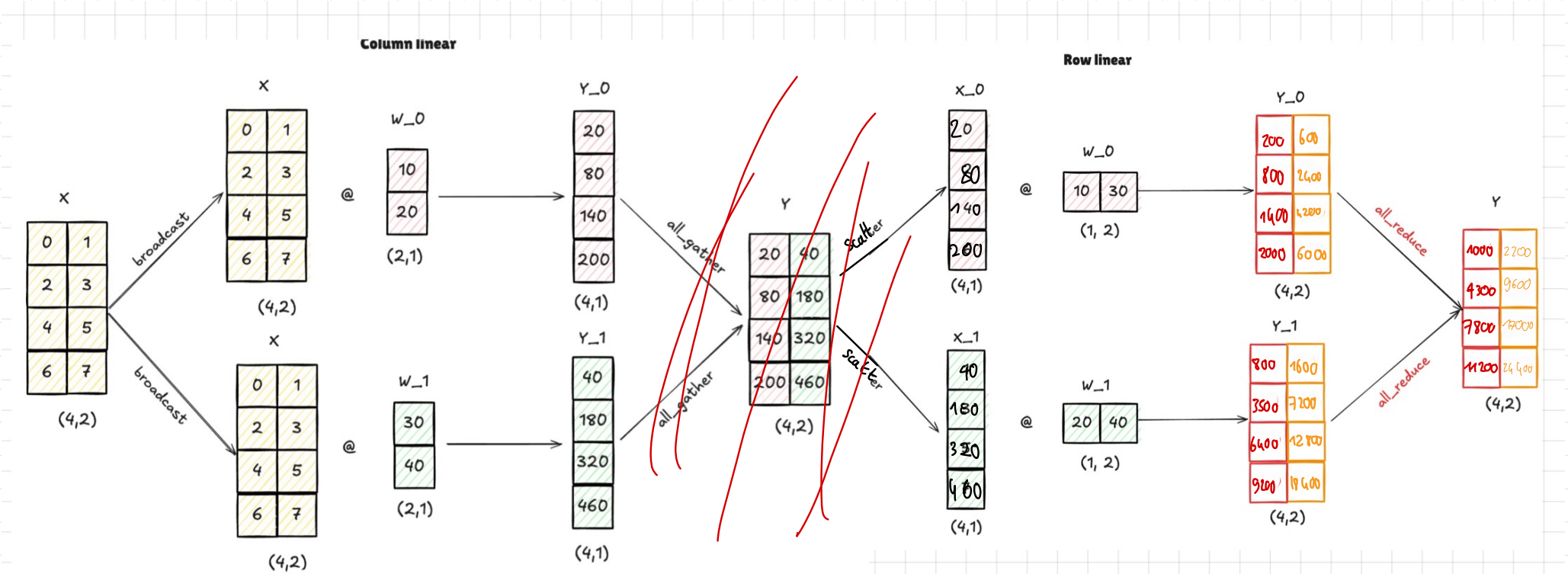
$\text{Gelu}[cat(x@W_0, x@W_1)] \rightarrow$ non-//

$// \rightarrow \text{Gelu}(x@W_0) + \text{Gelu}(x@W_1)$

$\neq$

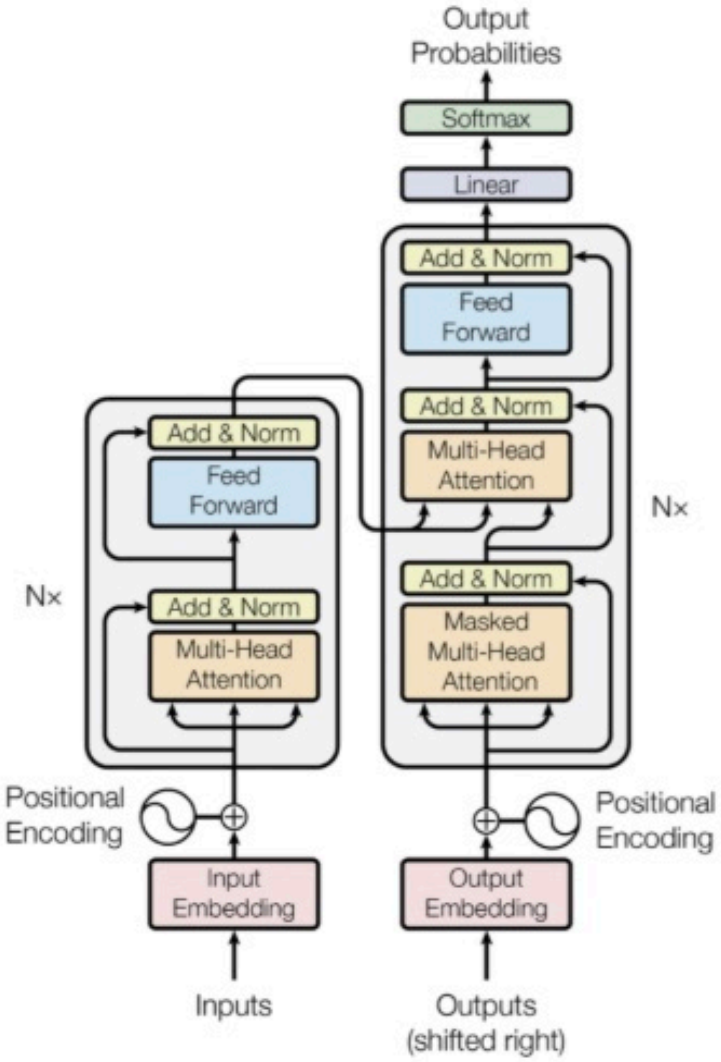
non-$// \rightarrow \text{Gelu}(x@W_0 + x@W_1)$

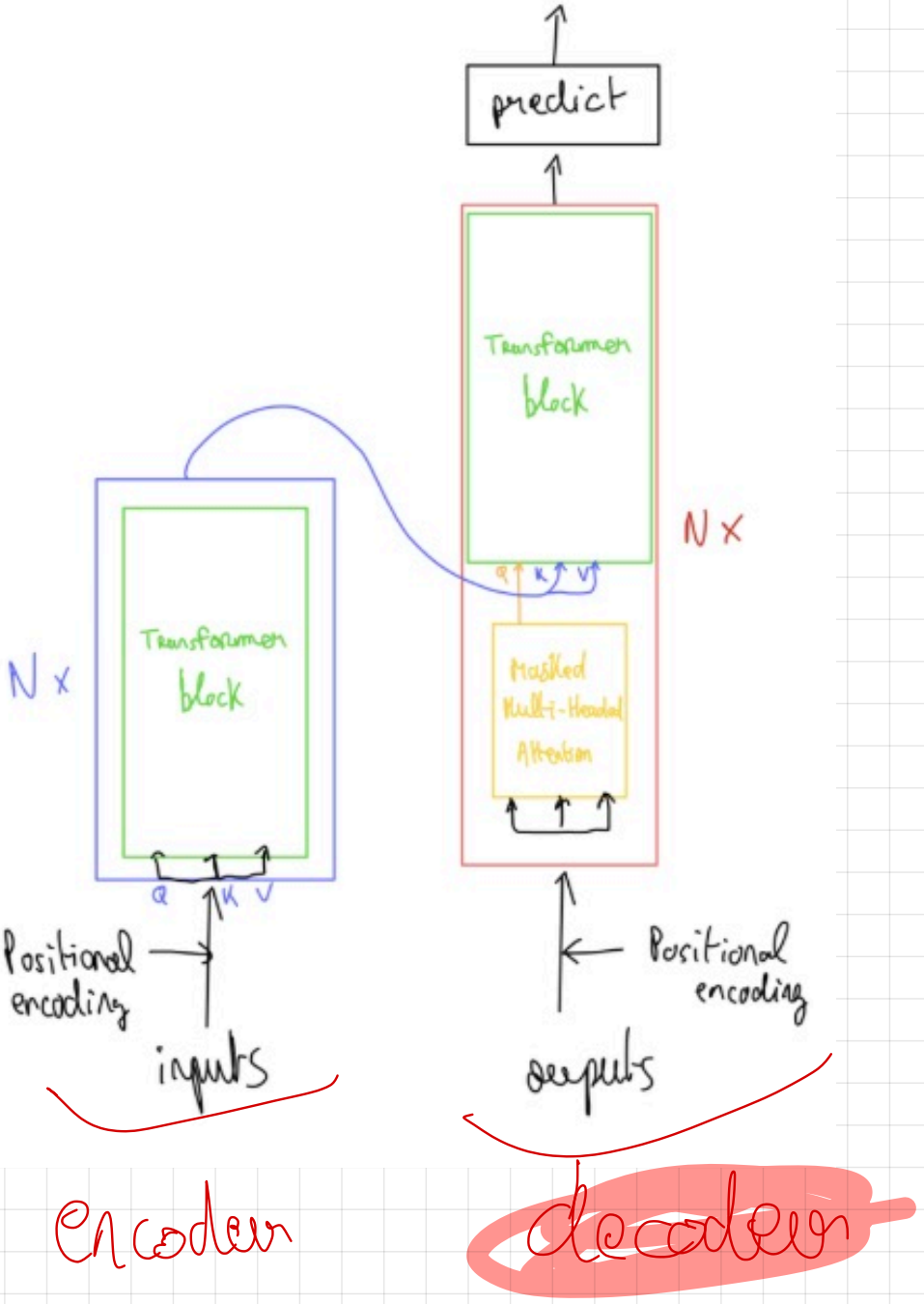
COLUMN + ROW linear

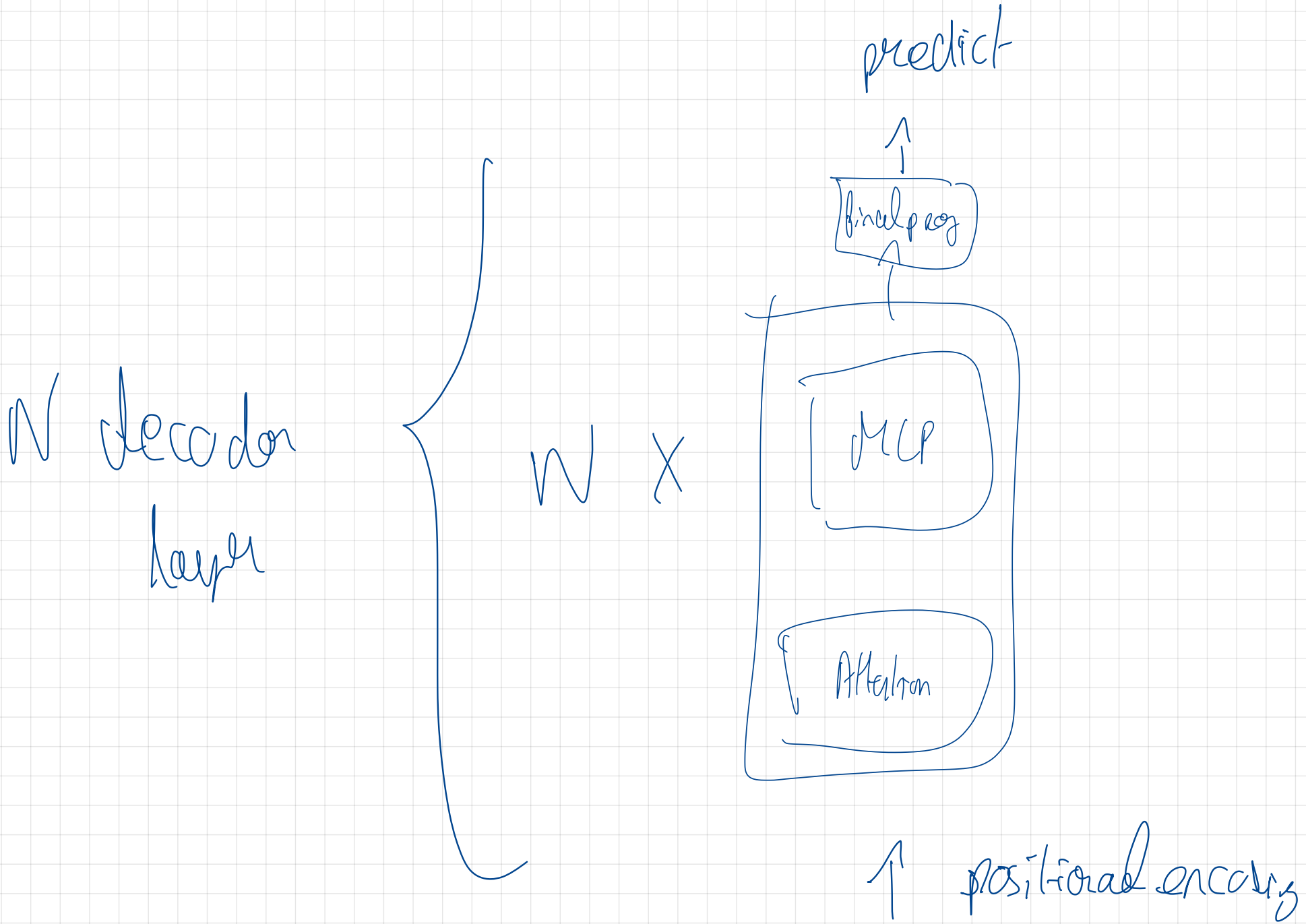


Saves 2 communication

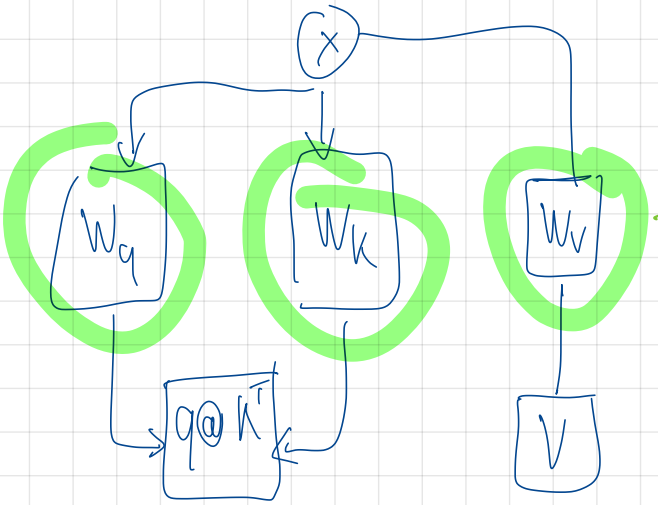
(compared to CR + CC)


Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Nx
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)

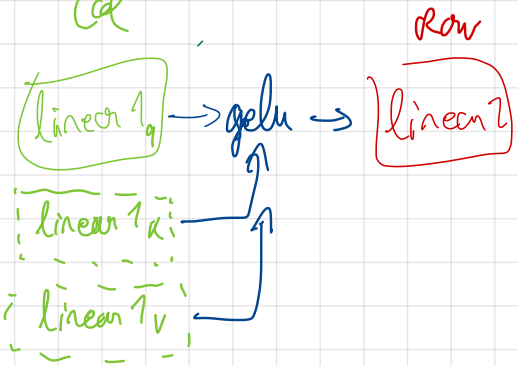

predict
Transformer block
N x
Transformer block
N x
Q K V
Masked Multi-Headed Attention
Positional encoding
inputs
Positional encoding
outputs
encoder
decoder


predict
final proj
MLP
Attention
N decoder layer
N x
positional encoding

Attention



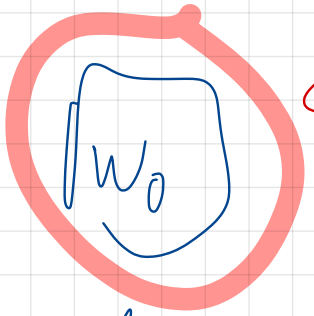
→ Column Parallel



$q: x @ W_q$

$K: x @ W_K$

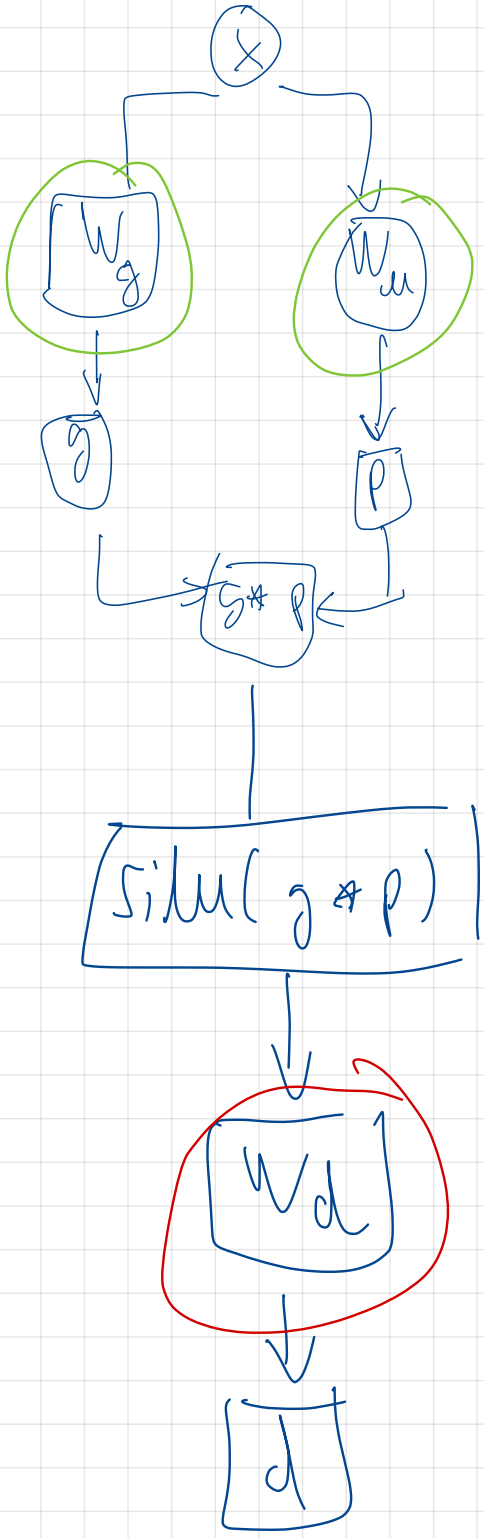
$V: x @ W_V$



← Row Parallel

$$O = (\text{softmax}(q @ K^T) @ V) @ W_O$$

MLP



column

row

d: silu(g * u) @ W_d