Object Classification using Capsule Network

Dhanaprakaash, Jayabrata, Jayateja, Joji

Indian Institute of Science

June 10, 2020

Table of Contents

1 Demo

2 Introduction

3 Capsule Networks

4 Implementation
- Handwritten Digit Recognizer
- Image Recognizer
- Web App

5 Results
- MNIST
- CIFAR 10

Table of Contents

1 Demo

2 Introduction

3 Capsule Networks

4 Implementation
- Handwritten Digit Recognizer
- Image Recognizer
- Web App

5 Results
- MNIST
- CIFAR 10

Demo
Introduction
Capsule Networks
Implementation
Results
Acknowledgement
Contributions
References

Demo

https://capsule-networks.herokuapp.com

Table of Contents

1 Demo

2 Introduction

3 Capsule Networks

4 Implementation
- Handwritten Digit Recognizer
- Image Recognizer
- Web App

5 Results
- MNIST
- CIFAR 10

Introduction

Demo
Introduction
Capsule Networks
Implementation
Results
Acknowledgement
Contributions
References

Capsule networks are an improvement on Convolutional Neural Networks (CNN). CNNs are now widely used in image recognition. Steps for CNN are as follows:

1. Given an input image, a set of filters scan it and perform convolution operation.
2. This creates a feature map inside the network. These features will next pass via activation (ex. ReLU) and pooling layers. Activation gives non-linearity. Pooling helps in reducing the training time. Pooling make summaries of each sub-region.
3. At the end, it will pass via a sigmoid classifier.

Introduction

Demo
Introduction
Capsule Networks
Implementation
Results
Acknowledgement
Contributions
References

Limitations of CNN

- Inability to recognize pose, texture and deformations of an image or parts of the image
- The pooling operation in CNN loses some features in the image.
- They therefore require lots of training data in order to compensate for this loss.
- CNNs are more prone to adversarial attacks such as pixel perturbations resulting in wrong classifications.

Introduction

Demo
Introduction
Capsule Networks
Implementation
Results
Acknowledgement
Contributions
References

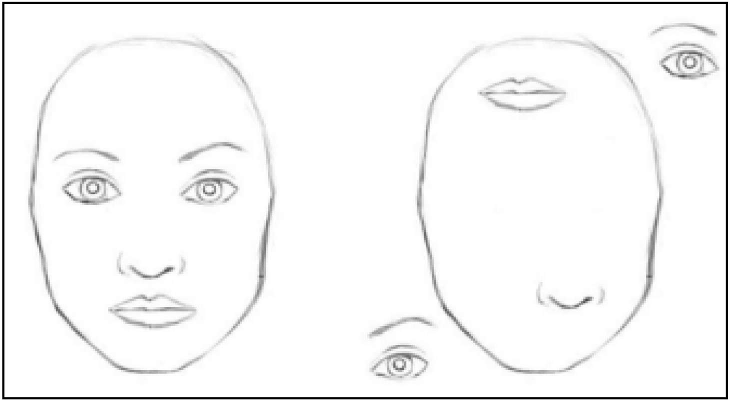

Figure: CNN will classify right image also as a face

Table of Contents

1 Demo

2 Introduction

3 Capsule Networks

4 Implementation
- Handwritten Digit Recognizer
- Image Recognizer
- Web App

5 Results
- MNIST
- CIFAR 10

Demo
Introduction
Capsule Networks
Implementation
Results
Acknowledgement
Contributions
References

Capsule Networks

- Hinton and his colleagues proposed Capsule Networks as an alternative to CNNs.
- In a capsule network, each capsule is made up of a group of neurons with each neuron's output representing a different property of the same feature.
- This provides the advantage of recognizing the whole entity by first recognizing its parts.
- Activity vector represents the instantiation parameters of a specific type of entity such as an object.
- Capsnet replace scalar-output feature detectors with vector-output capsules and max-pooling with routing-by-agreement mechanism.

Capsule Networks - Squashing

- We use **squashing** function to make length of output vector represent the probability of entity represented by that vector.

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

- where $\mathbf{v}_j$ is the vector output of capsule j and $\mathbf{s}_j$ is its total input.

Capsule Networks - Dynamic Routing

Demo
Introduction
Capsule Networks
Implementation
Results
Acknowledgement
Contributions
References

- For all but the first layer of capsules, the total input to a capsule $\mathbf{s}_j$ is a weighted sum over all "prediction vectors" $\hat{\mathbf{u}}_{j|i}$ from the capsules in the layer below and is produced by multiplying the output $\mathbf{u}_i$ of a capsule in the layer below by a weight matrix $\mathbf{W}_{ij}$

$$\mathbf{s}_j = \sum_i c_{ij} \hat{\mathbf{u}}_{j|i} \ , \qquad \hat{\mathbf{u}}_{j|i} = \mathbf{W}_{ij} \mathbf{u}_i \tag{2}$$

- where the c_{ij} are coupling coefficients that are determined by the iterative dynamic routing process.

Capsule Networks - Dynamic Routing Between Capsules

Algorithm 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: $\mathbf{c}_i \leftarrow \mathtt{softmax}(\mathbf{b}_i)$
        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 $(l+1)$: $\mathbf{v}_j \leftarrow \mathtt{squash}(\mathbf{s}_j)$
        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 $\mathbf{v}_j$

Dynamic Routing between Capsules

Demo
Introduction
Capsule Networks
Implementation
Results
Acknowledgement
Contributions
References

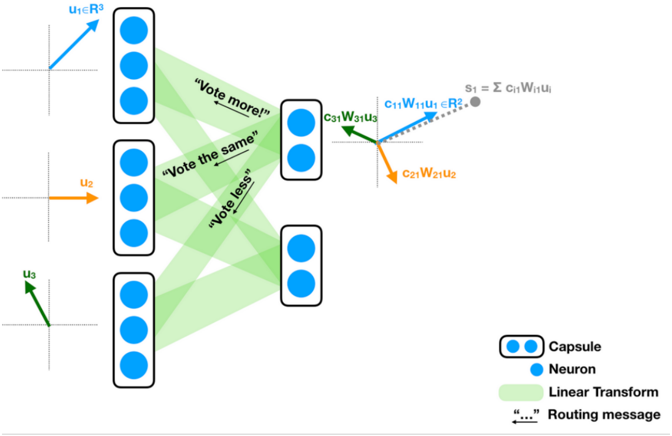

Figure: Dynamic Routing between Capsules

Capsule Networks - Margin Loss

- The margin loss is given by

$$L_k = T_k \ \max(0, m^+ - ||\mathbf{v}_k||)^2 + \lambda \ (1 - T_k) \ \max(0, ||\mathbf{v}_k|| - m^-)^2 \quad (3)$$

- where $T_k = 1$ iff a digit of class k is present and $m^+ = 0.9$, $m^- = 0.1$ and λ is the down-weighting constant.
- The total loss is simply the sum of the losses of all digit capsules.

Capsule Networks - Pros

Demo
Introduction
Capsule Networks
Implementation
Results
Acknowledgement
Contributions
References

1. Reaches high accuracy (state-of-the-art) on MNIST, and promising on CIFAR-10.
2. Requires less training data and more resistant against adversial attacks.
3. Position and pose information are preserved. (property of equivariance)
4. Capsnets proved to be effective when data is class imbalanced.
5. This is promising for image segmentation and object detection.
6. Capsule activations nicely map the hierarchy of parts.
7. Activation Vectors are easier to interpret.

Capsule Networks - Cons

Demo
Introduction
Capsule Networks
Implementation
Results
Acknowledgement
Contributions
References

1. Not the state of the art on CIFAR10 (Can be improved with modification).
2. Slow training, due to the inner loop. (Routing by agreement algorithm)(Can be improved by using EM algorithm)
3. Suffers from the problem of crowding (inability to recognize two or more objects if they are very closely spaced or overlapping to each other).

Demo
Introduction
Capsule Networks
Implementation
Results
Acknowledgement
Contributions
References

Capsule Networks - Applications

1. The Architecture of the capsule network has a property of rotational invariance and spatial awareness. In the field of astronomy, the capsule network is deployed to understand the morphological types of galaxies.
2. Protein family structure classification is another area where CapsNets have been applied.
3. Autonomous cars will benefit hugely from computer vision applications such as CapsNets that can be used for predicting traffic speed.
4. In medical imaging, It is used extensively in applications like lung cancer detection and brain tumour detection.

Table of Contents

1 Demo

2 Introduction

3 Capsule Networks

4 Implementation
- Handwritten Digit Recognizer
- Image Recognizer
- Web App

5 Results
- MNIST
- CIFAR 10

Implementation

Demo
Introduction
Capsule Networks
Implementation
Handwritten Digit Recognizer
Image Recognizer
Web App
Results
Acknowledgement
Contributions
References

- We used PyTorch[1] library to implement capsule networks
- We implemented models for MNIST[2] and CIFAR 10 [3] datasets.
- For training we used atleast 20 epochs with a batch size of 100.

Demo
Introduction
Capsule Networks
Implementation
Handwritten Digit Recognizer
Image Recognizer
Web App
Results
Acknowledgement
Contributions
References

Handwritten Digit Recognizer

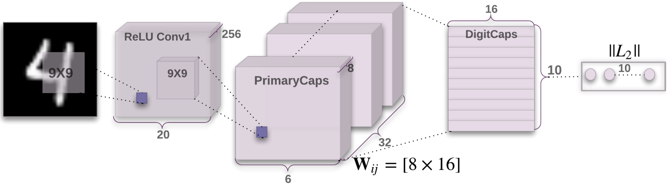

Figure: Architecture of Handwritten Digit Recognizer

Handwritten Digit Recognizer - Decoder

Demo
Introduction
Capsule Networks
Implementation
Handwritten Digit Recognizer
Image Recognizer
Web App
Results
Acknowledgement
Contributions
References

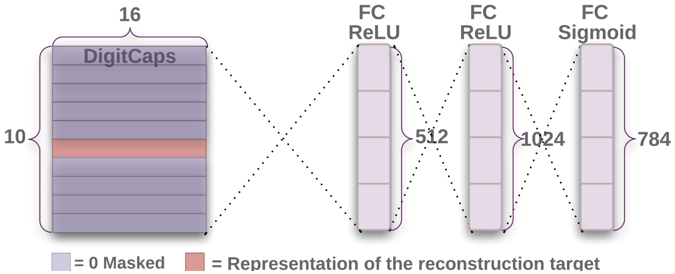

Figure: Decoder Architecure

Image Recognizer

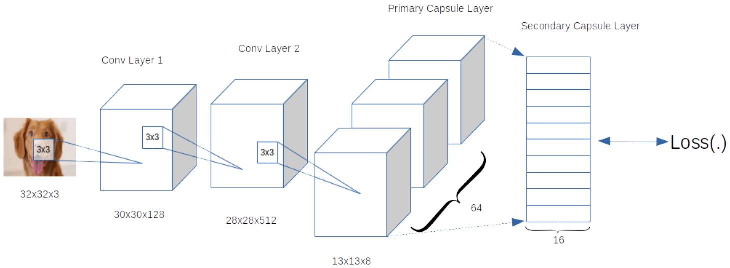

Figure: Architecture of Image Recognizer. Decoder section is not shown. But it is similar to that of handwritten digit recognizer except the size of decoder output is $32 \times 32 \times 3 = 3072$.

Demo
Introduction
Capsule Networks
Implementation
Handwritten Digit Recognizer
Image Recognizer
Web App
Results
Acknowledgement
Contributions
References

Web App

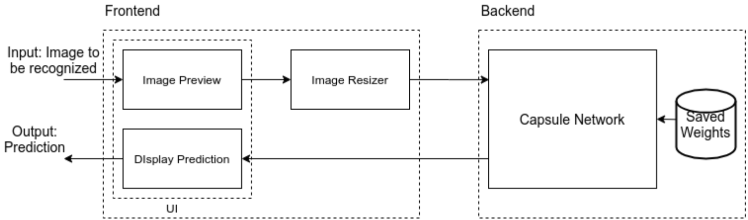

Figure: Architecture of Web Application

Table of Contents

1 Demo

2 Introduction

3 Capsule Networks

4 Implementation
- Handwritten Digit Recognizer
- Image Recognizer
- Web App

5 Results
- MNIST
- CIFAR 10

Results - Overview

Demo
Introduction
Capsule Networks
Implementation
Results
MNIST
CIFAR 10
Acknowledgement
Contributions
References

Dataset	Model Description	Accuracy
MNIST	Implementation of the paper [4], 9x9 kernels	99.44%
CIFAR 10	9x9 kernels	62.02%
	3x3 kernels, 2 Conv layers	76.42%

Table: Some models we implemented and validation accuracy at the end of training

Results - MNIST

Demo
Introduction
Capsule Networks
Implementation
Results
MNIST
CIFAR 10
Acknowledgement
Contributions
References

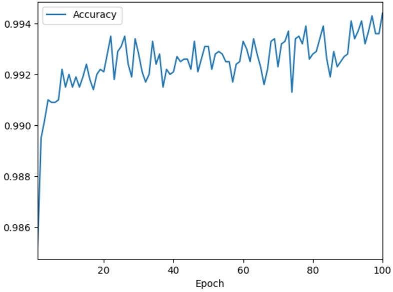

Figure: Validation accuracy vs epochs

Results - MNIST

Figure: The top half shows target images and bottom half shows reconstructed images

Results - MNIST

Figure: Reconstructions from perturbations of dimensions of output vector

Results - CIFAR 10

Demo
Introduction
Capsule Networks
Implementation
Results
MNIST
CIFAR 10
Acknowledgement
Contributions
References

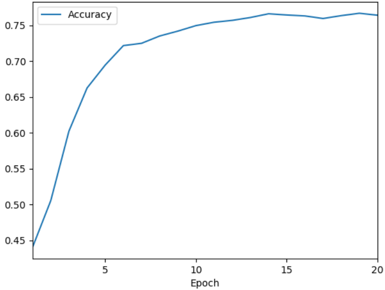

Figure: Validation accuracy vs Epochs

Results - CIFAR 10

Figure: The top half shows target images and bottom half shows reconstructed images

Acknowledgement

Demo
Introduction
Capsule Networks
Implementation
Results
Acknowledgement
Contributions
References

- We adapted the capsule network implementation of Dulat Yerzat. The source code is available at https://github.com/higgsfield/Capsule-Network-Tutorial/blob/master/Capsule%20Network.ipynb.

References

[1] Adam Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach et al. Curran Associates, Inc., 2019, pp. 8026–8037. URL: http://papers.nips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[2] Yann LeCun, Corinna Cortes, and Christopher JC Burges. *The mnist database of handwritten digits*. 1998. URL: http://yann.lecun.com/exdb/mnist/.

[3] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. *The CIFAR-10 dataset*. URL: https://www.cs.toronto.edu/~kriz/cifar.html.

Demo
Introduction
Capsule Networks
Implementation
Results
Acknowledgement
Contributions
References

References

[4] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. *Dynamic Routing Between Capsules*. 2017. arXiv: 1710.09829 [cs.CV].

Thank You