Recuperación de Información Multimedia

Deep Learning (imágenes)

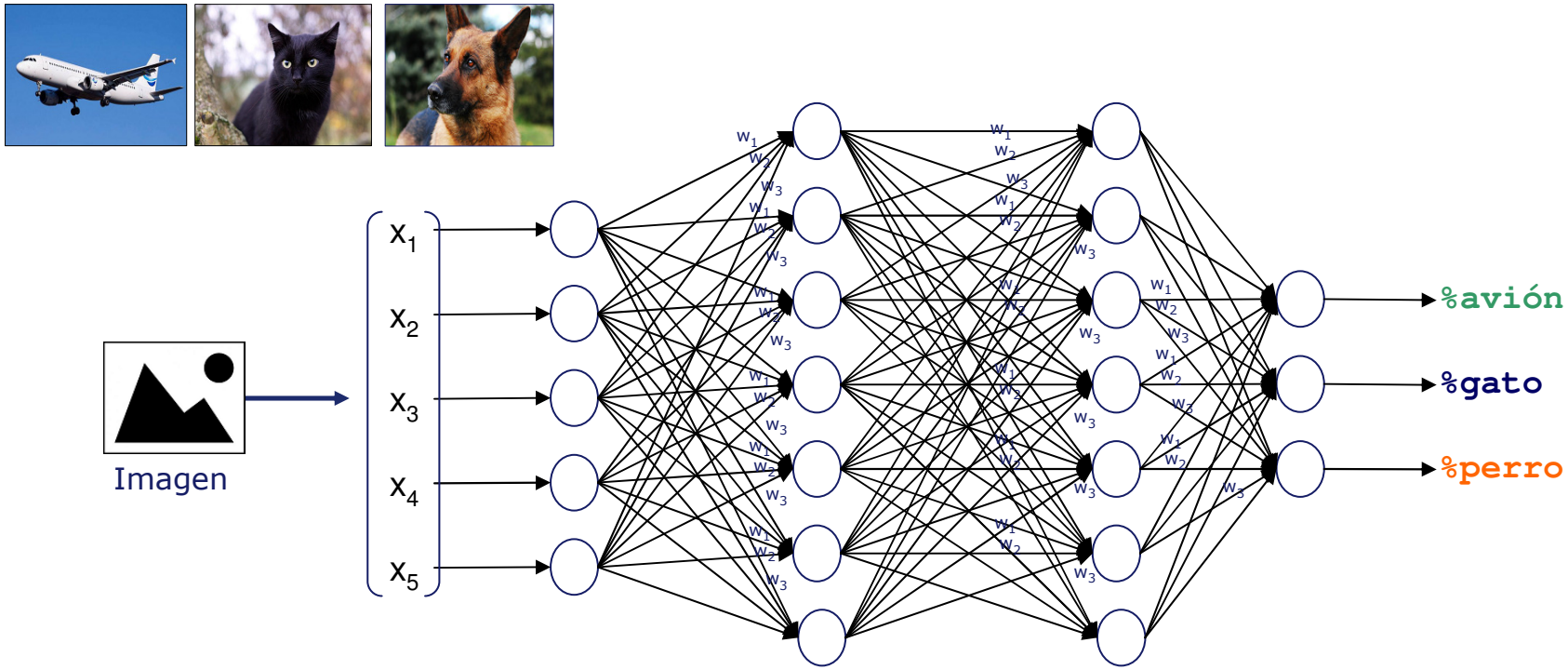
CC5213 – Recuperación de Información Multimedia
Departamento de Ciencias de la Computación
Universidad de Chile
Juan Manuel Barrios – https://juan.cl/mir/ – 2020

Deep Learning

- Se refiere al uso de redes neuronales muy grandes o profundas
- Se diferencia de redes neuronales tradicionales (MLP) en:
  - La entrada de la red es el **dato multimedia** mismo
    - Imágenes, Video, Audio, Texto
  - La red **entrena el cálculo del vector característico**
- Se definen diferentes arquitecturas (tipos de red) combinables entre sí (MLP, CNN, RNN, etc.)
- Las redes requieren hiper-parámetros (neuronas, capas, etc.) y entrenar muchos parámetros (pesos)

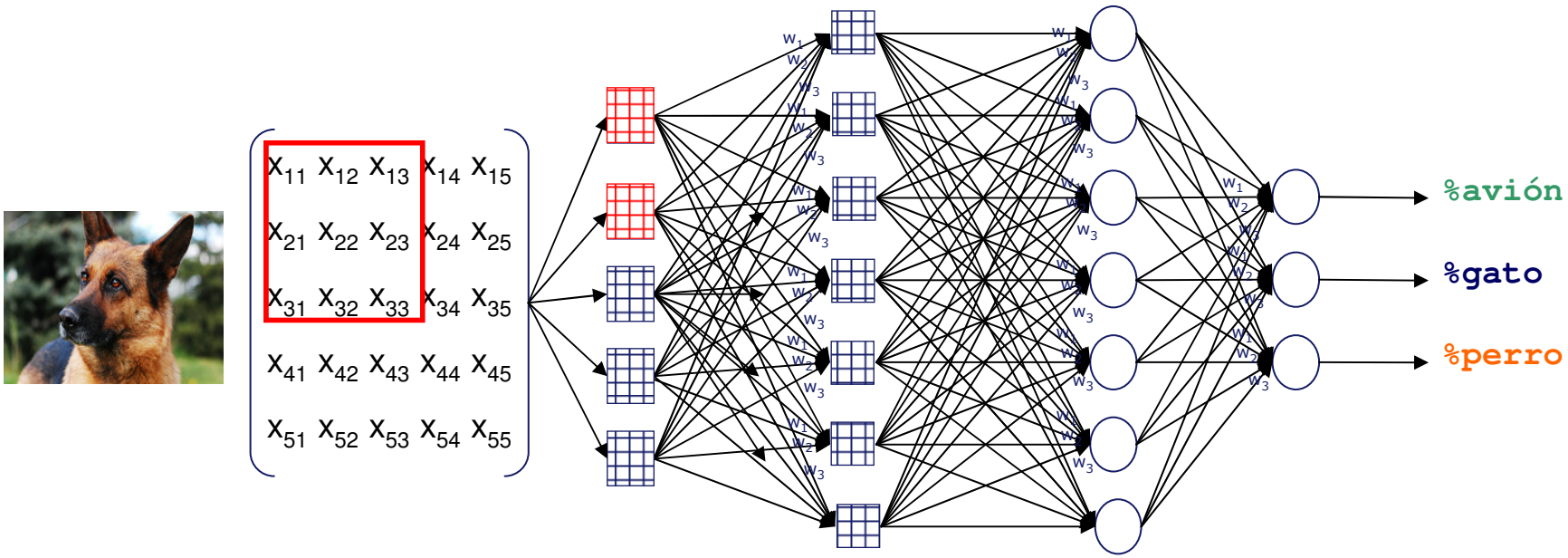
Enfoque Tradicional

- Cada imagen se representa por un vector
  - Ej: Histogramas de colores, Orientaciones de bordes, Ocurrencias de Texturas, etc.



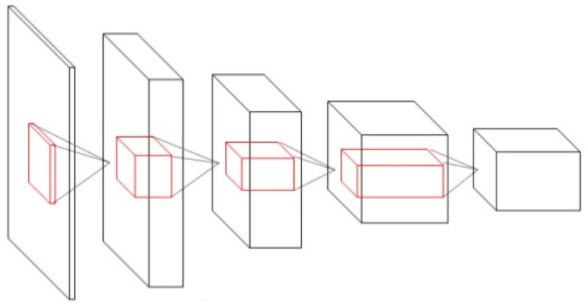
Red Neuronal Convolucional

- Operadores de **convolución** sobre una imagen
  - Ventana deslizante que **busca patrones** en la imagen
  - El resultado de una convolución es una imagen de (casi) el mismo tamaño
  - Los valores del filtro de convolución son los parámetros a entrenar
- El operador de pooling reduce el tamaño de una imagen
- El cálculo del vector característico es parte del entrenamiento



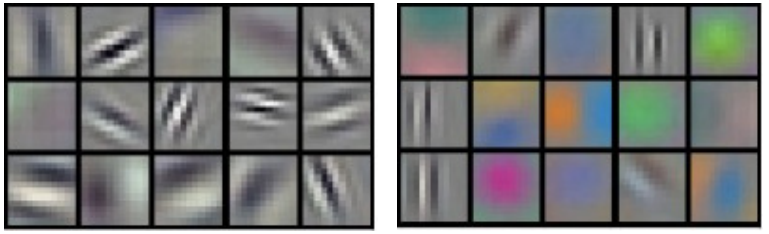
Operador de Convolución

- Imagen de entrada de W x H x D (alto x ancho x canales)
  - Normalizar la entrada: restar (127,127,127) u otro valor
- Tamaño del Filtro (3, 5, …)
  - Convolución con filtros cuadrado en toda la profundidad de la entrada: tamaño 3 implica filtro de 3x3xD (9D parametros)
- Cantidad de filtros (16, 32, 64, …)
  - Un filtro produce una canal de salida
- Paso de la convolución o *stride* (1, 2, …)
- Tamaño del borde o *zero-padding* (1, 2, …)
- Dilatación de la convolución



Capa de Convolución

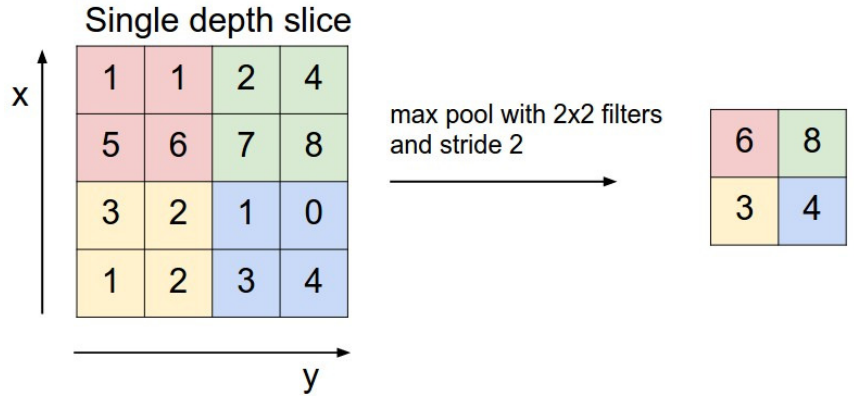
- Salida:
  - W=H=((Input – TamañoFiltro + 2*Padding) / Stride) + 1
  - Profundidad es la cantidad de filtros
  - Ej: Con stride=1 y padding=(TamañoFiltro - 1)/2 se produce una imagen con tamaño igual a la entrada
- Cantidad de parámetros a entrenar:
  - Parámetros por capa: NumFiltros x (F x F x D + 1 bias)
  - Notar que no se entrena un filtro por pixel si no que el filtro opera sobre toda la imagen (Parameter Sharing)
- Filtro de 1x1 se usa para reducir la profundidad



http://cs231n.github.io/convolutional-networks/

Capa de Pooling

- Operación fija en la coordenada espacial
- Entrada: W x H x D
- Tamaño de la ventana (F) usualmente 2
- Tamaño del paso o *stride* (S) usualmente 2
- Tipo de operación: MAX-Pooling
  - Se puede usar AVG-Pooling aunque los resultados muestran mejores resultados con Max (seleccionar el mayor)
- Output:
  - Ancho = (W-F)/S + 1
  - Alto = (H-F)/S + 1
  - Profundidad = D
- No hay parámetros a entrenar



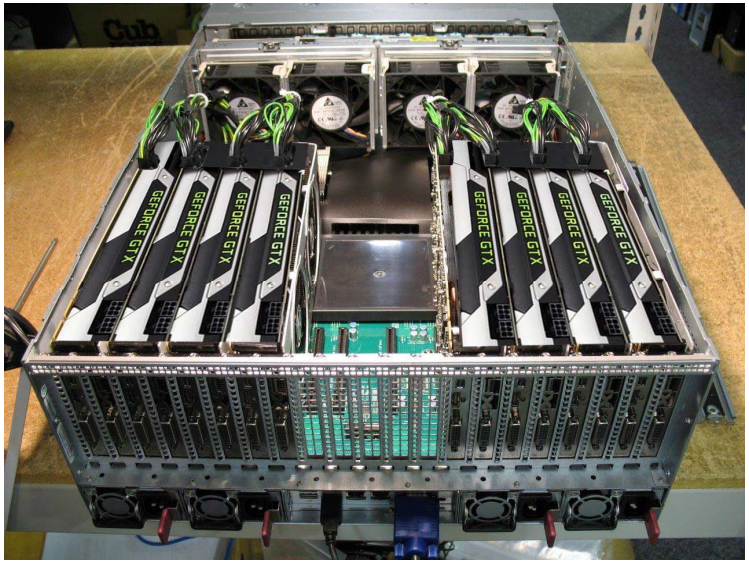
Capa Fully Connected

- Red MLP Tradicional
- Red densa, conecta todas las entradas con todas las salidas
- Simula un clasificador tradicional que opera sobre el descriptor de contenido
- Cantidad de parámetros a entrenar:
  - Entrada x (Salida + 1 bias)

Imágenes para Entrenamiento

- **ImageNet** http://www.image-net.org/
  - 1.5 millones de imágenes de entrenamiento (ahora ~15 millones)
  - Imágenes etiquetadas en 1000 categorías (~20 mil synsets)
- **Competencia ILSVRC**: ImageNet Large Scale Visual Recognition Competition (top-5 error rate):
  - (2010) 28% (enfoque tradicional)
  - (2011) 26% (enfoque tradicional)
  - **(2012) 16% (AlexNet, 8 capas)**
  - (2013) 12% (ZFNet, 8 capas)
  - (2014) 7% (GoogLeNet, 22 capas) 7.5% (VGG-16 VGG-19)
  - (2015) 4% (ResNet, 152 capas)
  - (2016) 3% (TSNet, emsemble)
  - (2017) 2.2% (SENet, ensemble)

Hardware



NVIDIA

Software


TensorFlow


PyTorch


NVIDIA
CUDA


cuDNN


python


scikit
learn


gensim
topic modelling for humans


K Keras


jupyter


colab


Caffe


Amazon
Machine
Learning


Google ML engine

Datos

IMAGENET

kaggle

Google Dataset Search

Search for Datasets

COCO Common Objects in Context

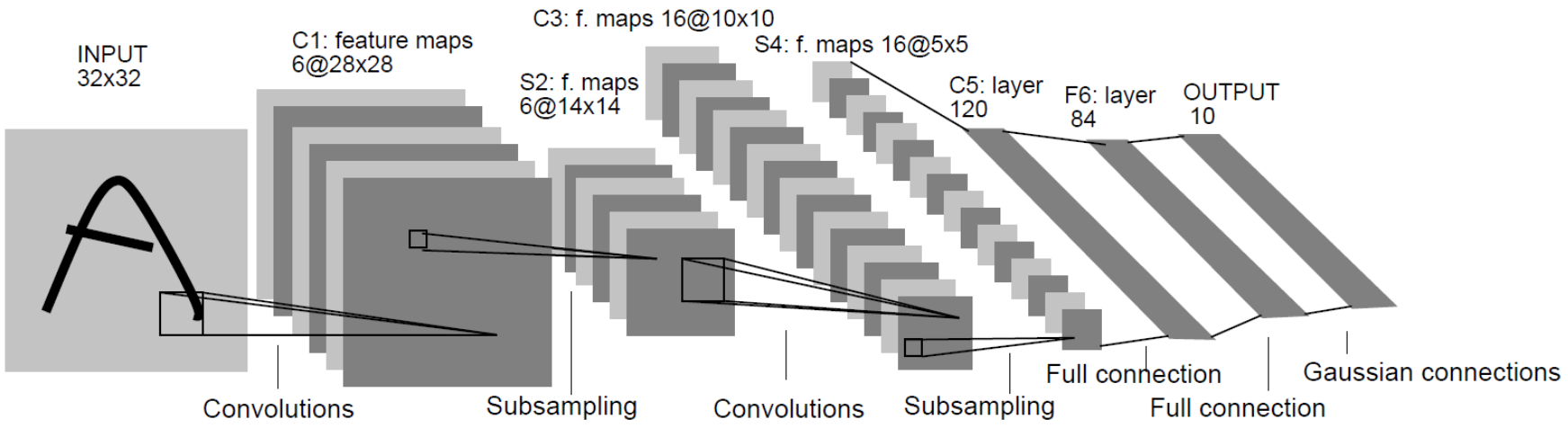
datos.gob.cl

Datos de buena **calidad** que representen adecuadamente el **problema** a resolver

"Garbage In, Garbage Out"

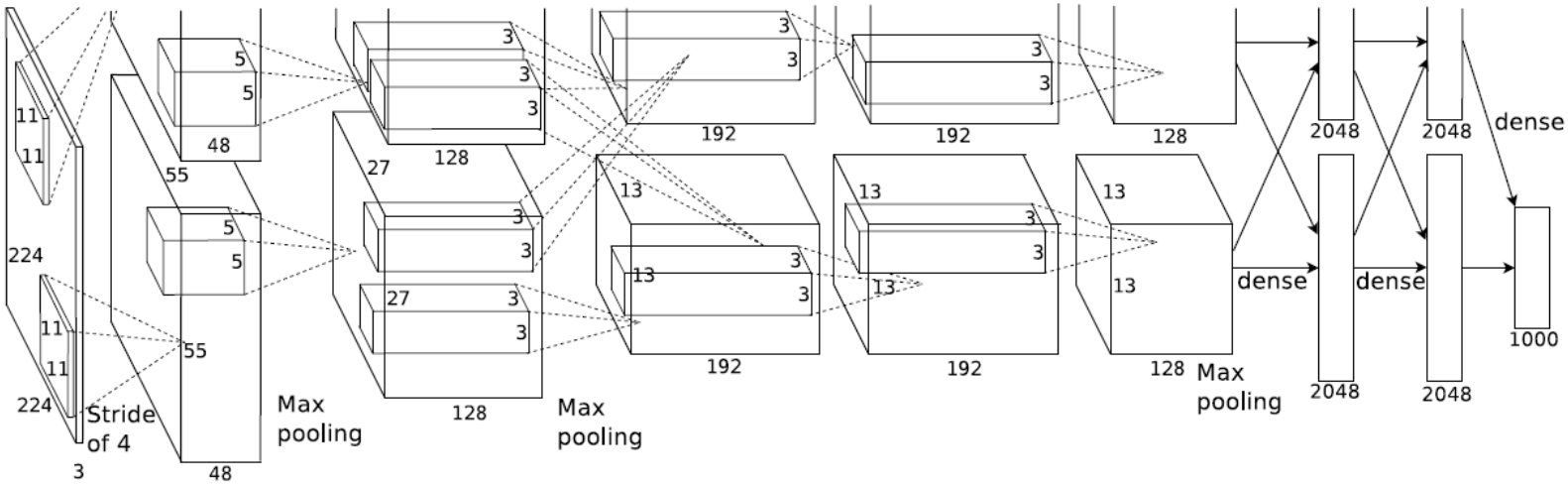
LeNet (1998)

- Una red convolucional (CNN) para dígitos



AlexNet (2012)

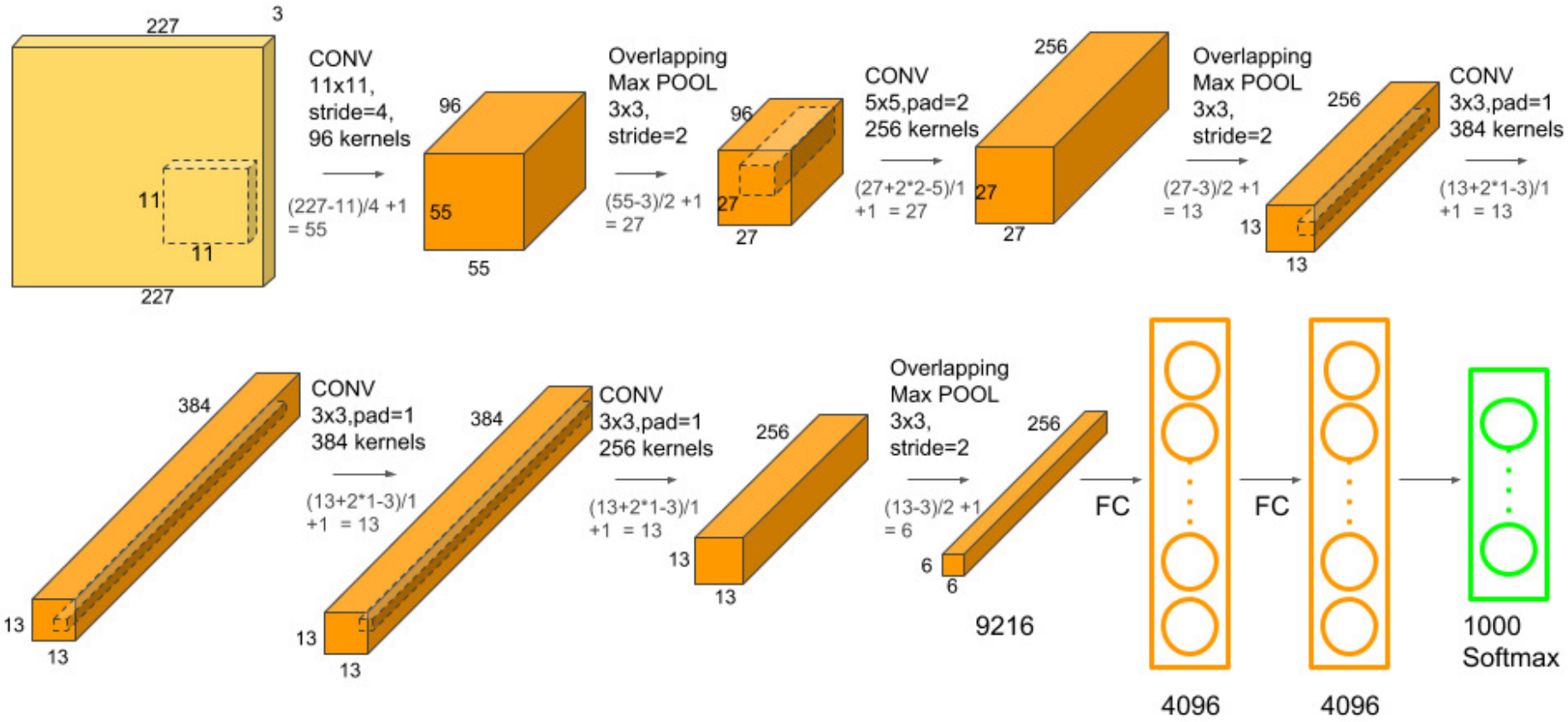
Krizhevsky, Sutskever, Hinton. ImageNet Classification with Deep Convolutional Neural Networks. 2012
http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf



- **Input**: 227 alto x 227 ancho x 3 canales (rgb)
- **Convolución 1**: 96 filtros 11x11x3, stride 4 → 55x55x96 (0,03 M params)
- **Max Pooling 1**: kernel=3, stride=2 → 27x27x96
- **Convolución 2**: 2 grupos 128 filtros 5x5x48, padding 2 → 27x27x256 (0,31 M params, 1%)
- **Max Pooling 2**: kernel=3, stride=2 → 13x13x256
- **Convolución 3**: 2 grupos 192 filtros 3x3x128, padding 1 → 13x13x384 (0,44 M params, 1%)
- **Convolución 4**: 2 grupos 192 filtros 3x3x192, padding 1 → 13x13x384 (0,66 M params, 1%)
- **Convolución 5**: 2 grupos 128 filtros 3x3x192, padding 1 → 13x13x256 (0,44 M params, 1%)
- **Max Pooling 5**: kernel=3, stride=2 → 6x6x256
- **Fully Connected 6**: Input 9216 → Output 4096 (37,8 M params, 62%)
- **Fully Connected 7**: Input 4096 → Output 4096 (16,8 M params, 28%)
- **Fully Connected 8**: Input 4096 → Output 1000 (4,1 M params, 7%)

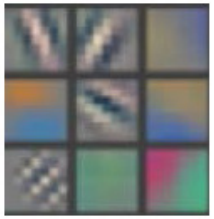
(Total=60,5 M params * 4 bytes = 242 MB)

AlexNet (2012)

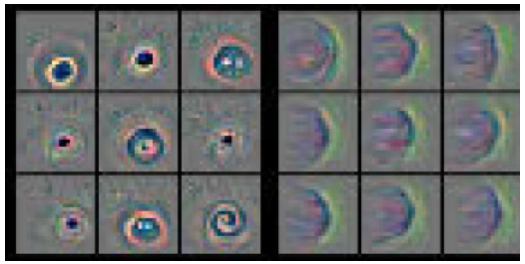


https://www.learnopencv.com/understanding-alexnet/

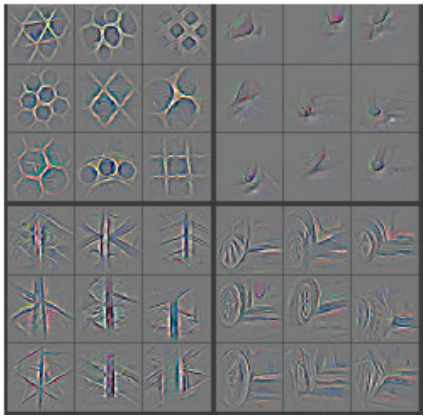
Filtros entrenados



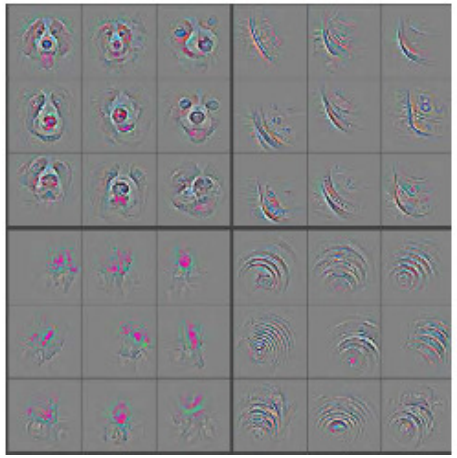
Filtros capa 1



Filtros capa 2



Filtros capa 3



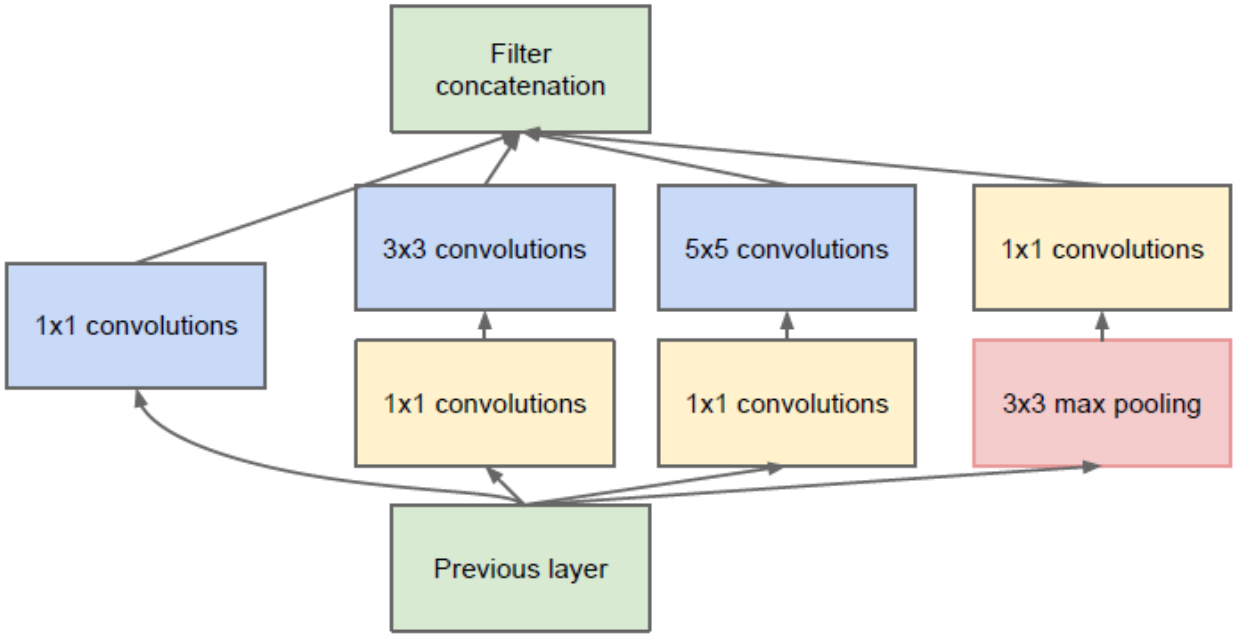
Filtros capa 4

Filtros capa 5

Zeiler, Fergus. Visualizing and Understanding Convolutional Networks. 2013.
https://arxiv.org/pdf/1311.2901

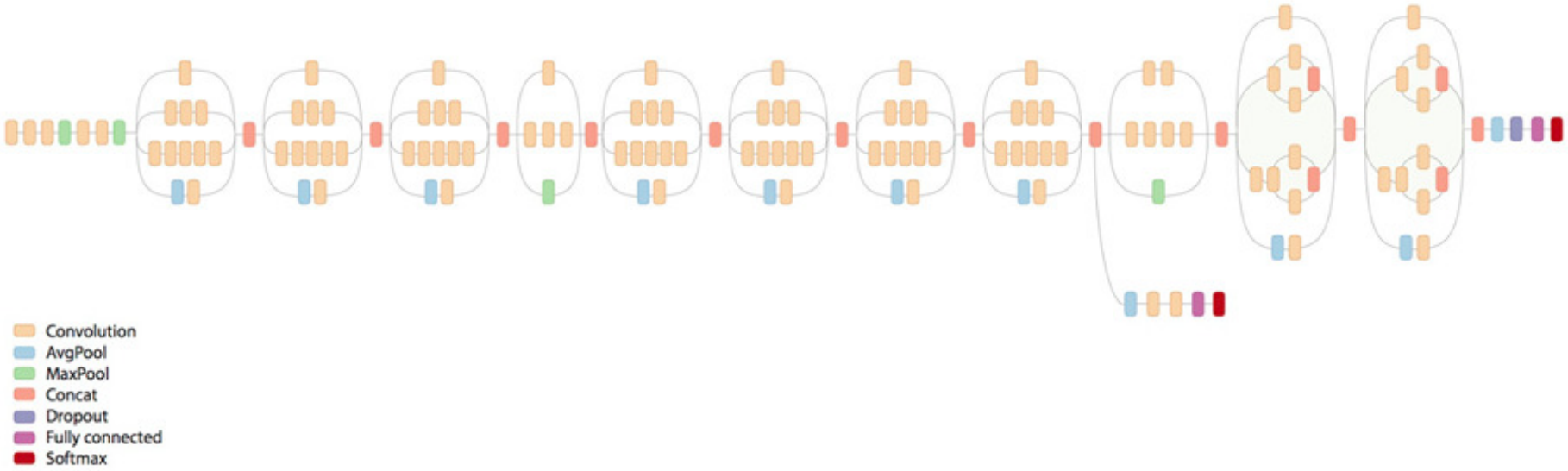
GoogleNet (2014)

- Procesar filtros de distintos tamaños en paralelo y concatenarlos (“Inception”)



GoogleNet (2014)

- No tiene capa Fully Connected



Szegedy et al. Going Deeper with Convolutions. 2015.

https://www.cv-foundation.org/openaccess/content_cvpr_2015/papers/Szegedy_Going_Deeper_With_2015_CVPR_paper.pdf

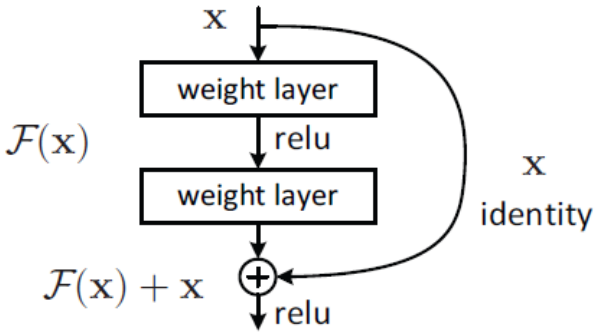
VGG (2014)

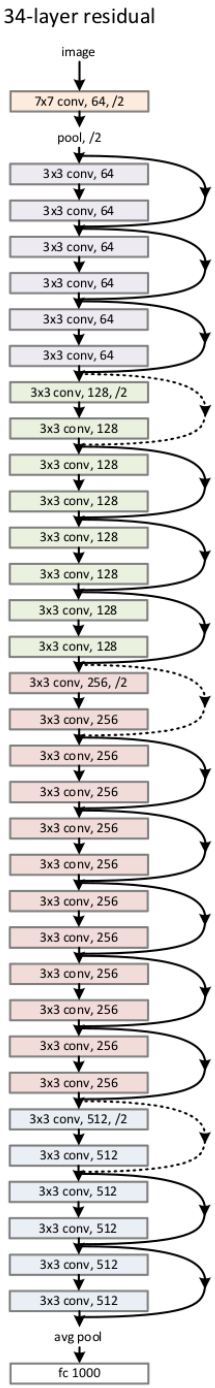
- Distintas versiones
  - VGG-16, VGG-19
    - http://www.robots.ox.ac.uk/~vgg/research/very_deep/
  - VGG-Face (VGG-16 entrenada con rostros)
    - http://www.robots.ox.ac.uk/~vgg/software/vgg_face/
- Redes muy grandes! (~500 MB)

Simonyan, Zisserman. Very Deep Convolutional Networks For Large-Scale Image Recognition. 2014.
https://arxiv.org/pdf/1409.1556.pdf

ResNet (2015)

- 152 capas
  - Una convolución de 7x7 y max-pooling, luego muchas convoluciones de 3x3…
- Bloque Residual
  - Intenta evitar el problema del “vanishing gradient”
  - Similar a una RNN



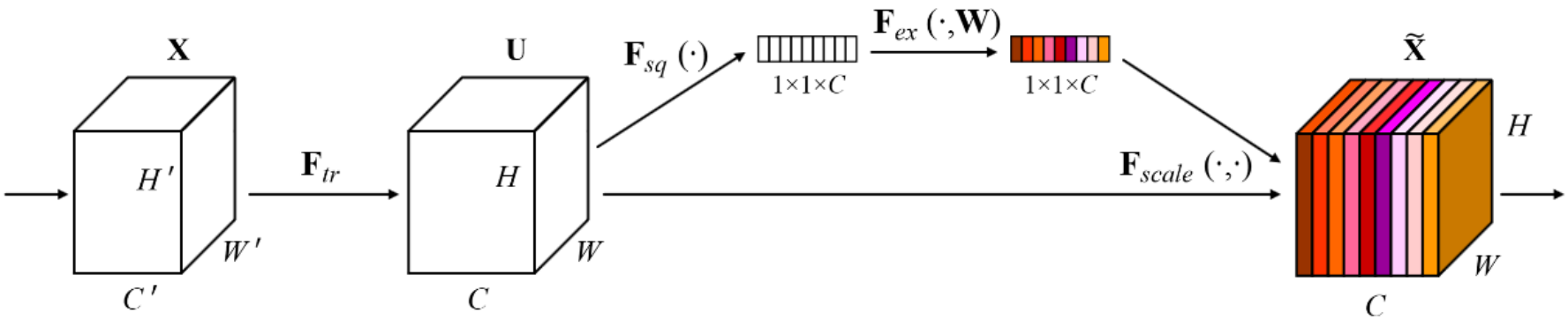


https://towardsdatascience.com/understanding-and-visualizing-resnets-442284831be8

He et al. Deep Residual Learning for Image Recognition. 2015.
http://openaccess.thecvf.com/content_cvpr_2016/papers/He_Deep_Residual_Learning_CVPR_2016_paper.pdf

SENet (2017)

- Squeeze-and-Excitation block
  - Entrenar un parámetro para escalar cada canal de la entrada al bloque



Hu et al. Squeeze-and-Excitation Networks. 2017.

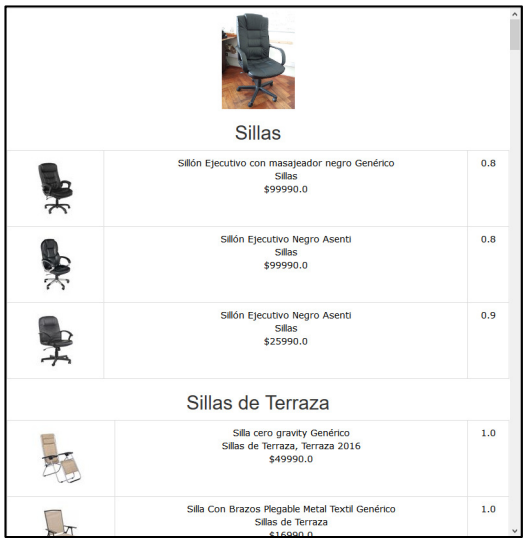
http://openaccess.thecvf.com/content_cvpr_2018/papers/Hu_Squeeze-and-Excitation_Networks_CVPR_2018_paper.pdf

Deep Features

- Obtener un descriptor global de imagen usando valores de una capa previa al output de clasificación
  - Para AlexNet y VGG usualmente capa fc6 o fc7 (4096)
  - Para ResNet la capa de la última convolución (2048)
- Descriptor con propiedades similares al obtenido a través de Bag-Of-Visual-Words
- Vector que permite encontrar objetos visualmente parecidos al objeto en la imagen de consulta

Uso de Deep Features

- Deep Features son descriptores de contenido que capturan información de alto y bajo nivel
  - Comparar descriptores con distancia L_2 o L_1 o coseno
  - Descriptores cercanos tienden a ser de la misma clase y además visualmente similares



Problema de Deep Features

- Deep Features no funciona muy bien buscando imágenes que comparten alguna zona (por ejemplo un mismo logo)
  - Caso productos de supemercado

Deep Features:

Similitud basada en Deep Features (como AlexNet) busca imágenes que contengan un objeto parecido (ignora el logo)

Descriptores Locales:

Similitud basada en patches similares (encuentra todos los productos con un mismo logo y colores)

Comparación con SIFT

- **Deep Features**
  - Usualmente son entrenados con ImageNet (1000 clases)
  - Representan características que identifican el tipo de objeto visible (muy semánticos)
  - No funciona muy bien para patrones visuales sin semántica (texturas, logos)
  - No funciona muy bien en problemas que no contengan los objetos de ImageNet
  - Es posible evitar los problemas anteriores pero se requiere un dataset y entrenamiento adhoc al problema
- **Descriptores locales** (SIFT o BOVW)
  - Basados en representar patches de imágenes (poco semánticos)
  - Robusto a rotaciones y oclusión parcial
  - Funcionan bien para encontrar logos y patrones de bajo nivel
  - Falla al buscar objetos según definición semántica

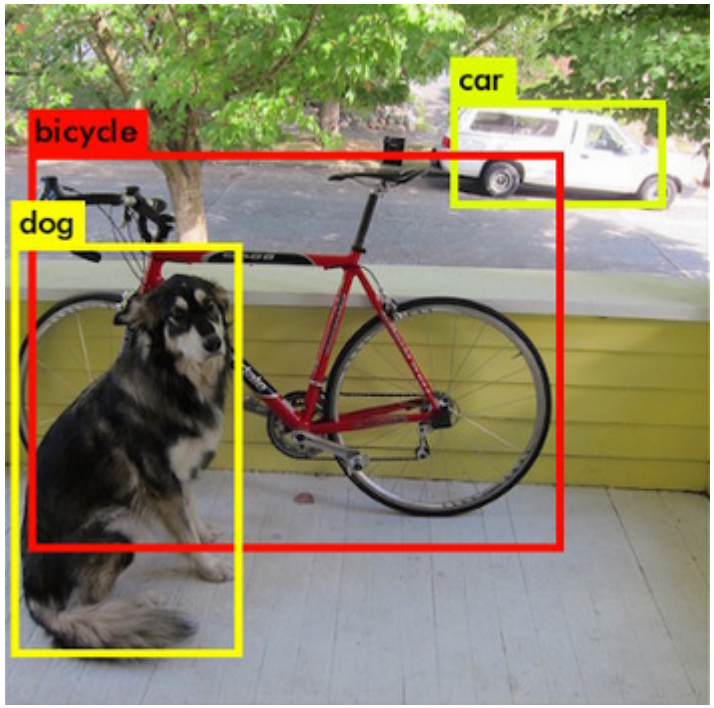
Ejercicio

- Una multitienda que vende productos en su página web contrata sus servicios:
  - Los productos están organizados en 20 categorías y cada categoría tienen 200 productos distintos
  - Cada producto cuenta con una única fotografía

- ¿Cómo implementaría un clasificador que dada la fotografía de un producto nuevo señale la **categoría de la tienda** a la que debe asignarse?
- ¿Cómo implementaría ese clasificador si algunas categorías tienen **pocos productos**? (ej: uno o dos)
- ¿Cómo implementaría ese clasificador si la tienda cuenta con un **gran árbol de categorías**? (10 mil categorías, organizadas en 4 niveles, cada nivel con 10 subcategorías)

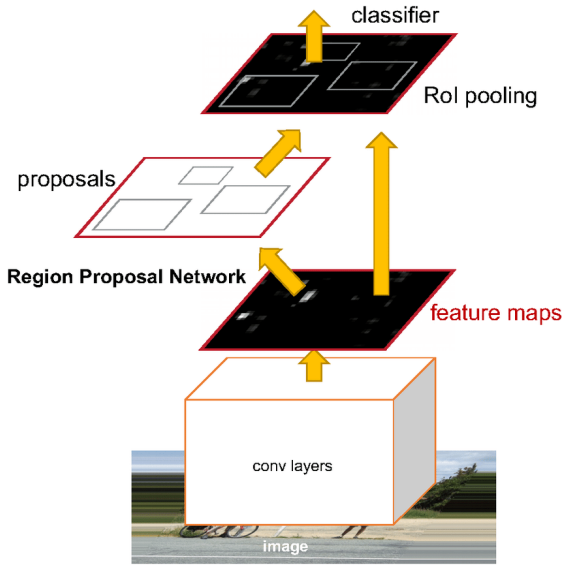
CNN con Regiones (R-CNN)

- Se desea clasificar una imagen y además dar un recuadro de la ubicación



Faster R-CNN

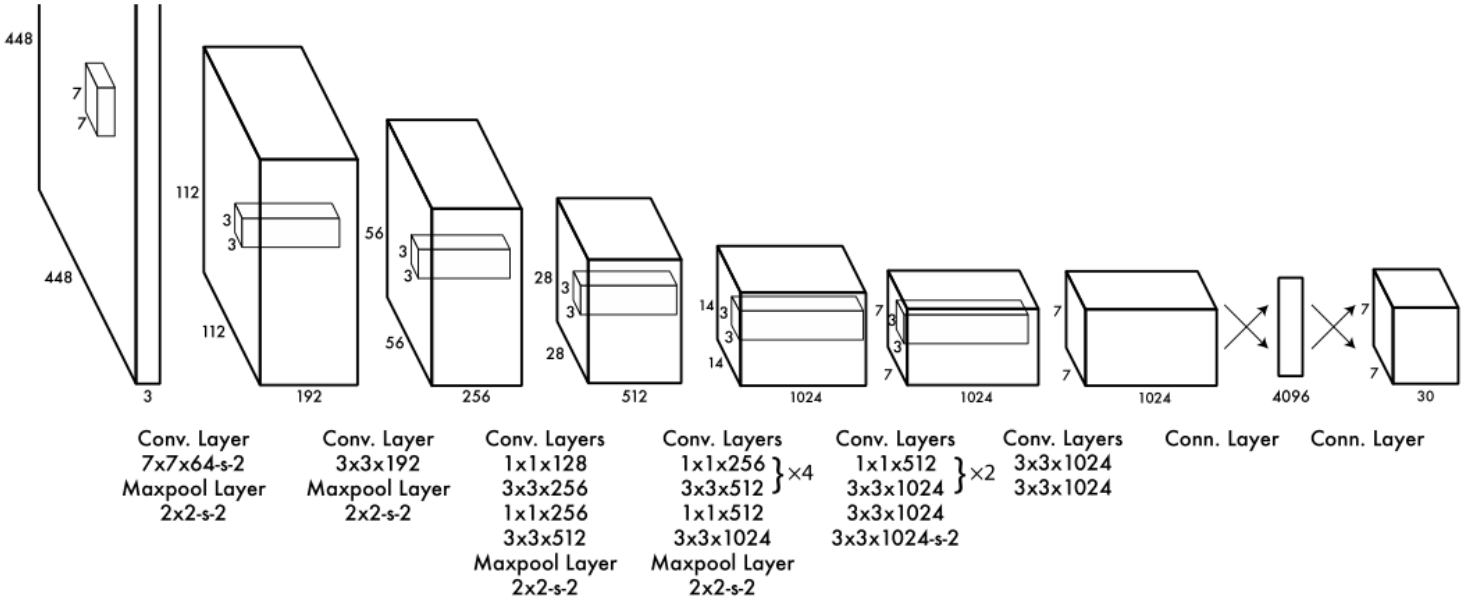
- Se usa en dos etapas: encontrar una propuesta de zonas y luego clasificar las zonas



Ren et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. 2015.
http://papers.nips.cc/paper/5638-faster-r-cnn-towards-real-time-object-detection-with-region-proposal-networks.pdf

YOLO

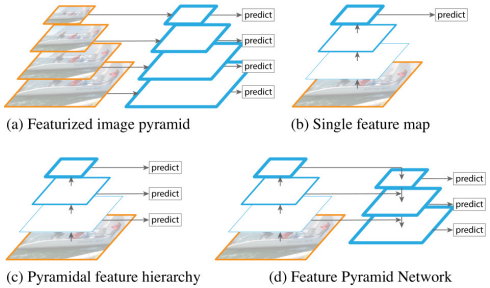
- Entrega cajas y clases en una única regresión

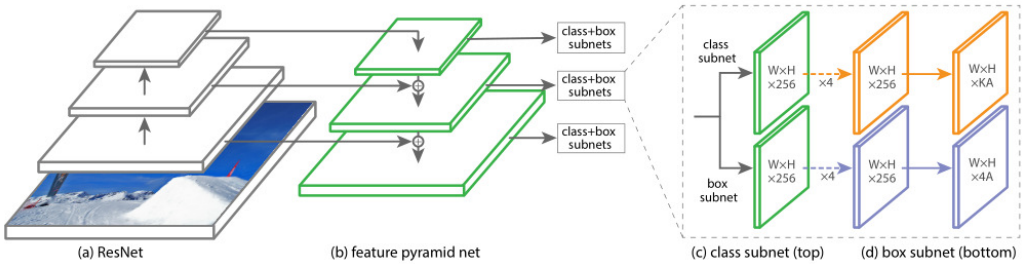


Redmon et al. You Only Look Once: Unified, Real-Time Object Detection. 2016.
https://www.cv-foundation.org/openaccess/content_cvpr_2016/papers/Redmon_You_Only_Look_CVPR_2016_paper.pdf

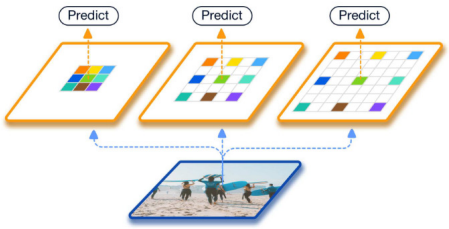
Bochkovskiy et al. YOLOv4: Optimal Speed and Accuracy of Object Detection. 2020.
https://arxiv.org/pdf/2004.10934

Feature Pyramids





Lin et al. Feature Pyramid Networks for Object Detection. 2017.
http://openaccess.thecvf.com/content_cvpr_2017/papers/Lin_Feature_Pyramid_Networks_CVPR_2017_paper.pdf



(RetinaNet) Lin et al. Focal Loss for Dense Object Detection. 2018.
https://openaccess.thecvf.com/content_ICCV_2017/papers/Lin_Focal_Loss_for_ICCV_2017_paper.pdf

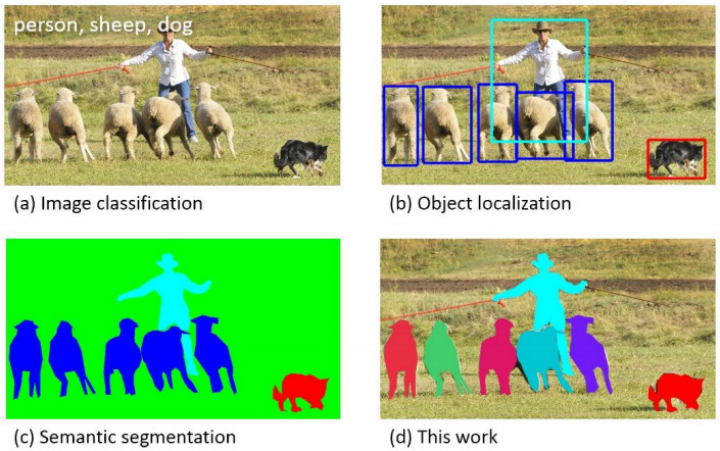
(TridentNet) Li et al. Scale-Aware Trident Networks for Object Detection. 2019.
https://openaccess.thecvf.com/content_ICCV_2019/papers/Li_Scale-Aware_Trident_Networks_for_Object_Detection_ICCV_2019_paper.pdf

Uso de RCNN

- RCNN permite localizar uno o más objetos en la imagen y la clase de cada objeto
- Usualmente con dataset **COCO Detection:**
  - 80 clases
  - Funciona muy bien para identificar la personas
  - No funciona bien con frames de videos de baja calidad
  - Existen otros datasets recientes para entrenar RCNN
- ¿Cómo detectar una mayor cantidad de objetos?

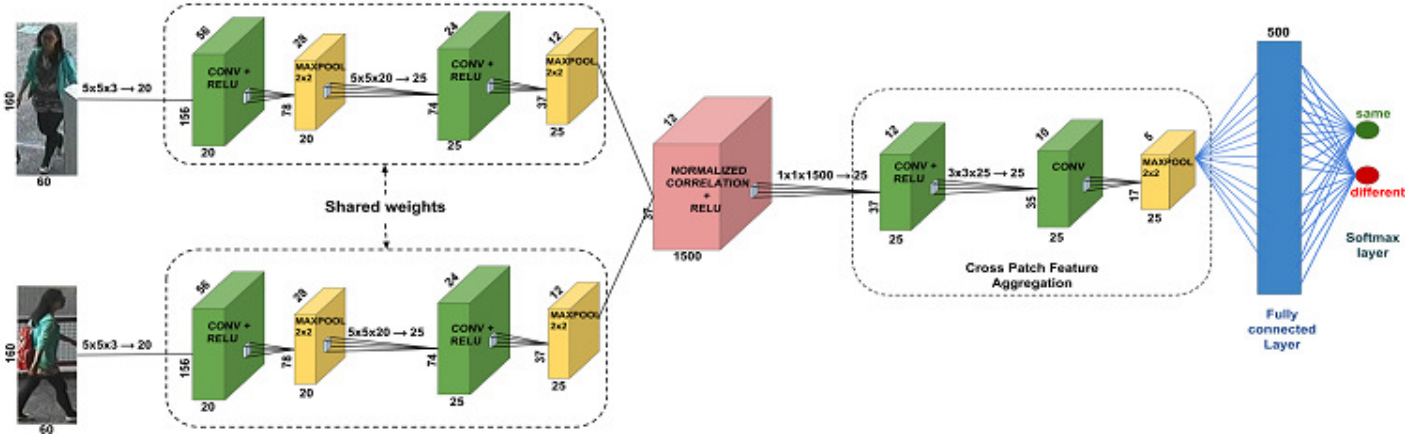
Segmentación de Imágenes

- La salida de la red es una máscara del tamaño de la imagen de entrada que:
  - Marca la ubicación de las clases presentes en la imagen
  - Marca la ubicación de los objetos presentes en la imagen



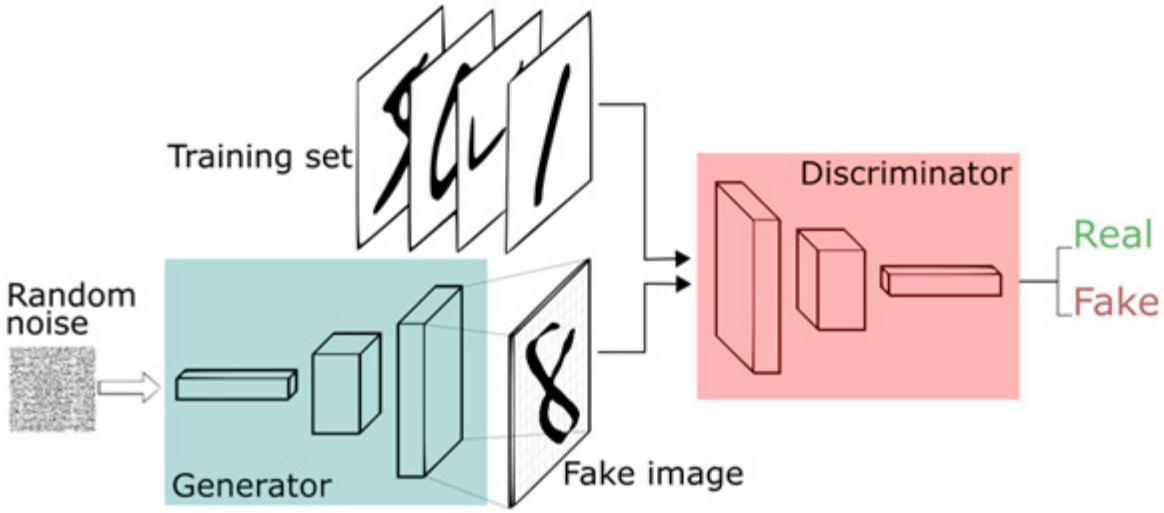
Redes Siamesas

- Una red siamesa es un tipo de CNN que procesa dos imágenes
  - Ambas imágenes se evalúan con los mismos pesos al inicio
  - Capas intermedias entrenan la comparación entre imágenes
- La salida de la red es:
  - Binaria, las imágenes son iguales o no
  - Continua, valor de similitud entre las imágenes



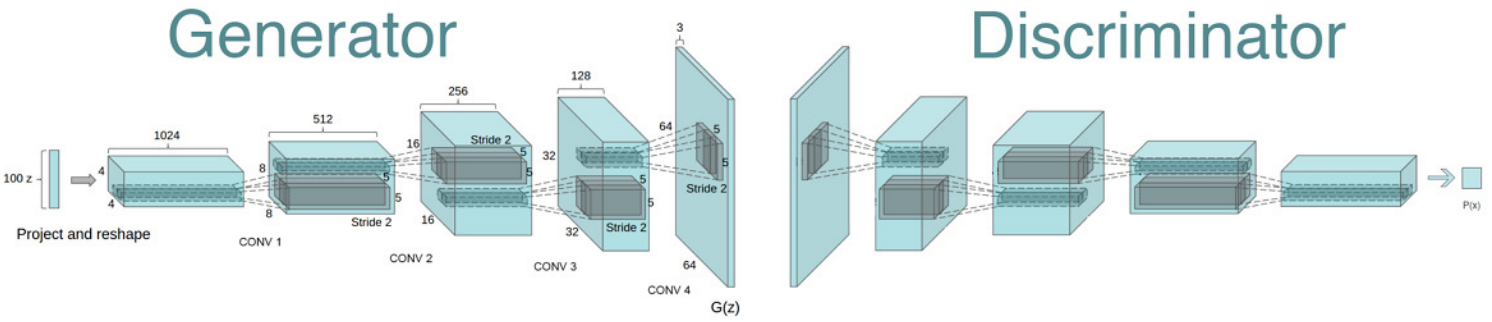
Redes Generativas

- Entrenar una red que permita generar imágenes realistas, es decir, que sea indistinguible de un conjunto de imágenes reales
- La primera red (generador)
  - Entrada: un valor semilla
  - Salida: una imagen generada (falsa)
- Segunda red (discriminador)
  - Entrada: una imagen (real o falsa)
  - Salida: un valor 0 si es imagen real un 1 si es imagen falsa



Redes Generativas

- Entrenamiento de cada red en forma alternada:
  1. El discriminador se entrena para distinguir entre imágenes reales e imágenes del generador (inicialmente al azar)
  2. El generador se entrena para confundir al discriminador
  3. El discriminador se entrena para diferenciar entre las imágenes del generador
  4. El generador se entrena para confundir al discriminador …
- Luego de mucho ciclos el generador produce imágenes similares a las del conjunto de entrenamiento



https://medium.com/@awjuliani/generative-adversarial-networks-explained-with-a-classic-spongebob-squarepants-episode-54deab2fce39

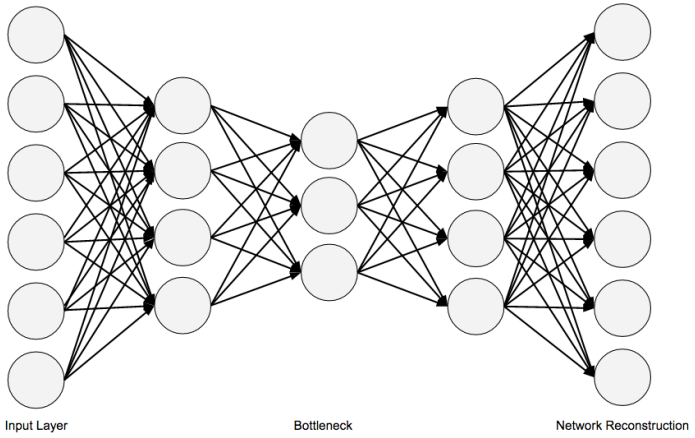
Redes Generativas

- Algunos usos de GANs:
  - Generar imágenes para un conjunto de entrenamiento de algún problema
  - Generación de fotos de rostros
    - https://thispersondoesnotexist.com/
  - Conversión de fotos en versión anime
    - https://waifu.lofiu.com/
  - Super resolución
  - Generación de objetos en escenarios
  - Conversión de dibujos en imágenes realistas
  - Generación de fotos a partir de texto

https://machinelearningmastery.com/impressive-applications-of-generative-adversarial-networks/

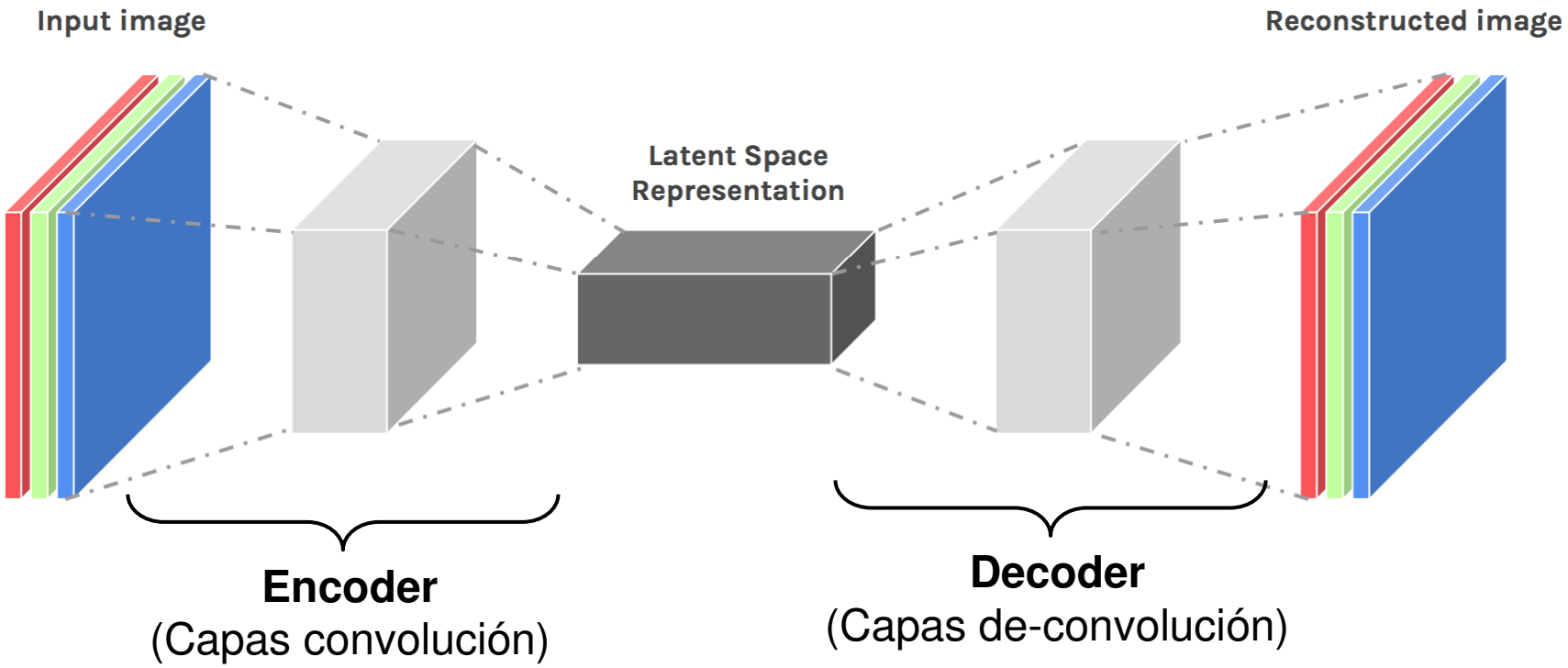
Auto-Encoders

- Red neuronal donde la entrada y salida tienen el mismo tamaño y contiene una capa oculta de menor tamaño
- Se entrena para que la salida sea idéntica a la entrada
  - Debe aprender patrones comunes en el dataset para poder reconstruir exitosamente todos los vectores
  - Cada neurona se va especializando en detectar patrones típicos
- Método no supervisado: usar cuando se tienen muchos datos sin etiquetar



CNN Auto-Encoders

- La entrada es una imagen, la salida es la misma imagen
  - En la capa intermedia se obtiene una representación comprimida del objeto
- La entrada es una imagen con ruido, la salida es la imagen sin ruido
  - Entrena un “denoiser”



CNN Auto-Encoders

- Para autoenconders usando CNN se requiere un paso de “deconvolution” (o transposed convolution) y “unpooling”
  - Agrandar una imagen y deshacer una convolución
- Es posible entrenar un autoencoder con muchos frames de videos de Youtube.
  - Neuronas se especializan en patrones comunes, por ejemplo detectar gatos

https://blog.manash.me/implementing-pca-feedforward-and-convolutional-autoencoders-and-using-it-for-image-reconstruction-8ee44198ea55

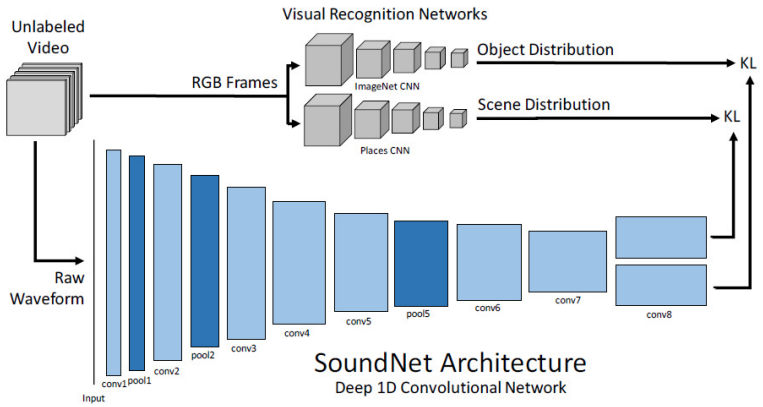
•Zeiler et al. Deconvolutional Networks. 2010
•Zeiler et al. Adaptive Deconvolutional Networks for Mid and High Level Feature Learning. 2011
•Le et al. Building High-level Features Using Large Scale Unsupervised Learning. 2012.

Uso de Auto-Encoders

- Calcular descriptores de contenido:
  - Se requieren muchos datos para que el entrenamiento logre detectar patrones relevantes
- Entrenar un denoiser:
  - Entrenar el auto-encoder donde la entrada es el objeto con ruido (agregado artificialmente) y la salida es el objeto original sin ruido
  - Muy útil para limpiar ruido de fondo en audio
- Detección de anomalías:
  - Si se usa el auto-enconder con una imagen muy distinta a los datos de entrenamiento (anomalía), la salida tendrá muy mala calidad y se podrá detectar

CNN para Audio

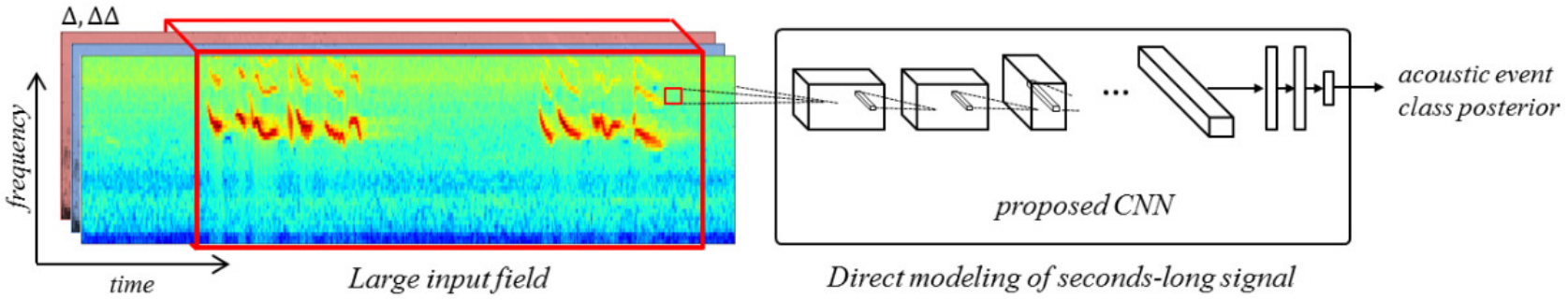
- Transfer Learning de Imágenes a Audio
  - Entrena un clasificador de audio a partir un clasificador de imagen
  - Asume que lo que es visible es lo que se está escuchando
  - Toma cada frame del video, lo ingresa a una red convolucional pre-entrenada con ImageNet y Places, obtiene la salida y entrena la red de audio para que genere la misma salida



Aytar et al. SoundNet: Learning Sound Representations from Unlabeled Video. 2016
http://soundnet.csail.mit.edu/

AENet

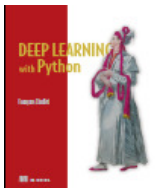
- Descriptor de audio



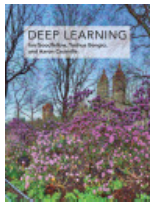
Takahashi et al. AENet: Learning Deep Audio Features for Video Analysis. 2017
https://arxiv.org/pdf/1701.00599.pdf
https://github.com/znaoya/aenet

Bibliografía

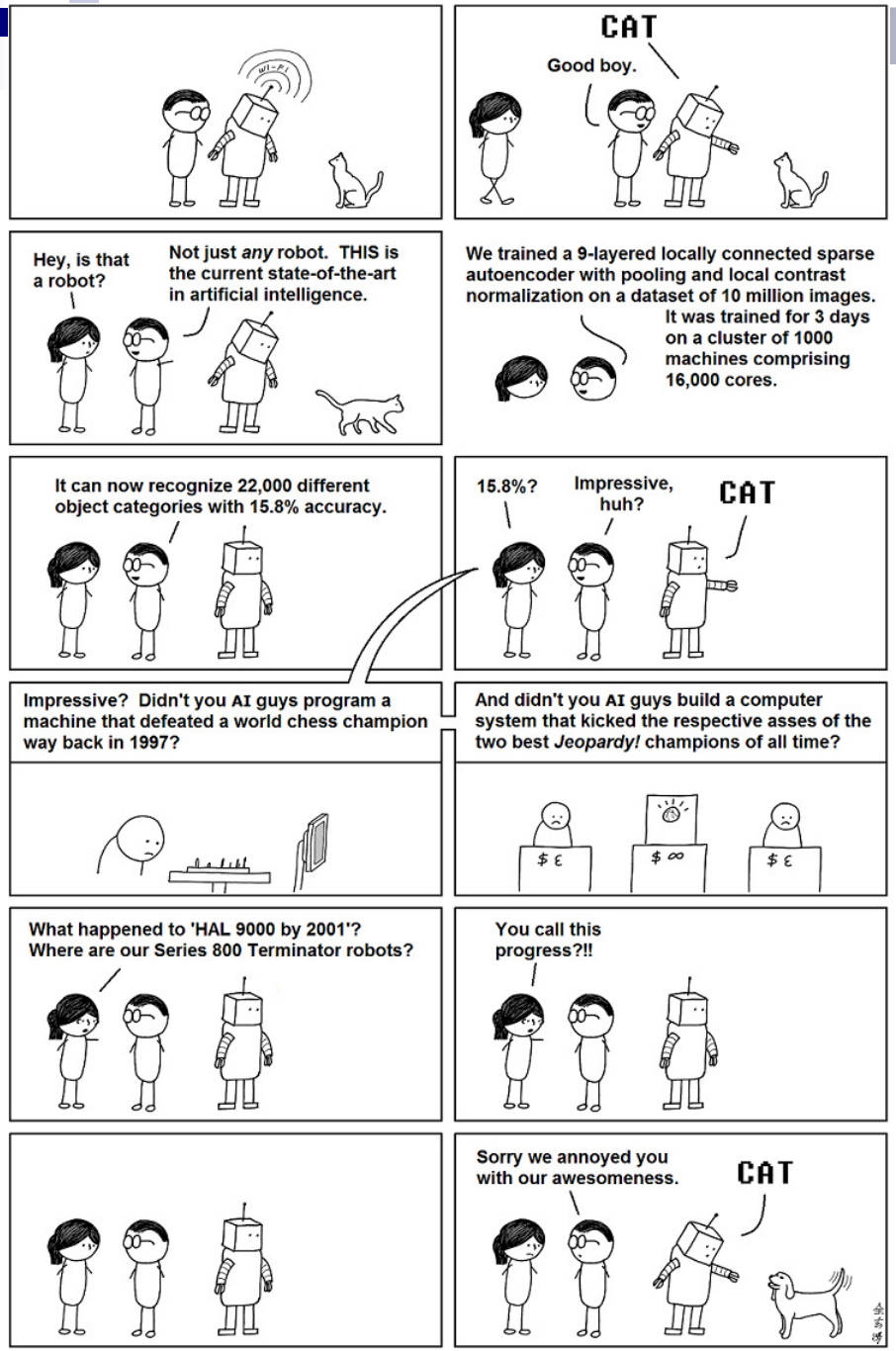
- **Deep Learning with Python**. Chollet. 2018.



- **Deep Learning**. Goodfellow, Bengio, Courville. 2016.



- Curso de Stanford (https://cs231n.github.io/)
  - https://cs231n.github.io/neural-networks-1/
  - https://cs231n.github.io/neural-networks-2/



http://abstrusegoose.com/496

Hace referencia a:
http://research.google.com/archive/unsupervised_icml2012.html

http://xkcd.com/1425/

(2014)

http://xkcd.com/1444/

(2014)