Capsule Networks

Rıza Velioğlu
Sinan Kuşçu
Sinan Harms

Content

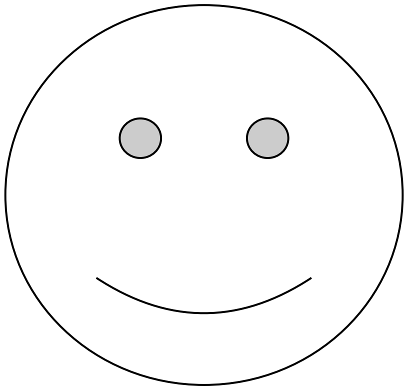
- Main Idea
  - CNN vs CapsNet
- Concepts
  - Hierarchy of parts
  - Inverse Graphics
  - Capsules
- Routing by agreement
- CapsNet Architecture
  - Training & Loss

Main Idea

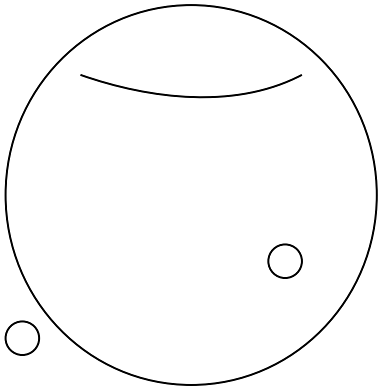
Convolutional Neural Networks

- most commonly applied in analyzing visual imagery
- main component → convolutional layer
    - detects important features in image pixel
    - deeper layers detect simple features
    - higher layers combine simple features into more complex features
- max pooling layers → reduce spacial size of data
- dense layer → combine very high level features and make predictions

CNN Drawbacks



This is a smiley face



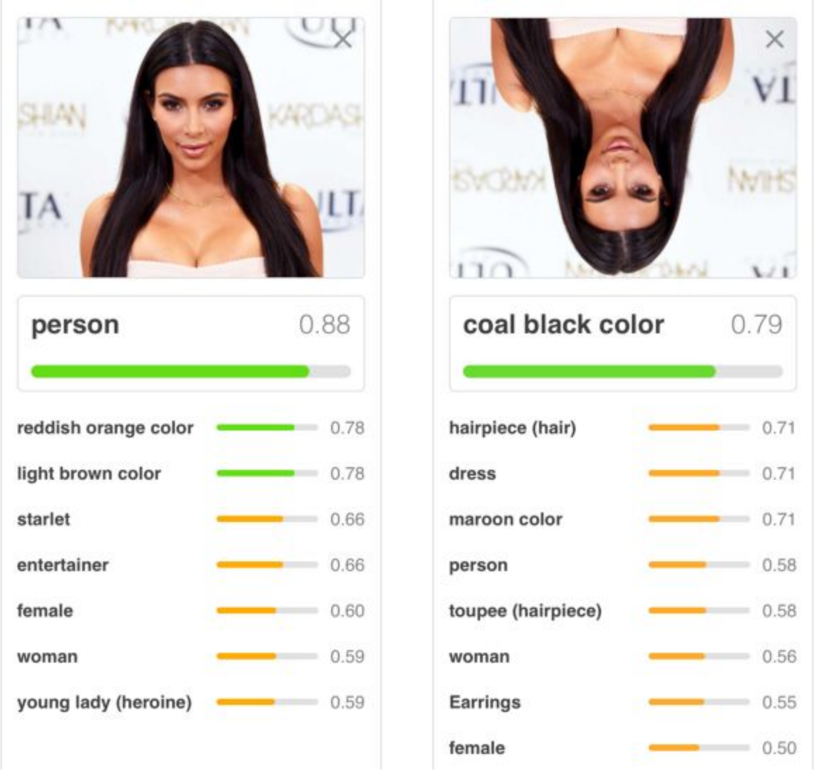
Is this also a smiley face?

Problem:
```
if (2 eyes & 1 nose & 1 mouth):
     it's a face
```

CNN Drawbacks - max pooling

- messenger between two layers
- tells layers about presence of parts, but not spatial relationship
- strips information to create translational invariance
- does not make the model view point invariant

CNN Drawbacks - Rotation Invariance



→ no consideration of rotational relationship

Source: https://www.freecodecamp.org/news/understanding-capsule-networks-ais-alluring-new-architecture-bdb228173ddc/

Capsule Networks - Intuition

Intuition:

- hierarchical pose relationships make it easy for the model to understand that the object it sees is the same, but from a different point of view

Example:

source

Capsule Networks

Solution:

```
if (2 adjacent eyes & nose under eyes & mouth under nose):

     It’s a face
```

- Replace scalar-output feature vectors with vector-output capsules
- Replace max-pooling layer with “routing-by-agreement”

CNN vs. CapsNet

smallNORB benchmark tests:

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%

CapsNets vs. CNNs

Pro

- view point invariance
- fraction of data required to learn model
- consideration of pose relationships

Potential Cons:

- slower than modern deep learning models
- not tested on different data sets

Concepts

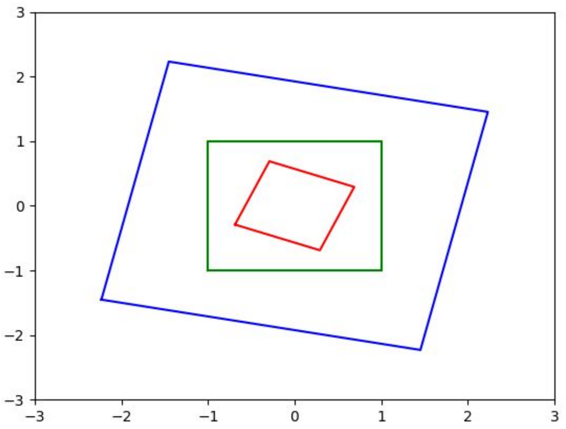
(Computer) Graphics

$$T_{\lambda,\theta} = \begin{pmatrix} \lambda \cos\theta & \sin\theta \\ -\sin\theta & \lambda\cos\theta \end{pmatrix}$$

T is a transformation matrix
Theta specifies the rotation
Lambda specifies the sizing

$$\begin{pmatrix} \lambda = 0.2 \\ \theta = 0.5 \end{pmatrix} \qquad T_{\square}\begin{pmatrix} \lambda = 0.2 \\ \theta = 0.5 \end{pmatrix}$$

$$\begin{pmatrix} \lambda = 0.4 \\ \theta = 2 \end{pmatrix} \qquad T_{\square}\begin{pmatrix} \lambda = 0.4 \\ \theta = 2 \end{pmatrix}$$

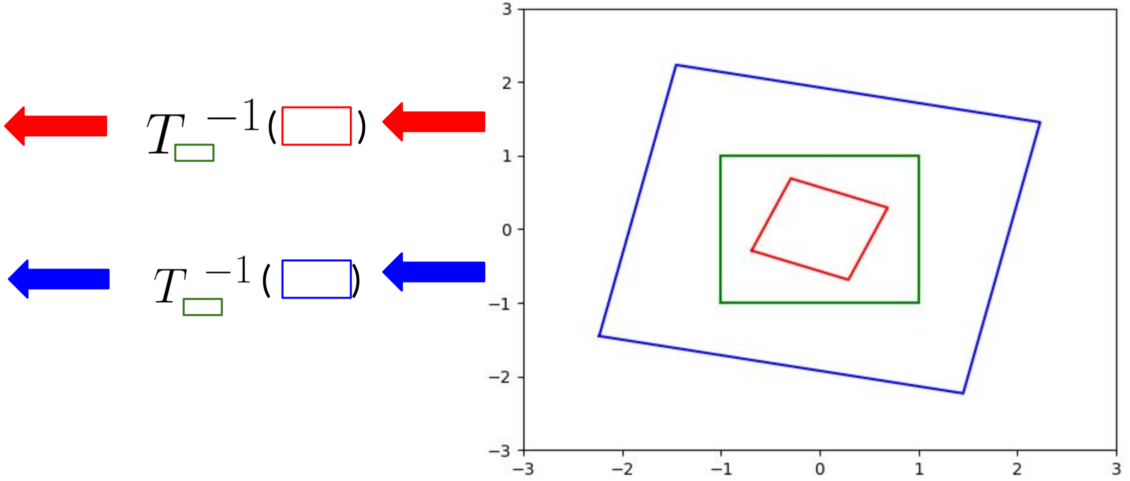


Inverse Graphics

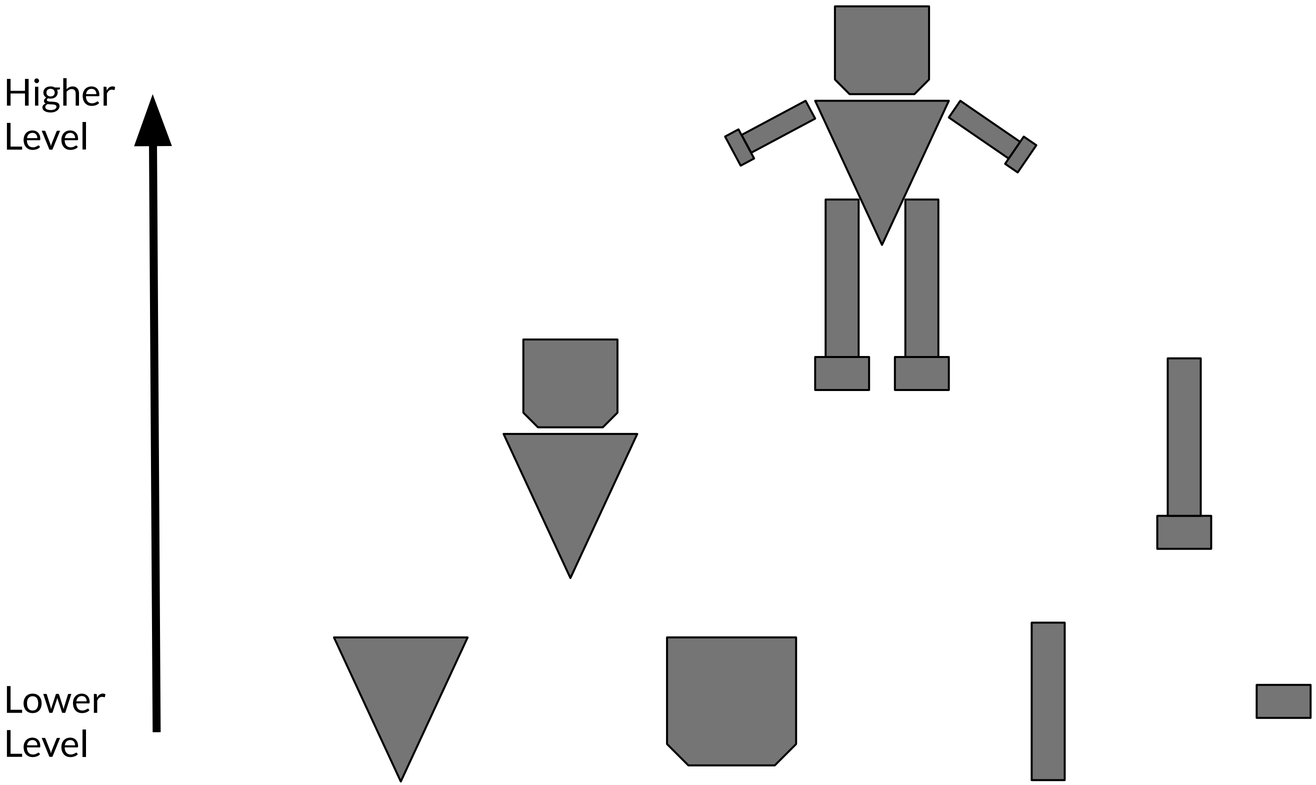
Extracting the pose parameters of an object (rotation,size…) is called inverse graphics

$$\begin{pmatrix} \lambda = 0.2 \\ \theta = 0.5 \end{pmatrix}$$

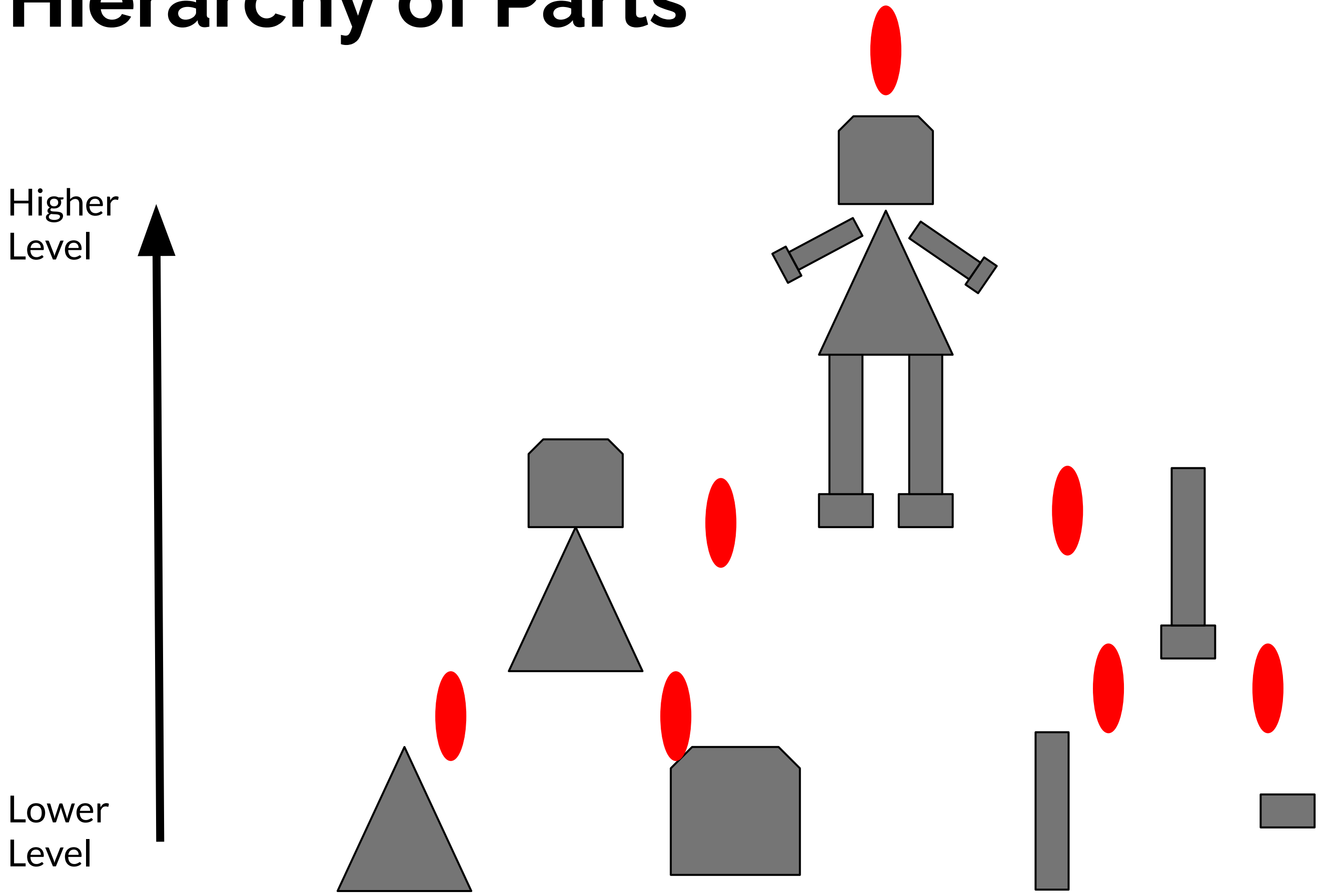
$$\begin{pmatrix} \lambda = 0.4 \\ \theta = 2 \end{pmatrix}$$



Hierarchy of Parts



Hierarchy of Parts



The pose of lower level parts also decides about higher level parts

The pose of a part is detected by capsules

Capsules

Feed your Neural Network with capsules and it will see the truth!

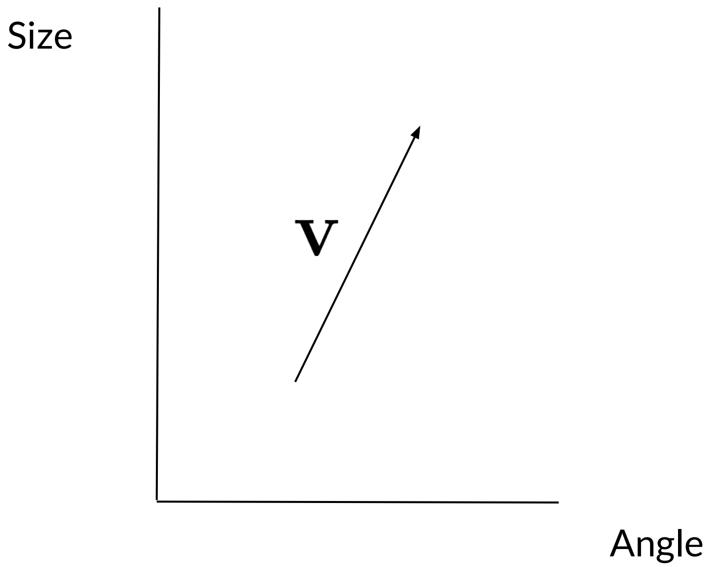
Capsules

- A capsule is a fully connected neural network
- It learns to extract the pose and presence for a given object
- It gets the inputs from a conv layer or lower capsule layers
    - The conv layer converts pixel intensities to the activities of local feature detectors
- It outputs a activation vector, which encodes
    - The probability that an object is present
    - A pose matrix

Capsules

<u>Activation vector</u>

Length = Estimated probability of presence
Direction = Objects estimated pose parameter



Capsules

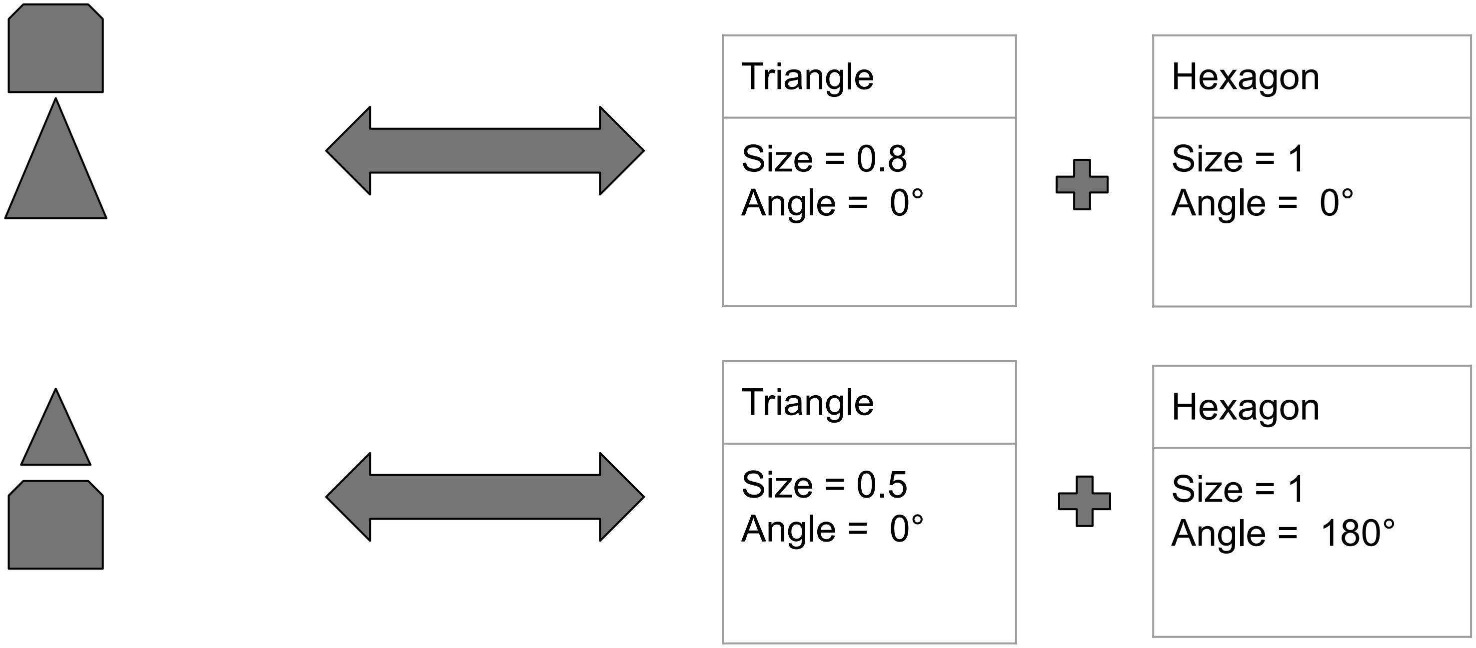
By **Squashing** it can be ensured that the output vector of the capsule j has a length between 0 and 1.

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

Weight factor

Normalized total input of capsule j

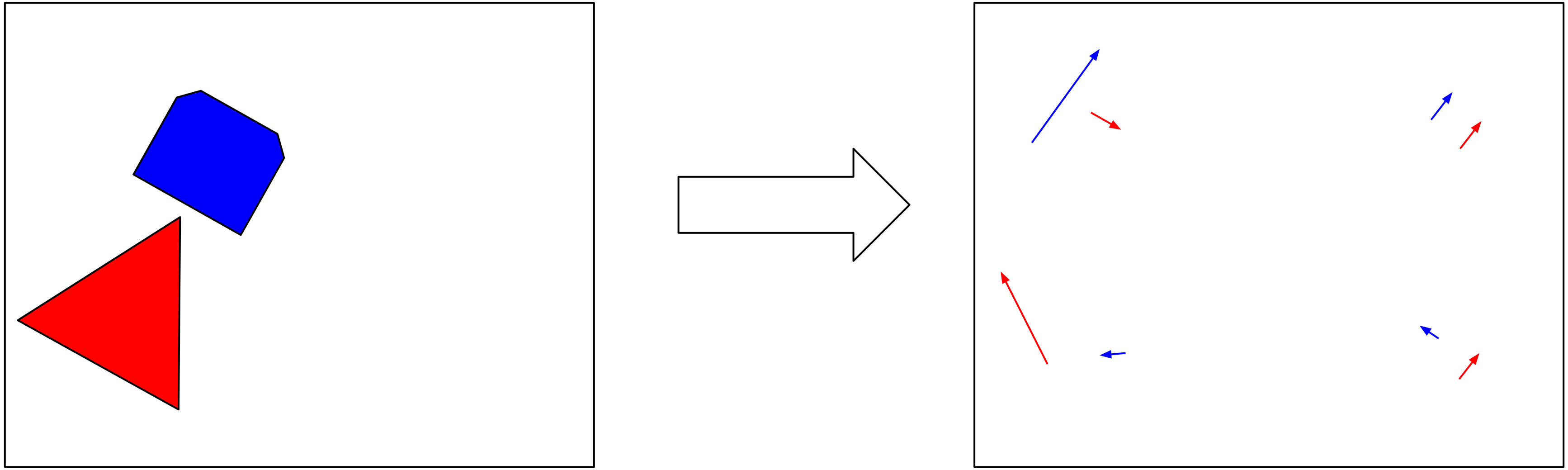
Routing by agreement



Routing by agreement

Output of convolutional layer

Activity vector of hexagon capsule
Activity vector of triangle capsule



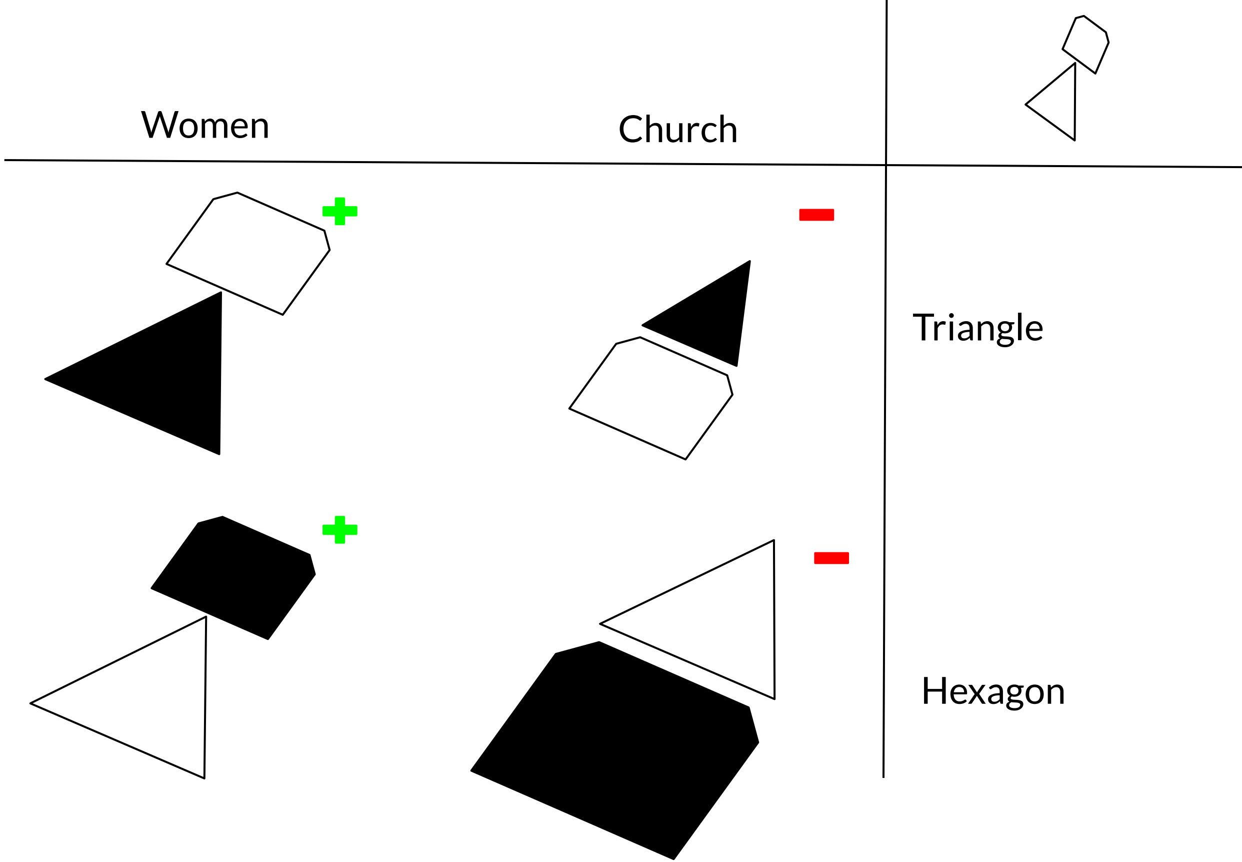
Hexagon capsule detects a hexagon in the upper-left corner
Triangle capsule detects a triangle in the lower-left corner

Does these parts belong to a church or women?

—

Routing by agreement

- The output of lower layer capsules is routed to a higher layer capsule only if there is an agreement between both
- An agreement is reached if lower parts predict similar pose and presence about the higher capsule as the higher capsule do



- The body and head capsule have an agreement about the women capsule
- The output of triangle and hexagon capsule is routed to the women capsule

Routing by agreement

The lower capsule i predicts the pose and presence of part j under the assumption, that the lower part i is part of the higher part j

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

Transformation matrix for lower part i and higher part j

Input of capsule i

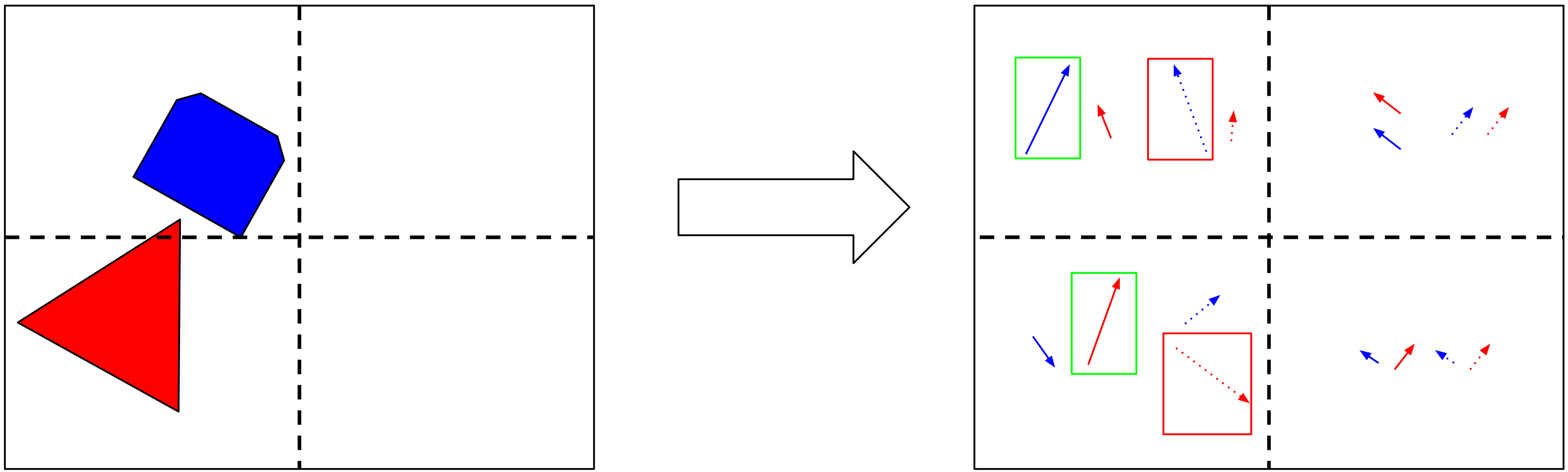
Routing by agreement

Prediction for church

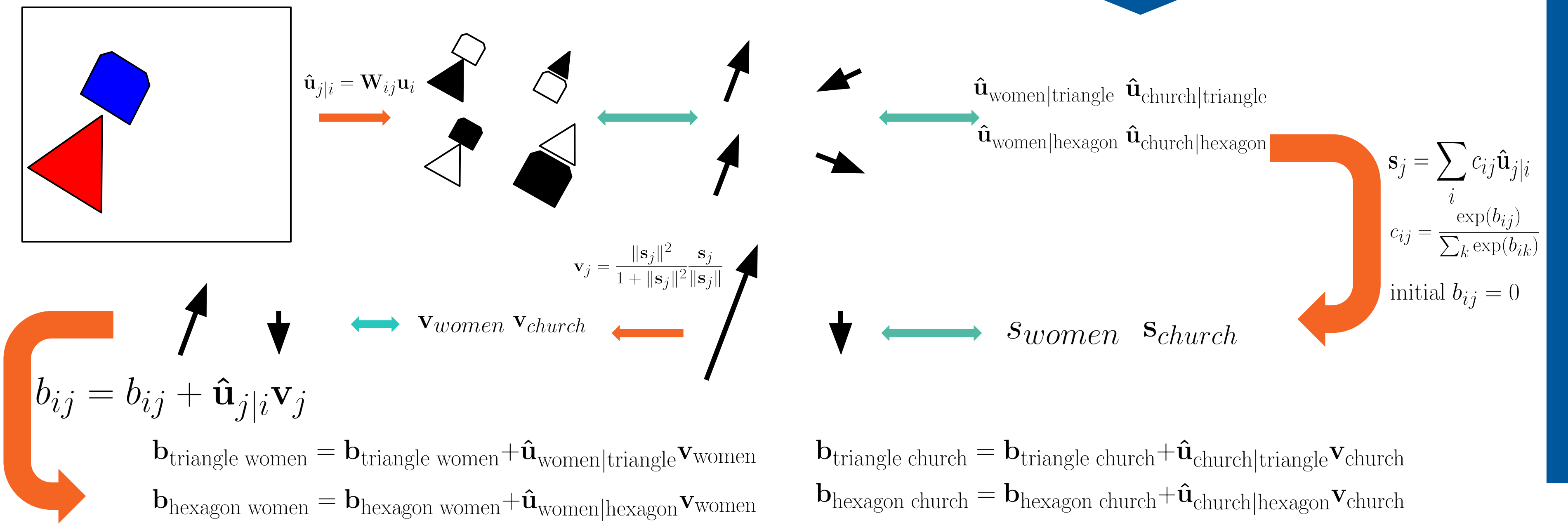
Prediction for women

Output of convolutional layer

Activity vector of hexagon capsule
Activity vector of triangle capsule



Routing by agreement



Routing by agreement

Total input of capsule j

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

Coupling coefficients can’t be calculated in first iteration step

Output of capsule j

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

Routing by agreement

$$c_{ij} = \frac{\exp(b_{ij})}{\sum_k \exp(b_{ik})}$$

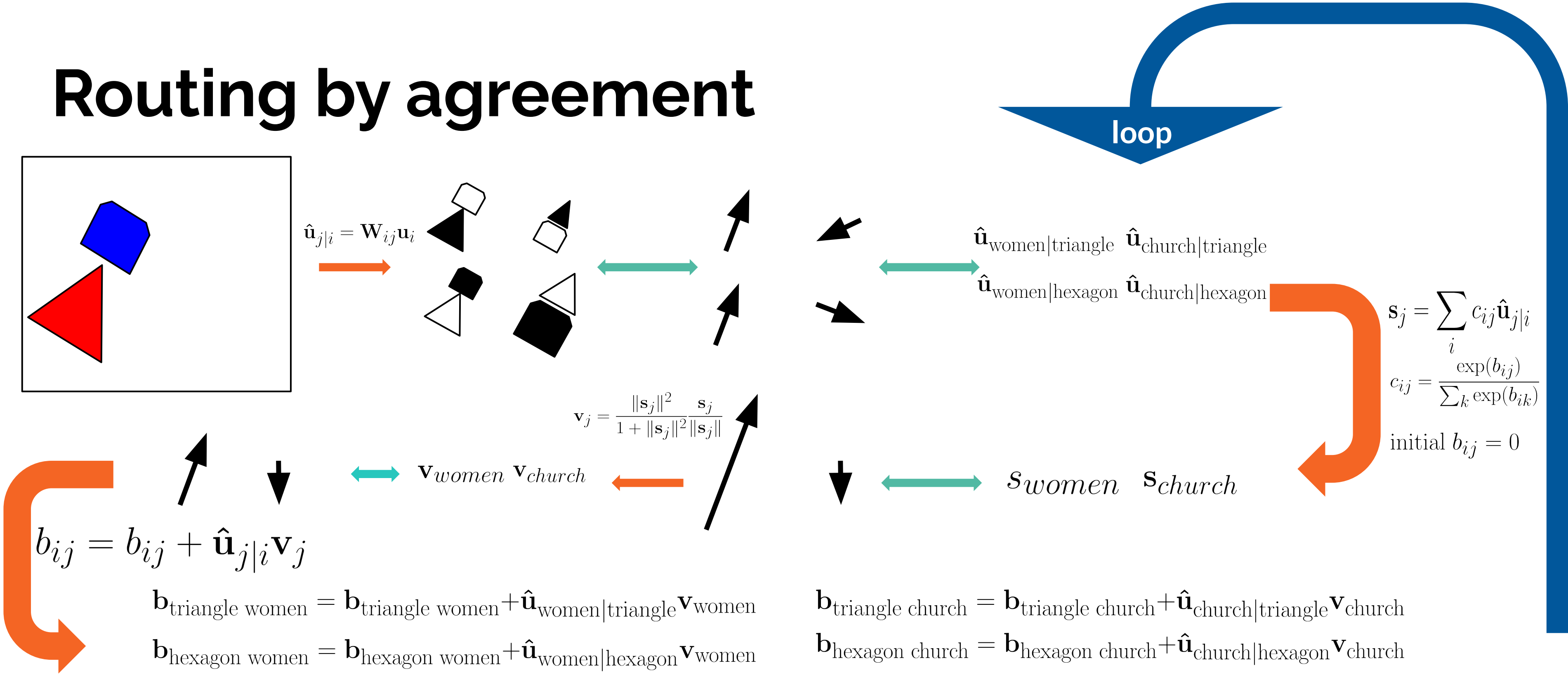
log prior probabilities that capsule i should be coupled to capsule j

$$b_{ij} = 0$$

Since in the first step no information can gained from agreements, it is sensible to assume, that each lower capsule has the same probability to be assigned to every higher capsule

$$\longrightarrow \quad c_{ij} = \frac{1}{\text{number of capsules in next layer}}$$

Routing by agreement



Routing by agreement

log prior probabilities that capsule i should be coupled to capsule j

$$b_{ij} = b_{ij} + \boxed{\hat{\mathbf{u}}_{j|i}\mathbf{v}_j}$$

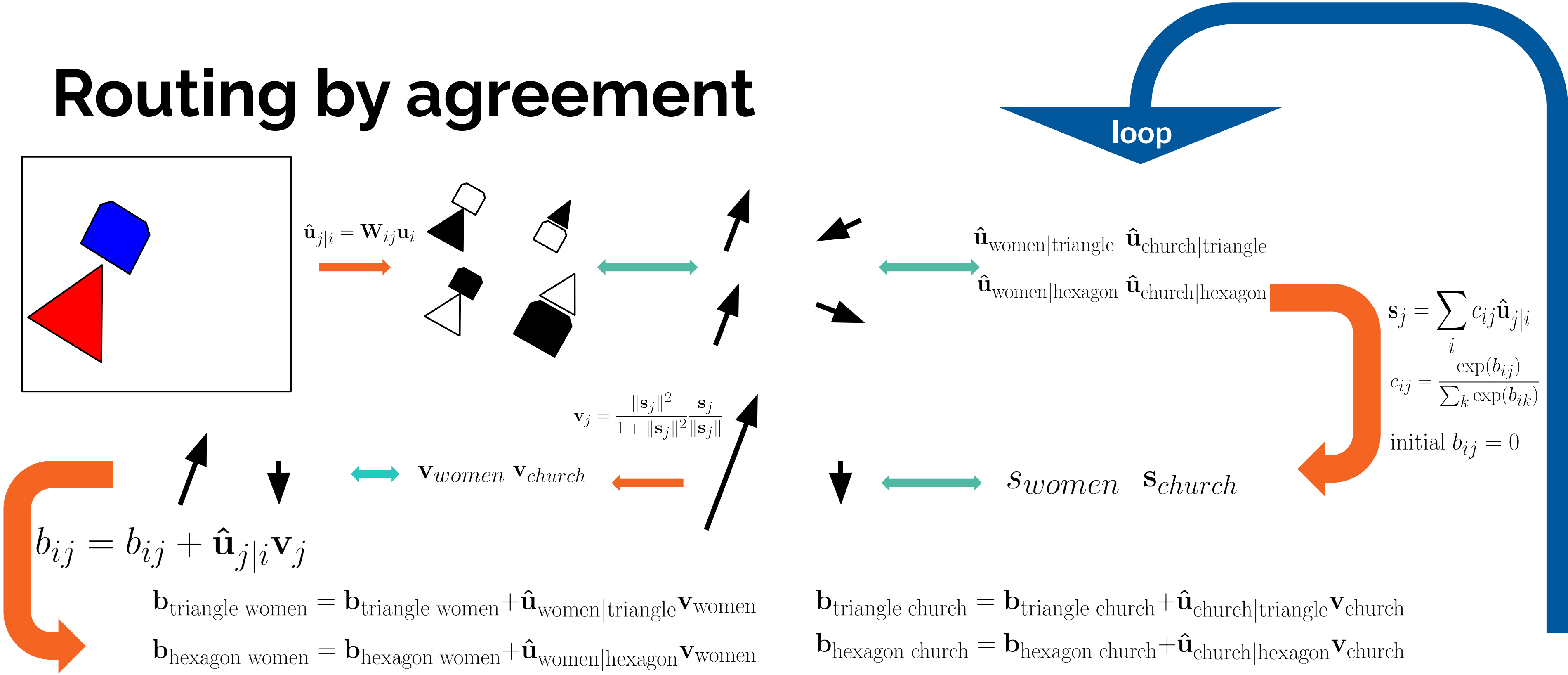
Agreement between capsule i and j is the scalar product of the prediction from capsule i about the output from capsule j and the output of capsule j

coupling coefficients

$$c_{ij} = \frac{\exp(b_{ij})}{\sum_k \exp(b_{ik})}$$

If the agreement between capsule i and capsule j is relatively strong, the coupling coefficient will be greater

Routing by agreement



Routing by agreement

- In our case the **coupling coefficients** between the lower capsules ...

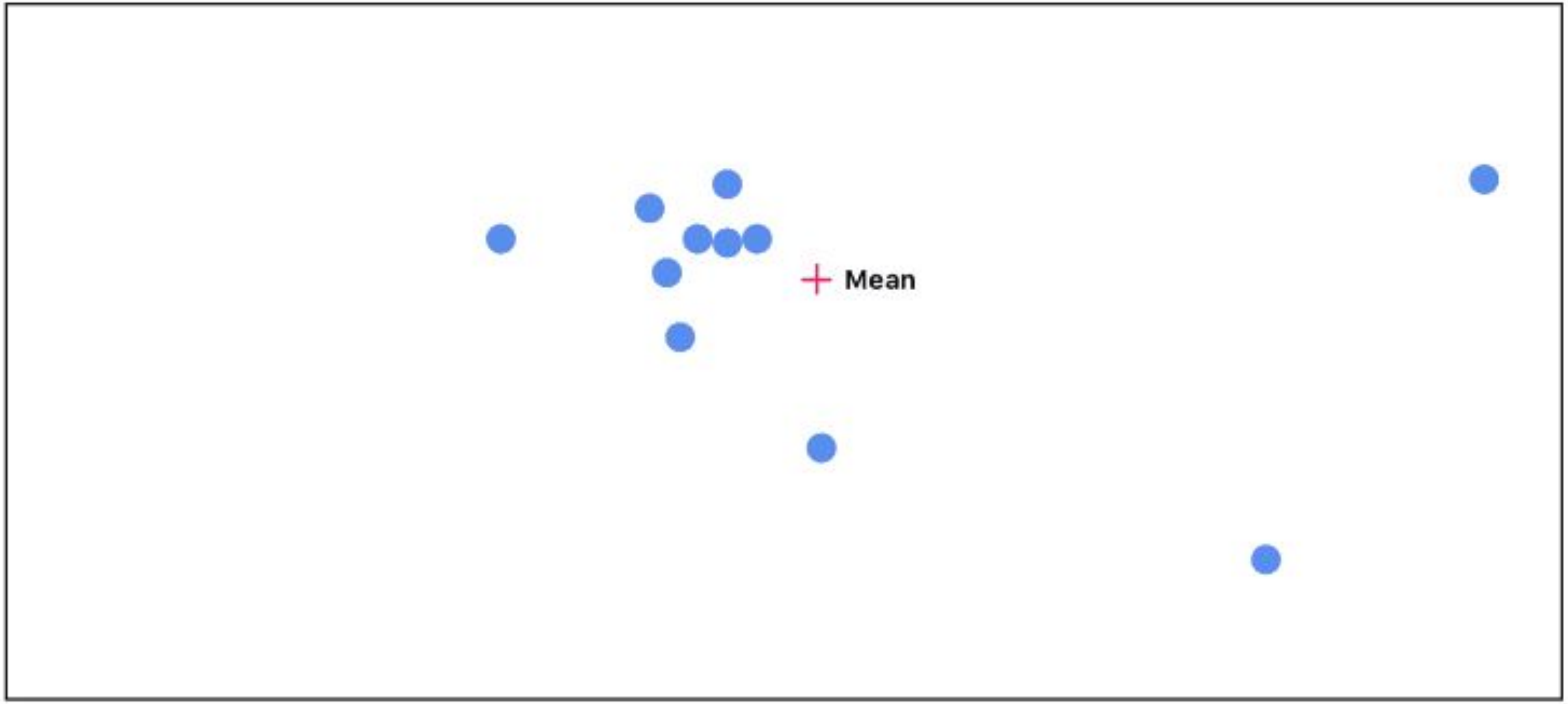
    ... and the church capsule decrease

    ... and the women capsule increase

  with each iteration step

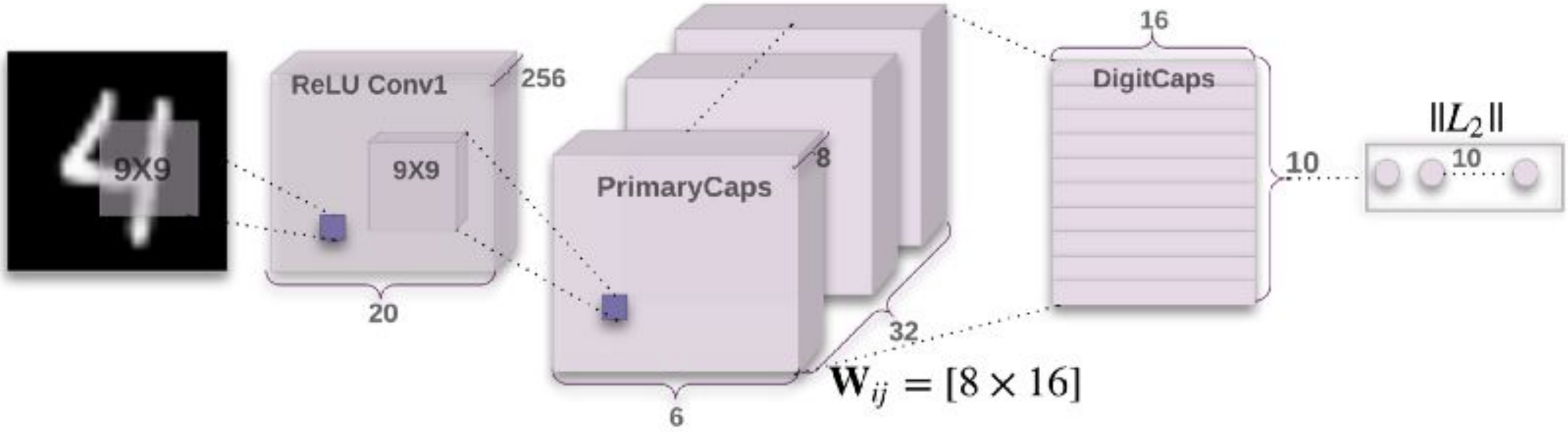
- After a sufficient number of steps, the **coupling coefficients** between the lower capsules and the ...
    - ... church capsule vanishes
    - ... women converges to 1

-> Winner takes it all


Mean

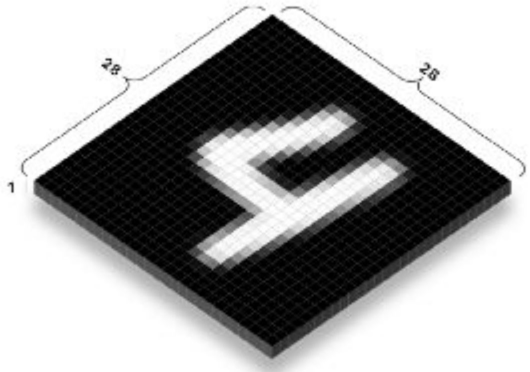
CapsNet Architecture

CapsNet

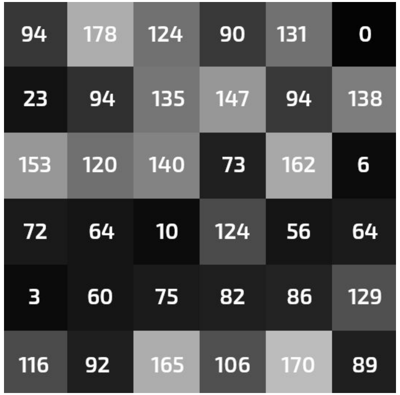


[Source]

The Input



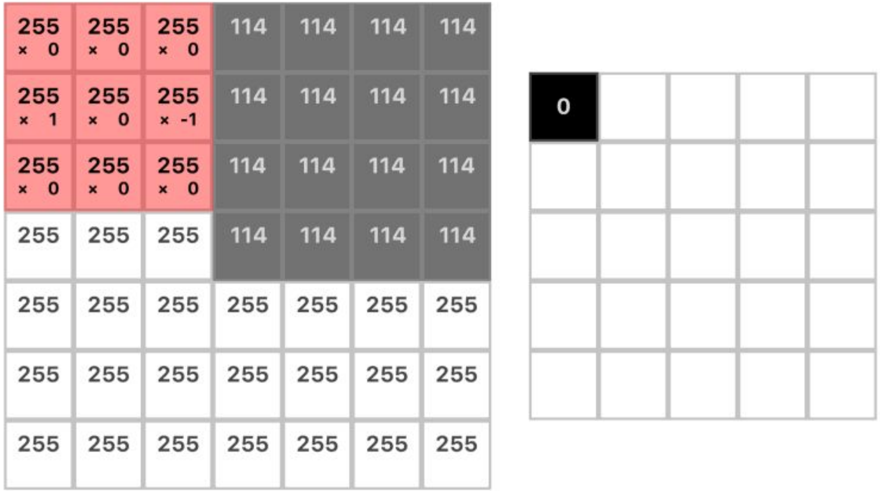
[28x28x1]



[0-255]

[Source]

Convolutions

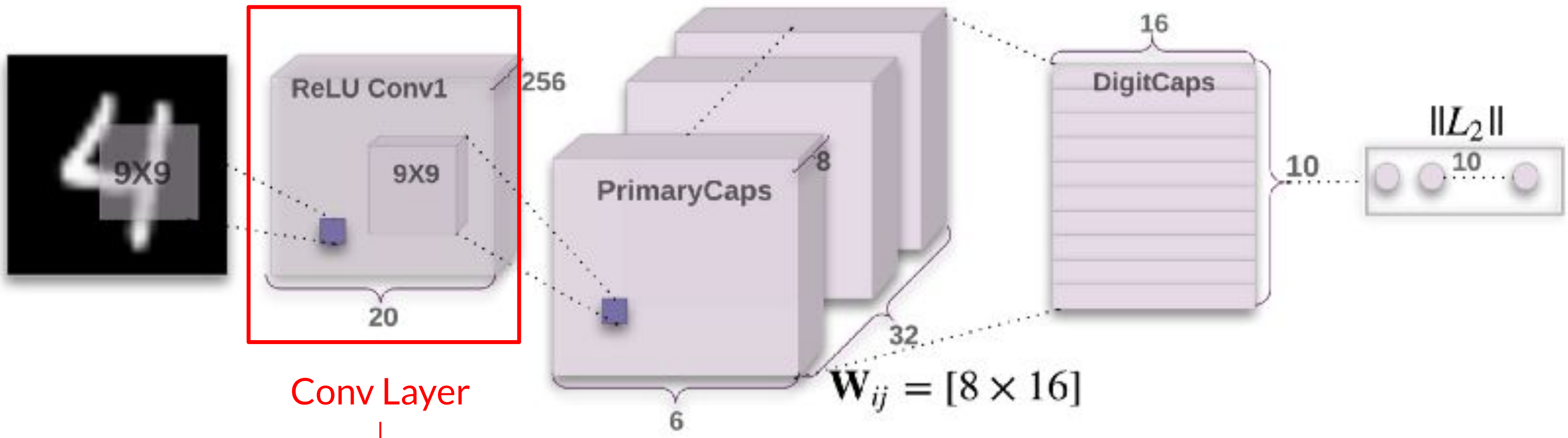


[(**input_size** - **kernel_size** + 2 * **padding**) / **stride**] +1

[(28 - 9 + 2 * 0) / 1] +1 = **20**

[Source]

CapsNet-Convolution Layer

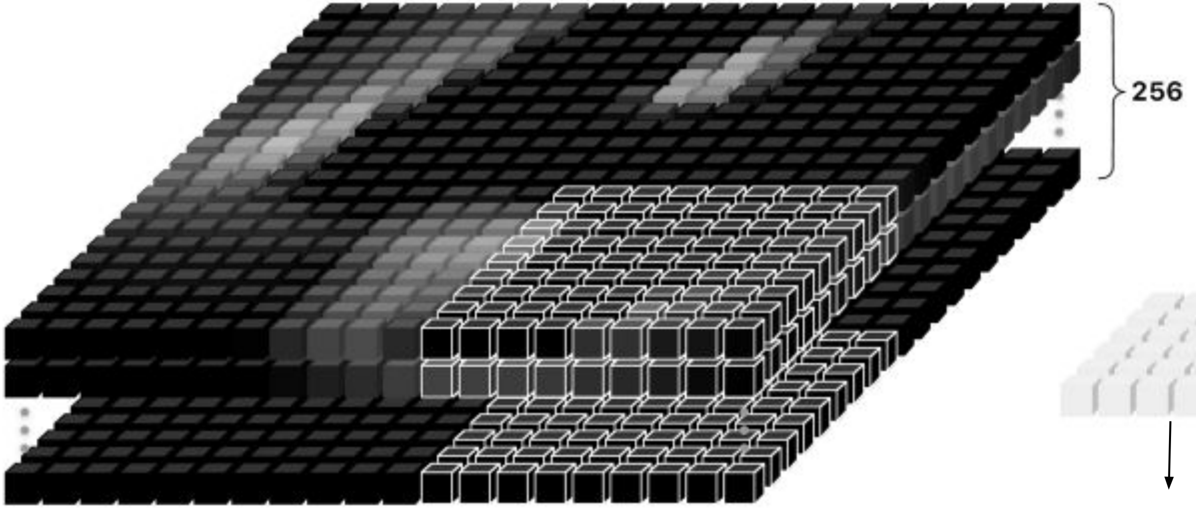


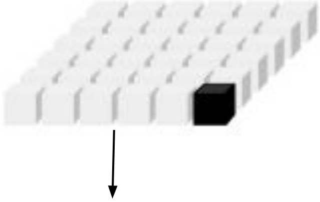
Conv Layer

Input: **28x28x1**
Output: **20x20x256**

[Source]

PrimaryCaps

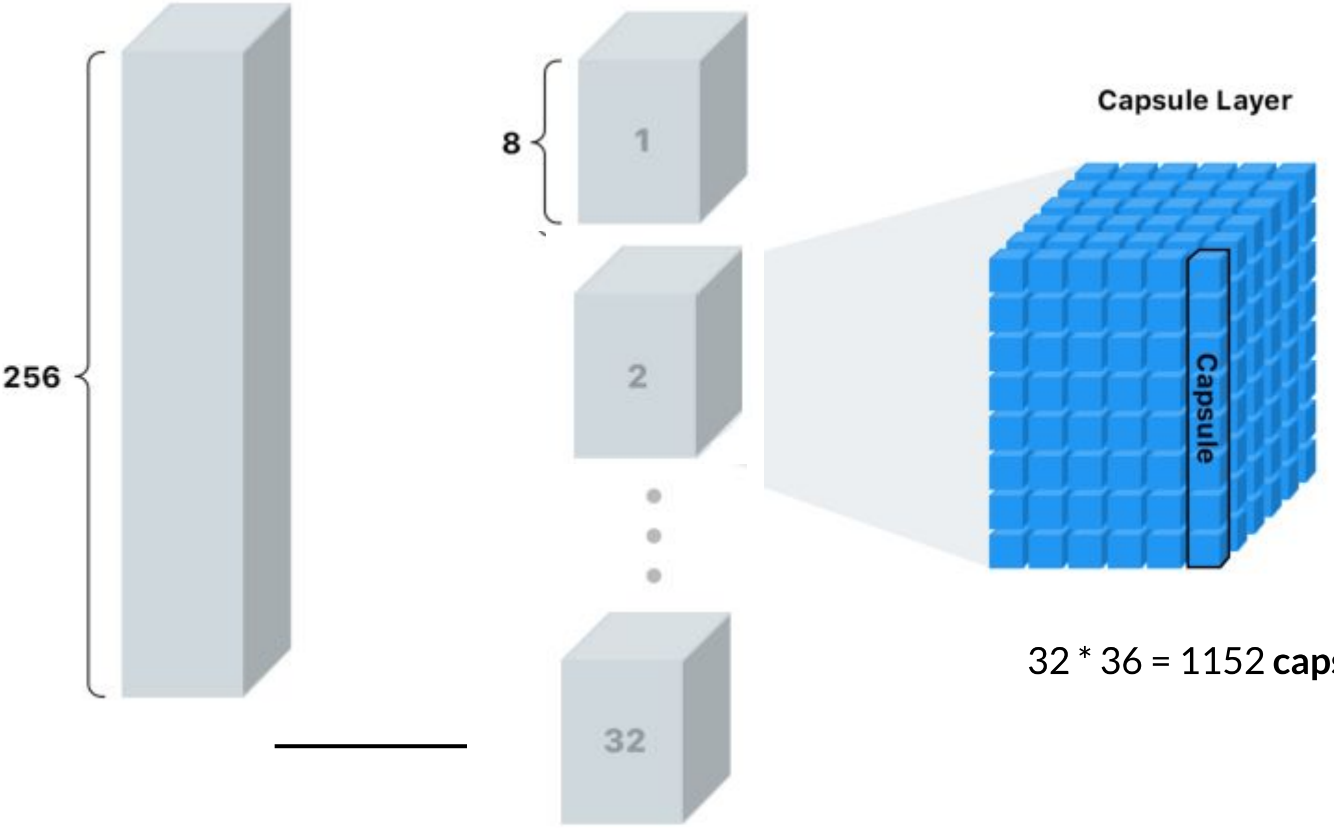




6x6x256

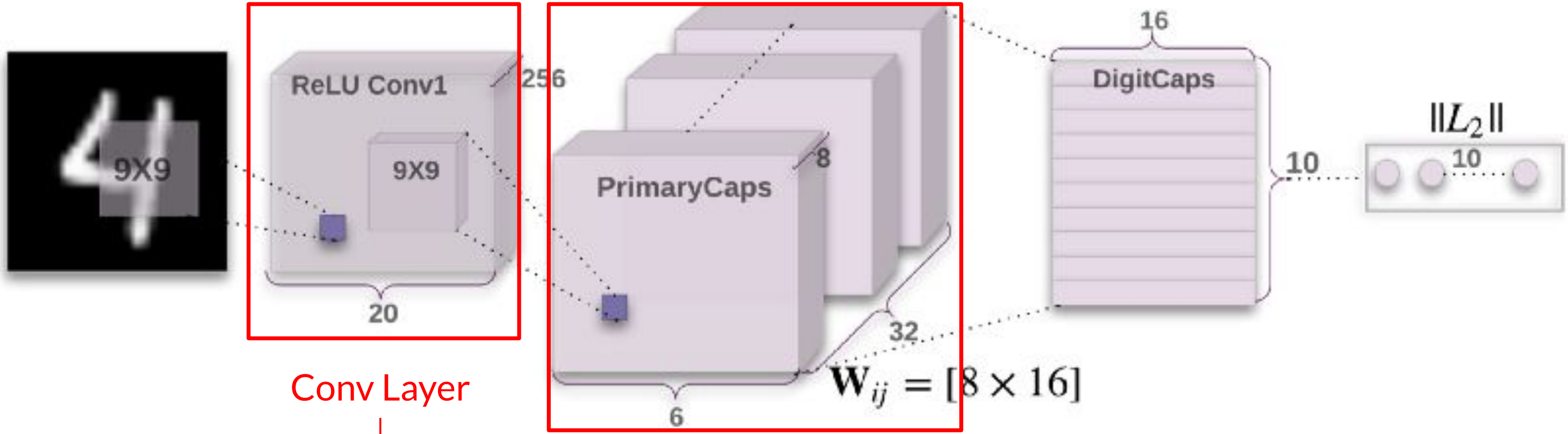
[(20 - 9 + 2 * 0) / 2] +1 = **6.5**

PrimaryCaps cont'd



32 * 36 = 1152 **capsules**

CapsNet-PrimaryCaps Layer



Conv Layer

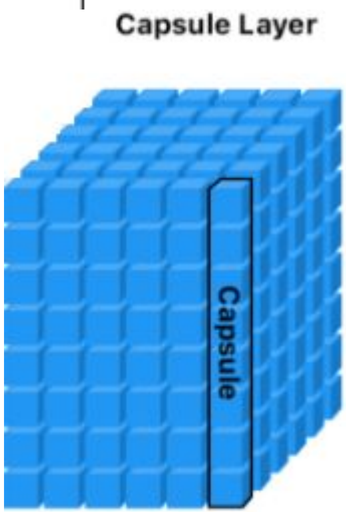
Input: **28x28x1**
Output: **20x20x256**

Primary Caps Layer

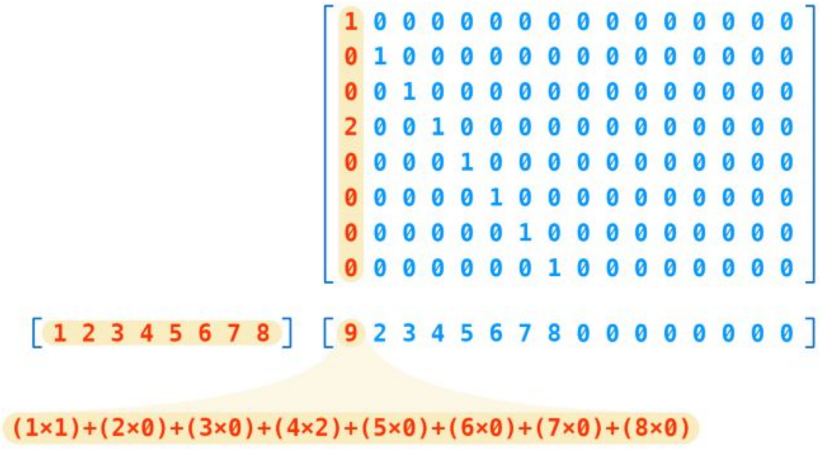
Input: **20x20x256**
Output: **6x6x8x32**

[Source]

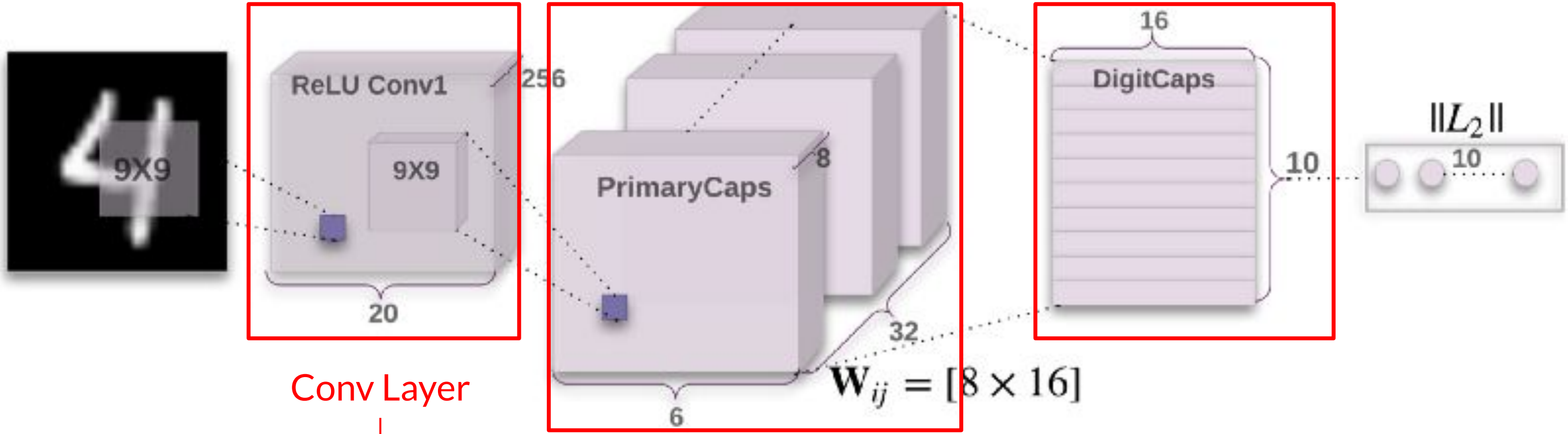
DigitCaps



1152 capsules * 10 classes = 11.520 predictions in total



CapsNet-DigitCaps Layer



Conv Layer

Input: **28x28x1**
Output: **20x20x256**

Primary Caps Layer

Input: **20x20x256**
Output: **6x6x8x32**

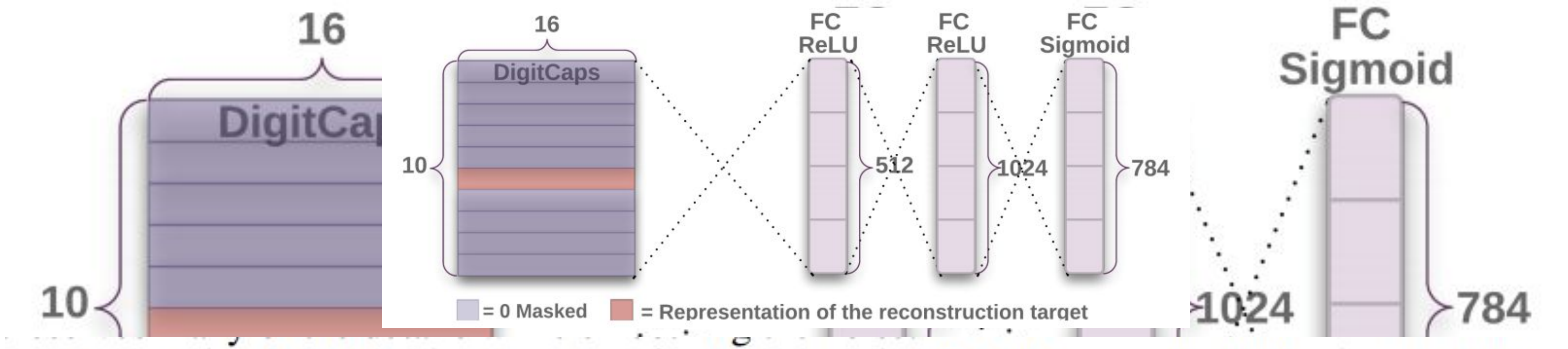
Digit Caps Layer

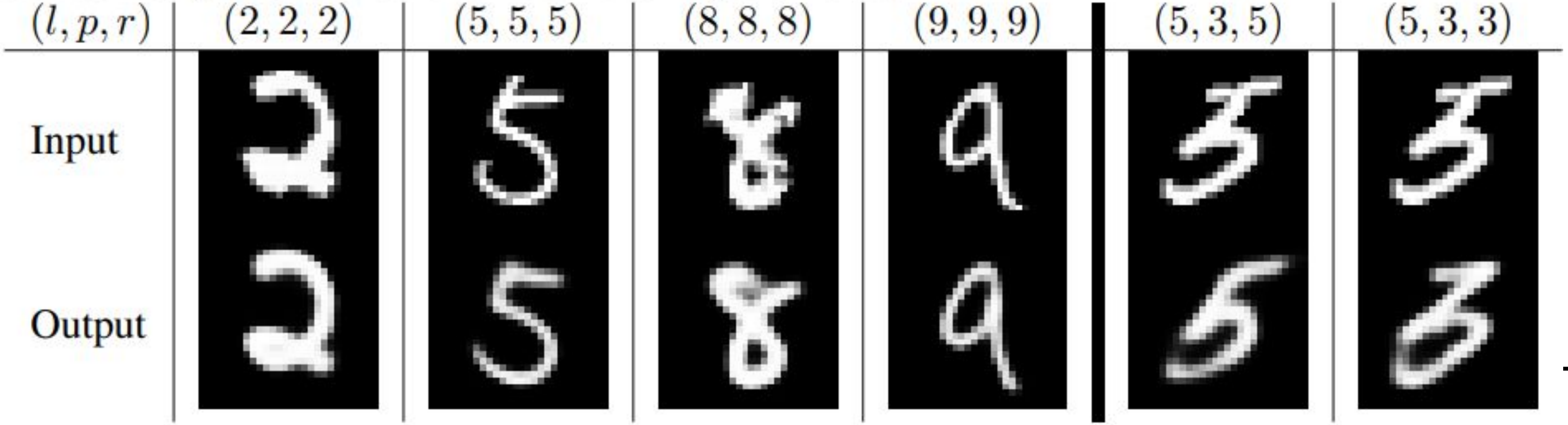
Input: **6x6x8x32**
Output: **10x16**

[Source]

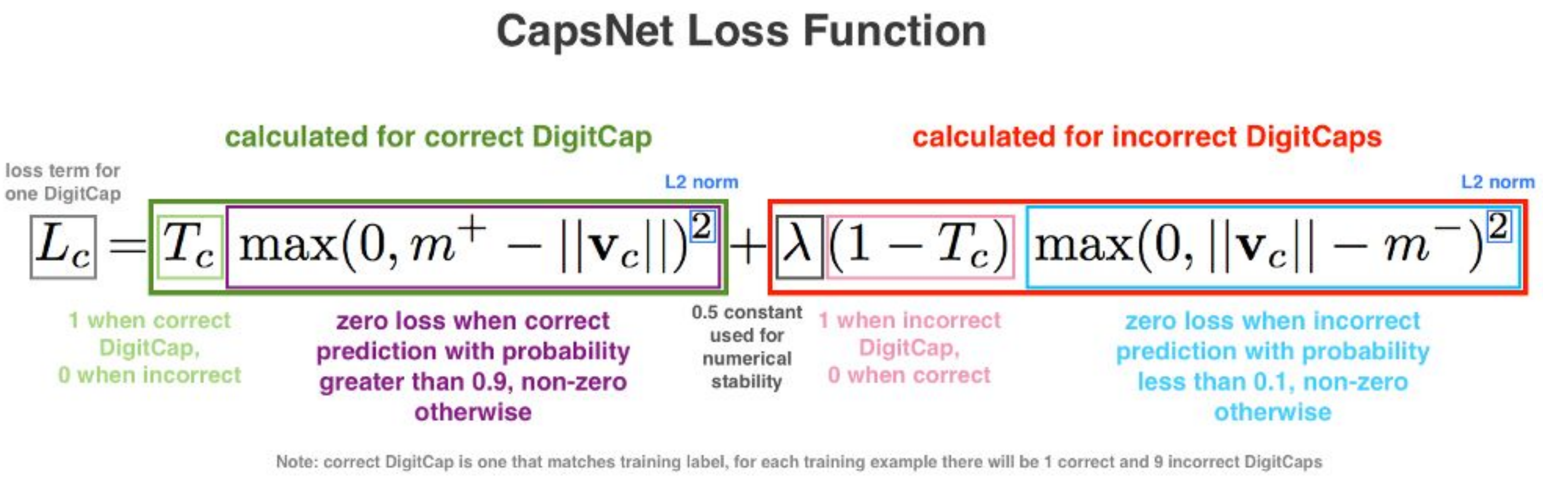
CapsNet-Predictions

Reconstruction

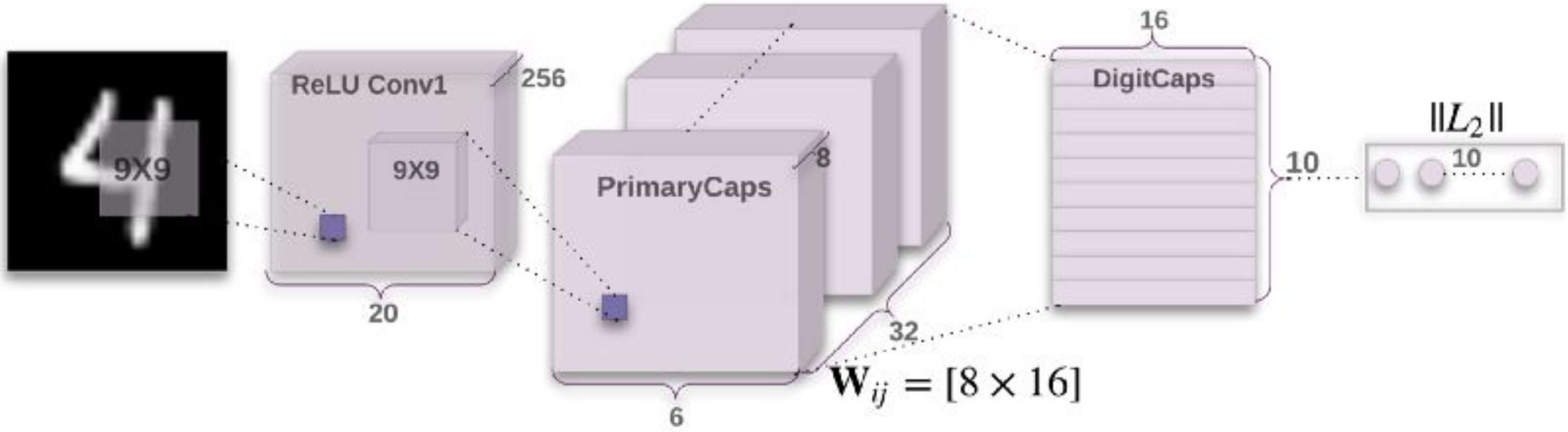




CapsNet-Loss function



CapsNet [reviewed]



[Source]

References

- **Original paper**
- **BlogPost#1**
- **BlogPost#2**
- **BlogPost#3**
- **Continuation of original paper**

Additional Resources

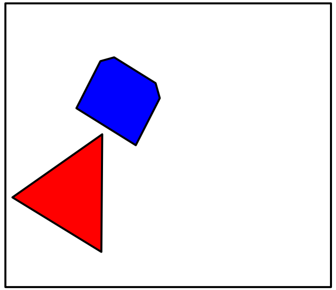
- BlogPost#4
- BlogPost#5
- BlogPost#6
- BlogPost#7
- https://www.youtube.com/watch?v=pPN8d0E3900&t=763s
- Zheng et al.

Q&A

- What would you say about training time of CapsNets?

Routing by agreement

loop



Calculate the predictions of lower-lever parts about the pose and presence of the higher-level parts

Update coupling coefficients

Calculate the activity vector of the higher-level parts based on the coupling coefficients and predictions of lower parts

Squash the output of higher parts

Calculate agreement between lower and higher capsules/parts

Calculate log prior probabilities that lower capsule i should be coupled with higher capsule j based on agreements

Routing by agreement

Procedure 1 Routing algorithm.

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

Equations Pool

$$T = \begin{pmatrix} \lambda\cos\theta & \sin\theta \\ -\sin\theta & \lambda\cos\theta \end{pmatrix}$$

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

$$c_{ij} = \frac{\exp(b_{ij})}{\sum_k \exp(b_{ik})}$$

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

$$\lambda$$

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

$$\hat{\mathbf{u}}_{i|j} \cdot \mathbf{v}_j$$