HW3 Prep

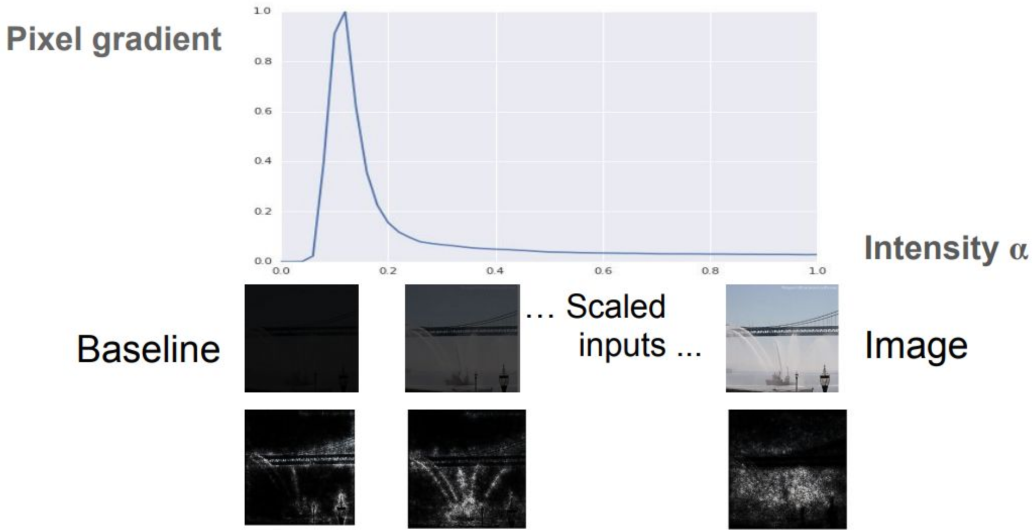
18739

Agenda

- HW3 is out
  - Due April 17 Before Class
- Review of explanation methods
  - Integrated Gradients
  - Influence-Directed Explanations
  - Their Relationships
- HW 3 Overview
- Generative Models

Integrated Gradients

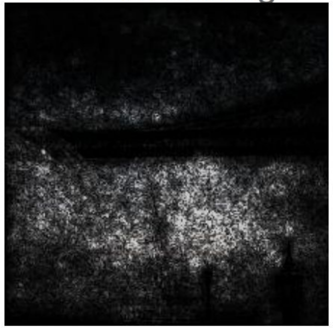
$$\mathrm{IG}(\mathrm{input}, \mathrm{base}) ::= (\mathrm{input} - \mathrm{base}) * \int_{0\text{-}1} \nabla F(\boldsymbol{\alpha}*\mathrm{input} + (1-\boldsymbol{\alpha})*\mathrm{base})\, d\boldsymbol{\alpha}$$



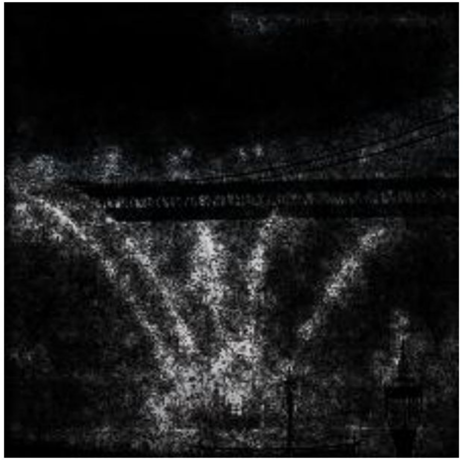
Integrated Gradients

Original image

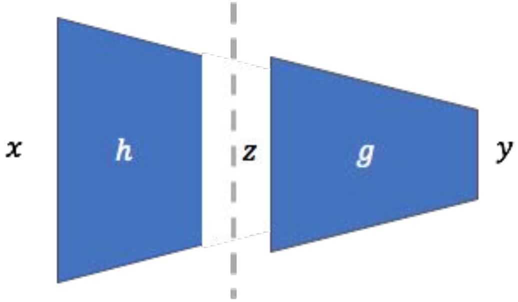
Gradient at image



Integrated gradient



Influence Directed Explanations



$$y = f(x) = g(h(x))$$

Definition 1. *The influence of an element j in the internal representation defined by $s = \langle g, h \rangle$ is given by*

$$\chi_j^s(f, P) = \int_{\mathcal{X}} \left. \frac{\partial g}{\partial z_j} \right|_{h(\mathbf{x})} P(\mathbf{x}) d\mathbf{x} \tag{1}$$

Influence Directed Explanations

Comparison between the two

- Integrated Gradients:

IG(input, base) ::= (input - base) * $\int_{0\text{-}1} \nabla F(\boldsymbol{\alpha}\text{*input} + (1\text{-}\boldsymbol{\alpha})\text{*base})\, d\boldsymbol{\alpha}$

- Internal / Distributional Influence

Definition 1. *The influence of an element j in the internal representation defined by $s = \langle g, h \rangle$ is given by*

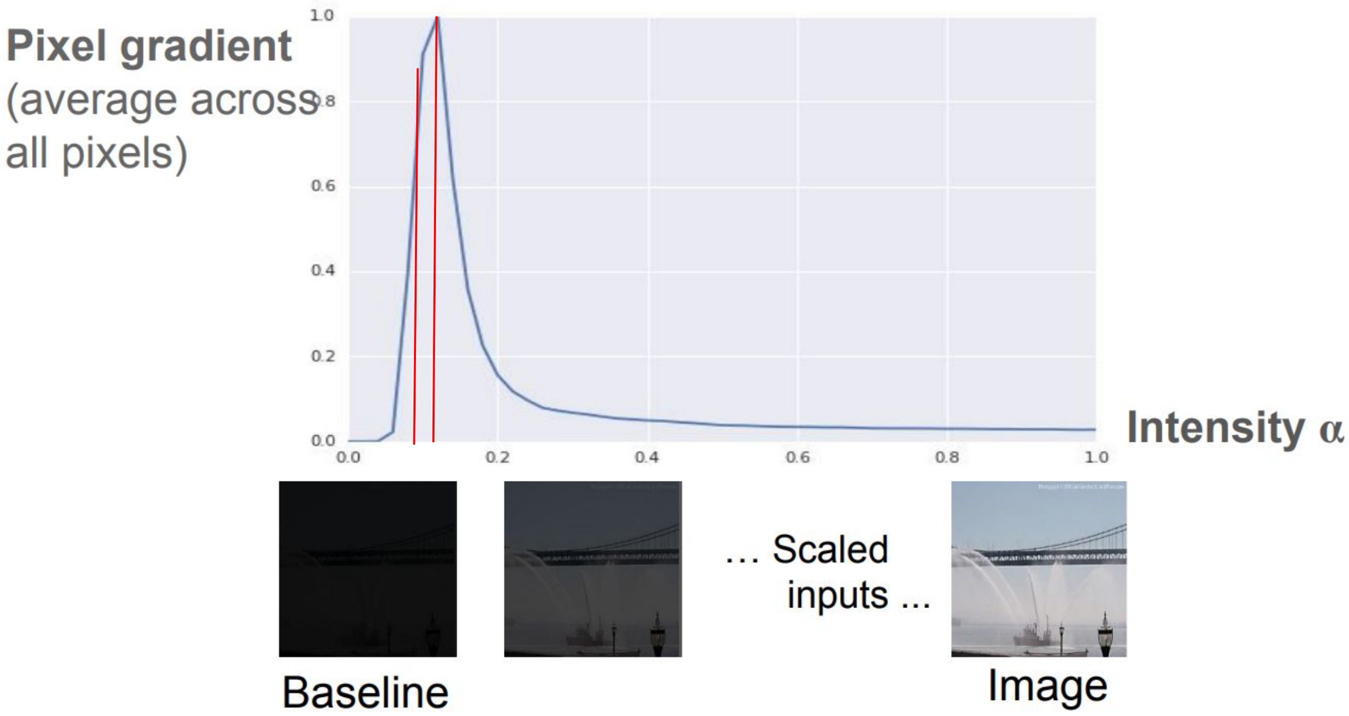
$$\chi_j^s(f, P) = \int_{\mathcal{X}} \frac{\partial g}{\partial z_j}\bigg|_{h(\mathbf{x})} P(\mathbf{x}) d\mathbf{x} \qquad (1)$$

Approximation Method for IG

IG(input, base) ::= (input - base) * $\int_{0\ -1}$ ∇F(**α***input + (1-**α**)*base) d**α**

$$\text{IntegratedGrads}_i^{approx}(x) ::= \\ (x_i - x'_i) \times \Sigma_{k=1}^{m} \frac{\partial F(x' + \frac{k}{m} \times (x - x')))}{\partial x_i} \times \frac{1}{m} \tag{3}$$

Approximation Method



Approximation Method

$$\text{IntegratedGrads}_i^{approx}(x) ::= (x_i - x'_i) \times \Sigma_{k=1}^{m} \frac{\partial F(x' + \frac{k}{m} \times (x - x'))}{\partial x_i} \times \frac{1}{m} \qquad (3)$$

```
def integrated_gradients(inp, base, label, steps=50):
  scaled_inps = [base + (float(i)/steps)*(inp-base) for i in range(0, steps)]
  predictions, grads = predictions_and_gradients(scaled_inputs, label)
  integrated_gradients = (img - base) * np.average(grads, axis=0)
  return integrated_gradients
```

HW3 Overview

Definition 1. *The influence of an element j in the internal representation defined by $s = \langle g, h \rangle$ is given by*

$$\chi_j^s(f, P) = \int_{\mathcal{X}} \frac{\partial g}{\partial z_j}\bigg|_{h(\mathbf{x})} P(\mathbf{x}) d\mathbf{x} \qquad (1)$$

- 3 Parts
- Focused Explanation of A Slice
  - What is the influence of neuron 0 towards the output class score C for an instance/Class
  - The attribution for a class is the mean attribution of every single instance in the class
  - Compare with Integrated Gradients
- Comparative Explanations
  - What is the influence of neuron 0 towards the output class score C1-C2 for an instance/Class
  - Compare with the explanations using only one class
- Model Compression - Visualizing the essence of a class
  - 5 random classes
  - Find the neurons with most negative/positive influences
  - Mask the slice with selected neurons to create a binary classifier for that class

Theano/Keras Overview

- In theano, everything is a variable
  - So to take the gradient of an output (Q) with respect to an input (inpt)
  - theano.grad(Q, wrt = inpt)
- How to calculate Q from a model?
  - Quantity of interest: the influence of a neuron towards a output class c1
  - Q = T.take(variable for output, c1, axis=1) (Equivalent to numpy v[:, c1])
  - Inpt = variable for input to the layer
- How to get the variable for output/input of a certain layer in a Keras model?
  - Using Theano Backend
  - V1 = model.layers[n].output
  - V2 = model.layers[n].output
  - W = model.layers[n].get_weights()

Top Neuron

- Given an attribution map of slice A (50 neurons), which neuron is the most influential?
  - 50 * 28 * 28
  - Max
  - Average(sum)
- Use max for this homework
  - A neuron is influential if a feature is influential

Visualization

- Saliency Map
    - Compute Gradients of neuron activation wrt input pixels
    - Scale pixels of image accordingly
- You can use the same integrated influence measures
    - Path-Integrated gradients of neuron activation wrt input pixels
    - Scale pixels of image accordingly

Linear Activation Layer

```
model.add(Flatten())
model.add(Dense(4096, activation='relu'))
#model.add(Dropout(0.5))
model.add(Activation('linear'))
model.add(Dense(4096, activation='relu'))
#model.add(Dropout(0.5))
model.add(Activation('linear'))
model.add(Dense(1000, activation='linear'))
```

- Avoid overshadowing of class scores
    - Softmax tends to decrease the score of the smaller value

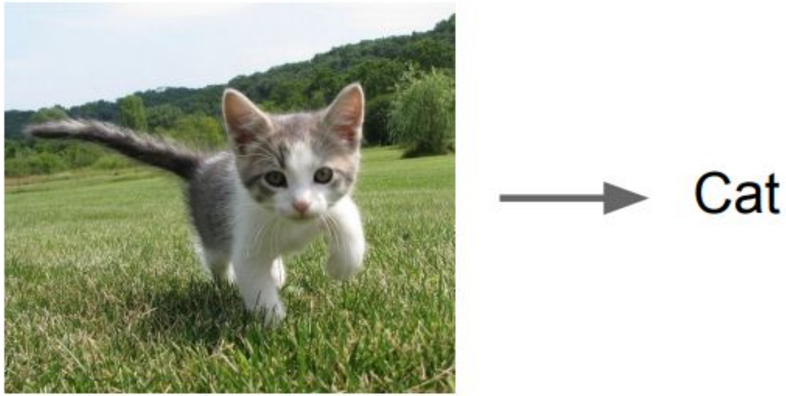
$$\boxed{\sigma(x_j) = \frac{e^{x_j}}{\sum_i e^{x_i}}}$$

Generative Models

18739

Supervised vs. Unsupervised learning

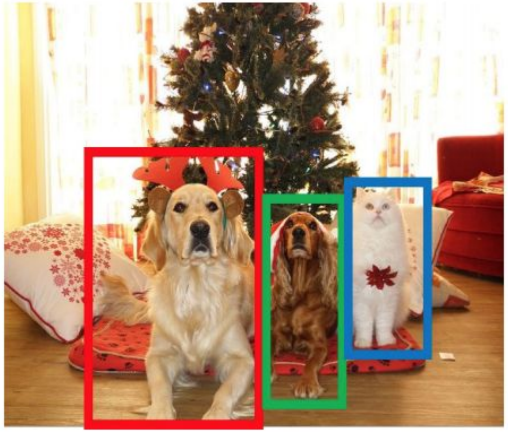
- Supervised learning
- Data (X,y)
  - X -> Data
  - Y-> Label
- Goal: Learn a function to map x->y
  - Map Image to label
- Examples:
  - Classification
  - Regression
  - Object Detection. etc.



Classification

Supervised vs. Unsupervised learning

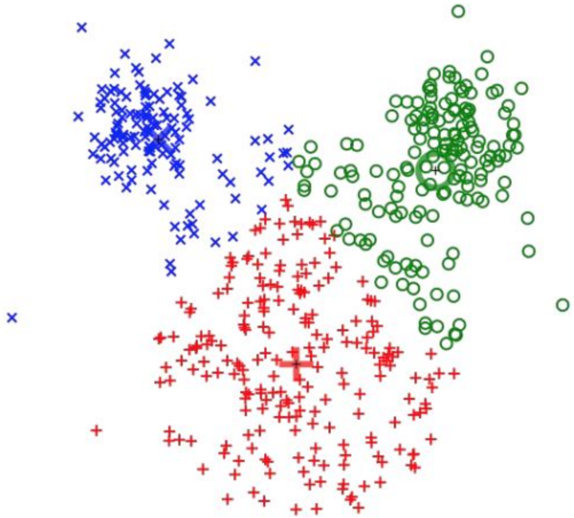
- Supervised learning
- Data (X,y)
  - X -> Data
  - Y-> Label
- Goal: Learn a function to map x->y
  - Map image to (bounding boxes, labels)
- Examples:
  - Classification
  - Regression
  - Object Detection. etc.



Object Detection

Supervised vs. Unsupervised learning

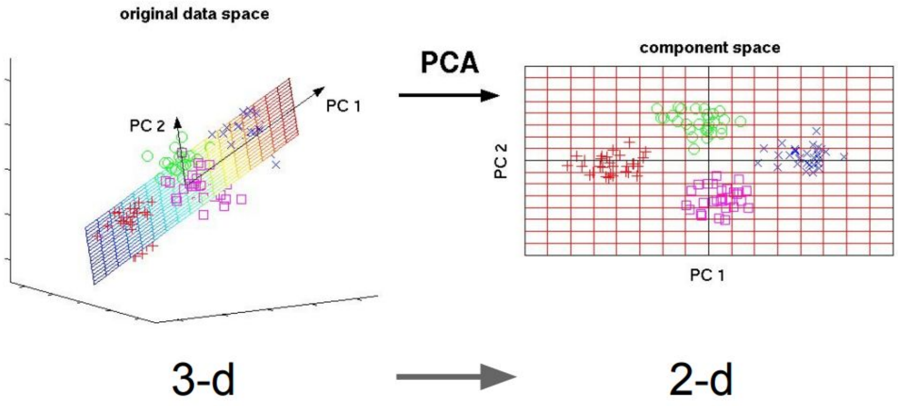
- Unsupervised learning
- Just Data X, No Labels!
- Goal: Learn some underlying hidden structure of the data
- Examples:
  - Clustering
  - Dimensionality reduction
  - Feature learning
  - Density estimation, etc.



K-means clustering

Supervised vs. Unsupervised learning

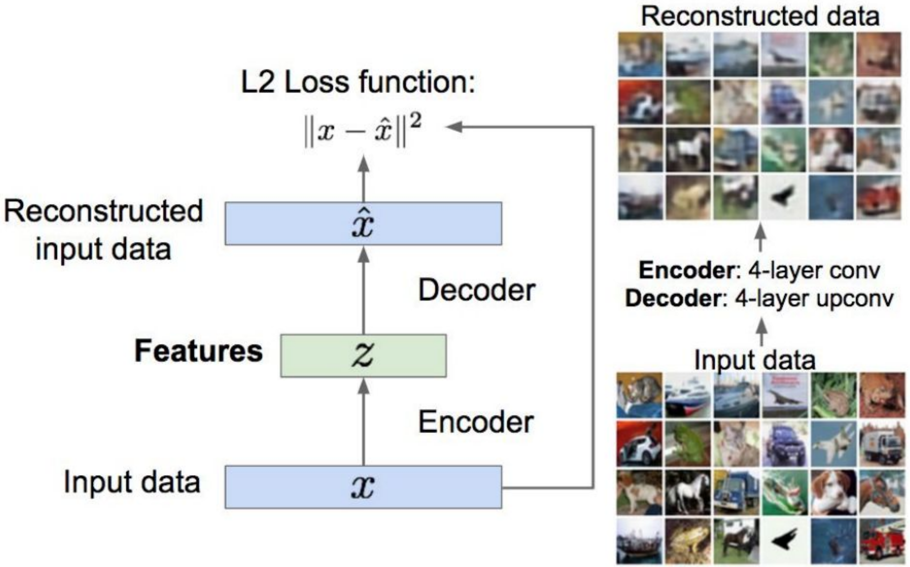
- Unsupervised learning
- Just Data X, No Labels!
- Goal: Learn some underlying hidden structure of the data
- Examples:
  - Clustering
  - Dimensionality reduction
  - Feature learning
  - Density estimation, etc.



Principal Component Analysis
(Dimensionality reduction)

Supervised vs. Unsupervised learning

- Unsupervised learning
- Just Data X, No Labels!
- Goal: Learn some underlying hidden structure of the data
- Examples:
  - Clustering
  - Dimensionality reduction
  - Feature learning
  - Density estimation, etc.



Autoencoders
(Feature learning)

Supervised Learning

- Data (X,y)
  - X -> Data
  - Y-> Label
- Goal: Learn a function to map x->y
  - Map Image to label
- Examples:
  - Classification
  - Regression
  - Object Detection. etc.

Unsupervised Learning

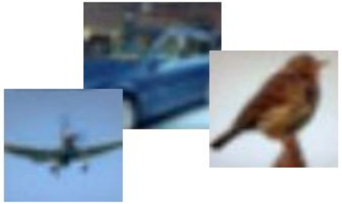
- Just Data X, No Labels!
- Goal: Learn some underlying hidden structure of the data
- Examples:
  - Clustering
  - Dimensionality reduction
  - Feature learning
  - Density estimation, etc.

Advantages of Unsupervised Learning

- Training Data is cheap!
- Solve unsupervised learning => understand structure of visual world
- Representation => Understanding => Explanations

Generative Model

Given training data, generate new samples from same distribution



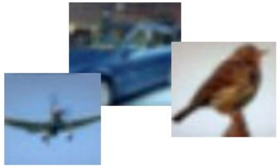
Training data ~ $p_{data}(x)$

Generated samples ~ $p_{model}(x)$

Want to learn $p_{model}(x)$ similar to $p_{data}(x)$

Generative Model

Given training data, generate new samples from same distribution



Training data ~ $p_{data}(x)$

Generated samples ~ $p_{model}(x)$

Want to learn $p_{model}(x)$ similar to $p_{data}(x)$

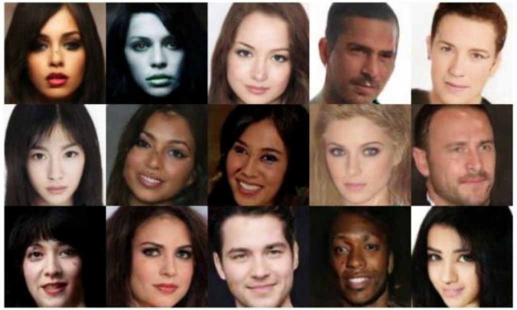
Addresses density estimation, a core problem in unsupervised learning

Several flavors:

- Explicit density estimation: explicitly define and solve for pmodel(x)
- Implicit density estimation: learn model that can sample from pmodel(x) w/o explicitly defining it

Generative Model

- Realistic samples for artwork, super-resolution, colorization, etc.


- Generative models of time-series data can be used for simulation and planning (reinforcement learning applications!)
- Training generative models can also enable inference of latent representations that can be useful as general features

Generative Adversarial Network

- Estimate the implicity density of image space

- Next Class

References

- http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture13.pdf