# PyTorch live talks

Capsules and routing techniques

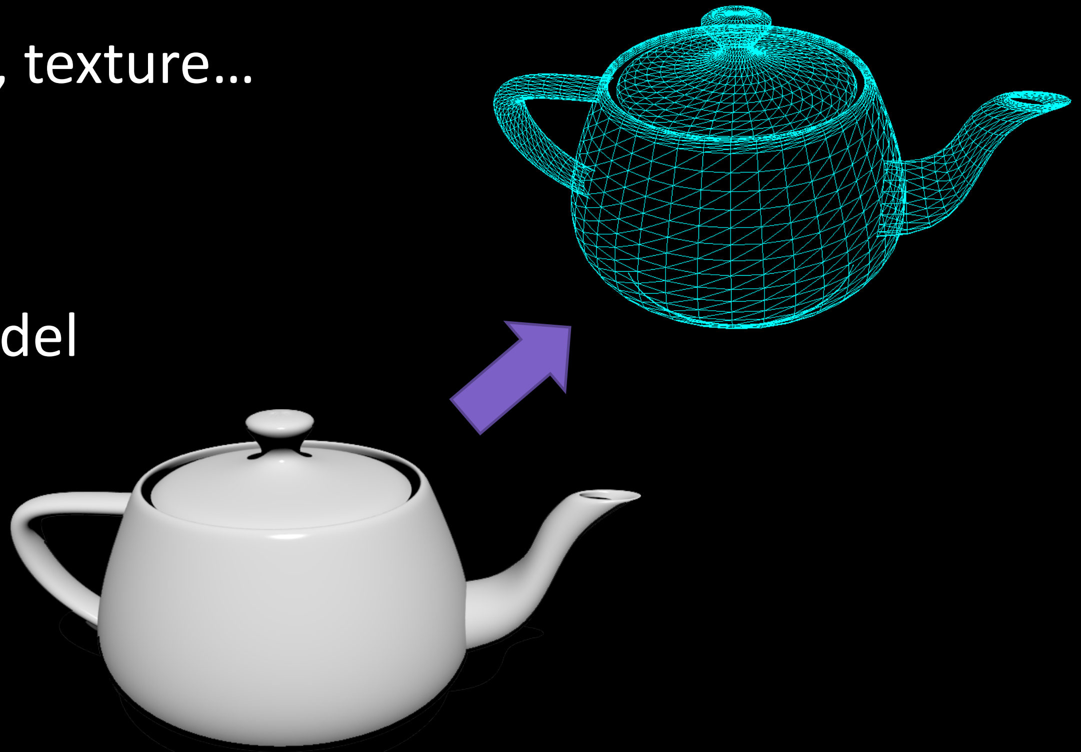
# Rationale

- CNN *can* deal with *translation*
- For other *affine transformations*
  - Exponential replicas of spatial feature detectors
  - Exponentially more labelled data

- Solution
  - Efficiently encode *viewpoint invariant knowledge*

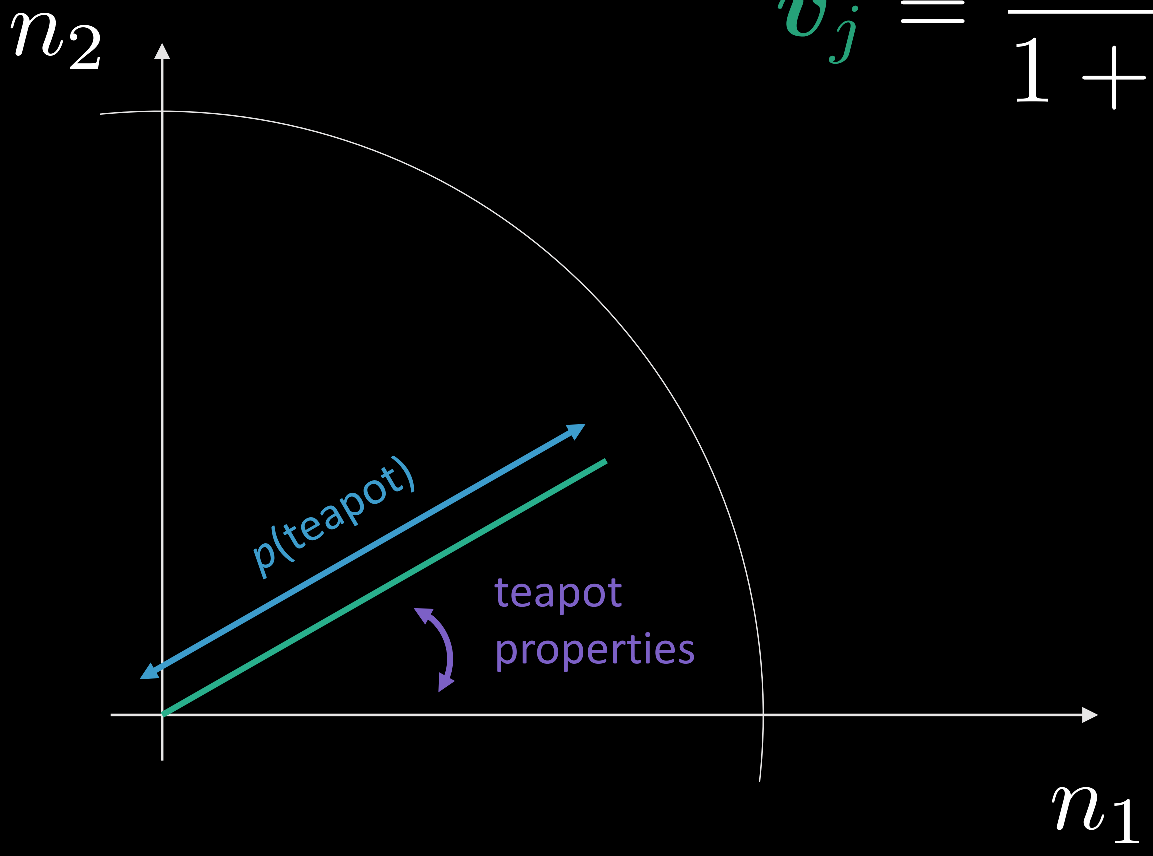
# Capsules

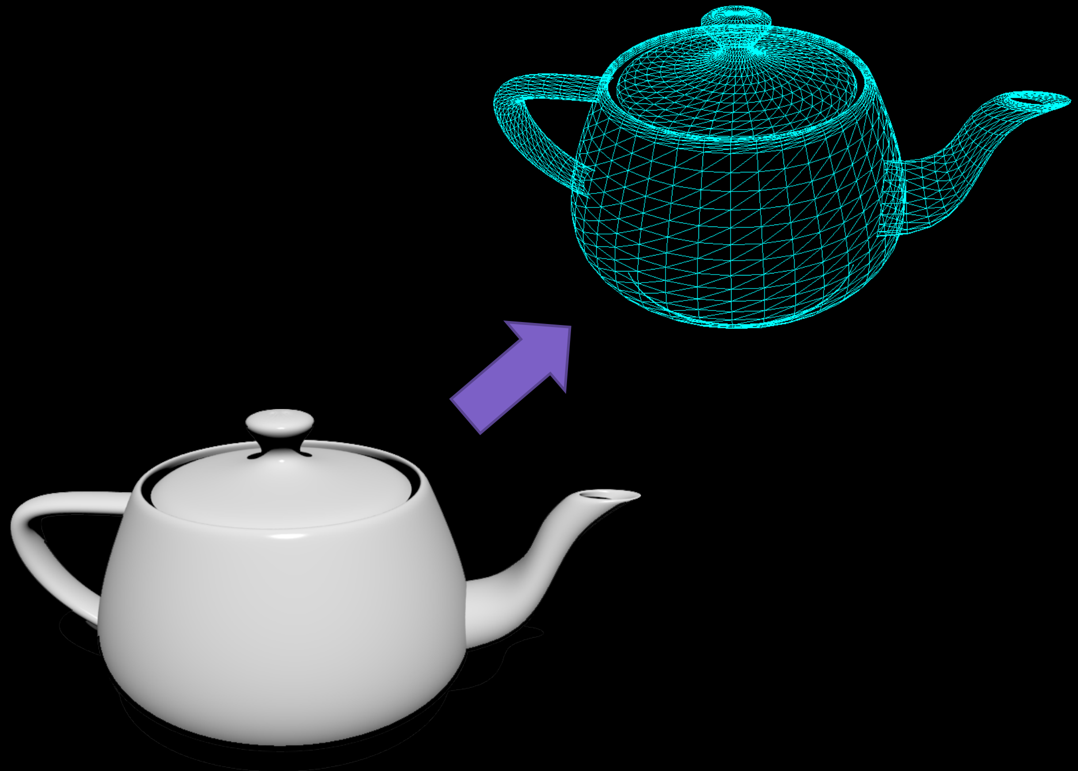
- Groups of neurons characterising an entity in the image
- Properties include
  - Pose, deformation, velocity, albedo, hue, texture…
- Role
  - Invert the rendering process
  - 2D camera projection → 3D abstract model



# Capsules

$$\boldsymbol{v}_j = \frac{\|\boldsymbol{s}_j\|^2}{1 + \|\boldsymbol{s}_j\|^2} \frac{\boldsymbol{s}_j}{\|\boldsymbol{s}_j\|}$$





# Capsules

$$\boldsymbol{v}_j = \frac{\|\boldsymbol{s}_j\|^2}{1 + \|\boldsymbol{s}_j\|^2} \frac{\boldsymbol{s}_j}{\|\boldsymbol{s}_j\|}$$

Capsule output

$$\boldsymbol{s}_j = \sum_i c_{ij} \hat{\boldsymbol{u}}_{j|i}$$

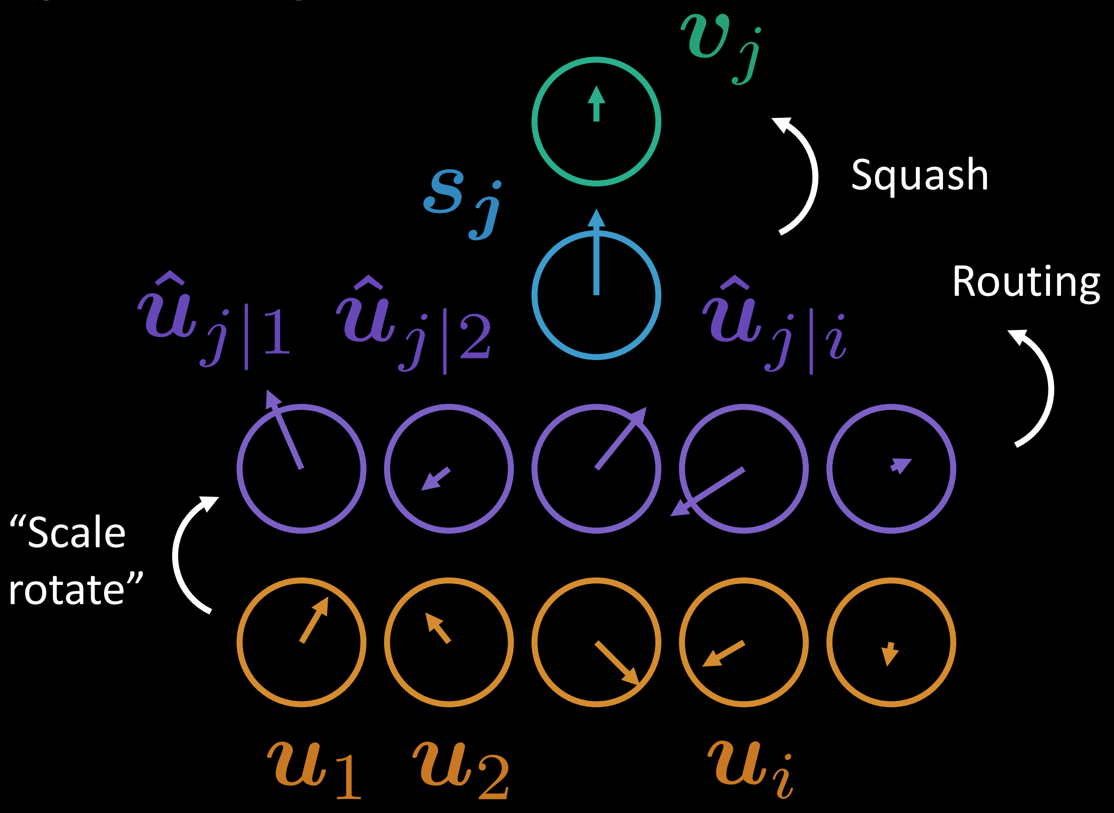
Capsule input

$$\hat{\boldsymbol{u}}_{j|i} = \boldsymbol{W}_{ij} \boldsymbol{u}_i$$

Prediction vectors

↑

learnt by back-prop



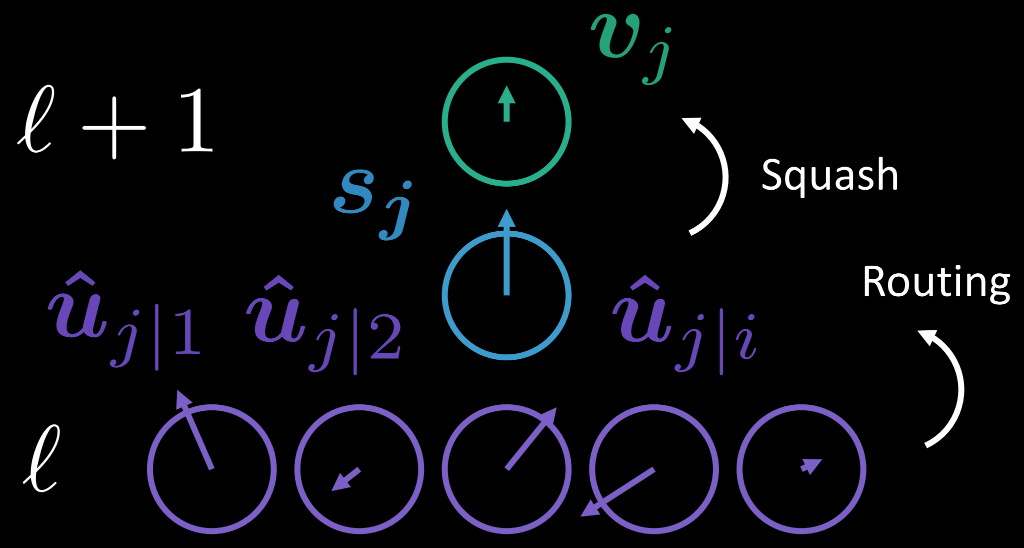
# Dynamic routing

$$\boldsymbol{b}_i \leftarrow \mathbf{0}, i = 1, \ldots, s_\ell$$

$$\boldsymbol{c}_i \leftarrow \operatorname{softmax}(\boldsymbol{b}_i)$$

$$\left.\begin{aligned} \boldsymbol{s}_j &= \sum_i c_{ij} \hat{\boldsymbol{u}}_{j|i} \\ \boldsymbol{v}_j &\leftarrow \operatorname{squash}(\boldsymbol{s}_j) \end{aligned}\right\} \longrightarrow j = 1, \cdots, s_{\ell+1}$$

$$b_{ij} \leftarrow b_{ij} + \hat{\boldsymbol{u}}_{j|i}^\top \boldsymbol{v}_j$$


$\boldsymbol{v}_j$
$\ell + 1$
$\boldsymbol{s}_j$
Squash
Routing
$\hat{\boldsymbol{u}}_{j|1}$
$\hat{\boldsymbol{u}}_{j|2}$
$\hat{\boldsymbol{u}}_{j|i}$
$\ell$

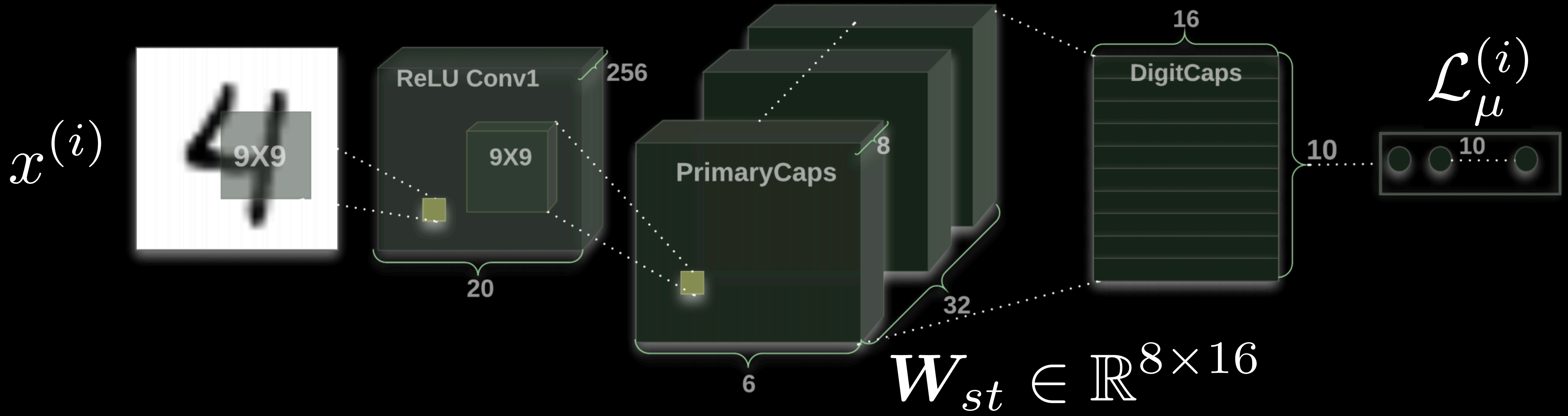
$s_\ell$: size of layer $\ell$

# Margin loss

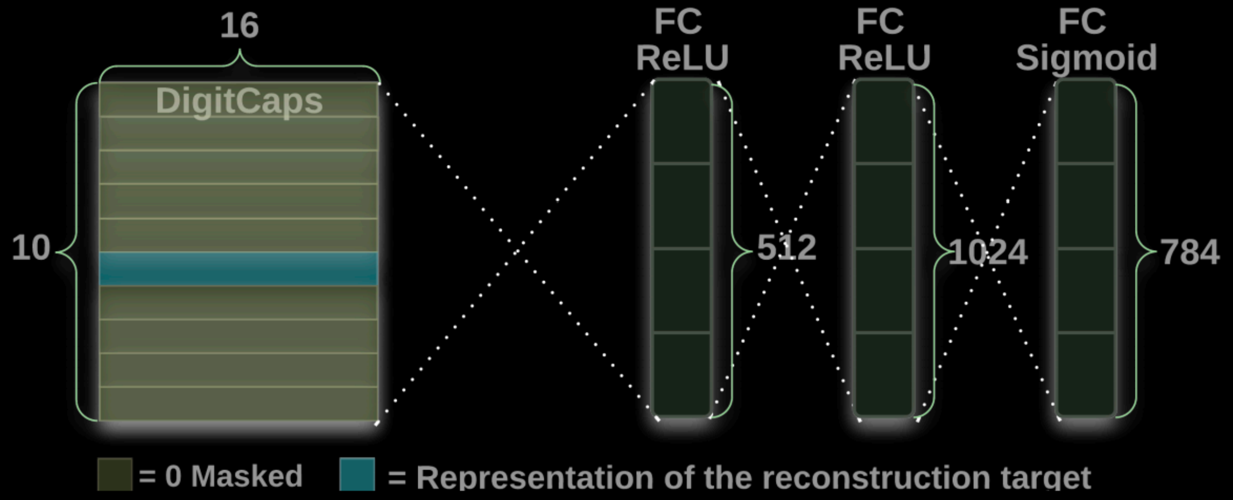
$$\mathcal{L}_k{}^{(i)} = \begin{cases} \left[\left(m_+ - \|\boldsymbol{v}_k^{(i)}\|\right)^+\right]^2, & \text{digit } k \text{ preset} \\ \lambda\left[\left(\|\boldsymbol{v}_k^{(i)}\| - m_-\right)^+\right]^2, & \text{otherwise} \end{cases}$$

$$\mathcal{L}_{\underset{\substack{\uparrow \\ \text{margin}}}{\mu}}^{(i)} = \sum_k \mathcal{L}_k{}^{(i)}$$

# CapsNet



$$\boldsymbol{W}_{st} \in \mathbb{R}^{8\times 16}$$

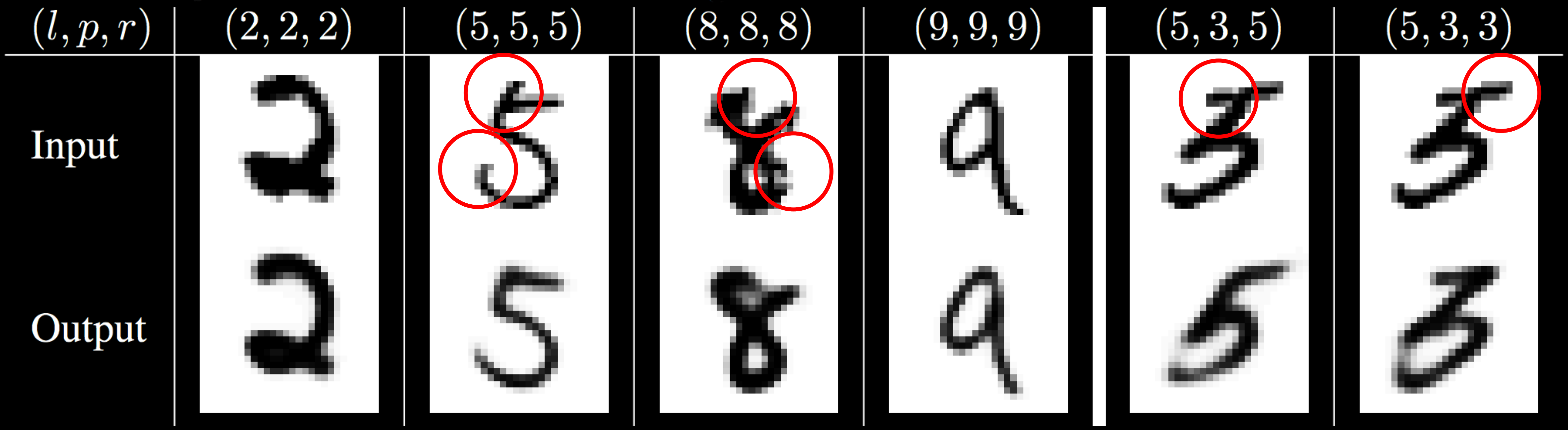


$$\mathcal{L}^{(i)} = \mathcal{L}_\mu^{(i)} + \rho \cdot \mathcal{L}_\rho^{(i)}$$

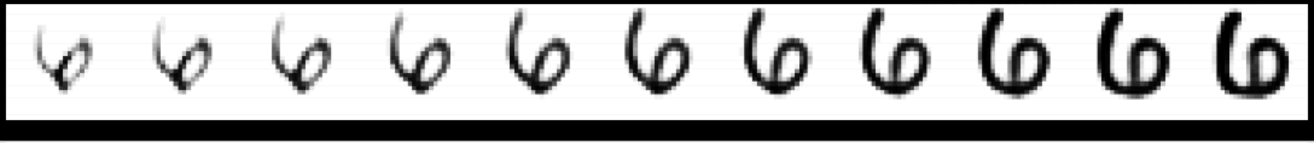
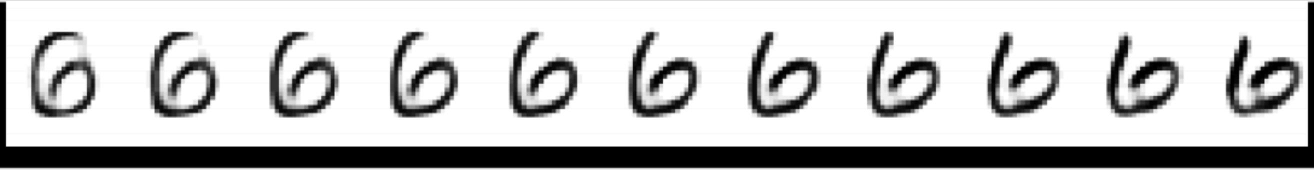
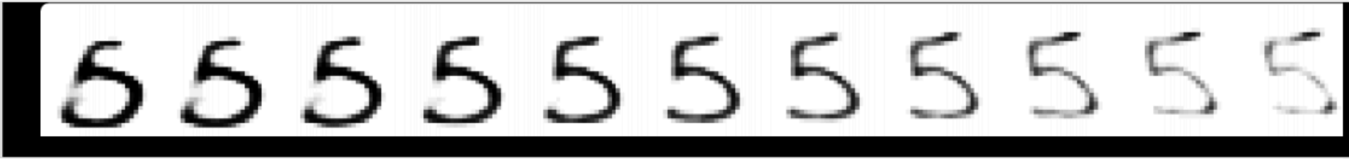
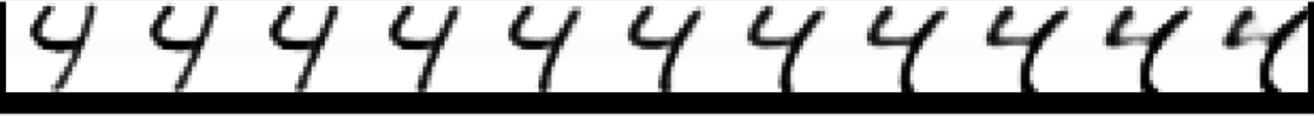
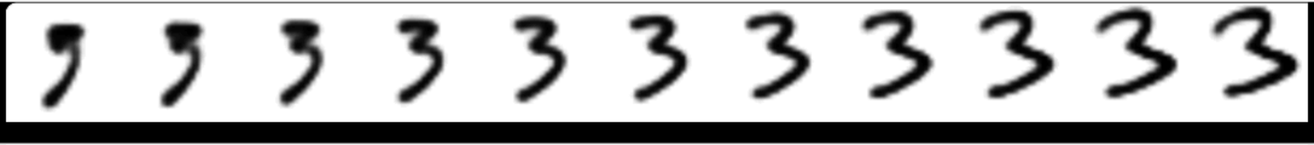
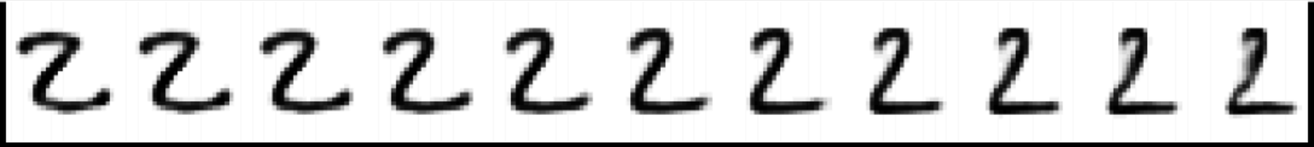
margin

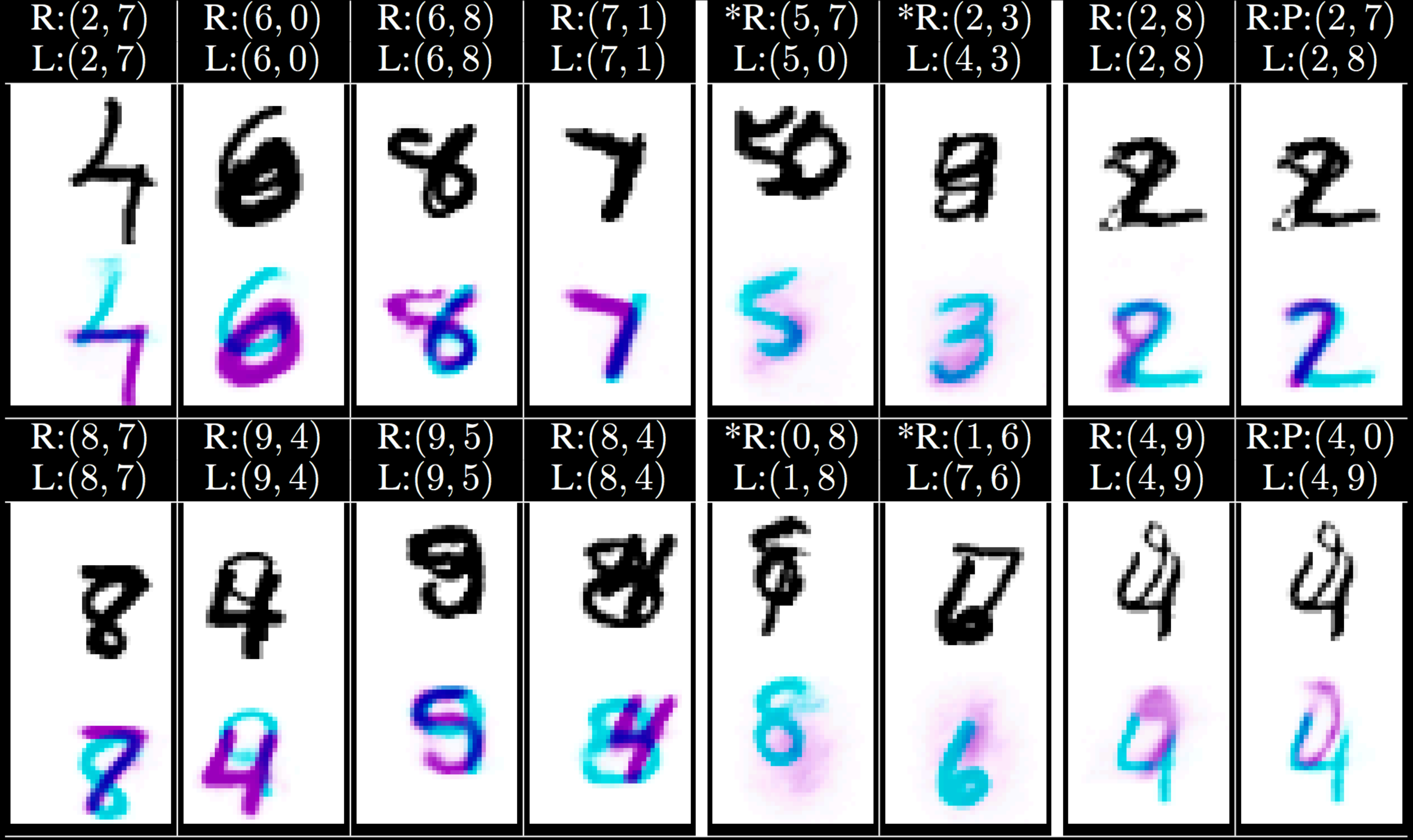
reconstruction

# Results (I)



# Results (II)

| | |
|---|---|
| Scale and thickness |  |
| Localized part |  |
| Stroke thickness |  |
| Localized skew |  |
| Width and translation |  |
| Localized part |  |


R:(2, 7)
L:(2, 7)
R:(6, 0)
L:(6, 0)
R:(6, 8)
L:(6, 8)
R:(7, 1)
L:(7, 1)
*R:(5, 7)
L:(5, 0)
*R:(2, 3)
L:(4, 3)
R:(2, 8)
L:(2, 8)
R:P:(2, 7)
L:(2, 8)
R:(8, 7)
L:(8, 7)
R:(9, 4)
L:(9, 4)
R:(9, 5)
L:(9, 5)
R:(8, 4)
L:(8, 4)
*R:(0, 8)
L:(1, 8)
*R:(1, 6)
L:(7, 6)
R:(4, 9)
L:(4, 9)
R:P:(4, 0)
L:(4, 9)