Deep Learning Crash Course

By: Josue Cuevas

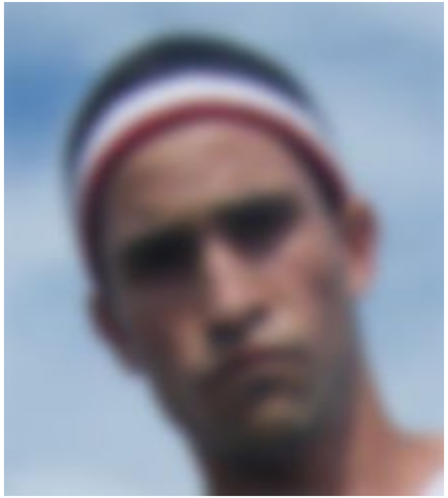
Sept-1st

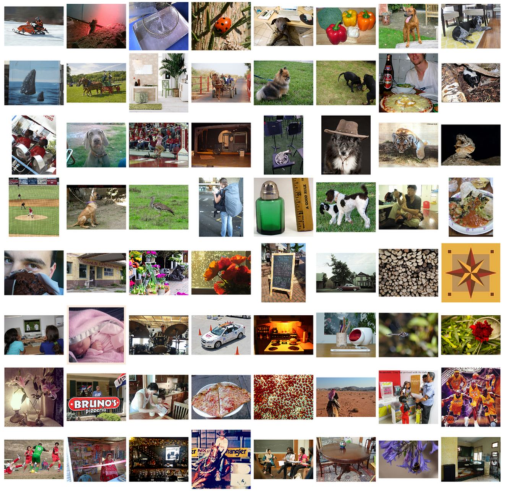
Outline

- Why Deep Learning?
- What and How much data do we need?
- Data Preparation and data collection
- Data augmentation techniques
- Network Topologies
    - AlexNet, VGG
    - Which topology is the best?
- Generalization and Regularization
- Invariance and Equivariance
- Training Neural Networks

Why Deep Learning

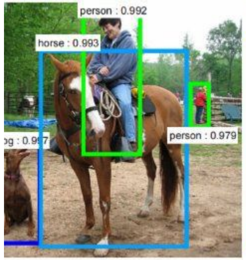
- You robustness for very challenging conditions (robustness)
- You don’t know what features to use to make your algorithm work properly (features)
- You want to be able to predict multiple things (capacity)

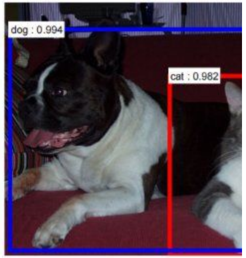




What data? … and How much?

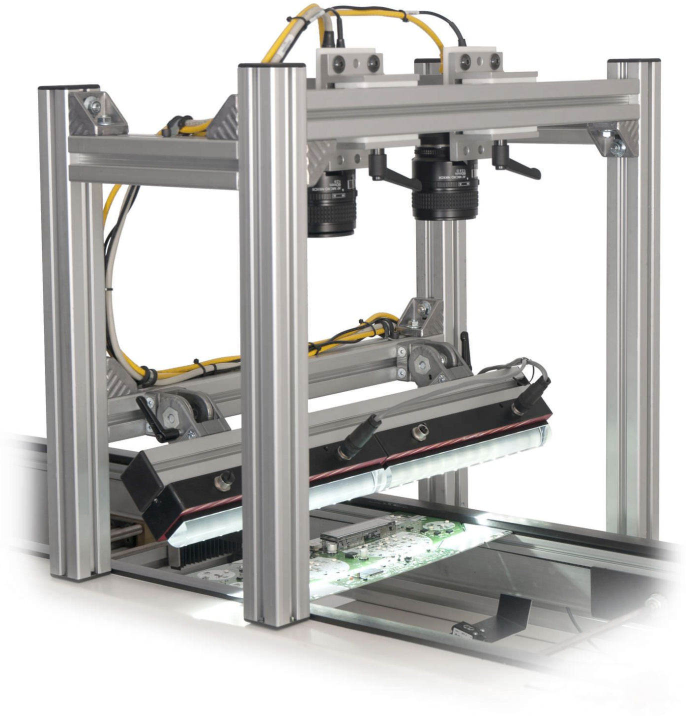
- First, define what kind of problem are you solving
  - Detection
  - Classification
  - Segmentation
  - etc.





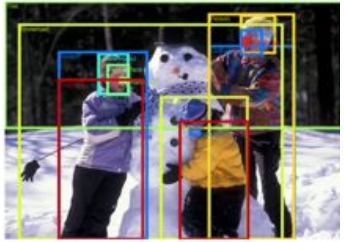
- As a rule of thumb, you need at least 100 observations of each class or objects you are predicting.
- The data has to be as general and variable as possible

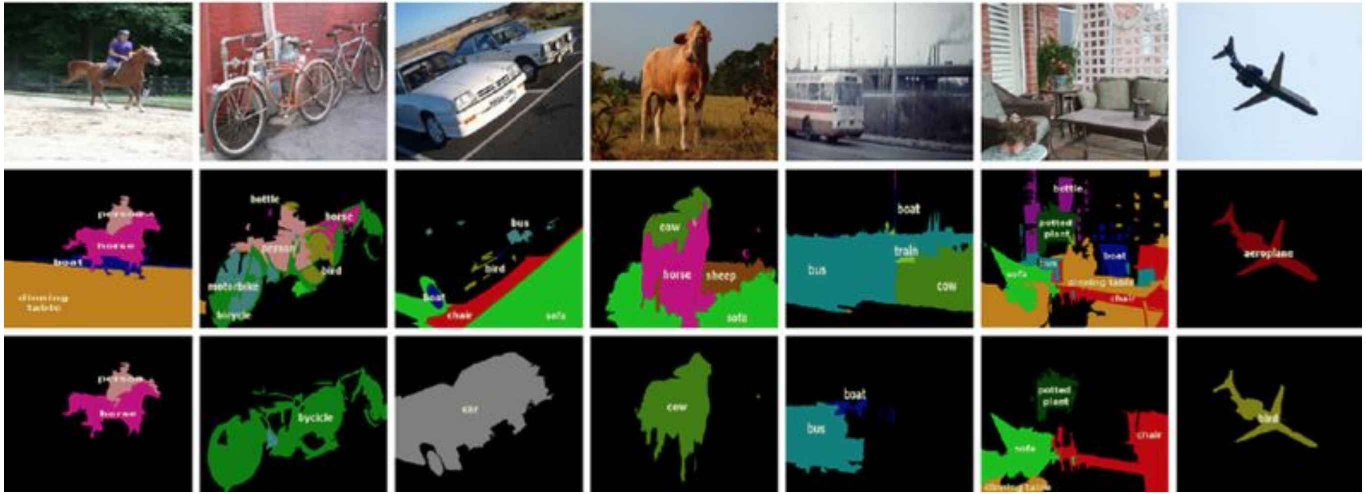
Deep Learning in Industrial Inspection



Data preparation and collection

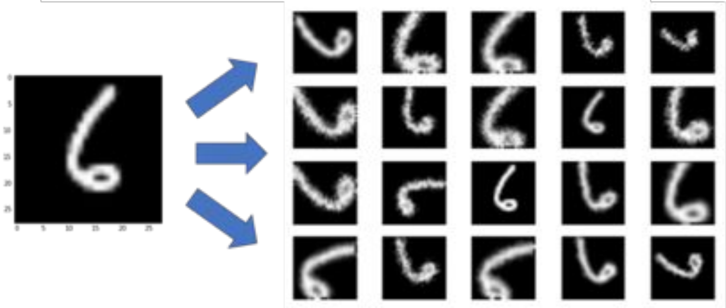
- Google dataset
- ImageNet
- COCO-dataset
- Youtube dataset

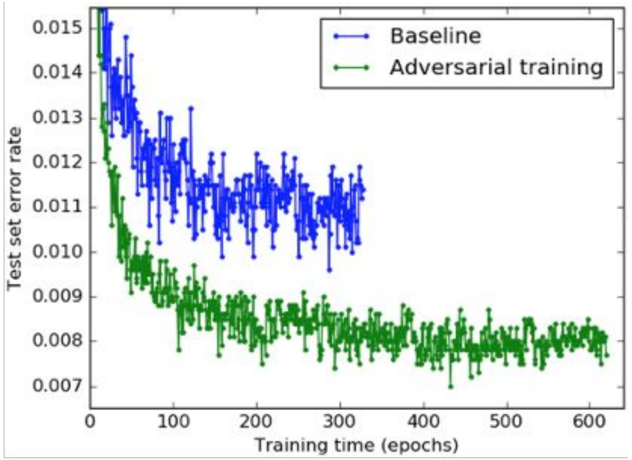


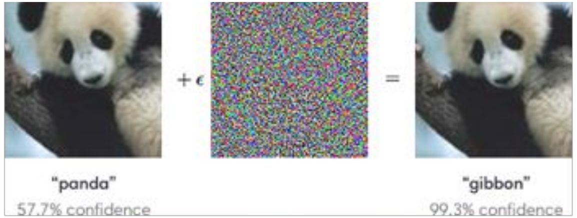


Data augmentation

- [Example of audio augmentation]
- [Example Image Augmentation]

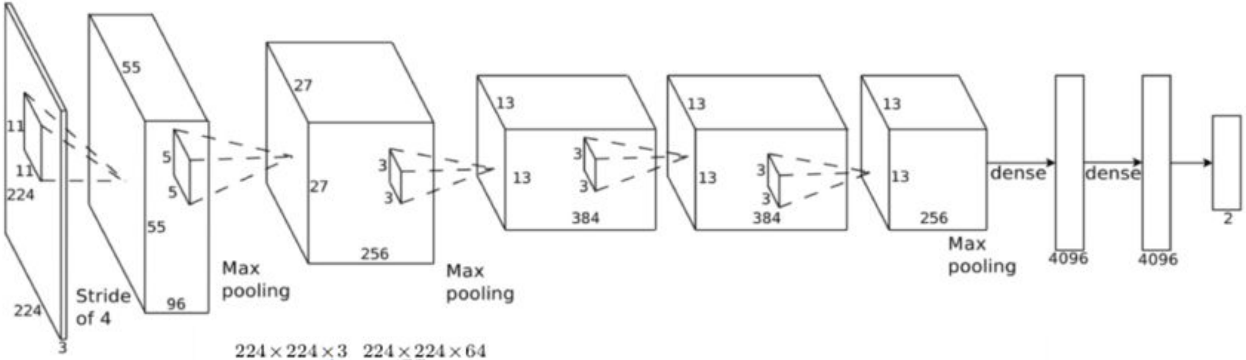




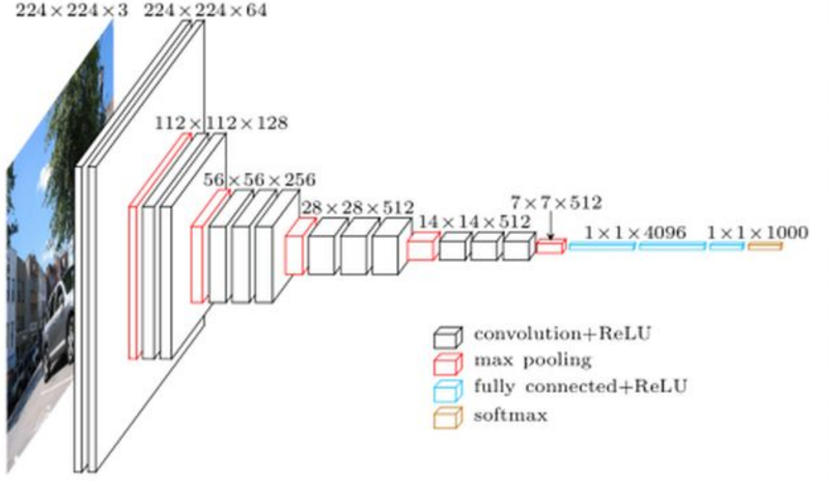


Network Topologies

- AlexNet



- VGG

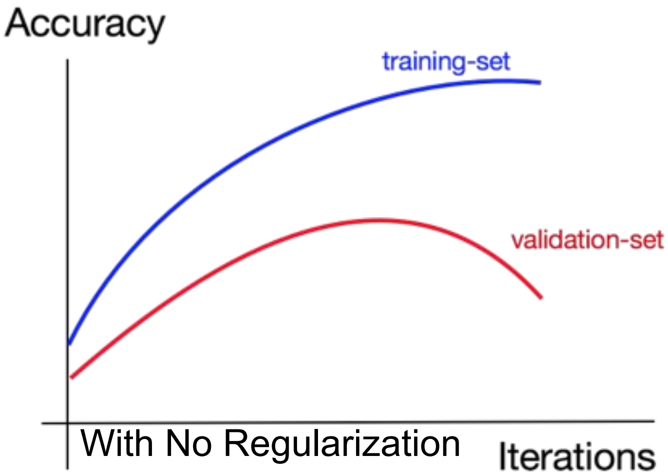


How to choose the topology?

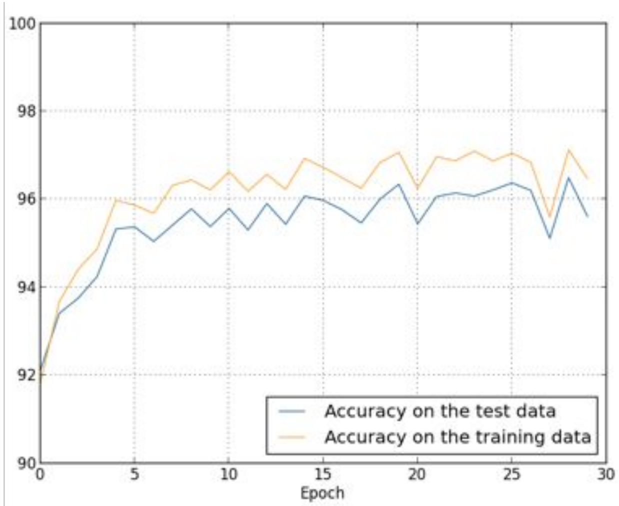
- How much capacity you need
- How much robustness you need
- How much computing power you have
- Where do you plan to deploy your network
- How fast do you want the network to be

Generalization and Regularization

- [Dropout layers]()
- [L2-Normalization]()
- [Batch-Normalization]()
- [Group-Normalization]()



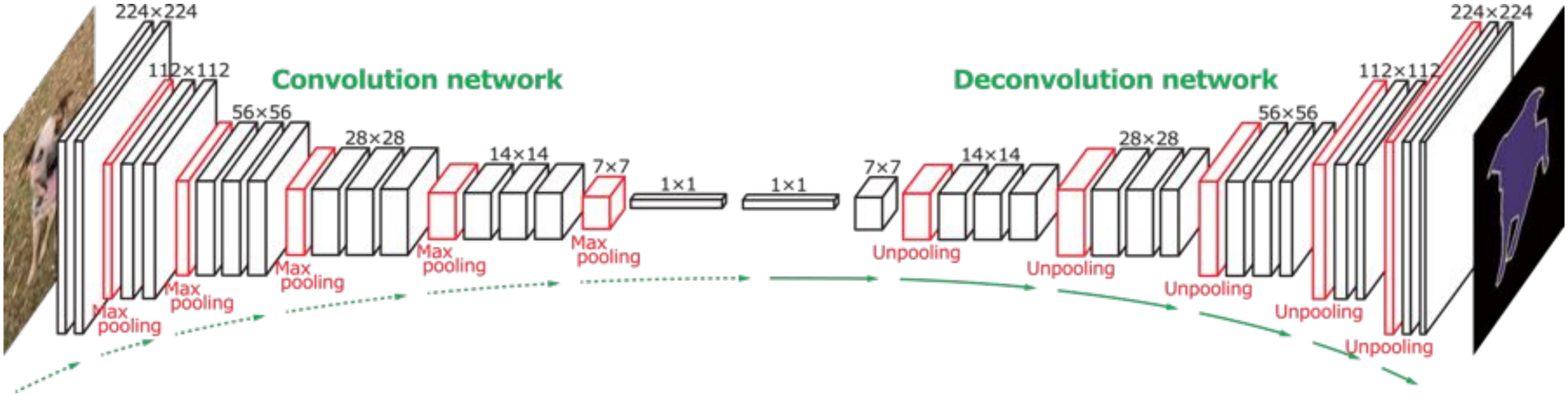
With No Regularization



With Regularization

How Deep? … How Wide

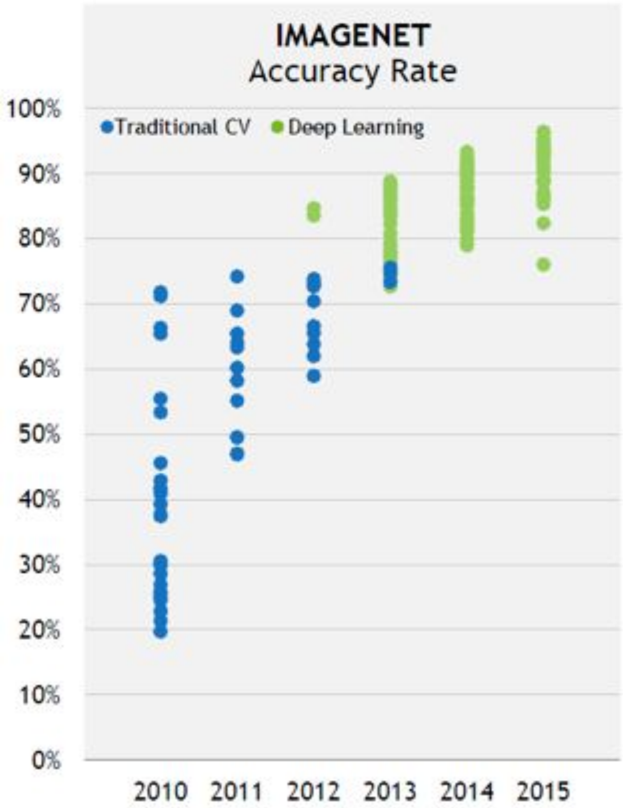
- Wide: How many neurons per layer?
- Deep: How many layers?

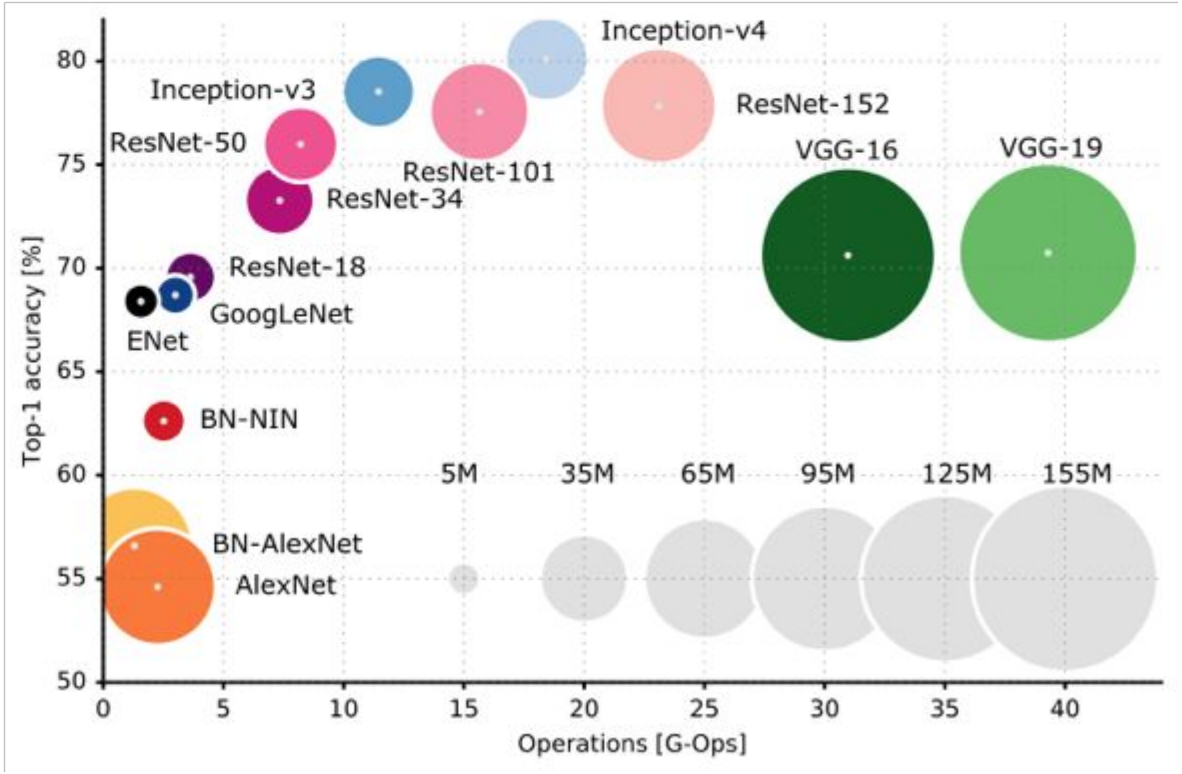


Things to consider

- How much data is available, and is it easy to augment?
- How much capacity for prediction or for a particular task we want, the more classes or objects in the training data, the more capacity we need.
- The complexity of the Task: classification, image reconstruction, object detection.
- How much computation power we need?
- How much generalization we want? Or in other words, under what conditions do we want the model to work well?

How crazy have we gone?





How crazy have we gone?

Relative computation per bach on P100 - Imagenet (224x224 RGB)				
Batch_size	Model demand (Gflops)	P100-Gflos-capacity	Forward-Backward pass time (sec)	Forward-Backward pass time (ms)
64	5	10600	0.030188679	30.18867925
64	10	10600	0.060377358	60.37735849
64	15	10600	0.090566038	90.56603774
64	20	10600	0.120754717	120.754717
64	25	10600	0.150943396	150.9433962
64	30	10600	0.181132075	181.1320755
64	35	10600	0.211320755	211.3207547
64	40	10600	0.241509434	241.509434

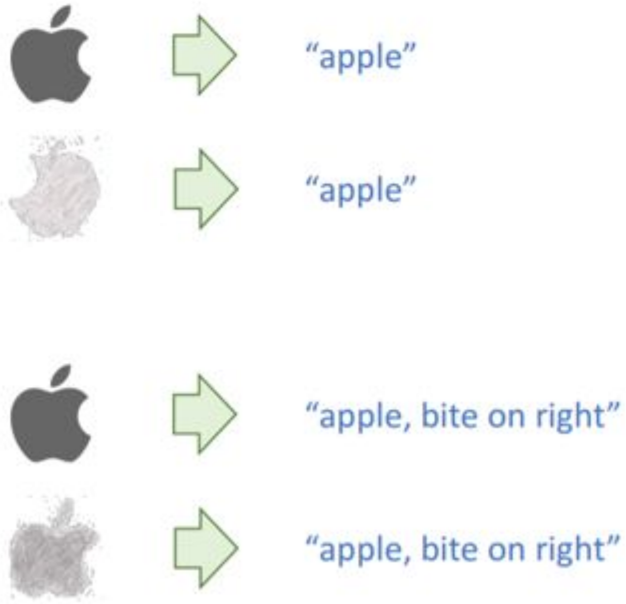
Relative computation per bach on V100 - Imagenet (224x224 RGB)				
Batch_size	Model demand (Gflops)	V100-Gflos-capacity	Forward-Backward pass time (sec)	Forward-Backward pass time (ms)
64	5	15700	0.020382166	20.38216561
64	10	15700	0.040764331	40.76433121
64	15	15700	0.061146497	61.14649682
64	20	15700	0.081528662	81.52866242
64	25	15700	0.101910828	101.910828
64	30	15700	0.122292994	122.2929936
64	35	15700	0.142675159	142.6751592
64	40	15700	0.163057325	163.0573248

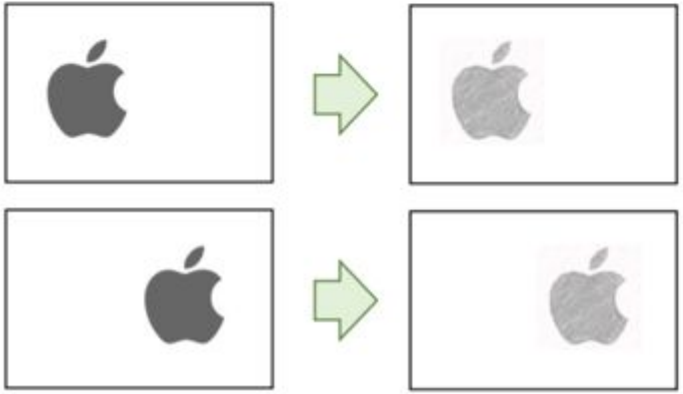
Invariance and Equivariance

- **Invariance**: is what we need when we perform classification, recognition. Random changes in the original input should not affect the output prediction significantly. In other words, the prediction of the original input should be the same even if I add some random changes in that same input. (very hard, and lots of people are working on this)

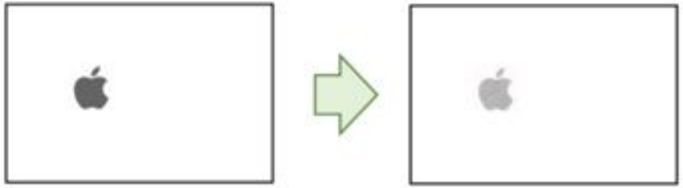
- **Equivariance**: this is a desirable feature for detection, segmentation. The changes in the input should be reflected in the prediction. (shape, translation, size, etc)

Invariance and Equivariance



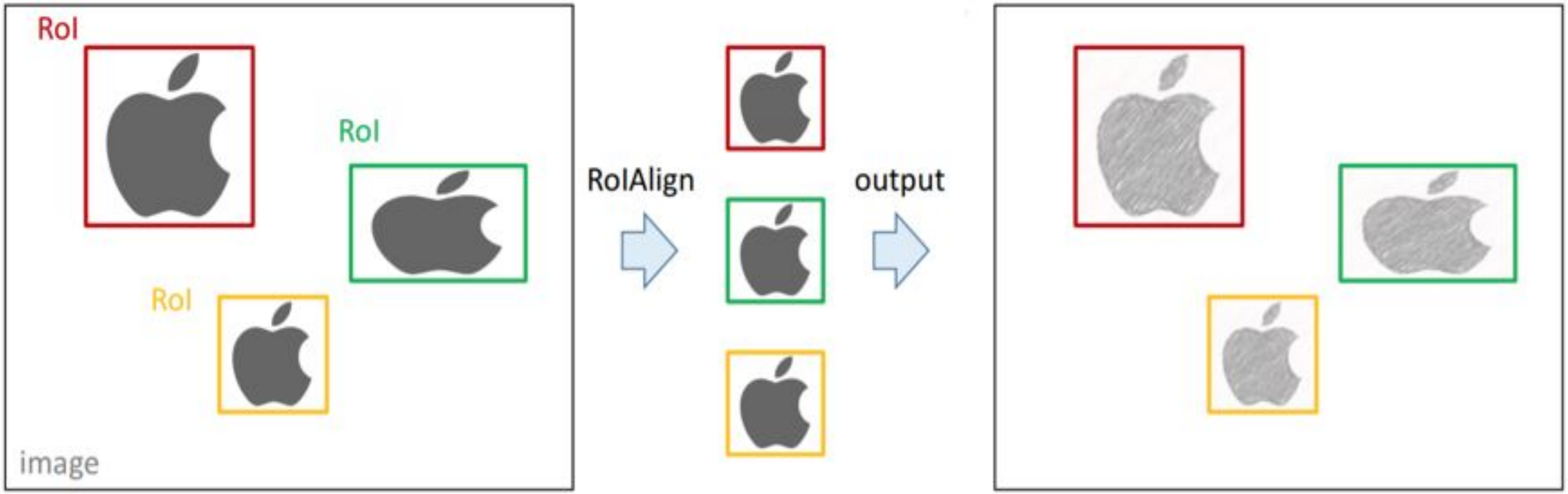


translation-equivariant



scale-equivariant

Invariance and Equivariance



Invariance and Equivariance

Invariance:

- Pooling layers: max, average, etc…
- Strided convolutions
- Fully Connected Layers
- More data with augmentation.

Equivariance

- Convolutional Layers: almost all of them !
- More data with augmentation

Training neural Networks

Okay, let’s do some training !

Final Thoughts

For years we have developed very powerful models, for very specific tasks, however a question has been always asked:
How do we combine all of the approaches to really get something more like a human being?

Short answer: one model for one task !

A more complicated answer: it is very hard to get all of these approaches to work together, simply because they all have errors, and one mistake in one of them could spell disaster for all of the others.

Final Thoughts

One model for everything:

- Attention is all you need
- YOLO
- NVDHM