Seminar: Explainable Machine Learning

Carsten Lüth 5/17/2018

Matrix Capsules with EM routing

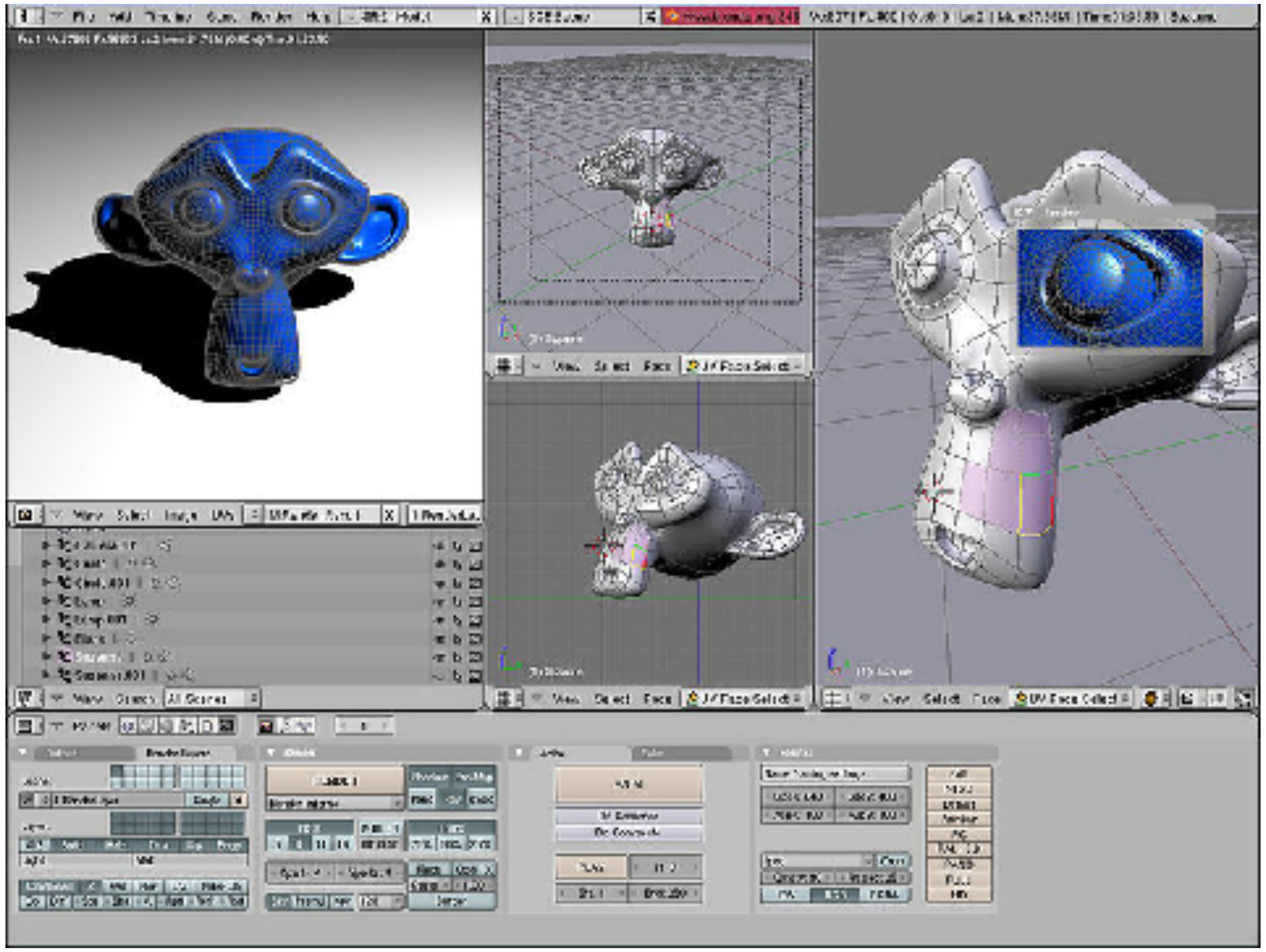
by Geoffrey Hinton,
Sara Sabour, Nicholas Frost

Capsule networks' core idea is to break up the neural net into chunks, or capsules, that work in teams and are assigned a portion of a problem; each capsule is pre-loaded with basic training on its portion of the object or process it will examine. The individual capsules work cooperatively, sharing their findings and contributing to solving the problem as a whole.

–Mike Fitzmaurice

Capsules General

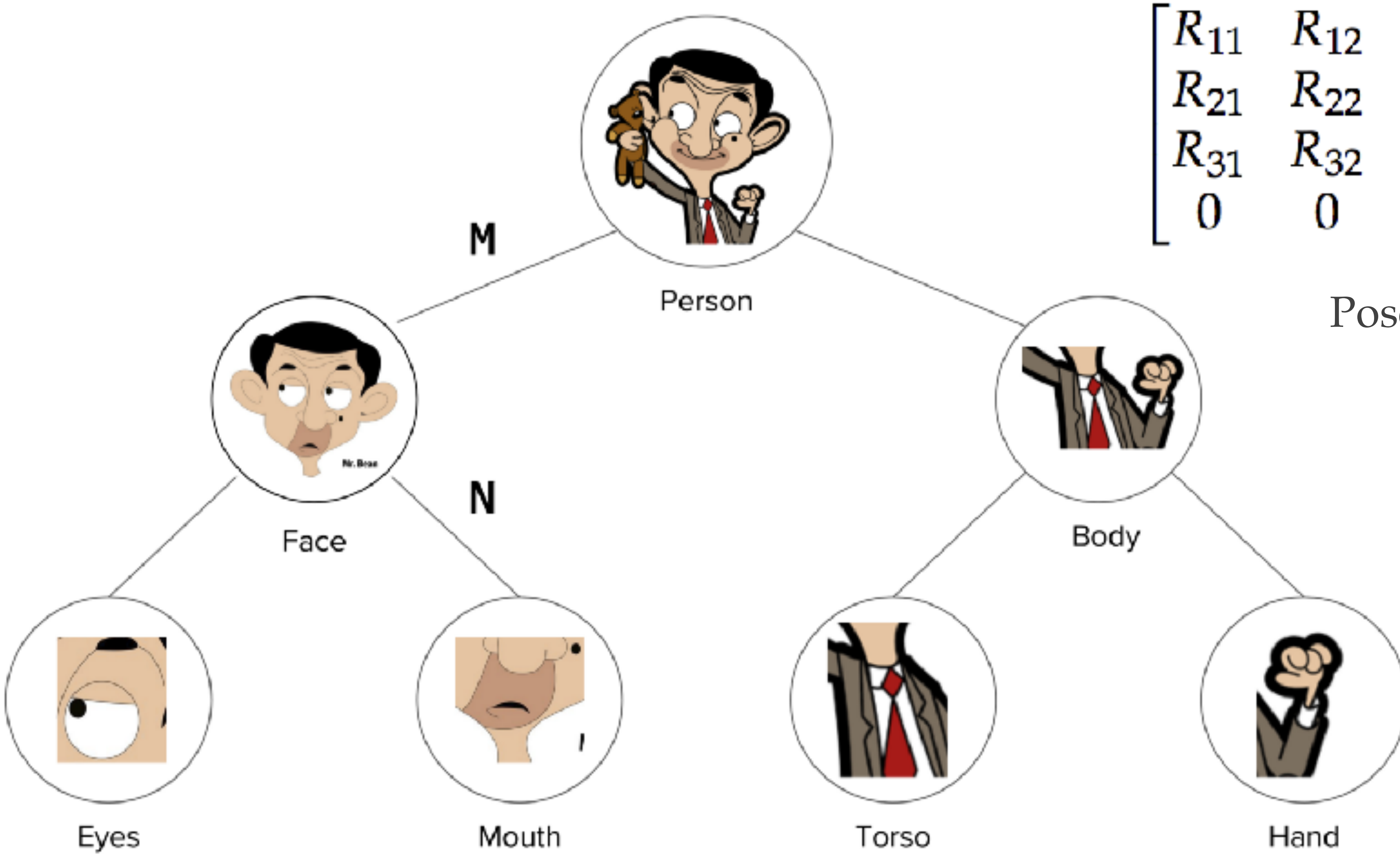
- contains both probability of occurence (activation) and parameters of the specific feature or entity (pose)
- filter out irrelevant information for the task without max-pooling
- works like an inverse graphical rendering program where entity is described by a vector or pose



Difference Capsule and Standard Network

- Activations are based on:
  - Standard Net:
    - compare single incoming activity vector and weight vector
  - Capsule Net:
    - compares multiple incoming pose predictions and their corresponding activations
      - thereby assigns parts to wholes via cluster finding

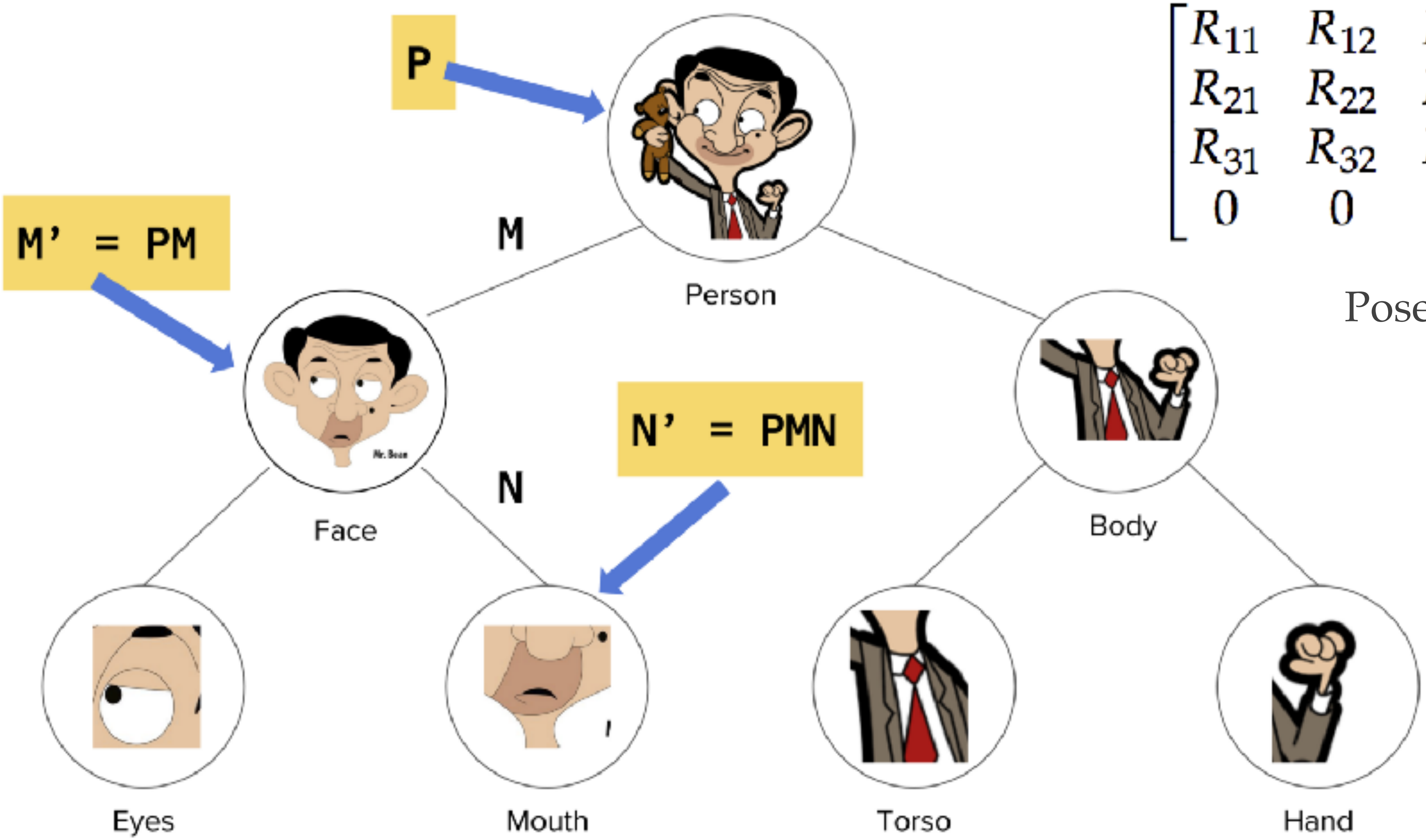
Rendering



$$\begin{bmatrix} R_{11} & R_{12} & R_{13} & T_1 \\ R_{21} & R_{22} & R_{23} & T_2 \\ R_{31} & R_{32} & R_{33} & T_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Pose Matrix

Rendering

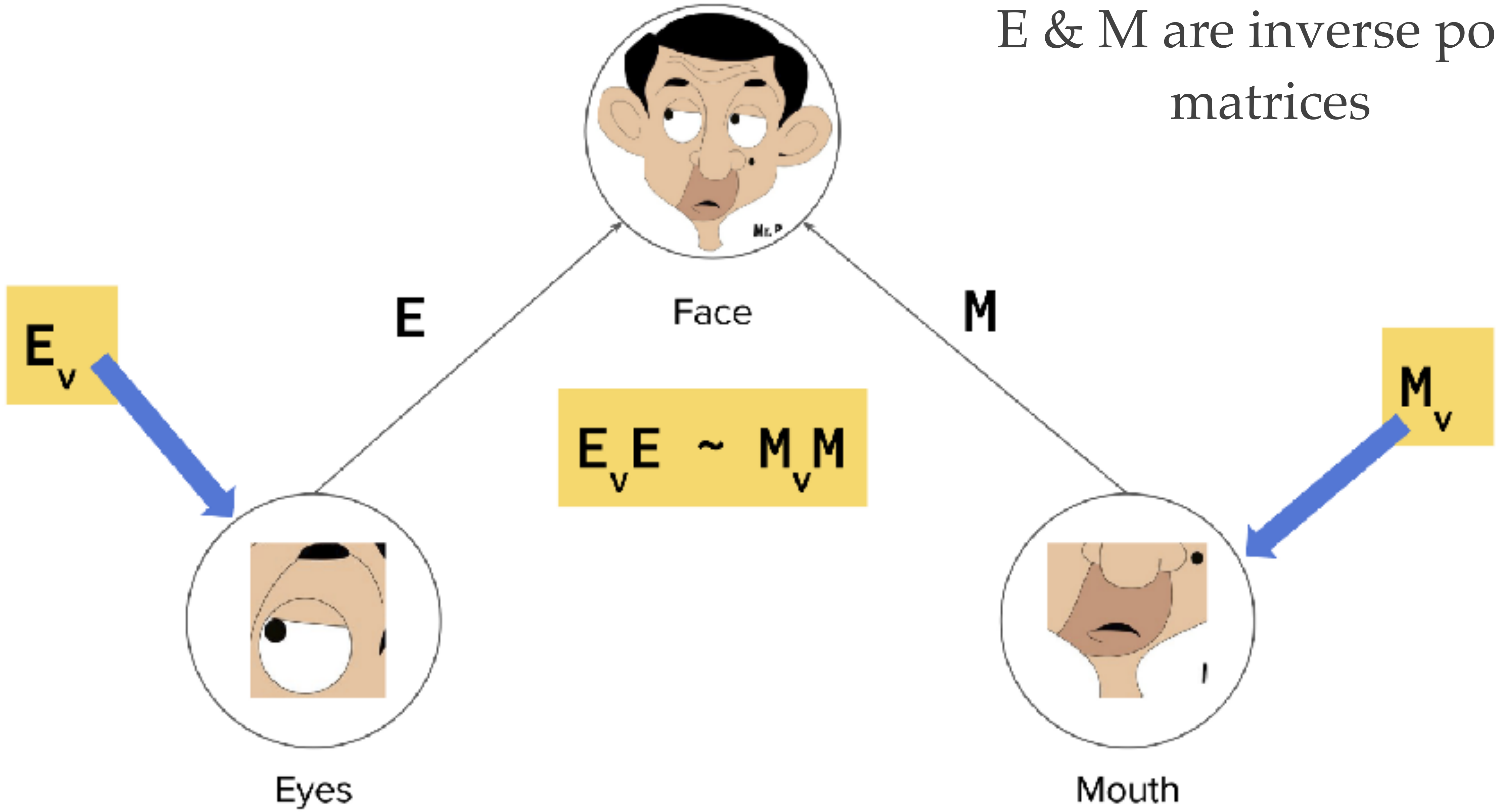


$$\begin{bmatrix} R_{11} & R_{12} & R_{13} & T_1 \\ R_{21} & R_{22} & R_{23} & T_2 \\ R_{31} & R_{32} & R_{33} & T_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Pose Matrix

Inverse Rendering

E & M are inverse pose matrices



Assigning parts to wholes

- done via routing algorithm between Capsules
  - utilization of high- dim. coincidence filtering: looking for agreement of capsule votes
  - votes produced by a learned linear transformation of the pose of each capsule
  - linear transf. represents viewpoint invariant relationship between part and whole

Improved Explainability

- Likelihood or the activation of the capsules can be interpreted as saliency for specific regions
- Instantiation parameter pose values usable to explain consistency among the layers
    - Capsules describing the same object are in an appropriate relationship with consistence parameters, leading to a parse tree like structure

Difference between dyn. routing and EM routing

- length of pose vector is activation
- Cosine of angle to measure agreement
- pose is vector with length n

- dedicated activation element
- log variance of a Gaussian cluster
- pose is matrix with n x n elements

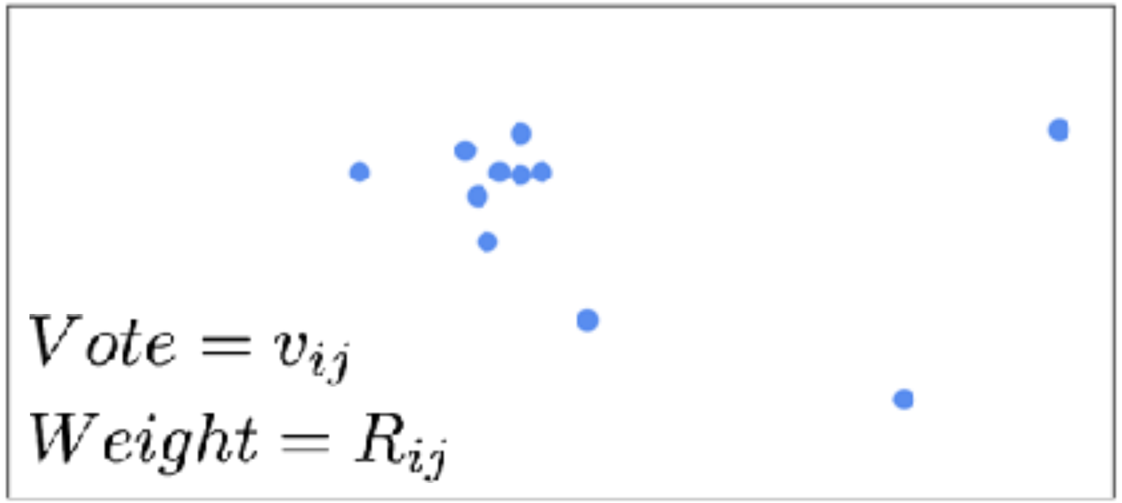
EM routing

- Each Capsule in Layer L+1 corresponds to Gaussian where μ is the pose
- Utilization of the “Minimum Description Length” principle (best hypothesis or regulariser allows strongest compression of data)
  - choice whether to activate capsule in L+1 based on votes from L

Computation of the votes

- Set of Capsules in Layer L : Ω_L
- Each Capsules has "Pose Matrix" $M \in \mathbb{R}^{4\times4}$ and activation $a \in (0,1)$
- Weight matrix connecting capsule i from Ω_L with j from Ω_{L+1} : $V_{ij} \in \mathbb{R}^{4\times4}$
- Vote from Ω_L i to Ω_{L+1} j: $V_{ij} = M_i W_{ij}$

EM - Schematic in 2d-space for one capsule j in Layer L+1

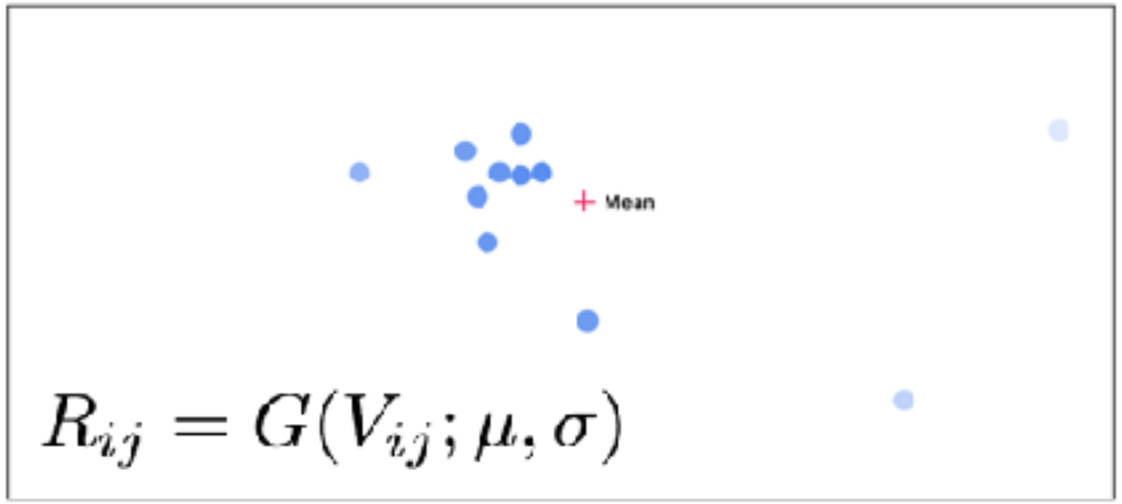


$Vote = v_{ij}$

$Weight = R_{ij}$

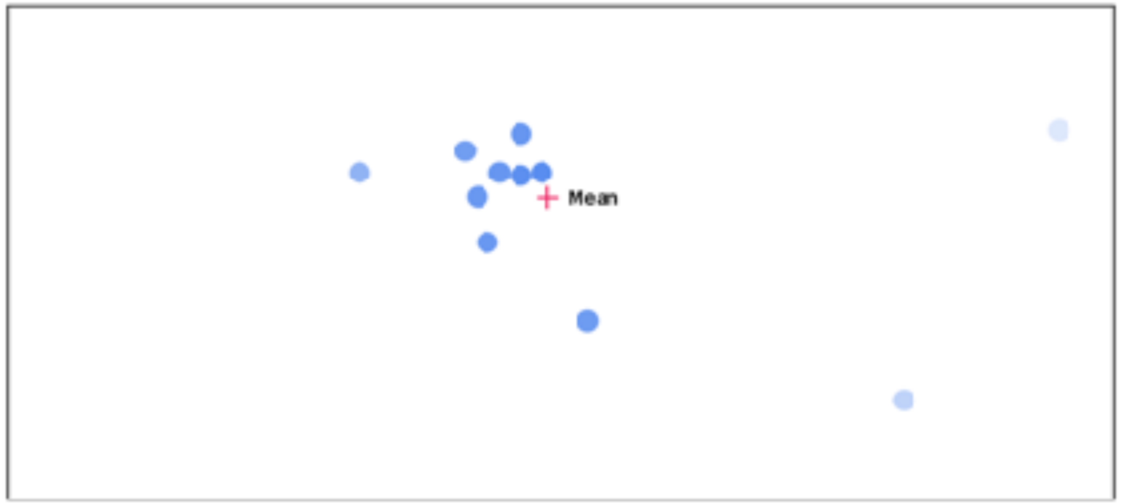
Start

M-Step $\mu = \dfrac{\sum_i R_{ij} V_{ij}}{\sum_i R_{ij}}$ σ



$R_{ij} = G(V_{ij}; \mu, \sigma)$

E-Step



M-Step

Choice of activation

- introduce a cost of explaining data point i by using capsule j: $cost_{j|i} = -ln(G(V_{ij}; \mu_j, \sigma_j))$
- cost for activation of j:

$$cost_j = -\sum_i R_{ij} \cdot ln(G(V_{ij}; \mu_j, \sigma_j)) = \sum_i R_{ij}(ln(\sigma_j) + \beta_v)$$

- activation of capsule j:

$$a_j = logistic(\lambda(\beta_a - cost_j))$$

$\sum_i R_{ij}$: amount of data assigned to j

Algorithm

Procedure 1 Routing algorithm returns **activation** and **pose** of the capsules in layer $L+1$ given the **activations** and **votes** of capsules in layer L. V_{ij}^h is the h^{th} dimension of the vote from capsule i with activation a_i in layer L to capsule j in layer $L+1$. β_a, β_v are learned discriminatively and the inverse temperature λ increases at each iteration with a fixed schedule.

procedure EM ROUTING($\boldsymbol{a}$, V)
    $\forall i \in \Omega_L, j \in \Omega_{L+1}$: $R_{ij} \leftarrow 1/|\Omega_{L+1}|$
    for t iterations **do**
        $\forall j \in \Omega_{L+1}$: M-STEP($\boldsymbol{a}$, R, V, j)
        $\forall i \in \Omega_L$: E-STEP(μ, σ, $\boldsymbol{a}$, V, i)
    return $\boldsymbol{a}$, M

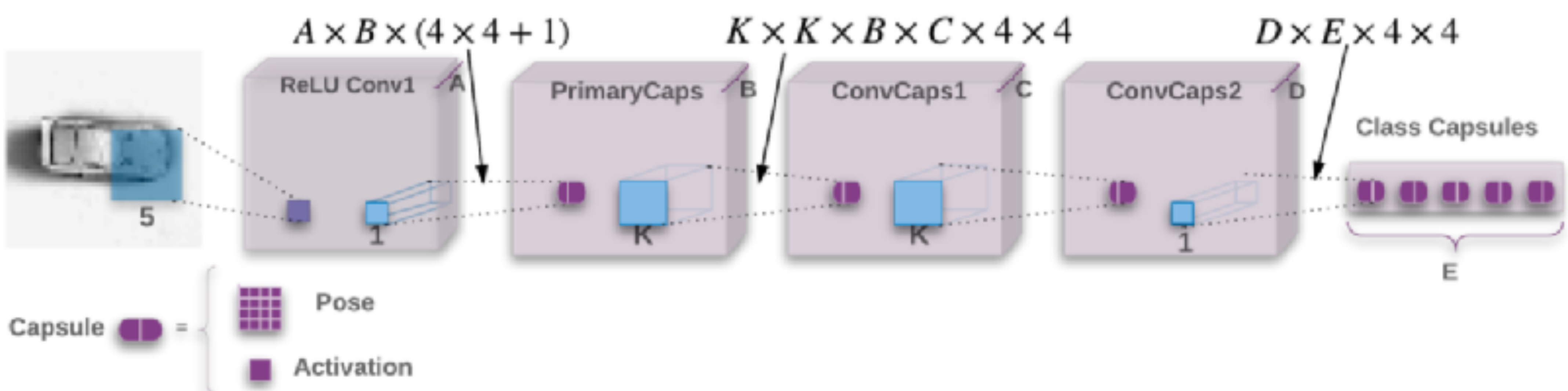
procedure M-STEP($\boldsymbol{a}$, R, V, j) ▷ for one higher-level capsule
    $\forall i \in \Omega_L$: $R_{ij} \leftarrow R_{ij} * \boldsymbol{a}_i$
    $\forall h$: $\mu_j^h \leftarrow \frac{\sum_i R_{ij} V_{ij}^h}{\sum_i R_{ij}}$
    $\forall h$: $(\sigma_j^h)^2 \leftarrow \frac{\sum_i R_{ij}(V_{ij}^h - \mu_j^h)^2}{\sum_i R_{ij}}$
    $cost^h \leftarrow (\beta_v + log(\sigma_j^h)) \sum_i R_{ij}$
    $a_j \leftarrow sigmoid(\lambda(\beta_a - \sum_h cost^h))$

$$\sum_j R_{ij} = a_i$$

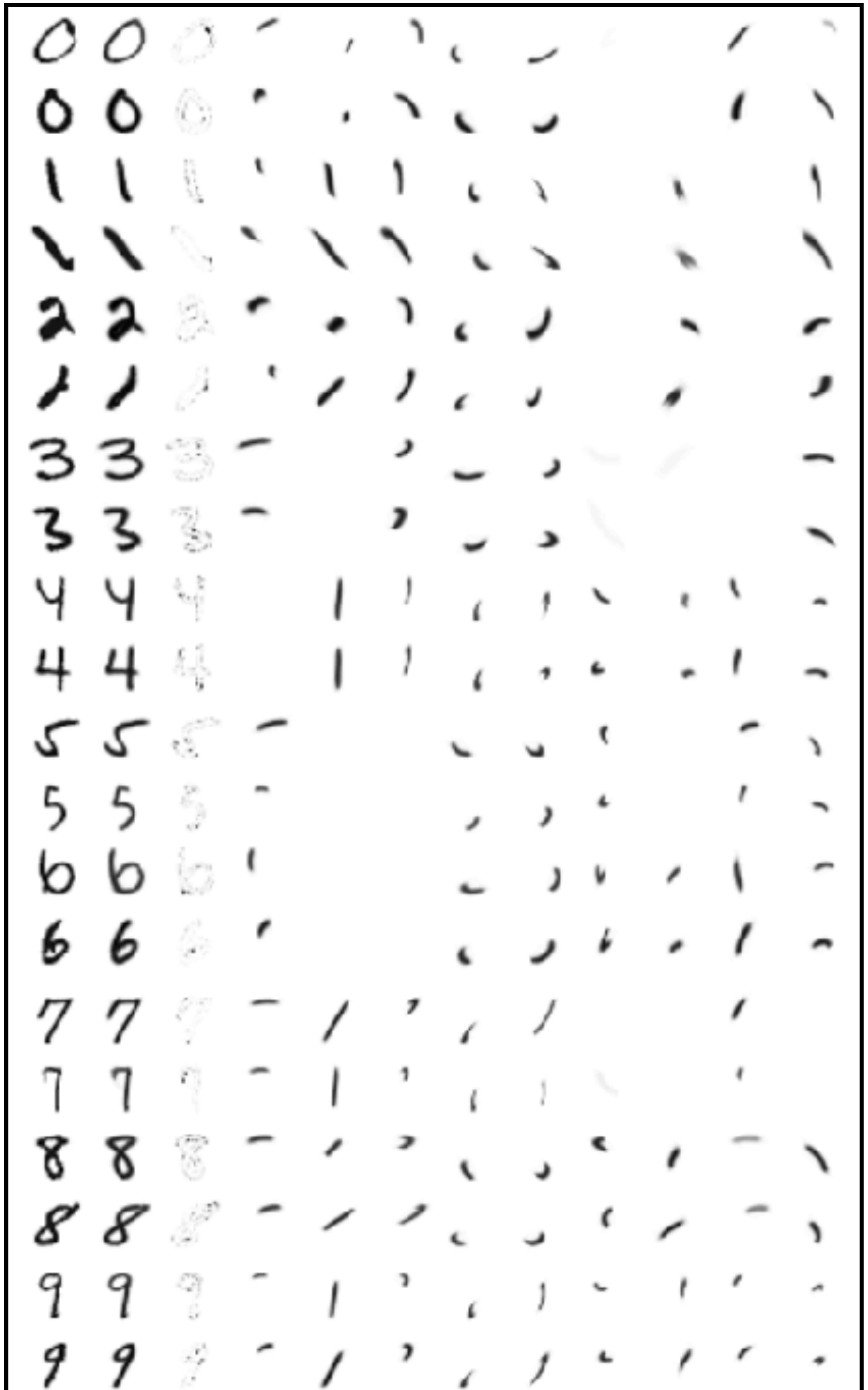
procedure E-STEP(μ, σ, $\boldsymbol{a}$, V, i) ▷ for one lower-level capsule
    $\forall j \in \Omega_{L+1}$: $\boldsymbol{p}_j \leftarrow \frac{1}{\sqrt{\prod_h^H 2\pi(\sigma_j^h)^2}} e^{-\sum_h^H \frac{(V_{ij}^h - \boldsymbol{\mu}_j^h)^2}{2(\sigma_j^h)^2}}$
    $\forall j \in \Omega_{L+1}$: $\boldsymbol{R}_{ij} \leftarrow \frac{\boldsymbol{a}_j p_j}{\sum_{u \in \Omega_{L+1}} \boldsymbol{a}_u p_u}$

$$\sum_j R_{ij} = 1$$

Architecture

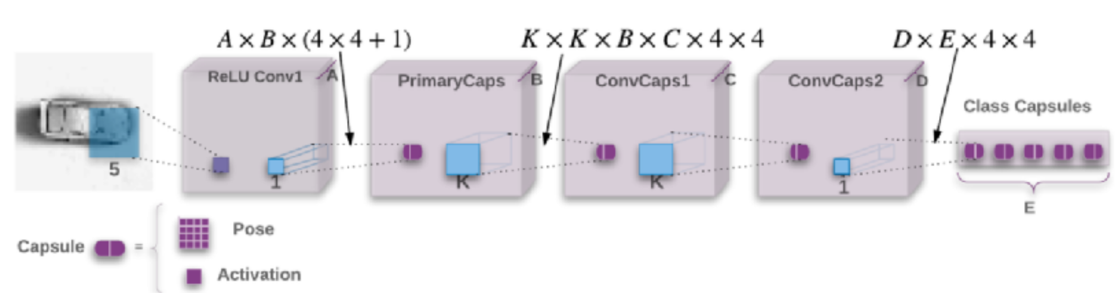


- Convolution + ReLU to filter low level features
- Primary Capsules with B different capsule types
- Convolutional Capsules with C/D different capsule types
- Class capsules detecting E different classes



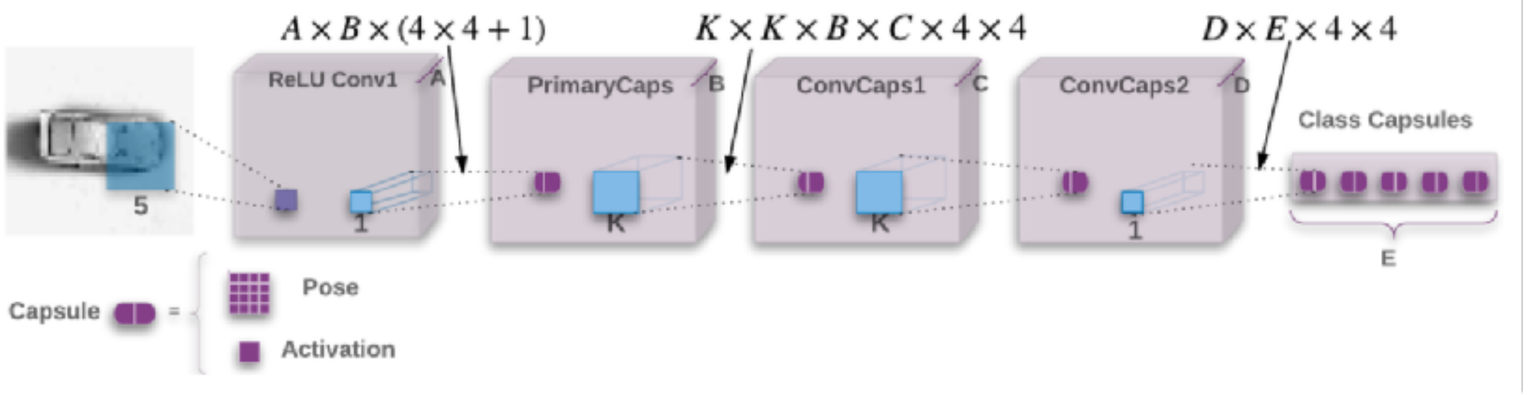
Primary Capsule

- detects specific low level features
- Domain Specific Auto Encoder
- keep spatial information
- learns task specific features



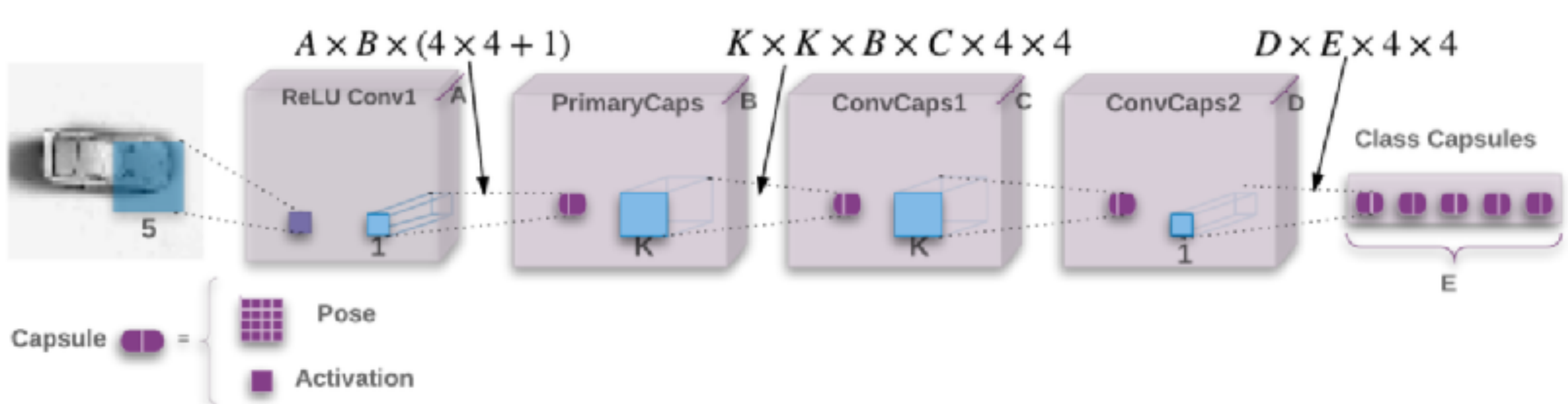
Convolutional Capsule

- routing procedure only for lower capsules in the receptive field of higher level capsules (kernel)
- divide the task of detecting high level features or classes into small steps
- speed up the model



Class Capsules

- use fact: all capsules of the same type detect same entity in different positions
- share transformation matrix between different positions of the same capsule type
- possible addition of receptive field of the lower layer capsules in the computation (coordinate addition)



Spread Loss

$$L_i = (max(0, m - (a_t - a_i))^2, \quad L = \sum_{i \neq t} L_i$$

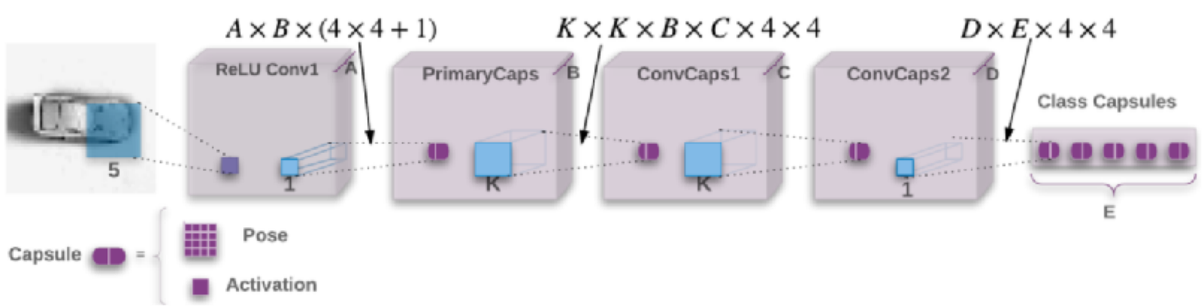
a_t: target class activation $\qquad$ a_i: wrong class activation

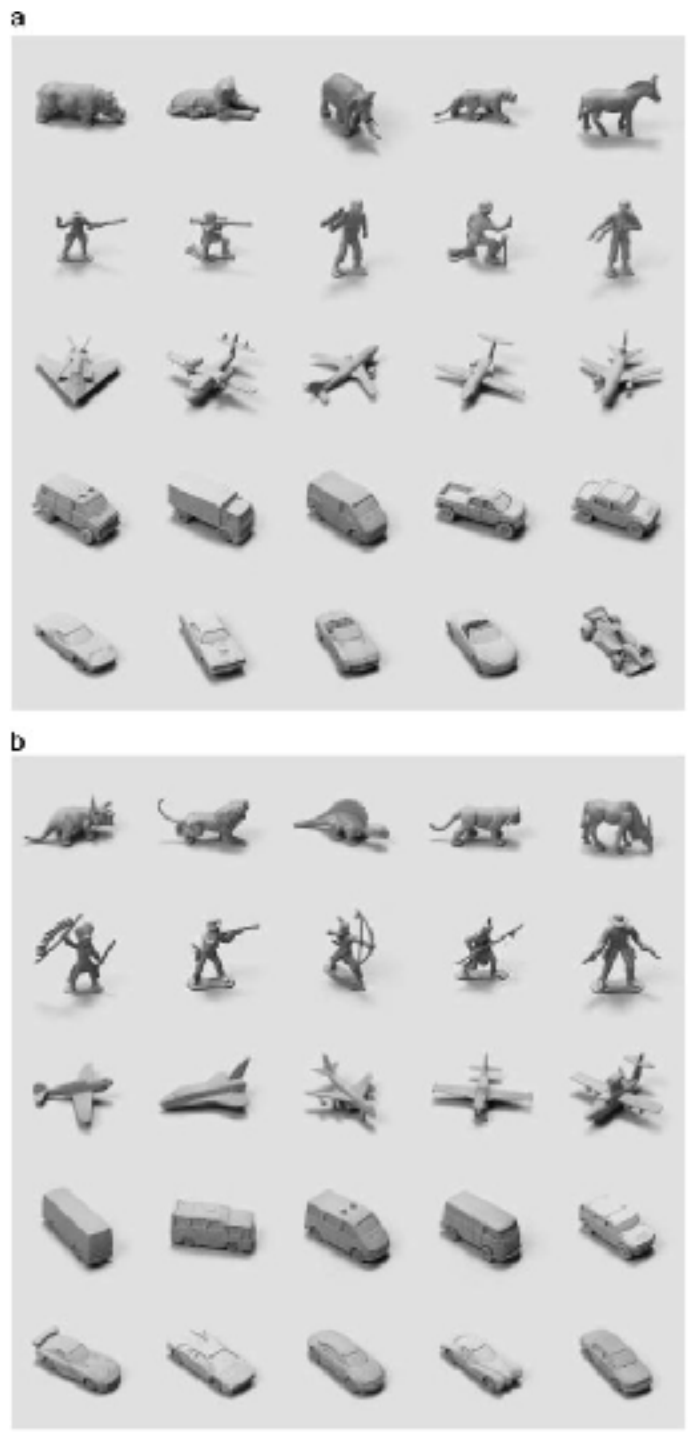
m: margin

- margin is linearly increased during training
- avoids dead capsules and makes training less sensitive

Experiment on smallNORB

- Dataset:
  - 5 classes of toys
  - same entity photographed in different positions
  - pure shape recognition task
  - no disturbing background
  - downsampled to 48x48 pixels
  - training: randomly cropped 32x32
  - test: center cropped 32x32
- Model:





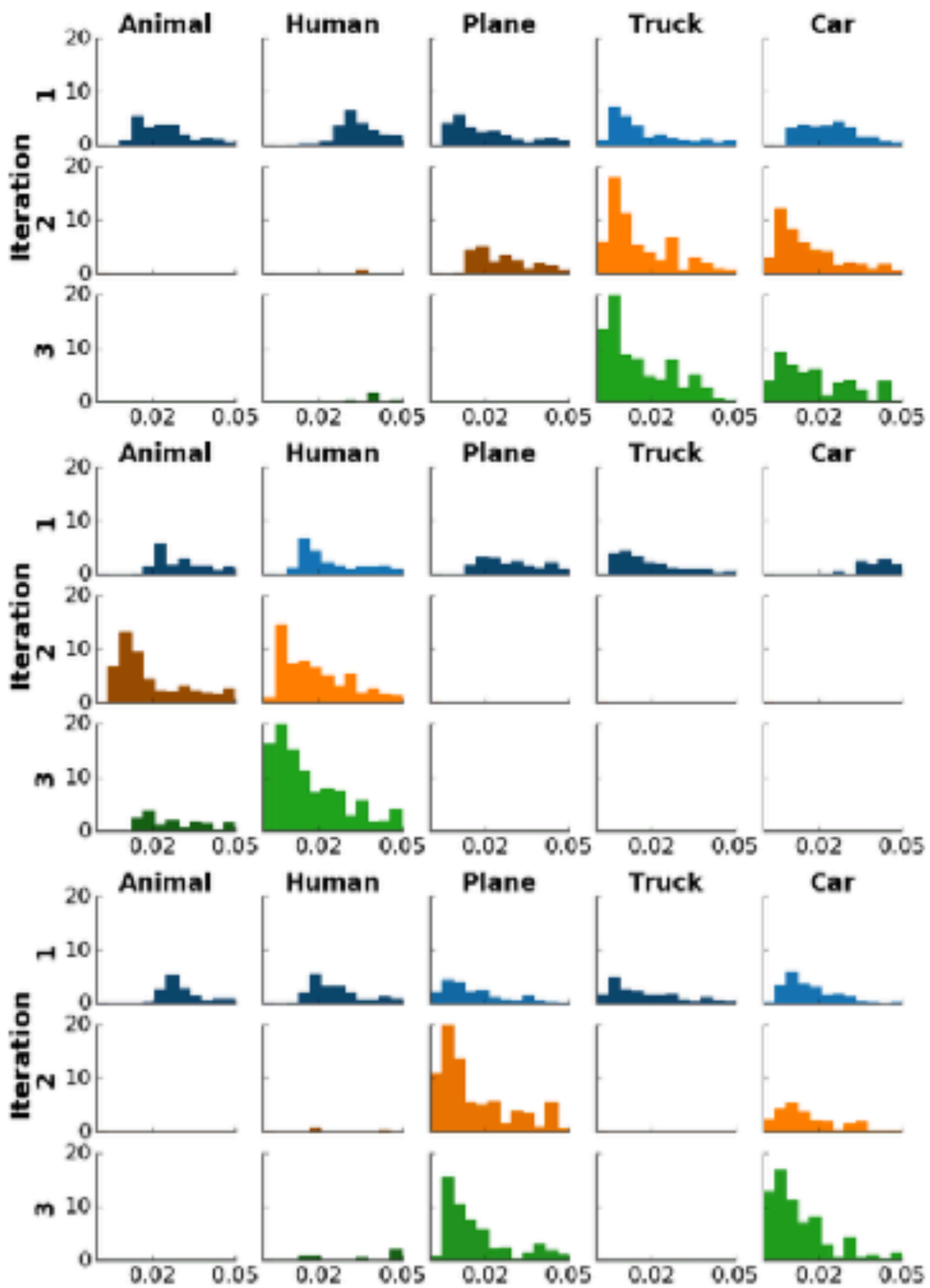
Results

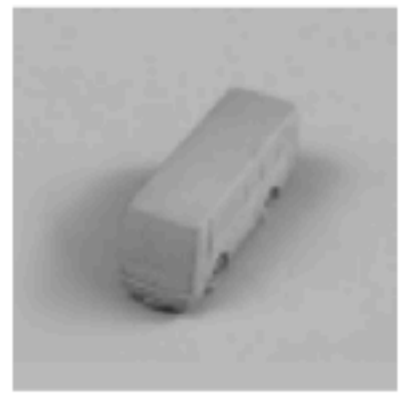
Routing iterations	Pose structure	Loss	Coordinate Addition	Test error rate
1	Matrix	Spread	Yes	9.7%
2	Matrix	Spread	Yes	2.2%
3	Matrix	Spread	Yes	**1.8%**
5	Matrix	Spread	Yes	3.9%
3	Vector	Spread	Yes	2.9%
3	Matrix	Spread	No	2.6%
3	Vector	Spread	No	3.2%
3	Matrix	Margin[1]	Yes	3.2%
3	Matrix	CrossEnt	Yes	5.8%
Baseline CNN with 4.2M parameters				5.2%
CNN of Cireşan et al. (2011) with extra input images & deformations				2.56%
Our Best model (third row), with multiple crops during testing				**1.4%**

- Sabour et al. Capsules achieve 2.7% test error rate
- Ciresan et al. CNN used classical methods to create additional input

- Baseline CNN fine-tuned for this task and trained like Capsule Net
- Matrix Capsule Net has 310k Parameters but smaller model with 68k Parameters achieved 2.2% test error rate

Routing





- Histogram of distances of votes to the main in the last layer
  - x-axis distance of the vote
  - y-axis amount of data

Generalization to novel viewpoints

Test set	Azimuth		Elevation	
	CNN	Capsules	CNN	Capsules
Novel viewpoints	20%	13.5%	17.8%	12.3%
Familiar viewpoints	3.7%	3.7%	4.3%	4.3%

- Trained on 1/3 of training data with specific azimuths/elevation levels
- Capsule Net is trained to match accuracy on familiar viewpoints

Results on other datasets

- NORB:
  MCN: 2.6% TE
  State of the Art: 2.7% TE

- MNIST:
  MCNet: 0.44% TE
  Sabour et. AL.: 0.25% TE
- CIFAR10:
  Matrix Capsule Net: 11.9%TE
  Convolutional Deep Belief Networks(2010): 21.9%

Disadvantages

- computational expensive routing
- models take a massive amount of RAM
- has a problem with backgrounds
- cannot detect two objects of the same class very near to each other
- difficult to scale to datasets with more classes & higher dimensions

Advantages

- design similar to a parse tree and thereby more understandable
- requires less training data
- smaller amount of parameters
- model learns to a degree generative properties
- capsules activation in lower levels works similary to saliency or Region Proposal Networks
- more robust against adversarial examples than standard CNN

Sources

- Slide 15: https://www.cs.toronto.edu/~tijmen/tijmen_thesis.pdf
- Slide 12: https://medium.freecodecamp.org/understanding-capsule-networks-ais-alluring-new-architecture-bdb228173ddc
- Slide 5 to 7: https://hackernoon.com/uncovering-the-intuition-behind-capsule-networks-and-inverse-graphics-part-i-7412d121798d
- Slide 2: https://www.enterprisetech.com/2017/11/22/ais-cool-new-thing-capsule-networks-explained/
- Paper: https://openreview.net/pdf?id=HJWLfGWRb
- Other Resources:
  http://web.cse.ohio-state.edu/~wang.3602/courses/cse5542-2013-spring/6-Transformation_II.pdf
  https://arxiv.org/abs/1802.10204v1

THANK YOU FOR YOUR ATTENTION!