OPEN DATA SCIENCE CONFERENCE

@ODSC

Boston | May 1 - 4 2018

POWER UP YOUR COMPUTER VISION SKILLS WITH TENSORFLOW-KERAS

Mo Patel @mopatel

ODSC East - Boston May 2018

http://bit.ly/odsckerascv

Keras & TensorFlow

TensorFlow & Keras

- TensorFlow: A system for movement of Tensors
- A general purpose framework for machine learning
- Intermediate to Advanced Users
- Extensive Ecosystem
- Large Community

- Keras: “It's a play on the words κέρας (horn) / κραίνω (fulfill), and ἐλέφας (ivory) / ἐλεφαίρομαι (deceive).”
- Best for beginners and prototyping
- Simple API that works with multiple backends
- TensorFlow now primary backend

Getting started with Keras

Deep Learning Software Quick-Start

- Get Anaconda Python Distribution
  - https://www.anaconda.com/download/
- Install Keras*
  - Open Anaconda Navigator
  - Go to Environments
  - Click on Channels, Click on Add, Add conda-forge and click on Update Channels
  - Search for TensorFlow and check the box in Name column
  - Search for Keras and check the box in Name column
  - Click Apply
- Launch Jupyter Notebook
  - Open Anaconda Navigator
  - From Home, click on Launch button for Jupyter Notebook

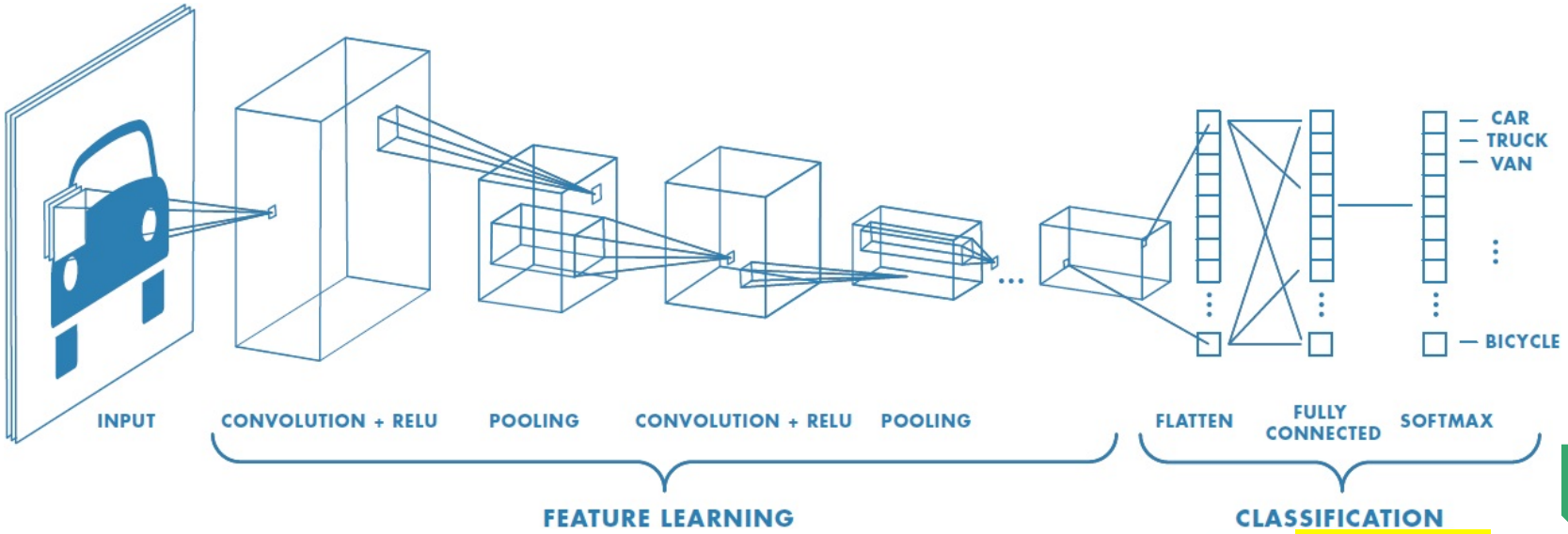
Simple:

```
python3 -m pip install --upgrade pip
pip install keras
python3 -m pip install jupyter
```

*your mileage may vary

Anatomy of a Keras Model

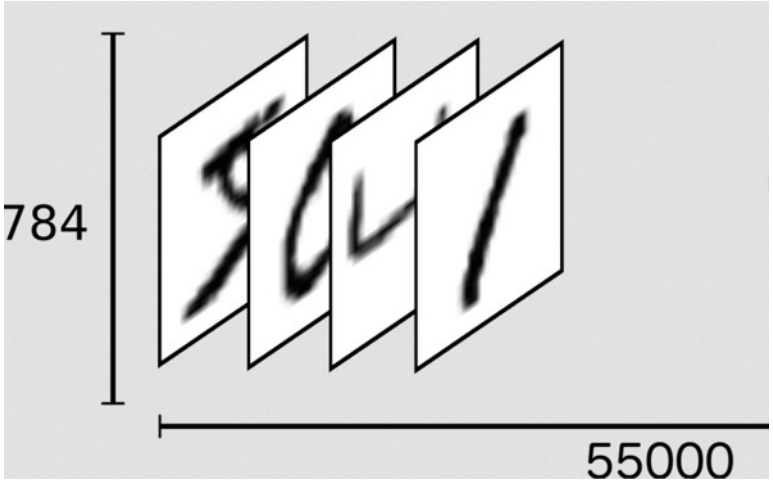
Image Classification



Source: Mathworks

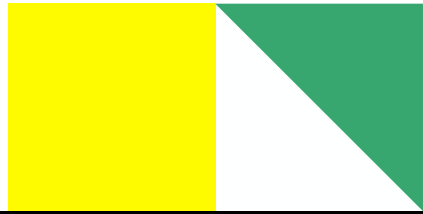
Learning by example: MNIST Handwritten Digit Classifcation

- Ten classes
  - 0,1,2,3,4,5,6,7,8,9
- Dataset
  - 55,000 28x28 Black and White images for training
  - 10,000 28x28 Black and White images for testing



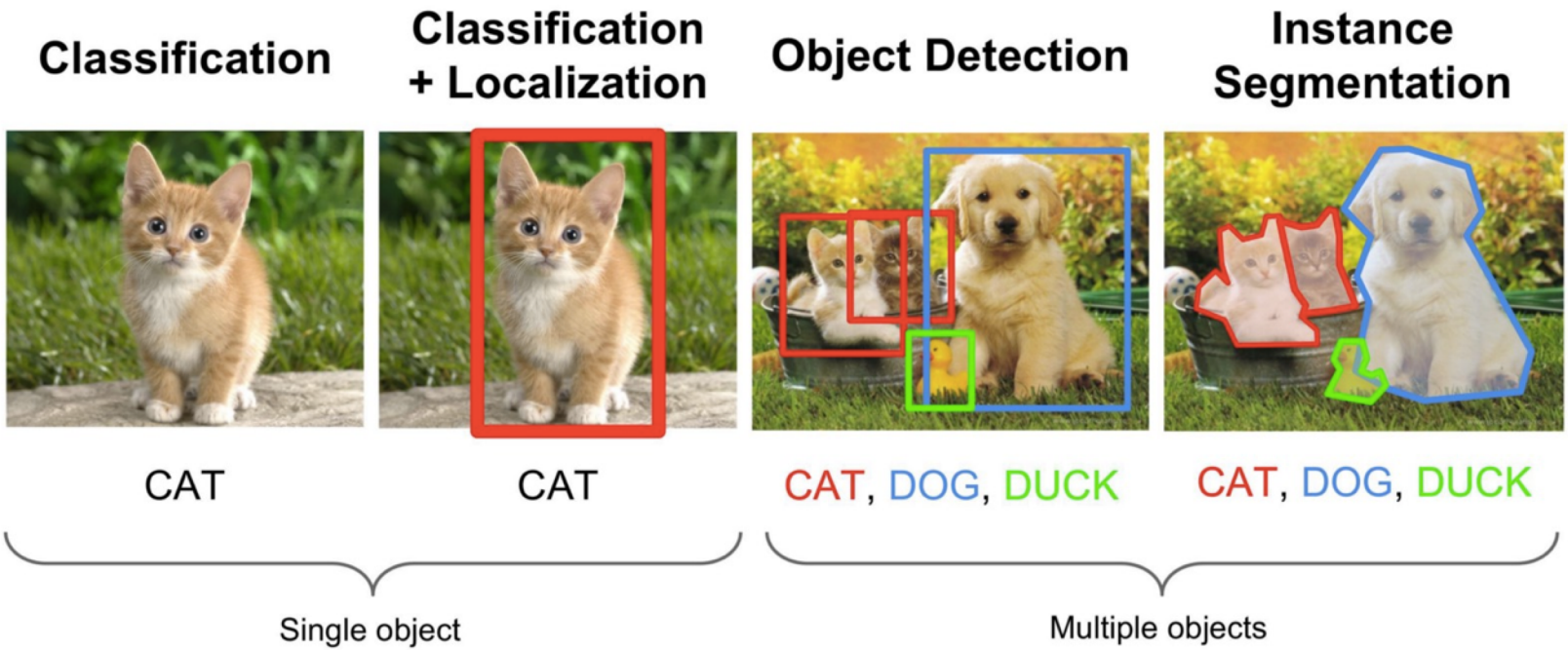
Anatomy of a Keras Model

- Using Jupyter Notebook, please open *Keras_MNIST.ipynb* from the tutorial repository on Github.



Computer Vision Models

Object Detection & Segmentation

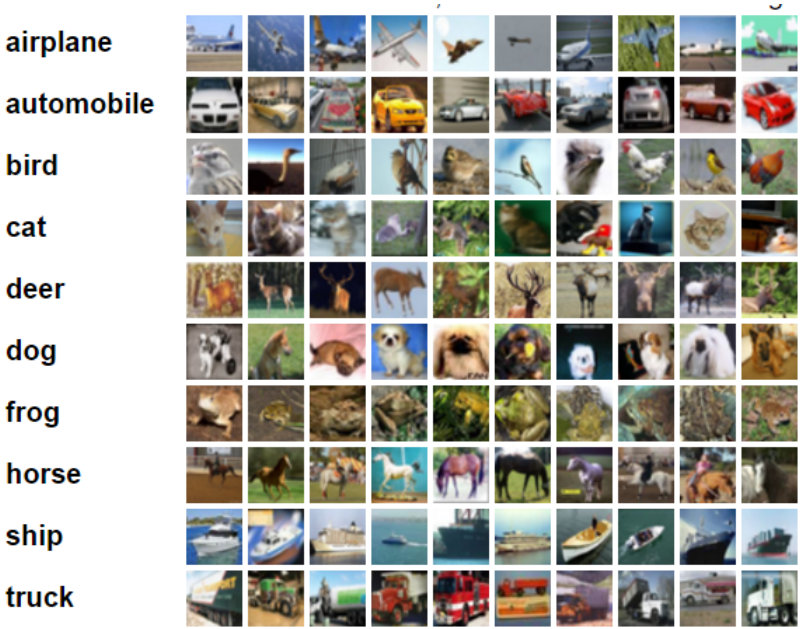


https://medium.com/comet-app/review-of-deep-learning-algorithms-for-object-detection-c1f3d437b852

Image Classification

PyTorch CIFAR10 Tutorial

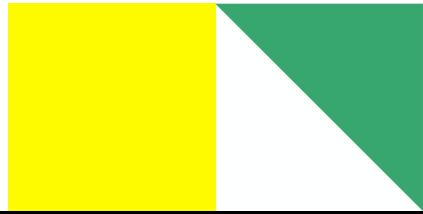
- Ten classes
- Dataset:
  - 60,000 32x32 color images in 10 classes
  - 6000 images per class.
  - 50,000 training images
  - 10,000 test images



https://www.cs.toronto.edu/~kriz/cifar.html

PyTorch CIFAR10 Tutorial

- Using Jupyter Notebook, please open *Keras_CIFAR10_Walk_Through.ipynb* from the tutorial repository on Github.



Transfer Learning

Why Transfer Learning?

Deep Learning Challenges

- Not enough data
- Not enough labeled data
- Not enough hardware
- Training time too long

Transfer Learning

Domain to Domain Learning

- Learn Features, Weights, Biases in one domain
- Find domain with similar data characteristics and distribution
- Apply model in new domain

Retraining

- Freeze layers in a "good" model up to tunable depth of feature extraction but before final prediction layer
- Remove final layer(s)
- Relearn final and classification layers using your own data

Bonus: Combine Domain to Domain and Retraining

http://bit.ly/odsckerascv

Freezing Layers in Keras

Existing Model

```
model = Sequential()
model.add(Conv2D(32, (3, 3), padding='same',
input_shape=x_train.shape[1:]))
model.add(Activation('relu'))
model.add(Conv2D(32, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Conv2D(64, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(512))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(num_classes))
model.add(Activation('softmax'))
opt = keras.optimizers.rmsprop(lr=0.0001, decay=1e-6)
model.compile(loss='categorical_crossentropy',
optimizer=opt,metrics=['accuracy'])
```

Retraining Model

```
model = Sequential()
model.add(Conv2D(32, (3, 3), padding='same',
input_shape=x_train.shape[1:]),trainable=False)
model.add(Activation('relu'),trainable=False)
model.add(Conv2D(32, (3, 3)),trainable=False)
model.add(Activation('relu'),trainable=False)
model.add(MaxPooling2D(pool_size=(2, 2)) ,trainable=False)
model.add(Dropout(0.25))
model.add(Conv2D(64, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(512))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(num_classes))
model.add(Activation('softmax'))
opt = keras.optimizers.rmsprop(lr=0.0001, decay=1e-6)
model.compile(loss='categorical_crossentropy',
optimizer=opt,metrics=['accuracy'])
```

Object Detection & Segmentation

Single Shot Multi-box Detector

- Keras: SSD_tutorial.ipynb

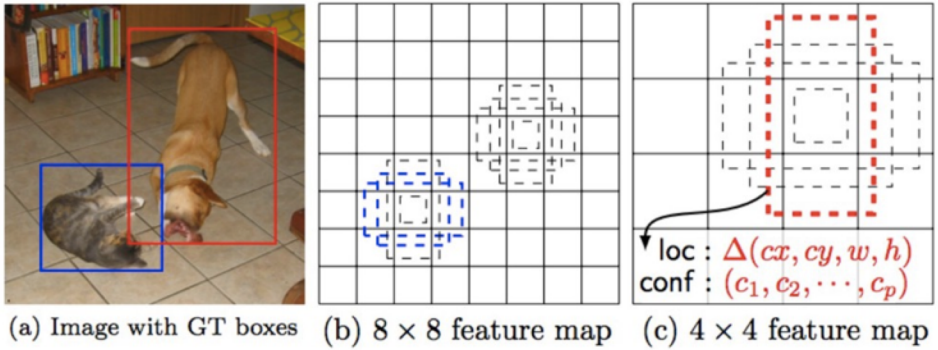


Fig. 1: **SSD framework.** (a) SSD only needs an input image and ground truth boxes for each object during training. In a convolutional fashion, we evaluate a small set (e.g. 4) of default boxes of different aspect ratios at each location in several feature maps with different scales (e.g. 8×8 and 4×4 in (b) and (c)). For each default box, we predict both the shape offsets and the confidences for all object categories ($(c_1, c_2, \cdots, c_p)$). At training time, we first match these default boxes to the ground truth boxes. For example, we have matched two default boxes with the cat and one with the dog, which are treated as positives and the rest as negatives. The model loss is a weighted sum between localization loss (e.g. Smooth L1 [6]) and confidence loss (e.g. Softmax).

Paper: https://arxiv.org/pdf/1512.02325v5.pdf

YOLO: You Only Look Once

- Keras: https://github.com/xiaochus/YOLOv3

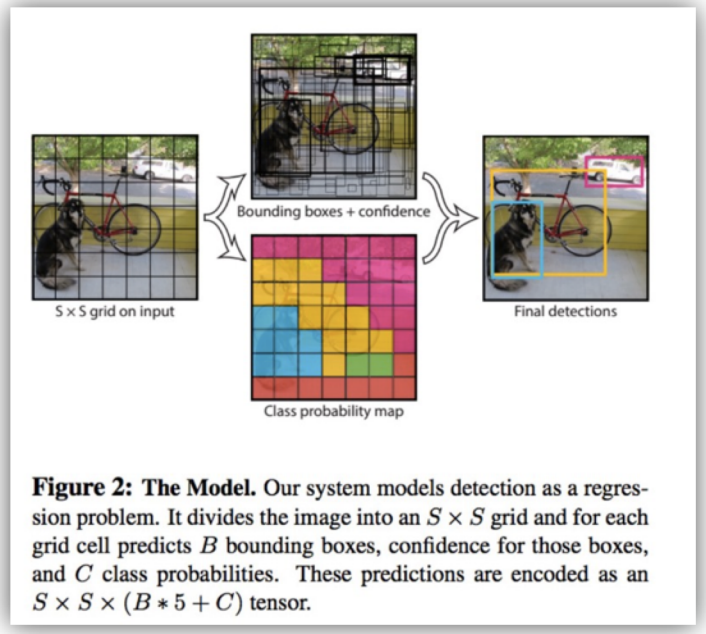
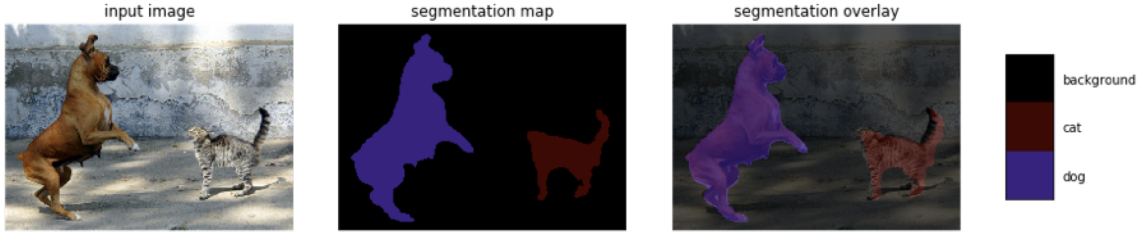


Figure 2: The Model. Our system models detection as a regression problem. It divides the image into an $S \times S$ grid and for each grid cell predicts B bounding boxes, confidence for those boxes, and C class probabilities. These predictions are encoded as an $S \times S \times (B * 5 + C)$ tensor.

Image Segmentation in Keras & TensorFlow

- Keras – DeepLab: https://github.com/bonlime/keras-deeplab-v3-plus
- TensorFlow: https://research.googleblog.com/2018/03/semantic-image-segmentation-with.html



Keras - DeepLab

Working with Computer Vision Datasets

Working with video

Extract Frames and Compose Video using FFMPEG
https://www.ffmpeg.org/

Extract

```
ffmpeg -i <video.file> image%03d.jpg
```

Compose

```
ffmpeg -framerate 24 -i img%03d.jpg <output_video.file>
```

Data Augmentation

- Data Augmentation in Keras
  - keras.preprocessing.image.ImageDataGenerator

```
featurewise_center
samplewise_center
featurewise_std_normalization
samplewise_std_normalization
zca_whitening
zca_epsilon
rotation_range
width_shift_range
height_shift_range
```

```
shear_range
zoom_range
channel_shift_range
fill_mode
cval
horizontal_flip
vertical_flip
rescale
```

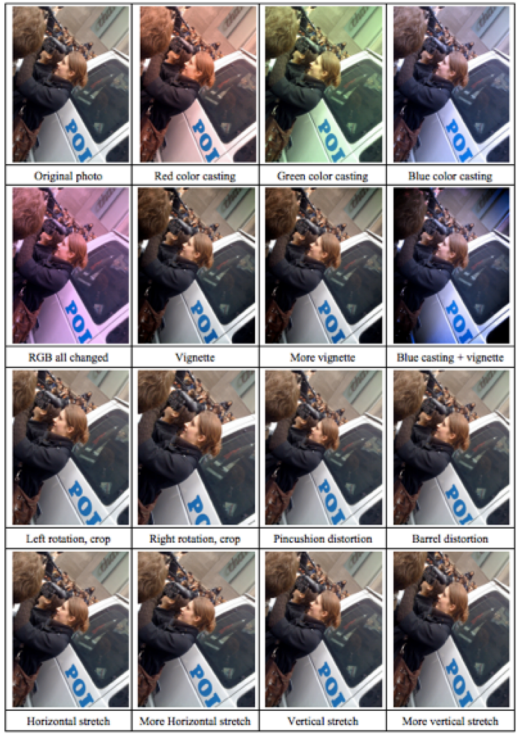


Image Source: Deep Image: Scaling up Image Recognition (Baidu)
https://arxiv.org/abs/1501.02876

Data Annotation


```
{"coco_url":"http://mscoco.org/images/405628",
"date_captured":"2013-11-21 00:26:09",
"file_name":"COCO_train2014_000000405628.jpg",
"flickr_url":"http://farm5.staticflickr.com/4130/5133418813_61f33c
9633_z.jpg",
"height":426,
"id":405628,
"license":5,
"width":640}
```

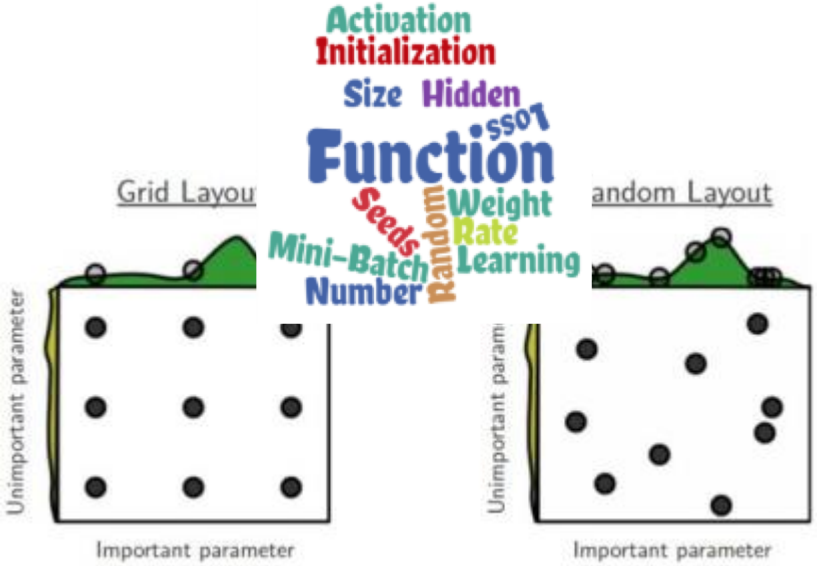
```
{"area":7581.427150000003,
"bbox":[95.28,146.43,131.92,142.0],
"category_id":1,"id":188293,
"image_id":405628,
"iscrowd":0,
"segmentation":[[219.87,177.58,224.45,172.08,225.37,157.42,218.95
,146.43,208.88,149.18,208.88,149.18,202.46,151.92,198.8,153.76,19
8.8,153.76,196.97,154.67,184.14,153.76,183.23,153.76,158.49,157.4
2,133.75,157.42,95.28,186.74,105.35,183.99,153.91,164.75,169.48,1
74.83,169.48,197.73,135.59,231.63,131.92,235.29,133.75,247.2,152.
99,247.2,155.74,247.2,175.9,241.71,179.56,228.88,184.14,278.35,19
5.14,280.18,194.22,288.43,200.63,282.93,198.8,271.02,224.45,227.0
5,224.45,227.05,227.2,196.82,219.87,183.99]]},
```


Data Annotation

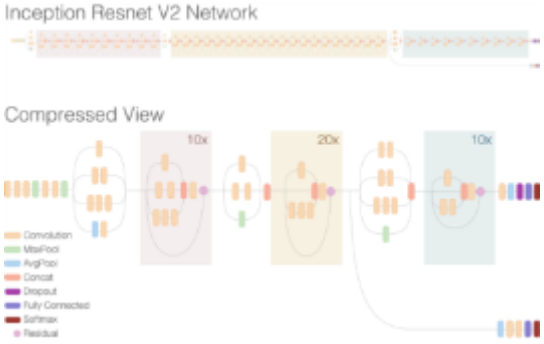
	BOUNDING BOX	DOTS	POLYGON	SEMANTIC SEGMENTATION
COST	Least expensive	More expensive	More expensive	Most expensive
TIME COMMITMENT	Lowest	Medium	Medium	Highest
PRECISION	Good	Great	Great	Excellent
INSTANCE-BASED (OUTPUT CONTAINS DISCRETE OBJECTS)	Yes	Yes	Yes	No
POSSIBLE TO LABEL SEVERAL OBJECTS?	Yes	Yes	Yes	Yes
POSSIBLE TO LABEL SEVERAL CLASSES OF OBJECT IN A SINGLE JOB?	No	No	No	Yes
OUTPUT*	X,Y coordinate, width and length of each box	Series of x,y coordinates	Series of x,y coordinates, with shapes resolving	Coded RGB pixels as an image

Source: CrowdFlower WHAT WE LEARNED LABELING 1 MILLION IMAGES

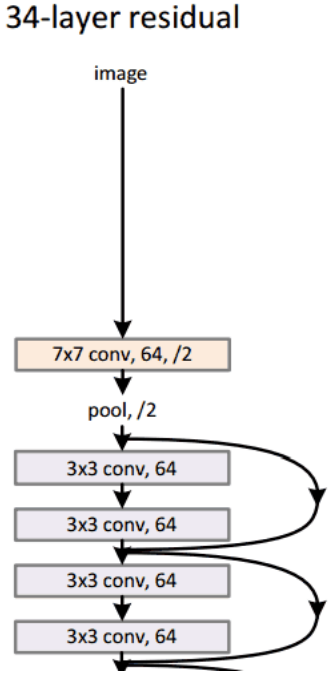
Training Challenges



Hyper-Parameter Optimization



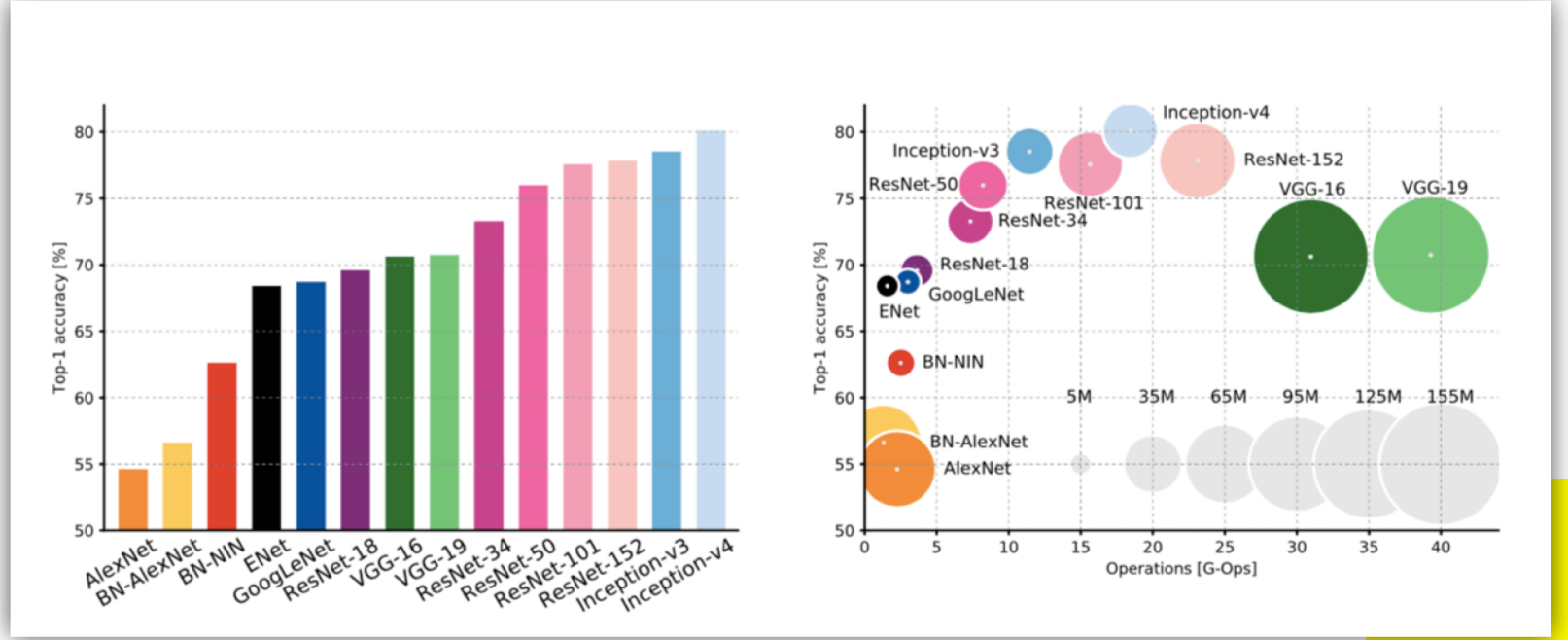
Google Inception v3



Microsoft Resnet

https://adeshpande3.github.io/adeshpande3.github.io/The-9-Deep-Learning-Papers-You-Need-To-Know-About.html
http://colinraffel.com/wiki/neural_network_hyperparameters

Analysis from training Computer Vision Models



An Analysis of Deep Neural Network Models for Practical Applications https://arxiv.org/abs/1605.07678

Training Computer Vision Models: Time & Cost

Batch Size	epochs	Top-1 Accuracy	hardware	cost ($)	time
256	100	58.7%	8-core CPU + K20 GPU	3,000	144h
512	100	58.8%	1 DGX station	129,000	6h 10m
4096	100	58.4%	1 DGX station	129,000	2h 19m
32K	100	58.5%	512 KNLs	1.2 million	24m

Table 7: The speed and hardware cost for training AlexNet.
For batch size=32K, we changed local response norm in AlexNet to batch norm.

Batch Size	epochs	Top-1 Accuracy	hardware	cost ($)	time
256	90	73.0%	1 DGX station	129,000	21h
8192	90	72.7%	1 DGX station	129,000	21h
8192	90	72.7%	32 DGX stations	4.1 million	1h
32K	90	72.4%	512 KNLs	1.2 million	1h

Table 8: The speed and hardware cost for training ResNet50. We did not use data augmentation.

ImageNet Training in 24 Minutes https://arxiv.org/abs/1709.05011

Deploying Keras Models

Deploying Keras Models

- Prototype using Keras Predict API: Building a simple Keras + deep learning REST API
  - `output = model.predict(input)`
  - https://blog.keras.io/#exporting-a-model-with-tensorflow-serving
- Production: TensorFlow* Serving
  - `estimator_model = keras.estimator.model_to_estimator(keras_model=model)`
  - Converting a Keras model to a TensorFlow Estimator: https://cloud.google.com/blog/big-data/2017/12/new-in-tensorflow-14-converting-a-keras-model-to-a-tensorflow-estimator
  - https://github.com/tensorflow/workshops/blob/master/extras/keras-to-estimator/keras-to-estimator.ipynb

*TensorFlow 1.4 or higher

Computer Vision *Next...*

New ideas in Deep Learning Computer Vision

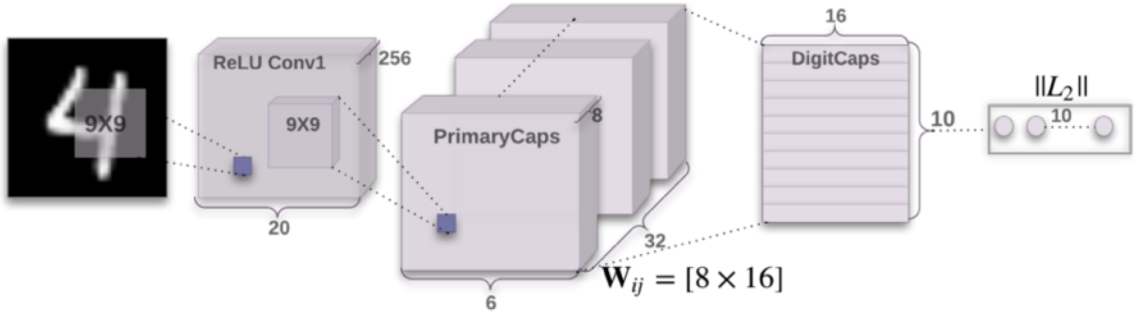
- Generative Adversarial Networks (GANs) https://arxiv.org/abs/1406.2661

https://github.com/eriklindernoren/Keras-GAN

- Capsule Networks

  Dynamic Routing Between Capsules (https://arxiv.org/abs/1710.09829)

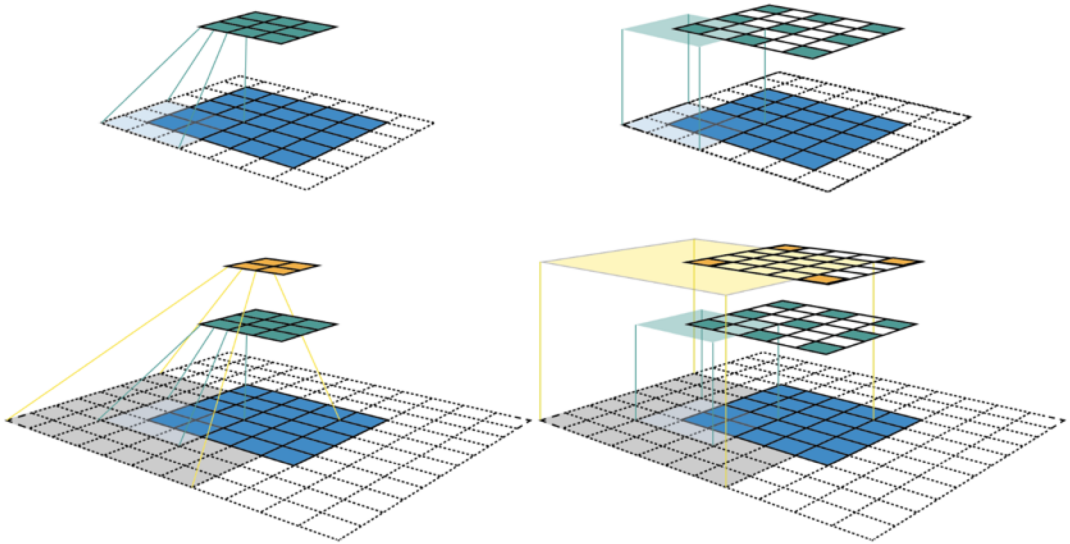
  https://github.com/XifengGuo/CapsNet-Keras



https://medium.com/ai%C2%B3-theory-practice-business/understanding-hintons-capsule-networks-part-i-intuition-b4b559d1159b

Features and Receptive Field

- A guide to receptive field arithmetic for Convolutional Neural Networks



https://medium.com/mlreview/a-guide-to-receptive-field-arithmetic-for-convolutional-neural-networks-e0f514068807

POWER UP YOUR COMPUTER VISION SKILLS WITH TENSORFLOW-KERAS

DISCUSSION