Capsule Networks for Low-Data Transfer Learning

Andrew Gritsevskiy
Mentor: Maksym Korablyov
Mentor: Gil Alterovitz

MIT PRIMES

May 20, 2018
Neural networks
Neural networks

- Universal function approximator
Neural networks

- Universal function approximator
  - Is this a dog?
Neural networks

- Universal function approximator
  - Is this a dog?
Neural networks

- Universal function approximator
  - Is this a dog?
Structure of a (linear) classifier
# Structure of a (linear) classifier

\[ W \begin{bmatrix} 3.5 & 0.2 & 2.1 & 1.6 \\ 0.0 & -0.5 & 0.8 & 0.1 \\ 1.1 & 0.8 & -0.9 & 1.4 \end{bmatrix} + b \begin{bmatrix} 102 \\ 40 \\ 255 \\ 14 \end{bmatrix} = \begin{bmatrix} 925.1 \\ 185.6 \\ -64.7 \end{bmatrix} \]

- **dog**: 925.1
- **cat**: 185.6
- **aardvark**: -64.7
Structure of a (linear) classifier

- Loss function

<p>| | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3.5</td>
<td>0.2</td>
<td>2.1</td>
<td>1.6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>-0.5</td>
<td>0.8</td>
<td>0.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.1</td>
<td>0.8</td>
<td>-0.9</td>
<td>1.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>102</td>
<td>40</td>
<td>255</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

dog: 925.1

cat: 185.6

aardvark: -64.7
Adjusting weights
Adjusting weights

- Method I: Random
  - Accuracy: 15.5%
Adjusting weights

- Method I: Random
  - Accuracy: 15.5%
- Method II: Random local search
  - Accuracy: 21.4%
Adjusting weights

- Method I: Random
  - Accuracy: 15.5%
- Method II: Random local search
  - Accuracy: 21.4%
- Method III: Gradient descent
Gradient descent

```python
while True:
    gradients = calculate_gradient(loss_function, data, weights)
    weights += - step_size * gradients
```
Backpropagation
Capsule networks
Capsule networks

- Neurons store information as vectors
Capsule networks

- Neurons store information as vectors
- Vectors store pose information
Capsule networks

- Neurons store information as vectors
- Vectors store pose information
  - Vector points in direction of object orientation
Capsule networks

- Neurons store information as vectors
- Vectors store pose information
  - Vector points in direction of object orientation
  - Length of vector is probability that object exists
Capsule networks

- Neurons store information as vectors
- Vectors store pose information
  - Vector points in direction of object orientation
  - Length of vector is probability that object exists
Capsule networks

- Neurons store information as vectors
- Vectors store pose information
  - Vector points in direction of object orientation
  - Length of vector is probability that object exists
Capsule networks

- Routing by agreement
Capsule networks

- Routing by agreement
Capsule networks

- Routing by agreement
  - Image segmentation?
Capsule networks

- Routing by agreement
  - Image segmentation!
MultiMNIST
MultiMNIST
MultiMNIST

[Diagram showing the architecture of a capsule network for MultiMNIST, including layers labeled LReLU Conv1, LReLU Conv2, Primary Caps, and Digit Caps, with dimensions and connections between layers indicated.]
MultiMNIST
MultiMNIST

```
Digit Caps 16
One-hot Digit Caps
FC ReLU 512 x10
FC ReLU 1024 x10
FC Sigmoid 1296 x10
Memo 36

LReLU Conv1
LReLU Conv2
Primary Caps
FC Dropout
FC
Memo Digit Caps
```

Andrew Gritsevskiy (MIT PRIMES)  Capsule networks for transfer learning  MIT PRIMES - May 20, 2018
Transfer learning
Transfer learning

- Use a model pre-trained on one dataset to learn another dataset
Transfer learning

- Use a model pre-trained on one dataset to learn another dataset
- subMMNIST dataset
Transfer learning

- Use a model pre-trained on one dataset to learn another dataset
- subMMNIST dataset
  - The MMNIST dataset, but without one digit
Transfer learning

- Use a model pre-trained on one dataset to learn another dataset
- subMMNIST dataset
  - The MMNIST dataset, but without one digit
- The idea:
  - Train on subMMNIST
  - Load full MMNIST dataset
  - See how the network does
The three networks

- Regular convolutional network
The three networks

- Regular convolutional network
- Regular capsule network
The three networks

- Regular convolutional network
- Regular capsule network
- Generative capsule network (CapsGAN)
Experiment I: CapsGAN vs Convnet

- Injection after 125,000 iterations

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Iterations to reach initial accuracy</th>
<th>Pre-injection accuracy</th>
<th>Peak accuracy on full test set with full injection</th>
<th>Peak accuracy on full test set with Id-100 injection</th>
</tr>
</thead>
</table>
| Convolutional Generative capsule | 2700  
<100                      | 80.7%  
81.9%                        | <82%                                              | 88.4%                                               |
|                         |                                      |                        | 96.3%                                              | 97.5%                                               |
Experiment I: CapsGAN vs Convnet

Testing accuracy of convolutional and capsule networks

![Graph showing the comparison between Convnet and CapsGAN with different injection rates over training steps.](image-url)
The LD dataset

- Use a small number of new examples
Experiment I: CapsGAN vs Convnet
### Experiment I: CapsGAN vs Convnet

- Injection after 125,000 iterations

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Iterations to reach initial accuracy</th>
<th>Pre-injection accuracy</th>
<th>Peak accuracy on full test set with full injection</th>
<th>Peak accuracy on full test set with ld-100 injection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Convolutional Generative capsule</td>
<td>2700, &lt;100</td>
<td>80.7%, 81.9%</td>
<td>&lt;82%, 96.3%</td>
<td>88.4%, 97.5%</td>
</tr>
</tbody>
</table>
Experiment II: Capsnet vs Convnet

- Injection after ~50k iterations

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Iterations to reach initial accuracy</th>
<th>Pre-injection accuracy</th>
<th>Peak accuracy on dataset after injection</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>1d-1</td>
</tr>
<tr>
<td>Convolutional Capsule</td>
<td>2300 &lt;100</td>
<td>81.4% 84.5%</td>
<td>82.9% 88.2%</td>
</tr>
</tbody>
</table>
Experiment II: Capsnet vs Convnet

Testing accuracy during training and after full-data injections

Accuracy vs Training step

- Capsule network
- Convolutional network
Experiment II: Capsnet vs Convnet

Testing accuracy after low-data injections

Accuracy vs Training step graph comparing Capsnet and Convnet architectures with different data injection scenarios.
Experiment II: Capsnet vs Convnet

- Injection after ~50k iterations

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Iterations to reach initial accuracy</th>
<th>Pre-injection accuracy</th>
<th>Peak accuracy on dataset after injection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Convolutional Capsule</td>
<td>2300 &lt;100</td>
<td>81.4% 84.5%</td>
<td>82.9% 88.2% 87.3% 91.3% 89.5% 92.8% 90.0% 93.0% 86.5% 91.1%</td>
</tr>
</tbody>
</table>
The Unused Capsule effect

- During training, nine of ten pathways are used
The Unused Capsule effect

- During training, nine of ten pathways are used
- Network recognizes that new data does not fit the existing pathways
The Unused Capsule effect

- During training, nine of ten pathways are used
- Network recognizes that new data does not fit the existing pathways
- Tenth pathway is now used
Experiment II: Capsnet vs Convnet

Testing accuracy during training and after full-data injections

Accuracy vs Training step
Future directions

- Dynamic addition of pathways
Future directions

- Dynamic addition of pathways
- Automated “guided” learning
  - Pre-injection is 7 p.p. less
  - Post-injection is 2.5 p.p. better
Future directions

- Dynamic addition of pathways
- Automated “guided” learning
  - Pre-injection is 7 p.p. less
  - Post-injection is 2.5 p.p. Better
- How much data is best?
Future directions

- Dynamic addition of pathways
- Automated “guided” learning
  - Pre-injection is 7 p.p. less
  - Post-injection is 2.5 p.p. Better
- How much data is best?
- More advanced tasks
Acknowledgements

- Slava Gerovitch, Pavel Etingof, Tanya Khovanova, Srinivas Devadas and the MIT PRIMES program
- Maksym Korablyov and Dr. Joseph Jacobson
- My parents
Questions?