CAP 4453
Robot Vision
Dr. Gonzalo Vaca-Castaño
gonzalo.vacacastano@ucf.edu
Administrative details

• Correction of the midterm exam
Credits

• Some slides comes directly from:
  • Yosesh Rawat
  • Andrew Ng
Robot Vision

16. Introduction to Deep Learning I
Outline

• What is Machine Learning?
  • Main basic problems: regression, classification
  • Supervised vs unsupervised
  • Generalization, overfitting

• What is Deep learning?
  • What is Neural network
  • Activation functions
  • Define error
  • What are you optimizing?
  • Chain rule
  • Back-propagation
  • Why deep? How deep?
    • Hyper-parameters
  • Problems with NN. What happened in the 80’s?
    • Vanishing gradient problem
    • Number of parameters

• What kind of problems DN can solve?
  • Regression, classification
  • Computer vision: object detection, semantic segmentation, super-resolution,
  • Time series: NLP, visual questioning/answering
  • Generative models: impersonators()
Introduction
What is object detection

Classification

Classification + Localization

Object Detection

Instance Segmentation

CAT

CAT

CAT, DOG, DUCK

CAT, DOG, DUCK

Single object

Multiple objects

Our goal in object classification

ML → "motorcycle"
Object Detection

• Score subwindow
  • extract features from the image window
  • classifier decides based on the given features.
Some feature representations

SIFT

Spin image

HoG

RIFT

Textons

GLOH
Some feature representations

Coming up with features is often difficult, time-consuming, and requires expert knowledge.
What is Machine Learning?

- *machine learning* is using data to detect patterns. It is the same thing as *AI*.
- What is new?
  - faster
  - cheaper
  - Bigger
  - Feature engineering is generally replaced by Feature learning
- What is the goal of the algorithms?
  - make predictions about future observations of data in the same format (generalization)
  - *input data + weights* $\rightarrow f(\text{weights})$

Today

Feature engineering
Expert knowledge

->

Feature learning
Data
Learning phases

Training
- Images
- Image Features
- Training
- Trained classifier

Testing
- Image not in training set
- Image Features
- Apply classifier
- Prediction
The machine learning framework

- Apply a prediction function to a feature representation of the image to get the desired output:

\[ f(\text{apple}) = \text{“apple”} \]
\[ f(\text{tomato}) = \text{“tomato”} \]
\[ f(\text{cow}) = \text{“cow”} \]
The machine learning framework

\[ f(x) = y \]

**Training:** Given a *training set* of labeled examples:

\[ \{(x_1, y_1), ..., (x_N, y_N)\} \]

Estimate the prediction function \( f \) by minimizing the prediction error on the training set.

**Testing:** Apply \( f \) to an unseen *test example* \( x_u \) and output the predicted value \( y_u = f(x_u) \) to *classify* \( x_u \).
What is machine learning?

• If let’s say f is a linear function in N dimensions, $X = [x_1, x_2, ..., x_N]$, what do you learn?
  • $f(w_1, w_2, ..., w_{N+1}) = w_1 x_1 + w_2 x_2 + \cdots w_N x_N + w_{N+1}$
  • You learn the weights $w$ that match better that function

• Simplest case N=1,
  • *Input Data is number (X axis)*
  • *output value is the Y axis*
  • $f(w_1, w_2) = w_1 x_1 + w_2$

Finding these values is called **training**
Basic problems in machine learning

• You can break most of the machine learning problems in 2 categories:
  • Regression: predicting a value (such as price or time to failure)
  • classification — predicting the category of something (dog/cat, good/bad, wolf/cow)
Basic problems in machine learning

- Supervised
- Unsupervised
- Semi-supervised

FROM SCIKIT-LEARN LIBRARY

scikit-learn algorithm cheat-sheet

classification

Supervised

Unsupervised

Semi-supervised

GET MORE DATA -> GET MORE DATA

>50 SAMPLES

<100K SAMPLES

<100 SAMPLES

<10K SAMPLES

Labeled Data

Clustering

Spectral Clustering

KMeans

MiniBatch KMeans

MeanShift

GMM

LOOKING

PREDICTING A QUANTITY

PREDICTING A CATEGORY

PREDICTING STRUCTURE

Tough Luck

Number of categories known

<10K Samples

<10K Samples

<10K Samples

few features should be important

>10K samples

randomized pca

Isomap

Spectral Embedding

LLE

Kernel approximation

SVR(kernel='linear')

EnsembleRegressor

Lasto ElasticNet

RidgeRegression

SGD Regressor

regression

dimensionality reduction
Generalization AND overfitting
WITH TRAINING DATA

Overfitting in regression

Overfitting in classification
Generalization AND overfitting
WITH NEW TESTING DATA

Overfitting in regression

Overfitting in classification
Generalization AND overfitting
WITH NEW TESTING DATA

Overfitting in regression

Overfitting in classification
So far ...

• Machine learning = AI
• Goal: general function for input data
• Training process: Find parameters for the model
• Supervised: you have labeled data
• Unsupervised: you do not have labeled data
• Semi-supervised: some of your data is labeled
• Overfitting: training adjust very well to your training data, but do not generalize
What is deep learning?
What is deep learning?

• A machine learning technique that solves problems with enormous amount of data.
  • Huge number of tunable parameters
  • Highly non-linear
  • Based on neural networks
    • A stack of neural networks layers
  • It is data driven (not hand-crafted features)
Neurons in the Brain

- Brain is composed of **neurons**
- A neuron receives input from other neurons (generally thousands) from its dendrites
- Inputs are approximately **summed**
- When the input exceeds a threshold, the neuron sends an electrical spike that travels from the body, down the axon, to the next neuron(s)
What is a neuron?

\[ z = w^T x \]

\[ a = \sigma(z) \]

\[ a = \hat{y} \]
What is a neural network?
Composition

Multiple inputs → Hidden layer 1 → Hidden layer 2 → output

Matrix! Matrix! Matrix!

It’s all just matrix multiplication!

GPUs -> special hardware for fast/large matrix multiplication.
Composition: activation function

• Activation function must be a non-linear function.
  • Other case the output will be a linear function
    • Image you have 2 layers

\[ z^{[1]} = W^{[1]} x + b^{[1]} \]
\[ z^{[2]} = W^{[2]} z^{[1]} + b^{[2]} \]
\[ z^{[2]} = W^{[2]} (W^{[1]} x + b^{[1]}) + b^{[2]} \]
\[ = W^{[2]} W^{[1]} x + W^{[2]} b^{[1]} + b^{[2]} \]
\[ = W x + b \]
\[ \hat{y} = z^{[2]} = W x + b \]

The output is always a linear function of the input!
Problem 1 with all linear functions

• We have formed chains of linear functions.
• We know that linear functions can be reduced
  • $g = f(h(x))$

Our composition of functions is really just a single function : ( 
Problem 2 with all linear functions

Linear classifiers:

small change in input can cause large change in binary output.

We want:
Activation function

Pros and cons of activation functions

sigmoid: $a = \frac{1}{1 + e^{-z}}$

tanh: $a = \frac{e^z - e^{-z}}{e^z + e^{-z}}$

ReLU: $a = \max(0, z)$

Leaky ReLU: $a = \max(0.01z, z)$

Andrew Ng
Mark 1 Perceptron

C.1960

20x20 pixel camera feed
Loss function

- Error: Difference between expected value and obtained value
- Example: Image classification
- Loss: sum errors in the training dataset

$$J_1 = \frac{1}{m} \sum_{\text{train}} |\hat{y}_i - y_i|$$

$$J_2 = \frac{1}{m} \sum_{\text{train}} (\hat{y}_i - y_i)^2$$
What are you optimizing?

- Goal: Minimize the loss function!!

What we learn: The parameters of the network

The network is a function $f()$ with parameters $W$ which must be set to the appropriate values to get the desired behavior from the net.

- **Given:** the architecture of the network
- **The parameters of the network:** The weights and biases
  - The weights associated with the blue arrows in the picture
- **Learning the network:** Determining the values of these parameters such that the network computes the desired function
How to minimize a function?

In our case the loss function $w_{\text{new}} = w_{\text{prev}} - \alpha \frac{dJ}{dW}$

Repeat until there is almost no change

$w_{\text{new}} = w_{\text{prev}} - \alpha \frac{dJ}{dW}$

How to compute this gradient?
Back-propagation

- It is a technique to compute the gradient
- Gradients are necessary to get closer to the solution
- FORWARD PASS: You take the inputs, compute the outputs and loss(saving intermedia results)
- From the loss, you start computing backwards to estimate the values of the gradients for all the parameters $w$
Back-propagation

• It is a technique to compute the gradient
• Gradients are necessary to get closer to the solution
• FORWARD PASS: You take the inputs, compute the outputs and loss (saving intermedia results)
• From the loss, you start computing backwards to estimate the values of the gradients for all the parameters $w$

Goal: find

From Layer 2: $\frac{dj}{dw_{11}^2}, \frac{dj}{dw_{12}^2}, \frac{dj}{dw_{13}^2}, \frac{dj}{dw_{14}^2}$

From Layer 1: $\frac{dj}{dw_{11}^1}, \frac{dj}{dw_{12}^1}, ..., \frac{dj}{dw_{33}^1}, \frac{dj}{dw_{34}^1}$
What is a deep network?

- A neural network with many layers
- Highly nonlinear
An example

Upper loop neuron...maybe...

We'll get back to this

Does the network actually do this?

But this is a hope that we might have. A sort of goal with the layered structure like this
What is a deep network?

- A neural network with many layers
- Highly non-linear
So far ...

• A deep network is a neural network with many layers
• A neuron in a linear function followed for an activation function
• Activation function must be non-linear
• A loss function measures how close is the created function (network) from a desired output
• The “training” is the process of find parameters (‘weights’) that reduces the loss functions
• Updating the weights as $w_{new} = w_{prev} - \alpha \frac{dJ}{dW}$ reduces the loss
• An algorithm named back-propagation allows to compute $\frac{dJ}{dW}$ for all the weights of the network in 2 steps: 1 forward, 1 backward
What kind of problems deep learning can solve?
What problems you can solve?

• The fundamental ones:
  • Regression: predict values
  • Classification: predict labels

• Computer vision:
  • object detection
  • semantic segmentation
  • super-resolution,

• Time series:
  • NLP
  • visual questioning/answering

• Generative models
  • impersonators ()
Computer vision

• Find region of interest (regression)
• Find a class label (classification)
Computer vision

• Find a class for each pixel
Computer vision
SUPER-RESOLUTION FROM A SINGLE IMAGE

Figure 5: Sketch of several deep architectures for SISR.
Computer vision
OTHER PROBLEMS

• Super resolution from multiple images
• Denoising
Time series (RNN, LSTM, Attention models)

USE MEASUREMENT TO CHANGE STATE, USE STATE TO PREDICT FUTURE

• Natural language Processing
  • Translation
  • Check Google Bert
  • Visual Questioning answer

• Stocks

• Signals
  • ECG
Generative models
GAN (GENERATIVE ADVERSARIAL NETWORKS)

• Predict the data based on some loose input.
• Looks like the network is able to create something
Generative models

Figure 1: Many problems in image processing, graphics, and vision involve translating an input image into a corresponding output image. These problems are often treated with application-specific algorithms, even though the setting is always the same: map pixels to pixels. Conditional adversarial nets are a general-purpose solution that appears to work well on a wide variety of these problems. Here we show results of the method on several. In each case we use the same architecture and objective, and simply train on different data.

Generative models

IMAGE CREATION FROM TEXT

• Generative Adversarial Text to Image Synthesis. Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, Honglak Lee. ICML 2016

Figure 1. Examples of generated images from text descriptions. Left: captions are from zero-shot (held out) categories, unseen text. Right: captions are from the training set.
Generative models

CREATE FAKE MODELS

• https://youtu.be/p1b5aiTrGzY
Questions?