Note: No recitation on Friday (schedule final project checkup with Jingboo instead)
Unstructured Data Analysis

Lecture 12: Intro to neural nets & deep learning

George Chen
2011: Traditional computer vision achieves accuracy ~74%  
2012: Initial deep neural network approach accuracy ~84%  
2015 onwards: Deep learning achieves accuracy 96%+  

Deep Learning

Extremely useful in practice:

• Near human level image classification (including handwritten digit recognition)

• Near human level speech recognition

• Improvements in machine translation, text-to-speech

• Self-driving cars

• *Better* than humans at playing Go
DeepMind’s StarCraft 2 AI is now better than 99.8 percent of all human players

AlphaStar is now grandmaster level in the real-time strategy game

By Nick Statt | @nickstatt | Oct 30, 2019, 2:00pm EDT
Turing Award Won by 3
Pioneers in Artificial Intelligence

From left, Yann LeCun, Geoffrey Hinton and Yoshua Bengio. The researchers worked on key developments for neural networks, which are reshaping how computer systems are built. From left, Facebook, via Associated Press; Aaron Vincent Elkaim for The New York Times; Chad Buchanan/Getty Images

By Cade Metz
March 27, 2019
Is it all hype?
panda
~58% confidence

adversarial noise

~99% confidence

Source: Gizmodo article “This Neural Network's Hilariously Bad Image Descriptions Are Still Advanced AI”. September 16, 2015. (They’re using the NeuralTalk image-to-caption software.)
<table>
<thead>
<tr>
<th>General</th>
<th>VIEW DOCS</th>
</tr>
</thead>
<tbody>
<tr>
<td>no person</td>
<td>0.991</td>
</tr>
<tr>
<td>beach</td>
<td>0.990</td>
</tr>
<tr>
<td>water</td>
<td>0.985</td>
</tr>
<tr>
<td>sand</td>
<td>0.981</td>
</tr>
<tr>
<td>sea</td>
<td>0.980</td>
</tr>
<tr>
<td>travel</td>
<td>0.978</td>
</tr>
<tr>
<td>seashore</td>
<td>0.972</td>
</tr>
<tr>
<td>summer</td>
<td>0.954</td>
</tr>
<tr>
<td>sky</td>
<td>0.946</td>
</tr>
<tr>
<td>outdoors</td>
<td>0.944</td>
</tr>
<tr>
<td>ocean</td>
<td>0.936</td>
</tr>
</tbody>
</table>

**Source:** Pietro Perona

cow is not among top objects found!
elephant is not among top objects found!

Source: David Lopez-Paz
Another AI Winter?

~1970’s: First AI winter over symbolic AI

~1980’s: Second AI winter over “expert systems”

Every time: Lots of hype, explosion in funding, then bubble bursts
Artificial Intelligence—The Revolution Hasn’t Happened Yet

Artificial Intelligence (AI) is the mantra of the current era. The phrase is intoned by technologists, academicians, journalists and venture capitalists.

What is deep learning?
Basic Idea

→ Brain/Machine

→ “clown fish”

Slide by Phillip Isola
Object Recognition

Feature extractors

- Edges
- Texture
- Colors

Classifier

- Segments
- Parts

“clown fish”

Slide by Phillip Isola
Object Recognition

Feature extractors:
- Edges
- Texture
- Colors

Classifier:
- Segments
- Parts

Learned

"clown fish"

Slide by Phillip Isola
Neural Network

Learned

“clown fish”
Deep Neural Network

Learned

“clown fish”

Slide by Phillip Isola
Crumpled Paper Analogy

binary classification: 2 crumpled sheets of paper corresponding to the different classes

deep learning: series ("layers") of simple unfolding operations to try to disentangle the 2 sheets

Analogy: Francois Chollet, photo: George Chen
Representation Learning

Each layer’s output is another way we could represent the input data.

Learned

“clown fish”
Representation Learning

Each layer’s output is another way we could represent the input data.

Visualize (e.g., t-SNE)

Learned

“clown fish”
Why Does Deep Learning Work?

Actually the ideas behind deep learning are old (~1980’s)

- Big data
  - Amazon, Twitter, Facebook, Lyft, Netflix, Fitbit, Google, UPMC

- Better hardware
  - CPU’s & Moore’s law
  - GPU’s
  - TPU’s

- Better algorithms
Structure Present in Data Matters

Neural nets aren’t doing black magic

- **Image analysis**: convolutional neural networks (convnets) neatly incorporates basic image processing structure

- **Time series analysis**: recurrent neural networks (RNNs) incorporates ability to remember and forget things over time
  
  - Note: text is a time series
  
  - Note: video is a time series
Handwritten Digit Recognition Example

Walkthrough of 2 extremely simple neural nets
Handwritten Digit Recognition

28x28 image

- flatten
- length 784 vector (784 input nodes)

- weighted sums (parameterized by a weight matrix $W$ and a bias $b$)

- activation (can be thought of as post-processing)

- linear layer with 10 nodes

- final output
Handwritten Digit Recognition

- Weighted sums:
  - (parameterized by a weight matrix $W$ and a bias $b$)
  - $W$ and $b$

- Input:
  - length 784 vector
  - (784 input nodes)
  - (1D numpy array with 784 entries)

- Linear layer:
  - linear with 10 nodes
  - (2D numpy array of dimensions 784-by-10)
  - (1D numpy array with 10 entries)
Handwritten Digit Recognition

weighted sums

(parameterized by a weight matrix \( W \) and a bias \( b \))

\[
\text{linear}[0] = \text{np.dot}(\text{input}, W[:, 0]) + b[0] \\
\text{linear}[1] = \text{np.dot}(\text{input}, W[:, 1]) + b[1] \\
\vdots \\
\text{linear}[j] = \sum_{i=0}^{783} \text{input}[i] \times W[i,j] + b[j]
\]
Handwritten Digit Recognition

weighted sums
(parameterized by a weight matrix $W$ and a bias $b$)

length 784 vector
(784 input nodes)

linear layer
with 10 nodes
Handwritten Digit Recognition

28x28 image

- flatten

28x28 image

- weight 784 vector
  - length 784 vector
  - (784 input nodes)

- flatten

28x28 image

- weight 784 vector
  - length 784 vector
  - (784 input nodes)

- linear layer
  - linear layer
  - with 10 nodes
  - (784 input nodes)

- activation
  - activation
  - (can be thought of as post-processing)

- final output
  - final output
Handwritten Digit Recognition

Many different activation functions possible

Example: **Rectified linear unit (ReLU)**
zeros out entries that are negative

```
final = np.maximum(0, linear)
```

linear layer with 10 nodes

ReLU
(can be thought of as post-processing)

linear
final output

final
Handwritten Digit Recognition

Many different activation functions possible

Example: **softmax** converts a table of numbers into a probability distribution

\[
\text{exp} = \text{np.exp}(\text{linear})
\]
\[
\text{final} = \text{exp} / \text{exp}.\text{sum}()
\]
Handwritten Digit Recognition

28x28 image

- Flatten
- Weighted sums: parameterized by a weight matrix $W$ and a bias $b$
- Linear layer with 10 nodes
- Softmax

Pr(digit 0)
Pr(digit 1)
Pr(digit 2)
Pr(digit 3)
Pr(digit 4)
Pr(digit 5)
Pr(digit 6)
Pr(digit 7)
Pr(digit 8)
Pr(digit 9)

Desired result
Handwritten Digit Recognition

Training label: 6

Input

Learning this neural net means learning $W$ and $b$

Flatten

Linear (10 nodes)

Softmax

Error is averaged across training examples

Popular loss function for classification (> 2 classes): **categorical cross entropy**

$\log \frac{1}{\text{estimated Pr(digit 6)}}$

⚠️ In PyTorch, softmax is included as part of the cross entropy loss

⚠️ Learning this neural net means learning $W$ and $b$.
This neural net has a name: multinomial logistic regression (when there are only 2 classes, it’s called logistic regression)
Handwritten Digit Recognition

Input

Flatten (512 nodes) → Linear (10 nodes) → ReLU → Linear → Softmax

Training label: 6

Loss → error

Categorical cross entropy

Basic building block of neural nets: linear layer with nonlinear activation

Learning this neural net means learning parameters of both linear layers!
Handwritten Digit Recognition

Training label: 6

This neural net is called a **multilayer perceptron** (# nodes need not be 512 & 10; activations need not be ReLU and softmax)

Important: in lecture, I will sometimes use this notation instead
Handwritten Digit Recognition

Demo
Architecting Neural Nets

- Basic building block that is often repeated: linear layer followed by nonlinear activation.
- Without nonlinear activation, two consecutive linear layers is mathematically equivalent to having a single linear layer!
- How to select # of nodes in a layer, or # of layers?
  - These are hyperparameters! Infinite possibilities!
  - Can choose between different options using hyperparameter selection strategy from earlier lectures
    - Very expensive in practice!
      (Active area of research: neural architecture search)
- Much more common in practice: modify existing architectures that are known to work well
  (e.g., ResNet for image classification/object recognition)
PyTorch GitHub Has Lots of Examples

[m] https://github.com/pytorch/examples

PyTorch Examples

A repository showcasing examples of using PyTorch

- Image classification (MNIST) using Convnets
- Word level Language Modeling using LSTM RNNs
- Training Imagenet Classifiers with Residual Networks
- Generative Adversarial Networks (DCGAN)
- Variational Auto-Encoders
- Superresolution using an efficient sub-pixel convolutional neural network
- Hogwild training of shared ConvNets across multiple processes on MNIST
- Training a CartPole to balance in OpenAI Gym with actor-critic
- Natural Language Inference (SNLI) with GloVe vectors, LSTMs, and torchtext
- Time sequence prediction - use an LSTM to learn Sine waves
- Implement the Neural Style Transfer algorithm on images
- Several examples illustrating the C++ Frontend

Additionally, a list of good examples hosted in their own repositories:

- Neural Machine Translation using sequence-to-sequence RNN with attention (OpenNMT)
Find a Massive Collection of Models at the Model Zoo

Model Zoo
Discover open source deep learning code and pretrained models.

Browse Frameworks  Browse Categories

Filter models...