SETR paper Resume

→ segmentation has always been done using a FCN leveraging an encoder-decoder architecture, This approach was very successful, until it was clear That The bottleneck of The Model was context in very precise segmentation Tasks (self driving, inventory...)

⟹ researchers Tried To augment context by increasing receptive field, but receptive field increased linearly with [number of conv layers × kernel size]

- increasing number of conv layers ⟹ more depth ⟹ more compute + risk of overfitting
- increasing kernel size ⟹ Model won't be able To detect intricate details which is especially important in segmentation Task, due To pixel-level labeling

⇒ receptive field remains unsolved until Now

→ SETR comes to solve this receptive field problem with Transformer architecture which enables context to the furthest extent from till now //

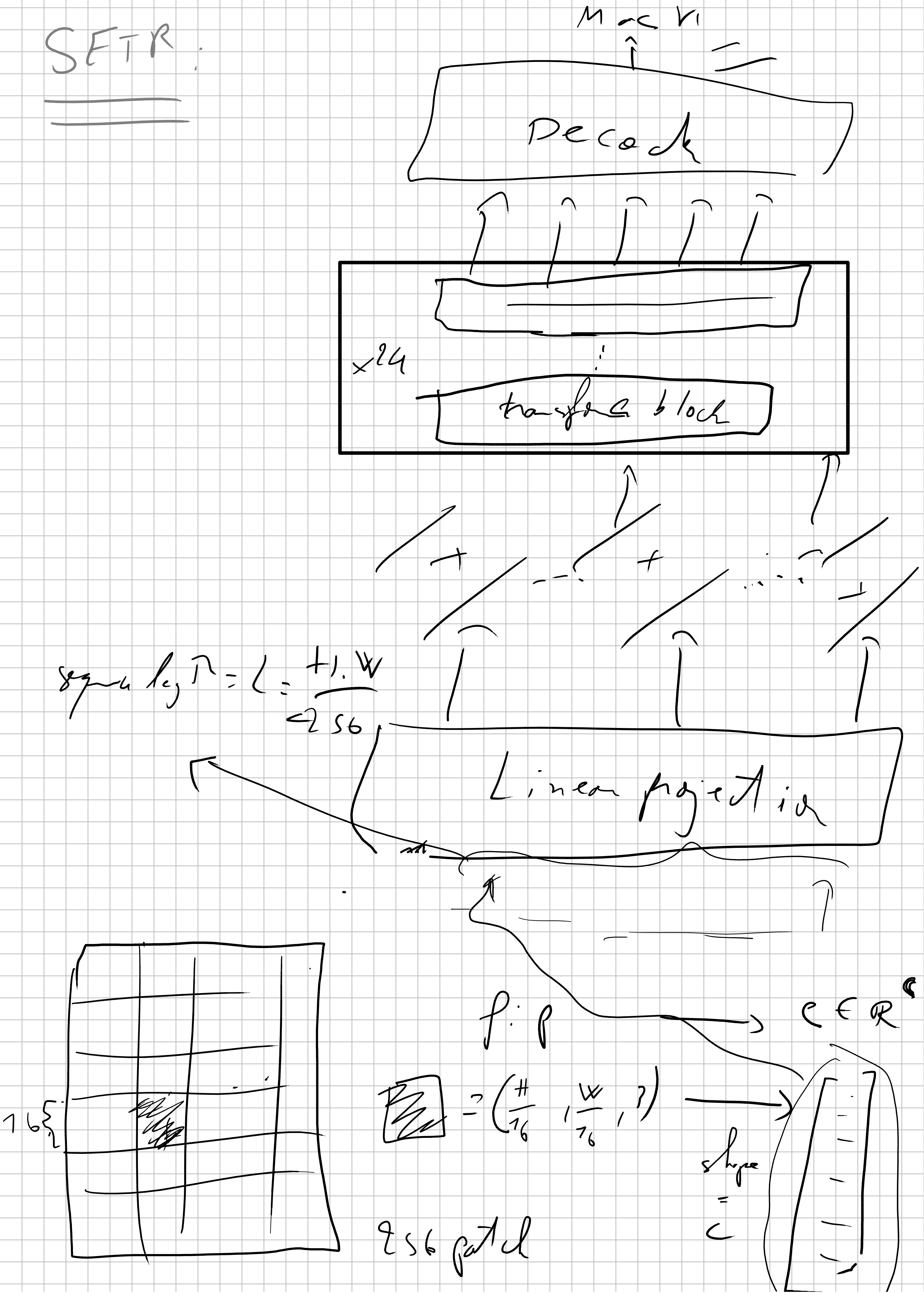
3 Main architecture proposed in the paper:

SETR MLA — multi-level feature aggregation

SETR — segmentation Transformer

SETR PUP — progressive upsampling

SETR:


mask
Decoder
x24
transformer block
+
sequence length = L = H·W / 256
Linear projection
p: p
c ∈ R^C
16
= (H/16, W/16, 3)
shape = C
256 patch

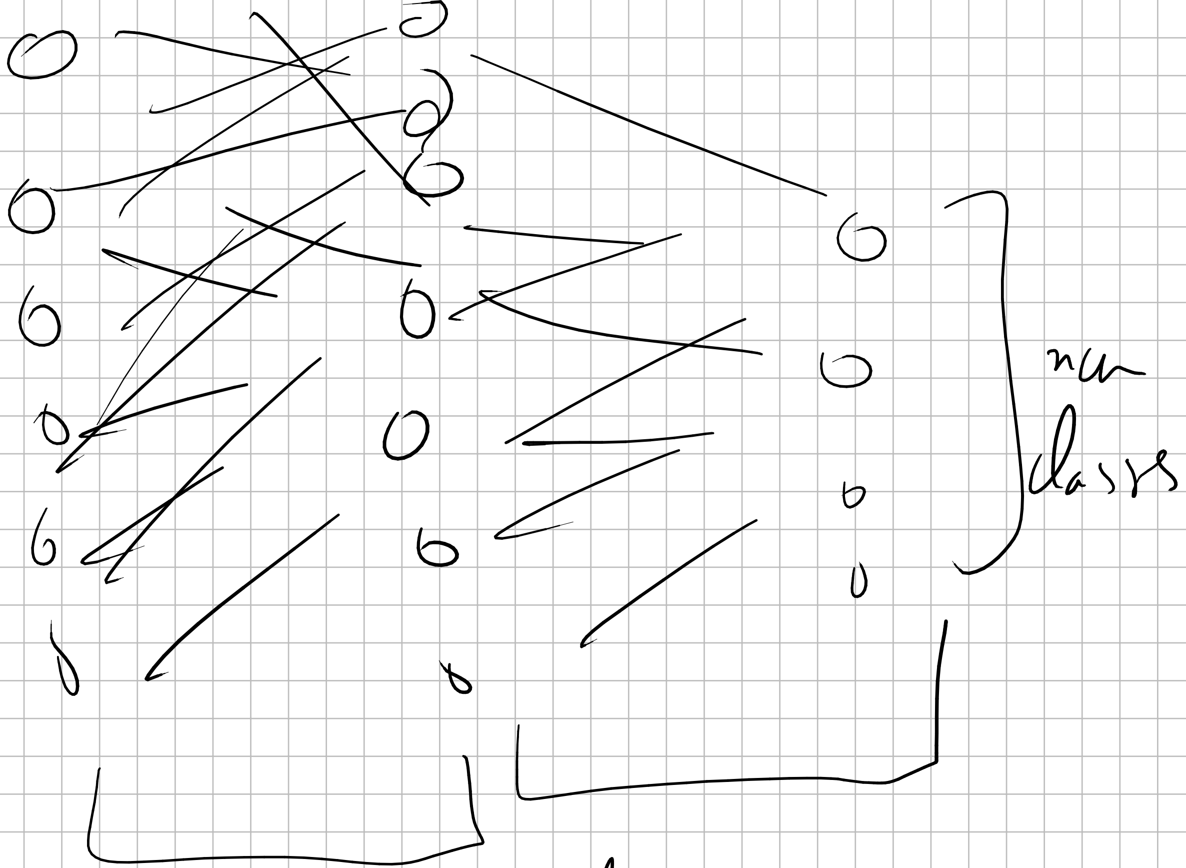
decoder design :

$$\text{decoder input} = \underbrace{\overbrace{\big[\, [\;]_{\in \mathbb{R}^{C}},\ [\;],\ \dots\ \big]}^{16}}_{L \text{ dim} = \frac{H \cdot W}{256} \text{ dims}}$$

$$\text{decoder output} = \big[\, \cancel{\otimes}_{\in \mathbb{R}^{\frac{H}{16} \times \frac{W}{16} \times C}},\ \cancel{\otimes},\ \big]$$

(1) Naive upsampling :

$$x \in \mathbb{R}^{C}\quad C\begin{cases} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{cases} = \overbrace{\underbrace{\big[\ \text{1x1 conv} + \text{batchnorm}\ \big]}_{19\ [\text{number of classes}]\ \text{filters}}}^{\times 2}$$

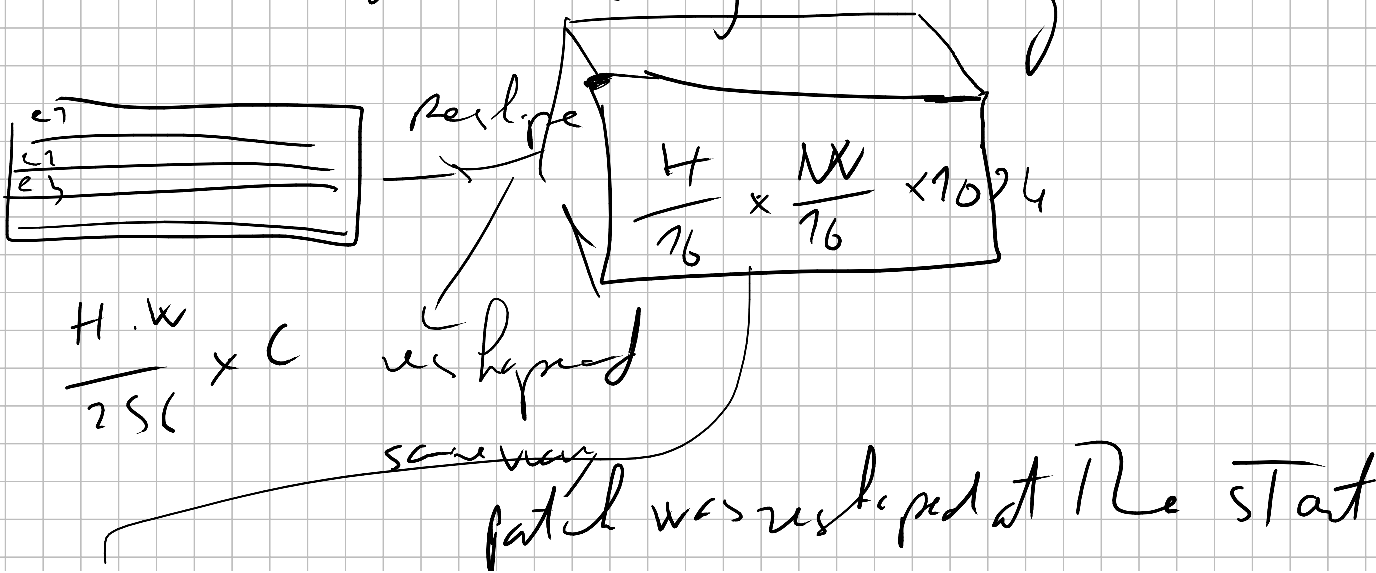


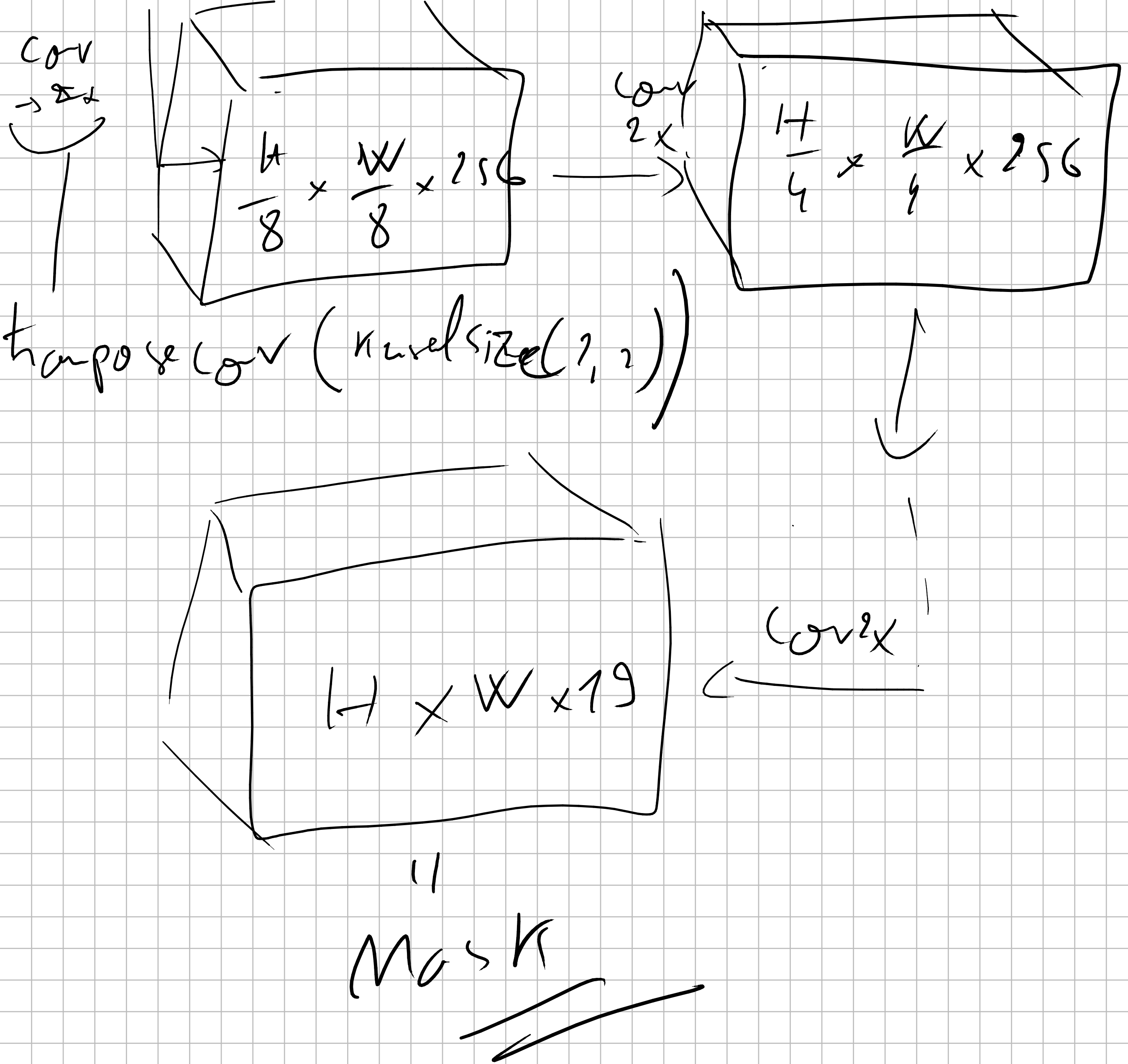
1×1 conv + batch Norm + 1×1 Conv

(2) PUP :

$\frac{H \cdot W}{256}$ = number of embedding as input to decoder

C = dimension of embedding





→ we do This since one-step upscaling introduced in naive decoder might introduce Noise as it might underfit The scaling process

(!) SETR-PUP have more params Than SETR-Naive because of This deeper decoder

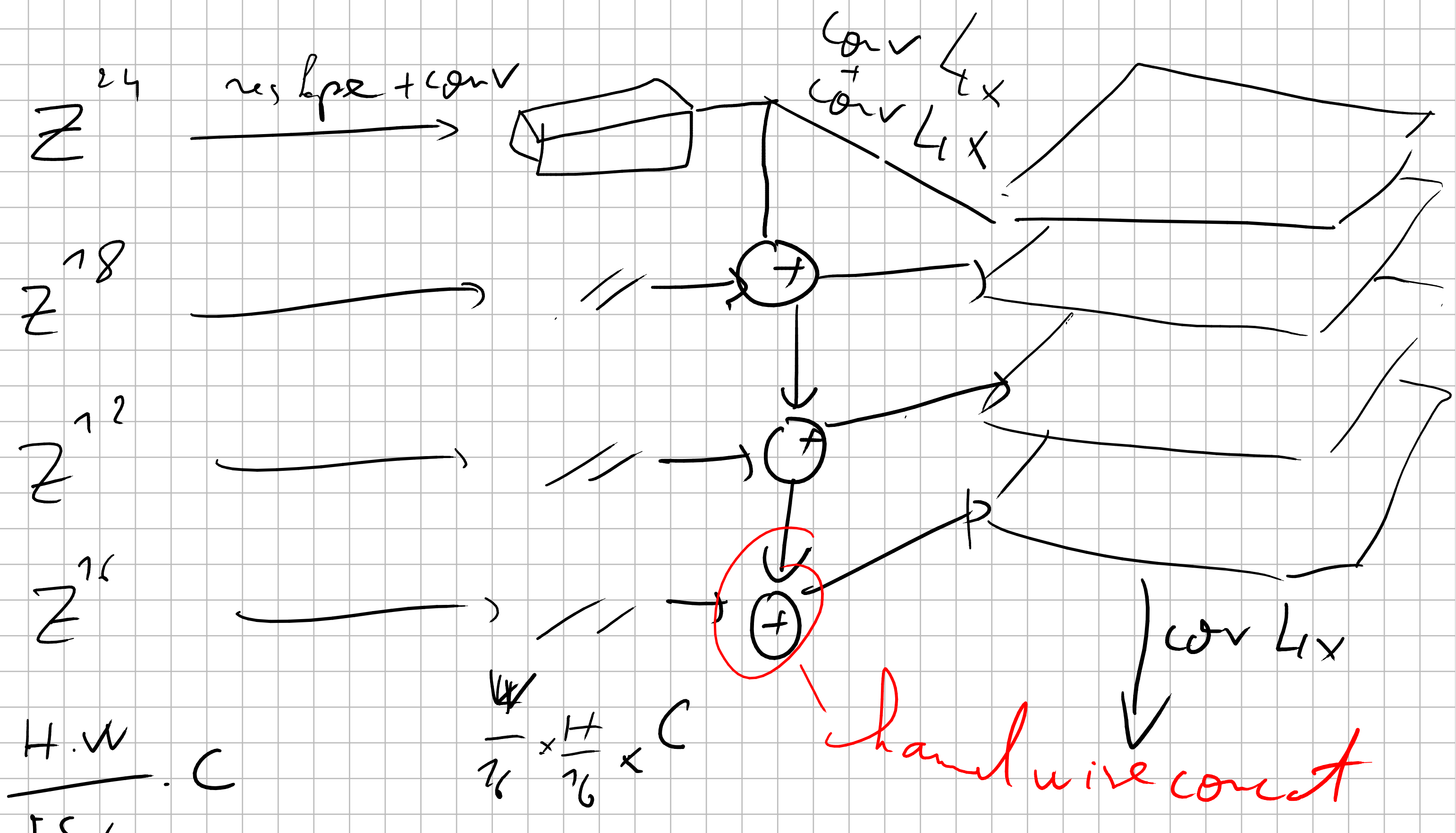
(3) MLA:

→ This decoder is even larger (Params ++)

further proving scale MATTERS, as this decoder outperformed (2) & (1)

→ Multi-level feature aggregation:

Z^m = output of m^{th} Transformer block in the encoder : $\left(\frac{H.W}{256}, C\right)$



⇒ We fuse all features from the transformer layers

⟹ SETR-MLA

(!) $params_{MLA}$ > $params_{PUP}$ > $params_{Naive}$

⟵ scale

(.) in the paper ViT & DEIT

weights were used + embeddings

to make up for data shortage

⟹ ViT : image understanding
(long-range context)

finetuned for segmentation

Task using 3 different

decoders.

$\Longrightarrow$ achieves

- SOTA on Pascal
- Competitive results on cityscapes
- 1st on ADE20K