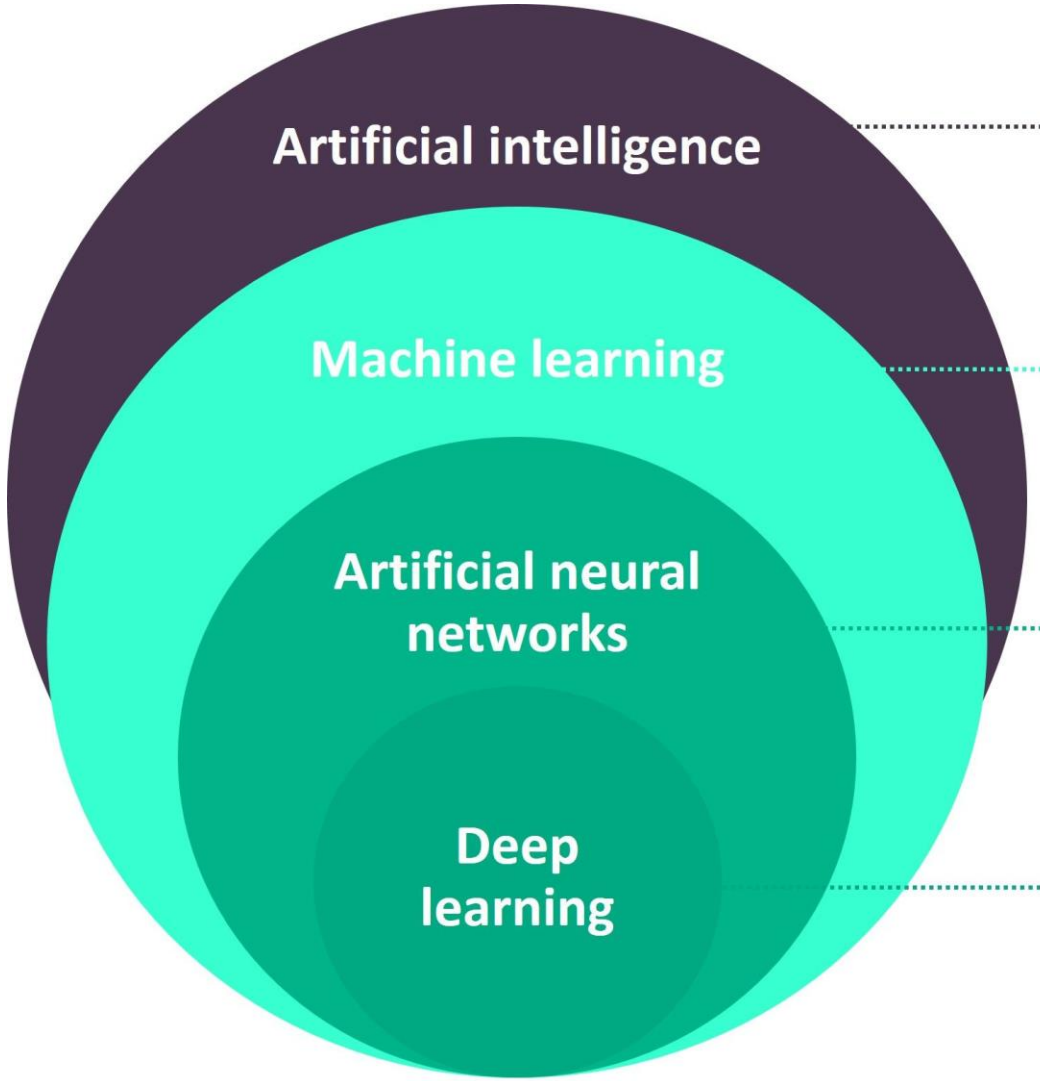
IA aplicada a SIG



Artificial intelligence (AI)

Any techniques that enable machines to solve a task in a way like humans do

Machine learning (ML)

Algorithms that allow computers to learn from examples without being explicitly programmed

Artificial neural networks (ANN)

Brain-inspired machine learning models

Deep learning (DL)

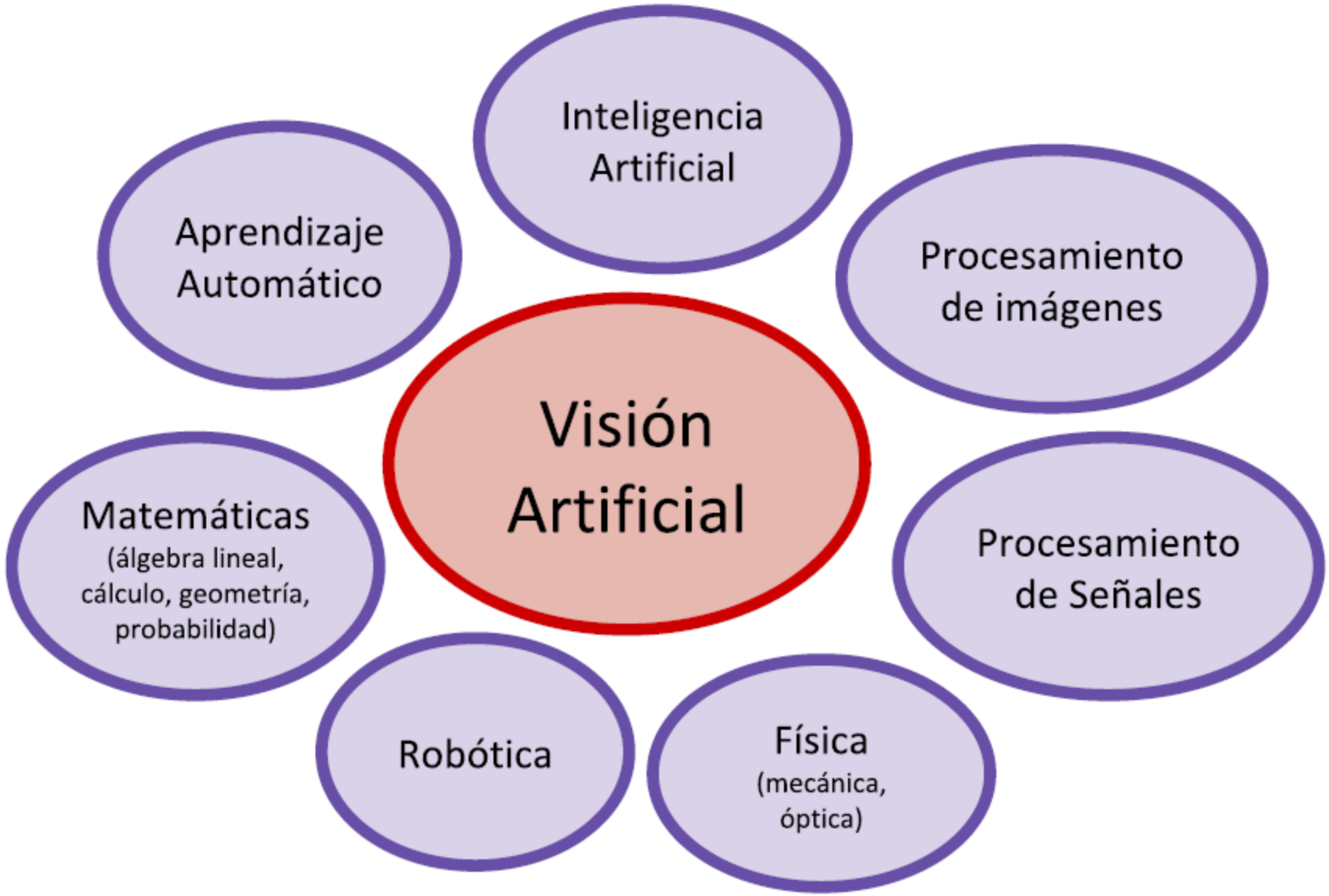
A subset of ML which uses deep artificial neural networks as models and automatically builds a hierarchy of data representations

Aprendizaje automático

“En ciencias de la computación el aprendizaje automático es una rama de la inteligencia artificial cuyo objetivo es desarrollar técnicas que permitan a las computadoras aprender”. (Wikipedia)

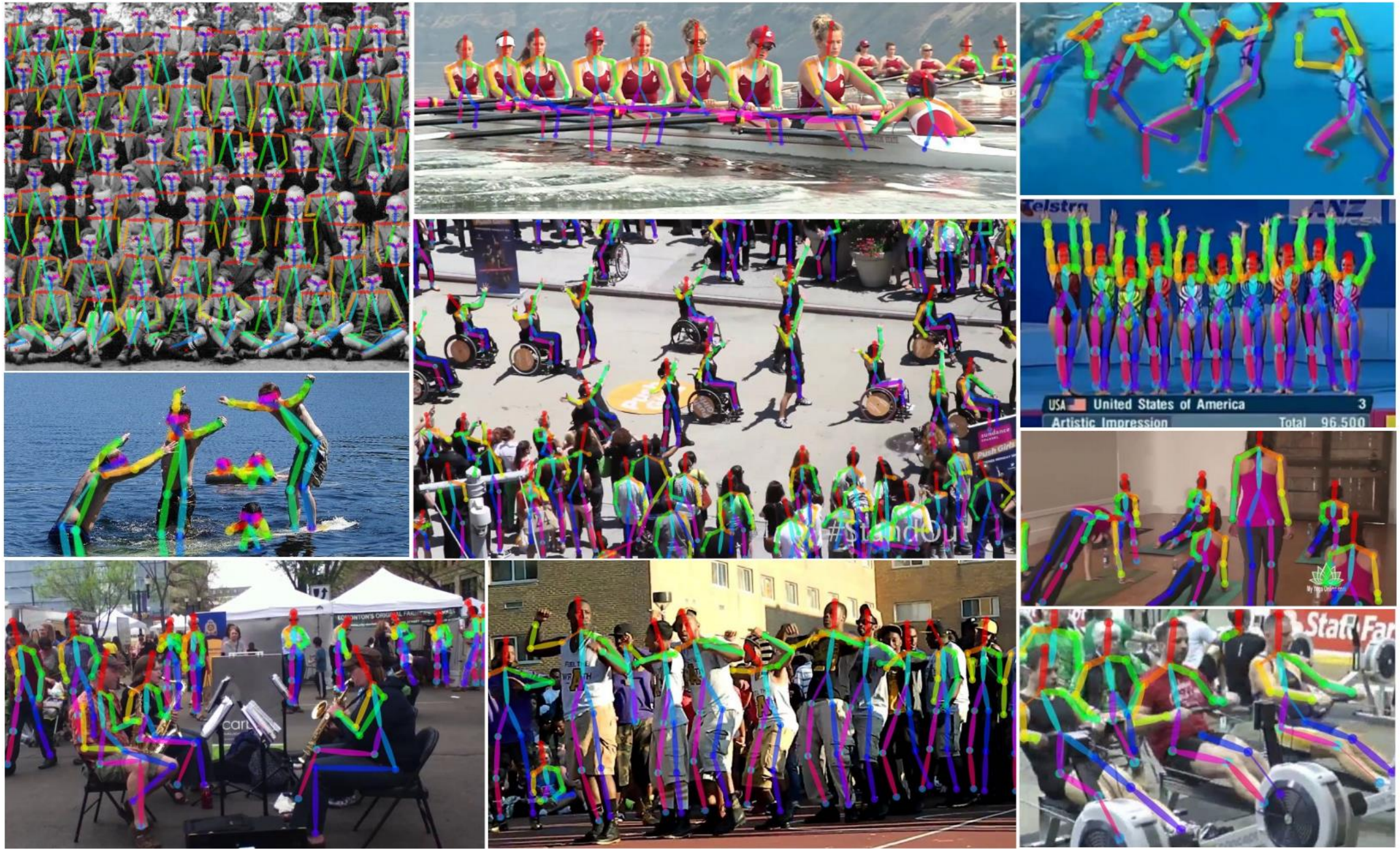
Visión Artificial

“La visión artificial es una disciplina científica que incluye métodos para adquirir, procesar, analizar y comprender las imágenes del mundo real con el fin de producir información numérica o simbólica para que puedan ser tratados por un computador”. (Wikipedia)


Inteligencia Artificial
Aprendizaje Automático
Procesamiento de imágenes
Visión Artificial
Matemáticas (álgebra lineal, cálculo, geometría, probabilidad)
Procesamiento de Señales
Robótica
Física (mecánica, óptica)

Viola and Jones. “Robust real-time face detection” IJCV 2004

Farabet, et al. “Learning hierarchical features for scene labeling” PAMI 2013



Cao, et al. “Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields” arXiv 2016

Formas de aprendizaje automático

Aprendizaje supervisado

- Predecir una salida dada una entrada.
- Regresión (continuo) o Clasificación (discreto)
- Se necesitan datos de entrenamiento con la salida especificada

Aprendizaje no supervisado

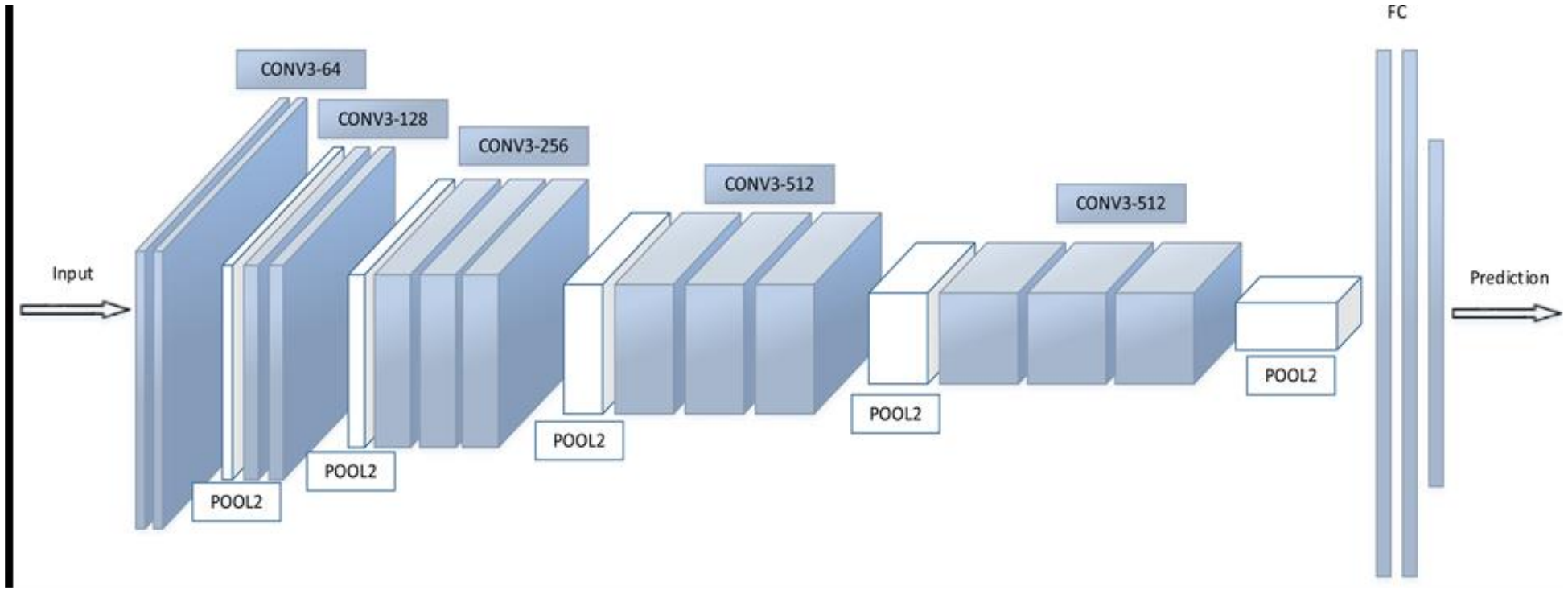
- Encontrar una buena representación de los datos.
- Encontrar estructura, patrones (explorar los datos)

Aprendizaje por refuerzos

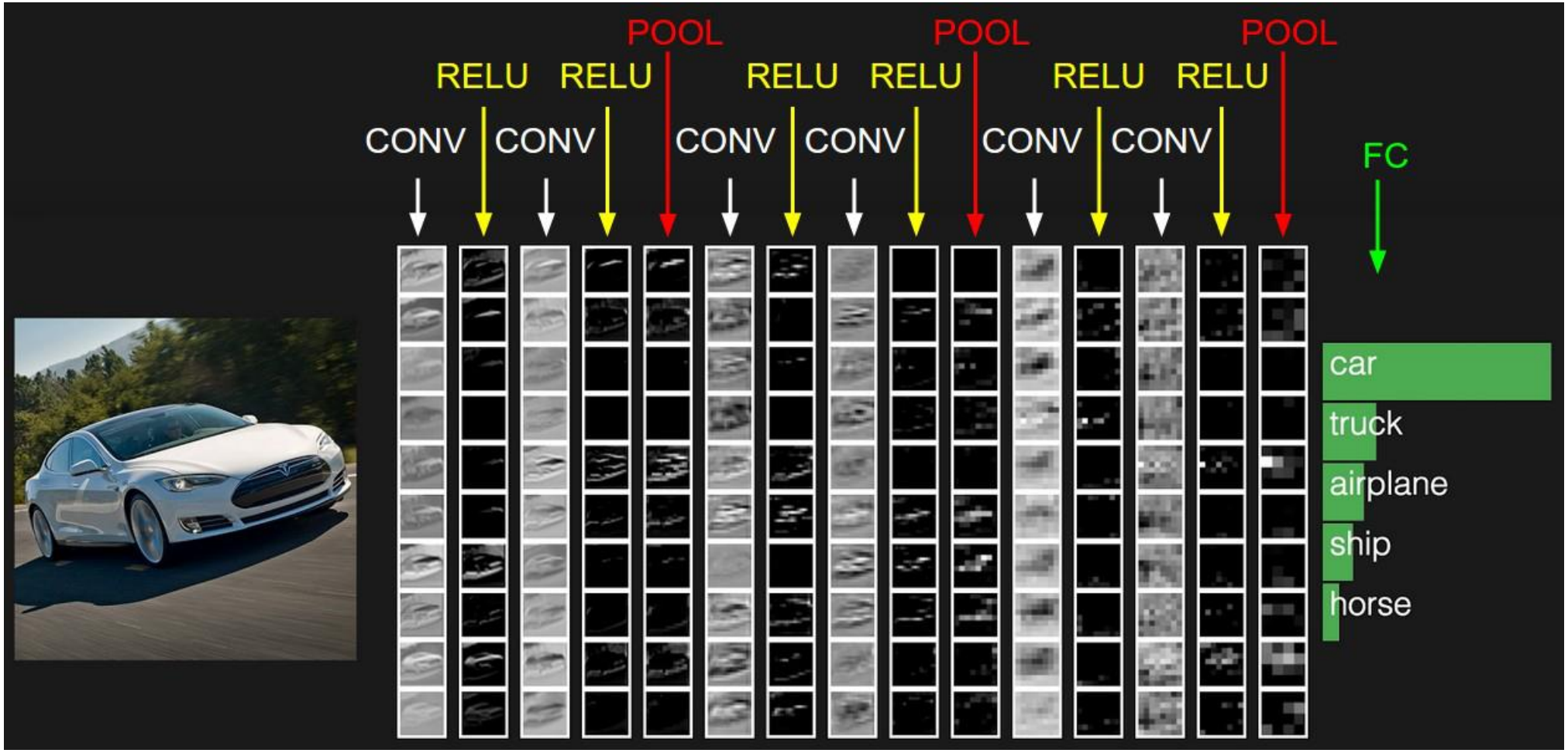
- Aprender a elegir *acciones* con el fin de maximizar alguna noción de recompensa acumulada.
- El programa interactúa con un entorno dinámico

Aprendizaje Profundo (Deep Learning)

Clase de representaciones paramétricas no lineales capaces de codificar las características del problema y de ser optimizadas de forma eficiente usando métodos de optimización estocástica.



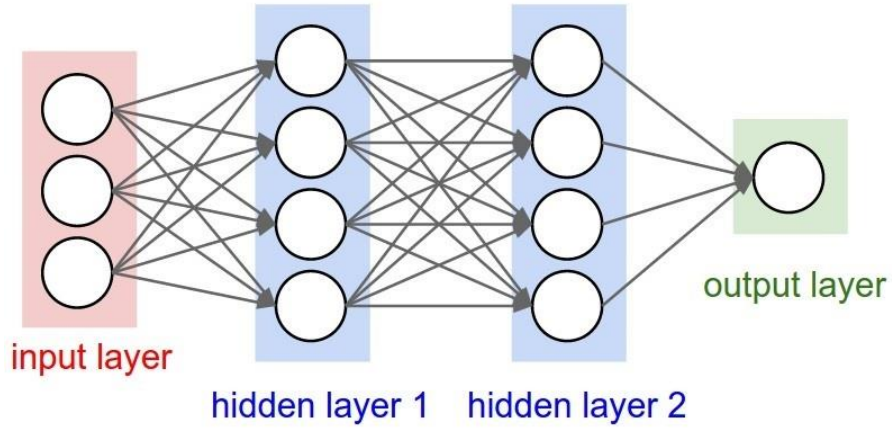
Aprendizaje Profundo (Deep Learning)

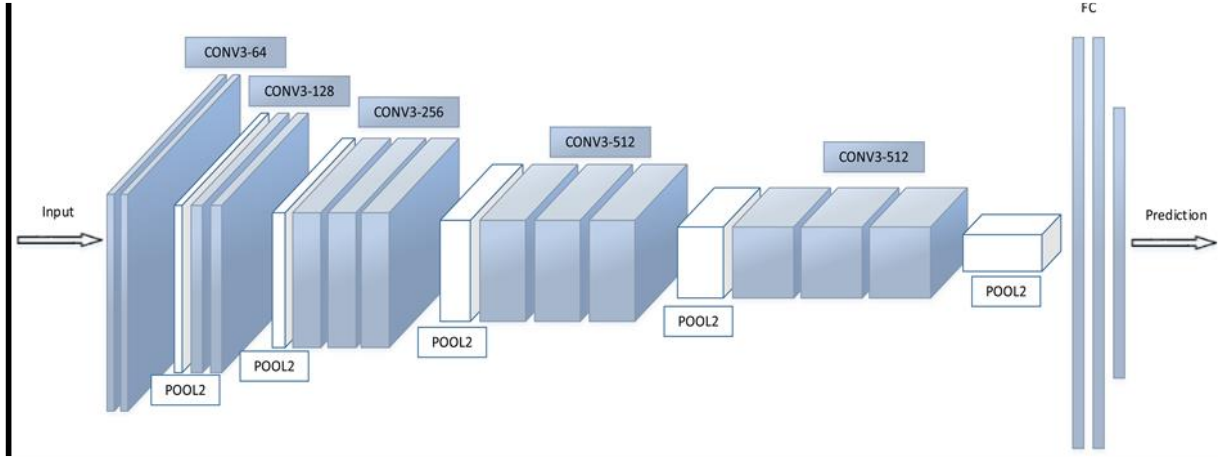


Aprendizaje Profundo (Deep Learning)

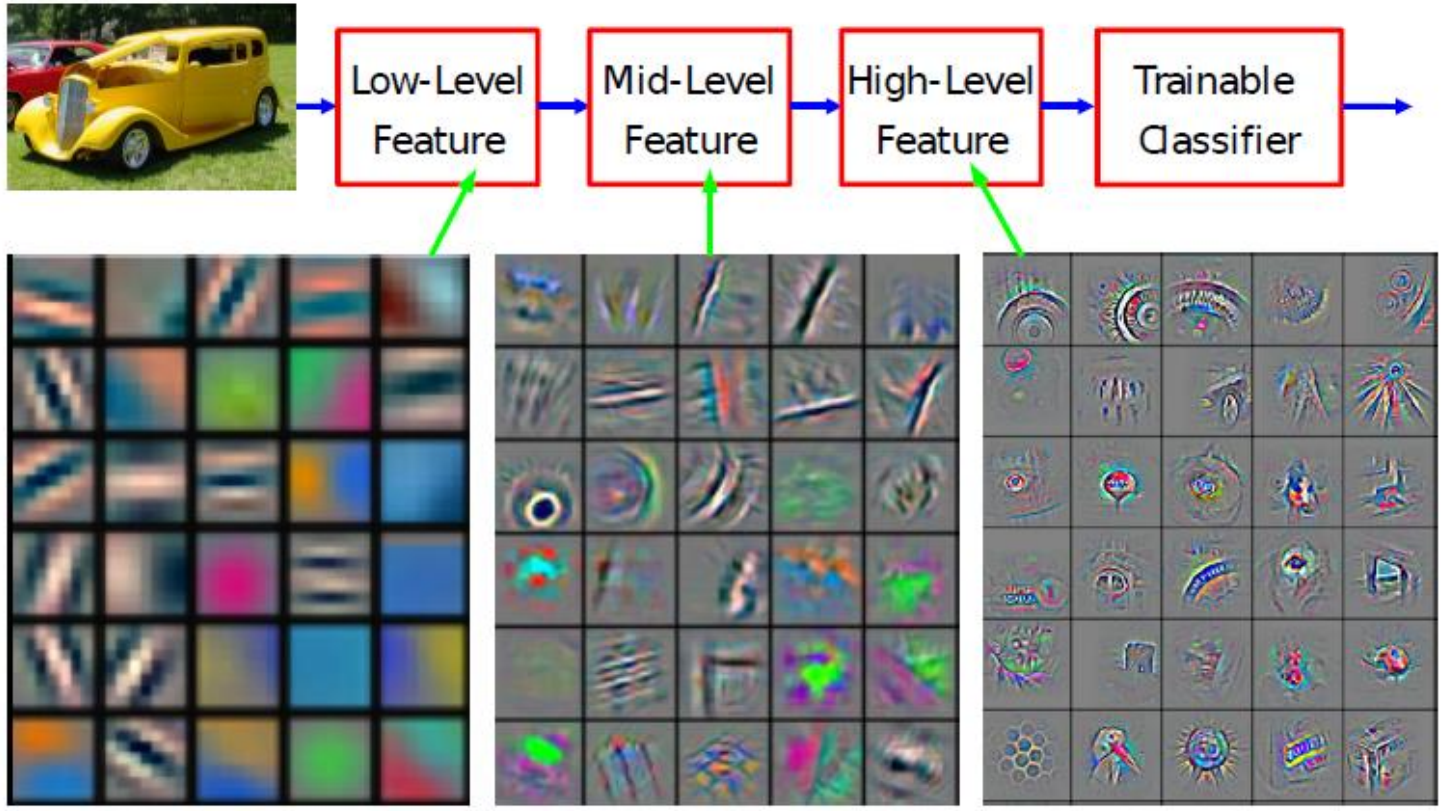
Cascada de capas de procesamiento no lineal para extraer y transformar variables.

- Cada capa usa la salida de la capa anterior como entrada:
- Transforma la representación anterior a otra de mayor nivel de abstracción
- Múltiples niveles de representación se corresponden con diferentes niveles de abstracción (jerarquía de conceptos).
- Representaciones de alto nivel son más globales y más invariantes
- Representaciones de bajo nivel son comunes a las distintas categorías





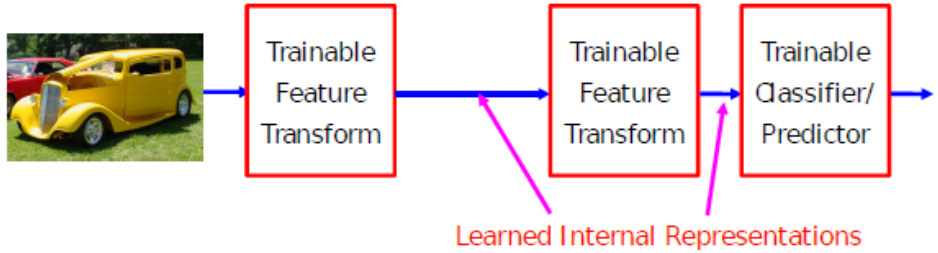
Deep Learning – representación de jerarquías adaptativas.



Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

Jerarquía de representaciones (capas) nivel creciente de abstracción

- Cada capa es una representación adaptiva (entrenable)



Imágenes:

- Pixel → bordes → partes → objetos

Texto:

- Caracteres → palabras → grupos de palabras → frases → historias

FeedForward

Sea una red con **L** capas ocultas.

- Entrada:

$$\mathbf{x} \quad (= \mathbf{h}^{(0)})$$

- Pre-activación capa (j):

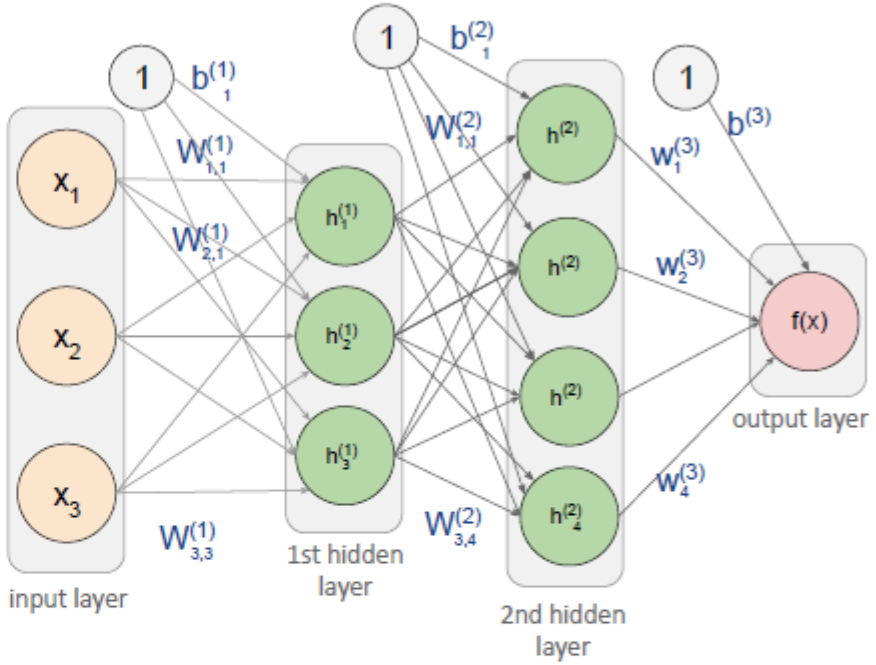
$$\mathbf{a}^{(j)}(\mathbf{x}) = \left(\mathbf{W}^{(j)T}\mathbf{h}^{(j-1)} + \mathbf{b}^{(j)}\right)$$

- Salida capa (j)

$$\mathbf{h}^{(j)}(\mathbf{x}) = g(\mathbf{a}^{(j)}(\mathbf{x}))$$

- Salida de la red (capa $L+1$)

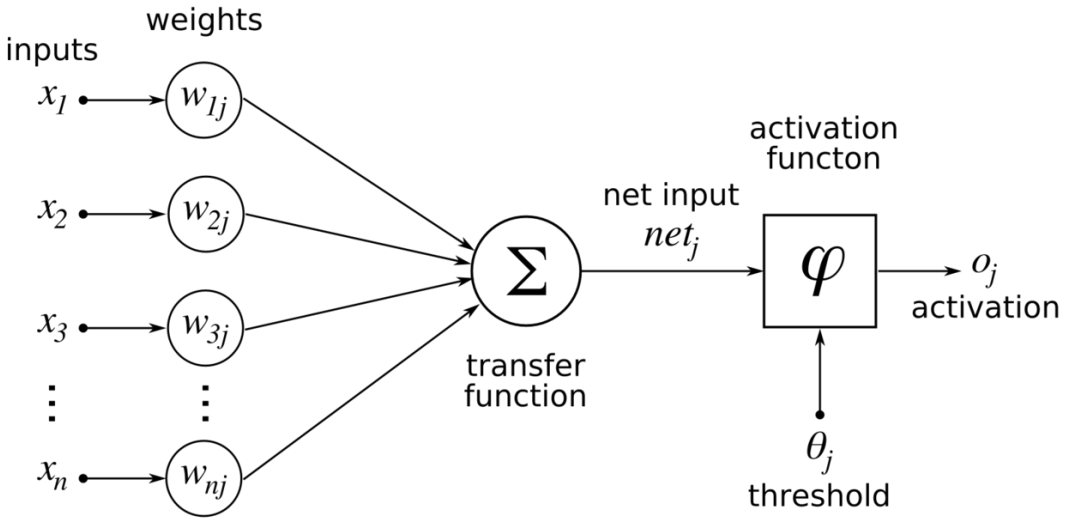
$$f(\mathbf{x}) = o\left(\mathbf{a}^{(L+1)}\right)$$



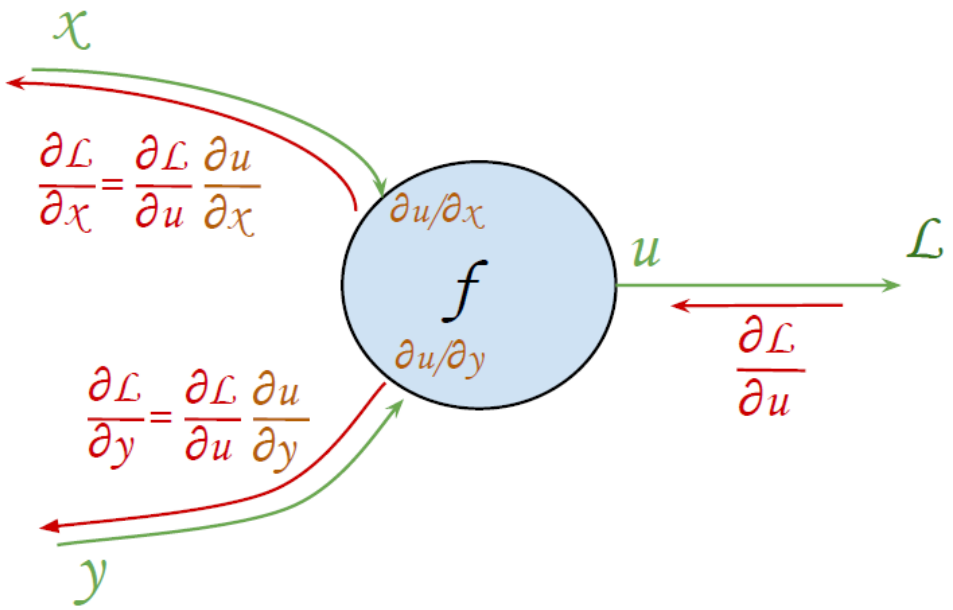
Nomenclatura:

- "3-layer neural net" o "2-hidden-layer neural net"
- Capas totalmente conectadas ("Fully-connected layers")

BackPropagation



Gradiente hacia abajo: Regla de la cadena



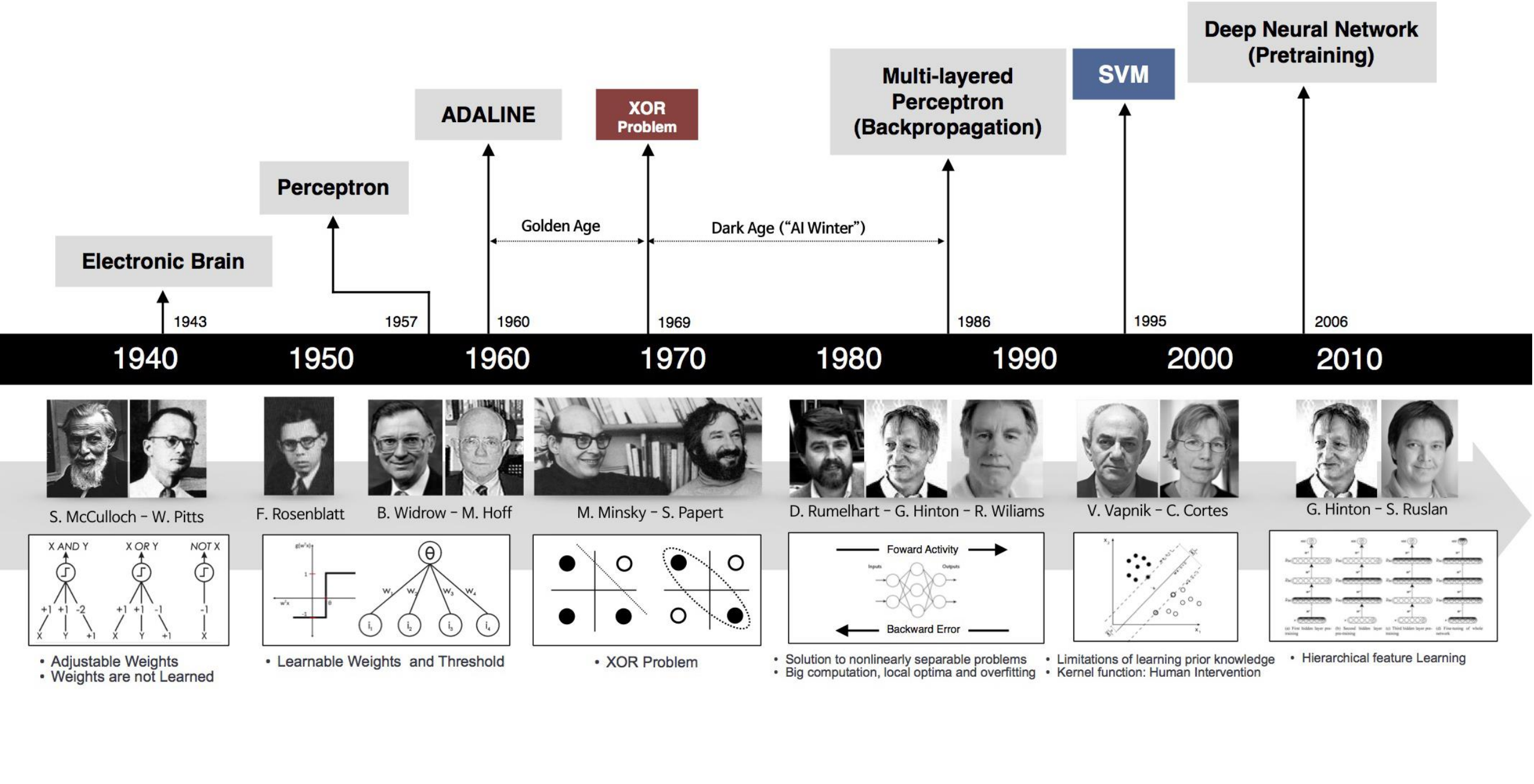
Breve historia

Perceptrón

Creada por Rosenblatt, 1957, en la universidad de Cornell.

- De las primeras “máquinas” capaces de “aprender”.
- Una única capa (capaz de separar datos linealmente separables)
- Modelo actual es esencialmente el mismo (pero multi-capa)

Rosenblatt, 1958


Deep Neural Network (Pretraining)
SVM
Multi-layered Perceptron (Backpropagation)
XOR Problem
ADALINE
Perceptron
Electronic Brain
Golden Age
Dark Age ("AI Winter")
1943
1957
1960
1969
1986
1995
2006
1940
1950
1960
1970
1980
1990
2000
2010
S. McCulloch - W. Pitts
F. Rosenblatt
B. Widrow - M. Hoff
M. Minsky - S. Papert
D. Rumelhart - G. Hinton - R. Wiliams
V. Vapnik - C. Cortes
G. Hinton - S. Ruslan
X AND Y
X OR Y
NOT X
Foward Activity
Backward Error
• Adjustable Weights
• Weights are not Learned
• Learnable Weights and Threshold
• XOR Problem
• Solution to nonlinearly separable problems
• Big computation, local optima and overfitting
• Limitations of learning prior knowledge
• Kernel function: Human Intervention
• Hierarchical feature Learning

Antes del 2012

Primera red convolucional entrenada con backpropagation

Communication, pages 41–46, November 1989. invited paper.

Handwritten Digit Recognition: Applications of Neural Network Chips and Automatic Learning

Y. Le Cun
L. D. Jackel
B. Boser
J. S. Denker
H. P Graf
I. Guyon
D. Henderson
R. E. Howard
W. Hubbard

THIS ARTICLE DESCRIBES TWO NEW METHODS for achieving handwritten digit recognition. The task of handwritten digit recognition was chosen for investigation not only because it has considerable practical interest, but because it is relatively well-defined and is sufficiently complex to constitute a meaningful test of connectionist methods.

Simple classification techniques applied to pixel images do not provide high recognition rates because systems based on these techniques contain little prior knowledge about the to-

is highly test-set dependent. A system may successfully recognize 99% of test data consisting of well-formed digits but score only 80% when confronted with the poorly-formed digits that are both routinely produced and easily recognized by people. We choose to perform our experiments on a rather difficult data set: isolated handwritten digits that were taken from postal zip codes. The zip code images were collected by the U.S. Postal Service from envelopes that passed through the Buffalo, NY Post Office. A postal service contractor converted the original zip code images to binary images, and segmented the dig-

Le Cun, Yann, L. D. Jackel, B. Boser, J. S. Denker, H. P. Graf, I. Guyon, D. Henderson, R. E. Howard, and W. Hubbard. "Handwritten digit recognition: Applications of neural network chips and automatic learning." IEEE Com. Mag. 1989

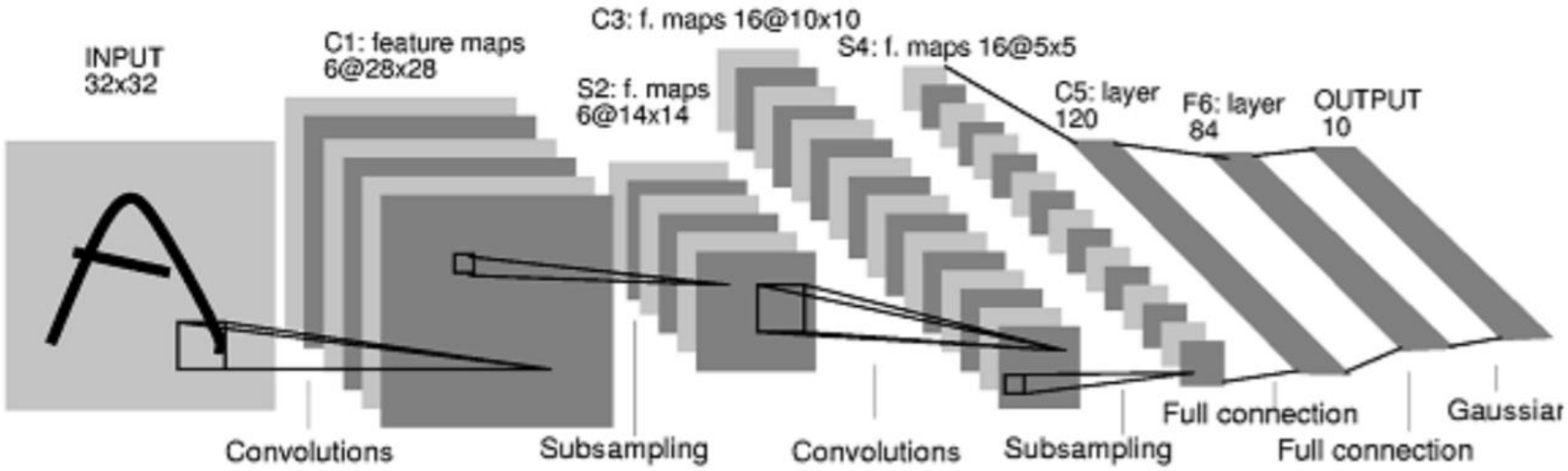
Antes del 2012

Primera red convolucional entrenada con backpropagation

- En los ’90 resolvía problemas en serio (reconocimiento de dígitos)
- En otros dominios era también competitiva, pero no se uso en producción

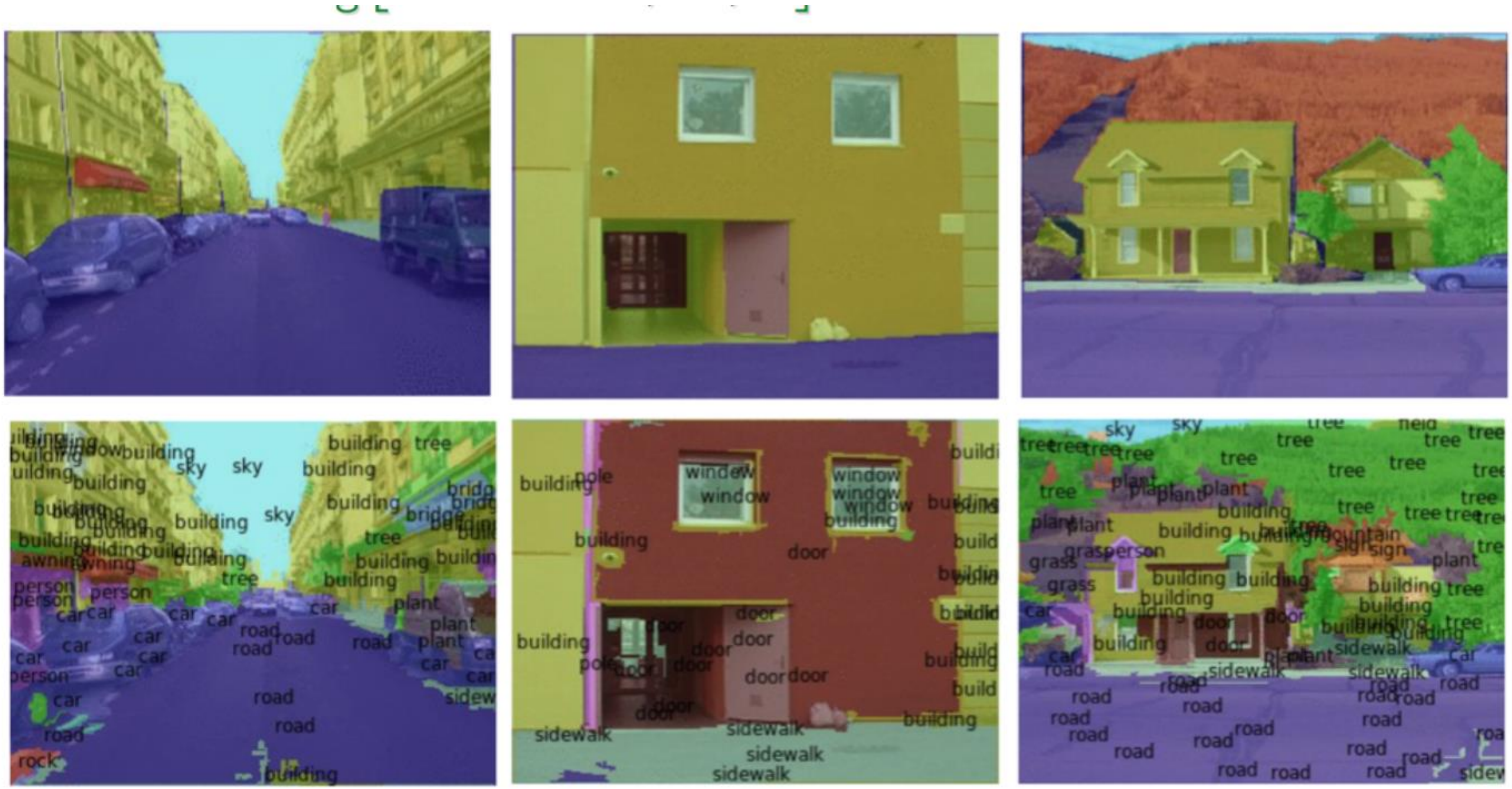
1998

LeCun et al.



LeNet-5

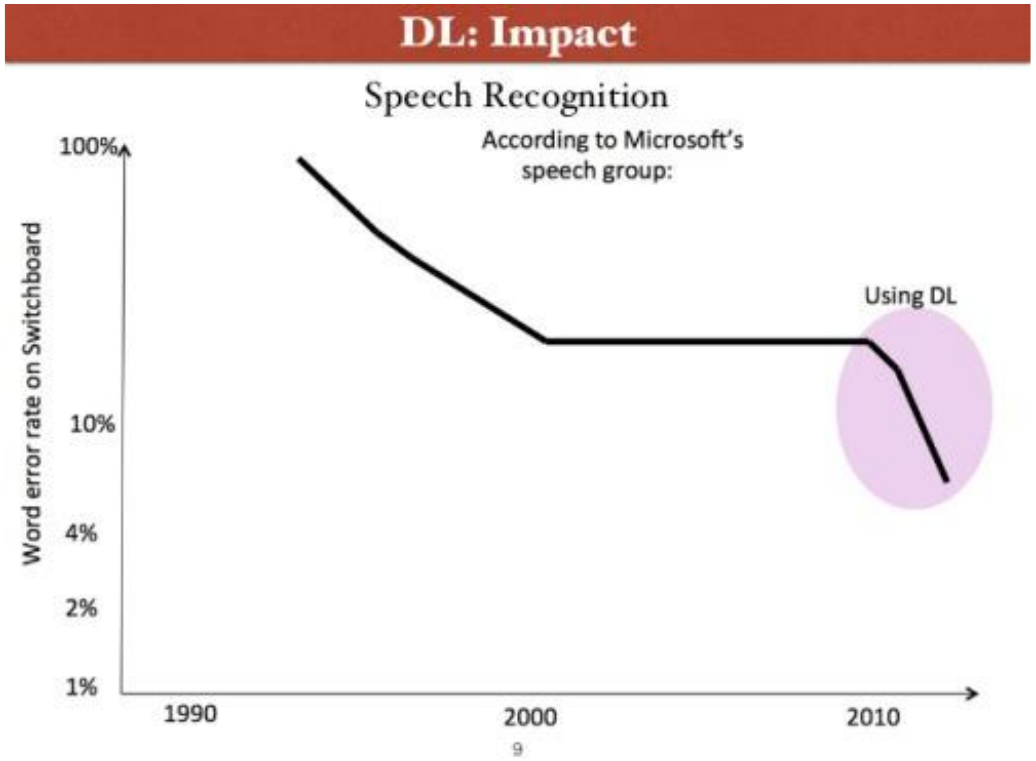
2012



Segmentacion de escenas [Farabet et al, 2012]

Primer Break-through: reconocimiento de voz

- Nuevo método en base a redes profundas produce buenos resultados
- Incluido en Android en 2012.



Mohamed, Abdel-rahman, George Dahl, and Geoffrey Hinton. "Deep belief networks for phone recognition." In NIPS workshop, 2009.

ImageNet Large Scale Visual Recognition Competition

Competencia de reconocimiento de objetos

- 1000 categorías de imágenes; 1.4 millones de imágenes de entrenamiento
- Respuesta correcta entre las primeras 5 imágenes.
- Krizhevsky, Sutskever, Hinton, 2012 [Supervision/AlexNet].

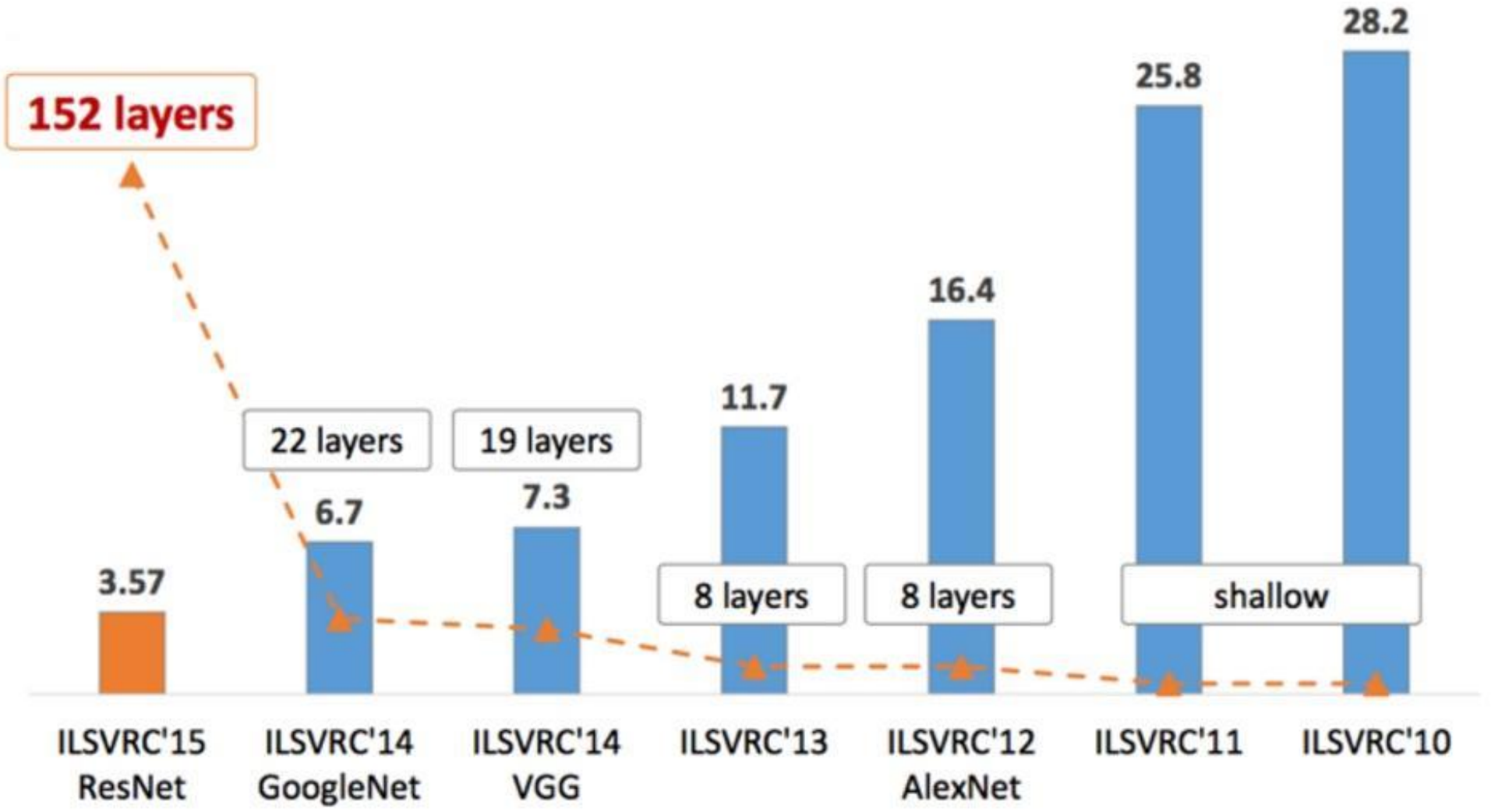
2012 Teams	%error
Supervision (Toronto)	15.3
ISI (Tokyo)	26.1
VGG (Oxford)	26.9
XRCE/INRIA	27.0
UvA (Amsterdam)	29.6
INRIA/LEAR	33.4

2013 Teams	%error
Clarifai (NYU spinoff)	11.7
NUS (singapore)	12.9
Zeiler-Fergus (NYU)	13.5
A. Howard	13.5
OverFeat (NYU)	14.1
UvA (Amsterdam)	14.2
Adobe	15.2
VGG (Oxford)	15.2
VGG (Oxford)	23.0

2014 Teams	%error
GoogLeNet	6.6
VGG (Oxford)	7.3
MSRA	8.0
A. Howard	8.1
DeeperVision	9.5
NUS-BST	9.7
TTIC-ECP	10.2
XYZ	11.2
UvA	12.1

Técnicas que usan CNN - Otros métodos

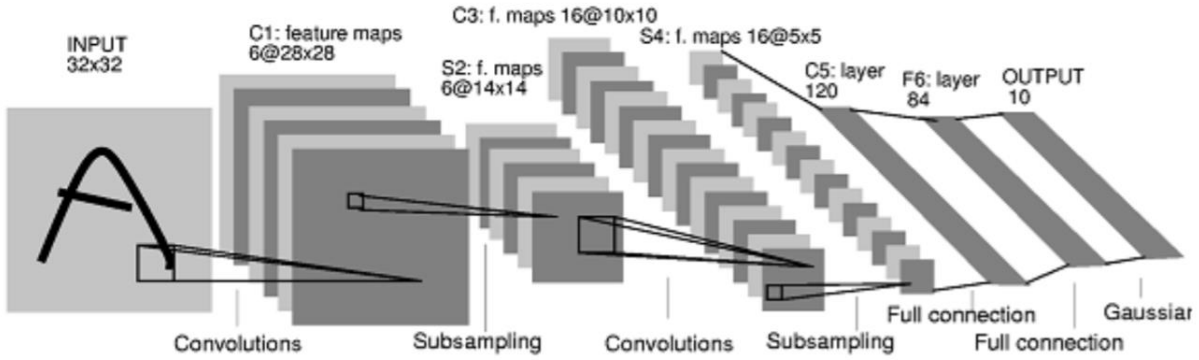
ImageNet Large Scale Visual Recognition Competition



Evolución de las arquitecturas

1998

LeCun et al.



# of transistors

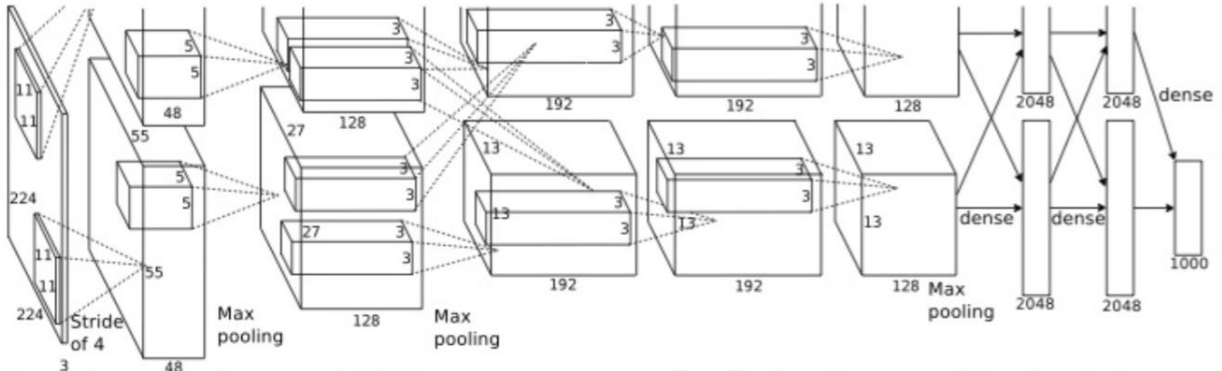
10^6

# of pixels used in training

10^7 NIST

2012

Krizhevsky et al.



# of transistors GPUs

10^9

# of pixels used in training

10^{14} IMAGENET

Evolución de las arquitecturas

Year 2010

NEC-UIUC

Dense grid descriptor: HOG, LBP

Coding: local coordinate, super-vector

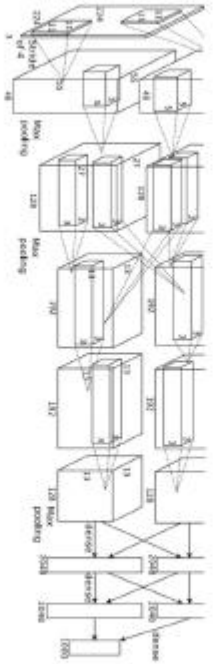
Pooling, SPM

Linear SVM

[Lin CVPR 2011]

Year 2012

SuperVision



[Krizhevsky NIPS 2012]

Year 2014

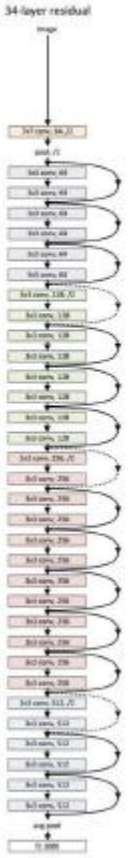
GoogLeNet

VGG

[Szegedy arxiv 2014] [Simonyan arxiv 2014]

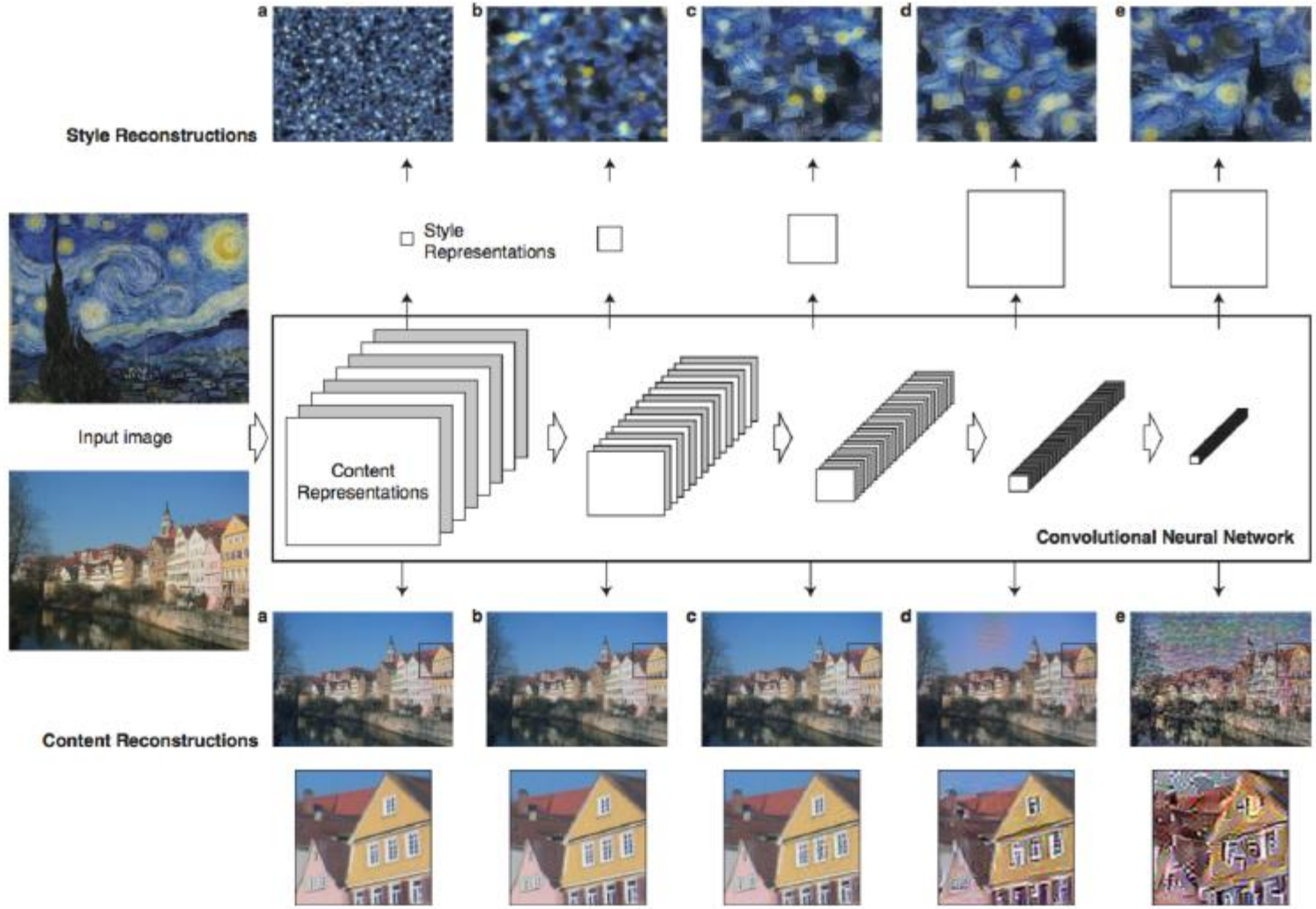
Year 2015

Res-nets

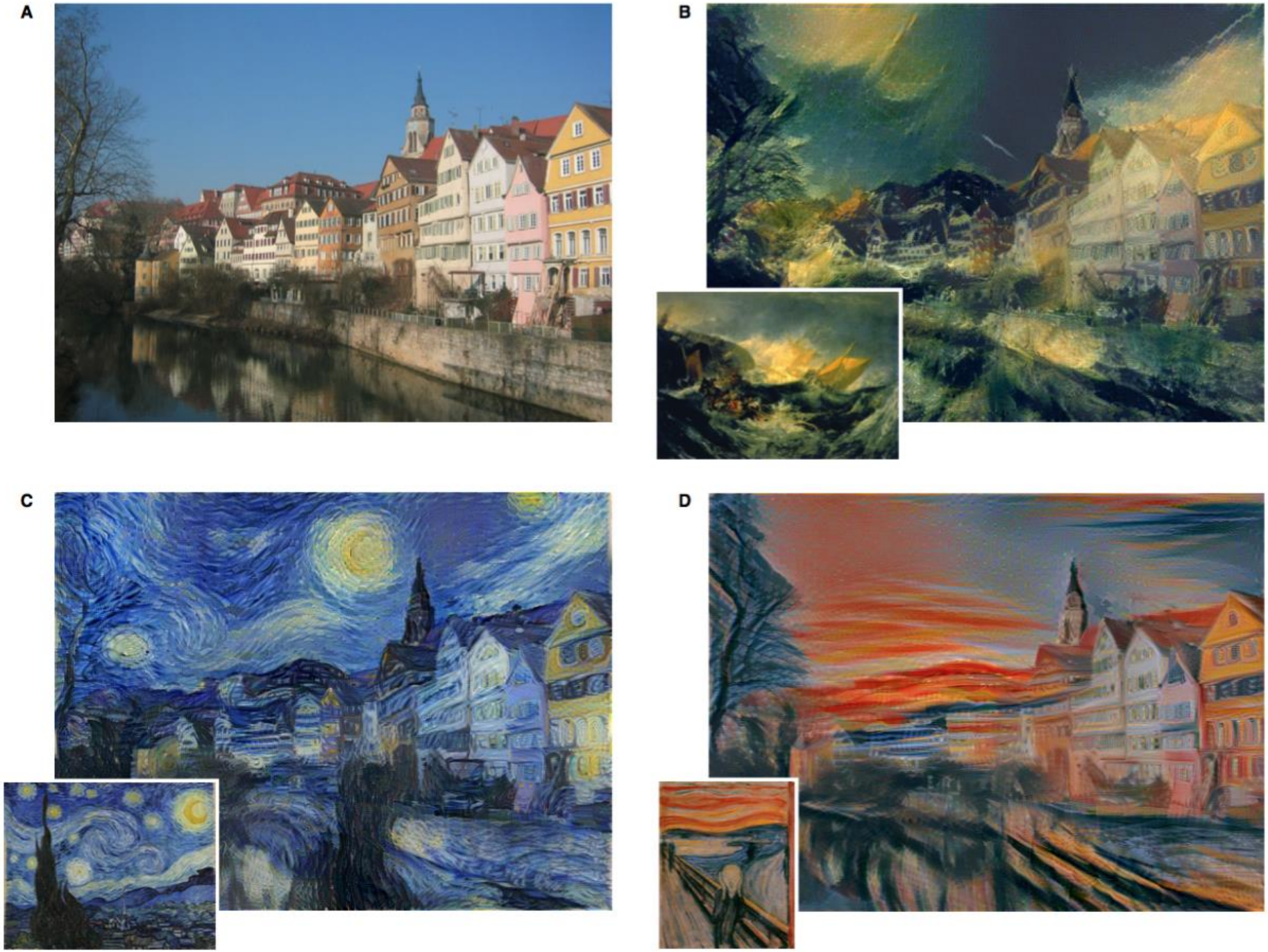


[He et al 2015]

- Separar textura de estilo en imágenes

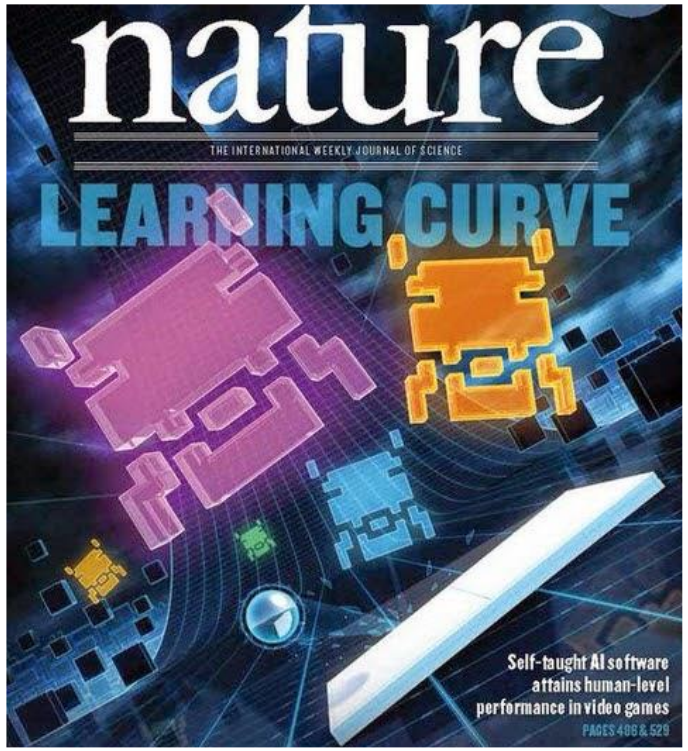


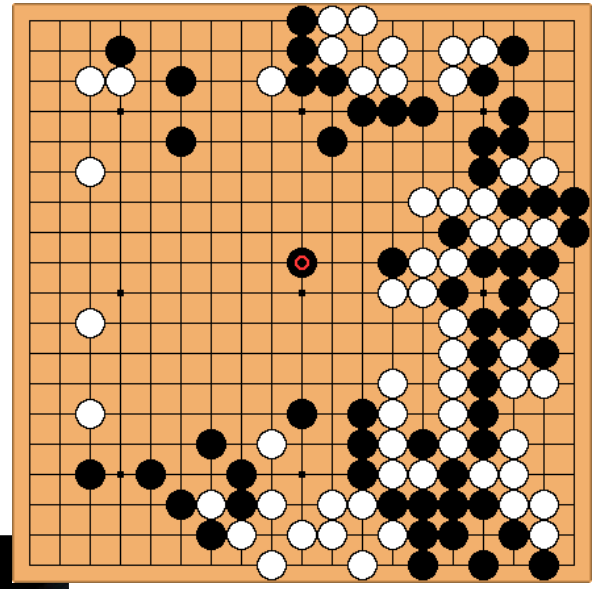
Gatys et al. 2015


A
B
C
D

Aprendizaje por refuerzo

DeepMind: Atari games y Alpha Go



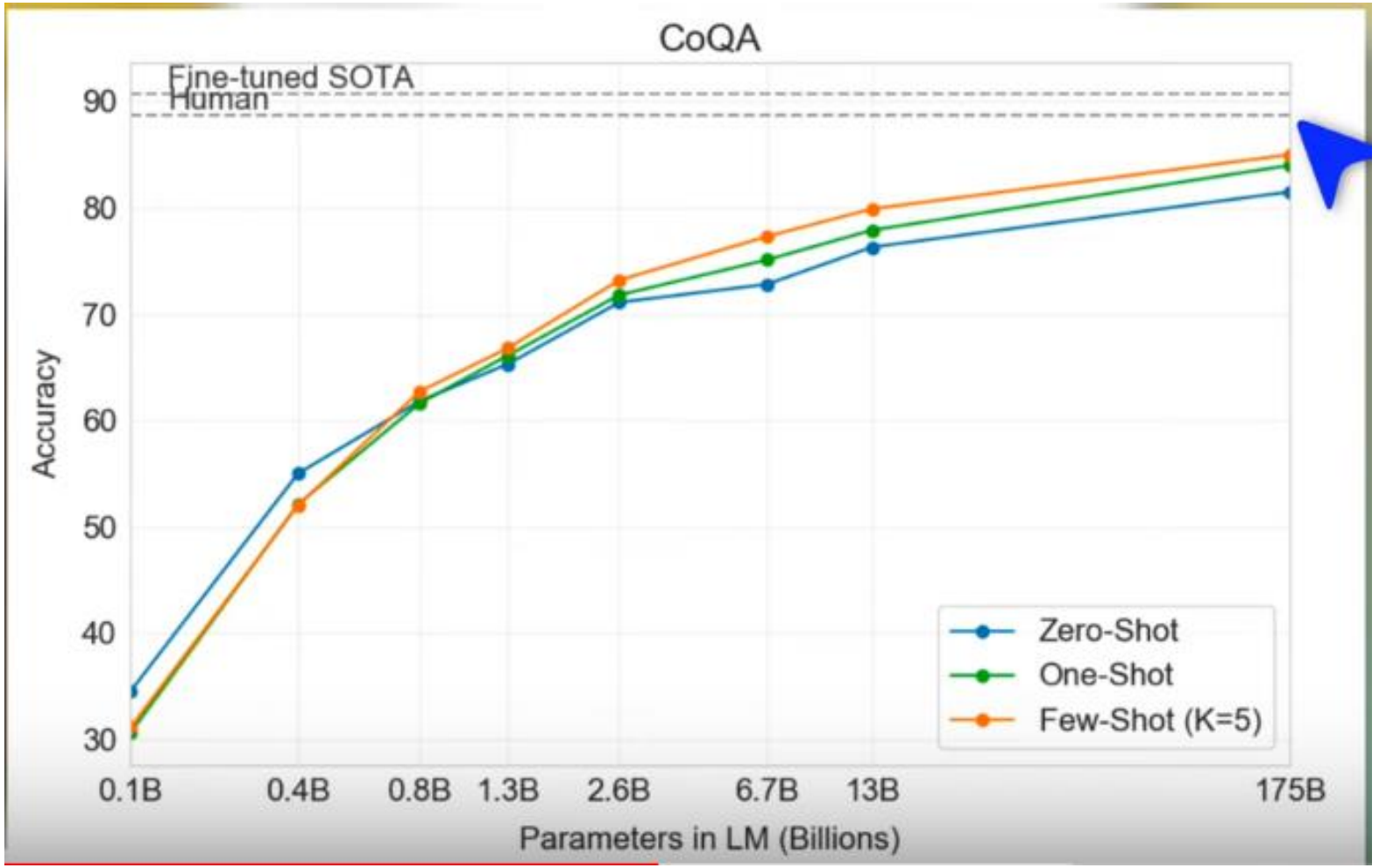


https://www.youtube.com/watch?v=WXuK6gekU1Y

GPT-3

- OpenAI
- 175000 M de parámetros
- Julio 2020

https://arxiv.org/pdf/2005.14165.pdf

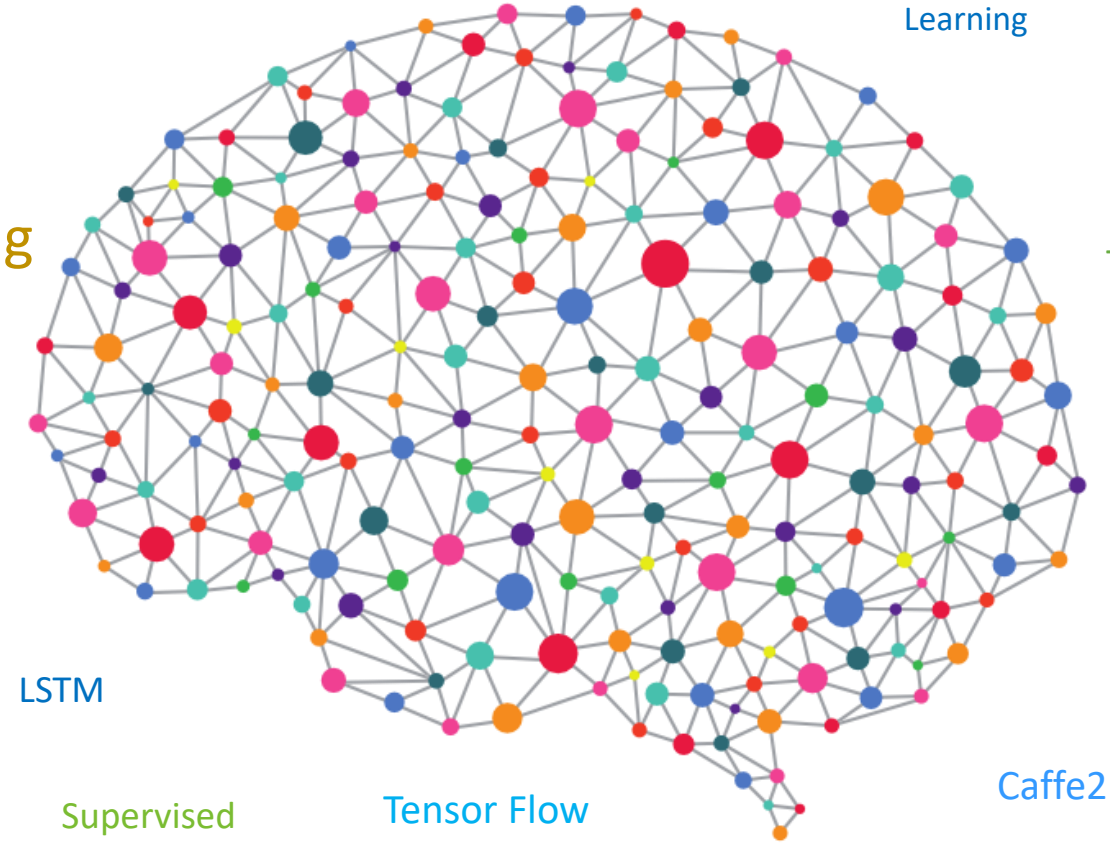


Gshard

- Google
- 600000 M de parámetros
- Julio 2020

https://arxiv.org/pdf/2006.16668.pdf

Software



Artificial Intelligence

Machine Learning

Random Forest

Linear Regression

Unsupervised Learning

PyTorch

SKLearn

Theano

Keras

Google Cloud ML

MonkeyLearn

Microsoft Azure ML Studio

Matlab

Caffe2

CNTK

Tensor Flow

TPU

Supervised Learning

LSTM

Reinforcement Learning

CNN

SVM

Data Science

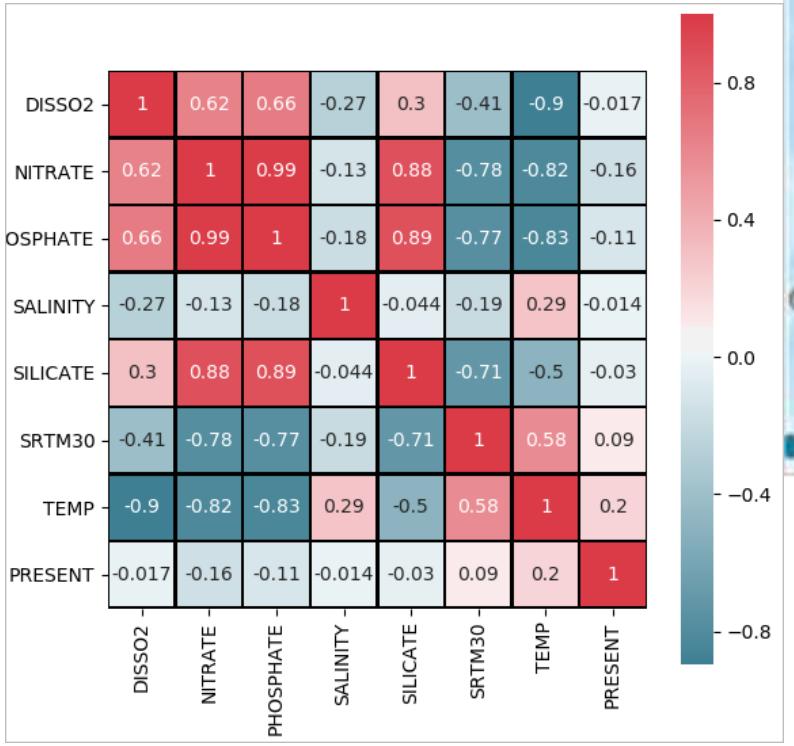
Deep Learning

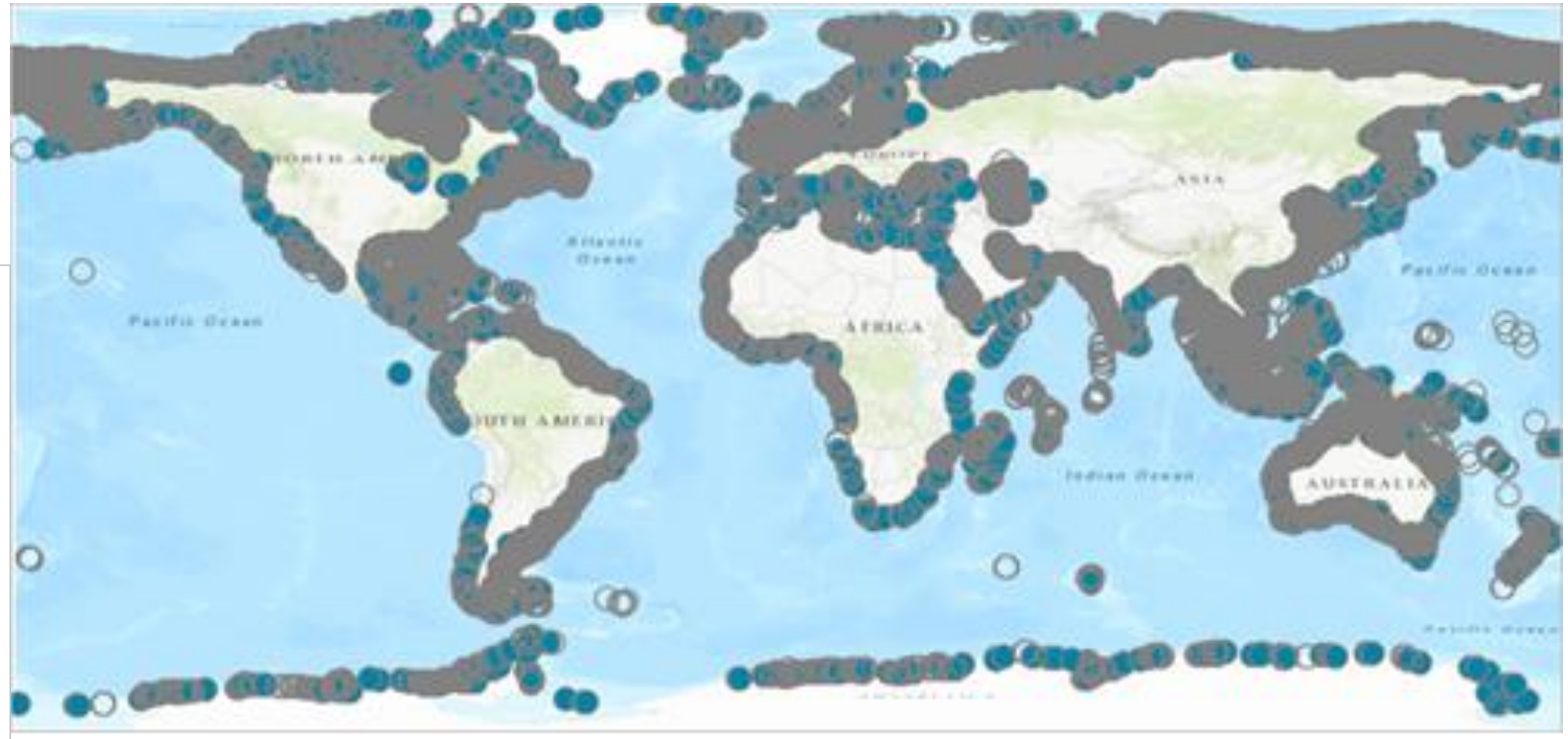
Ejemplos de ML usado para predicción tomando en cuenta la variable espacial

- Seagrass (ESRI)
- Elecciones de un país

Seagrass sample ESRI

- Random Forest





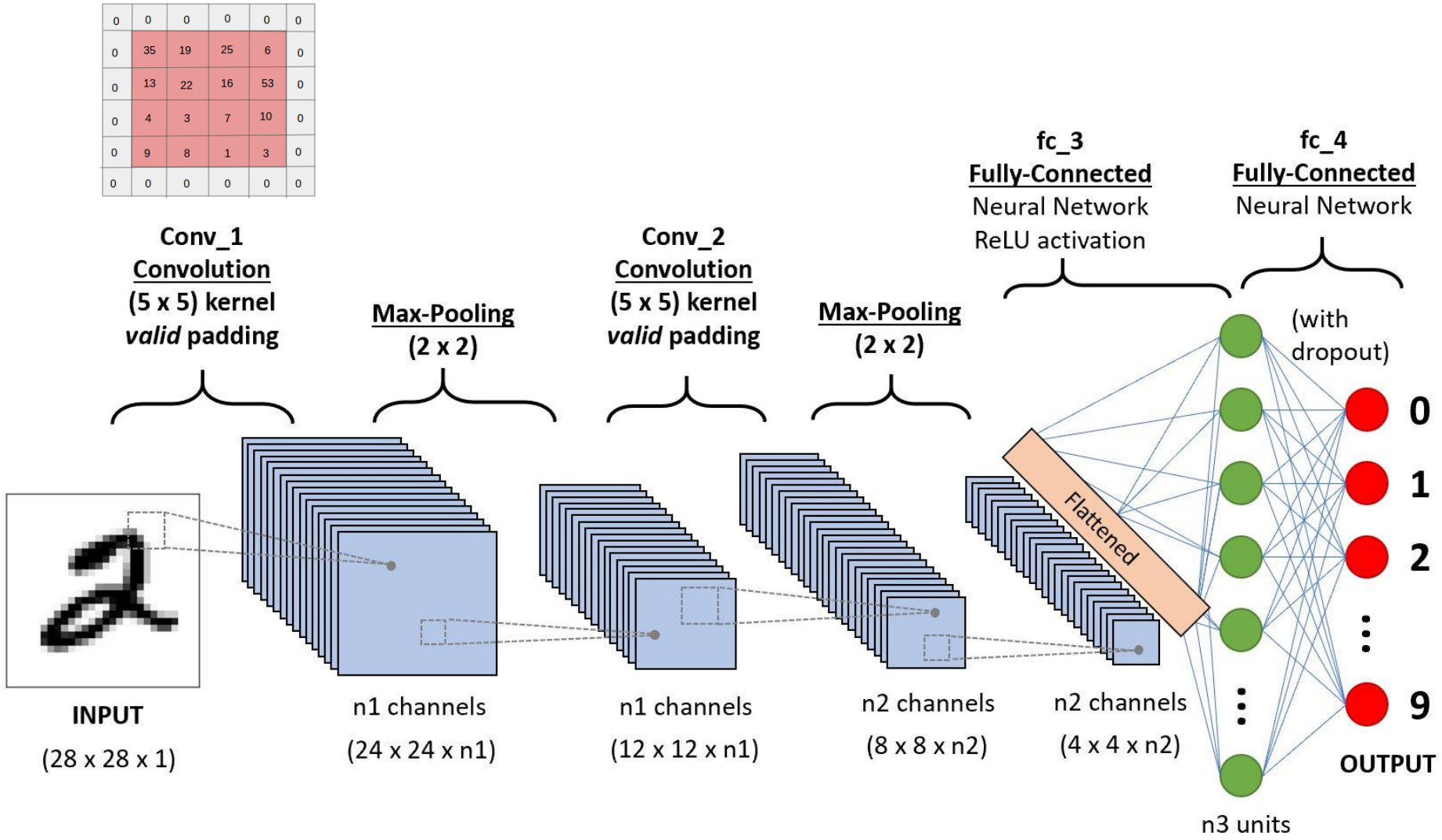

```
predictVars = ['DISSO2', 'NITRATE', 'PHOSPHATE', 'SALINITY', 'SILICATE', 'SRTM30', 'TEMP']
classVar = ['PRESENT']
allVars = predictVars + classVar

trainFC = DA.FeatureClassToNumPyArray(inputFC, ["SHAPE@XY"] + allVars)
spatRef = ARCPY.Describe(inputFC).spatialReference
```

```
rfco = RandomForestClassifier(n_estimators = 500, oob_score = True)
rfco.fit(train_set[predictVars], indicator)
```

IA sobre gráficos – Deep Learning

CNN – convolutional neurtal network



IA de ESRI

https://medium.com/geoai/integrating-deep-learning-with-gis-70e7c5aa9dfe

https://medium.com/geoai/how-we-did-it-end-to-end-deep-learning-in-arcgis-dd5b10d87b8

Casos de uso de tecnología.

En particular basada en RCNN, etc.)

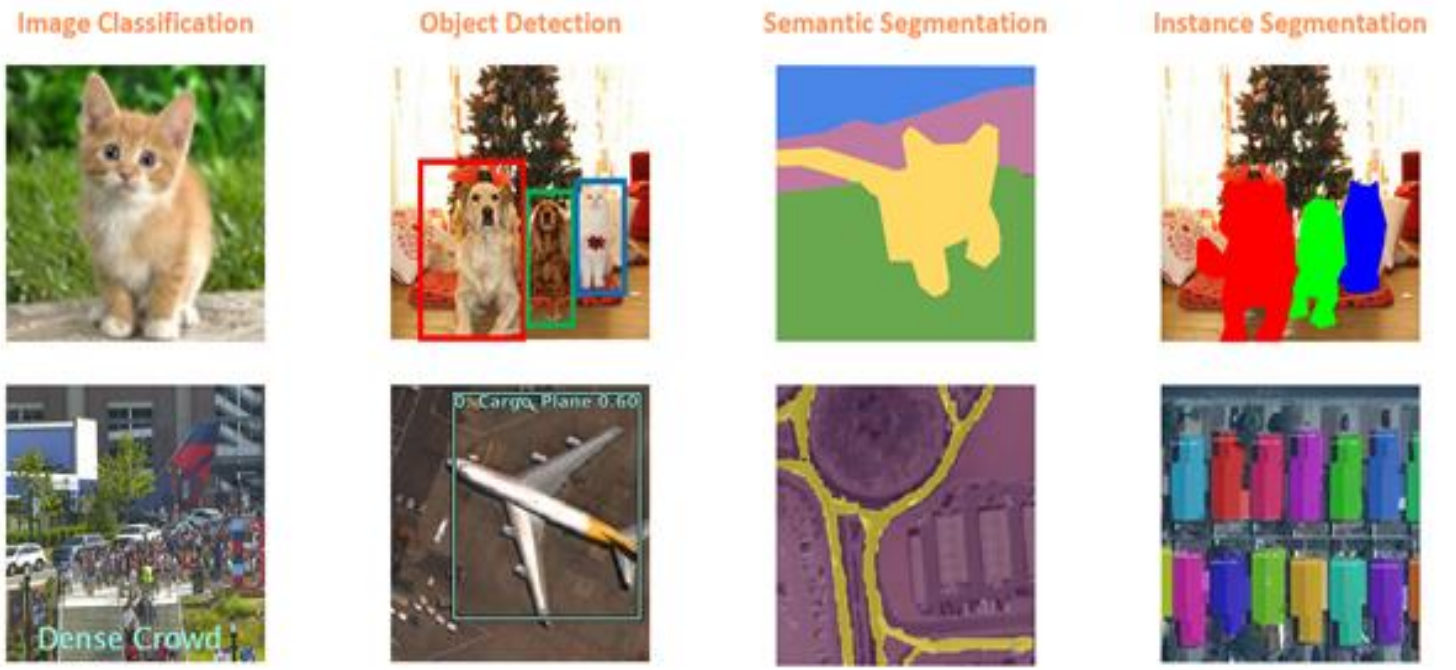
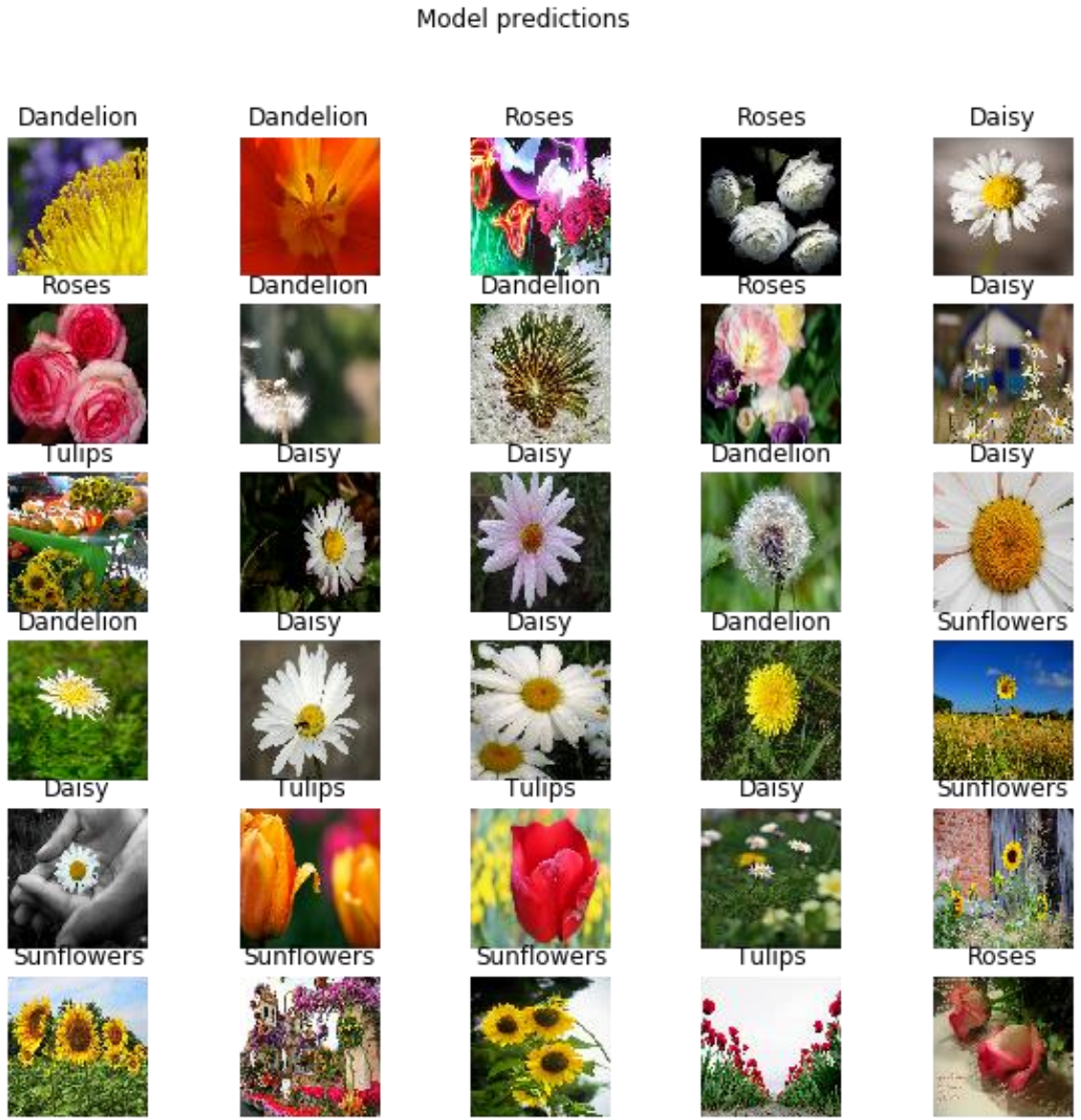


Image Classification

- Es el caso en que dada una imagen, se clasifica esta en base a su contenido.
- Ejemplo: imágenes que contiene una flor se puede determina el nombre de la flor (https://www.tensorflow.org/tutorials/images/hub_with_keras)



Posibles aplicaciones

- Previamente entrenando a la red con las imágenes apropiadas
- Detección de condiciones particulares y estado de objetos
- Clasificación de patrones en imágenes si esta solo incluye un patrón solo en toda la imagen.
- Reconocer situaciones particulares en las vías de transporte: ejemplo, manifestación, aumento de peatones, choque de vehículos, etc.

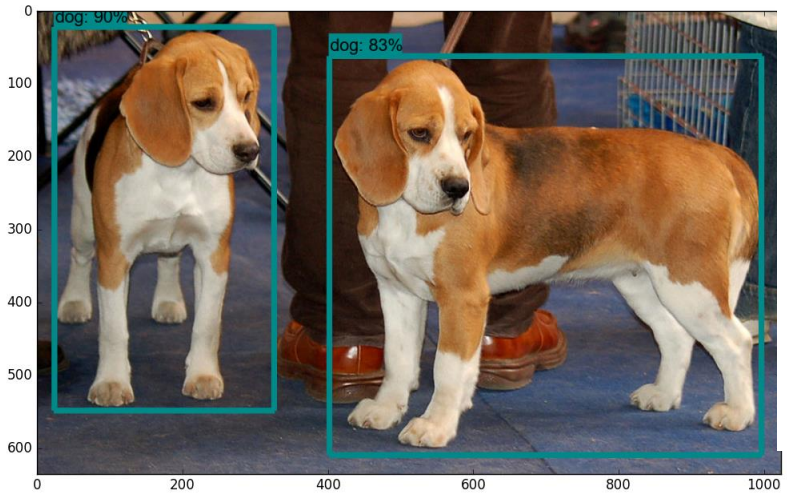
Image Classification

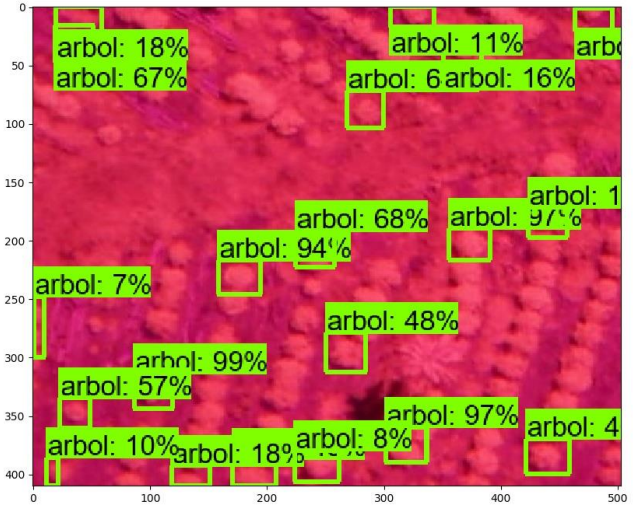
- tutorial de Tensorflow que incluye KERAS, que reconoce flores, usando un modelo entrenado en base a una base de datos de imágenes de flores: https://www.tensorflow.org/tutorials/images/hub_with_keras
- Se modifica ese código para que reconozca otras cosas
- Para esto se puede usar herramienta de ESRI para generar chips de entrenamiento, en este caso con árboles, y otras no de ESRI como labelIMG.
- A las imágenes de muestra le puede aplicar la técnica de “Augmentation” que modifica cada imagen para generar nuevas y así tener mayor cantidad de elementos de entrenamiento

- Se entrena el modelo con las nuevas imágenes
- Se guarda el modelo entrenado.
- Se recupera el modelo entrenado guardado para uso inmediato
- Se utiliza el modelo entrenado recuperado para aplicarlo a imágenes dadas, no usadas en el entrenamiento
- Todo corre desde una consola CMD de Windows, en Python, y puede usarse desde Jupyter Notebook también

Segundo nivel: Object Detection

- Es el segundo caso en el esquema anterior
- Es el detector objetos dentro de una imagen. Pueden ser varios, y los recuadra, no los extrae a partir de su borde



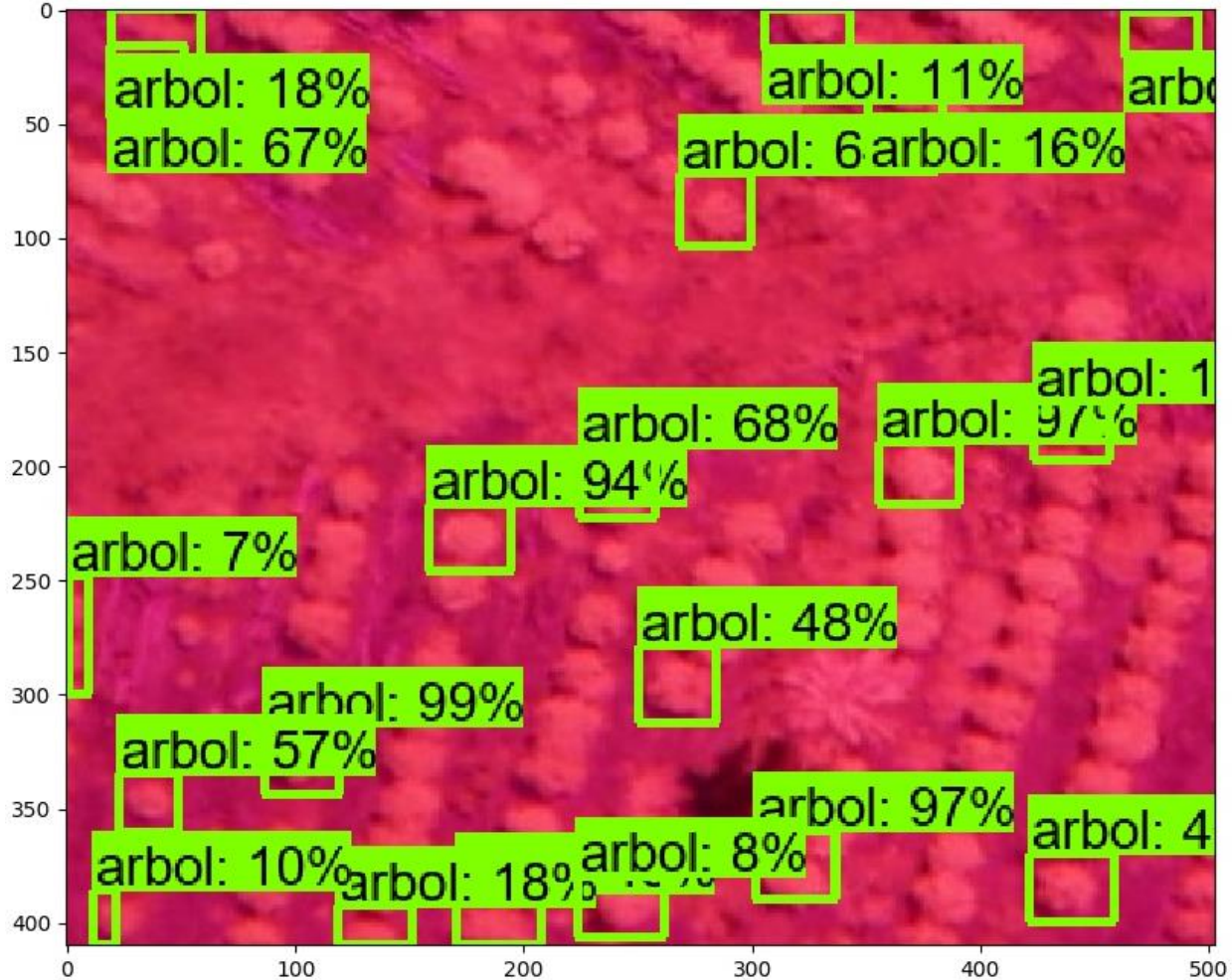


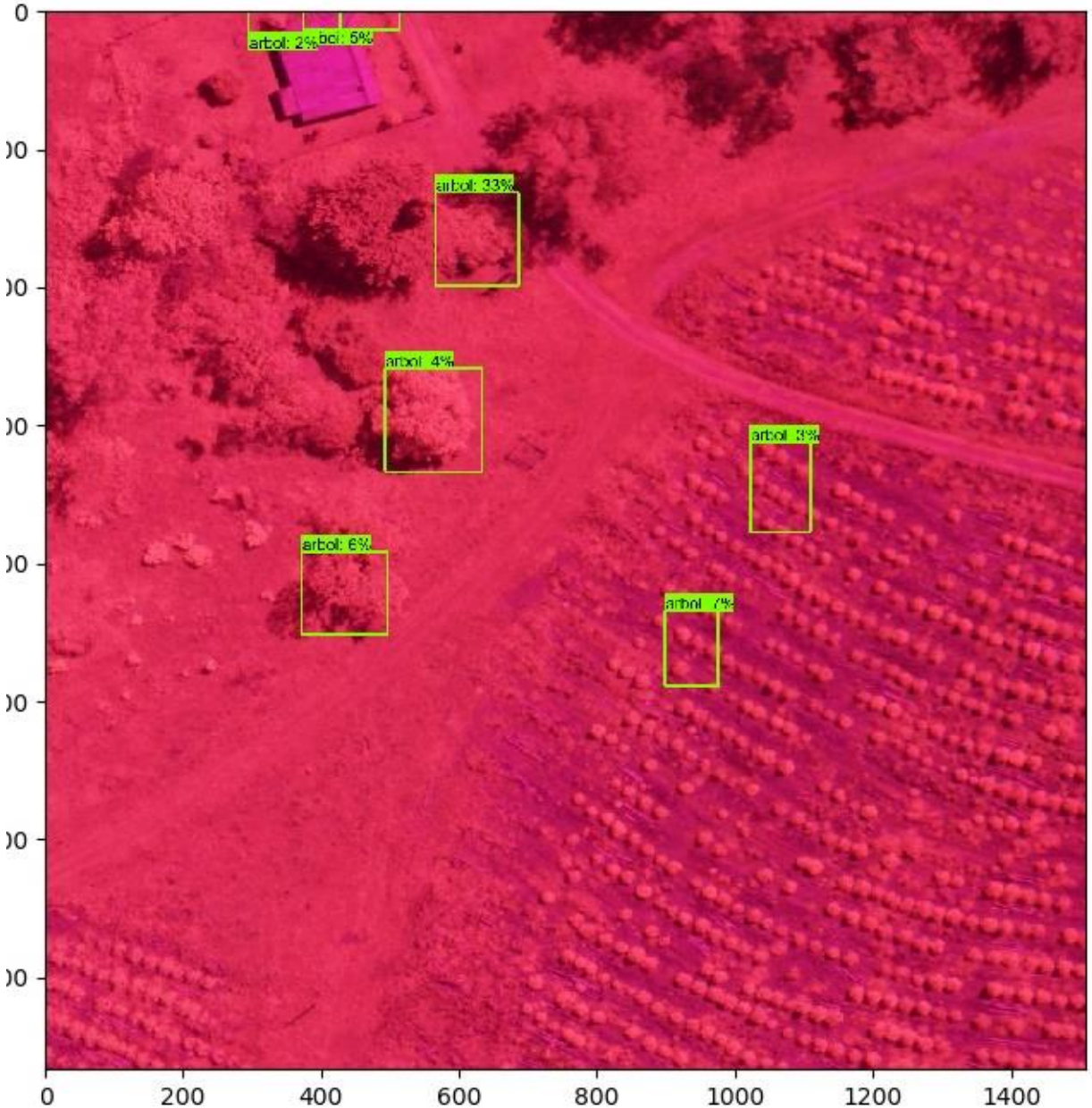
- tutorial bajo el sitio Python Programming: **Introduction and Use - Tensorflow Object Detection API Tutorial (https://pythonprogramming.net/introduction-use-tensorflow-object-detection-api-tutorial/)**
- Basado en Tensorflow

- Tutorial del sitio de Pythonprogramming.net (https://pythonprogramming.net/custom-objects-tracking-tensorflow-object-detection-api-tutorial/)
- **Tracking Custom Objects Intro – Python, Tensorflow Object Detection API Tutorial**
- Por ejemplo se puede modificar el código para reconocer ciervos en imágenes dadas.
- Para eso se deben bajar imágenes desde la web y se seguir los pasos del tutorial para reentrenar.
- Es todo muy artesanal, corre desde CMD o Jupyter Notebook
- Tiempo considerable de entrenamiento

- A partir del código anterior aplicado a detectar ciervos
- Se puede levemente el código y reentrenar para aplicarlo otras cosas
- Se generan imágenes de entrenamiento sobre imágenes de lo que quieran reconocer

- software LabelIMG para etiquetar elementos a detectar dentro de imágenes para poder entrenar la red
- El tiempo de entrenamiento en CPU es hasta 24 veces más lento que usar la GPU de 4GB.
- Se debe de parar cuando al indicador LOSS llegue a menos de 1
- LOSS evoluciona muy lento a medida que alcanza a 1
- Aumentar las imágenes de entrenamiento parece algo necesario, aunque relativo.
- Mejor es asegurar la resolución para no perder detalles
- Se usó un modelo más performante que Faster-RCNN que permite aplicarlo a videos.


arbol: 18%
arbol: 67%
arbol: 11%
arbol: 6
arbol: 16%
arbol: 68%
arbol: 94%
arbol: 7%
arbol: 48%
arbol: 99%
arbol: 57%
arbol: 97%
arbol: 10%
arbol: 8%
0
50
100
150
200
250
300
350
400
0
100
200
300
400
500


arbol: 33%
arbol: 4%
arbol: 6%
0
200
400
600
800
1000
1200
1400

- No se mejora más luego de varios pasos de entrenamiento
- Se requieren más imágenes, o imágenes con mayor detalle, o cambiar algunos parámetros de la red (batch size, etc.)
- Imágenes de 256x256 se pierden detalles, cuidado, este es parte del problema de que no funcione.
- Imágenes más grandes requieren de mayor cantidad de neuronas y eso hace el entrenamiento más lento.

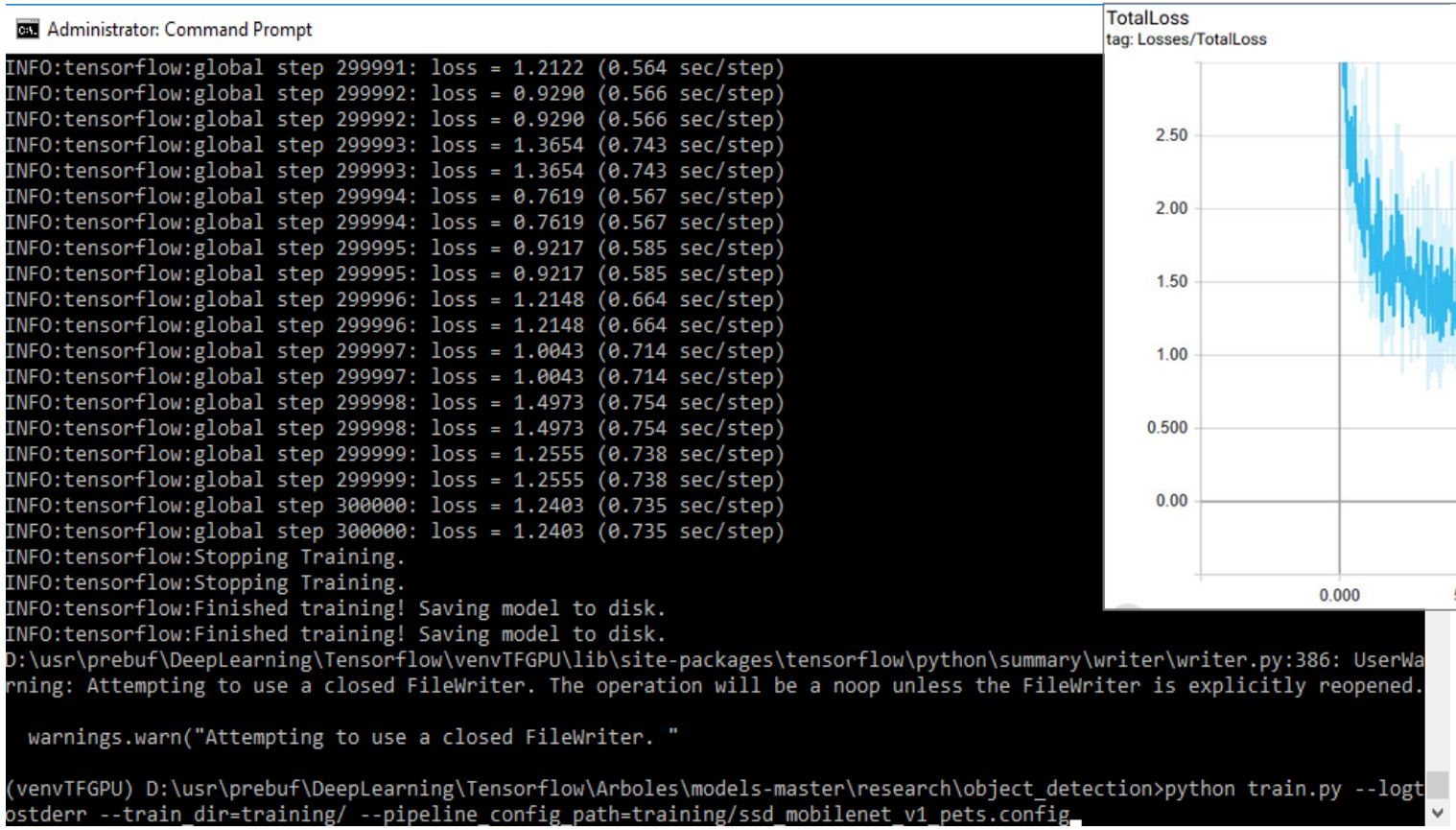
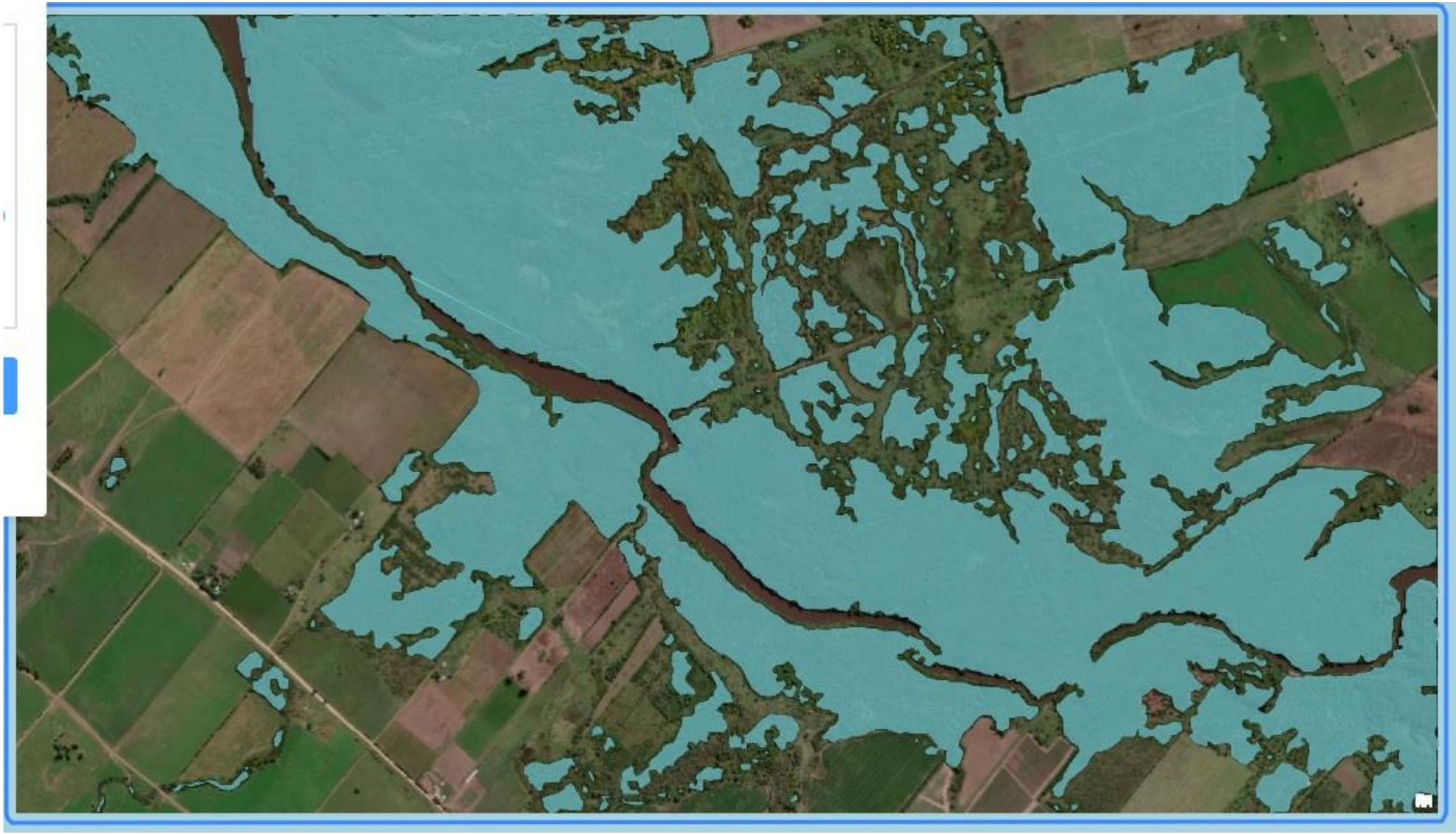
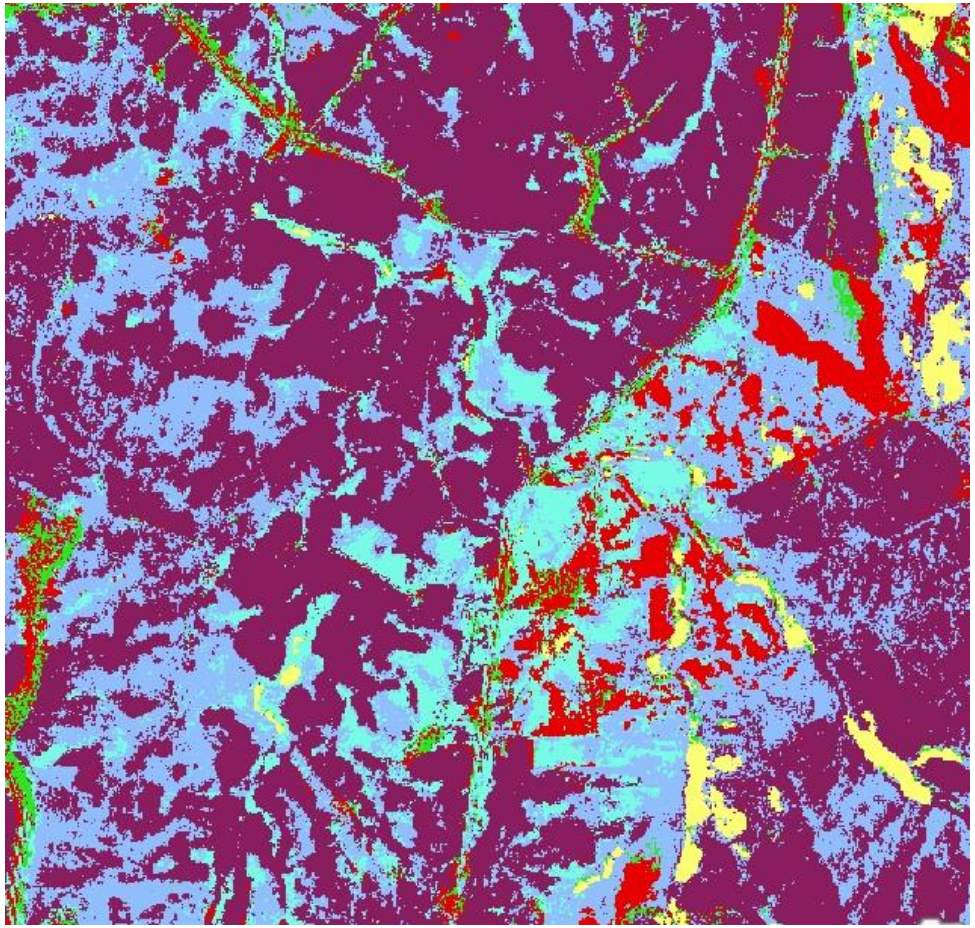


Image Segmentation

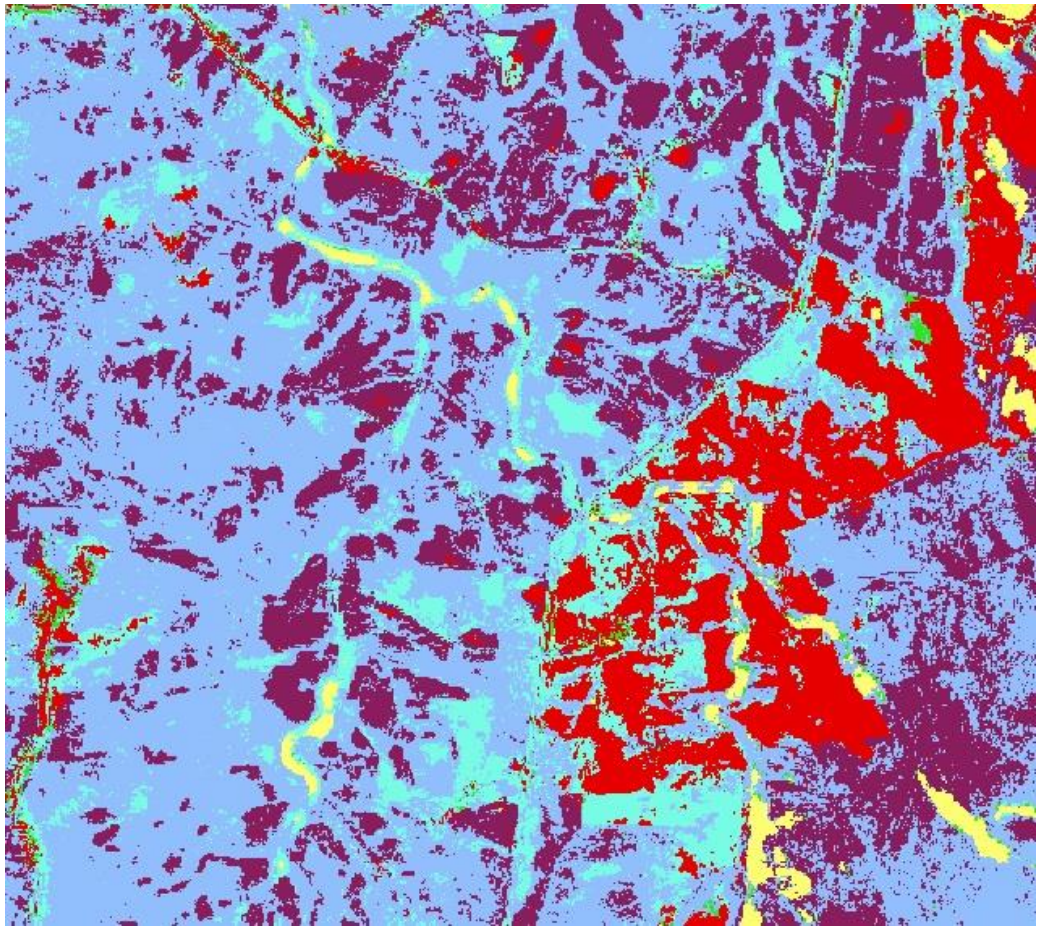
Detección de bosques



Clasificación Forestación Inventario forestal

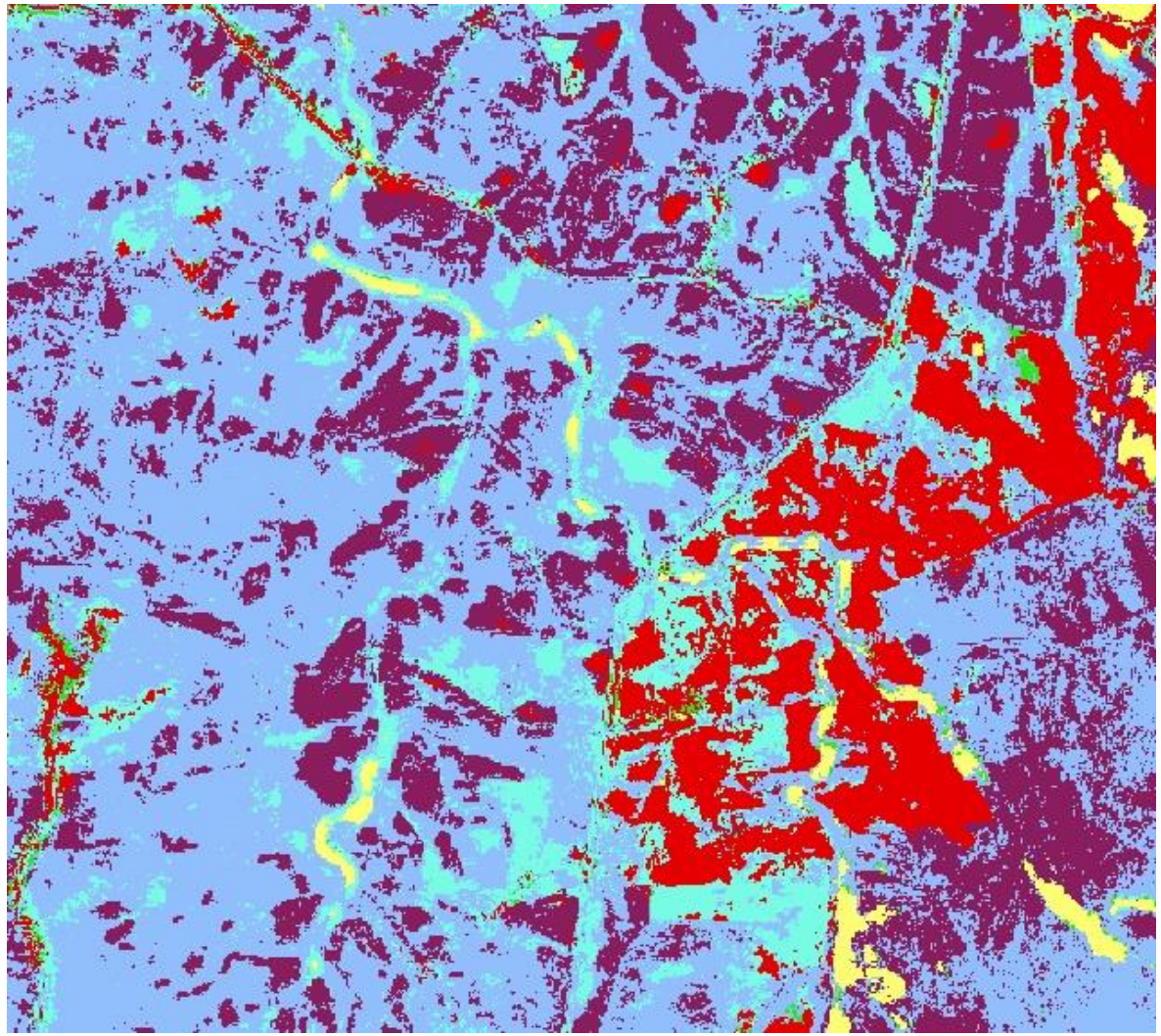


10 Epoch



70 Epoch

RGB Sentinel 2

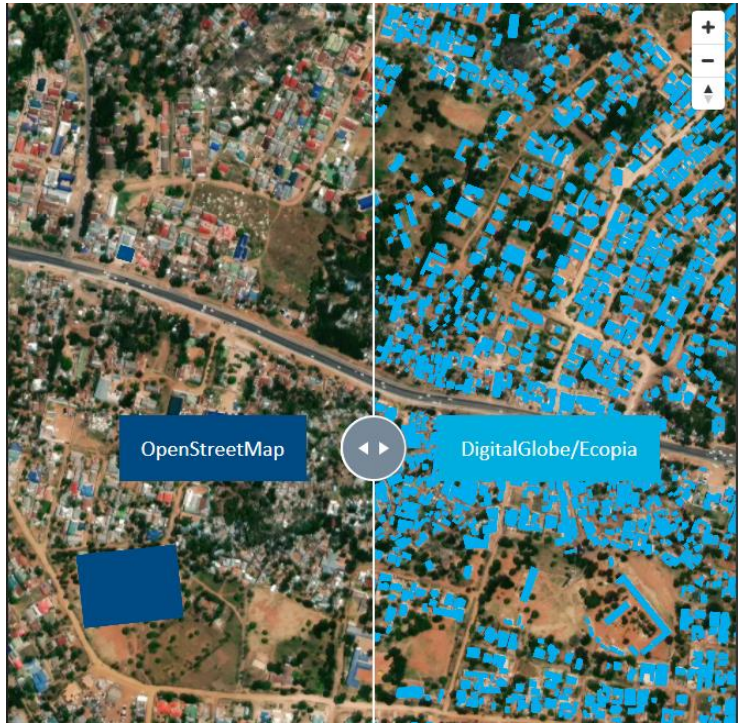


Classifier 70 Epoch

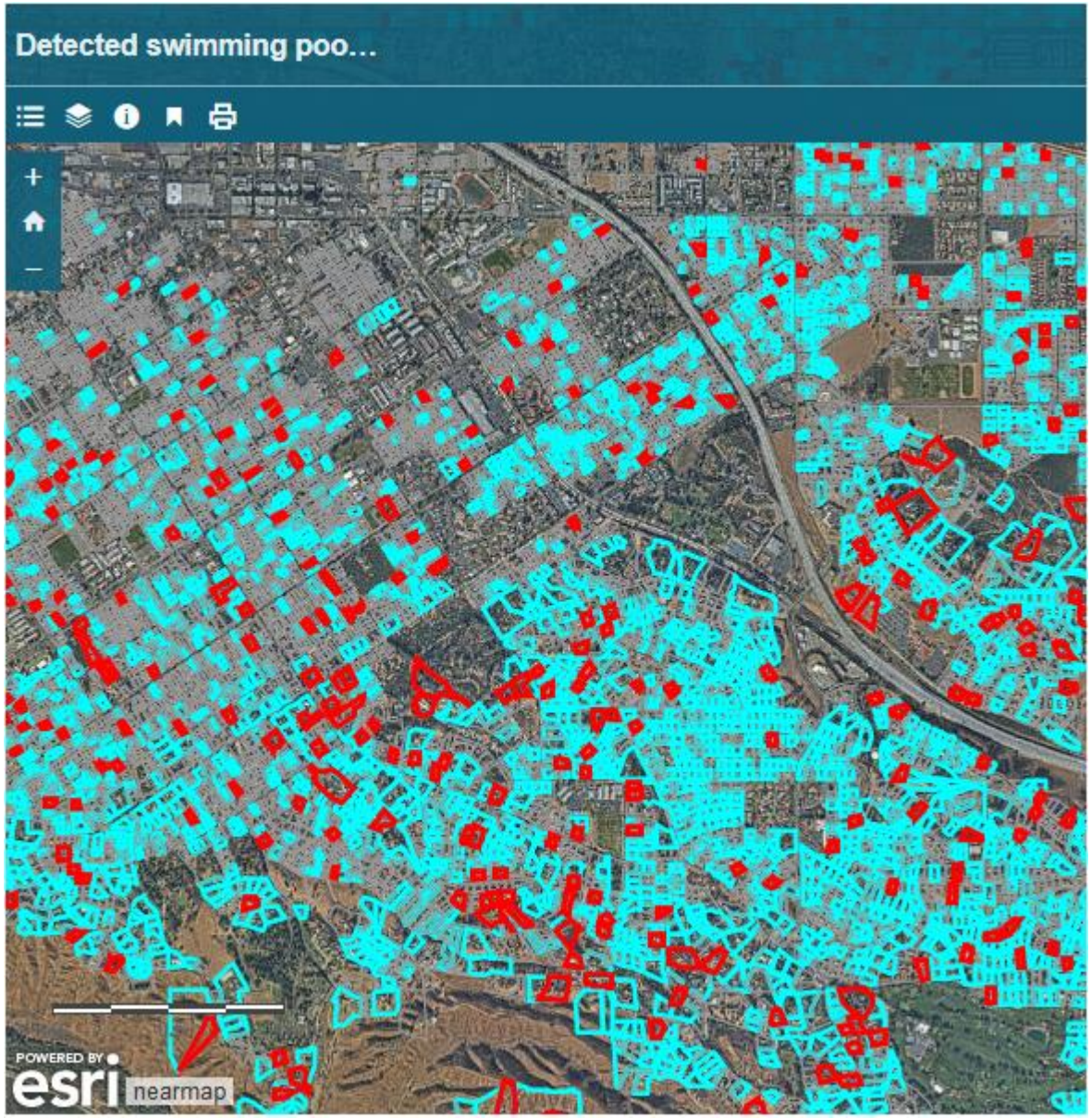

Area per class (km²)
Area (km²)
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
0.00
0.00
0.04
0.01
0.12
0.00
0.00
Solar panels
Man hole
Trees
Vehicles
Roofs
Swimming pool
Road marks
Count per class
Count
3500
3000
2500
2000
1500
1000
500
0
2502
554
2530
833
630
31
19
Solar panels
Man hole
Trees
Vehicles
Roofs
Swimming pool
Road marks

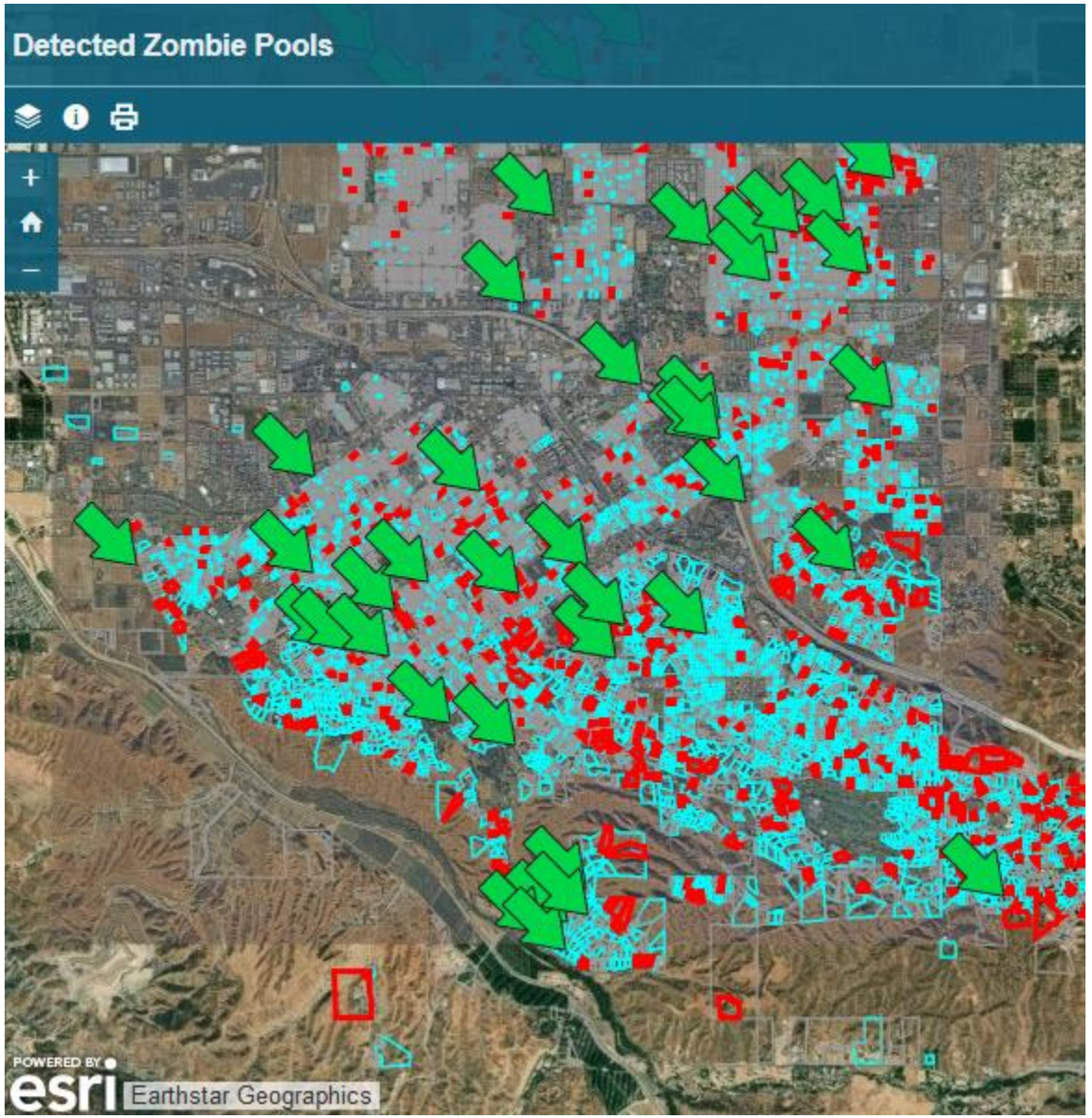
DigitalGlobe

- DigitalGlobe provee un servicio de Image Segmentation sobre imágenes de satélite
- Utilizaron 8000 imágenes para entrenar
- Los resultados son excelentes
- https://explore.digitalglobe.com/Tanzania-Building-Footprints.html



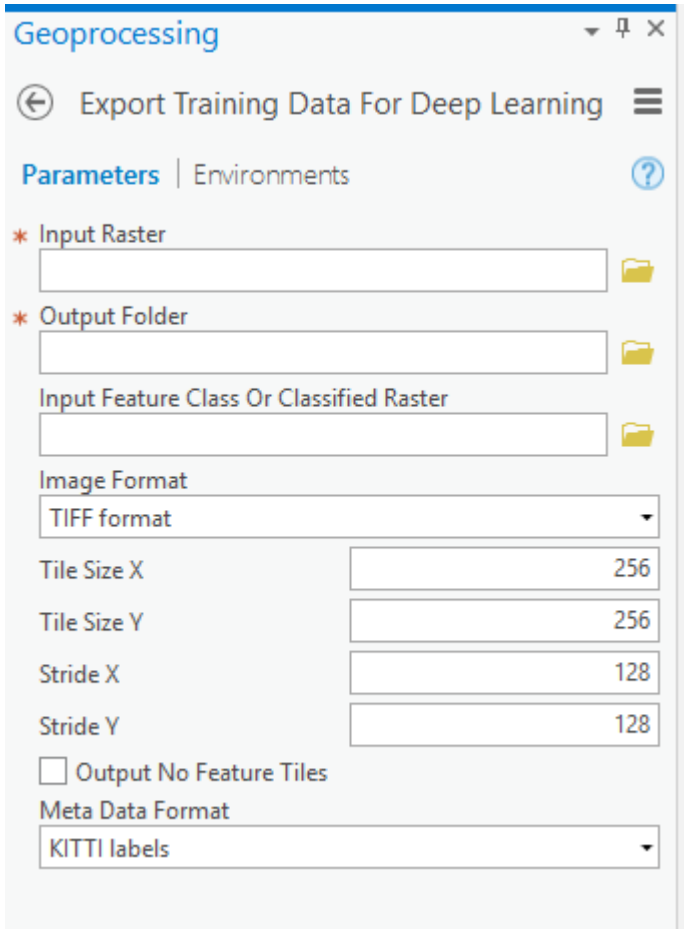
ArcGIS y Deep Learning



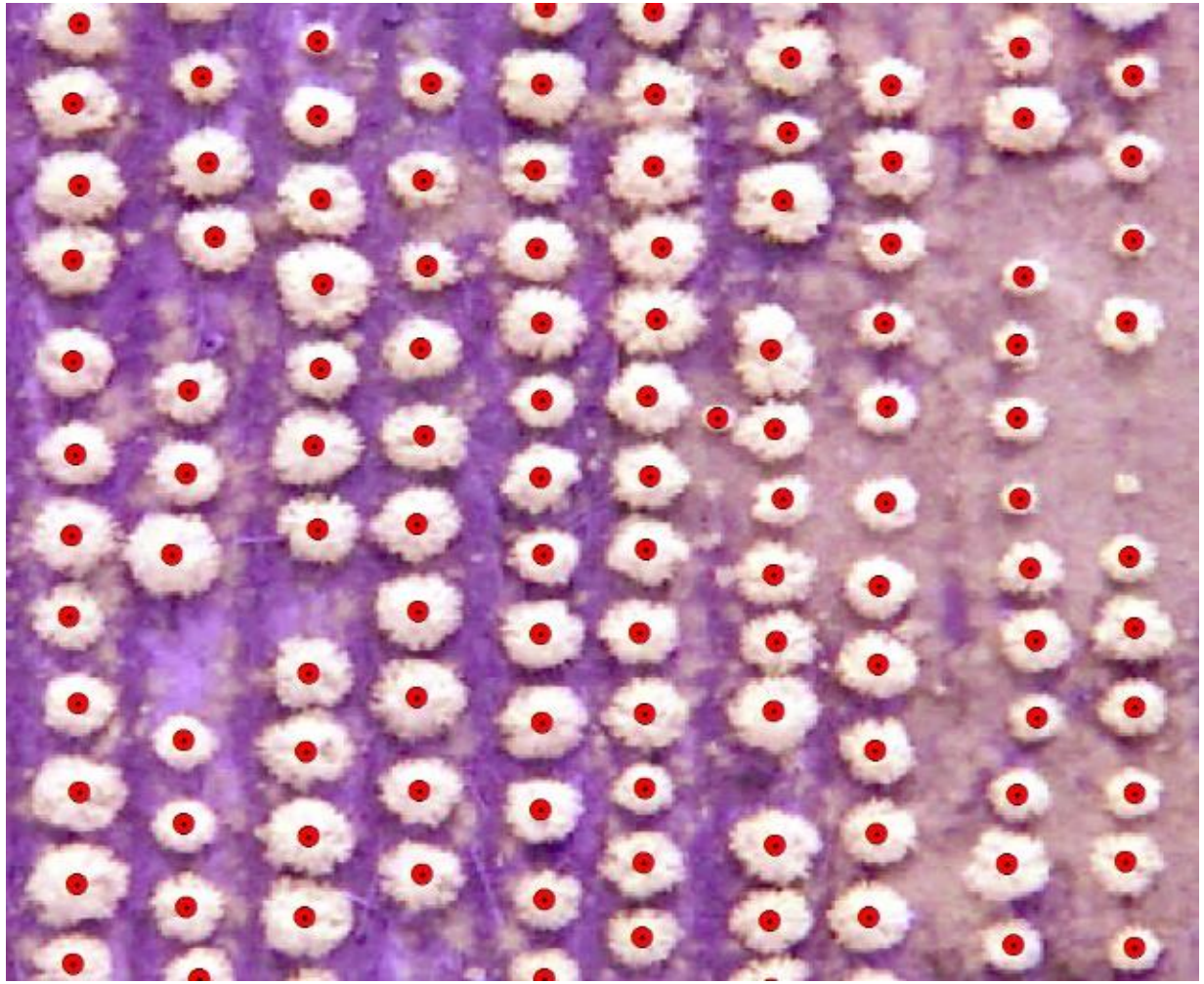


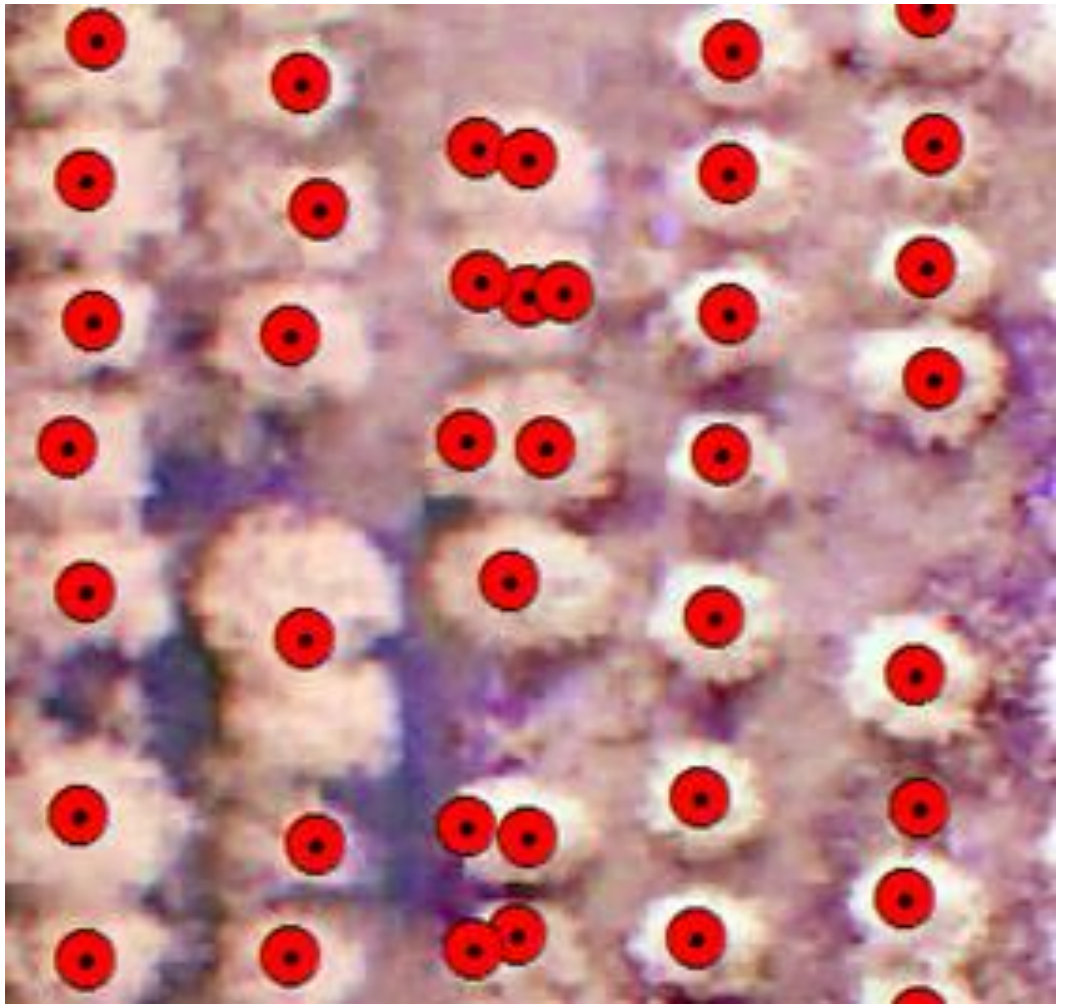
ArcGIS y Deep Learning – Detectando Piscinas

- Detectar cuales de las piscinas pueden estar asociadas a Denge
- Artículo
- Código en GitHub
- 2000 piscinas etiquetadas para entrenar a la red a partir de imágenes infrarrojo cercano, R, G, B (4 bandas).
- Pro para generar las etiquetas (CHIPS, de 224 x 224 pixels) (Export Training data for Deep Learning)
- PyTorch y fast.ai usados para generar una red entrenada.



Conteo de plantas





Superresolution (UNET)

Input / Prediction / Target

Object Tracking (YOLOv3)

Técnicas de IA serán incluidas cada vez más en Software SIG

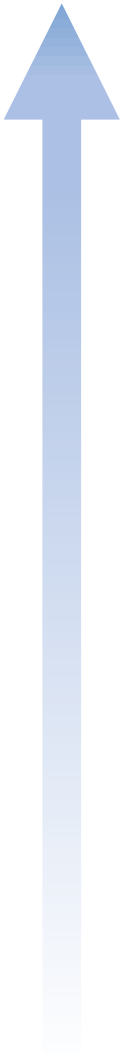
Dicho por Sarah Ambrose, Product Engineer de GeoAnalytics el 6 de Marzo de este año (2018), respecto a las versiones 10.6.1 que son expuestas en Pro 2.2

The focus of the new tools can be broadly categorized into a few groups:

1. *Classic GIS tasks (similar to join features, this could be something like appending data or applying overlays, as well as more track analysis to tie in with real-time data)*
2. *More sophisticated analysis – we're currently researching and working on machine learning and statistical techniques that are relevant to big data.*

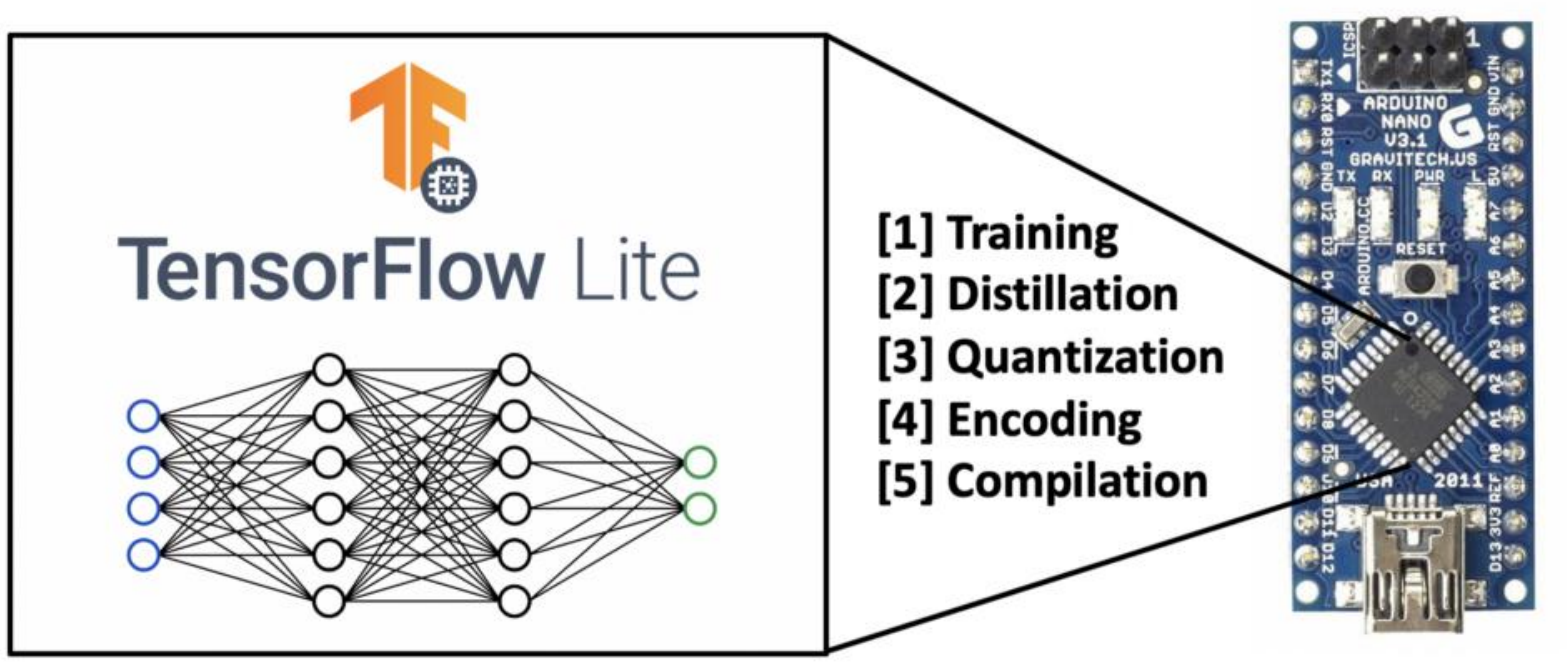
Niveles de IA

- Usuarios finales no saben que están usando IA: traductor Google, Google search, face recognition FB, etc.
- Herramientas tipo wizard, poco flexibles, más automatizadas: ArcGIS Pro SVM, ArcGIS Pro RF, arcgis.learn
- Herramientas que permiten “programación gráfica”: Microsoft Azure Machine Learning Studio.
- Programación tradicional a modo de scripting usando bibliotecas de IA: Python, TensorFlow, Pytorch, CNTK, Keras, etc.
- Diseño de algoritmos de IA: Open AI, Google Deep Mind, Microsoft, Facebook, Alibaba, IBM, la academia, etc.



TinyML

https://towardsdatascience.com/tiny-machine-learning-the-next-ai-revolution-495c26463868

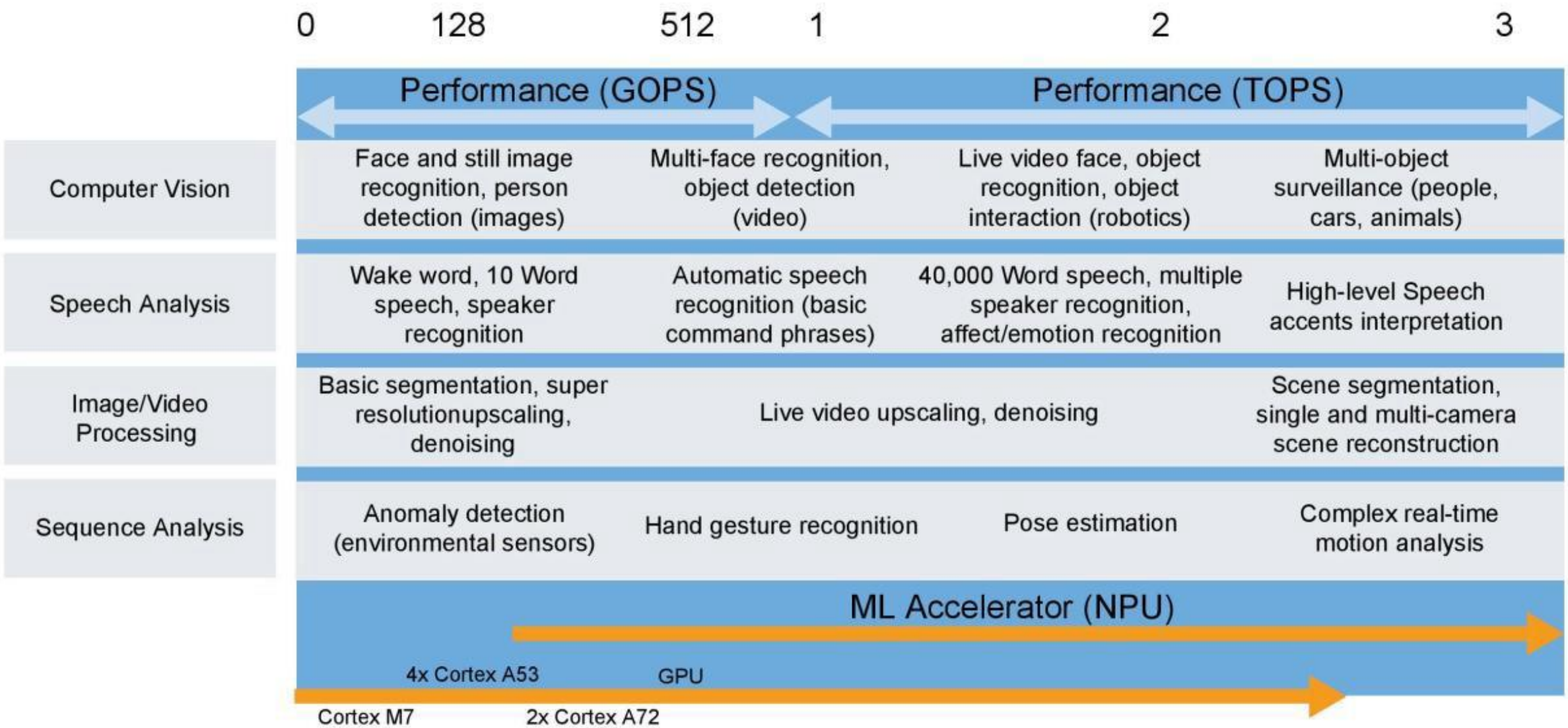


TinyML

Aplicaciones:

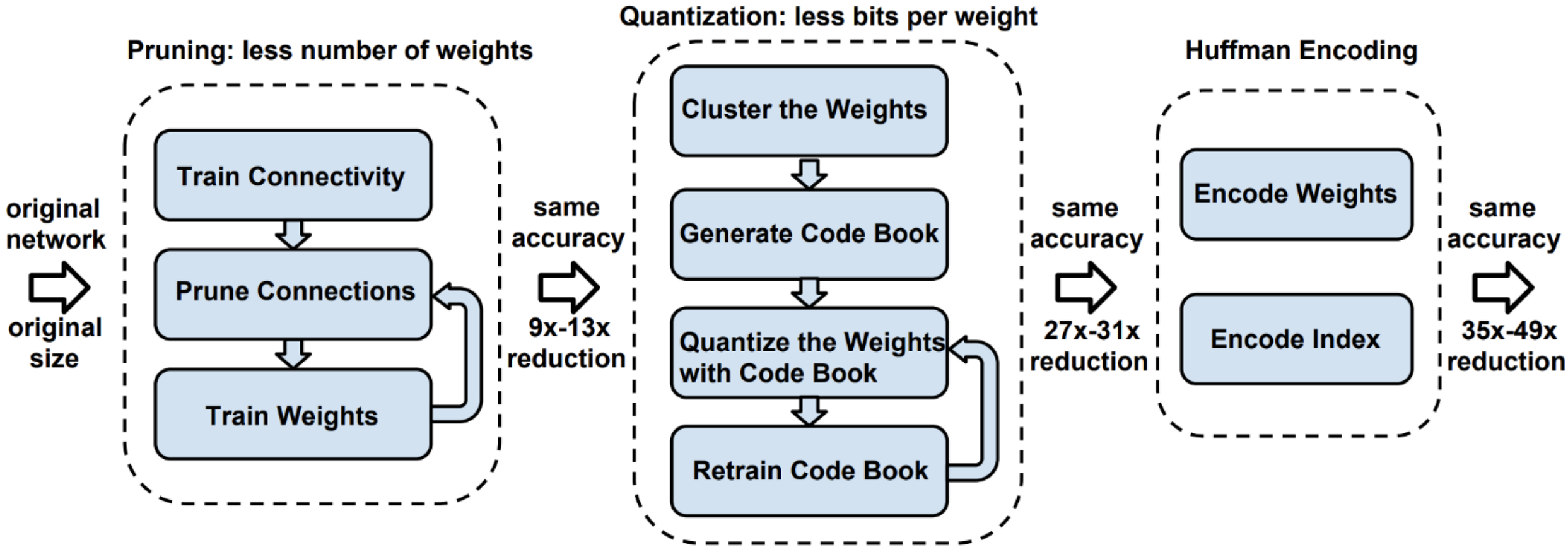
- IoT inteligentes de verdad
- Keyword Spotting: Hey Siri, Hey Google (CPU and energy demanding at the moment). Speech recognition
- Visual Wake Words:
  - Security camara activate at presence of a person
  - Light turns on at presence of a persona
  - Detecting bird sounds in forest

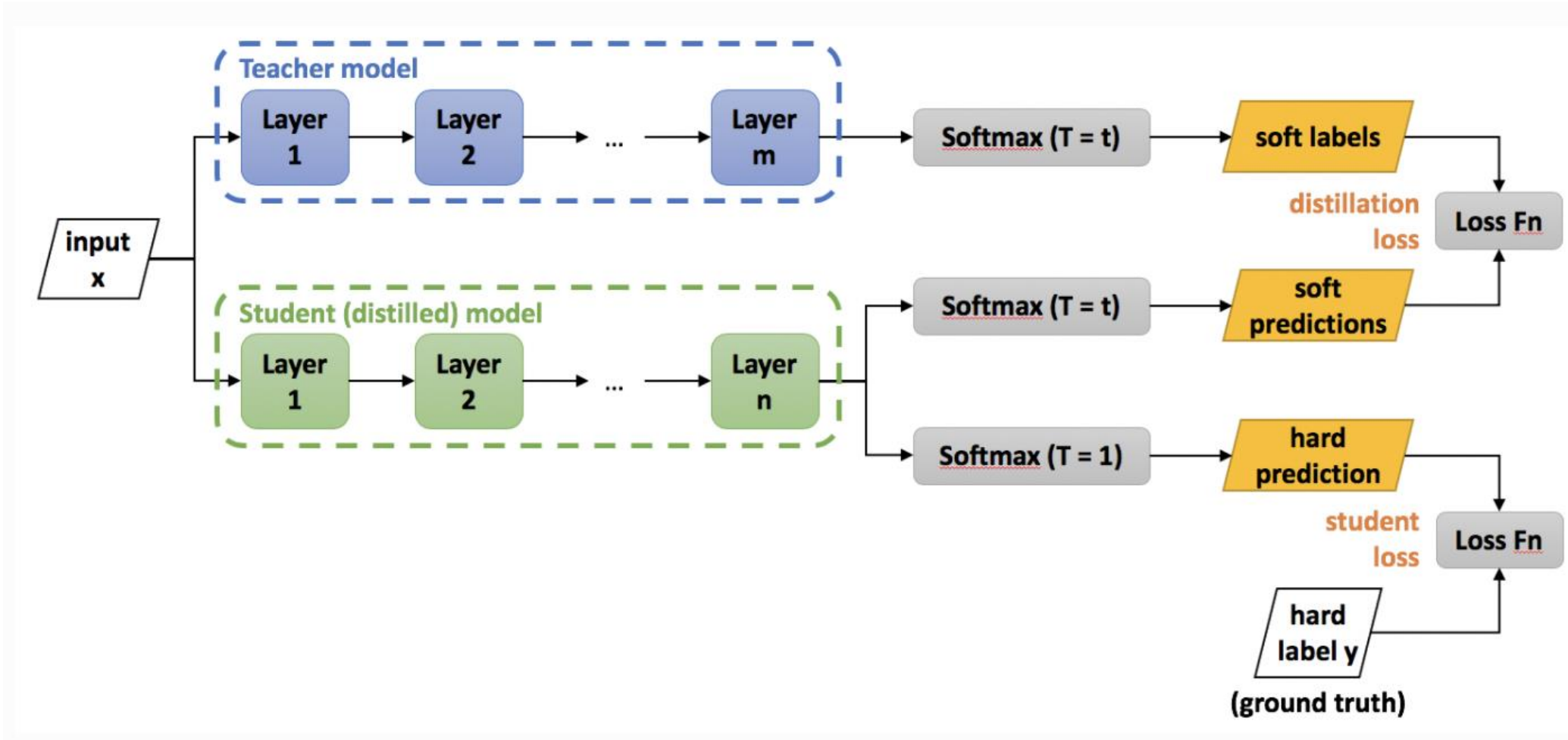
Machine Learning Use Cases



How TinyML Works

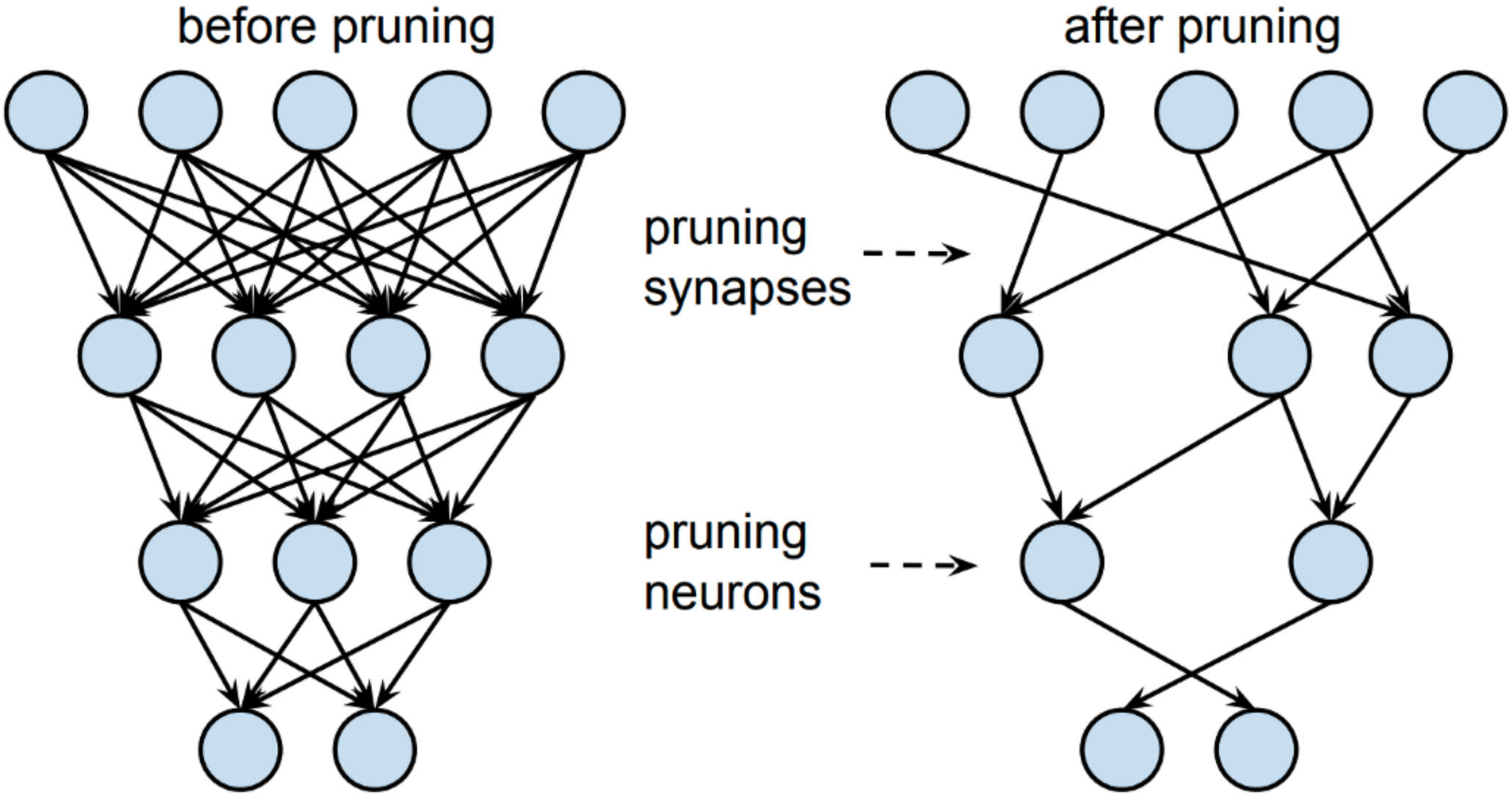
TinyML algorithms work in much the same way as traditional machine learning models. Typically, the models are trained as usual on a user's computer or in the cloud. Post-training is where the real tinyML work begins, in a process often referred to as **deep compression**.


Pruning: less number of weights
Quantization: less bits per weight
Huffman Encoding
original network
original size
Train Connectivity
Prune Connections
Train Weights
same accuracy
9x-13x reduction
Cluster the Weights
Generate Code Book
Quantize the Weights with Code Book
Retrain Code Book
same accuracy
27x-31x reduction
Encode Weights
Encode Index
same accuracy
35x-49x reduction



Knowledge distillation - In this diagram, the 'teacher' is a trained neural network model. The teacher is tasked with transferring its 'knowledge' to a smaller network model with fewer parameters, the 'student'. This process is used to enshrine the same knowledge in a smaller network, providing a way of compressing the knowledge representation, and hence the size, of a neural network such that they can be used on more memory-constrained devices.

Prunning


before pruning
after pruning
pruning
synapses
pruning
neurons

Not train on the device

- neural networks have been trained using 16-bit and 8-bit floating-point numbers.
- Differentiation and Backpropagation requires high numerical precision -- more bits
- 8 bit backpropagation results in poor result