Handwritten digit recognition with neural networks

EENG 512

Kevin Carper

4/27/15
Outline

1. Introduction
2. Previous Work
3. Neural Networks
4. Representation
5. Demonstration
6. Conclusions
### Outline

1. **Introduction**
2. **Previous Work**
3. **Neural Networks**
4. **Representation**
5. **Demonstration**
6. **Conclusions**
Problem statement

Goal

- Classify single hand-written digits (0-9) using artificial neural networks
- Attempt to minimize recognition error percentage by
  - Varying number of hidden layer nodes
  - Altering input representation (wavelet transform)
MNIST dataset

- 28×28 pixel handwritten digits
- Digit is centered in image
- Antialiased
- 60,000 training images
- 10,000 test images

<table>
<thead>
<tr>
<th>Digit</th>
<th>Quantity in dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>5923</td>
</tr>
<tr>
<td>1</td>
<td>6742</td>
</tr>
<tr>
<td>2</td>
<td>5958</td>
</tr>
<tr>
<td>3</td>
<td>6131</td>
</tr>
<tr>
<td>4</td>
<td>5842</td>
</tr>
<tr>
<td>5</td>
<td>5421</td>
</tr>
<tr>
<td>6</td>
<td>5918</td>
</tr>
<tr>
<td>7</td>
<td>6265</td>
</tr>
<tr>
<td>8</td>
<td>5851</td>
</tr>
<tr>
<td>9</td>
<td>5949</td>
</tr>
</tbody>
</table>
## Leading Methods

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Test Error Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-NN with non-linear deformation</td>
<td>0.52</td>
</tr>
<tr>
<td>40 PCA + quadratic classifier</td>
<td>3.3</td>
</tr>
<tr>
<td>Virtual SVM, deg-9 poly</td>
<td>0.56</td>
</tr>
<tr>
<td>6-layer NN</td>
<td>0.35</td>
</tr>
<tr>
<td>8-layer convolutional NN</td>
<td>0.35</td>
</tr>
</tbody>
</table>
Neural Network Architecture

Input layer

$X_1$

$X_2$

$X_3$

$X_4$

$X_5$

$X_n$

Hidden layer

Output layer

$\text{Out}_1$

$\text{Out}_2$

$\text{Out}_m$
# Results: For Various Hidden Layer Nodes

<table>
<thead>
<tr>
<th>Hidden Layer Nodes</th>
<th>Cross-entropy Error</th>
<th>Error Rate (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>9.59937</td>
<td>3.94285</td>
</tr>
<tr>
<td>100</td>
<td>9.92417</td>
<td>3.52380</td>
</tr>
<tr>
<td>150</td>
<td>11.59373</td>
<td>2.95238</td>
</tr>
<tr>
<td>1000</td>
<td>16.99909</td>
<td>3.11428</td>
</tr>
</tbody>
</table>

**Observation**

- Diminishing returns with respect to hidden layer size
Results: 50 Nodes Confusion Matrix
Wavelet Transform Theory

Wavelet Transform

- Discretize continuous wavelet transform and do it in 2D
  \[ [W_\psi f](a, b) = \frac{1}{\sqrt{|a|}} \int_{-\infty}^{\infty} \psi \left( \frac{x - b}{a} \right) f(x) dx \]

- Use Daubechies family of orthogonal wavelets
- Use the approximation image as input to the neural network

Why do this?
- More sparse representation
- Essentially denoises the digit images
- Computer may have better performance on different representation
## Results: Wavelet Transform

<table>
<thead>
<tr>
<th>Hidden Layer Nodes</th>
<th>Cross-entropy Error</th>
<th>Error Rate (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>10.38515</td>
<td>3.58095</td>
</tr>
<tr>
<td>100</td>
<td>11.17715</td>
<td>2.90476</td>
</tr>
<tr>
<td>150</td>
<td>12.24351</td>
<td>3.25714</td>
</tr>
</tbody>
</table>

**Observation**

- Not enough data to support that this method is better
Outline

1. Introduction
2. Previous Work
3. Neural Networks
4. Representation
5. Demonstration
6. Conclusions
Cool Matlab demo

https://www.youtube.com/watch?v=mCRL4TQyZig
Remarks

- Increasing number of hidden layer nodes reduces recognition error
- Wavelet transform of input images may reduce recognition error

Future Directions

- Acquire more data for more meaningful wavelet analysis
- Additional hidden layers
- Add noise and rotation to elements in training set for more training samples
References


Questions?