MATRIX CAPSULES WITH EM ROUTING

hc3040, jak2294, st3186

Columbia University

Overview

1. The Model
   - What is a capsule ?
   - Using EM for Routing-by-agreement
   - Global Architecture
2. Experiments
   - Results on SmallNORB Dataset
   - Novel Viewpoints
   - Adverserial Robustness

Capsule

A capsule network has several layers of capsules
A capsule i in layer L contains :

- M_i = 4x4 "pose" matrix
- $a_i \in [0, 1]$ an activation probability

In between each capsule i in layer L and j in layer $L + 1$ there is a 4x4 trainable transformation matrix W_{ij}

Capsule i can cast a **vote** $V_{ij} = M_i W_{ij}$ for the pose matrix M_j

We calculate the poses and activations of $L + 1$ with an **Expectation-Maximisation** procedure (**EM**)
We do this **DYNAMICALLY** at each **every single forward pass** of the network !
This replaces pooling in a normal CNN.

EM Routing

EM = Expectancy Maximisation

Notations :

Ω_L is the layer L
i and j represent capsules
R_{ij} is the amount of data coming from i that is assigned to j
a is the activation value
$V_{ij} = M_i W_{ij}$ is vote value of i on j

procedure EM ROUTING(a, V):

$\forall i \in \Omega_L, \forall j \in \Omega_{L+1} : R_{ij} \leftarrow \frac{1}{|\Omega_{L+1}|}$
FOR t iterations do : {
$\forall j \in \Omega_{L+1}$ **M step**(a, R, V, j)
$\forall i \in \Omega_L$ **E step**(μ, σ, a, V, i) }
RETURN **(a,M)**

E-step

E-step $=$ Calculate expectation with respect to the latent variables, for the previous value of the parameters

Notations

μ_j^h and σ_j^h are the parameters of Gaussian j along dimension h
a_i is the activation of capsule i
$V_{ij} = M_i W_{ij}$ with M and W being 4x4. So $V_{ij} \in \mathbb{R}^4$
V_{ij}^h is the h th dimension of the vote of capsule i on j

procedure **E step**(μ, σ, a, V, i):

$$\forall j \in \Omega_{L+1} : p_j \leftarrow \frac{1}{\sqrt{\Pi_h^H 2\pi(\sigma_j^h)^2}} exp(-\sum_h^H \frac{(V_{ij}-\mu_j^h)^2}{2(\sigma_j^h)^2})$$

$$\forall j \in \Omega_{L+1} : R_{ij} \leftarrow \frac{a_j p_j}{\sum_{k \in \Omega_{L+1}} a_k p_k}$$

M-step

M-step = Maximize this expectation

Notations

μ_j^h and σ_j^h are the parameters of Gaussian j along dimension h
a_i is the activation of capsule i
V_{ij}^h is the h th dimension of the vote of capsule i on j
β_a, β_u are learned discriminatively and the inverse temperature λ increases at each iteration with a fixed schedule.

procedure **M step**(a, R, V, j) :

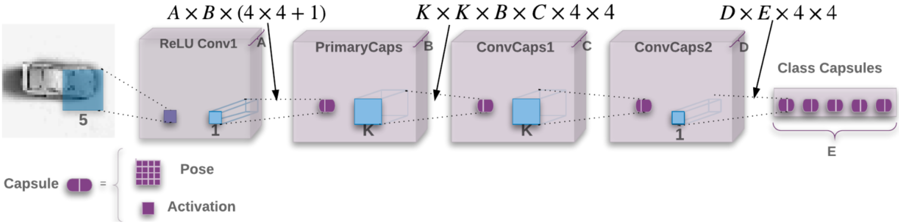
$\forall i \in \Omega_L \,:\, R_{ij} \leftarrow a_i$

$\forall h : \mu_j^h \leftarrow \frac{\sum_i R_{ij} V_{ij}^h}{\sum_i R_{ij}}$ and $(\sigma_j^h)^2 \leftarrow \frac{\sum_i R_{ij}(V_{ij} - \mu_j^h)^2}{\sum_i R_{ij}}$

$cost^h \leftarrow (\beta_u + log(\sigma_j^h)) \sum_i R_{ij}$

$ai \leftarrow logistic(\lambda(\beta_a - \sum_h cost^h))$

Architecture



A network with one ReLU convolutional layer followed by a primary convolutional capsule layer and two more convolutional capsule layers.

Loss function

Spread Loss

We use a margin m,
The loss for a the target class a_t :
$L = \sum_{i \neq t} L_i$
$L_i = max(0, m - (a_t - a_i))^2$

It is like a multi-dimensional hinge loss.
"By starting with a small margin of 0,2 and linearly increasing it during training to 0,9, we avoid dead capsules in the earlier layers"

Experiments

Dataset: SmallNORB

- Five classes of toys photographed from different viewpoints
- Each toy pictured at 18 azimuths, 9 elevations, 6 lighting conditions
- 5 instances of each toy class for training/testing
- Each instance 24,300 stereo pairs of 96x96 images

Why this dataset?

- Natural images in different forms
- Intuition behind capsules

Data Processing

- Downsampled each snallNORB image to 48x48 pixels
- Normalized each image (0 mean,unit variance)
- Took 32x32 patches and added random brightness and contrast during training
- Applied model to 32x32 patch cropped from center of image for testing

Results

Table 1: The effect of varying different components of our capsules architecture on smallNORB.

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%**

Results Cont...

- Best model variation: 1.8% test error (1.4% if averaged class activations over multiple crops)
- Previous best reported on smallNORB: 2.56% (Ciresanetal.(2011))
- Achieved 2.6% on NORB dataset (snallNORB with added background), compared to best reported 2.7% (Ciresan et al. (2012))

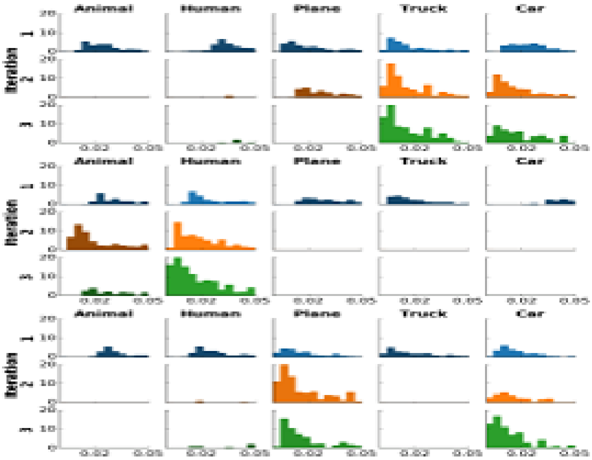
Baseline Model

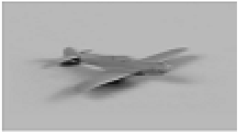
CNN (result of optimized hyperparameter search)

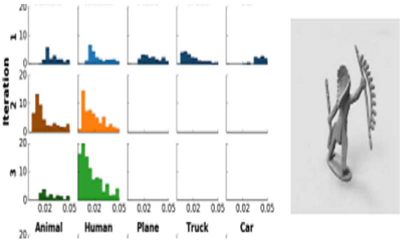
- 2 conv. layers with 32/64 channels
- Both with kernel size of 5, stride of 1, 2x2 max pooling
- 3rd layer: 1024 unit fully connected layer with dropout
- *Hidden units: ReLU non-linearity
- 5 class softmax output layer

Result: 5.2% test error, 4.2M params (compare to 310k of Capsule network)

EM-Iteration Visual







Each iteration, vote assignments are adjusted to identify clusters of votes. Histogram shows distances of votes to each class mean (distances are weighted by assignment probability)

- Iteration 1: votes are equally distributed between classes
- Iteration 2: assignment probability for agreeing votes increases, majority of votes assigned to detected clusters

Testing Novel Viewpoints

Isolating performance on novel viewpoints from performance on all viewpoints:

- Use limited set of viewpoints for training and test on a much wider range
- Match error rate between Capsule model and baseline CNN model, then compare on testing set

Testing Novel Viewpoints Cont...

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%

Testing Robustness

Adverserial Attack Testing

Slightly changing inputs to neural network to confuse network into making incorrect classifications; designed to exploit model vulnerability.

Notable Adverserial Attack Method: FGSM (Goodfellowetal.(2014))

Testing Robustness

FGSM

Computes gradient of loss with respect to each pixel's intensity and adjusts pixel intensity by fixed amount in direction that increases the loss; only includes one hyper parameter.

Basic Iterative Method (Kurakin et al (2016))

Similar to FGSM method but multiple small steps as opposed to one step.

Robustness Cont...

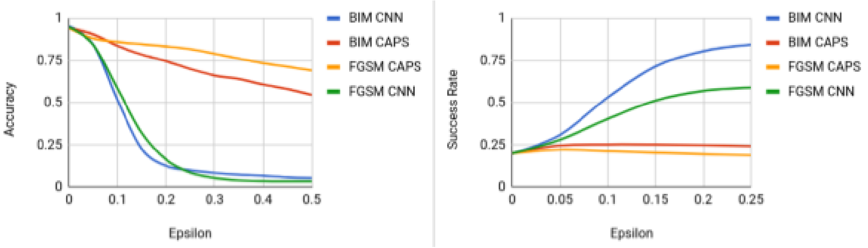


Figure 3: Accuracy against ϵ after an adversarial attack (left) and Success Rate after a targeted adversarial attack (right). The targeted attack results were evaluated by averaging the success rate after the attack for each of the 5 possible classes.

Conclusions

1. Ability to classify regardless of configuration/viewpoint
2. Effective generalization to novel viewpoints
3. Better defense against white-box adverserial attacks

Drawback: Time for EM dynamic routing layer to layer; not efficient for GPU parallelization.

References

G.Hinton et al (2012)
Matrix Capsules with EM Routing
ICLR 2018

The End