Intriguing properties of neural networks


Presented by Juran Zhang
Two counter-intuitive properties of neural networks

1. There is no distinction between individual high level units and random linear combinations of high level units

2. Input and output mapping are discontinuous
Datasets

MNIST dataset

ImageNet

QuocNet:

~ 10M image samples from Youtube

Unsupervised, ~1 billion learnable parameters
First property

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. **Rich feature hierarchies for accurate object detection and semantic segmentation.**


The inspection of individual units makes the implicit assumption that the units of the last feature layer have semantic information.
First property

“It is the space, rather than the individual units, that contains the semantic information in the high layers of neural networks.”
Figure 1: An MNIST experiment. The figure shows images that maximize the activation of various units (maximum stimulation in the natural basis direction). Images within each row share semantic properties.

(a) Unit sensitive to lower round stroke.
(b) Unit sensitive to upper round stroke, or lower straight stroke.
(c) Unit sensitive to left, upper round stroke.
(d) Unit sensitive to diagonal straight stroke.

Figure 2: An MNIST experiment. The figure shows images that maximize the activations in a random direction (maximum stimulation in a random basis). Images within each row share semantic properties.

(a) Direction sensitive to upper straight stroke, or lower round stroke.
(b) Direction sensitive to lower left loop.
(c) Direction sensitive to round top stroke.
(d) Direction sensitive to right, upper round stroke.
Figure 3: Experiment performed on ImageNet. Images stimulating single unit most (maximum stimulation in natural basis direction). Images within each row share many semantic properties.

(a) Unit sensitive to white flowers.  
(b) Unit sensitive to postures.  
(c) Unit sensitive to round, spiky flowers.  
(d) Unit sensitive to round green or yellow objects.

Figure 4: Experiment performed on ImageNet. Images giving rise to maximum activations in a random direction (maximum stimulation in a random basis). Images within each row share many semantic properties.

(a) Direction sensitive to white, spread flowers.  
(b) Direction sensitive to white dogs.  
(c) Direction sensitive to spread shapes.  
(d) Direction sensitive to dogs with brown heads.
A single neuron’s feature is no more interpretable as a meaningful feature than a random set of neurons
Second property

The deep neural networks learn input-output mappings that are fairly discontinuous.

It is possible to cause the network to misclassify an image by applying a certain hardly perceptible perturbation.
Figure 6: Adversarial examples for QuocNet [10]. A binary car classifier was trained on top of the last layer features without fine-tuning. The examples on the left are recognized correctly as cars, while the images in the middle are not recognized. The rightmost column is the magnified absolute value of the difference between the two images.
Consider a state-of-the-art deep neural network that generalizes well on an object recognition task.

Figure 5: Adversarial examples generated for AlexNet [9]. (Left) is correctly predicted sample, (center) difference between correct image, and image predicted incorrectly magnified by 10x (values shifted by 128 and clamped), (right) adversarial example. All images in the right column are predicted to be a “ostrich, Struthio camelus”, which is fast-running African flightless bird with two-toed feet, largest living bird. Average distortion based on 64 examples is 0.006508.
“Perturbations are found by optimizing the input to maximize the prediction error”

Pedro Tabacof and Eduardo Valle, *Exploring the Space of Adversarial Images*

Second property: observation

Cross networks generalization:
   Applies to all dataset

Cross model generalization:
   Hyper-parameters

Cross training-set generalization:
   Applies to different subsets
Second property: cross model generalization

Adversarial examples stay hard for different hyper-parameters.
Figure 7: Adversarial examples for MNIST compared with randomly distorted examples. Odd columns correspond to original images, and even columns correspond to distorted counterparts. The adversarial examples generated for the specific model have accuracy 0% for the respective model. Note that while the randomly distorted examples are hardly readable, still they are classified correctly in half of the cases, while the adversarial examples are never classified correctly.
<table>
<thead>
<tr>
<th>Model</th>
<th>FC10($10^{-4}$)</th>
<th>FC10($10^{-2}$)</th>
<th>FC10(1)</th>
<th>FC100-100-10</th>
<th>FC200-200-10</th>
<th>AE400-10</th>
<th>Av. distortion</th>
</tr>
</thead>
<tbody>
<tr>
<td>FC10($10^{-4}$)</td>
<td>100%</td>
<td>11.7%</td>
<td>22.7%</td>
<td>2%</td>
<td>3.9%</td>
<td>2.7%</td>
<td>0.062</td>
</tr>
<tr>
<td>FC10($10^{-2}$)</td>
<td>87.1%</td>
<td>100%</td>
<td>35.2%</td>
<td>35.9%</td>
<td>27.3%</td>
<td>9.8%</td>
<td>0.1</td>
</tr>
<tr>
<td>FC10(1)</td>
<td>71.9%</td>
<td>76.2%</td>
<td>100%</td>
<td>48.1%</td>
<td>47%</td>
<td>34.4%</td>
<td>0.14</td>
</tr>
<tr>
<td>FC100-100-10</td>
<td>28.9%</td>
<td>13.7%</td>
<td>21.1%</td>
<td>100%</td>
<td>6.6%</td>
<td>2%</td>
<td>0.058</td>
</tr>
<tr>
<td>FC200-200-10</td>
<td>38.2%</td>
<td>14%</td>
<td>23.8%</td>
<td>20.3%</td>
<td>100%</td>
<td>2.7%</td>
<td>0.065</td>
</tr>
<tr>
<td>AE400-10</td>
<td>23.4%</td>
<td>16%</td>
<td>24.8%</td>
<td>9.4%</td>
<td>6.6%</td>
<td>100%</td>
<td>0.086</td>
</tr>
<tr>
<td>Gaussian noise, stddev=0.1</td>
<td>5.0%</td>
<td>10.1%</td>
<td>18.3%</td>
<td>0%</td>
<td>0%</td>
<td>0.8%</td>
<td>0.1</td>
</tr>
<tr>
<td>Gaussian noise, stddev=0.3</td>
<td>15.6%</td>
<td>11.3%</td>
<td>22.7%</td>
<td>5%</td>
<td>4.3%</td>
<td>3.1%</td>
<td>0.3</td>
</tr>
</tbody>
</table>

Table 2: Cross-model generalization of adversarial examples. The columns of the Tables show the error induced by distorted examples fed to the given model. The last column shows average distortion wrt. original training set.

Linear classifiers with various weight decay parameters lamda

All others: lossdecay = $\lambda \sum w_i^2 /k$, k is number of units in the layer
Second property: cross training set generalization

Adversarial examples stay hard for models trained on different subsets of a dataset.
60000 MNIST partitioned into P1 and P2, trained on non-CNNs

Two networks on P1: cumulative effect of changing hyper-parameters and training sets at the same time

Magnify distortion to make stddev 0.1 before feeding back
Second property: main contribution

For deep neural networks, the smoothness assumption that tiny perturbations of a given image do not normally change the classification result does not hold!
Second property: highlight

They use adversarial examples to train a non-CNN with test error below 1.2% (improved from 1.6% and 1.3%)
Perturbation’s effect on class distributions

Clean Example

Adversarial Example
Pertubation’s effect after adversarial training
Clean examples

Validation set error

0.0782% error on MNIST

Training time (epochs)

Adversarial examples

Validation set error (adversarial)

Training time (epochs)

Standard training
Adversarial training
Second property: fails to illustrate

1. Only experiment results on a subset of MNIST are presented.
2. Does not provide results on convolutional models.
3. Lacks a focus: write several papers
Second property: thoughts

Is this a critical problem for practical applications?

How often the adversarial examples appear?

Deep flaw or more cautious interpretation?

Security implications?