FourthBrain

# MLE Program, Cohort 11 (MLE11)

## Week 9: Computer Vision Benchmarks, Dealing with Images, Object Detectors, Semantic Segmentation, Explainability & Saliency

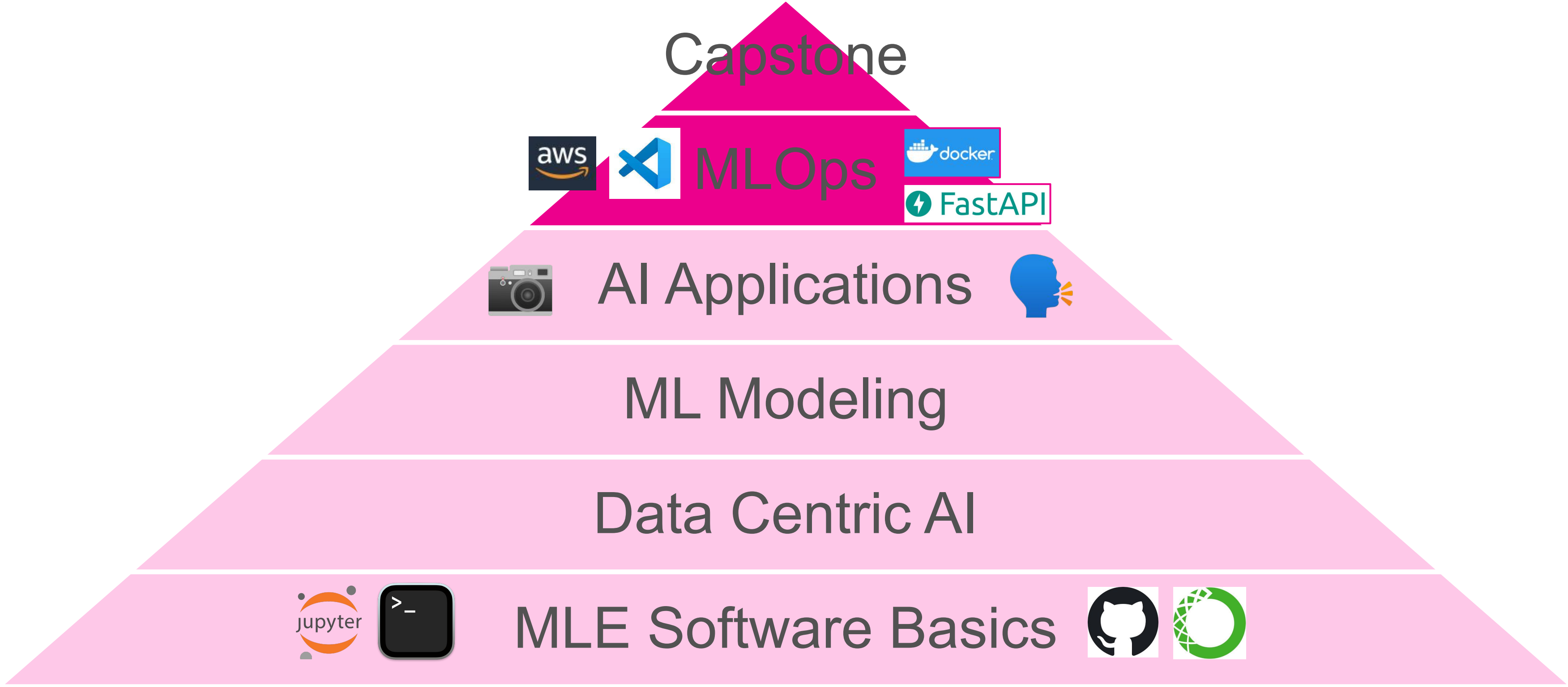
# 📢 FourthBrain Update!

- Following up from our slack message in December
- FourthBrain has continued evolving since then (i.e., Andrew's post)
- *If you are looking for a job*, we strongly encourage you to **maximize the Career Services resources** starting now
- 🦸 Do not underestimate the benefits capstone + interviewing!

# 💼 Career Services To-Do

- Confirm that you are actively looking for a job here: Survey
- Join a job search weekly meetup group now!: Job Search Groups
- James' booking link: https://calendly.com/jamesfourthbrain
- Greg's booking link: https://calendly.com/ai-greg/30min

# Becoming a Machine Learning Engineer



# Our Updated Curriculum!

1. ML Project Scoping
2. Real, Live Data Streams
3. Data Wrangling & Exploratory Analysis
4. Big Data

DATA CENTRIC AI

5.Supervised ML
6. Deep Learning & AutoML
7. Unsupervised, Semi- & Self-supervised Learning

ML MODELING

8. Computer Vision
9. Natural Language Processing
10. Transformers & Fine Tuning Pre-Trained Networks

AI APPLICATIONS

11. Building ML Web Apps
12. Containerization
13. Model Serving
14. Machine Learning in Production

MLOps

# Last Week!

## Concepts

- Dealing with Unstructured Data
- Clustering
- Dimensionality Reduction
- Label propagation/label spreading
- Co-training algorithms
- Zero-shot learning

## Hands on

- Predicting customer responses and metadata tagging using data visualization with Tensorboard
- Midterm Project Assignment

# 🤖 This Week!

**Concepts**

- Dealing with Images
- Convolutional Networks
- Object Detectors
- Semantic Segmentation
- Computer Vision Benchmarks
- Explainability & Saliency

**Hands on**

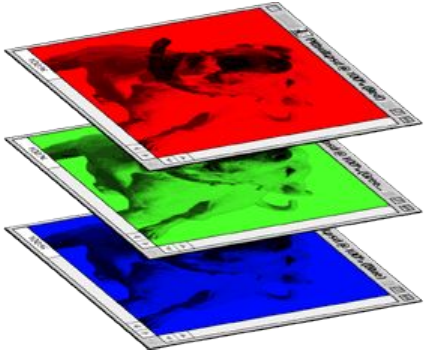
- Few-shot object detection

What questions do you have?

# Dealing with Images

# RGB vs Grayscale

- RGB :
  - Red Green Blue
  - Image consists of 3 channels (one for each color)
  - These channels are superposed on each other to give us the colors we know

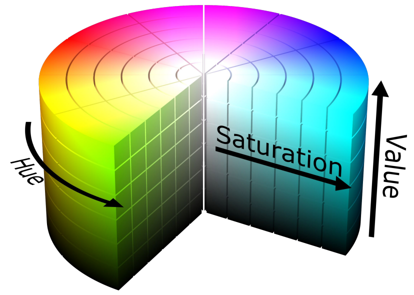



- Grayscale:
  - Could be referred to as black and white
  - Consist of only one channel

- The concept of image channels is very important in order to fully understand how Computer Vision algorithms work

# What is HSV?

- Unlike RGB, which uses primary colors, HSV is closer to how humans perceive colors
- It has 3 components:
    - Hue
    - Saturation
    - Value




- This color space describes colors (hue or tint) in terms of their shade (saturation or amount of gray) and their brightness value.
- Some people call HSV as HSB where “B” represents “Brightness”

# Hue

- Hue is the color portion of the model, expressed as a number from 0 to 360 degrees:
  - Red: 0 to 60 degrees
  - Yellow: 61 to 120 degrees
  - Green: 121 to 180 degrees
  - Cyan: 181 to 240 degrees
  - Blue: 241 to 300 degrees
  - Magenta: 301 to 360 degrees

# Saturation and Value

- Saturation:
  - Describes the amount of gray in a particular color
  - Ranges from 0 to 100
  - Reducing this component toward zero introduces more grey and produces a faded effect
  - Sometimes, saturation appears as a range from 0 to 1, where 0 is grey and 1 is a primary color
- Value (Brightness)
  - Value works in conjunction with saturation
  - It describes the intensity of the color, from 0 to 100 percent
  - 0 is completely black and 100 is the brightest and reveals the most color

# Dealing with Images | Python

- There are many libraries that handle images in python

- Examples:
    - PIL
    - OpenCV

- These libraries provide general image handling and a lot of basic image operations like:
    - Resizing
    - Cropping
    - Rotating
    - Color Conversions

# Dealing with Images | Example

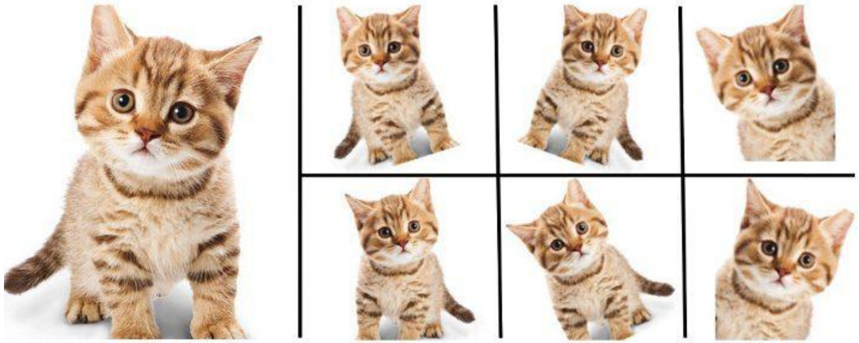
# Dealing with images | Augmentation

- Image Augmentation is a technique that can be used to artificially expand the size of a training set by creating modified Images from the existing one

- It is mainly used to:
  - Generate more images
  - Prevent Overfitting
- There are many ways to work with image augmentations in python:
  - Tensorflow / Keras
  - PyTorch
  - Augmentor
  - Imgaug

# Dealing with images | Augmentation

- You can do a lot of things to augment images:
    - Perspective Skewing: look at the image from a different angle
    - Elastic distortions: add distortions to an image
    - Mirroring: apply different types of flips
    - Shearing: tilt an image along with one of its sides
    - Rotating
    - Cropping

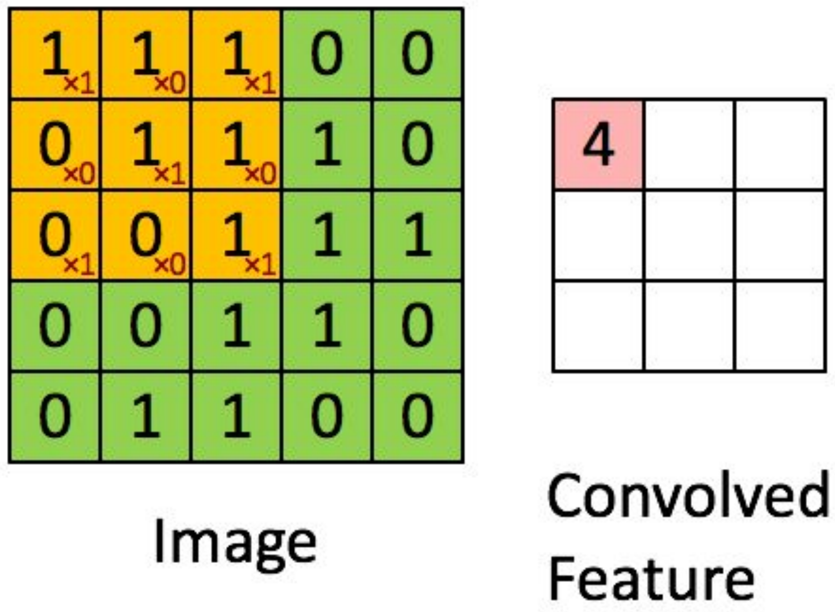


# What questions do you have?

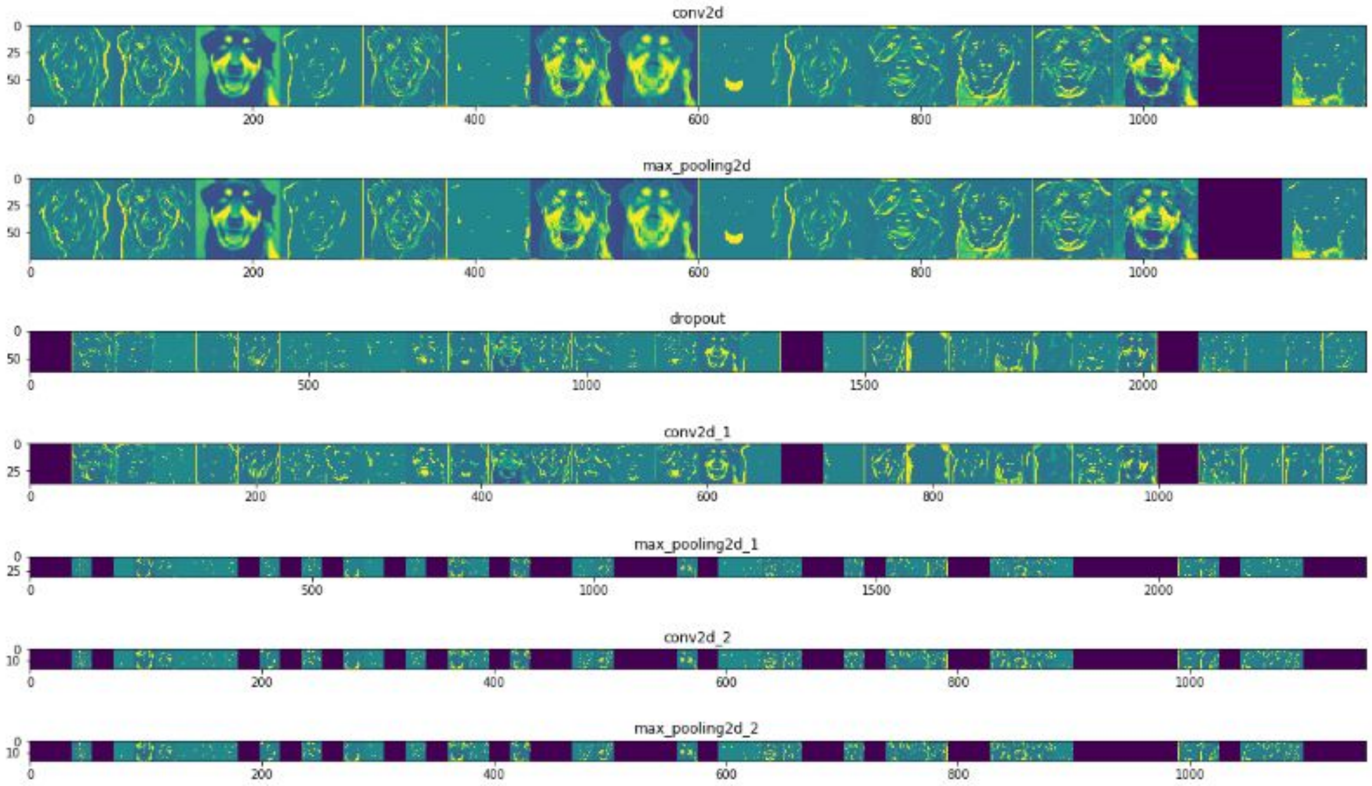
# Convolutional Networks

# Convolution

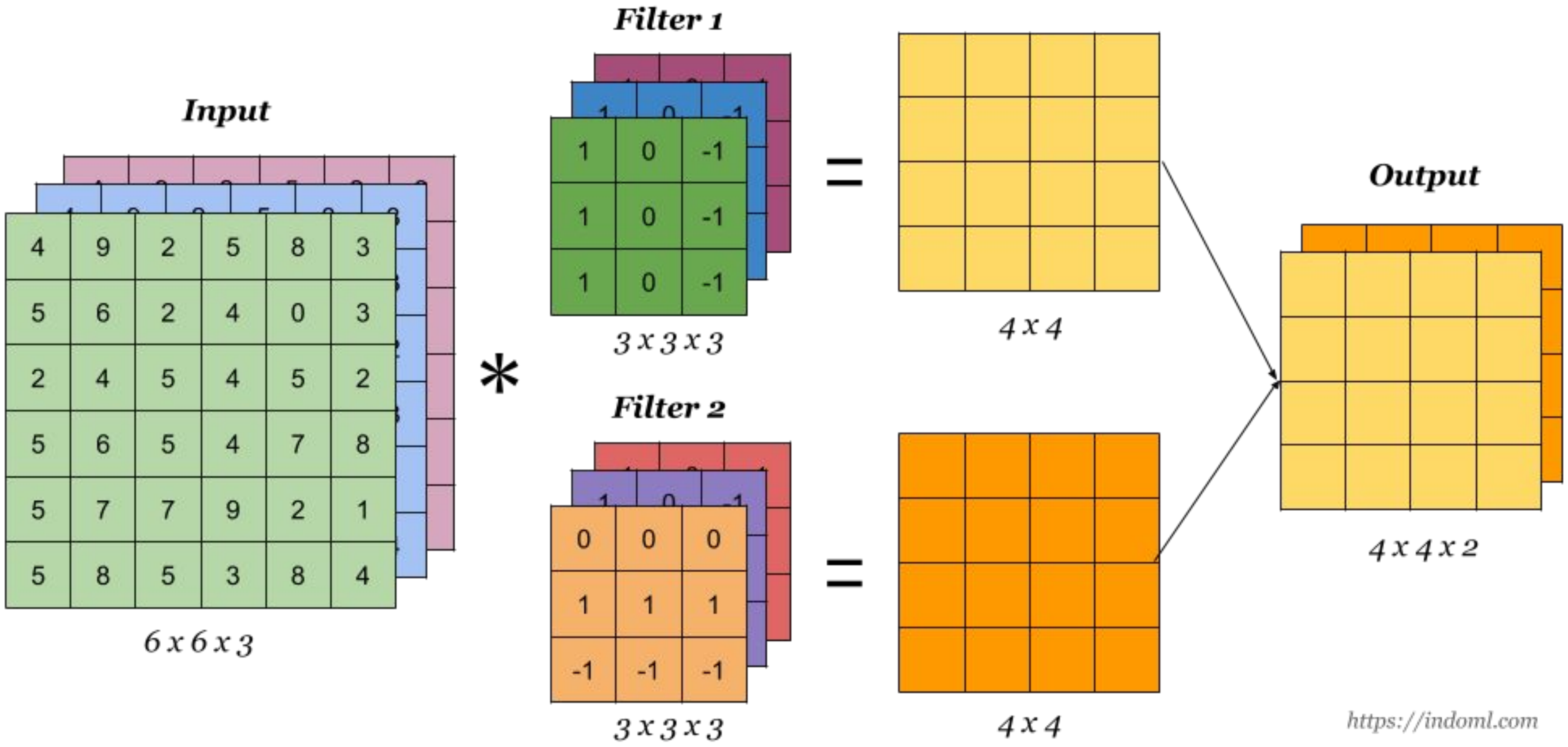
- Captures space relationships
- Kernel/Filter (Yellow)
- Local receptive field
- Parameter sharing
- Location independent features



# Convolution



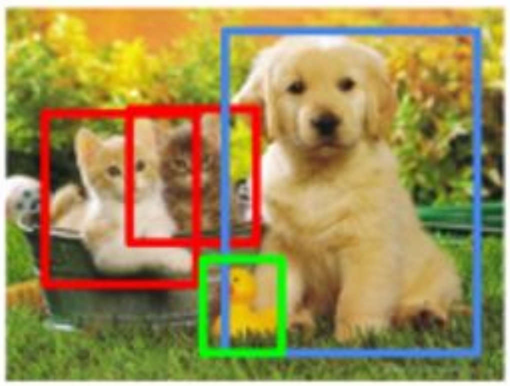
# Convolutional Layer



Classification

Cat

Detection



Cat, Duck

Segmentation

Cat, Duck

# Discussion

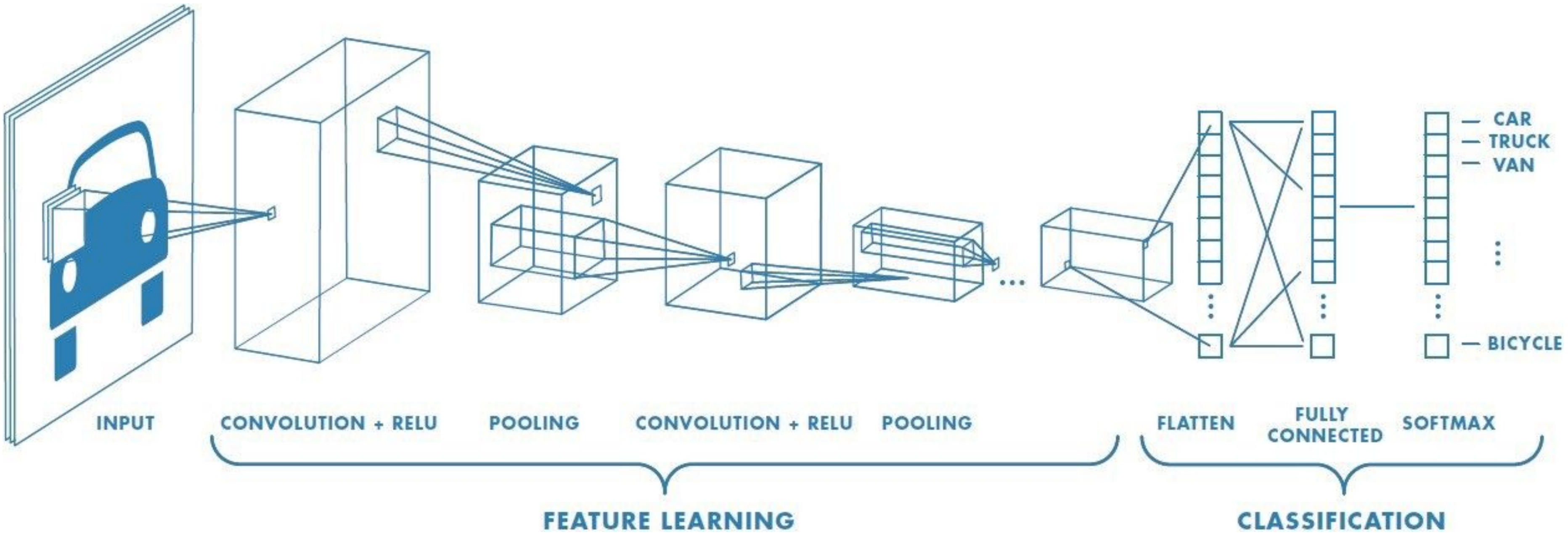
15 min

1. **What is the difference between CNN vs. Dense?**
2. **What are some applications of:**
   - **Classification**
   - **Detection**
   - **Segmentation**
3. **How would the architecture look like in each case?**

Designate *one person to share* from your breakout room

# Classification

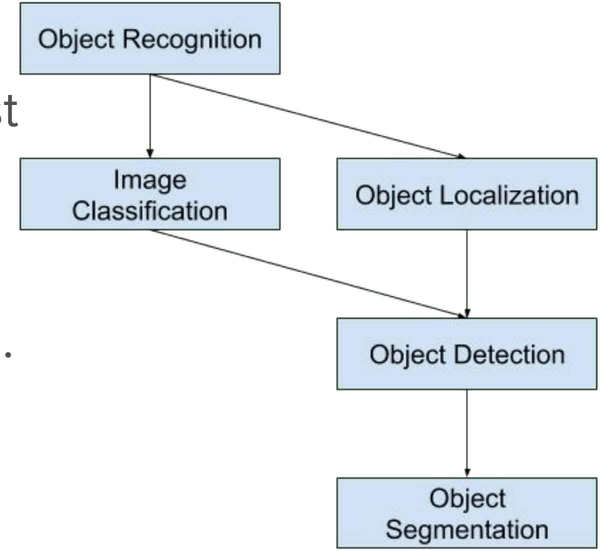
# Classifier


CAR
TRUCK
VAN
BICYCLE
INPUT
CONVOLUTION + RELU
POOLING
CONVOLUTION + RELU
POOLING
FLATTEN
FULLY CONNECTED
SOFTMAX
FEATURE LEARNING
CLASSIFICATION

# Object Detectors

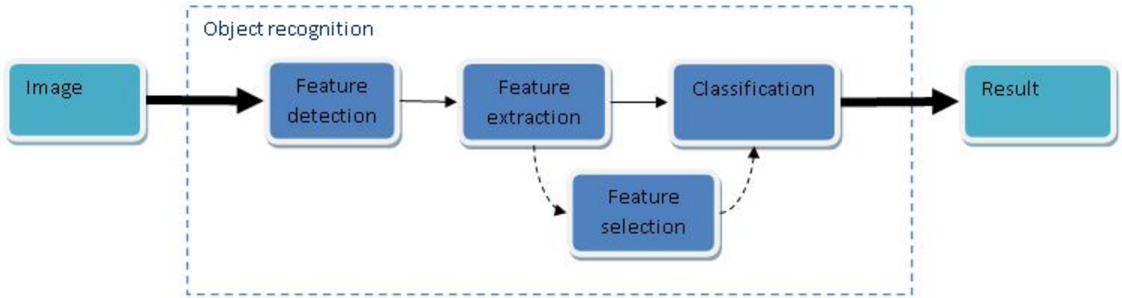
# Object Detectors

- Object Detection models attracted a lot of attention due to the boom in the Computer Vision market
- To interpret an image / video, the computer has to first detect the objects and also precisely estimate their location in the image / video before classifying them.
- There are multiple architectures from traditional techniques to modern and state-of-the-art techniques. These architectures differ from each other based on the accuracy, speed, and hardware resources required.



# Object Detection | Traditional Approach

- The traditional object detection has usually 3 stages
  - Informative Region Selection
  - Feature Extraction
  - Classification



# Object Detection | Traditional Approach

Step 1: Informative Region Selection

- Try to find the object's location
- Objects have different sizes and aspect ratios
- Object might appear at different locations in the images
- This is why we scan the whole image using a multiscale sliding window
- Note: This method is computationally expensive and produces many irrelevant candidates

# Object Detection | Traditional Approach

Step 2: Feature Extraction

- Using techniques like SIFT, and HOG to extract the visual feature for recognizing the object
- These visual features provide a semantic and robust representation
- Note: It is very difficult to manually design a robust feature descriptor to perfectly describe all types of objects due to the differences in:
    - Illumination conditions
    - Viewpoint
    - Backgrounds

# Object Detection | Traditional Approach

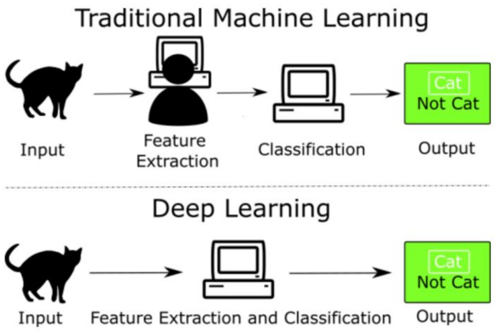
Step 3: Classification Stage

- Make the representations more hierarchical, semantic, and informative for visual recognition

- Make the classification of target objects from all other categories using:
  - Support Vector Machine (SVM)
  - Adaboost

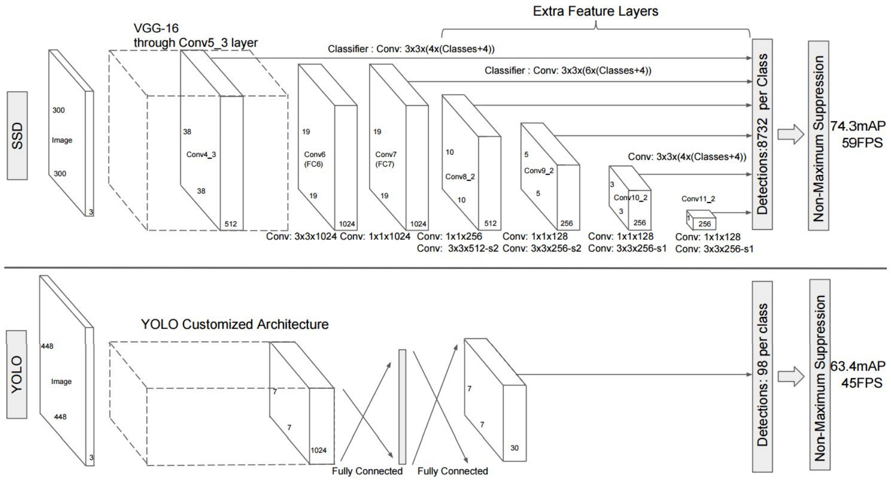
# Object Detection | Traditional Approach

- Problems
  - Generation of candidate bounding boxes using the sliding window technique is computationally expensive
  - Hand-engineered features are not always sufficient to perfectly describe all types of objects
- Solution?
  - Usage of modern approaches with Deep Learning
  - R-CNN
  - SPP-Net
  - Fast R-CNN
  - Faster R-CNN
  - YOLO
  - SSD



# Regression/Classification Based Frameworks

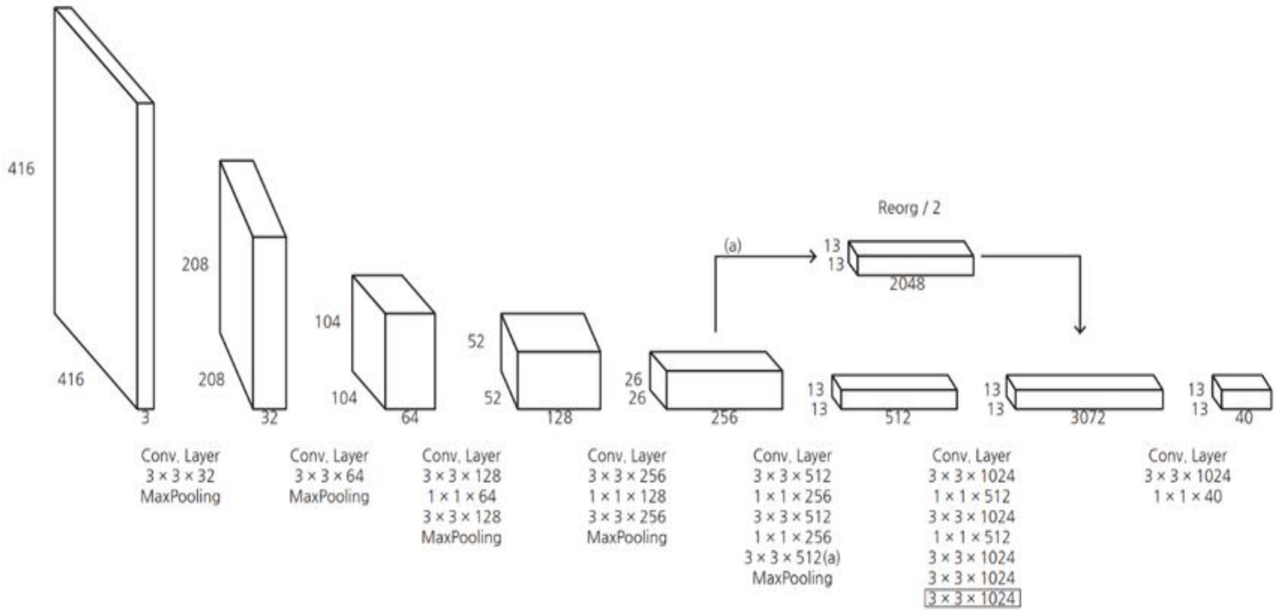
- Really good for real-time object detection
- One-Step frameworks based on global regression/classification maps straightly from image pixels to bounding box coordinates and class probabilities
- Reduce the time complexity
- Examples:
  - SSD
  - YOLO



# YOLO

- YOLO – You Only Look Once
- A single convolutional network predicts the bounding boxes and the class probabilities for these boxes
- Yolo divides the input image into an S x S grid and each grid cell is responsible for predicting the object centered in that grid cell
- Each grid cell predicts bounding boxes and their correspondence confidence scores
- Multiple versions of YOLO algorithms emerged lately, the latest are YOLOv4, YOLOv5 and YOLOv7

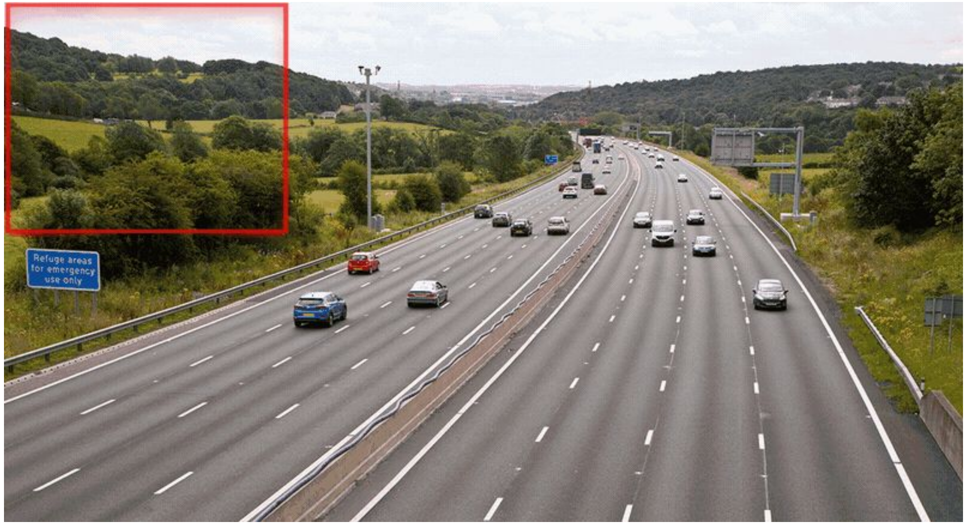
# YOLO architecture



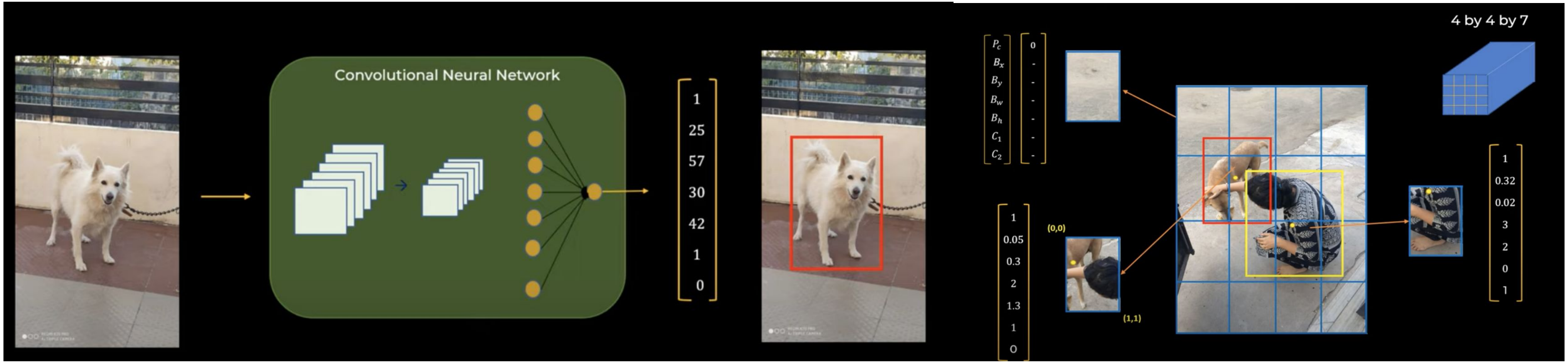
# YOLO - Limitations

- Sometimes has lower accuracy compared to R-CNN family of algorithms due to having a one-step object detection
- Difficulties in dealing with small objects in groups
- Difficulties in generalizing to objects in new/unusual aspect ratios or configurations
- Produces relatively coarse features due to multiple down-sampling operations

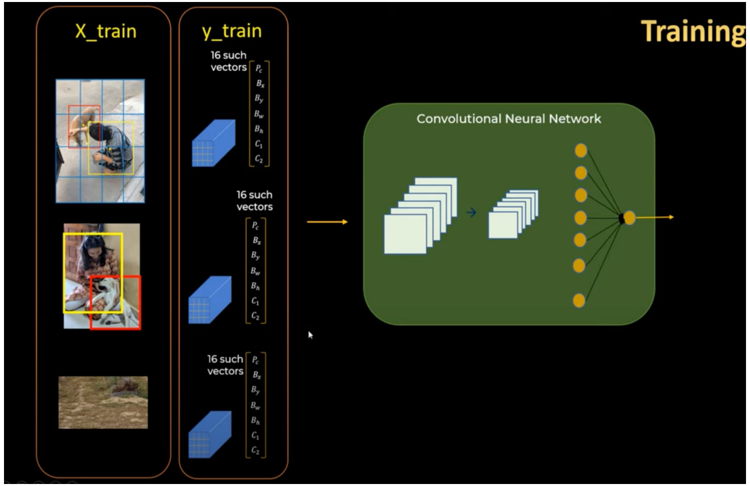
# SAHI: Slicing Aided Hyper Inference

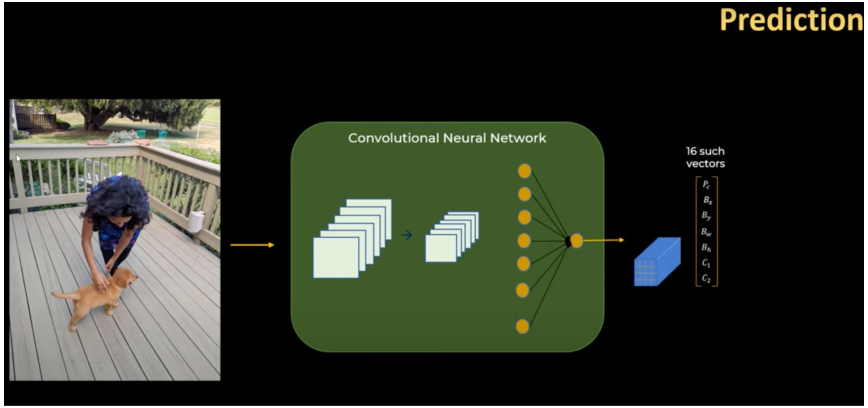



# YOLO

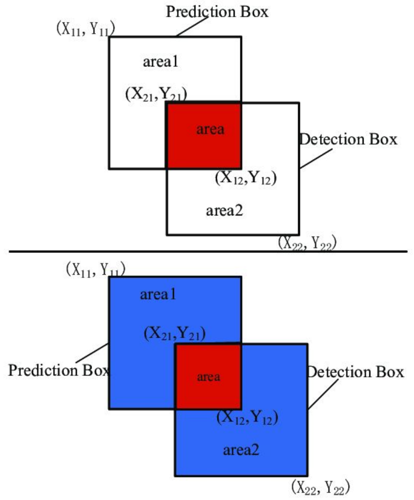

Convolutional Neural Network
1
25
57
30
42
1
0
4 by 4 by 7
$P_c$
$B_x$
$B_y$
$B_w$
$B_h$
$C_1$
$C_2$
0
(0,0)
(1,1)
1
0.05
0.3
2
1.3
1
0
1
0.32
0.02
3
2
0
1

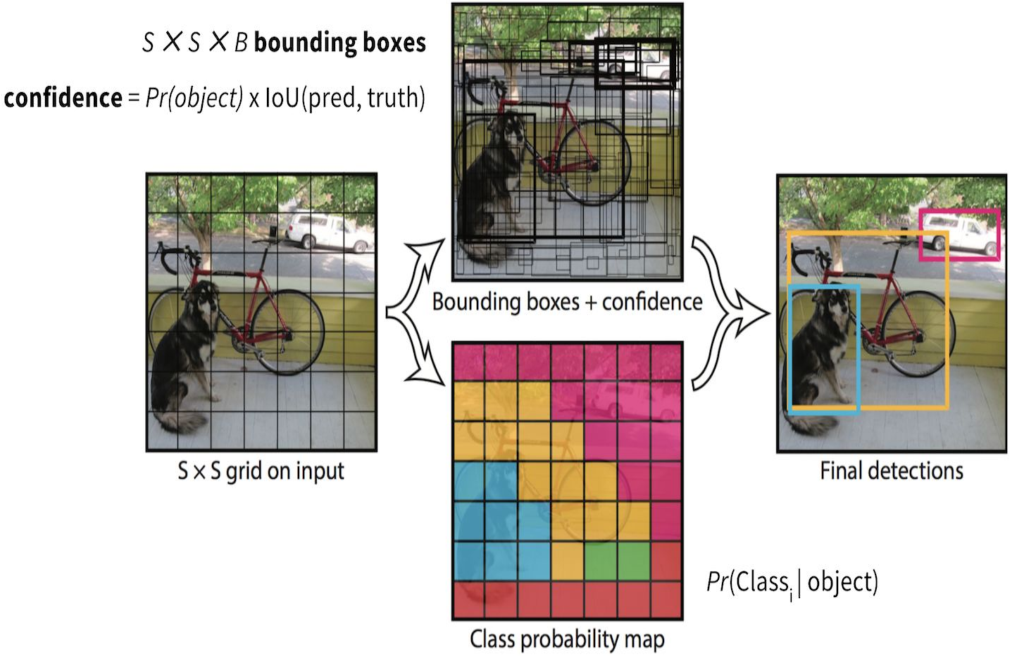
# Yolo


X_train
y_train
Training
16 such vectors
16 such vectors
16 such vectors
Convolutional Neural Network


Prediction
Convolutional Neural Network
16 such vectors

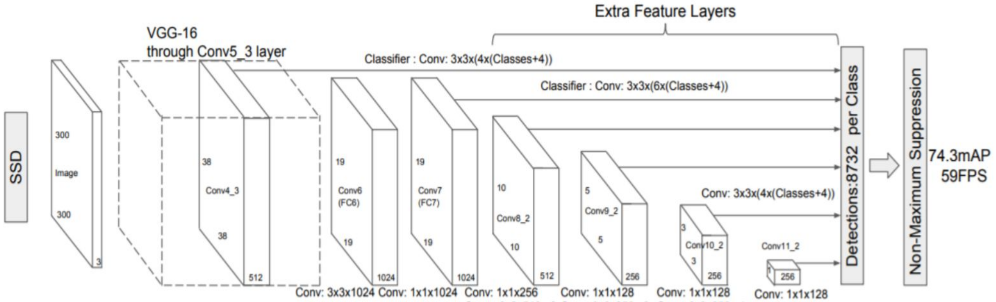
IOU =


Prediction Box
(X11, Y11)
area1
(X21,Y21)
area
Detection Box
(X12,Y12)
area2
(X22, Y22)
(X11, Y11)
area1
(X21,Y21)
Prediction Box
area
Detection Box
(X12,Y12)
area2
(X22, Y22)

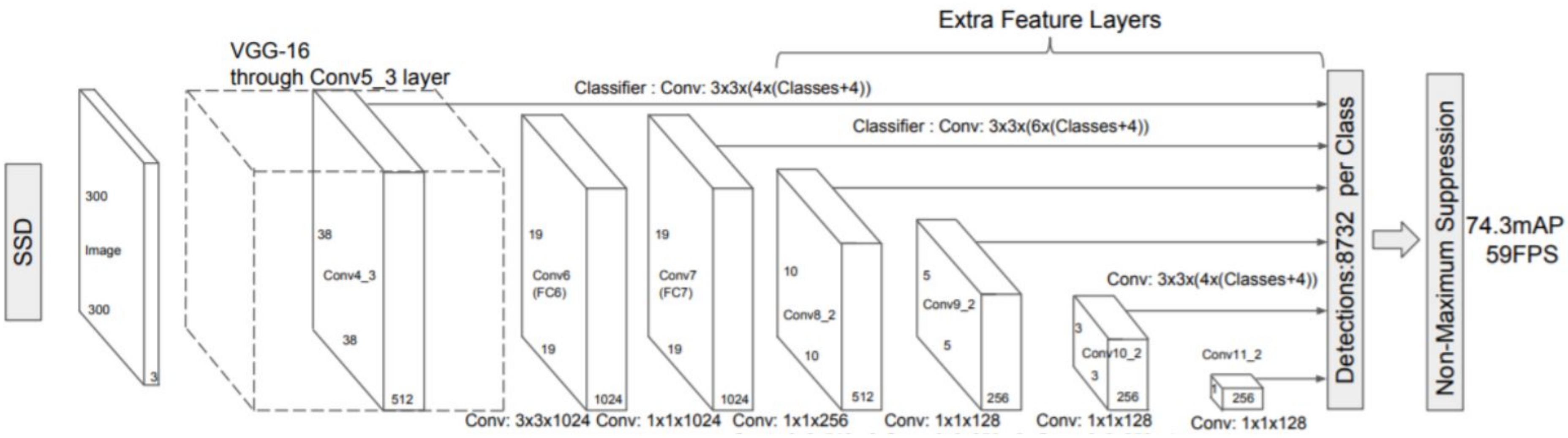

S X S X B bounding boxes
confidence = Pr(object) x IoU(pred, truth)
Bounding boxes + confidence
S × S grid on input
Final detections
Pr(Class_i | object)
Class probability map

# SSD

- SSD – Single Shot Detector
- Avoids some of the limitations of YOLO
- Uses a specific feature map instead of fixed grids
- Takes advantage of a set of default anchor boxes with different aspect ratios and scales to discretize the output space of bounding boxes
- The network fuses predictions from multiple feature maps with different resolutions to handle objects from various sizes
- Limitation: Accuracy



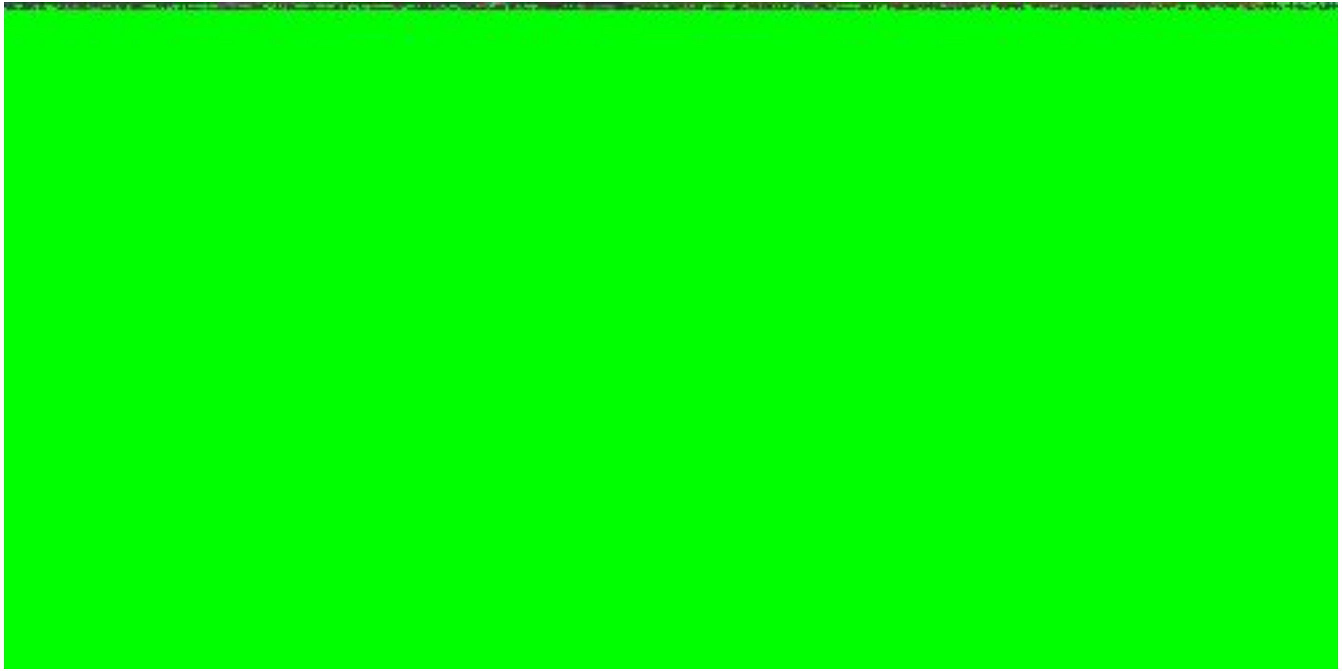
# SSD | Architecture



# What questions do you have?

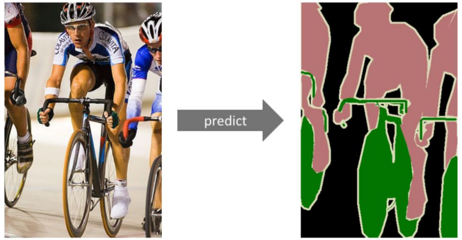
# Semantic Segmentation

# Semantic Segmentation



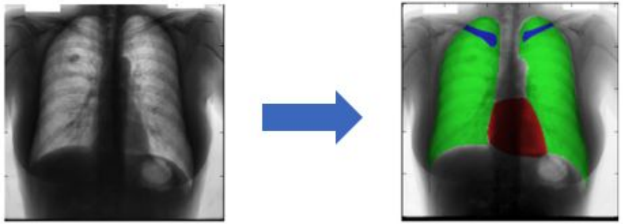
# Semantic Segmentation

- The goal is to label each pixel of an image with a corresponding class
- Because we're predicting for every pixel in the image, this task is commonly referred to as **dense prediction**
- Note:
  - We're not separating instances of the same class
  - We only care about the category of each pixel
  - If you have 2 objects of the same category in your input image, the segmentation map v



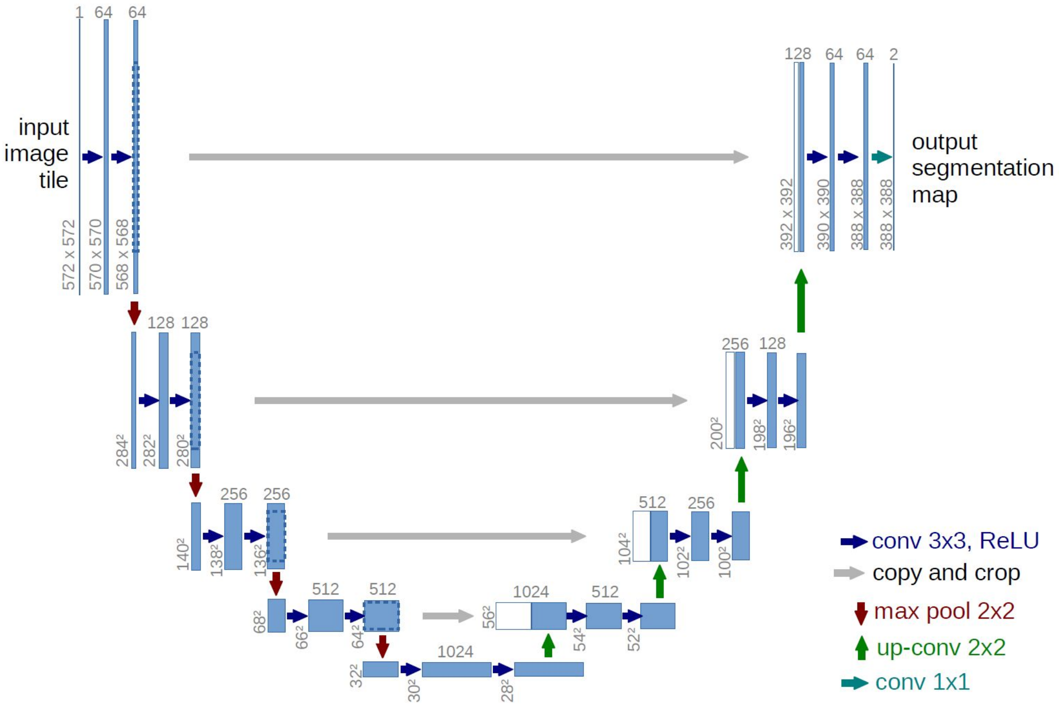

# Semantic Segmentation | Applications

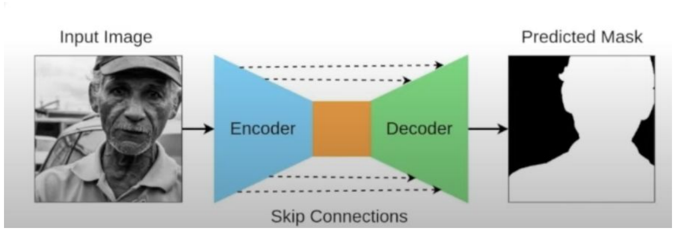
- Autonomous vehicles
- Biomedical Image Diagnosis
- Geo Sensing
- Precision Agriculture

Input Image    Segmented Image

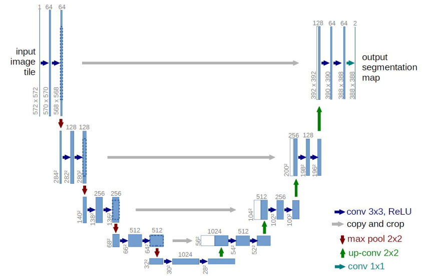
# Semantic Segmentation | UNET



# Semantic Segmentation | UNET

- The architecture contains 2 paths:




  - First Path: contraction path – **encoder**
    - Is used to capture the context in the image
    - A traditional stack of convolutional and max-pooling layers
  - Second Path: symmetric expanding path – **decoder**
    - Is used to enable precise localization using transposed convolutions
    - It is an end-to-end fully convolutional network (FCN)
    - It only contains Convolutional layers
    - It does not contain any Dense layer, which means that it can accept an image of any size

Must read article

# What questions do you have?

# Computer Vision Benchmarks

# Computer Vision Benchmark Datasets

- Good Computer Vision benchmark datasets will reflect the setting of the real-world application of the model you are developing.
- Examples of Datasets:
- CIFAR-10
- MS COCO
- Fashion-MNIST
- ImageNet
- IMDB-Wiki dataset
- Kinetics-700
- ObjectNet
- MPII Human Pose Dataset
- Open Images
- Cityscapes
- The 20BN-something something Dataset V2
- KITTI
- Waymo OD
- nuScenes

# Having the right Dataset Benchmark

- The first and most important question to ask while working for computer vision is “how can you identify the right dataset benchmark”?

- Many publicly available real-world and simulated benchmark datasets have emerged lately

- Problems we’re facing:
    - The organization and adoption as standards between the sources are inconsistent
    - Many existing benchmarks lack diversity to benchmark computer vision algorithms effectively

# Good Benchmarks

- Good benchmark datasets allow you to evaluate several machine learning methods in a direct and fair comparison
- A common problem with these benchmarks is that they are not an accurate depiction of the real world
- Methods ranking high on popular computer vision benchmarks could perform low on average when tested outside the data they were created with

# How to spot a Bad Benchmark Dataset?

- Contains mainly images that were taken in ideal / perfect / unrealistic conditions

- Inadequate at handling the messiness found in the real world

- Example: **ImageNet**
  - Although it is a very popular dataset for computer vision, the images do not adequately represent reality, and thus, ImageNet is not the best computer vision benchmark.

# What type of dataset benchmarks exist?

- **Best dataset benchmarks for segmentation**
  - The Berkeley Segmentation Dataset and Benchmark (link)
  - KITTI semantic segmentation benchmark (link). Check out the Hub equivalent for the test, train, and validation KITTI datasets.
- **Best dataset benchmarks for classification**
  - ObjectNet Benchmark Image Classification (link)
- **Best dataset benchmarks for scene understanding**
  - Scene Understanding on ADE20K val (link)
  - Scene Understanding on Semantic Scene Understanding Challenge Passive Actuation & Ground-truth Localisation (link)

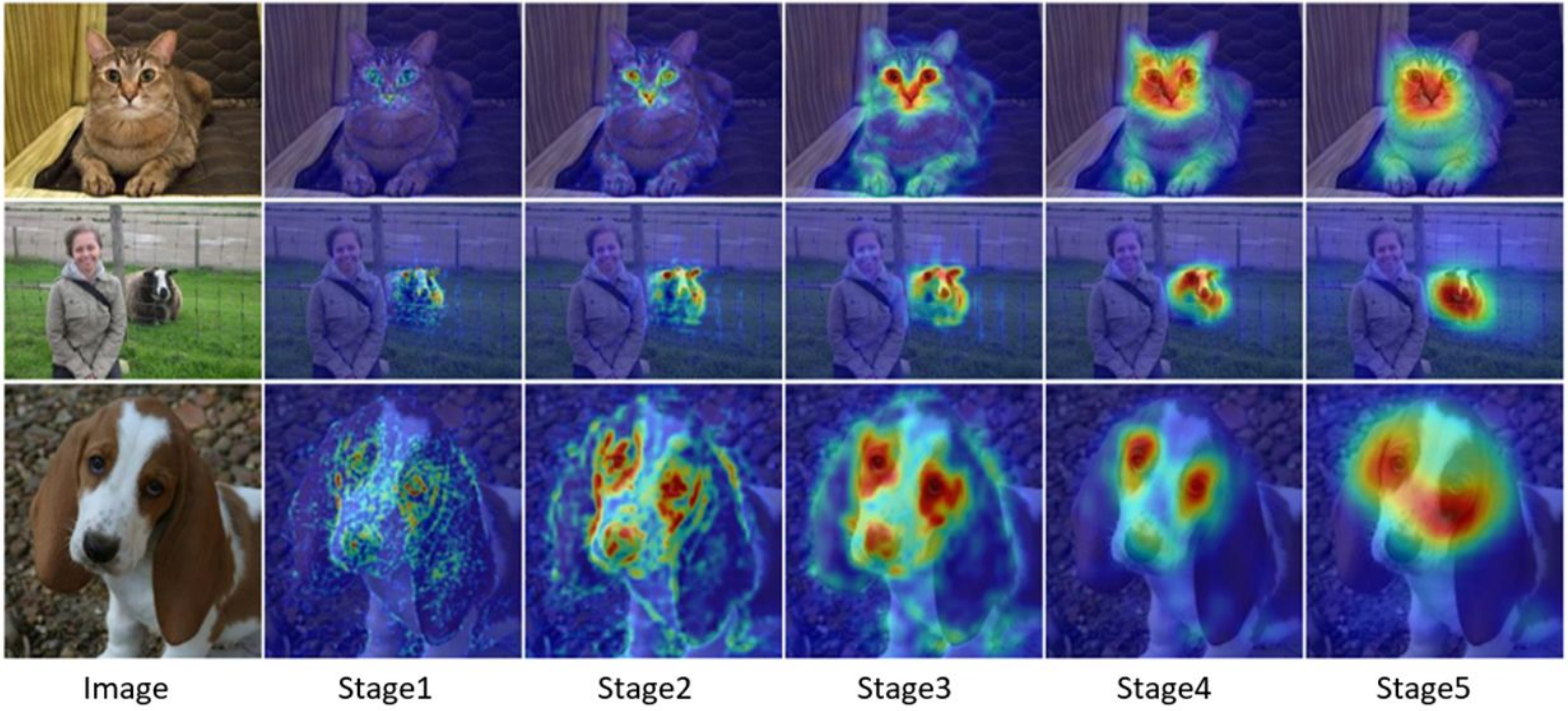
What questions do you have?

# Explainability and Saliency

# Explainability

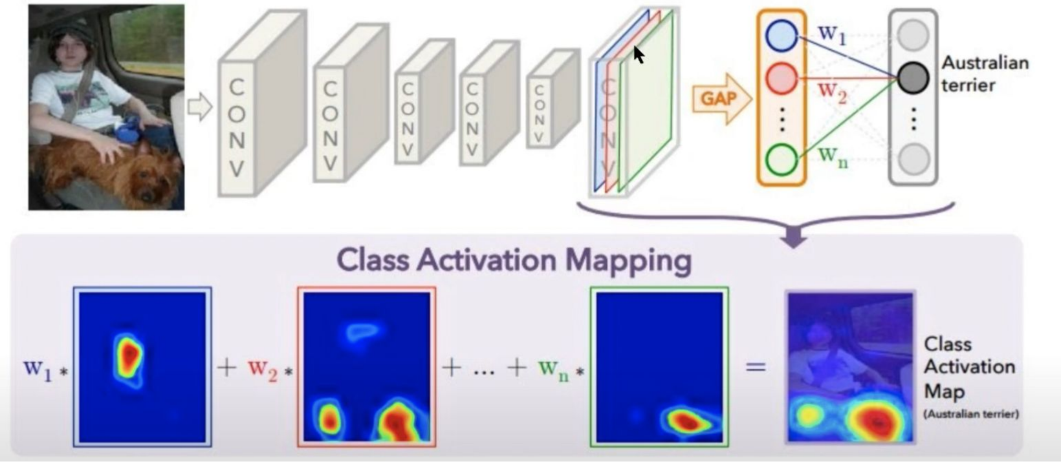
- Often referred to as “Interpretability”
- It is the concept that a Machine Learning Model and its output can be explained in a way that “makes sense” to a human being at an acceptable level
- Certain classes of algorithms (like traditional ML algorithms) tend to be more readily explainable, with potential less performance
- Deep Learning systems, are more performant, but are also much harder to explain

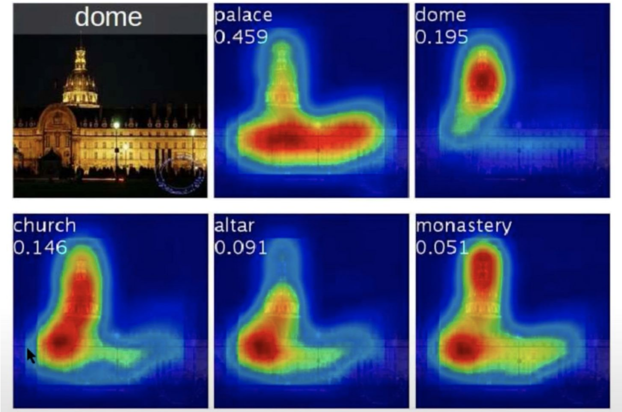
# Class Activation Map



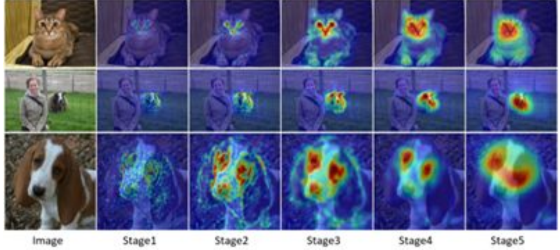
# Class Activation Map Intuition

What led to the positive feature?



# Class Activation Map

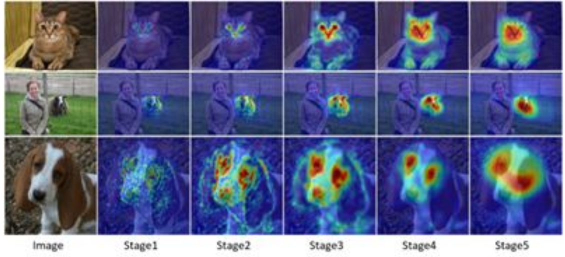



- Class Activation Mapping (CAM) indicates the discriminative region of the image for a particular class (category), which influenced the Deep Learning model to make the Decision

- The architecture is very similar to a convolutional neural network

- It comprises several convolution layers, with the layer just before the final output performing Global Average Pooling

# Class Activation Map

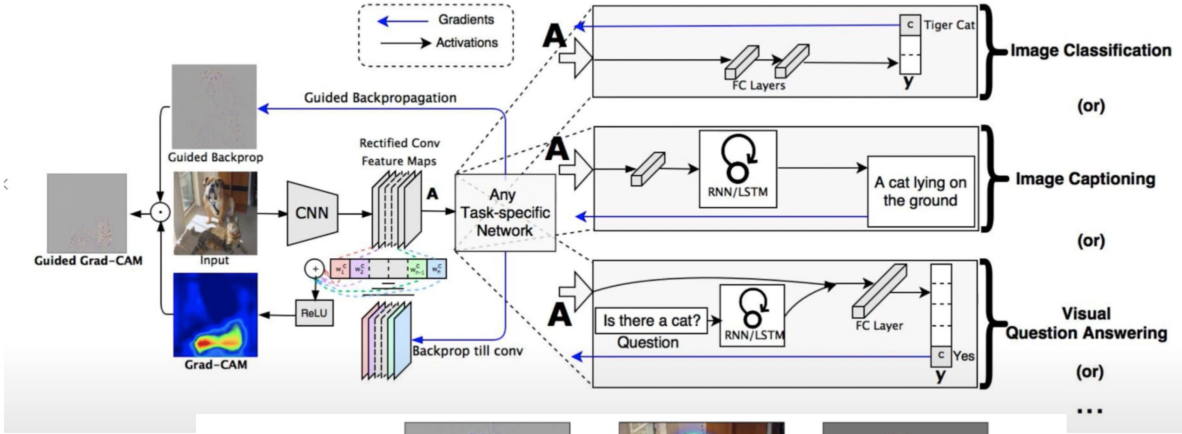


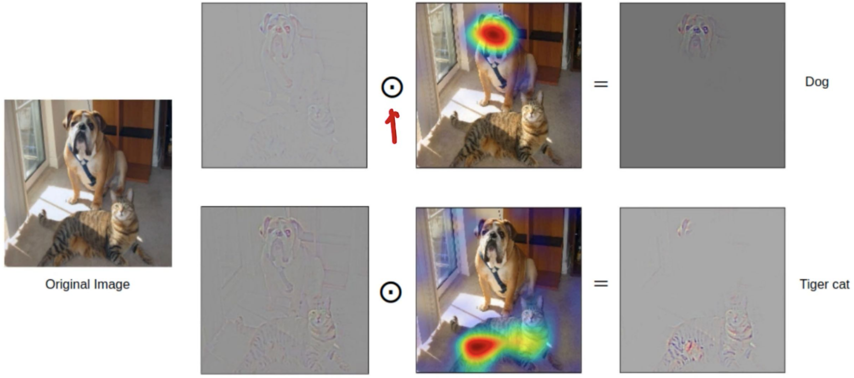

- The features that are obtained are fed into the Fully Connected Neural Network layer governed by the SoftMax activation function

- Then we get the output required probabilities

- The importance of the weights with respect to a category can be found by projecting back the weights onto the last convolution layer's feature map

# Grad CAM


Gradients
Activations
Guided Backpropagation
Guided Backprop
Rectified Conv
Feature Maps
Guided Grad-CAM
Input
CNN
A
Any
Task-specific
Network
ReLU
Grad-CAM
Backprop till conv
Tiger Cat
FC Layers
y
Image Classification
(or)
RNN/LSTM
A cat lying on
the ground
Image Captioning
(or)
Is there a cat?
Question
RNN/LSTM
FC Layer
Yes
y
Visual
Question Answering
(or)
...


Dog
Tiger cat
Original Image
Guided Backprop
(fine-grained details)
Grad-CAM
(Class discriminative)
Guided Grad-CAM
(Fine-grained +Class discriminative)

# Saliency | History

- Saliency maps in Deep learning were first witnessed in the paper titled *“Deep Inside Convolutional Networks: Visualizing Image Classification Models and Saliency Maps”*

- The paper was presented by researchers of the Visual Geometry Group at the University of Oxford

- It highlighted the visualization techniques to compute images, saliency maps being one of them

# Saliency Maps

- This method is derived from the concept of saliency in images
- **Saliency** refers to unique features (pixels, resolution, etc.) of the image in the context of visual processing
- These unique features depict the visually alluring locations in an image
- Saliency maps are topographical representations of them.

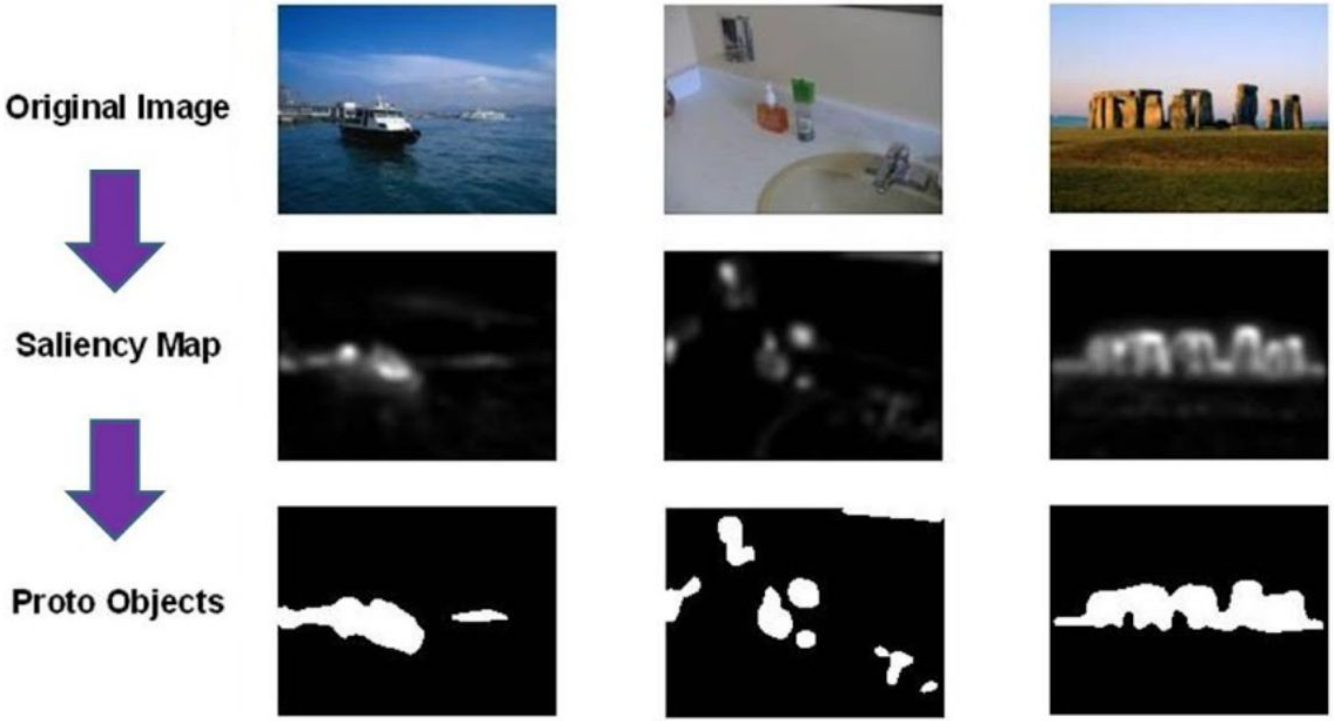
# Saliency

Vanilla Saliency Method

$$Y_c = score\ of\ class\ c$$

$$saliency = \max_{r,g,b}\left(\left|\frac{\partial Y_c}{\partial I}\right|\right)$$

# How to create Saliency Maps? | Example



# What questions do you have?

# Feedback on Lecture and Concepts?

# See you next week!