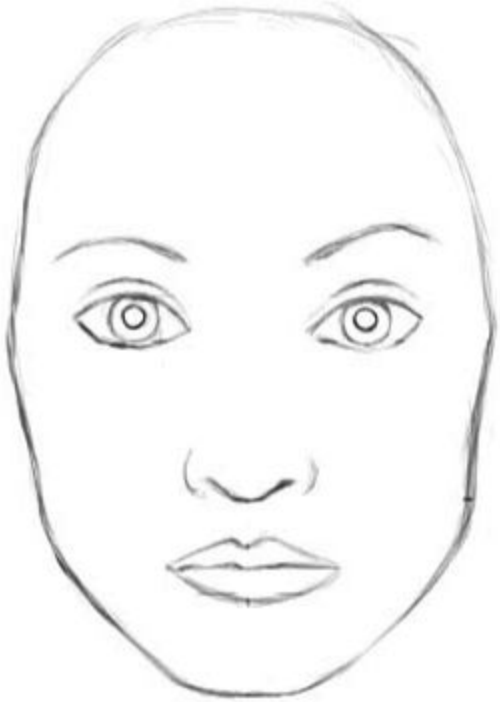
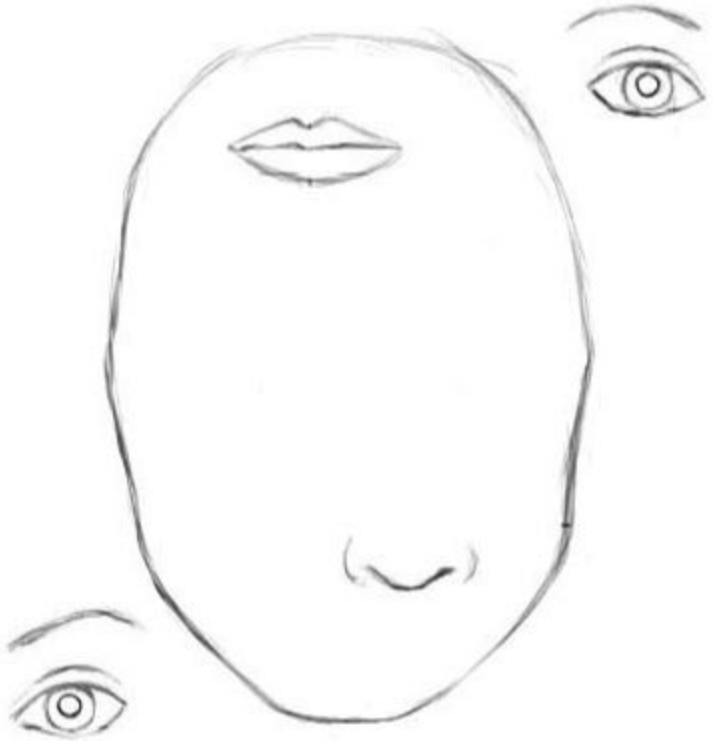
Dynamic Routing Between Capsules

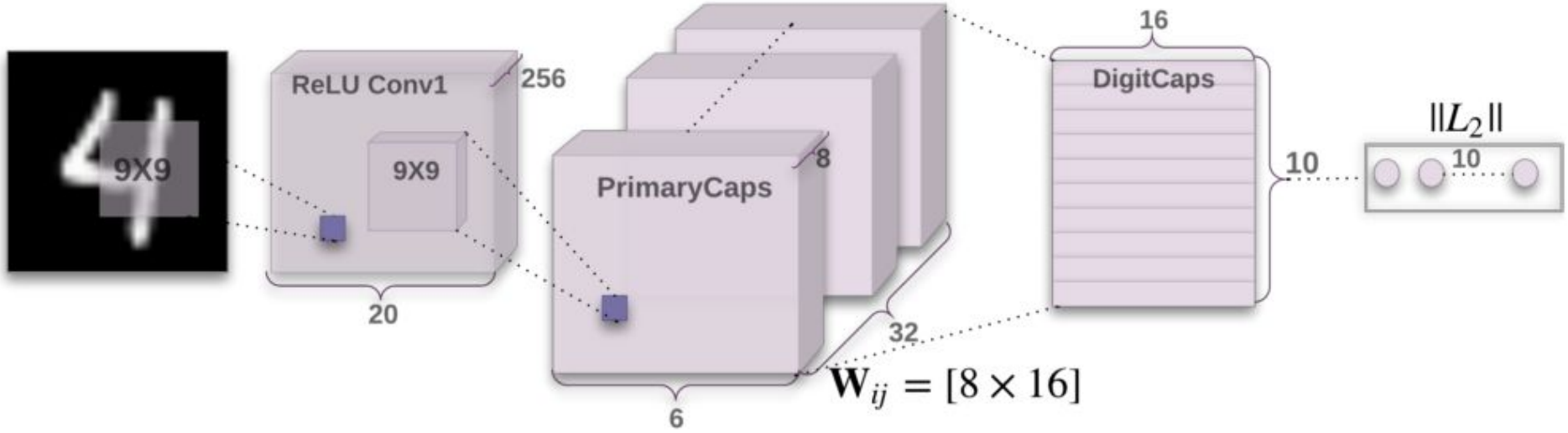
Main Idea

- Conv nets are good at translational invariance, but not great at other invariances (rotation, shadow, etc), room for improvement
- Capsules attempt to capture ‘instantiation parameters’ of features
  - Essentially doing ‘inverse graphics’
- Each feature is a vector rather than a scalar
  - Magnitude is ‘probability of the feature’
  - Direction is ‘parameters of the feature’
- Higher level features can depend on the parameters as well as the probability

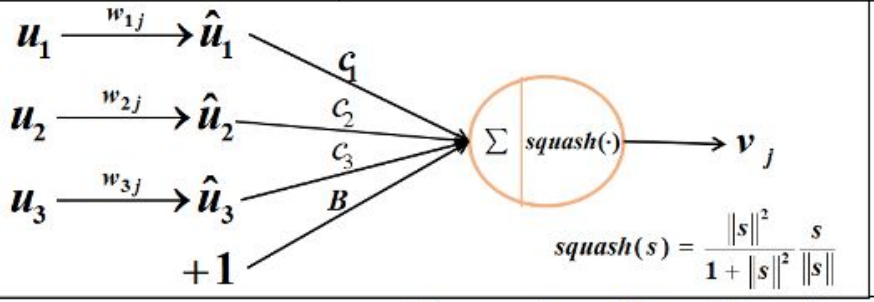


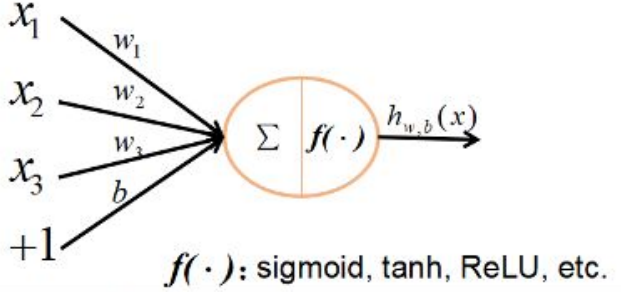


Encoder Architecture



		capsule	VS.	traditional neuron
Input from low-level neurons/capsules		$\text{vector}(u_i)$		$\text{scalar}(x_i)$
Operations	Linear/Affine Transformation	$\hat{u}_{j\|i} = W_{ij} u_i + B_j$ (Eq. 2)		$a_{j\|i} = w_{ij} x_i + b_j$
	Weighting	$s_j = \sum_i c_{ij} \hat{u}_{j\|i}$ (Eq. 2)		$z_j = \sum_{i=1}^{3} 1 \cdot a_{j\|i}$
	Summation			
	Non-linearity activation	$v_j = squash(s_j)$ (Eq. 1)		$h_{w,b}(x) = f(z_j)$
output		$\text{vector}(v_j)$		$\text{scalar}(h)$





Capsule = New Version Neuron!
vector in, vector out VS. scalar in, scalar out

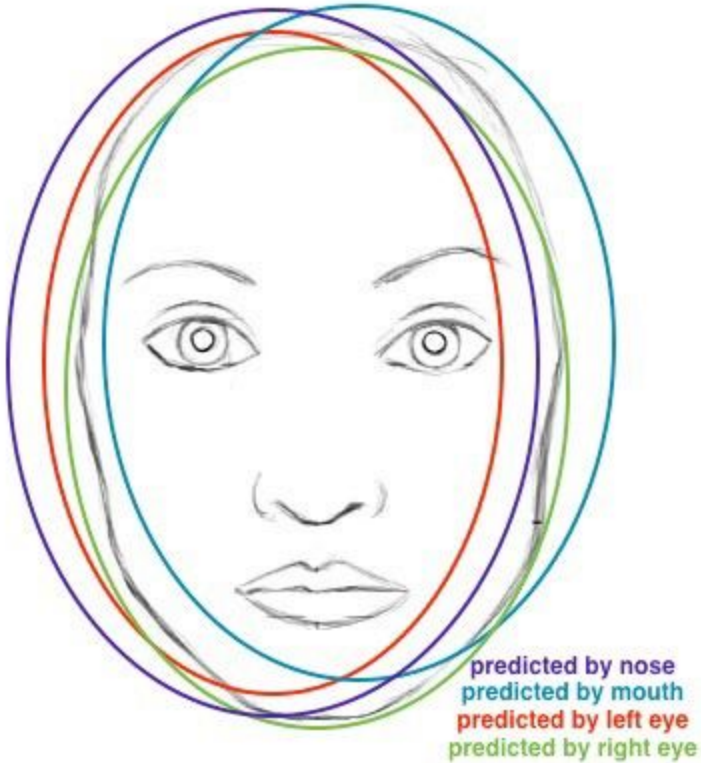
For each capsule i, the sum of c_ij is 1

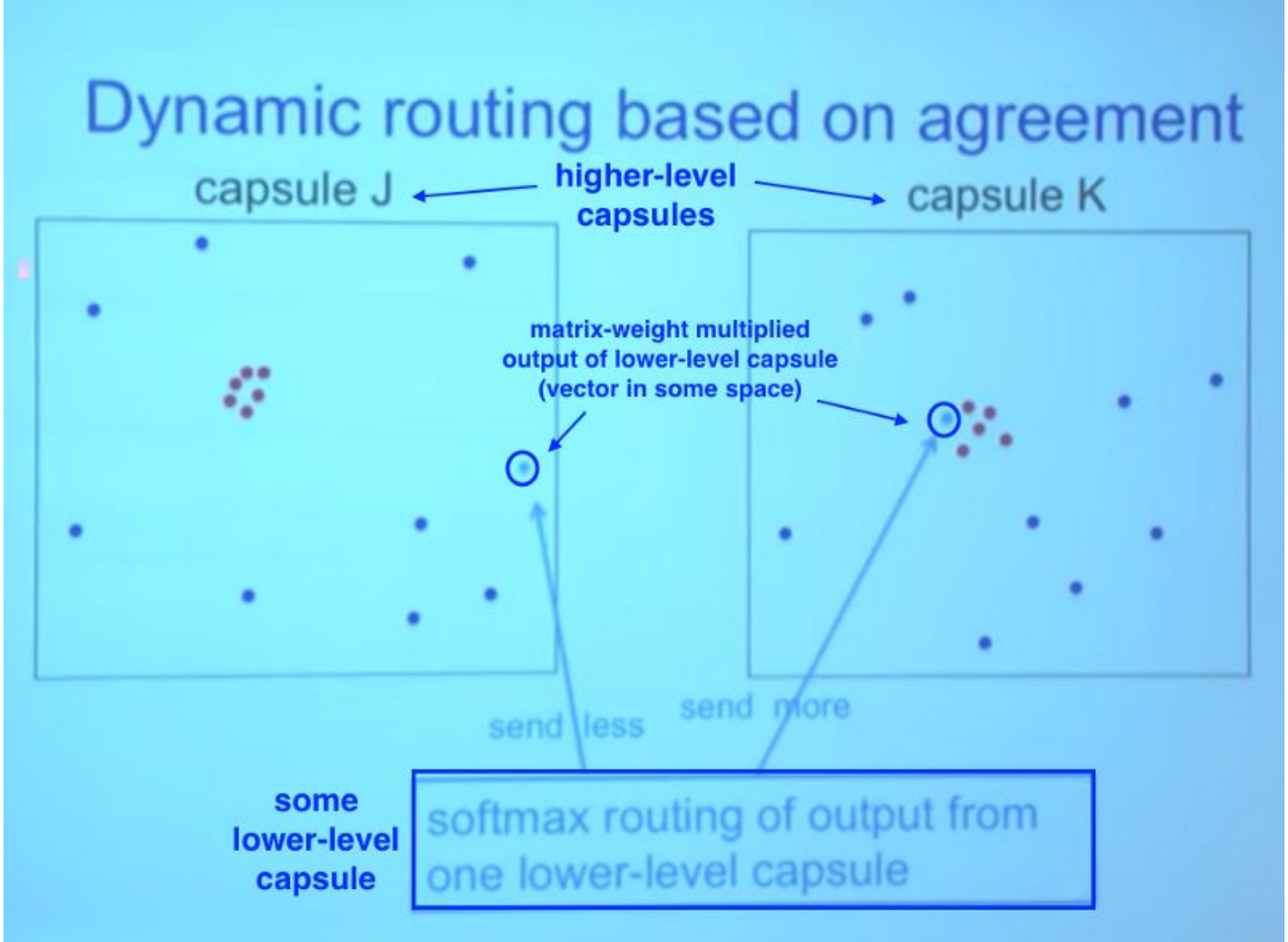
$$\mathbf{v}_j = \frac{\|\mathbf{s}_j\|^2}{1+\|\mathbf{s}_j\|^2} \frac{\mathbf{s}_j}{\|\mathbf{s}_j\|}$$

additional “squashing” unit scaling

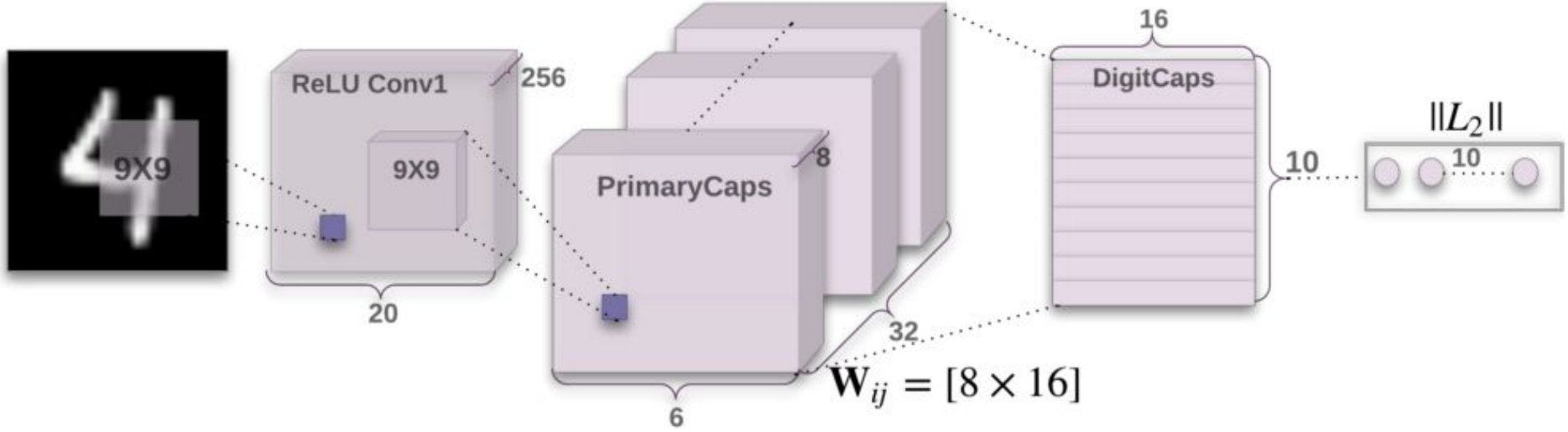
Procedure 1 Routing algorithm.

1: **procedure** ROUTING($\hat{\boldsymbol{u}}_{j|i}, r, l$)
2: for all capsule i in layer l and capsule j in layer $(l+1)$: $b_{ij} \leftarrow 0$.
3: **for** r iterations **do**
4: for all capsule i in layer l: $\mathbf{c}_i \leftarrow \texttt{softmax}(\mathbf{b}_i)$ ▷ `softmax` computes Eq. 3
5: for all capsule j in layer $(l+1)$: $\mathbf{s}_j \leftarrow \sum_i c_{ij} \hat{\mathbf{u}}_{j|i}$
6: for all capsule j in layer $(l+1)$: $\mathbf{v}_j \leftarrow \texttt{squash}(\mathbf{s}_j)$ ▷ `squash` computes Eq. 1
7: for all capsule i in layer l and capsule j in layer $(l+1)$: $b_{ij} \leftarrow b_{ij} + \hat{\mathbf{u}}_{j|i}.\mathbf{v}_j$
return $\mathbf{v}_j$


predicted by nose
predicted by mouth
predicted by left eye
predicted by right eye


Dynamic routing based on agreement
capsule J
higher-level capsules
capsule K
matrix-weight multiplied output of lower-level capsule (vector in some space)
send less
send more
some lower-level capsule
softmax routing of output from one lower-level capsule

Encoder


9X9
ReLU Conv1
9X9
256
20
PrimaryCaps
8
6
32
16
DigitCaps
10
$\|L_2\|$
10
$\mathbf{W}_{ij} = [8 \times 16]$

CapsNet Loss Function

calculated for correct DigitCap

calculated for incorrect DigitCaps

loss term for one DigitCap

L2 norm

L2 norm

$$L_c = T_c \max(0, m^+ - ||\mathbf{v}_c||)^2 + \lambda(1 - T_c) \max(0, ||\mathbf{v}_c|| - m^-)^2$$

1 when correct DigitCap, 0 when incorrect

zero loss when correct prediction with probability greater than 0.9, non-zero otherwise

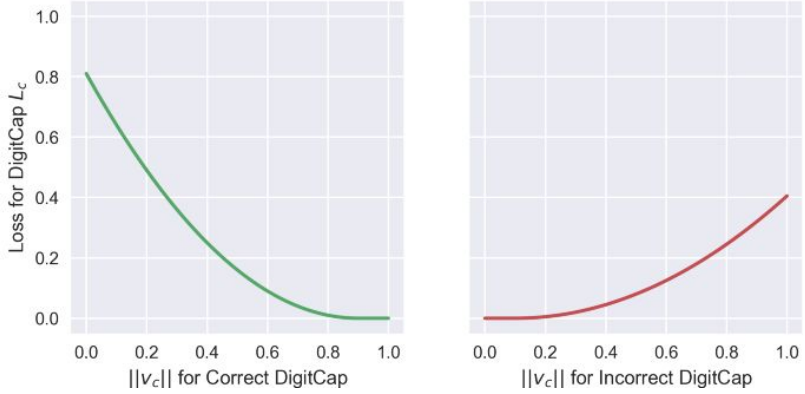
0.5 constant used for numerical stability

1 when incorrect DigitCap, 0 when correct

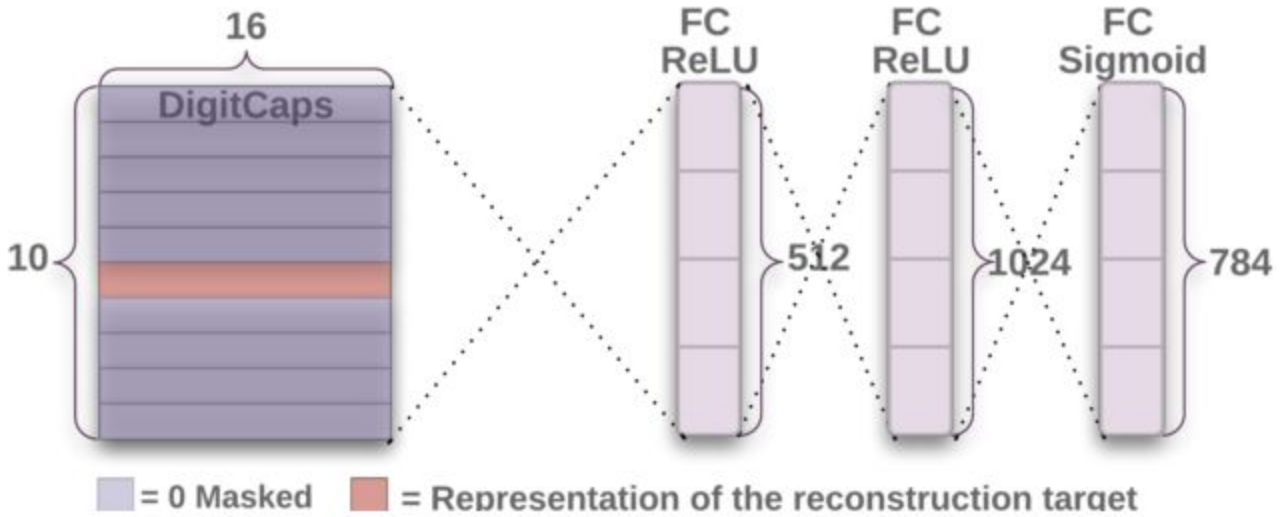
zero loss when incorrect prediction with probability less than 0.1, non-zero otherwise

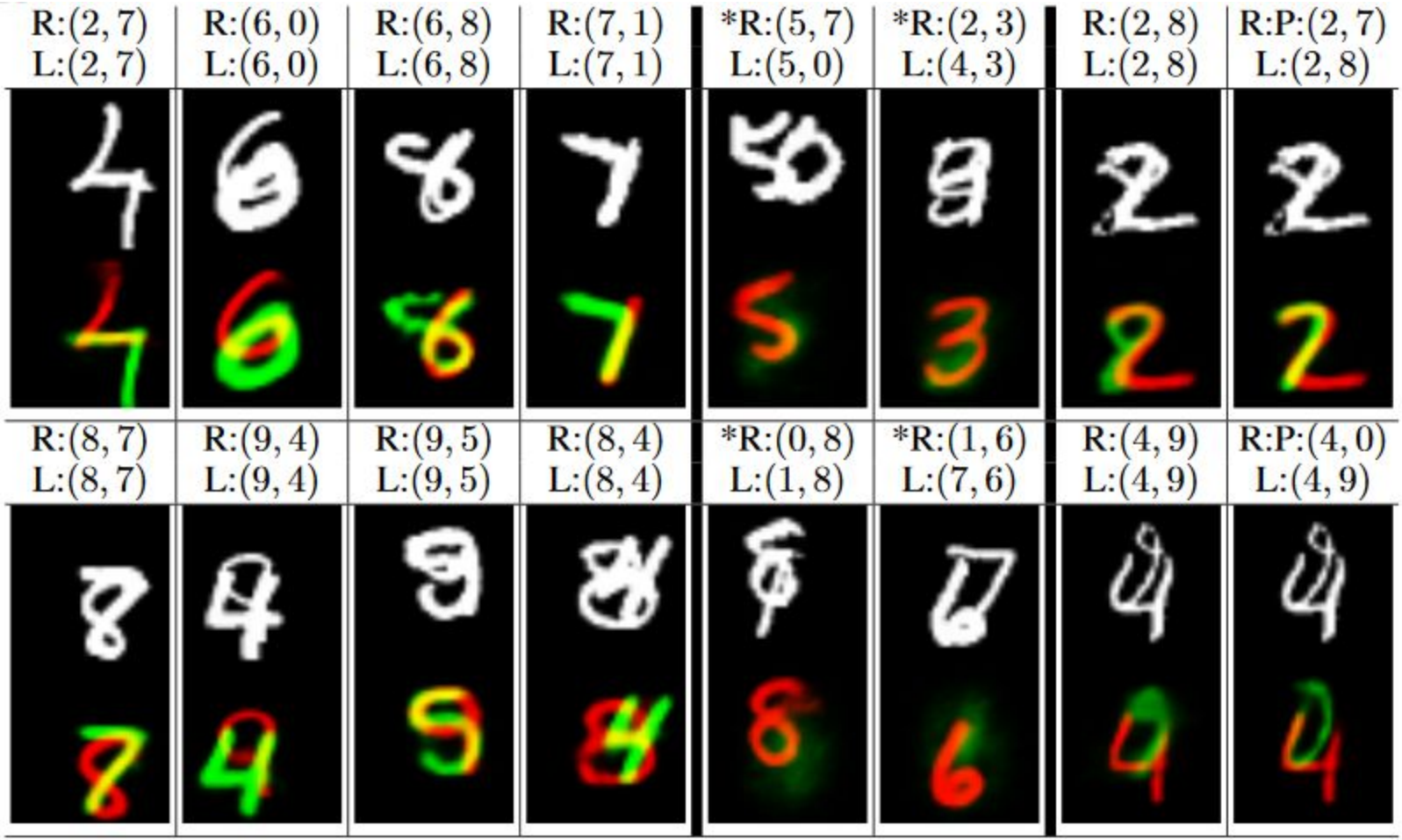
Note: correct DigitCap is one that matches training label, for each training example there will be 1 correct and 9 incorrect DigitCaps

Loss Function Value for Correct and Incorrect DigitCap



Decoder


16
DigitCaps
10
FC
ReLU
512
FC
ReLU
1024
FC
Sigmoid
784
= 0 Masked
= Representation of the reconstruction target


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)

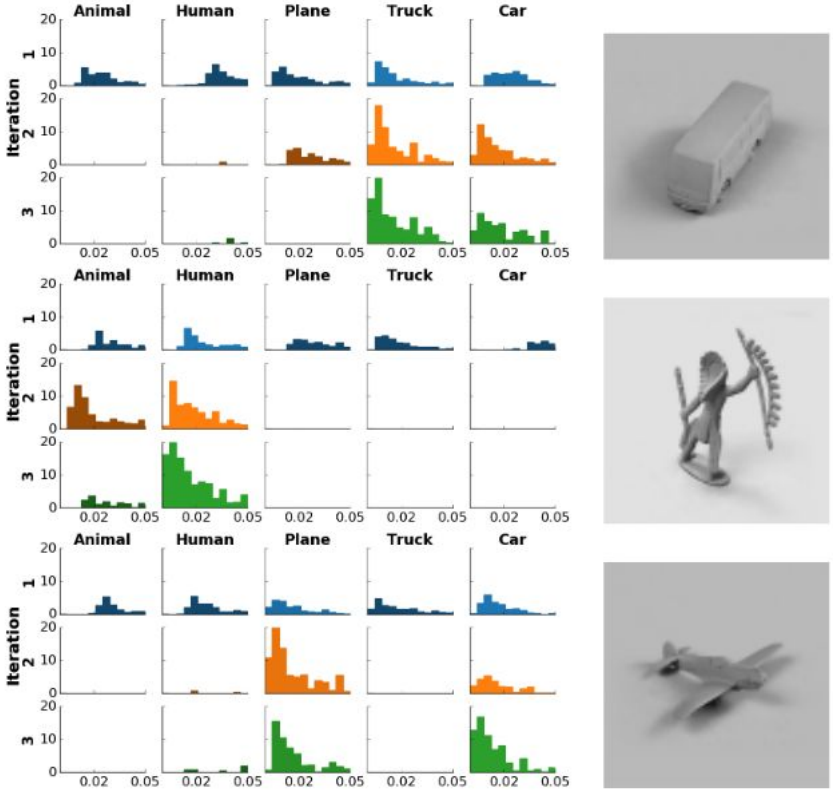


Figure 2: Histogram of distances of votes to the mean of each of the 5 final capsules after each routing iteration. Each distance point is weighted by its assignment probability. All three images are selected from the smallNORB test set. The routing procedure correctly routes the votes in the truck and the human example. The plane example shows a rare failure case of the model where the plane is confused with a car in the third routing iteration. The histograms are zoomed-in to visualize only votes with distances less than 0.05. Fig. B.2 shows the complete histograms for the "human" capsule without clipping the x-axis or fixing the scale of the y-axis.

Procedure 1 Routing algorithm[1] returns **activation** and **pose** of the capsules in layer $L+1$ given the **activations** and **votes** of capsules in layer L. V_{ich} is an H dimensional vote from capsule i with activation a_i in layer L to capsule c 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)
  $\forall i, c$: $R_{ic} \leftarrow 1/size(L+1)$
  for t iterations **do**
    $\forall c$: $M_{c:}, S_{c:}, \boldsymbol{a}'_c \leftarrow$ M-STEP($R_{:c}$, $\boldsymbol{a}$, $V_{:c:}$)
    $\forall i$: $R_{i:} \leftarrow$ E-STEP(M, S, $\boldsymbol{a}'$, $V_{i::}$)
  return $\boldsymbol{a}'$, M

procedure M-STEP($\boldsymbol{r}, \boldsymbol{a}, V'$) ▷ for one higher-level capsule
  $\forall i$: $\boldsymbol{r}'_i \leftarrow \boldsymbol{r}_i * \boldsymbol{a}_i$
  $\forall h$: $\boldsymbol{\mu}_h \leftarrow \frac{\sum_i \boldsymbol{r}'_i V'_{ih}}{\sum_i \boldsymbol{r}'_i}$
  $\forall h$: $\boldsymbol{\sigma}_h^2 \leftarrow \frac{\sum_i \boldsymbol{r}'_i (V'_{ih} - \boldsymbol{\mu}_h)^2}{\sum_i \boldsymbol{r}'_i}$
  $cost_h \leftarrow (\beta_v + log(\boldsymbol{\sigma}_h)) \sum_i \boldsymbol{r}'_i$
  $a' \leftarrow sigmoid(\lambda(\beta_a - \sum_h cost_h))$
  return $\boldsymbol{\mu}, \boldsymbol{\sigma}, a'$

procedure E-STEP($\boldsymbol{a}'$, S, M, V'') ▷ for one lower-level capsule
  $\forall c$: $\boldsymbol{p}_c \leftarrow \frac{1}{\sqrt{\prod_h^H 2\pi \boldsymbol{S}_{ch}^2}} e^{-\sum_h^H \frac{(V''_{ch} - \boldsymbol{M}_{ch})^2}{2\boldsymbol{S}_{ch}^2}}$
  $\forall c$: $\boldsymbol{r}_c \leftarrow \frac{\boldsymbol{a}'_c p_c}{\sum_j \boldsymbol{a}'_j p_j}$
  return $\boldsymbol{r}$

Resources

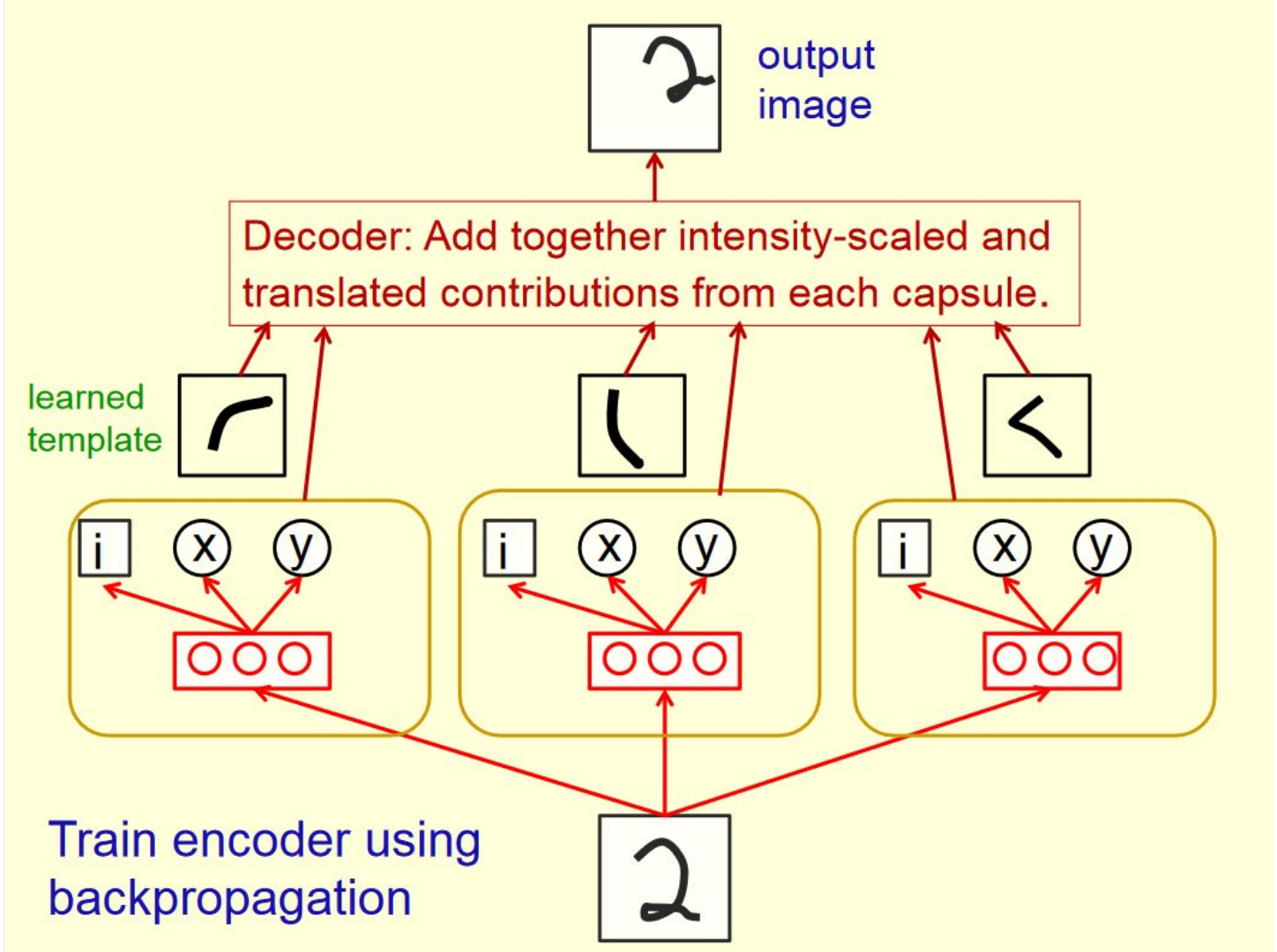
https://arxiv.org/abs/1710.09829

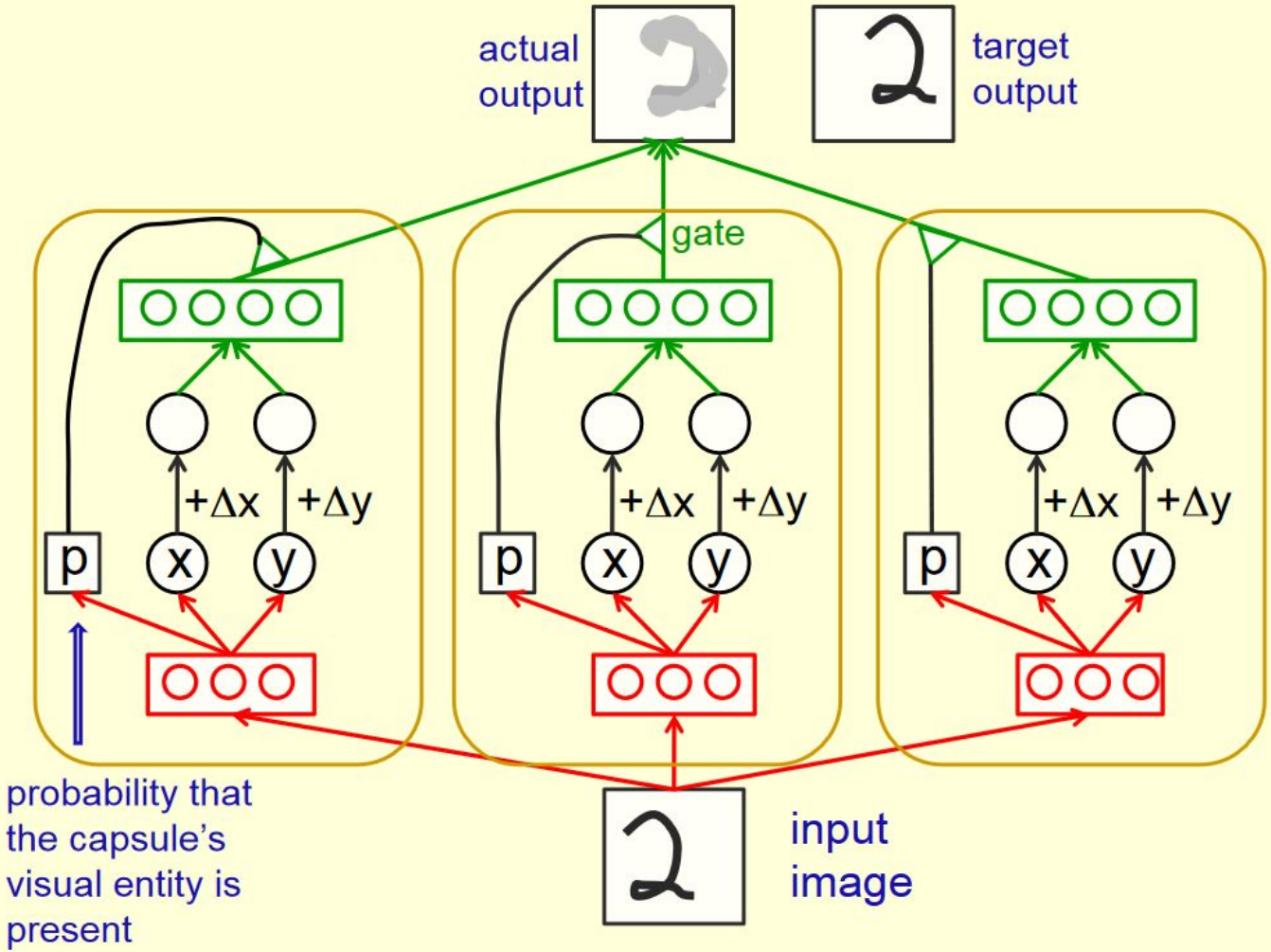
https://pechyonkin.me/capsules-1/

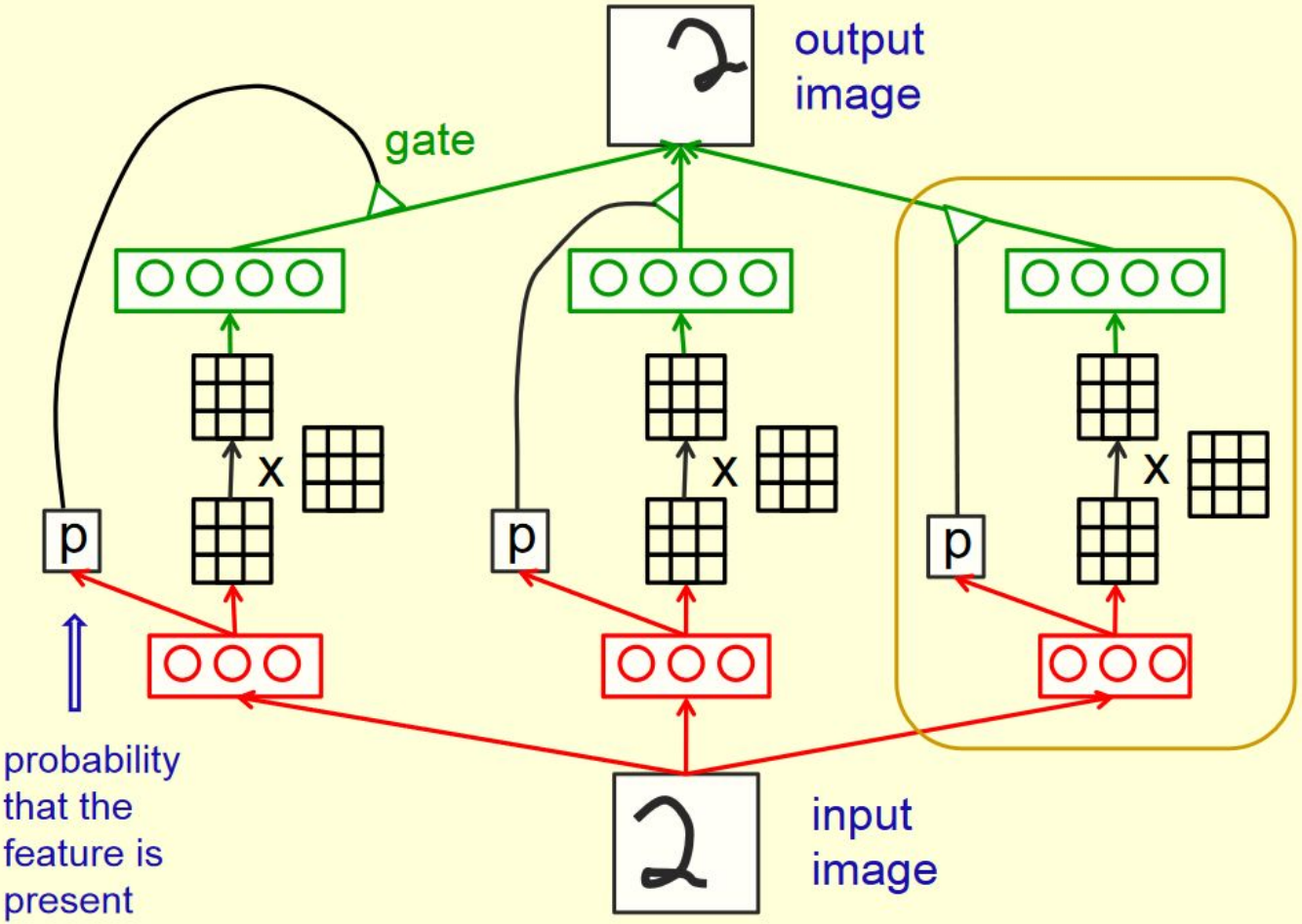
https://github.com/naturomics/CapsNet-Tensorflow/

https://github.com/sekwiatkowski/awesome-capsule-networks

http://helper.ipam.ucla.edu/publications/gss2012/gss2012_10754.pdf


output
image
Decoder: Add together intensity-scaled and
translated contributions from each capsule.
learned
template
i
x
y
i
x
y
i
x
y
Train encoder using
backpropagation


actual
output
target
output
gate
+Δx
+Δy
p
x
y
probability that
the capsule's
visual entity is
present
input
image


output
image
gate
x
x
x
p
p
p
probability
that the
feature is
present
input
image