DEEP LEARNING FOR OBJECT DETECTION: A VISION OF THE FUTURE

Presented by: ANEEQ UR REHMAN & THOMAS ROOKS
Date: 09/09/2020

softwaresolved.com

Software SOLVED

Innovate UK
Knowledge Transfer Network

UNIVERSITY OF PLYMOUTH

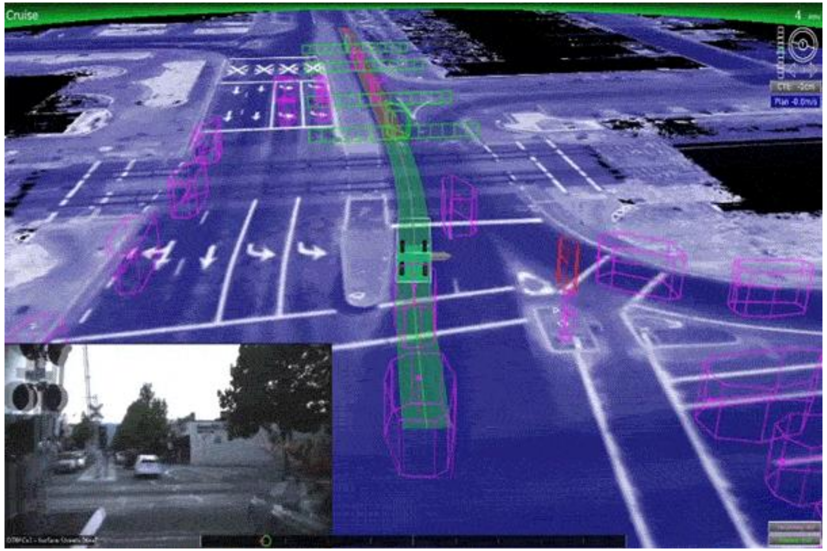
Gold
Microsoft Partner
Microsoft

bsi. ISO 9001 Quality Management FS 542819
ISO/IEC 27001 Information Security Management IS 713601

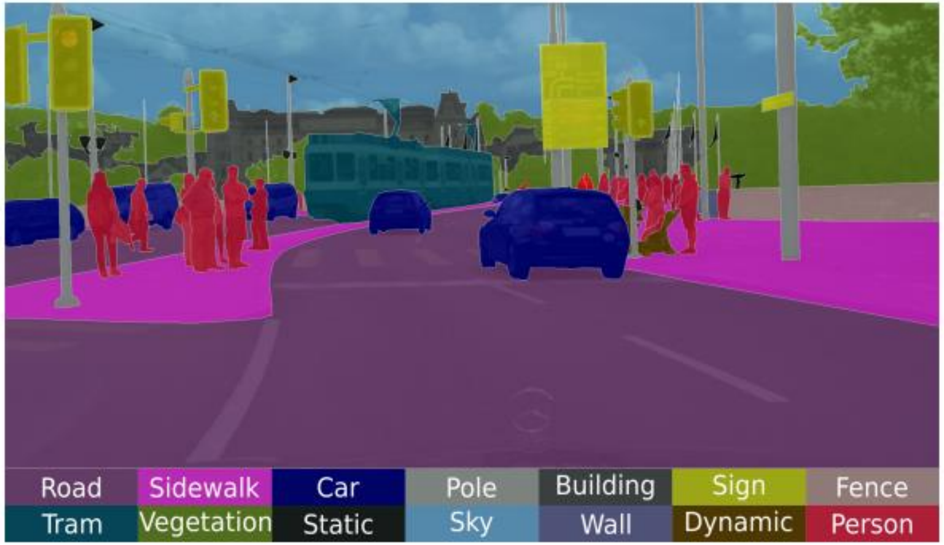
Computer Vision

- Study of designing algorithms that enables artificial systems (computers) to interpret visual data.
- Massively widespread.
- Lots of applications!

Object Detection and Semantic Segmentation

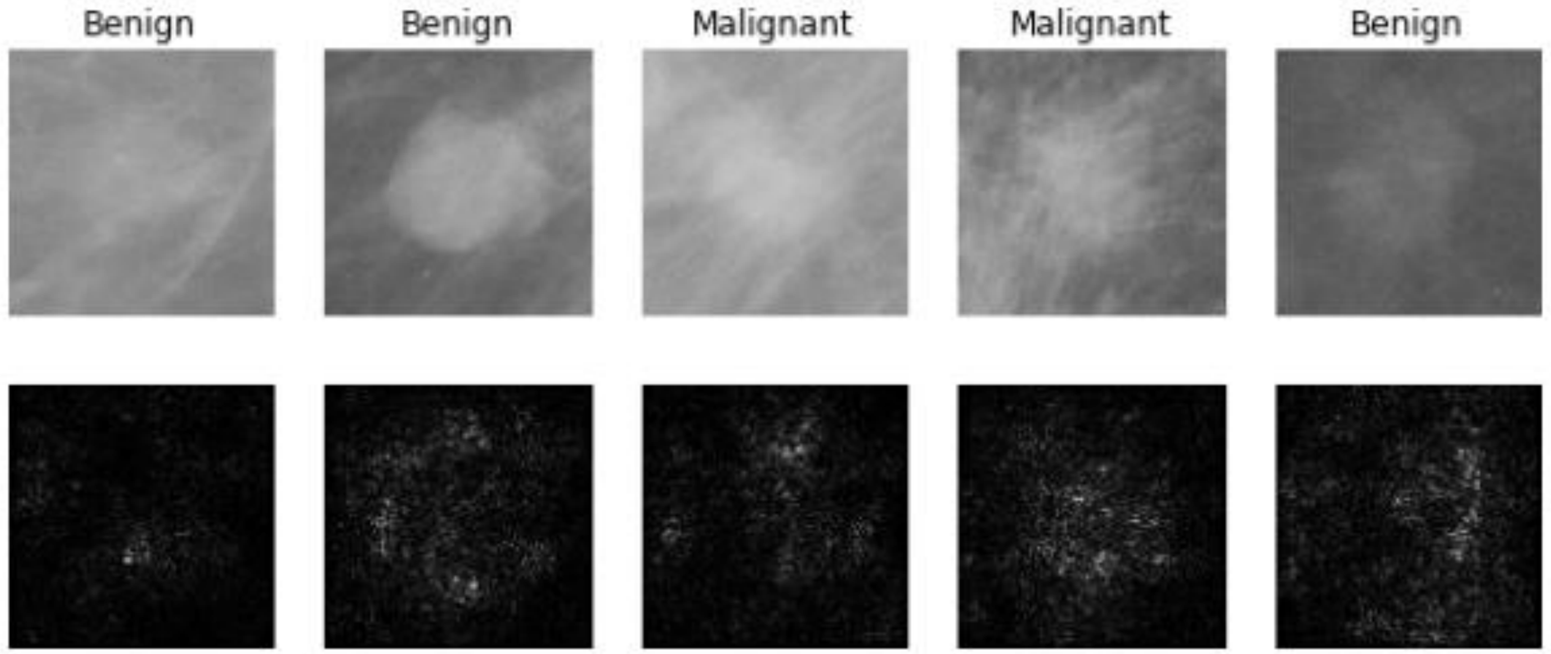


Obstacle Detection in Google's self driving cars



Semantic segmentation for autonomous vehicles

Classification



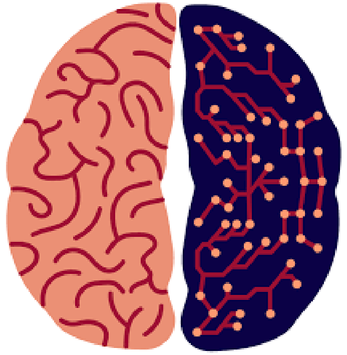
Neural net “dreams” Art generation.

DeepArt.io

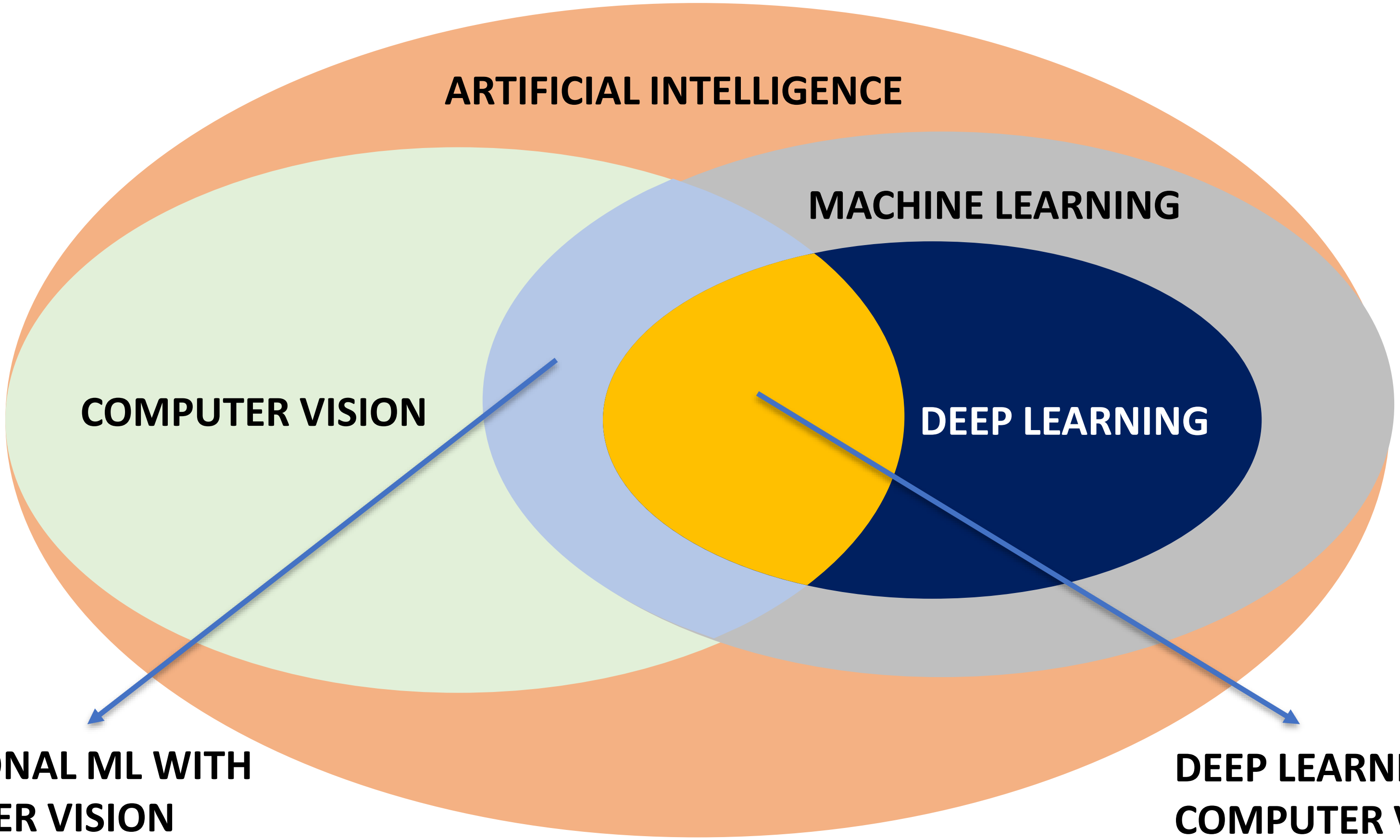
https://www.thispersondoesnotexist.com/

Deep Learning for Computer Vision

- A set of core algorithms focused on training ***deep*** neural networks to **learn** from visual data.
- These algorithms are inspired by the brain.
- Learning can be supervised, semi-supervised or unsupervised.



Classification: Confidential

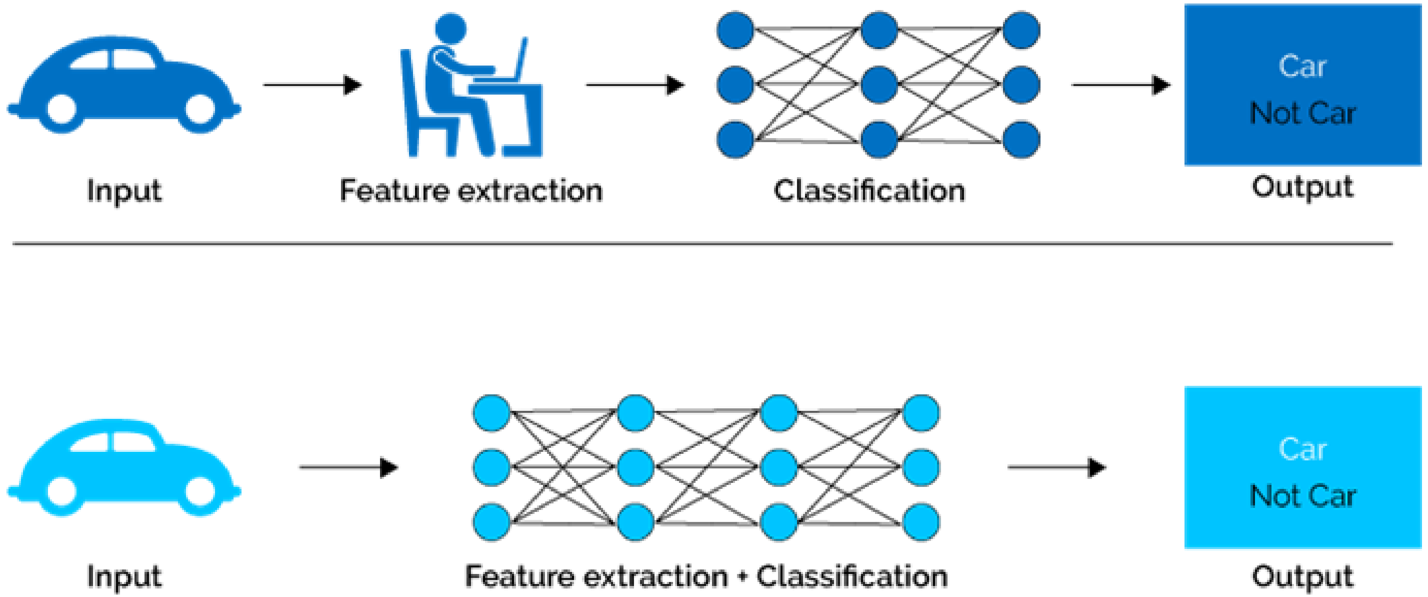


TRADITIONAL ML WITH COMPUTER VISION

DEEP LEARNING FOR COMPUTER VISION

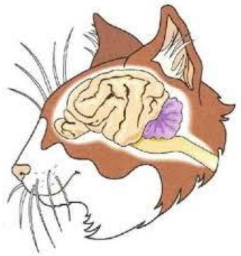
Traditional Computer Vision vs Deep Learning

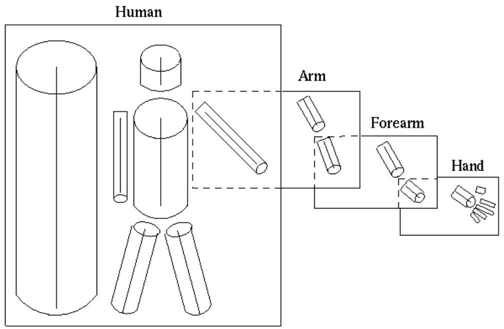
- Deep Learning bypasses the need of **manual feature engineering**

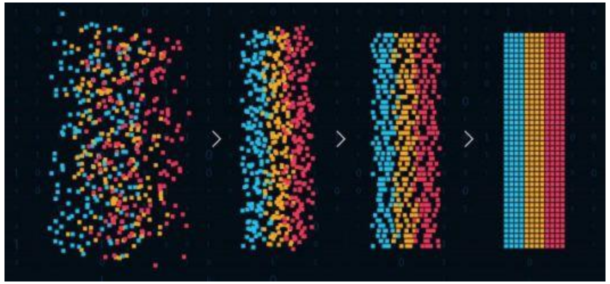


Computer Vision and Deep Learning Revolution

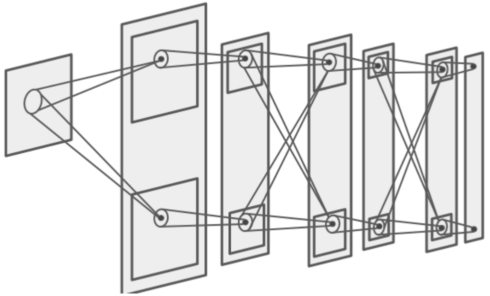
COMPUTER VISION

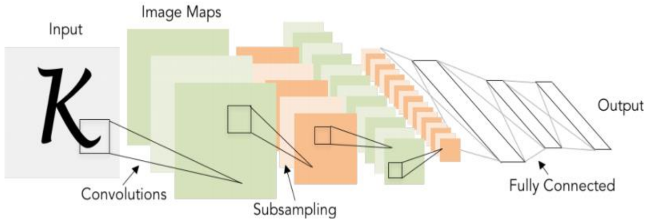


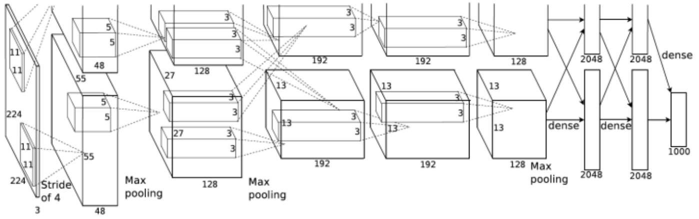




DEEP LEARNING



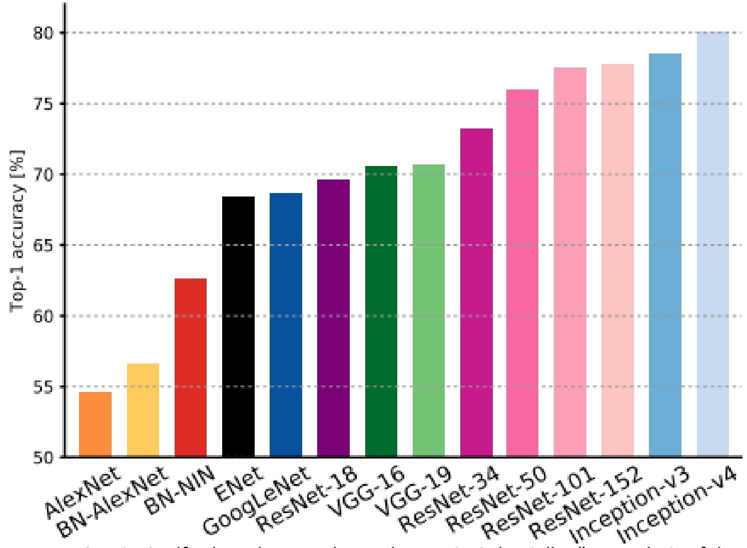


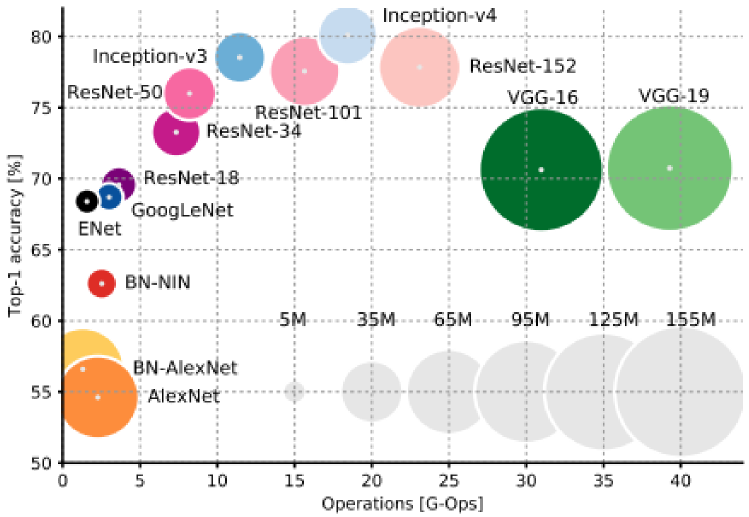


CNN Explainer: Learning Convolutional Neural Networks with Interactive Visualization

- Interactive CNN visualisation for the tiny VGG architecture.
- **Paper: https://arxiv.org/abs/2004.15004**
- **Demo: https://poloclub.github.io/cnn-explainer/**

Training ConvNets: Accuracy vs Computation



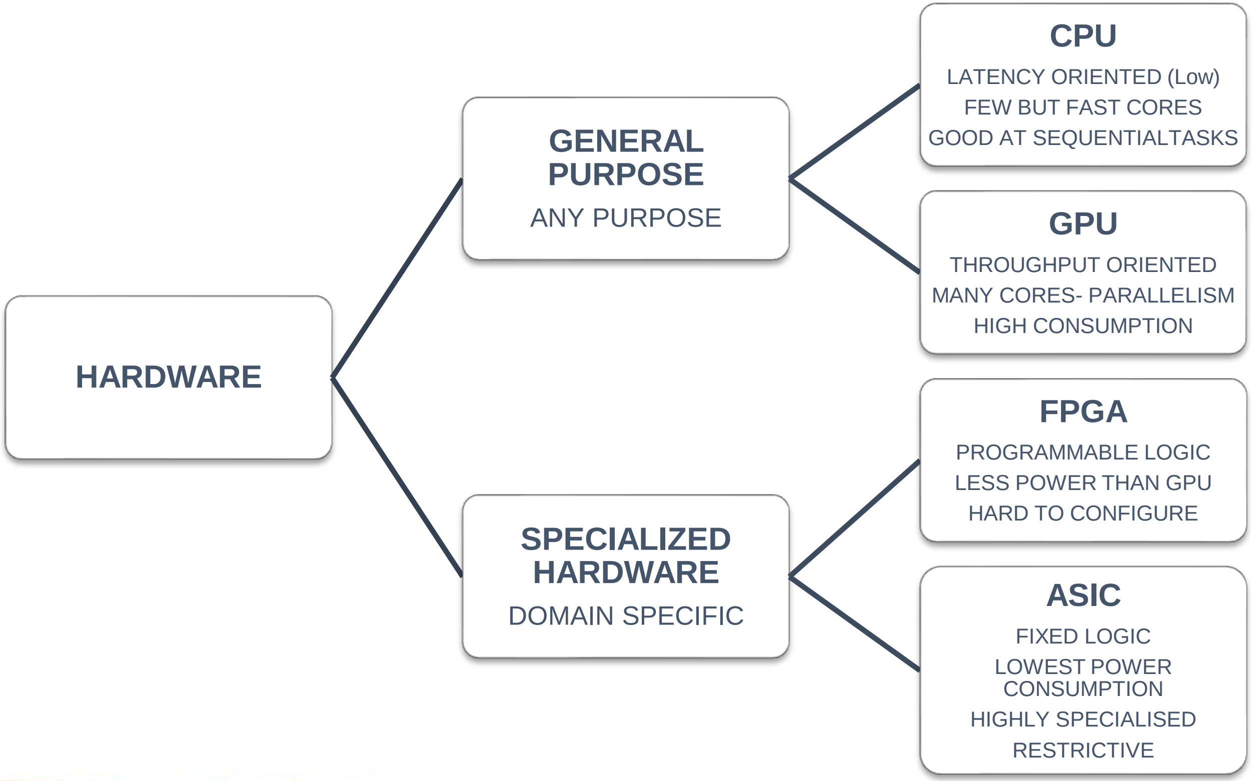


Canziani, Alfredo, Adam Paszke, and Eugenio Culurciello. "An analysis of deep neural network models for practical applications." arXiv preprint arXiv:1605.07678 (2016).

- **ImageNet classification challenge: Advanced STOA but lacked resource utilisation.**
- DL models reach an **inflection point- complexity costs start to outweigh gains in accuracy!**

DEEP LEARNING HARDWARE

- ❑ CPUs are designed for sequential tasks.
  - ❑ CPU clock speeds haven't improved much.
  - ❑ Impractical for large scale deep nets.
- ❑ **Challenges:**
  - ❑ DL Model complexity and size.
  - ❑ Computational speed.
  - ❑ Energy Efficiency



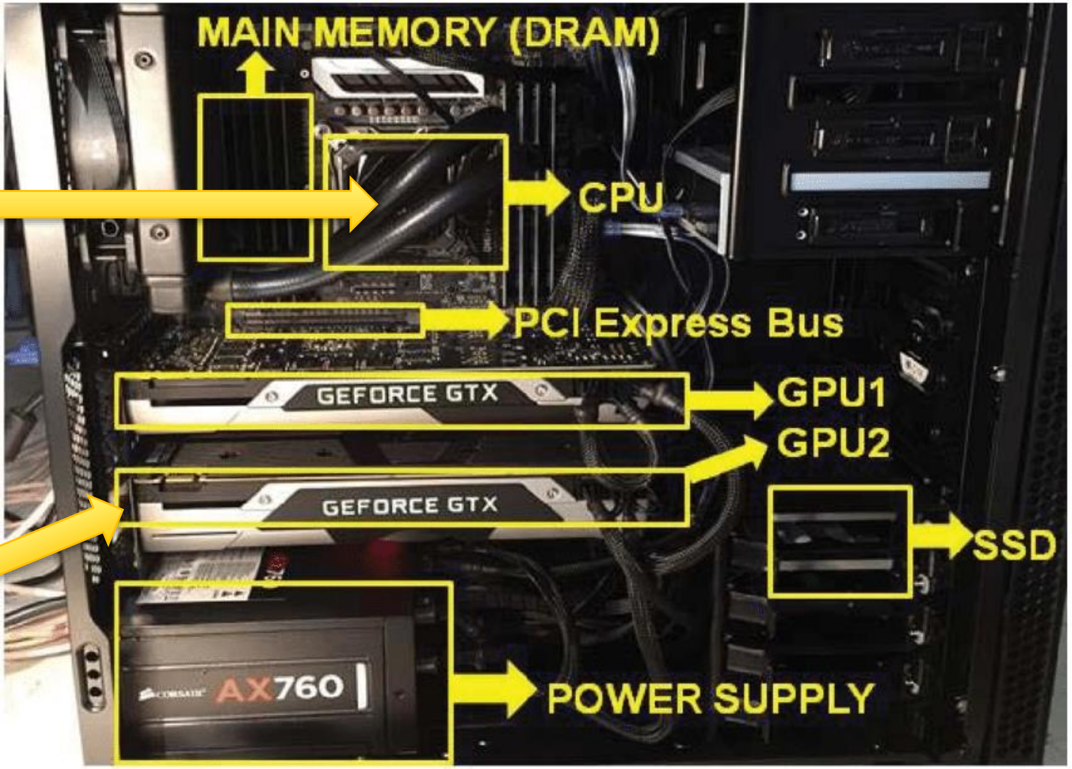
Inside your Computer

Central Processing Unit (CPU)

Intel Core i7-5930K CPU

Graphics Processing Unit (GPU)

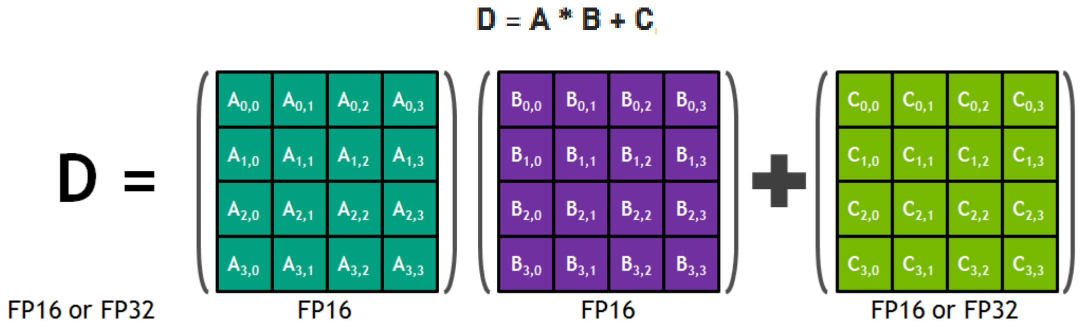
GEFORCE GTX TITAN Z GPU



Soyata, Tolga. (2018). GPU Parallel Program Development Using CUDA.

Graphics Processing Units- Tensor Cores

- ❑ Introduced in **Volta architecture** by NVIDIA.
- ❑ Programmable matrix-multiply-and-accumulate units.
- ❑ Increase floating-point compute throughput.
- ❑ **Mixed precision.**
- ❑ Efficient than CUDA cores.



$$D = \begin{pmatrix} A_{0,0} & A_{0,1} & A_{0,2} & A_{0,3} \\ A_{1,0} & A_{1,1} & A_{1,2} & A_{1,3} \\ A_{2,0} & A_{2,1} & A_{2,2} & A_{2,3} \\ A_{3,0} & A_{3,1} & A_{3,2} & A_{3,3} \end{pmatrix} \begin{pmatrix} B_{0,0} & B_{0,1} & B_{0,2} & B_{0,3} \\ B_{1,0} & B_{1,1} & B_{1,2} & B_{1,3} \\ B_{2,0} & B_{2,1} & B_{2,2} & B_{2,3} \\ B_{3,0} & B_{3,1} & B_{3,2} & B_{3,3} \end{pmatrix} + \begin{pmatrix} C_{0,0} & C_{0,1} & C_{0,2} & C_{0,3} \\ C_{1,0} & C_{1,1} & C_{1,2} & C_{1,3} \\ C_{2,0} & C_{2,1} & C_{2,2} & C_{2,3} \\ C_{3,0} & C_{3,1} & C_{3,2} & C_{3,3} \end{pmatrix}$$

matrix multiply inputs **A** and **B** are FP16 matrices

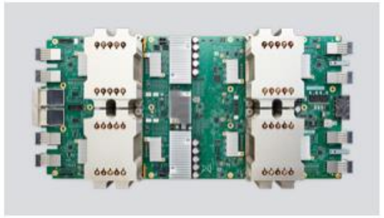
accumulation matrices **C** and **D** may be FP16 or FP32 matrices.

Tensor Core 4x4x4 matrix multiply and accumulate.

https://developer.nvidia.com/blog/programming-tensor-cores-cuda-9/

Tensor Programming Units

- ❑ **Good Cloud Platform (GCP).**
- ❑ Google's Custom-developed (ASICs).
- ❑ Very similar to NVIDIA Tensor Cores.
- ❑ Useful for training large transformers such as BERT, GPTs, machine translation models and CNNs.
- ❑ Restricted to Tensor Flow mostly.

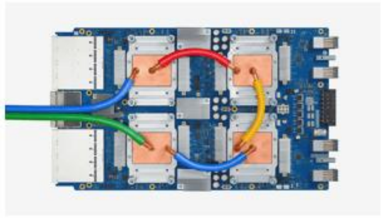


Cloud TPU v2

180 teraflops

64 GB High Bandwidth Memory (HBM)

$4.50 / TPU hour



Cloud TPU v3

420 teraflops

128 GB HBM

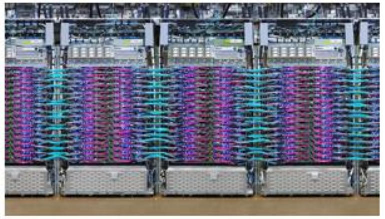
$8.00 / TPU hour

$384 USD

Cloud TPU v2 Pod

11.5 petaflops

4 TB HBM



Cloud TPU v3 Pod

100+ petaflops

32 TB HBM

https://cloud.google.com/tpu

SCALING DEEP LEARNING: VENDORS

NVIDIA

intel

AMD

ON-PREMISES

Google Cloud

amazon web services

Microsoft Azure

CLOUD PROVIDERS

Selecting your Nvidia GPUs!

Card	Release	Architecture	RAM (GB)	SM	Memory Type	CuDA Cores	Tensor Cores	32 bit TFLOPs	Tensor FLOPs	Boost Clock (GHz)	Memory Bandwidth (GB/sec)	Compute Capability	Launch Price/ USD
GTX 1080	27/05/2016	Pascal	8	20	GDDR5X	2,560	NA	8.87	NA	1.73	320.3	6.1	$599.00
Titan X	02/08/2016	Pascal	12	28	GDDR5X	3,584	NA	10.97	NA	1.53	480.4	6.1	$1,199.00
GTX 1080 Ti	10/03/2017	Pascal	11	28	GDDR5X	3,584	NA	11.34	NA	1.58	484.4	6.1	$699.00
V100 PCIe	21/06/2017	Tesla	16	80	HBM2	5,120	640	14.13	113.0	1.38	897.0	7.0	$10,664.00
V100 SXM2	21/06/2017	Tesla	16	80	HBM2	5,120	640	15.67	125.3	1.53	897.0	7.0	$10,664.00
Titan V	07/12/2017	Volta	12	80	HBM2	5,120	640	14.90	119.2	1.46	651.3	7.0	$2,999.00
Quadro GV100	27/03/2018	Volta	32	80	HBM2	5,120	640	16.66	133.3	1.63	868.4	7.0	$8,999.00
RTX 2080	20/09/2018	Turing	8	46	GDDR6	2,944	368	10.07	80.6	1.71	448.0	7.5	$699.00
RTX 2080 Ti	**20/09/2018**	**Turing**	**11**	**68**	**GDDR6**	**4,352**	**544**	**13.45**	**107.6**	**1.55**	**616.0**	**7.5**	**$999.00**
RTX 2070	17/10/2018	Turing	8	36	GDDR6	2,304	288	7.47	59.8	1.62	448.0	7.5	$499.00
TITAN RTX	**18/12/2018**	**Turing**	**24**	**72**	**GDDR6**	**4,608**	**576**	**16.31**	**130.5**	**1.77**	**672.0**	**7.5**	**$2,499.00**
RTX 2060	07/01/2019	Turing	6	30	GDDR6	1,920	240	6.45	51.6	1.68	336.0	7.5	$349.00
RTX 2060 SUPER	**09/07/2019**	**Turing**	**8**	**34**	**GDDR6**	**2,176**	**272**	**7.18**	**57.4**	**1.65**	**448.0**	**7.5**	**$399.00**
RTX 2070 SUPER	**09/07/2019**	**Turing**	**8**	**40**	**GDDR6**	**2,560**	**320**	**9.06**	**72.5**	**1.77**	**448.0**	**7.5**	**$499.00**
RTX 2080 SUPER	**23/07/2019**	**Turing**	**8**	**48**	**GDDR6**	**3,072**	**384**	**11.15**	**89.2**	**1.82**	**495.9**	**7.5**	**$699.00**

https://www.techpowerup.com/gpu-specs/geforce-gtx-1080-ti.c2877

Selecting your Nvidia GPUs!

- NVIDIA A100 GPU based on the new NVIDIA **Ampere** architecture.
- Next generation Tensor Cores.
- New data types supported.
- Lots of new features.

Card	Release	Architecture	RAM (GB)	SM	Memory Type	CuDA Cores	Tensor Cores	32 bit TFLOPs	Tensor TFLOPs	Boost Clock (GHz)	Memory Bandwidth (GB/sec)	Compute Capability	Launch Price/ USD
RTX 3080	**2020**	**Ampere**	**10**	**68**	**GDDR6X**	**4352**	**544**	**16.58**	**132.6**	**1.905**	**760.0**	**~8.0**	**$799.00**
RTX 3080 Ti	**2020**	**Ampere**	**12**	**80**	**GDDR6X**	**5120**	**640**	**17.82**	**142.6**	**1.740**	**912.0**	**~8.0**	**$999.00**
RTX 3090	**2020**	**Ampere**	**24**	**84**	**GDDR6X**	**5376**	**672**	**18.71**	**149.7**	**1.740**	**912.0**	**~8.0**	**$1999.00**
A100 SXM4	14/05/2020	Ampere	40	108	HBM2E	6912	432	19.49	77.96	1.410	1555.0	8.0	$12,500
A100 PCIe	22/06/2020	Ampere	40	108	HBM2E	6912	432	19.49	77.96	1.410	1555.0	8.0	$12,500

$$\text{Tensor TFLOPs} = \frac{Tensor\ Cores}{SM} * TFLOPS\ (16\ or\ 32\ bit)$$

Deep Learning Software Frameworks

Caffe

Microsoft Cognitive Toolkit

aka.ms/CognitiveToolkit

Microsoft

Keras

TensorFlow™

PYTORCH

Caffe2

DL4J

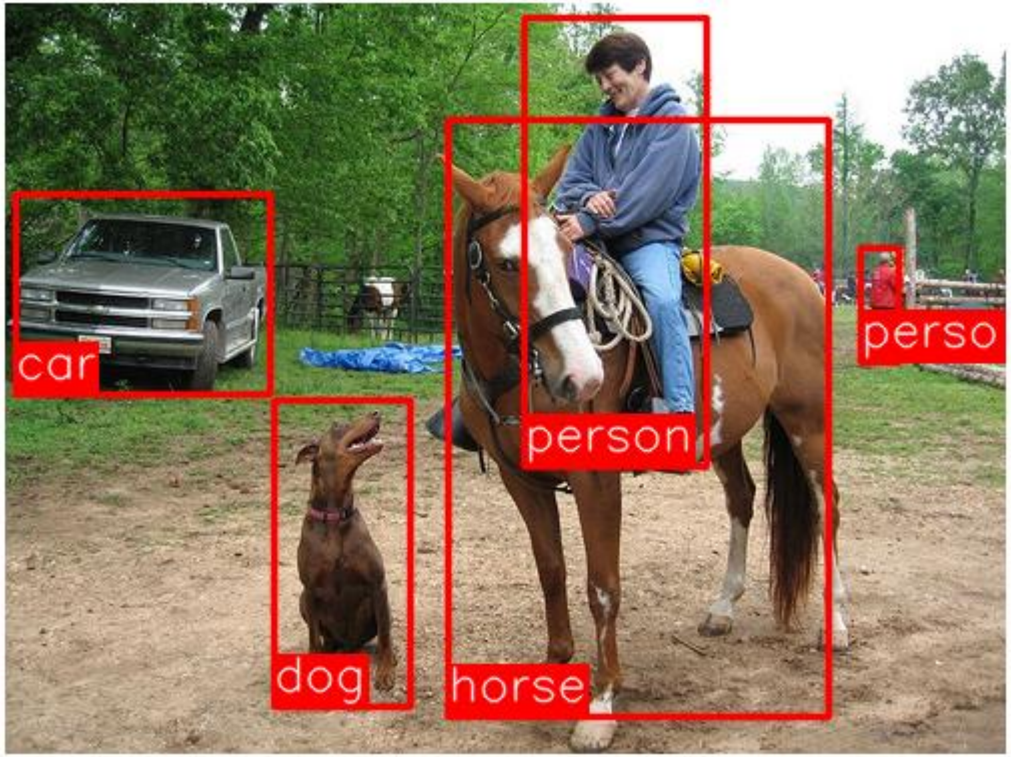
ONNX

mxnet

Object Detection

- ❑ **Input:** **Single RGB Image**
- ❑ **Output:**
- ❖ A set of detected objects and each object contains:
  - ❖ **Class label**
  - ❖ **Class probabilities**
  - ❖ **Bounding box coordinates**

    (x,y, width, height)

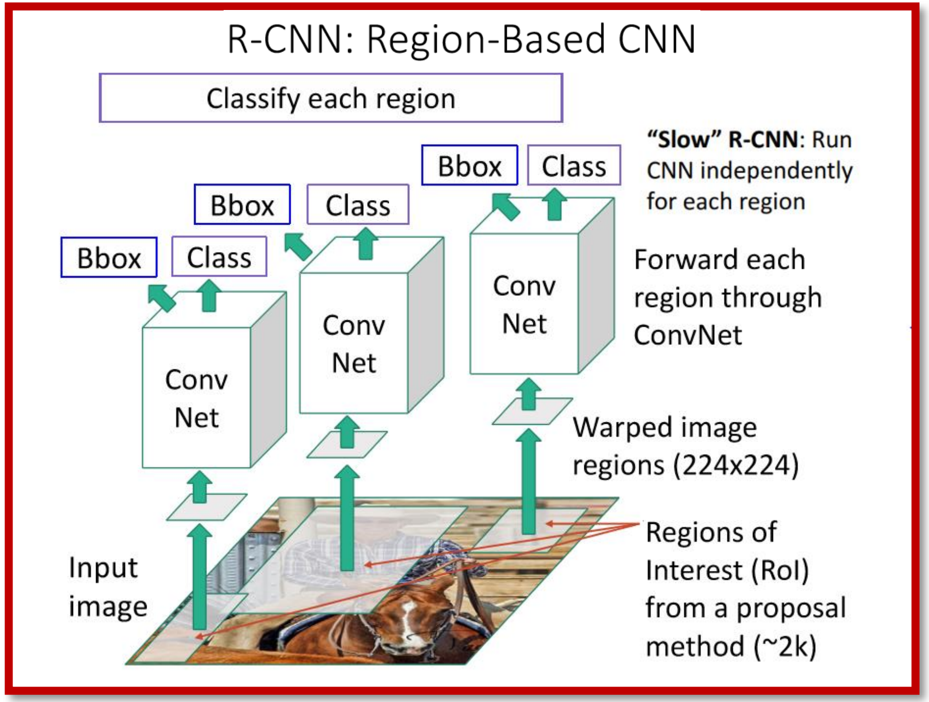


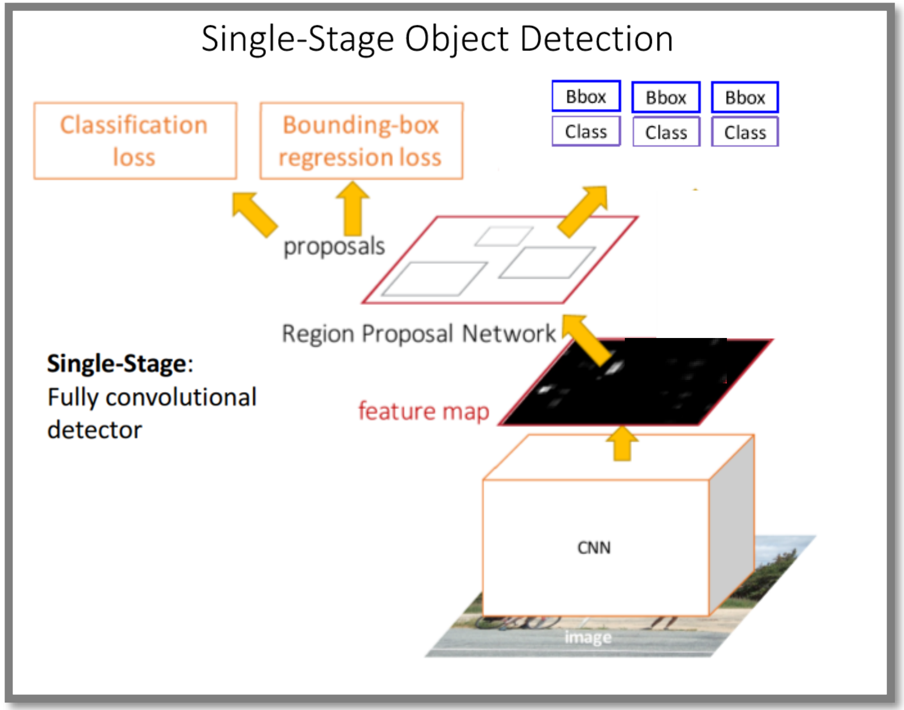
Girshick, Ross, et al. "Rich feature hierarchies for accurate object detection and semantic segmentation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2014.
Blog

Object Detection Challenges

- Multiple outputs
  - Different number of object classes per image.
- Imbalance Problems in Object Detection
- Overlapping detections.
  - Images with objects of high overlap.
  - Algorithms like **Non-Maximum Suppression (NMS)** may eliminate bounding boxes which are objects!
- Need good spatial resolution of images.
- Spatial resolution also puts constraints on computation.
  - Fewer images per batch in training increases training time.

Two shot vs Single Shot Detectors

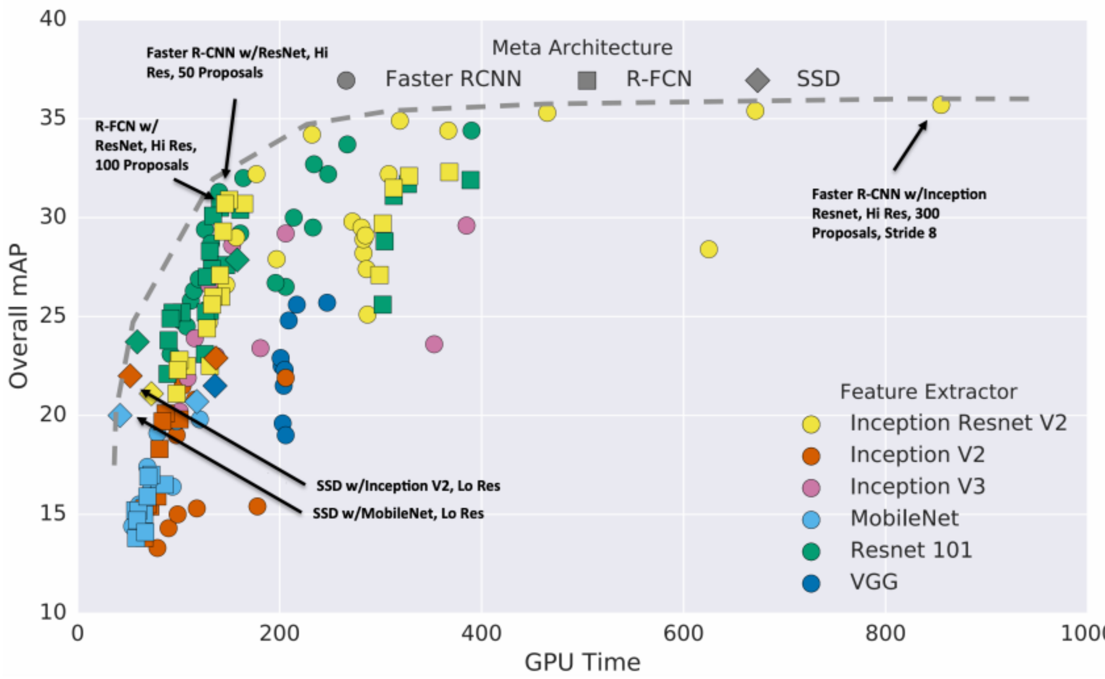

R-CNN: Region-Based CNN
Classify each region
"Slow" R-CNN: Run CNN independently for each region
Bbox
Class
Bbox
Class
Bbox
Class
Conv Net
Conv Net
Conv Net
Forward each region through ConvNet
Warped image regions (224x224)
Input image
Regions of Interest (RoI) from a proposal method (~2k)


Single-Stage Object Detection
Classification loss
Bounding-box regression loss
Bbox
Class
Bbox
Class
Bbox
Class
proposals
Region Proposal Network
Single-Stage: Fully convolutional detector
feature map
CNN
image

Girshick, Ross, et al. "Rich feature hierarchies for accurate object detection and semantic segmentation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2014.

Object Detection- Architecture Selection

- ❑ Single-shot architectures are faster with comparable accuracy.
- ❑ Bigger feature extractors improve performance but are slower.
- ❑ R-FCN only partially minimizes this computational load.
- ❑ MobileNet significantly outperforms two-shot architectures on the same meta architecture!
- ❑ **Lots of choices!**
- ❑ **Try to build your detectors on top of existing frameworks!**



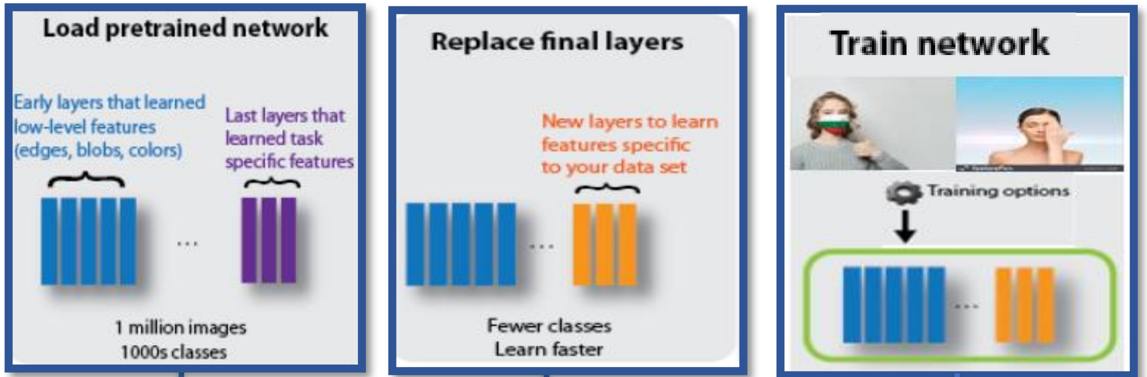
Huang et al, "Speed/accuracy trade-offs for modern convolutional object detectors", CVPR 2017

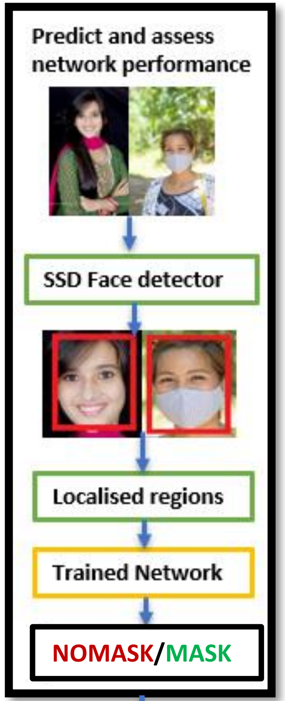
Latest state-of-the-art performance Read more

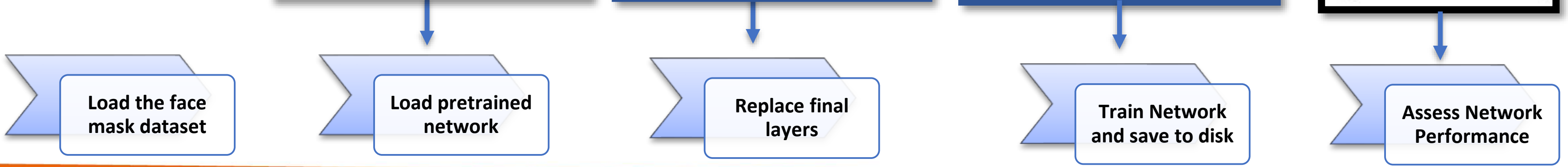
DEMO Methodology

Two Phase Face Detector

MobileNetV2 with weights from training on ImageNet


Load pretrained network
Early layers that learned low-level features (edges, blobs, colors)
Last layers that learned task specific features
1 million images
1000s classes
Replace final layers
New layers to learn features specific to your data set
Fewer classes
Learn faster
Train network
Training options




Load the face mask dataset
Load pretrained network
Replace final layers
Train Network and save to disk
Assess Network Performance

DEMO

Final Thoughts

- Deep learning has a played a major part in revolutionising Computer Vision however it is still limited to data.
- Computer vision techniques still remain relevant to date and should be used where they can.
- Deep learning needs dedicated hardware for training.
- Nvidia offers the greatest support in terms of GPUs.
- For Prototyping and inference, consider using GPUs.
- For training very large models, consider cloud platforms.
- Tensorflow and Pytorch are good choices for deep learning software.
- There are many datasets available for computer vision. Web scraping is also a good option.
- Object detection is achieved with a backbone feature extractor (CNN) and a meta-architecture (SSD, RCNN etc.)
- Dedicated DL frameworks allow computer vision tasks in a few lines of code!

Useful Resources

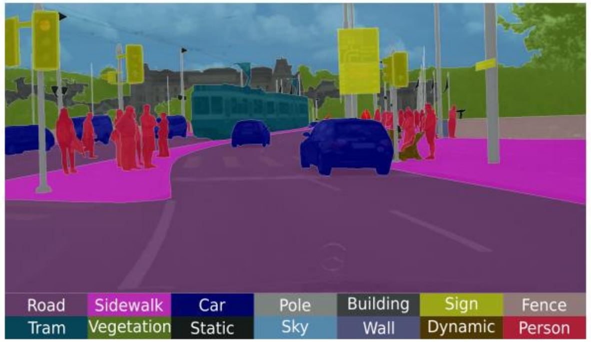
- Deep Learning for Computer Vision by Justin Johnson
- Pyimagesearch by Adrian Rosebrock
- GPU Specs Database
- Colab

GitHub Link

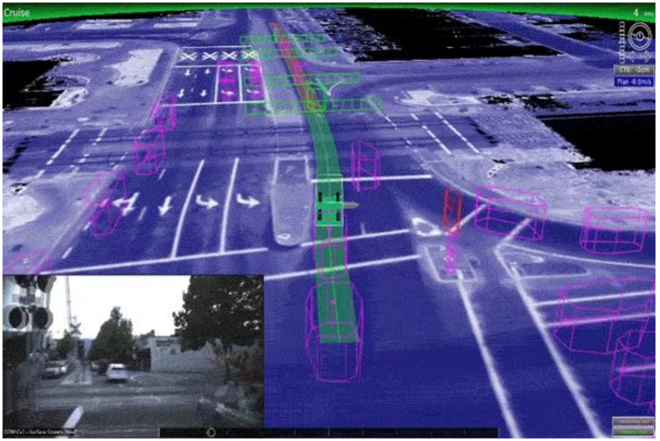
- https://github.com/aneeqr/Object-Detection

APPENDIX

Computer Vision is everywhere!

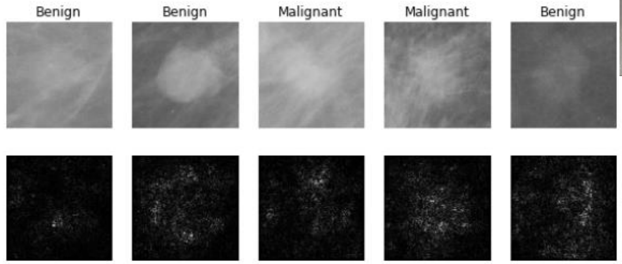


Semantic segmentation for autonomous vehicles



Obstacle Detection in Googles self driving cars

Neural net "dreams" Art generation.



Breast Mass classification

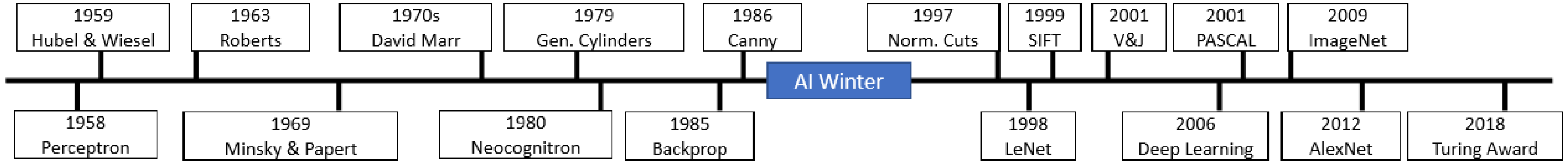
Pose Estimation

Traditional Computer Vision vs Deep Learning

	COMPUTER VISION	DEEP LEARNING
Domain Knowledge	Relatively more skilled expertise needed.	Less expert analysis required as neural nets are **trained** rather than programmed.
Flexibility	More domain-specific.	Superior flexibility because frameworks can be re-trained using a custom dataset.
Feature Extraction	Often Cumbersome and may involve a blend of different CV algorithms such as edge, corner detection or threshold segmentation.	DL Models trained on annotated data decipher most salient and descriptive features with every object class.
Parameters tuning	Often manual by CV Engineers. Trial and error needed. Heuristics.	Fine Tuning carried out automatically during the training phase of the neural net by a variety of approaches.
Transparency	Full transparency. CV engineer can transfer insights in training to their algorithm.	Often criticised as a **black box**.
Performance	Classic CV algorithms are well-established, transparent, and optimized for performance.	Usually high accuracy and versatility but often limited to data.
Computation	Less time and data required for training.	Require large amounts of computing resources.

O'Mahony, Niall, et al. "Deep learning vs. traditional computer vision." Science and Information Conference. Springer, Cham, 2019.

Historical Context- Computer vision and Deep Learning



Deep Learning for Computer Vision by Justin Johnson

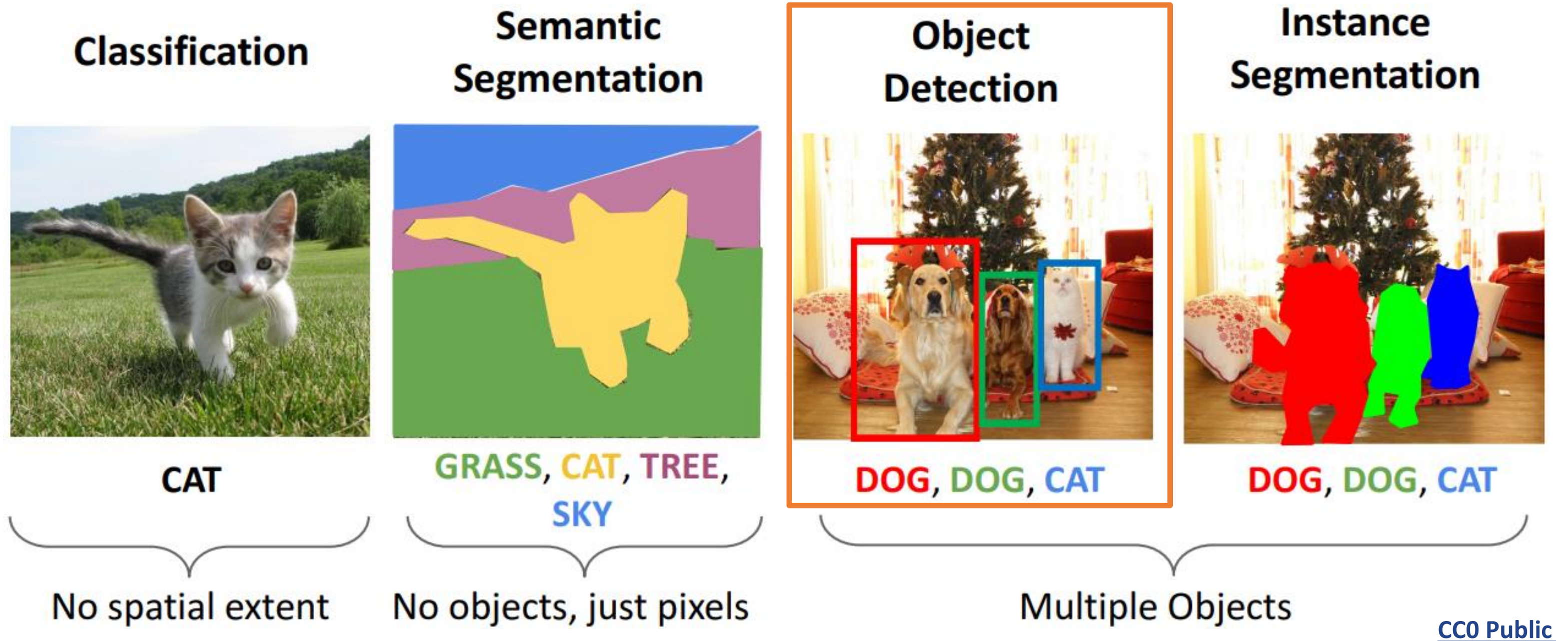
Computer Vision

- 1959- Hubert and Wiesel's research on how the ***mammalian brain*** *works*.
- 1970s- ***Stages of Visual Representation*** by David Marr.
- 1980s- ***Recognition via edge detection*** by David Lowe.
- 1990s- ***Recognition via Grouping with image semantics*** by Shi and Malik.
- 2000s- ***Scale-invariant feature transform (SIFT)*** by David Lowe.
- 2001- ***Face Detection algorithm*** by Viola and Jones.

Deep Learning

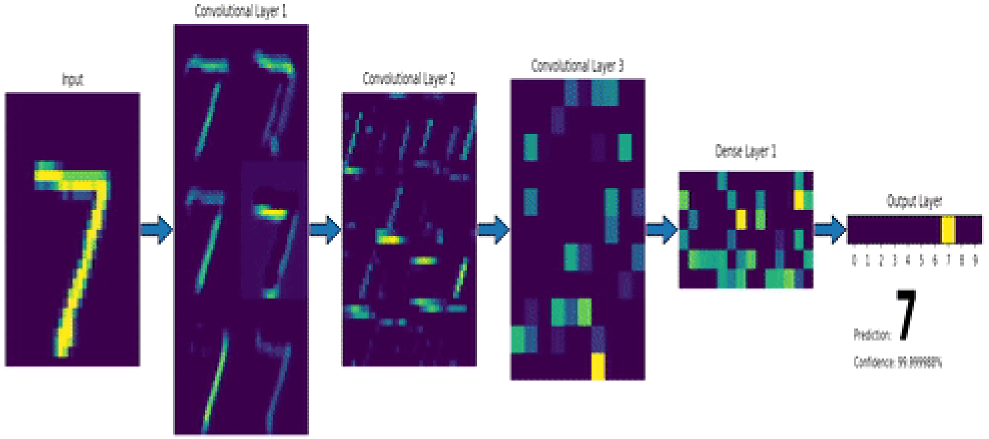
- 1958- ***Perceptron*** by Frank Rosenblatt.
- 1980s – ***Neocognitron***: hierarchical, multilayered ANN proposed by Kunihiko Fukushima in 1979.
- 1986- ***Backpropagation*** by Rumelhart, Hinton and Williams.
- 1998- LeNet- ***Backpropagation applied to the Neocognitron***.
- In the 2000s people started to train **deep neural nets**.
- 2005 to 2011- ***PASCAL Visual object challenge (2007), ImageNet (2009). Datasets for computer vision introduced.***
- 2012 to Present, ***ConvNets everywhere! AlexNet game changer.***

Computer vision tasks



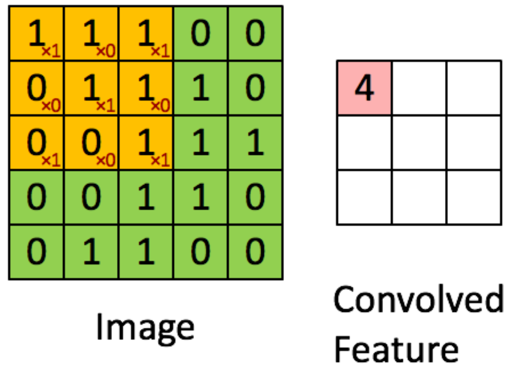
Convolutional Neural Networks (ConvNets or CNNs)

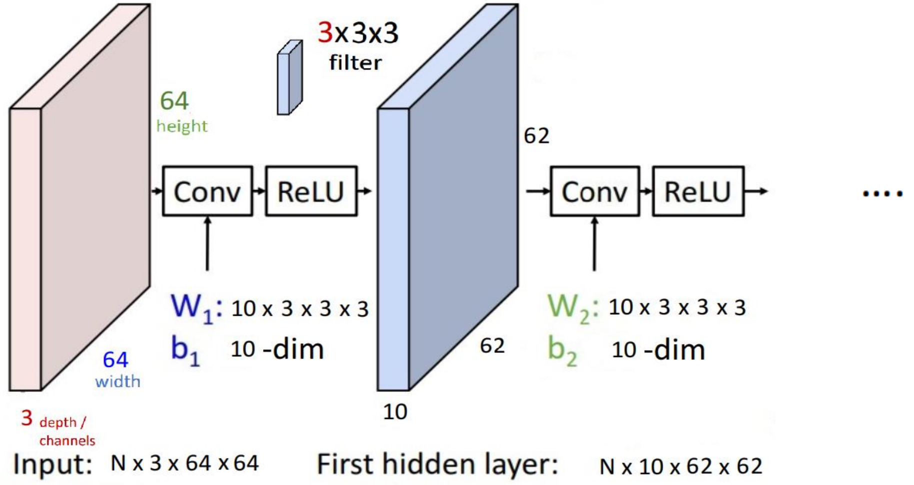
- Class of deep neural networks mostly used for visual image analysis.
- Components:
  - Fully-connected layers.
  - Convolution layers.
  - Pooling layers.
  - Activation Function (RELU).
  - Normalization.



https://gfycat.com/smoggylittleflickertailsquirrel-machine-learning-neural-networks-mnist

CNNs components: Convolutional layers





[*Tiny VGG: CNN Explainer*]

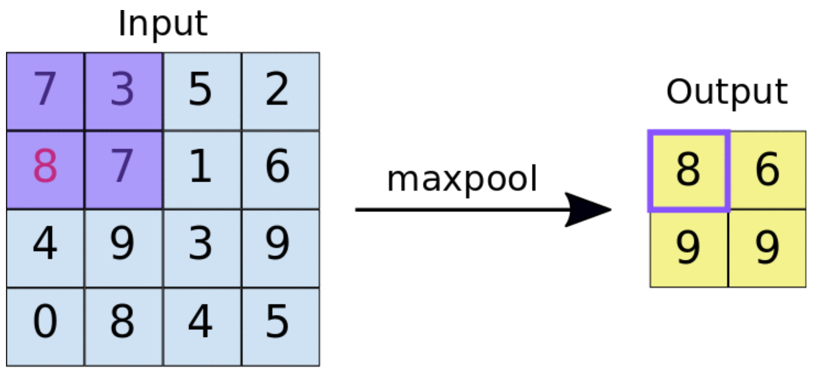
- Preserve spatial structure of input.
- Conv filters compute dot products.
- Output 2-D **activation maps.**
- Serve as **Feature Extractors**.
- **Hyperparameters:** *Stride(S), Kernel Size (KS), No of filters, Padding(P).*

- Output H/W = $\frac{(Image\ H/W - KS + 2*P)}{S} + 1$
- Output H/W $= \frac{(64-3+0)}{1} + 1$ = 62x62

Convolution layers parameters

CNNs components: Pooling layers

- Usually after non-linear activations to introduce spatial invariance.
- **Down sampling.**
    - Reduce feature map dimensions, while still **retaining the most critical feature information**.
- **Hyperparameters:** *Stride (S), Kernel size (KS), Pooling function (Max, Average etc.)*
- **Advantage: No learnable parameters!**



- Output H/W = $\frac{(Image\ H/W - Kernel\ Size)}{Stride}$

  $= \frac{4-2}{1}$ = 2X2

conv layers

DL HARDWARE: VENDORS COMPARISON

On-Premises:

- Nvidia offers highest support.
  - Tensor Cores, CUDA Toolkit, optimized APIs (cuDNN etc.), useful libraries like Apex.
- AMD GPUs are powerful but have limited support.
  - **Navi** architecture aimed to achieve performance targets comparable to Tesla V100.
- You can get pre-built solutions or build your own workstation as well.

Cloud Providers:

- Google Cloud Platform (GCP):
  - Has GPU AND Tensor Programming Units (TPUs).
  - GCP lets your connect GPUs to their instances. AWS and Azure have pre-built templates.
- AWS most expensive.
- **Spot instances** useful for quick experimentation and cheaper than **on-demand** instances.
- Colab useful for experimentation and prototyping.

Infrastructure & Tooling - Full Stack Deep Learning

Selecting your NVDIA GPUs!

- ❑ Remember that computation would depend on:
  - ❑ **Floating point operations per second (FLOPs)**
  - ❑ Your **deep learning architecture** and **task**.
- ❑ Things to look at while selecting a GPU:
  - ❑ **Tensor Cores . (Mixed precision)**
  - ❑ **32-bit TFLOPs (Higher better)**
  - ❑ **Memory Bandwidth (Higher better)**
  - ❑ **16-bit computing capability.**
  - ❑ **RAM (Appropriate batch size)**
  - ❑ **Compute capability (> 7.0 to benefit from Mixed precision)**
- ❑ **Convolutional networks and Transformers**:

Tensor Cores > FLOPs > Memory Bandwidth > 16-bit capability

- ❑ **Recurrent networks:**

Memory Bandwidth > 16-bit capability > Tensor Cores > FLOPs

https://timdettmers.com/2019/04/03/which-gpu-for-deep-learning/
https://www.techpowerup.com/gpu-specs/geforce-gtx-1080-ti.c2877

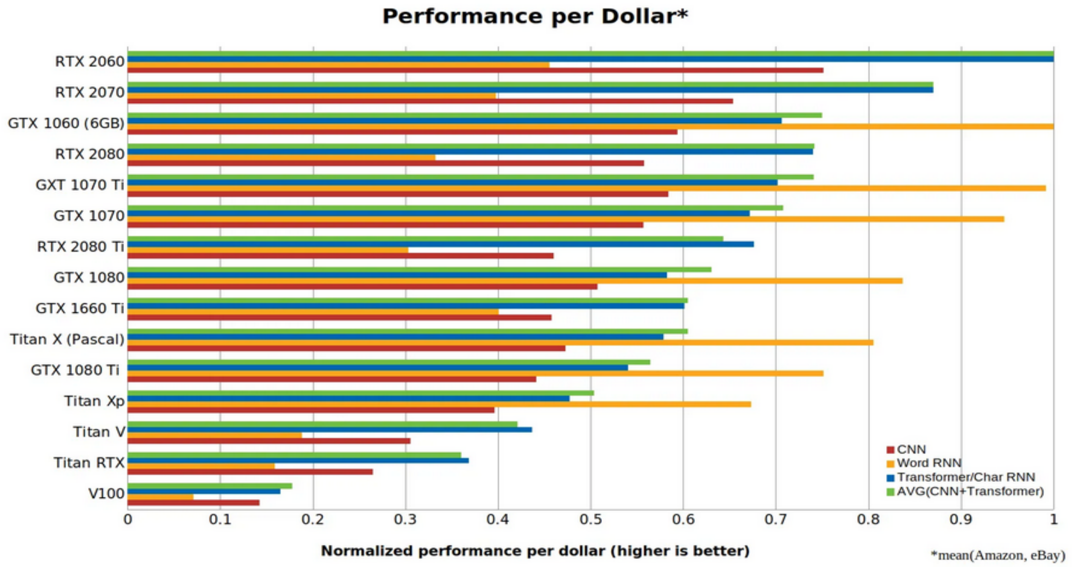


Figure 3: Normalized performance/cost numbers for convolutional networks (CNN), recurrent networks (RNN) and transformers. Higher is better. An RTX 2060 is more than 5 times more cost-efficient than a Tesla V100. The word RNN numbers refer to biLSTM performance for short sequences of length <100. Benchmarking was done using PyTorch 1.0.1 and CUDA 10.

Advice on GPUs

- If new to GPUs- stick with Nvidia.
- RTX 2080 Ti with 11 GB more than enough memory to run most of your State-of-the-Art Models.
  - Train at half-precision (16-bit) to double GPU memory and achieve reasonable accuracy.
- Go for Titan RTX if memory is an issue.
- If cost is not an issue, you can aim for Tesla/Titan cards.
- Try and use latest architectures for support reasons.
- SSD considerations and RAM are important considerations.
- With multi-GPU training, over-heating can occur. Also need Scalable Link Interfaces (SLI).
- Wait for the new Ampere Graphic cards to come out.

https://lambdalabs.com/blog/titan-v-deep-learning-benchmarks/

DL HARDWARE SUMMARY

CPUs

- Quick for prototyping with simple models with low training times.
- Effective for small models with small effective batch sizes.
- Models that are dominated by custom TensorFlow operations written in C++.
- Models that are limited by available I/O bottleneck or the networking bandwidth of the host system.

GPUs

- Models for which source does not exist or is too onerous to change
- Models with a significant number of custom TensorFlow operations that must run at least partially on CPUs.
- Models with TensorFlow operations that are not available on Cloud TPU (see Available Tensor Flow Operations on Cloud TPU)
- Medium-to-large models with larger effective batch sizes.
- In the prototyping and inference phase, you should rely on non-cloud options to reduce costs.

TPUs

- Recommended for training phases.
- Models dominated by matrix computations.
- Models with no custom TensorFlow operations inside the main training loop.
- Models that train for weeks or months.
- Very large models with very large effective batch sizes.

https://cloud.google.com/tpu/docs/tpus

Deep Learning Software Frameworks

Framework	Written in	Maintained by	Salient Features
Tensorflow	Python C++,CUDA	Google open source project	• Lots of resources and support available. • Features like: **Tensorboard/Tensorflow Lite/Tensorflow JS etc.** • Allows serving models at scale in a production environment with distributed training. • **Tensorflow 2.0- Eager Execution Mode-** More control over the static computation graph**. *Define and run vs*** Define by run. • Preferred for **large scale production environments** across platforms**.** • Best choice for production on **Google Cloud Platform.**
Pytorch	Python, C++, CUDA	François Chollet, a researcher at Google	• Attracted attention because of the ability to generate **dynamic computational graphs.** • More transparency in modelling and useful for development purposes. Good for prototyping and small projects. • **1.0** and above, **Caffe2** integrated in its codebase, makes it more **production ready**. And static graphs introduced as well.
Keras	Python	François Chollet but various developers.	• Easy-to-understand and consistent APIs • Seamlessly integrates with TensorFlow workflow and GCP. • Good at beginner level! Too high level;.

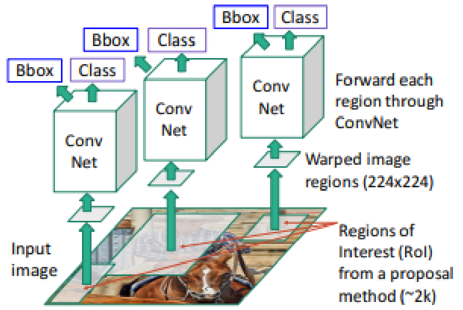
Deep Learning Software Frameworks

Framework	Operating Systems and Platforms	Written in	Maintained by	Salient Features
Caffe2	Linux, Mac, Windows	C++	Berkeley, Facebook	• Now part of PyTorch. • C++ library comes with a Python interface. • Easier to set up and train, without having to build onto the network. • The configuration defines models without hard-coding
Apache MXNet	Linux, Windows, Mac, Android, iOS, Javascript	C++, Python, R, Java, Julia, JavaScript, Scala, Go, Perl	Amazon	• Designed specifically for high efficiency, productivity on AWS. • Hybrid Programming massively improves computational efficiency by combining declarative and imperative programming. • Nearly linear on GPU clusters which provides excellent scalability.
Microsoft CNTX	Linux, Windows, Mac with Docker	C++	Microsoft	• Useful for CNNs, NLP, image and speech-based data. • Highly efficient and scalable for multiple machines.
Open Neural Network Exchange (ONNX)	Windows, Linux	C++, Python	Facebook, Microsoft	• Allows frameworks to share models. • Provides model interoperability and flexibility with compatible libraries. • Maximizes performance across hardwares.
DL4J	Cross-platform software	Java, CUDA, C++	Various	• Full stack Java machine learning pipelines. Useful for android apps. • Can process massive amounts of data quickly.

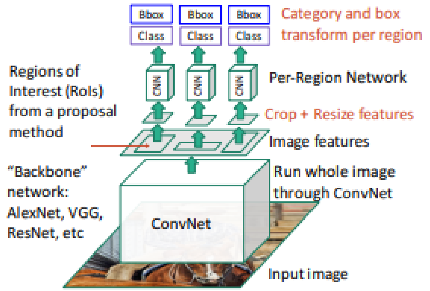
https://marutitech.com/top-8-deep-learning-frameworks/

Object Detection Methods Overview

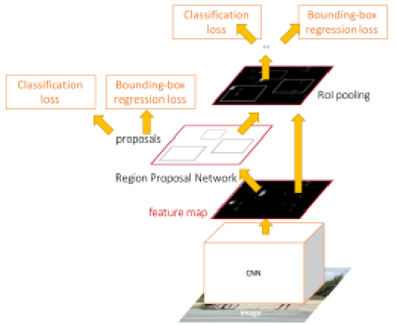
"Slow" R-CNN: Run CNN independently for each region


Bbox
Class
Bbox
Class
Bbox
Class
Conv Net
Conv Net
Conv Net
Forward each region through ConvNet
Warped image regions (224x224)
Regions of Interest (RoI) from a proposal method (~2k)
Input image

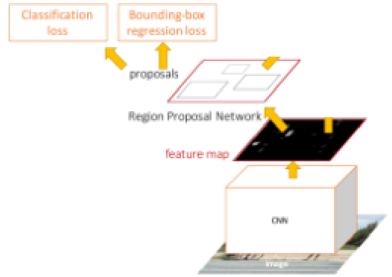
Fast R-CNN: Apply differentiable cropping to shared image features


Bbox
Class
Bbox
Class
Bbox
Class
Category and box transform per region
Regions of Interest (RoIs) from a proposal method
CNN
CNN
CNN
Per-Region Network
Crop + Resize features
Image features
"Backbone" network: AlexNet, VGG, ResNet, etc
ConvNet
Run whole image through ConvNet
Input image

Faster R-CNN: Compute proposals with CNN



Single-Stage: Fully convolutional detector



Ren et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NIPS 2015

Deep Learning for Computer Vision by Justin Johnson

Girshick, Ross, et al. "Rich feature hierarchies for accurate object detection and semantic segmentation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2014.

Object Detection Methods

Two-shot Detection

- Two stages.
  - ROIs (Regions of interest) from a proposal method. (Selective Search)
  - Classification of regions and then refinement of location prediction
- Slower but more accurate.
  - Handle imbalance problems during training much better.
- Object Detection architectures:
  - R-CNN and its variants.
  - Region Based Fully Convolutional Networks (R-FCN)

Single-Shot Detection

- One stage final localization and content prediction.
  - **Region Proposal Network (RPN) identifies regions**, **classifies them as objects** and generates the **final bounding boxes**.
- Faster and has considerable improvements over the recent years over its accuracy.
- Object Detection architectures:
  - SSD: Single Shot MultiBox Detector
  - You Only Look Once: Unified, Real-Time Object Detection

Open CV

- Popular library for Computer Vision.
- Version 3.3, Deep Learning support improved.
- DNN Module.
- Has supports for Caffe, TensorFlow, PyTorch.
- Allows us to use pre-trained models from our deep learning frameworks.

Datasets for Deep Learning

- MNIST: Digit classification
- MS-COCO: Detection, segmentation and recognition.
- ImageNet (On this one) many image categories.
- Open Images Dataset
- VisualQA
- SVHN
- CIFAR-10: simple images.
- Web scarping as well (selenium, agility pack html).
- Microsoft BING API.

https://www.analyticsvidhya.com/blog/2018/03/comprehensive-collection-deep-learning-datasets/