Lecture 22:
Course Recap
Open Problems in Computer Vision
Assignment 6: Generative Models

Generative Adversarial Networks
Variational Autoencoders

Due on Wednesday 12/9, 11:59pm EST
This Course:
Deep Learning for Computer Vision
Deep Learning for Computer Vision

Building artificial systems that process, perceive, and reason about visual data
Problem: Semantic Gap

What you see

What computer sees

This image by Nikita is licensed under CC BY 2.0
Problem: Visual Data is Complex!

Viewpoint

Illumination

Deformation

Occlusion

Clutter

Intraclass Variation
Machine Learning: Data-Driven Approach

1. Collect a dataset of images and labels
2. Use Machine Learning to train a classifier
3. Evaluate the classifier on new images

Example training set

- airplane
- automobile
- bird
- cat
- deer

```
def train(images, labels):
    # Machine learning!
    return model
```

```
def predict(model, test_images):
    # Use model to predict labels
    return test_labels
```
Model: Deep Convolutional Networks

Krizhevsky, Sutskever, and Hinton, NeurIPS 2012
The Image Classification Challenge:
1,000 object classes
1,431,167 images

Output:
Scale
T-shirt
Steel drum
Drumstick
Mud turtle

Deng et al, 2009
Russakovsky et al. IJCV 2015
Enter Deep Learning

<table>
<thead>
<tr>
<th>Year</th>
<th>Score</th>
<th>Authors/Technique</th>
</tr>
</thead>
<tbody>
<tr>
<td>2010</td>
<td>28.2</td>
<td>Lin et al</td>
</tr>
<tr>
<td>2011</td>
<td>25.8</td>
<td>Sanchez &amp; Perronnin</td>
</tr>
<tr>
<td>2012</td>
<td>16.4</td>
<td>Krizhevsky et al (AlexNet)</td>
</tr>
<tr>
<td>2013</td>
<td>11.7</td>
<td>Zeiler &amp; Fergus</td>
</tr>
<tr>
<td>2014</td>
<td>7.3</td>
<td>Simonyan &amp; Zisserman (VGG)</td>
</tr>
<tr>
<td>2014</td>
<td>6.7</td>
<td>Szegedy et al (GoogLeNet)</td>
</tr>
<tr>
<td>2015</td>
<td>3.6</td>
<td>He et al (ResNet)</td>
</tr>
<tr>
<td>2016</td>
<td>3.0</td>
<td>Shao et al</td>
</tr>
<tr>
<td>2017</td>
<td>2.3</td>
<td>Hu et al (SENet)</td>
</tr>
<tr>
<td></td>
<td>5.1</td>
<td>Russakovsky et al</td>
</tr>
</tbody>
</table>
2012 to Present: Deep Learning Explosion

CVPR Papers

- Submitted
- Accepted
2012 to Present: Deep Learning Explosion

CVPR Papers

CVPR 2021 deadline: 11/16/2020

- Submitted
- Accepted
Deep Learning wasn’t invented overnight!
Deep Learning wasn’t invented overnight!

1958 Perceptron
1963 Roberts
1970s David Marr
1979 Gen. Cylinders
1986 Canny
1997 Norm. Cuts
1999 SIFT
2001 V&J
2001 PASCAL
2009 ImageNet
2018 Turing Award

1959 Hubel & Wiesel
1969 Minsky & Papert
1980 Neocognitron
1985 Backprop
1998 LeNet
2006 Deep Learning
2012 AlexNet
2019 This class

Perceptron
Frank Rosenblatt, ~1957
Deep Learning wasn’t invented overnight!

1958 Perceptron
1963 Roberts
1970s David Marr
1979 Gen. Cylinders
1986 Canny
1997 Norm. Cuts
1999 SIFT
2001 V&J
2001 PASCAL
2009 ImageNet
2019 This class

1959 Hubel & Wiesel
1969 Minsky & Papert
1980 Neocognitron
1985 Backprop
1998 LeNet
2006 Deep Learning
2012 AlexNet
2018 Turing Award

Perceptron

Simple and Complex cells

Stimulus
Response

Frank Rosenblatt, ~1957
Hubel and Wiesel, 1959

Justin Johnson
Lecture 22 - 15
December 2, 2020
Deep Learning wasn’t invented overnight!

- **1958**: Hubel & Wiesel
- **1963**: Roberts
- **1970s**: David Marr
- **1979**: Gen. Cylinders
- **1986**: Canny
- **1980**: Neocognitron
- **1985**: Backprop
- **1986**: Canny
- **1997**: Norm. Cuts
- **1999**: SIFT
- **2001**: V&J
- **2001**: PASCAL
- **2009**: ImageNet
- **2019**: This class

**Key Events**

- **1958**: Perceptron
- **1969**: Minsky & Papert
- **1980**: Neocognitron
- **1985**: Backprop
- **1998**: LeNet
- **2006**: Deep Learning
- **2012**: AlexNet
- **2018**: Turing Award

**Timeline**

- **1958**: Perceptron
- **1969**: Minsky & Papert
- **1980**: Neocognitron
- **1985**: Backprop
- **1998**: LeNet
- **2006**: Deep Learning
- **2012**: AlexNet
- **2018**: Turing Award

**Key Figures**

- **Frank Rosenblatt**, ~1957
- **Hubel and Wiesel**, 1959
- **Fukushima**, 1980

**Figures**

- **Perceptron**
- **Simple and Complex cells**
- **Neocognitron**
Deep Learning wasn’t invented overnight!

1959: Hubel & Wiesel
1963: Roberts
1970s: David Marr
1979: Gen. Cylinders
1986: Canny
1998: LeNet
1999: SIFT
2001: V&J
2001: PASCAL
2009: ImageNet
2019: This class

1958: Perceptron
1969: Minsky & Papert
1980: Neocognitron
1985: Backprop

Al Winter

1997: Norm. Cuts
1999: SIFT
2001: Deep Learning
2001: AlexNet
2009: ImageNet
2018: Turing Award

Perceptron

Simple and Complex cells

Hubel and Wiesel, 1959

Neocognitron

Fukushima, 1980

Convolutional Networks

LeCun et al, 1998

Frank Rosenblatt, ~1957

Stimulus
Response

Image Maps
Convolutions
Subsampling
Fully Connected

Input
Output

Justin Johnson
Lecture 22 - 17
December 2, 2020
Deep Learning wasn’t invented overnight!

1958 Perceptron
1963 Roberts
1970s David Marr
1979 Gen. Cylinders
1986 Canny
1997 Norm. Cuts
1999 SIFT
2001 V&J
2001 PASCAL
2009 ImageNet
2019 This class

1959 Hubel & Wiesel
1969 Minsky & Papert
1980 Neocognitron
1985 Backprop
1998 LeNet
2006 Deep Learning
2012 AlexNet
2018 Turing Award

AlexNet

Krizhevsky, Sutskever, and Hinton, 2012

Justin Johnson
Lecture 22 - 18
December 2, 2020
Deep Learning wasn’t invented overnight!

AlexNet

Krizhevsky, Sutskever, and Hinton, 2012

2018 Turing Award

Yoshua Bengio
Geoffrey Hinton
Yann LeCun

Justin Johnson
Lecture 22 - 19
December 2, 2020
Deep Learning wasn’t invented overnight!

- 1959: Hubel & Wiesel
- 1963: Roberts
- 1970s: David Marr
- 1979: Gen. Cylinders
- 1986: Canny
- 1997: Norm. Cuts
- 1999: SIFT
- 2001: V&J
- 2001: PASCAL
- 2009: ImageNet
- 2020: This class

- 1958: Perceptron
- 1969: Minsky & Papert
- 1980: Neocognitron
- 1985: Backprop
- 1998: LeNet
- 2006: Deep Learning
- 2012: AlexNet
- 2018: Turing Award

Fall 2020: This class
Simple Classifiers: kNN and Linear Classifiers

1-NN classifier

5-NN classifier

Linear Classifiers: $y = Wx + b$
Optimization with Gradient Descent

```
# Vanilla gradient descent
w = initialize_weights()
for t in range(num_steps):
    dw = compute_gradient(loss_fn, data, w)
    w -= learning_rate * dw
```
Problems with Gradient Descent

Local Minima  Saddle points

Poor Conditioning

Gradient Noise

SGD  SGD+Momentum
## Gradient Descent Improvements

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Tracks first moments (Momentum)</th>
<th>Tracks second moments (Adaptive learning rates)</th>
<th>Leaky second moments</th>
<th>Bias correction for moment estimates</th>
</tr>
</thead>
<tbody>
<tr>
<td>SGD</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>SGD+Momentum</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Nesterov</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>AdaGrad</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>RMSProp</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>Adam</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
More Complex Models: Neural Networks

Input: 3072
Hidden layer: 100
Output: 10

\[ f = W_2 \max(0, W_1 x) \]
More Complex Models: Neural Networks

Input: 3072

Hidden layer: 100

Output: 10

Learns bank of templates

\[ f = W_2 \max(0, W_1 x) \]
More Complex Models: Neural Networks

Input: 3072
Hidden layer: 100
Output: 10

\[ f = W_2 \max(0, W_1 x) \]

Universal Approximation

We can build a “bump function” using four hidden units

With 4K hidden units we can build a sum of K bumps
More Complex Models: Convolutional Networks

- Convolution Layers
- Pooling Layers
- Fully-Connected Layers
- Activation Function
- Normalization

\[ \hat{x}_{i,j} = \frac{x_{i,j} - \mu_j}{\sqrt{\sigma_j^2 + \varepsilon}} \]
CNN Architectures: Efficiency

CNN Architecture: Efficiency

- Conv(1x1, c->4C)
- Conv(3x3, c->c)
- Conv(1x1, 4C->c)

G parallel pathways

- ResNeXt
- MobileNets

- Conv(3x3, C->C, groups=C)
- Batch Norm
- ReLU
- Conv(1x1, C->C)
- Batch Norm
- ReLU
Representing Networks: Computational Graphs

\[ f = Wx \]

\[ L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1) \]

[Diagram showing the computational graph with nodes and edges, including symbols for operations and functions.]
Computing Gradients: Backpropagation

\[ \frac{\partial L}{\partial x} = \frac{\partial z}{\partial x} \frac{\partial L}{\partial z} \]

Downstream gradients

\[ \frac{\partial L}{\partial y} = \frac{\partial z}{\partial y} \frac{\partial L}{\partial z} \]

Upstream gradient

Local gradients

\[ \frac{\partial z}{\partial x}, \frac{\partial z}{\partial y} \]

Upstream gradient
Deep Learning Hardware and Software

CPU

GPU

TPU

Static Graphs vs Dynamic Graphs

PyTorch vs TensorFlow
Training Neural Networks: Activation Functions

Sigmoid
\[ \sigma(x) = \frac{1}{1 + e^{-x}} \]

\text{tanh}
\[ \tanh(x) \]

\text{ReLU}
\[ \max(0, x) \]

Leaky ReLU
\[ \max(0.1x, x) \]

Maxout
\[ \max(w_1^T x + b_1, w_2^T x + b_2) \]

ELU
\[ \begin{cases} 
  x & \text{if } x \geq 0 \\
  \alpha(e^x - 1) & \text{if } x < 0 
\end{cases} \]
Training Neural Networks: Data Preprocessing

- Original data
- Zero-centered data
- Normalized data
Training Neural Networks: Weight Initialization

```python
import numpy as np

dims = [4096] * 7
hs = []
x = np.random.randn(16, dims[0])
for Din, Dout in zip(dims[:-1], dims[1:]):
    W = np.random.randn(Din, Dout) / np.sqrt(Din)
    x = np.tanh(x.dot(W))
    hs.append(x)
```

“Just right”: Activations are nicely scaled for all layers!

Glorot and Bengio, “Understanding the difficulty of training deep feedforward neural networks”, AISTAT 2010
Training Neural Networks: Data Augmentation

Load image and label → "cat" → Transform image → CNN → Compute loss
Training Neural Networks: Regularization

**Training**: Add randomness

**Testing**: Marginalize out randomness

**Examples**:
- Batch Normalization
- Data Augmentation
- Dropout
- DropConnect
- Fractional pooling
- Stochastic Depth
- Cutout
- Mixup

---

Justin Johnson  
Lecture 22 - 39  
December 2, 2020
Training Neural Networks: Learning Rate Schedules

Training Loss

Reduce learning rate

Training Loss

Loss

Epoch

Loss

Epoch

Justin Johnson
Lecture 22 - 40
December 2, 2020
Training Neural Networks: Choosing Hyperparameters

**Grid Layout**
- Important Parameter
- Important Parameter
- Unimportant Parameter

**Random Layout**
- Important Parameter
- Important Parameter
- Unimportant Parameter

![Graphs showing training loss and accuracy over iterations for Grid and Random Layouts.](image-url)
Visualizing and Understanding CNNs

Maximally Activating Patches

Nearest Neighbor

Synthetic Images via Gradient Ascent

(Guided) Backprop

Feature Inversion
Making Art with CNNs

DeepDream

Style Transfer
Recurrent Neural Networks: Process Sequences
Recurrent Neural Networks: Architectures

Vanilla Recurrent Network

Long Short Term Memory (LSTM)
Recurrent Neural Networks: Image Captioning

Captions generated using neuraltalk2
All images are CC0 Public domain: cat, suitcase, cat tree, dog, bear, surfers, tennis, giraffe, motorcycle

A dog is running in the grass with a frisbee
A white teddy bear sitting in the grass
Two giraffes standing in a grassy field
A man riding a dirt bike on a dirt track

Attention

e_{t,i,j} = f_{\text{att}}(s_{t-1}, h_{i,j})
\alpha_{t,:,:} = \text{softmax}(e_{t,:,:})
\mathbf{c}_t = \sum_{i,j} \alpha_{t,i,j} h_{i,j}

Each timestep of decoder uses a different context vector that looks at different parts of the input image.

Use a CNN to compute a grid of features for an image.

Self-Attention Layer

One query per input vector

Inputs:
Input vectors: \( X \) (Shape: \( N_X \times D_X \))
Key matrix: \( W_K \) (Shape: \( D_X \times D_Q \))
Value matrix: \( W_V \) (Shape: \( D_X \times D_V \))
Query matrix: \( W_Q \) (Shape: \( D_X \times D_Q \))

Computation:
Query vectors: \( Q = XW_Q \)
Key vectors: \( K = XW_K \) (Shape: \( N_X \times D_Q \))
Value Vectors: \( V = XW_V \) (Shape: \( N_X \times D_V \))
Similarities: \( E = QK^T \) (Shape: \( N_X \times N_X \)) \( E_{i,j} = Q_i \cdot K_j / \sqrt{D_Q} \)
Attention weights: \( A = \text{softmax}(E, \text{dim}=1) \) (Shape: \( N_X \times N_X \))
Output vectors: \( Y = AV \) (Shape: \( N_X \times D_V \)) \( Y_i = \sum_j A_{i,j} V_j \)
Attention is all you need: The Transformer

Vaswani et al, “Attention is all you need”, NeurIPS 2017
Computer Vision Tasks

Classification

Semantic Segmentation

Object Detection

Instance Segmentation

CAT

GRASS, CAT, TREE, SKY

DOG, DOG, CAT

DOG, DOG, CAT

No spatial extent

No objects, just pixels

Multiple Objects

Justin Johnson

Lecture 22 - 50

December 2, 2020
**Object Detection: Single Stage vs Two Stage**

**Single-Stage:**
- YOLO, SSD, RetinaNet
- Make all predictions with a CNN

**Two-Stage:**
- Faster R-CNN
- Use RPN to predict proposals, classify them with second stage
Semantic Segmentation: Fully Convolutional Network

**Downsampling:** Pooling, strided convolution

Design network as a bunch of convolutional layers, with **downsampling** and **upsampling** inside the network!

**Upsampling:** Interpolation, transposed conv

Input: 3 x H x W

High-res: \( D_1 \times \frac{H}{2} \times \frac{W}{2} \)

Low-res: \( D_2 \times \frac{H}{4} \times \frac{W}{4} \)

Med-res: \( D_2 \times \frac{H}{4} \times \frac{W}{4} \)

High-res: \( D_3 \times \frac{H}{4} \times \frac{W}{4} \)

Predictions: \( H \times W \)

Loss function: Per-Pixel cross-entropy

Instance Segmentation: Detection + Segmentation

He et al, “Mask R-CNN”, ICCV 2017
Adding a Dimension: 3D Deep Learning

Predicting 3D Shapes from single image

Processing 3D input data

Mesh R-CNN

Gkioxari, Malik, and Johnson, ICCV 2019

3D Shape Representations
Adding a Dimension: Deep Learning on Video

3D CNNs

H = 224
T = 3
W = 224

3D CNNs

Two Stream Networks

CNN + LSTM

Self-Attention

CNN

CNN

CNN

CNN

CNN

Time

Justin Johnson

Lecture 22 - 55

December 2, 2020
Generative Models

**Autoregressive Models** directly maximize likelihood of training data:

\[
p_\theta(x) = \prod_{i=1}^{N} p_\theta(x_i|x_1, ..., x_{i-1})
\]

Good image quality, can evaluate with perplexity. Slow to generate data, needs tricks to scale up.

**Variational Autoencoders** introduce a latent \( z \), and maximize a lower bound:

\[
p_\theta(x) = \int_z p_\theta(x|z)p(z)dz \geq E_{z \sim q_\phi(z|x)}[\log p_\theta(x|z)] - D_{KL} \left( q_\phi(z|x), p(z) \right)
\]

Latent \( z \) allows for powerful interpolation and editing applications.

**Generative Adversarial Networks** give up on modeling \( p(x) \), but allow us to draw samples from \( p(x) \). Difficult to evaluate, but best qualitative results today.
Reinforcement Learning

RL trains **agents** that interact with an **environment** and learn to maximize **reward**

**Q-Learning**: Train network $Q_\theta(s, a)$ to estimate future rewards for every (state, action) pair. Use **Bellman Equation** to define loss function for training $Q$

**Policy Gradients**: Train a network $\pi_\theta(a | s)$ that takes state as input, gives distribution over which action to take in that state. Use **REINFORCE Rule** for computing gradients
What’s Next?
Prediction #1:
We will discover interesting new types of deep models
Example: Neural ODE

**Residual Network:** $h_{t+1} = h_t + f(h_t, \theta_t)$

Looks kind of like numerical integration...

Chen et al, “Neural Ordinary Differential Equations”, NeurIPS 2018
Example: Neural ODE

**Residual Network:** \( h_{t+1} = h_t + f(h_t, \theta_t) \)

Looks kind of like numerical integration...

**Neural ODE:** Hidden “states” are the solutions of
\[
\frac{dh}{dt} = f(h(t), t, \theta)
\]

A deep network with infinitely many layers!

Chen et al, “Neural Ordinary Differential Equations”, NeurIPS 2018
Prediction #2: Deep Learning will find new applications
Deep Learning for Graphics: NVIDIA DLSS

1080p Aliased, Jittered Pixels

Convolutional Autoencoder

4K Anti-aliased Output

16K Anti-aliased Ground Truth

Temporal Feedback

1080p Motion Vectors

Deep Learning for Graphics: NVIDIA DLSS

Deep Learning for Graphics: Nerfie

(a) Capture Process  (b) Input

Deep Learning for scientific applications

Medical Imaging

Levy et al, 2016

Figure reproduced with permission

Galaxy Classification

Dieleman et al, 2014

Whale recognition

Kaggle Challenge

This image by Christin Khan is in the public domain and originally came from the U.S. NOAA.
Deep Learning for Science: **Protein Folding**

**Input:** 1D sequence of amino acids

**Output:** 3D protein structure
Deep Learning for Science: AlphaFold 2

Deep Learning for Science: AlphaFold 2

Median Free-Modelling Accuracy

Deep Learning for Computer Science

Traditional Hash Table

Kraska et al, “The Case for Learned Index Structures”, SIGMOD 2018
Learn to assign keys to buckets in a way that minimizes hash collisions for the types of data you encounter.

Kraska et al, “The Case for Learned Index Structures”, SIGMOD 2018
Deep Learning for Mathematics

Convert mathematical expressions into graphs, process then with graph neural networks!

\[ \forall f \exists x (f(c, x) \land P(x, f)) \rightarrow \forall f \exists x (f(c, x) \land P(x, f)) \rightarrow \forall \]

Applications: Theorem proving, symbolic integration

Wang et al, “Premise Selection for Theorem Proving by Deep Graph Embedding”, NeurIPS 2017
Kaliszyk et al, “Reinforcement Learning of Theorem Proving”, NeurIPS 2018
Prediction #3: Deep Learning will use more data and compute
Source: https://openai.com/blog/ai-and-compute/
New Hardware for Deep Learning

Cerebras Wafer Scale Engine

Cerebras WSE
1.2 Trillion Transistors
46,225 mm² Silicon

Largest GPU
21.1 Billion Transistors
815 mm² Silicon

SPECIFICATIONS

<table>
<thead>
<tr>
<th>Feature</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sparse Linear Algebra Compute Cores</td>
<td>400,000</td>
</tr>
<tr>
<td>On-chip Memory</td>
<td>18 GB SRAM</td>
</tr>
<tr>
<td>Memory Bandwidth</td>
<td>9.6 PB/sec</td>
</tr>
<tr>
<td>Core-to-Core Bandwidth</td>
<td>100 Pb/sec</td>
</tr>
<tr>
<td>Maximum Power Requirement</td>
<td>20 kW</td>
</tr>
<tr>
<td>System IO</td>
<td>12x100 GbE</td>
</tr>
<tr>
<td>Cooling</td>
<td>Air-cooled</td>
</tr>
<tr>
<td>Dimensions</td>
<td>15 Rack Units (26.25&quot;)</td>
</tr>
</tbody>
</table>

Problem #1: Models are biased
Recall: Vector Arithmetic with GANs

Samples from the model:

- Smiling woman
- Neutral woman
- Neutral man

Average Z vectors, do arithmetic:

\[ \text{Smiling Woman} = \text{Smiling Woman} - \text{Neutral Woman} + \text{Neutral Man} \]

Radford et al, ICLR 2016
Vector Arithmetic with Word Vectors

**Training**: Input a large corpus of text, learn to represent each word with a vector

Can used trained vectors to solve analogies: 
**Man** is to **King** as **Woman** is to **x**?

Find nearest neighbor to: **Man – King + Woman**

Mikolov et al, “Distributed Representations of Words and Phrases and their Compositionality”, NeurIPS 2013
Mikolov et al, “Linguistic Regularities in Continuous Space Word Representations”, NAACL HLT 2013
## Gender Bias in Word Vectors

### Extreme *she* vs. Extreme *he*

<table>
<thead>
<tr>
<th>Extreme <em>she</em></th>
<th>Extreme <em>he</em></th>
</tr>
</thead>
<tbody>
<tr>
<td>1. homemaker</td>
<td>1. maestro</td>
</tr>
<tr>
<td>2. nurse</td>
<td>2. skipper</td>
</tr>
<tr>
<td>3. receptionist</td>
<td>3. protege</td>
</tr>
<tr>
<td>4. librarian</td>
<td>4. philosopher</td>
</tr>
<tr>
<td>5. socialite</td>
<td>5. captain</td>
</tr>
<tr>
<td>6. hairdresser</td>
<td>6. architect</td>
</tr>
<tr>
<td>7. nanny</td>
<td>7. financier</td>
</tr>
<tr>
<td>8. bookkeeper</td>
<td>8. warrior</td>
</tr>
<tr>
<td>9. stylist</td>
<td>9. broadcaster</td>
</tr>
<tr>
<td>10. housekeeper</td>
<td>10. magician</td>
</tr>
</tbody>
</table>

### Gender stereotype *she-he* analogies

<table>
<thead>
<tr>
<th>Gender stereotype <em>she-he</em> analogies</th>
</tr>
</thead>
<tbody>
<tr>
<td>sewing-carpentry</td>
</tr>
<tr>
<td>nurse-surgeon</td>
</tr>
<tr>
<td>blond-burly</td>
</tr>
<tr>
<td>giggle-chuckle</td>
</tr>
<tr>
<td>sassy-snappy</td>
</tr>
<tr>
<td>volleyball-football</td>
</tr>
<tr>
<td>registered nurse-physician</td>
</tr>
<tr>
<td>interior designer-architect</td>
</tr>
<tr>
<td>feminism-conservatism</td>
</tr>
<tr>
<td>vocalist-guitarist</td>
</tr>
<tr>
<td>diva-superstar</td>
</tr>
<tr>
<td>cupcakes-pizzas</td>
</tr>
<tr>
<td>housewife-shopkeeper</td>
</tr>
<tr>
<td>softball-baseball</td>
</tr>
<tr>
<td>cosmetics-pharmaceuticals</td>
</tr>
<tr>
<td>petite-lanky</td>
</tr>
<tr>
<td>charming-affable</td>
</tr>
<tr>
<td>lovely-brilliant</td>
</tr>
</tbody>
</table>

### Gender appropriate *she-he* analogies

<table>
<thead>
<tr>
<th>Gender appropriate <em>she-he</em> analogies</th>
</tr>
</thead>
<tbody>
<tr>
<td>queen-king</td>
</tr>
<tr>
<td>warrior</td>
</tr>
<tr>
<td>sister-brother</td>
</tr>
<tr>
<td>mother-father</td>
</tr>
<tr>
<td>ovarian cancer-prostate cancer</td>
</tr>
<tr>
<td>convent-monastery</td>
</tr>
</tbody>
</table>

---


**Justin Johnson**  
**Lecture 22 - 80**  
**December 2, 2020**
Economic Bias in Visual Classifiers

**Ground-Truth**: Soap
**Source**: UK, $1890/month

DeVries et al, “Does Object Recognition Work for Everyone?”, CVPR Workshops, 2019
Economic Bias in Visual Classifiers

**Ground-Truth:** Soap
**Source:** UK, $1890/month

**Azure:** toilet, design, art, sink
**Clarifai:** people, faucet, healthcare, lavatory, wash closet
**Google:** product, liquid, water, fluid, bathroom accessory
**Amazon:** sink, indoors, bottle, sink faucet
**Watson:** gas tank, storage tank, toiletry, dispenser, soap dispenser
**Tencent:** lotion, toiletry, soap dispenser, dispenser, after shave

DeVries et al, “Does Object Recognition Work for Everyone?”, CVPR Workshops, 2019
Economic Bias in Visual Classifiers

**Ground-Truth:** Soap  
**Source:** Nepal, $288/month

**Azure:** toilet, design, art, sink  
**Clarifai:** people, faucet, healthcare, lavatory, wash closet  
**Google:** product, liquid, water, fluid, bathroom accessory  
**Amazon:** sink, indoors, bottle, sink faucet  
**Watson:** gas tank, storage tank, toiletry, dispenser, soap dispenser  
**Tencent:** lotion, toiletry, soap dispenser, dispenser, after shave

DeVries et al, “Does Object Recognition Work for Everyone?”, CVPR Workshops, 2019
Economic Bias in Visual Classifiers

**Ground-Truth:** Soap  
**Source:** Nepal, $288/month

Azure: food, cheese, bread, cake, sandwich  
Clarifai: food, wood, cooking, delicious, healthy  
Google: food, dish, cuisine, comfort food, spam  
Amazon: food, confectionary, sweets, burger  
Watson: food, food product, turmeric, seasoning  
Tencent: food, dish, matter, fast food, nutriment

**Ground-Truth:** Soap  
**Source:** UK, $1890/month

Azure: toilet, design, art, sink  
Clarifai: people, faucet, healthcare, lavatory, wash closet  
Google: product, liquid, water, fluid, bathroom accessory  
Amazon: sink, indoors, bottle, sink faucet  
Watson: gas tank, storage tank, toiletry, dispenser, soap dispenser  
Tencent: lotion, toiletry, soap dispenser, dispenser, after shave

Commercial object recognition systems work best for objects found in high-income western households

DeVries et al, “Does Object Recognition Work for Everyone?”, CVPR Workshops, 2019
Racial Bias in Visual Classifiers

Google Photos, y'all fucked up. My friend's not a gorilla.

Source: https://twitter.com/jackyalcine/status/615329515909156865 (2015)
### Racial Bias in Visual Classifiers

Commercial gender classifiers fail much more often for women with dark skin

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Metric</th>
<th>All</th>
<th>F</th>
<th>M</th>
<th>Darker</th>
<th>Lighter</th>
<th>DF</th>
<th>DM</th>
<th>LF</th>
<th>LM</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSFT</td>
<td>PPV(%)</td>
<td>93.7</td>
<td>89.3</td>
<td>97.4</td>
<td>87.1</td>
<td>99.3</td>
<td>79.2</td>
<td>94.0</td>
<td>98.3</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td>Error Rate(%)</td>
<td>6.3</td>
<td>10.7</td>
<td>2.6</td>
<td>12.9</td>
<td>0.7</td>
<td>20.8</td>
<td>6.0</td>
<td>1.7</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>TPR (%)</td>
<td>93.7</td>
<td>96.5</td>
<td>91.7</td>
<td>87.1</td>
<td>99.3</td>
<td>92.1</td>
<td>83.7</td>
<td>100</td>
<td>98.7</td>
</tr>
<tr>
<td></td>
<td>FPR (%)</td>
<td>6.3</td>
<td>8.3</td>
<td>3.5</td>
<td>12.9</td>
<td>0.7</td>
<td>16.3</td>
<td>7.9</td>
<td>1.3</td>
<td>0.0</td>
</tr>
<tr>
<td>Face++</td>
<td>PPV(%)</td>
<td>90.0</td>
<td>78.7</td>
<td>99.3</td>
<td>83.5</td>
<td>95.3</td>
<td>65.5</td>
<td>99.3</td>
<td>94.0</td>
<td>99.2</td>
</tr>
<tr>
<td></td>
<td>Error Rate(%)</td>
<td>10.0</td>
<td>21.3</td>
<td>0.7</td>
<td>16.5</td>
<td>4.7</td>
<td>34.5</td>
<td>0.7</td>
<td>6.0</td>
<td>0.8</td>
</tr>
<tr>
<td></td>
<td>TPR (%)</td>
<td>90.0</td>
<td>98.9</td>
<td>85.1</td>
<td>83.5</td>
<td>95.3</td>
<td>98.8</td>
<td>76.6</td>
<td>98.9</td>
<td>92.9</td>
</tr>
<tr>
<td></td>
<td>FPR (%)</td>
<td>10.0</td>
<td>14.9</td>
<td>1.1</td>
<td>16.5</td>
<td>4.7</td>
<td>23.4</td>
<td>1.2</td>
<td>7.1</td>
<td>1.1</td>
</tr>
<tr>
<td>IBM</td>
<td>PPV(%)</td>
<td>87.9</td>
<td>79.7</td>
<td>94.4</td>
<td>77.6</td>
<td>96.8</td>
<td>65.3</td>
<td>88.0</td>
<td>92.9</td>
<td>99.7</td>
</tr>
<tr>
<td></td>
<td>Error Rate(%)</td>
<td>12.1</td>
<td>20.3</td>
<td>5.6</td>
<td>22.4</td>
<td>3.2</td>
<td>34.7</td>
<td>12.0</td>
<td>7.1</td>
<td>0.3</td>
</tr>
<tr>
<td></td>
<td>TPR (%)</td>
<td>87.9</td>
<td>92.1</td>
<td>85.2</td>
<td>77.6</td>
<td>96.8</td>
<td>82.3</td>
<td>74.8</td>
<td>99.6</td>
<td>94.8</td>
</tr>
<tr>
<td></td>
<td>FPR (%)</td>
<td>12.1</td>
<td>14.8</td>
<td>7.9</td>
<td>22.4</td>
<td>3.2</td>
<td>25.2</td>
<td>17.7</td>
<td>5.20</td>
<td>0.4</td>
</tr>
</tbody>
</table>

Buolamwini and Gebru, “Gender Shades: Intersectional Accuracy Disparities in Commercial Gender Classification”, FAT* 2018
Making ML Work for Everyone


Hutchinson and Mitchell, ”50 Years of Test (Un) fairness: Lessons for Machine Learning”, CFAT 2019

Mitchell et al, “Model Cards for Model Reporting”, CFAT 2019

Zhang et al, “Mitigating unwanted biases with adversarial learning”, AAAI 2018

Buolamwini and Gebru, “Gender Shades: Intersectional Accuracy Disparities in Commercial Gender Classification”, CFAT 2018
Problem #2: 
Need new theory?
Empirical Mystery: Good Subnetworks

Step 1: Randomly initialize a network

Step 2: Train on your favorite dataset

Han et al, “Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding”, ICLR 2016
Empirical Mystery: Good Subnetworks

**Step 1:** Randomly initialize a network

**Step 2:** Train on your favorite dataset

**Step 3:** Remove weights of small magnitude

Pruned network works about the same as full network in (2)!

---

Han et al, “Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding”, ICLR 2016
Empirical Mystery: Good Subnetworks

Step 1: Randomly initialize a network

Step 2: Train on your favorite dataset

Step 3: Remove weights of small magnitude

Step 4: Return pruned network weights to initial values

Pruned network works about the same as full network in (2)!

Empirical Mystery: Good Subnetworks

**Step 1:** Randomly initialize a network

**Step 2:** Train on your favorite dataset

**Step 3:** Remove weights of small magnitude

**Step 4:** Return pruned network weights to initial values

**Step 5:** Train pruned network; it works almost as good as (2)!

**Lottery Ticket Hypothesis:**
Within a random deep network is a good subnet that won the “initialization lottery”

Empirical Mystery: Good Subnetworks

Step 1: Randomly initialize a network

Step 2: Find an untrained subnet that works for classification!

I think we are missing something about how to train and initialize deep nets, what training actually does

Ramanujan et al, “What's Hidden in a Randomly Weighted Neural Network?”, arXiv 2019
Empirical Mystery: Generalization

What we expect from classical statistical learning theory:

- Model complexity
- Error

Train vs. Test: Overfitting
Empirical Mystery: Generalization

What we expect from classical statistical learning theory:

- Model complexity
- Error
  - Train
  - Test

Deep networks can achieve 0 training loss on CIFAR with random labels. When we train the same model on real data, why doesn’t it overfit?
Empirical Mystery: Generalization

What we expect from classical statistical learning theory:

- Model complexity
- Error
- Train
- Test
- Overfitting

“Double Descent” for fully-connected models on CIFAR does not match theory!

Belkin et al, “Reconciling modern machine learning practice and the bias-variance trade-off”, PNAS 2019
Empirical Mystery: Generalization

What we expect from classical statistical learning theory:

Similar result for ResNets on CIFAR (when training with label noise)

Problem #3: Deep Learning needs a lot of labeled training data
New Datasets for Low-Shot Learning

**MNIST Dataset**
- **10 classes**: Digits 0 to 9
- **28x28** grayscale images
- **6k images per class** (5k train, 1k test)

**Omniglot Dataset**
- **1623 classes**: Letters from 50 alphabets
- **20 images per class**

Lake et al, “Human-level concept learning through probabilistic program induction,” Science 2015
New Datasets for Low-Shot Learning

KMNIST Dataset
10 classes: 3832 Kanji characters
64x64 grayscale images
1 to 1766 images per class

Omniglot Dataset
1623 classes: Letters from 50 alphabets
20 images per class

Lake et al, “Human-level concept learning through probabilistic program induction,” Science 2015
New Datasets for Low-Shot Learning

**COCO Dataset**
- 118k images
- 80 categories
- 1.2M object instances

New Datasets for Low-Shot Learning

**COCO Dataset**
- 118k images
- 80 categories
- 1.2M object instances

**LVIS Dataset**
- 160k images
- >1000 categories
- ~2M object instances

Gupta et al, “LVIS: A Dataset for Large Vocabulary Instance Segmentation”, CVPR 2019
Using Unlabeled Data: Self-Supervised Learning

**Step 1:** Train a CNN on some "pretext task" that does not require labeled data

**Step 2:** Fine-tune CNN on target task (hopefully using not much labeled data)
Self-Supervised Learning: Jigsaw Puzzles

Source Image

Shuffled patches

Network unscrambles

Self-Supervised Learning: Colorization

**Input:** Grayscale image

**Output:** Color Image


Self-Supervised Learning: Inpainting

**Input:** Image with a hole

**Output:** Hole filled in

Self-Supervised Learning: Contrastive Learning

Unlabeled Images

Random image transformations

Extract features

Patches from the same source image should give similar features
Self-Supervised Learning: Contrastive Learning

- Chen et al, “Improved Baselines with Momentum Contrastive Learning”, arXiv 2020 (MoCo-v2)
- Chen et al, “Big Self-Supervised Models are Strong Semi-Supervised Learners”, arXiv 2020 (SimCLR-v2)
- Chen and He, “Exploring Simple Siamese Representation Learning”, arXiv 2020
Problem #4: Deep Learning doesn’t “Understand” the world
Language Models lack common sense

**Bold = prompt written by me**
*Italic = completion written by GPT-2*

I was born in 1950. In the year 2025 my age will be

Examples generated using [https://talktotransformer.com/](https://talktotransformer.com/)
Language Models lack common sense

I was born in 1950. In the year 2025 my age will be **35**. *That was only a few years ago.* *Most things in life just continue to improve.*

Examples generated using [https://talktotransformer.com/](https://talktotransformer.com/)
I was born in 1950. In the year 2025 my age will be 35. That was only a few years ago. Most things in life just continue to improve.

I see a black dog and a brown horse. The bigger animal's color is
Language Models lack common sense

**Bold** = prompt written by me

*Italic* = completion written by GPT-2

I was born in 1950. In the year 2025 my age will be **35**. *That was only a few years ago.* Most things in life just continue to improve.

I see a black dog and a brown horse. The bigger animal's color is **black**, and the smaller is brown.

Examples generated using [https://talktotransformer.com/](https://talktotransformer.com/)
Language Models lack common sense

Bold = prompt written by me
Italic = completion written by GPT-2

I was born in 1950. In the year 2025 my age will be 35. That was only a few years ago. Most things in life just continue to improve.

I see a black dog and a brown horse. The bigger animal's color is black, and the smaller is brown.

One of my parents is a doctor and the other is a professor. My father is a professor. My mother is

Examples generated using https://talktotransformer.com/
I was born in 1950. In the year 2025 my age will be 35. *That was only a few years ago.* Most things in life just continue to improve.

I see a black dog and a brown horse. The bigger animal's color is black, and the smaller is brown.

One of my parents is a doctor and the other is a professor. My father is a professor. My mother is a social worker. They're super smart people.
Language Models lack common sense

**Bold = prompt written by me**

**Italic = completion written by GPT-2**

I was born in 1950. In the year 2025 my age will be 35. That was only a few years ago. Most things in life just continue to improve.

I see a black dog and a brown horse. The bigger animal's color is black, and the smaller is brown.

One of my parents is a doctor and the other is a professor. My father is a professor. My mother is a social worker. They're super smart people.

What about GPT-3?

Examples generated using [https://talktotransformer.com/](https://talktotransformer.com/)
Language Models lack common sense

Bold = prompt written by me
Italic = completion written by GPT-2

I was born in 1950. In the year 2025 my age will be 35. That was only a few years ago. Most things in life just continue to improve.

I see a black dog and a brown horse. The bigger animal's color is black, and the smaller is brown.

One of my parents is a doctor and the other is a professor. My father is a professor. My mother is a social worker. They're super smart people.

Examples generated using https://talktotransformer.com/
Language Models lack common sense

I was born in 1950. In the year 2025 my age will be 35. That was only a few years ago. Most things in life just continue to improve. One of my parents is a doctor and the other is a professor. My father is a professor. My mother is a social worker. They're super smart people.

Exclusive Microsoft license means I can’t play with it!

Microsoft exclusively licenses OpenAI’s groundbreaking GPT-3 text generation model

Microsoft will get to use the underlying technology of the AI model in its products

Exclusive Microsoft license means I can’t play with it!
Modern object detectors seem to work well!

Rosenfeld et al, “The Elephant in the Room”, arXiv 2018
"The Elephant in the Room"

We add an out-of-context elephant to the scene; Sometimes it is detected Sometimes it messes up other objects: cup

Rosenfeld et al, “The Elephant in the Room”, arXiv 2018
“The Elephant in the Room”

We add an out-of-context elephant to the scene; Sometimes it is missed.

Rosenfeld et al, “The Elephant in the Room”, arXiv 2018
“The Elephant in the Room”

We add an out-of-context elephant to the scene; Sometimes it is assigned the wrong label Or mess up other objects! (cup, couch)

Rosenfeld et al, “The Elephant in the Room”, arXiv 2018
We add an out-of-context elephant to the scene; Sometimes it is assigned the wrong label Or mess up other objects! (cup, couch)

Conclusion: CNNs “see” in a very different way from us. They can fail catastrophically on images even slightly different from those seen during training. How can we fix this?
Deep Learning: Problems and Predictions

Predictions:
- New deep learning models
- New applications
- More compute, new hardware

Problems:
- Models are biased
- Need new theory
- Using less data
- Understanding the world
Deep Learning: Problems and Predictions

**Predictions:**
- New deep learning models
- New applications
- More compute, new hardware

**Problems:**
- Models are biased
- Need new theory
- Using less data
- Understanding the world

Now is a great time to be working in computer vision and machine learning!
Thanks GSIs!

Yunseok Jang  Mohamed El Banani  Danish Syed  Yashmeet Gambhir
Thank You!