



Aprendizaje de Máquinas

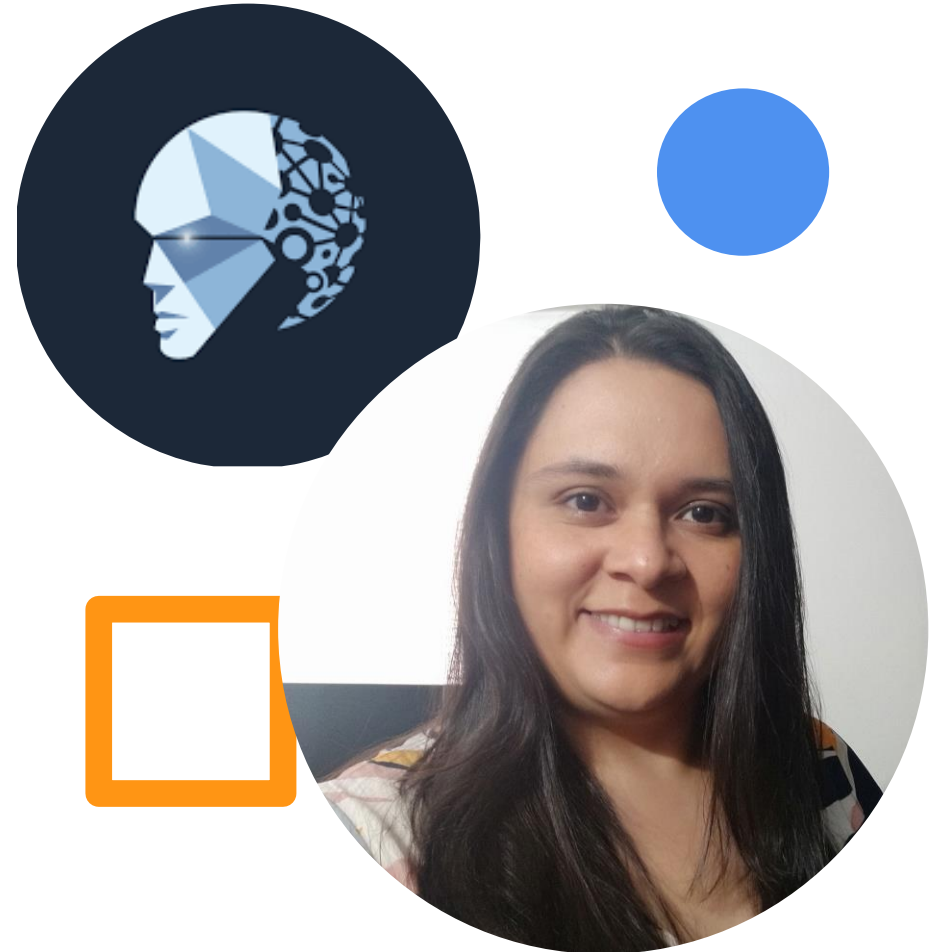
Maria C. Torres


Maria C. Torres

Ing. Electrónica (UNAL)
M.E. Ing. Eléctrica (UPRM)
Ph.D. Ciencias e Ingeniería de la
Computación y la Información (UPRM)
Profesora asociada
Dpto. Ciencias de la Computación y la
Decisión

mctorresm@unal.edu.co

Oficina: Bloque M8A 313





Contenido del Curso

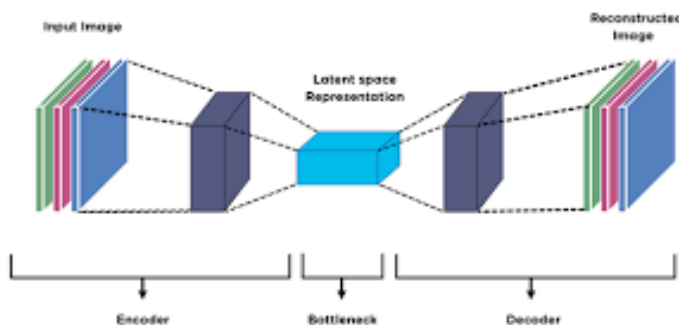
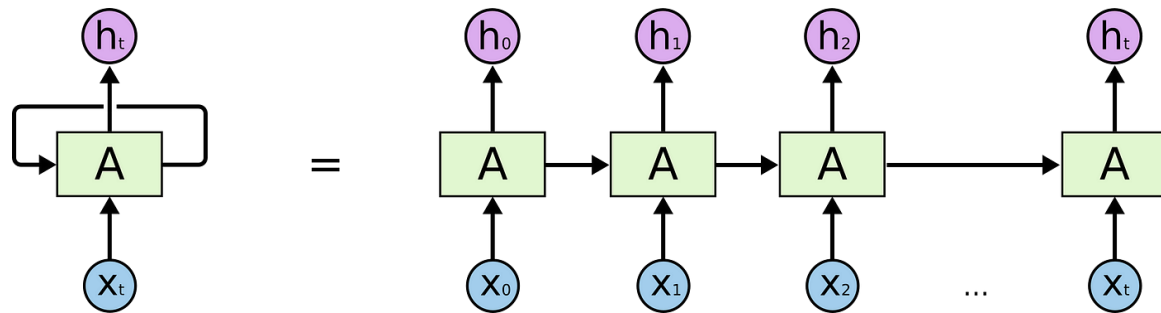
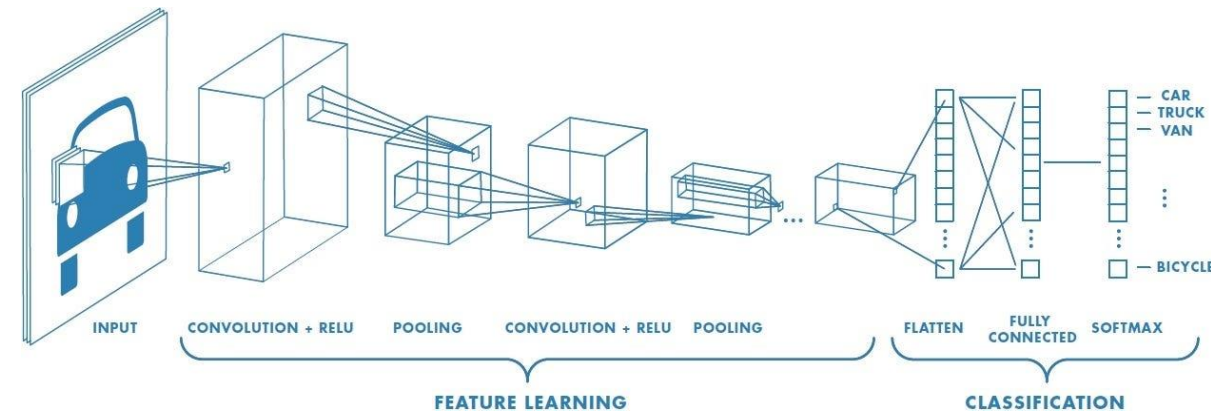
- ☐ Intro a ML
- ☐ **Aprendizaje supervisado**
- ☐ Aprendizaje no supervisado
- ☐ Aprendizaje reforzado



Aprendizaje Supervisado

- ☐ Regresión y Clasificación
- ☐ Métricas de desempeño
- ☐ Regresión lineal, polinomial y regularizaciones
- ☐ Clasificación bayesiana
- ☐ Vecinos más cercanos
- ☐ Máquinas de soporte
- ☐ Árboles de decisión
- ☐ Redes neuronales
- ☐ Métodos de ensamble
- ☐ Bosques aleatorios
- ☐ **Redes profundas**

Deep Learning – Aprendizaje Profundo



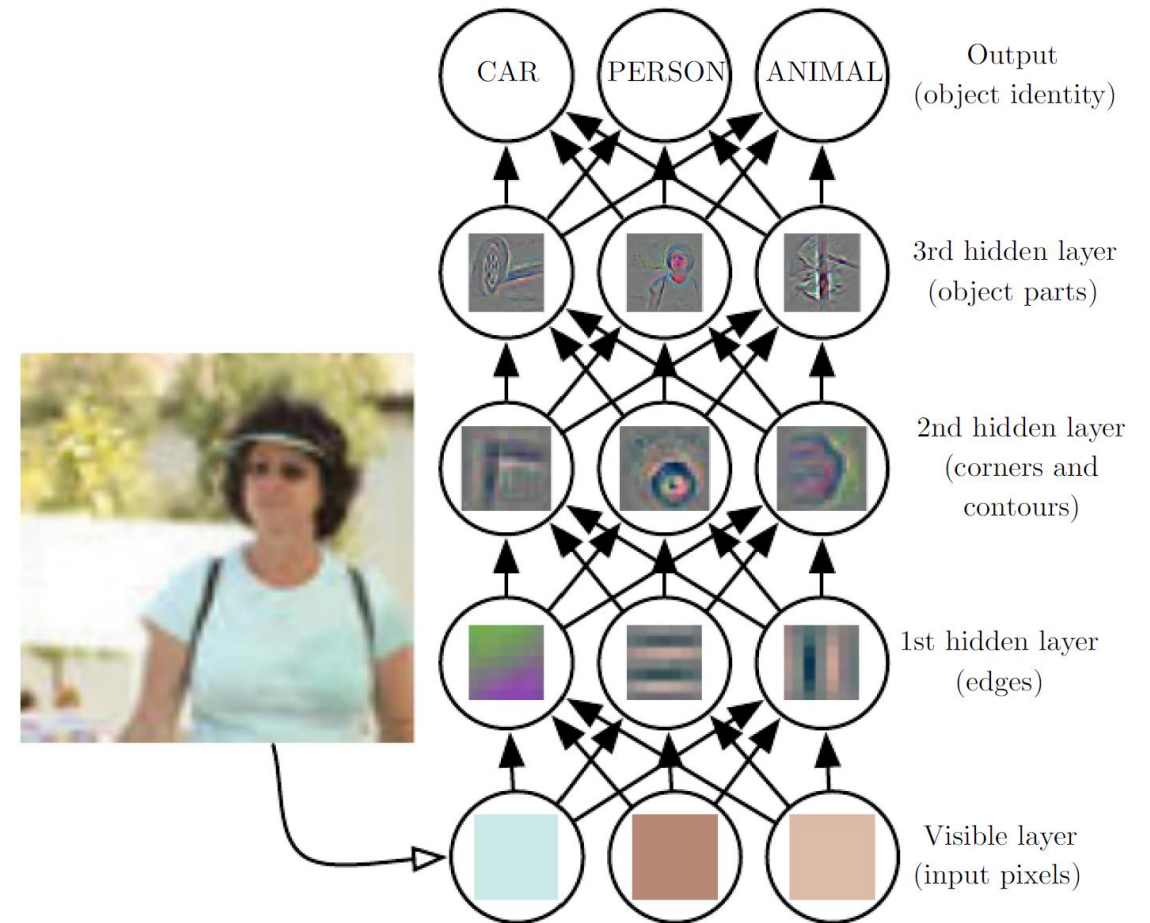
- ❑ DL es más que redes neuronales con más de dos capas ocultas
- ❑ Las arquitecturas en DL son más complejas y se caracterizan por:
 - Más neuronas que en tradicionales NN
 - Conexiones más complejas entre neuronas y capas
 - Toman ventaja de la capacidad de cómputo para el entrenamiento
 - Realizan la extracción de características de forma automática
- ❑ Entre las arquitecturas de redes en DL se encuentran:
 - Redes neuronales convolucionales
 - Redes neuronales recurrentes
 - Redes preentrenadas no supervisadas

Por qué se denomina aprendizaje profundo?

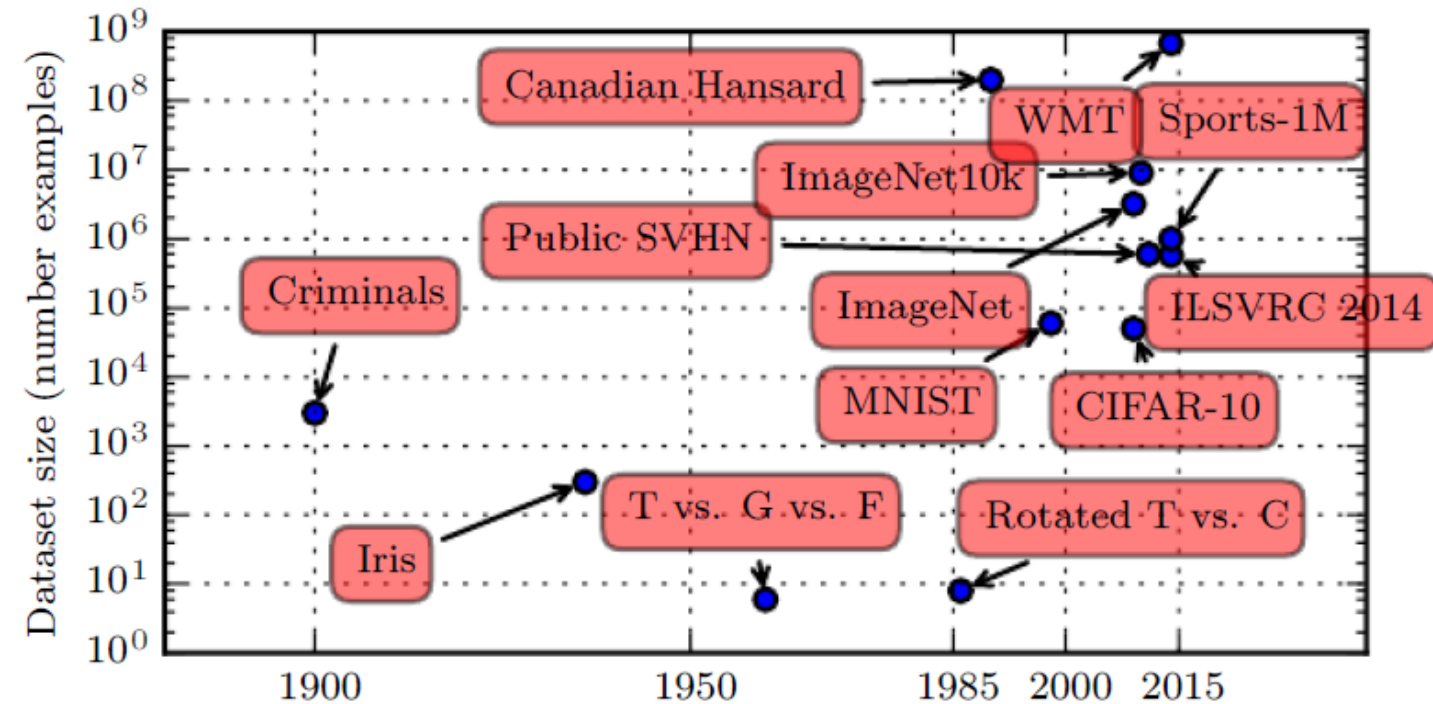
Representación del aprendizaje

- ❑ DL soluciona una los problemas centrales en la búsqueda de representar el aprendizaje
- ❑ DL expresa una representación compleja en términos de otras más simples
- ❑ Ejemplo: en visión por computador, una imagen se representa por conceptos más simples como sus bordes, contornos, partes de los objetos, etc.
- ❑ Las capas ocultas son las encargadas de extraer las características abstractas desde los datos de entrada.
- ❑ Las capas se denominan ocultas porque sus valores deben ser determinados en el entrenamiento, de tal forma que expliquen los patrones de los datos.

[Zeiler & Fergus, 2014]



Deep Learning – Aprendizaje Profundo



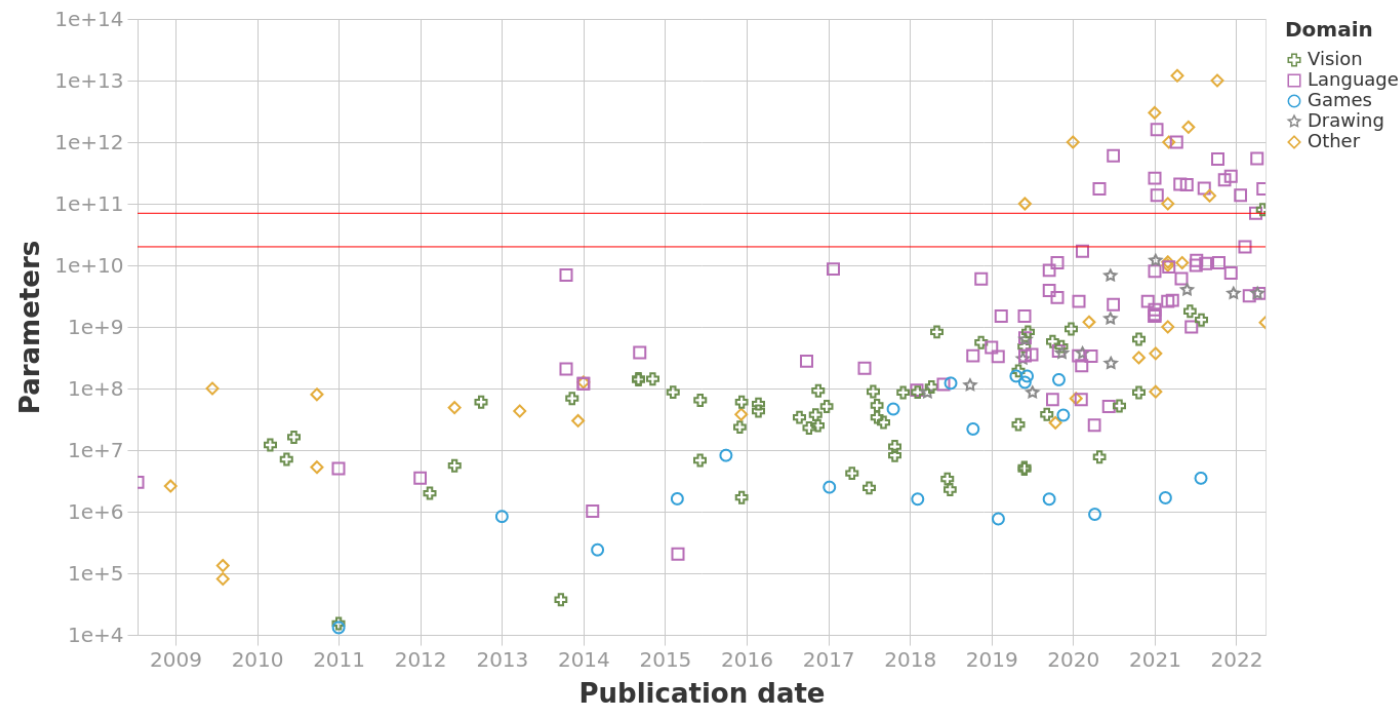
- ❑ Uno de los factores que ha contribuido al desarrollo de DL es la disponibilidad de datos de entrenamiento
- ❑ La era de la BigData ha facilitado el desarrollo de los modelos de ML
- ❑ Se estima que un desempeño aceptable de un sistema DL se puede obtener con 5000 muestras etiquetadas, y que puede exceder el desempeño de un humano con un conjunto de datos de 10 millones de muestras
- ❑ Es un tópico de investigación, desarrollar sistemas no supervisados o semi-supervisados que puedan lograr buenos desempeños con pocos datos

Deep Learning – Aprendizaje Profundo

- ❑ Otro de los elementos que ayudo a posicionar DL es la facilidad para construir complejas redes neuronales
- ❑ Las arquitecturas DL aun están lejos de parecerse a las conexiones neuronales del ser humano, sin embargo, las redes actuales permiten solucionar problemas más complejos que los MLP de los 90's
- ❑ Los modelos DL han incrementado de tamaño significativamente debido al incremento de capacidad en la infraestructura computacional (hardware y software).

Parameters of milestone Machine Learning systems over time

n = 203



Deep Learning – Aprendizaje Profundo

❑ Existen diferentes plataformas para el desarrollo de sistemas DL, algunos de estos son:

Keras

- Lenguaje: Python
- Fácil uso - Amigable con el usuario
- Recomendado para experimentos y diseños rápidos
- API de alto nivel con modelos y capas pre-construidas
- Soportada por TensorFlow, Theano, CNTK
- Amplia comunidad que proporciona soporte



Chainer

- Lenguaje: Python
- Introduce el enfoque "define-by-run" que permite inspeccionar los datos como fluyen en la red
- Más rápido que otras plataformas principalmente en GPU

TensorFlow

- Lenguaje: Python, C++
- No tan fácil de usar
- Múltiples niveles de abstracción para la construcción de modelos
- Uso eficiente de memoria
- Amplia comunidad que proporciona soporte, librerías y herramientas
- Soporta Android



- Lenguaje: C++
- Se puede evaluar modelos con Python, C++, C#

PyTorch

- Lenguaje: Python
- Fácil de usar
- Múltiples niveles de abstracción
- Uso eficiente de memoria
- Comunidad en crecimiento, con un desarrollo activo desde el grupo de investigación de Facebook
- Recomendado para aplicaciones NPL

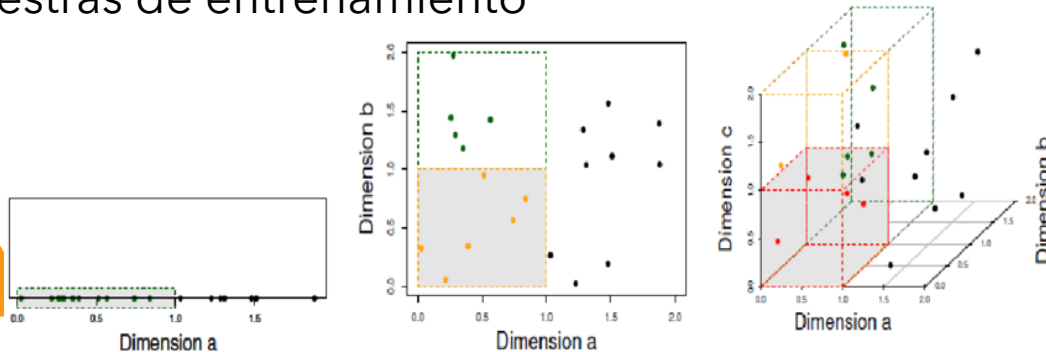
Caffe

- Lenguaje: Python, C++
- Fácil implementación de CNN
- Gran velocidad para el procesamiento de imágenes en GPU
- Popular en aplicaciones de visión por computador y multimedia
- Cuenta con un gran repositorio de modelos pre-entrenados

Motivaciones para el Aprendizaje Profundo

The Curse of Dimensionality

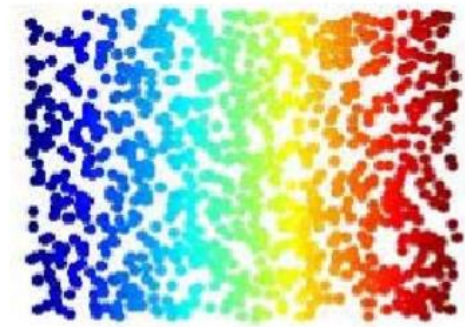
- ❑ La “maldición de la dimensionalidad” se refiere al limitado desempeño de los enfoques de ML en datos con espacios de características de muy altas dimensiones
- ❑ Los enfoques de ML son por naturaleza estadísticos, parten del conteo de las muestras de entrenamiento en regiones de interés para definir las fronteras de decisión
- ❑ Entre más característica tengamos los espacios de representación son de mayor dimensión, y por tanto, para generalizar una región requerimos más muestras de entrenamiento



Manifold learning



(a) Swiss roll



(b) Original manifold

- ❑ Son técnicas de reducción de dimensiones no lineal
- ❑ Los algoritmos para esta tarea se basan en la idea de que la dimensionalidad de muchos conjuntos de datos es sólo artificialmente alta
- ❑ Cuando los datos residen en un manifold de menor dimensión, los enfoques de ML funcionan mejor usando los datos proyectados en el manifold que el espacio de alta dimensión



Redes Profundas

- ☐ Deep feedforward network
- ☐ Convolutional neural network
- ☐ Recurrent neural network
- ☐ Autoencoders

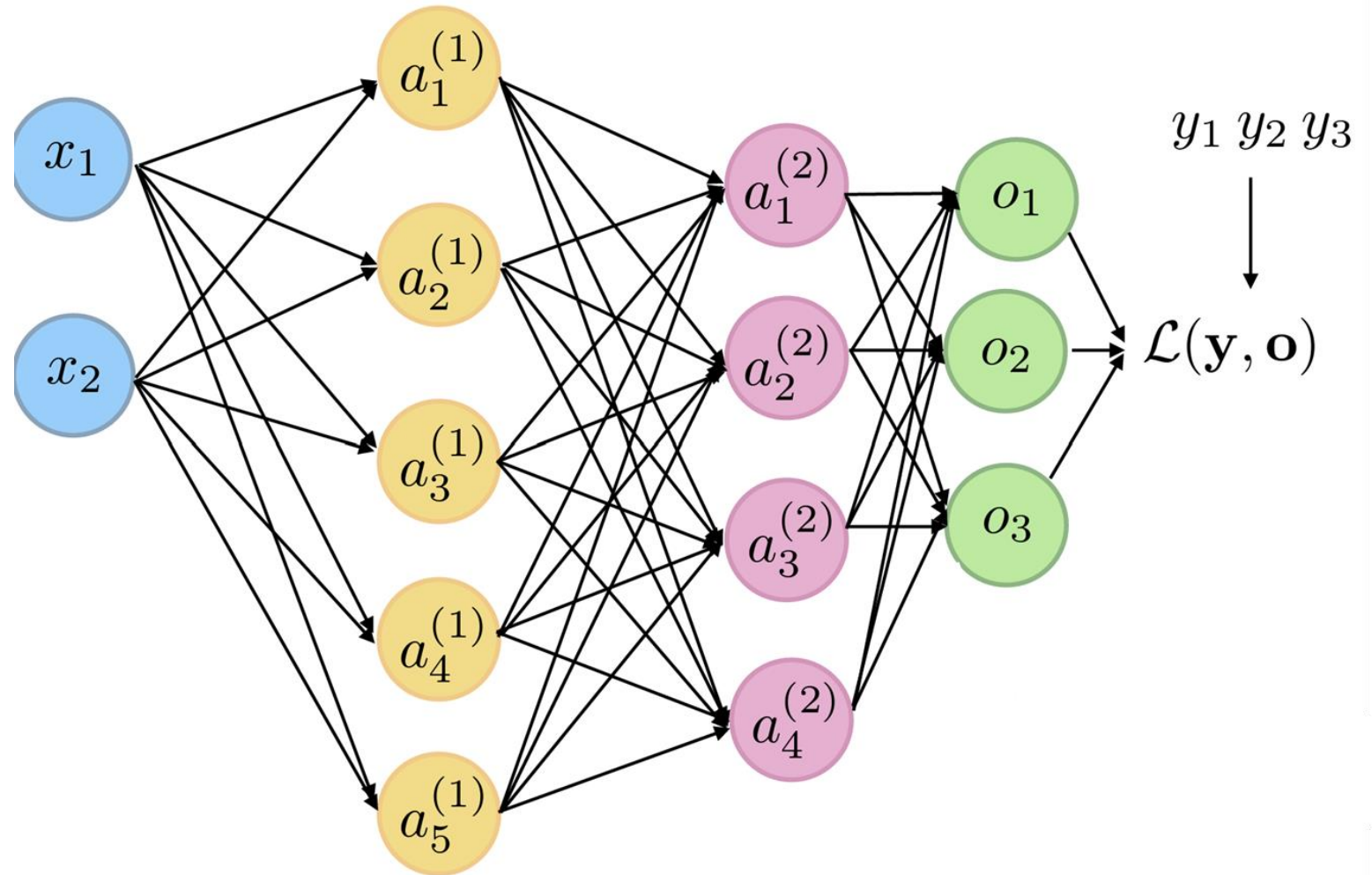


Redes Profundas

- ❑ **Deep feedforward network**
- ❑ Convolutional neural network
- ❑ Recurrent neural network
- ❑ Autoencoders

Deep Feedforward Network

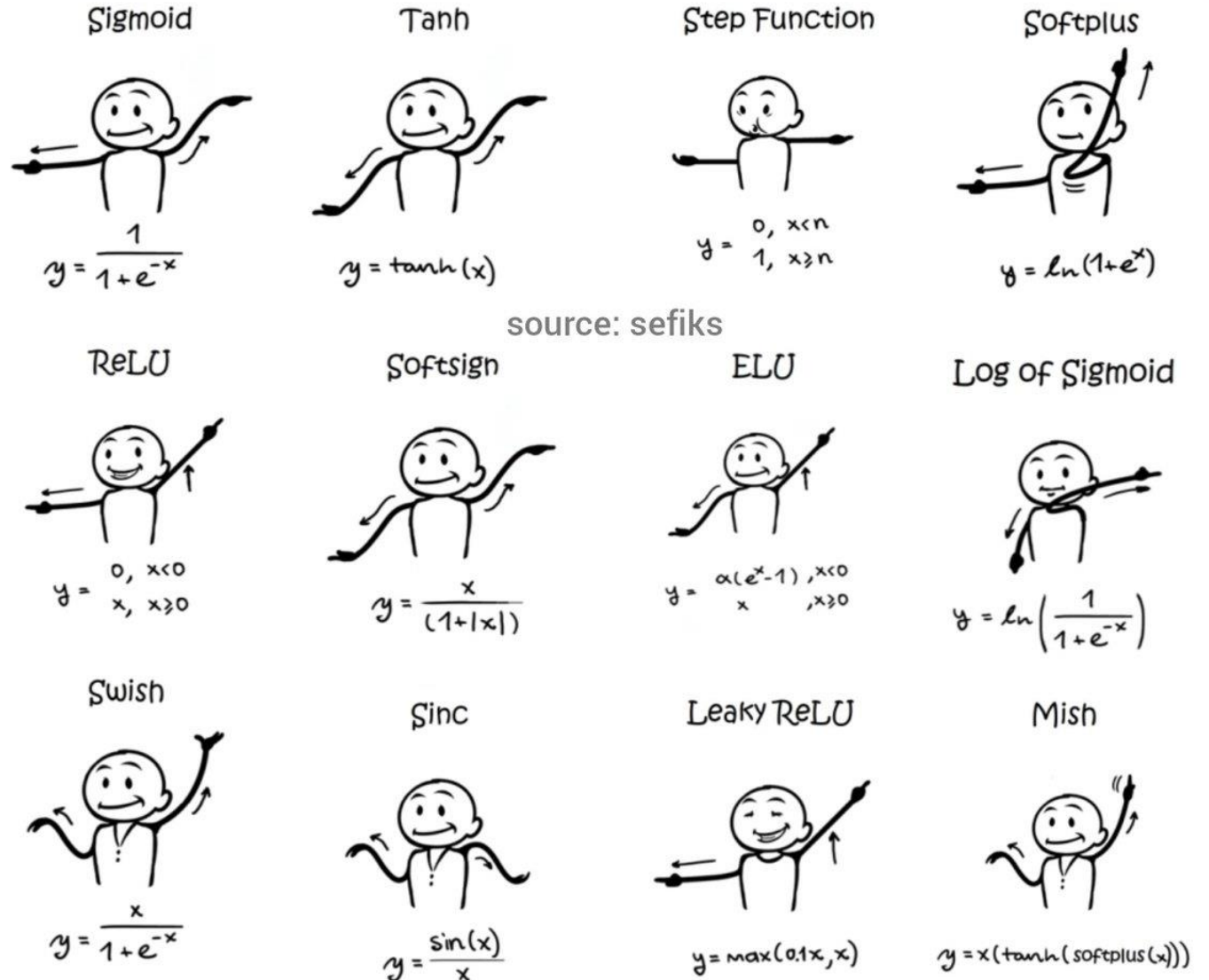
- ❑ Usualmente, si tiene más de una capa oculta ya se refiere a una red profunda
- ❑ El entrenamiento se realiza usando el algoritmo de "backpropagation"
- ❑ Dado que nos interesa aprender espacios de decisión no lineal, se agrega funciones de activación en cada capa oculta
- ❑ El problema de aprendizaje deja de ser convexo y tiene una alta complejidad



Deep Feedforward Network

- ❑ ReLU is la función de activación más popular: fácil de calcular y brinda buenos resultados
- ❑ Sin embargo, durante el entrenamiento variar neuronas pueden morir (gradiente 0): esto puede considerarse una forma de regularización

Dance Moves of Deep Learning Activation Functions



Mejorar el desempeño de generalización

❑ Conjunto de datos

- Recolectar más datos
- Aumentar los datos
- Suavizar el etiquetamiento
- Aprovechar datos no etiquetados: técnicas semi-supervisadas o autoaprendizaje

❑ Configuración de la arquitectura

- Estrategias de inicialización de pesos
- Funciones de activación

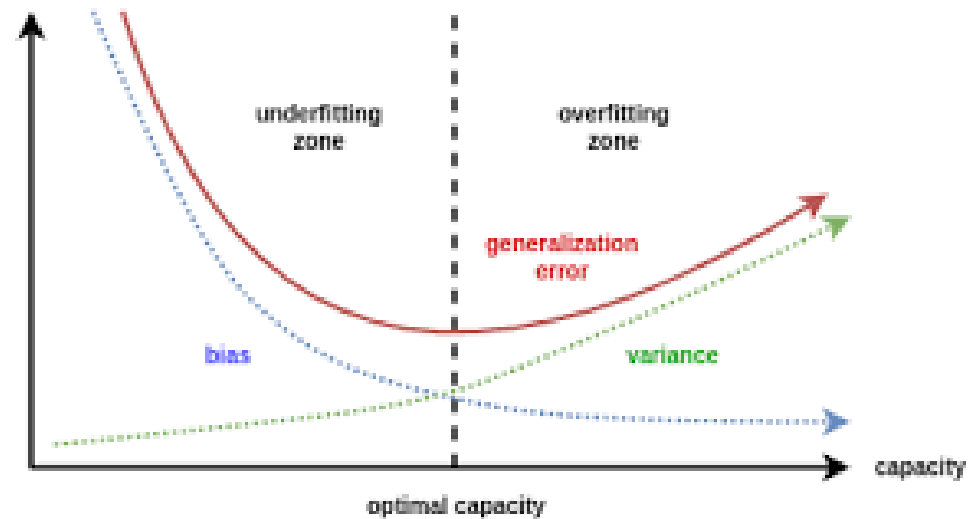
❑ Normalización

- Estandarización de entradas
- BatchNorm
- Estandarización ponderada

❑ Ciclo de entrenamiento

- Pasos de aprendizaje adaptivos

❑ Regularización

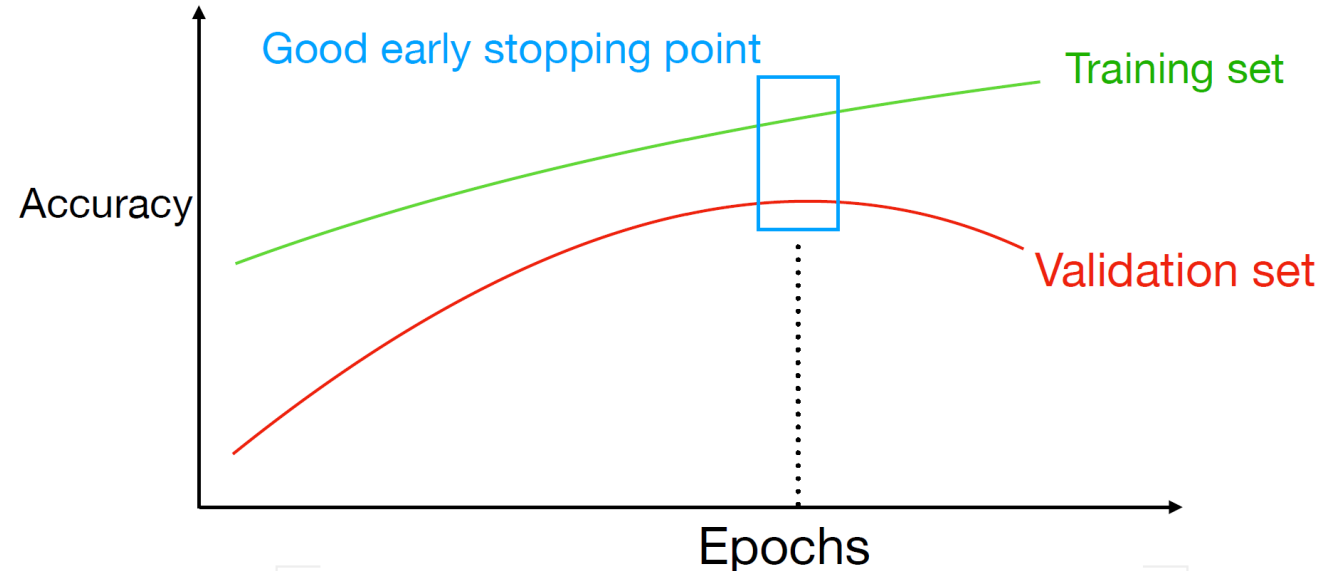


Regularización

- ❑ Se busca reducir el sobre-entrenamiento controlando la complejidad del modelo o reduciendo la varianza en las predicciones
- ❑ Las técnicas más comunes para DNN son:
 - Early stopping
 - Regularización por penalidad de la norma L1/L2
 - Dropout

Early stopping

- Reduce el sobreajuste observando la diferencia entre la precisión de entrenamiento y validación y parando el proceso en el “punto justo”



Regularización por penalidad

- ❑ La idea es la misma que en la regresión Ridge (regularización L2) y LASSO (regularización L1)
- ❑ Por ejemplo, la función de costo para el MLP con regularización L2 es:

$$\mathcal{L}_{regularizedL2} = \frac{1}{n} \sum_{i=1}^n \mathcal{L}(y^{[i]}, \hat{y}^{[i]}) + \underbrace{\frac{\lambda}{n} \sum_{l=1}^L \|w^{(l)}\|_2^2}_{\text{Suma sobre todas las capas}}$$

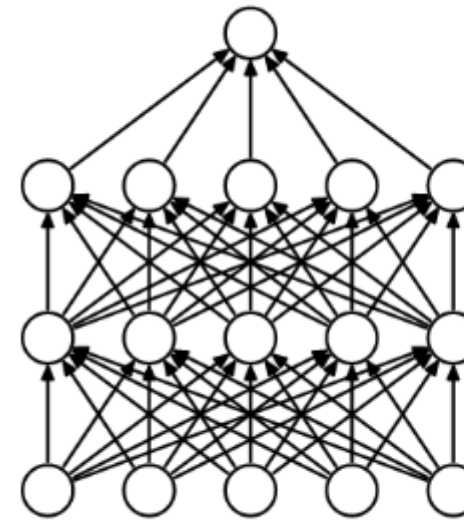
λ controla la regularización:

- Para valores altos se obtiene un alto sesgo
- Para valores bajo se obtiene una baja varianza

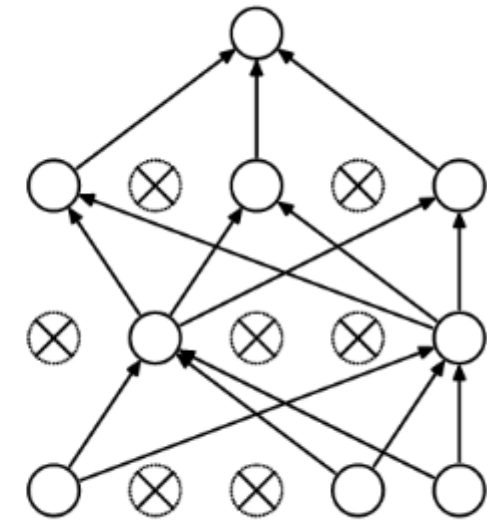
Dropout

- ❑ Se inspira en las técnicas de ensamble: usar múltiples redes con diferentes arquitecturas
- ❑ *Dropout* se refiere a “eliminar” nodos de una red neuronal (eliminando las conexiones hacia adelante y hacia atrás temporalmente), creando una arquitectura nueva desde la red principal.
- ❑ Los nodos son eliminados con una probabilidad igual a p
- ❑ Se recomienda 0.2 para probabilidad de eliminar (0.8 mantener el nodo)

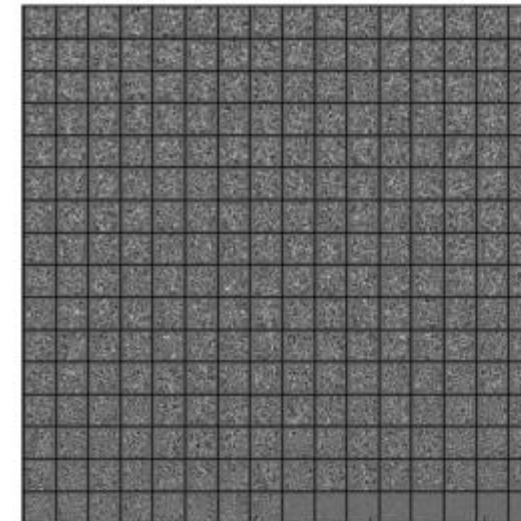
<https://towardsdatascience.com/dropout-in-neural-networks-47a162d621d9>



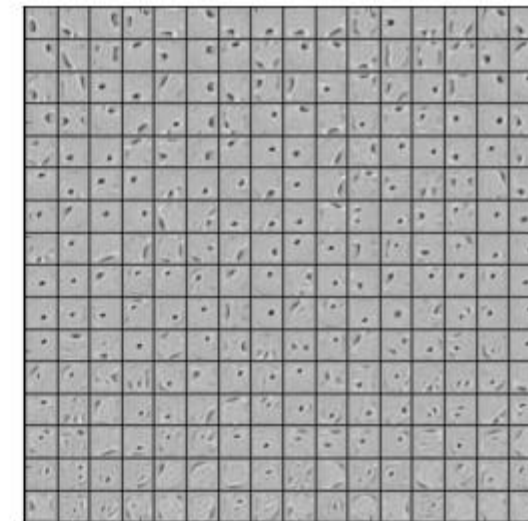
(a) Standard Neural Net



(b) After applying dropout.



(a) Without dropout



(b) Dropout with $p = 0.5$.

Dropout

Implementación

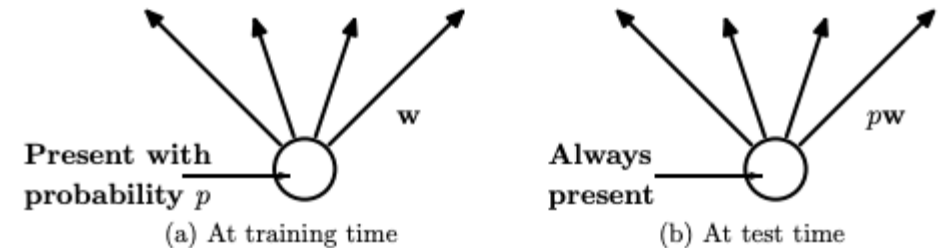
- ❑ Durante entrenamiento, una neurona en una capa se mantiene con una probabilidad de $1 - p$: esto crea una arquitectura más simple en el batch de entrenamiento (cada batch tendrá una arquitectura diferente)

Calculo forward en NN:

$$z_i^{(l+1)} = \mathbf{w}_i^{(l+1)} \mathbf{y}^l + \mathbf{b}_i^{(l+1)}$$
$$y_i^{(l+1)} = f(z_i^{(l+1)})$$

Calculo forward en NN con dropout:

$$r_j^{(l)} \sim \text{Bernoulli}(p)$$
$$\tilde{\mathbf{y}}^{(l)} = \mathbf{r}^{(l)} * \mathbf{y}^{(l)}$$
$$z_i^{(l+1)} = \mathbf{w}_i^{(l+1)} \tilde{\mathbf{y}}^l + \mathbf{b}_i^{(l+1)}$$
$$y_i^{(l+1)} = f(z_i^{(l+1)})$$



\mathbf{z} : vector salida de la capa $l + 1$ antes de aplicar la función de activación

\mathbf{y} : vector salida de la capa l

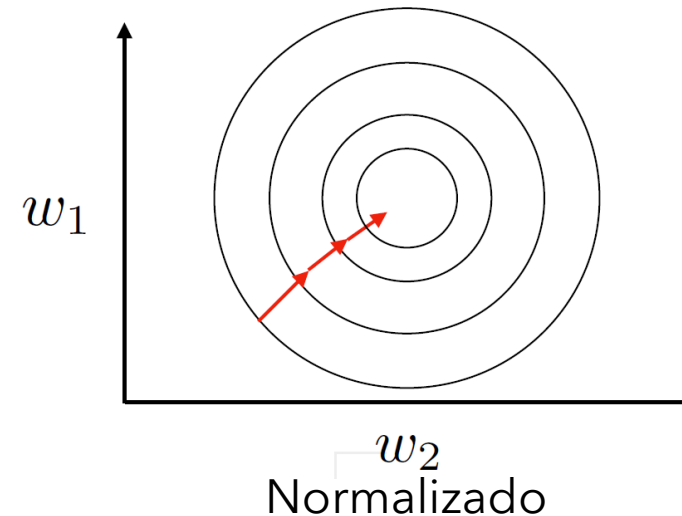
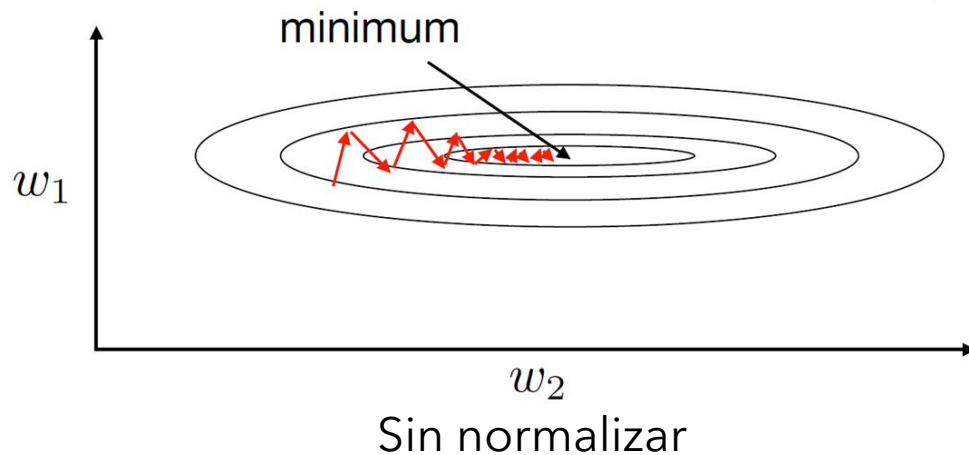
\mathbf{w} : pesos de la capa l

\mathbf{b} : bias de la capa l

Todas las unidades se consideran durante la predicción. Pero, los pesos finales serán mayores de lo esperado. Para solucionar este problema, los pesos se escalan por la probabilidad de eliminación p .

Normalización

- ❑ Durante el entrenamiento de una red neuronal profunda se hace necesario normalizar las entradas
 - Media cero y desviación estándar igual a 1
 - Valores entre $[0, 1]$
- ❑ De lo contrario, características de mayor tamaño dominar la actualización de los pesos



- ❑ Sin embargo, ¿esta normalización solo afecta la capa de entrada, que pasa con las capas ocultas?

Batch Normalization - BatchNorm

- ❑ Normaliza las entradas de las capas ocultas
- ❑ Ayuda a solucionar problemas con la explosión o desvanecimiento del gradiente
- ❑ Incrementa estabilidad en el entrenamiento y la tasa de convergencia
- ❑ Parámetros:
 - ϵ : se emplea por estabilidad numérica con un valor pequeño, por ejemplo: $1e - 5$
 - γ : controla la escala
 - β : controla el promedio

Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift

Sergey Ioffe, Christian Szegedy Proceedings of the 32nd International Conference on Machine Learning, PMLR 37:448-456, 2015.

<https://proceedings.mlr.press/v37/ioffe15.html>

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;

Parameters to be learned: γ, β

Output: $\{y_i = \text{BN}_{\gamma, \beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m} \sum_{i=1}^m x_i \quad // \text{ mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m} \sum_{i=1}^m (x_i - \mu_{\mathcal{B}})^2 \quad // \text{ mini-batch variance}$$

$$\hat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \quad // \text{ normalize}$$

$$y_i \leftarrow \gamma \hat{x}_i + \beta \equiv \text{BN}_{\gamma, \beta}(x_i) \quad // \text{ scale and shift}$$

Batch Normalization - BatchNorm

How Does Batch Normalization Help Optimization?

Shibani Santurkar*
MIT
shibani@mit.edu

Dimitris Tsipras*
MIT
tsipras@mit.edu

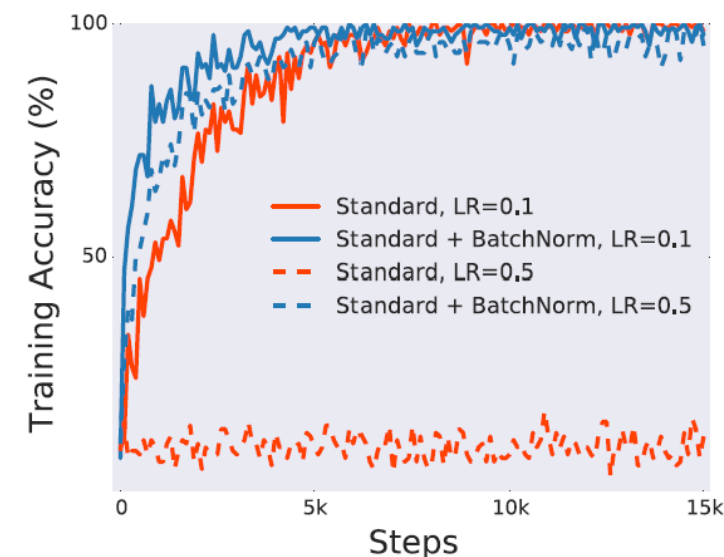
Andrew Ilyas*
MIT
ailyas@mit.edu

Aleksander Mądry
MIT
madry@mit.edu

Abstract

Batch Normalization (BatchNorm) is a widely adopted technique that enables faster and more stable training of deep neural networks (DNNs). Despite its pervasiveness, the exact reasons for BatchNorm's effectiveness are still poorly understood. The popular belief is that this effectiveness stems from controlling the change of the layers' input distributions during training to reduce the so-called "internal covariate shift". In this work, we demonstrate that such distributional stability of layer inputs has little to do with the success of BatchNorm. Instead, we uncover a more fundamental impact of BatchNorm on the training process: it makes the optimization landscape significantly smoother. This smoothness induces a more predictive and stable behavior of the gradients, allowing for faster training.

- Permite convergencias más rápidas:



Santurkar, S., Tsipras, D., Ilyas, A., & Madry, A. (2018). How does batch normalization help optimization?. In *Advances in Neural Information Processing Systems* (pp. 2488-2498). <https://arxiv.org/abs/1805.11604>



Redes Profundas

- ☐ Deep feedforward network
- ☐ **Convolutional neural network**
- ☐ Recurrent neural network
- ☐ Autoencoders

CNN – Convolutional Neural Network

Aplicaciones:

- ❑ Clasificación
- ❑ Detección de objetivos
- ❑ Segmentación de objetos
- ❑ Reconocimiento fácil
- ❑ Síntesis de imágenes



Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 779-788).

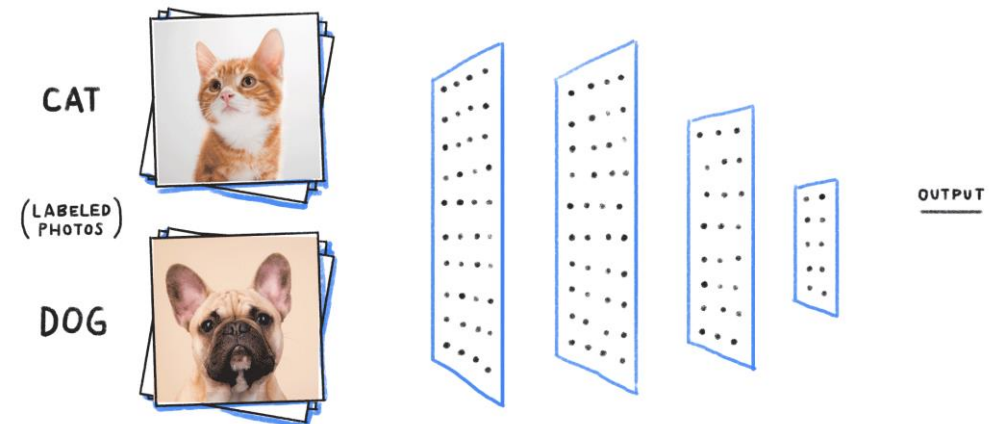


Figure 2. Mask R-CNN results on the COCO test set. These results are based on ResNet-101 [15], achieving a *mask* AP of 35.7 and running at 5 fps. Masks are shown in color, and bounding box, category, and confidences are also shown.

Por qué la clasificación de imágenes es tan difícil?

Perspectiva



Escala



Deformación



Oclusión



Iluminación



Fondo

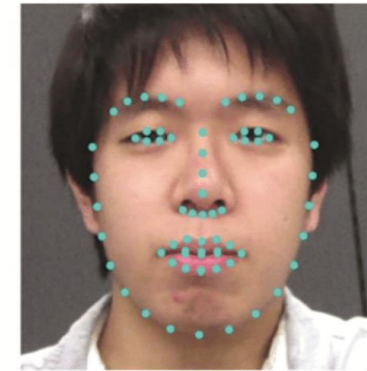
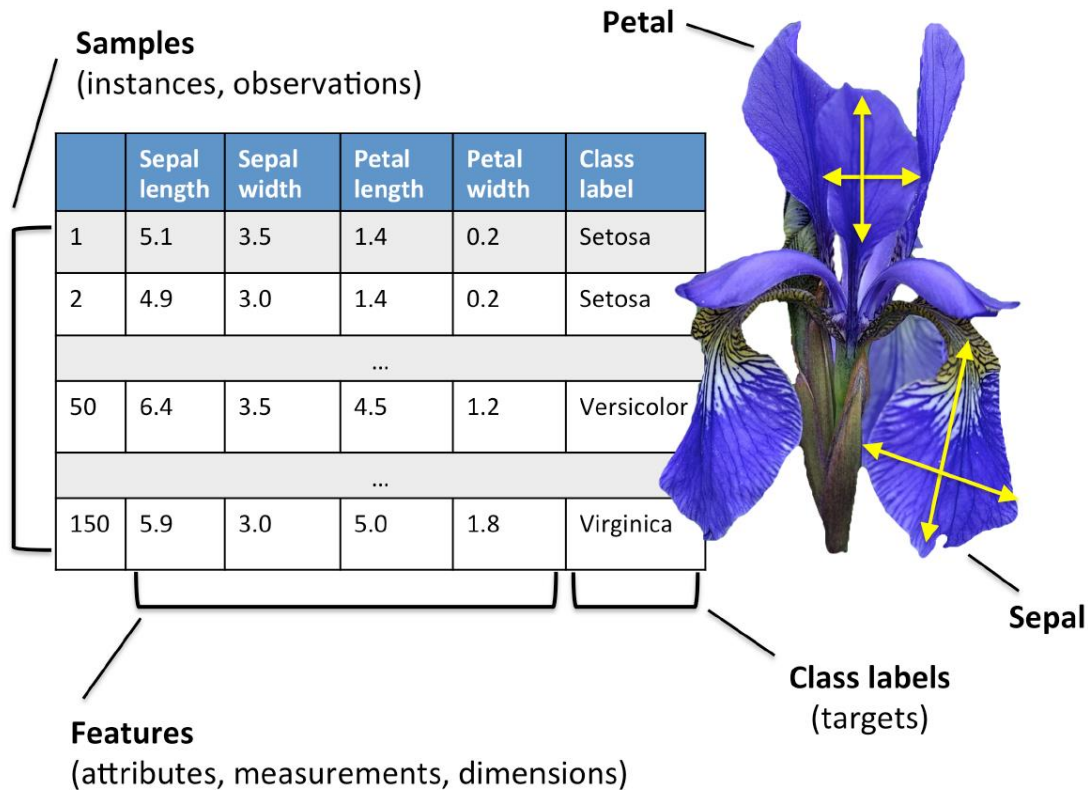


Variabilidad



Enfoques tradicionales

❑ Extracción de características manual



(a) Detected facial keypoints

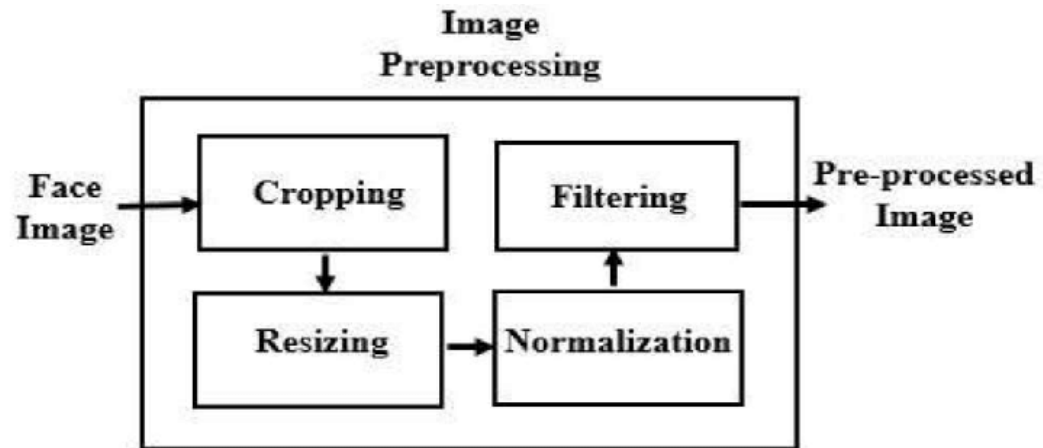
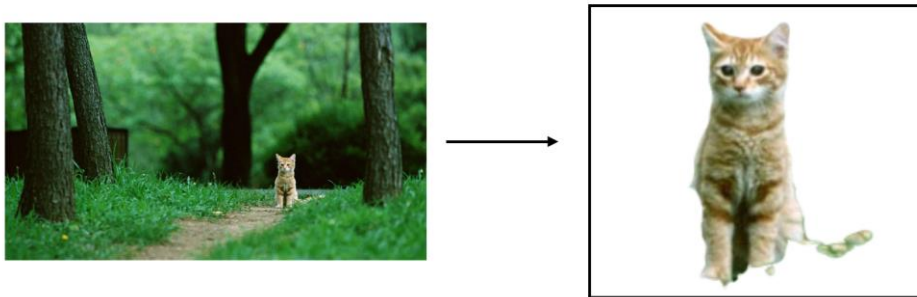


(b) Facial organ keypoints

Sasaki, K., Hashimoto, M., & Nagata, N. (2016). Person Invariant Classification of Subtle Facial Expressions Using Coded Movement Direction of Keypoints. In *Video Analytics. Face and Facial Expression Recognition and Audience Measurement* (pp. 61-72). Springer, Cham.

Enfoques tradicionales

❑ Preprocesamiento de imágenes



CNN – Convolutional Neural Network

Backpropagation Applied to Handwritten Zip Code Recognition

543

80322-4129 80206

40004 14310

37879 05153

33502 75216

35460 44209

1011915485726803226414186
4359720299299722310046701
3084114591010615406103631
106411103047526200979966
8912086708557101427955460
2018730187112993089970984
0109707597331972015519035
1075318255182814358090943
1787521655460354603546055
18255108503047520439401

548

LeCun, Boser, Denker, Henderson, Howard, Hubbard, and Jackel

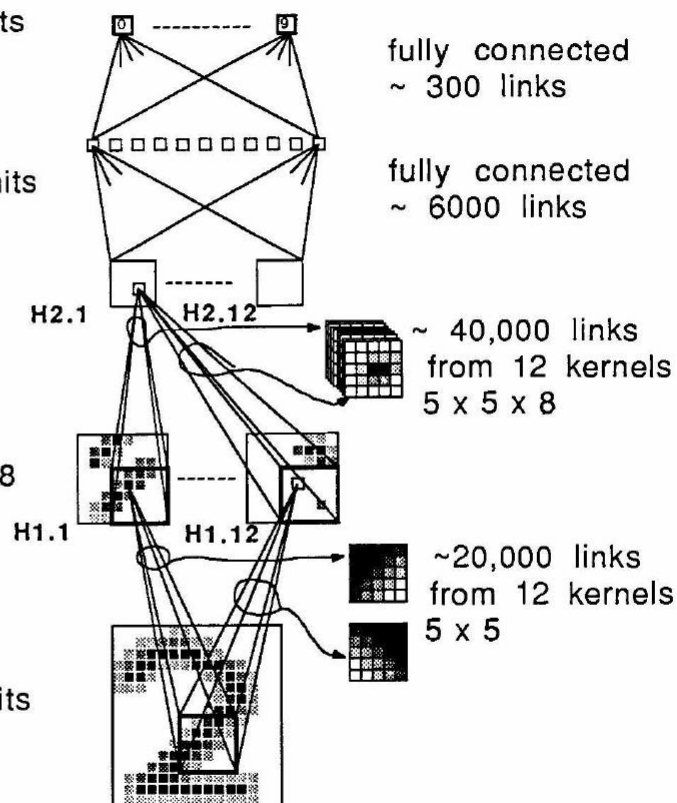
10 output units

layer H3
30 hidden units

layer H2
12 x 16 = 192
hidden units

layer H1
12 x 64 = 768
hidden units

256 input units



Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard and L. D. Jackel: [Backpropagation Applied to Handwritten Zip Code Recognition](#), Neural Computation, 1(4):541-551, Winter 1989.

CNN – Convolutional Neural Network

PROC. OF THE IEEE, NOVEMBER 1998

7

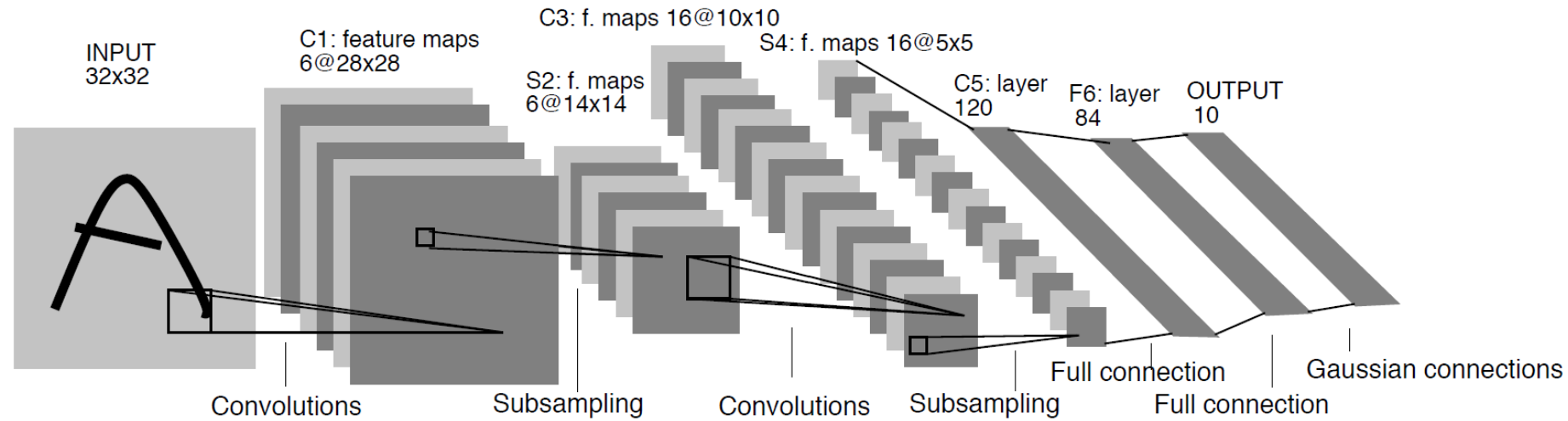


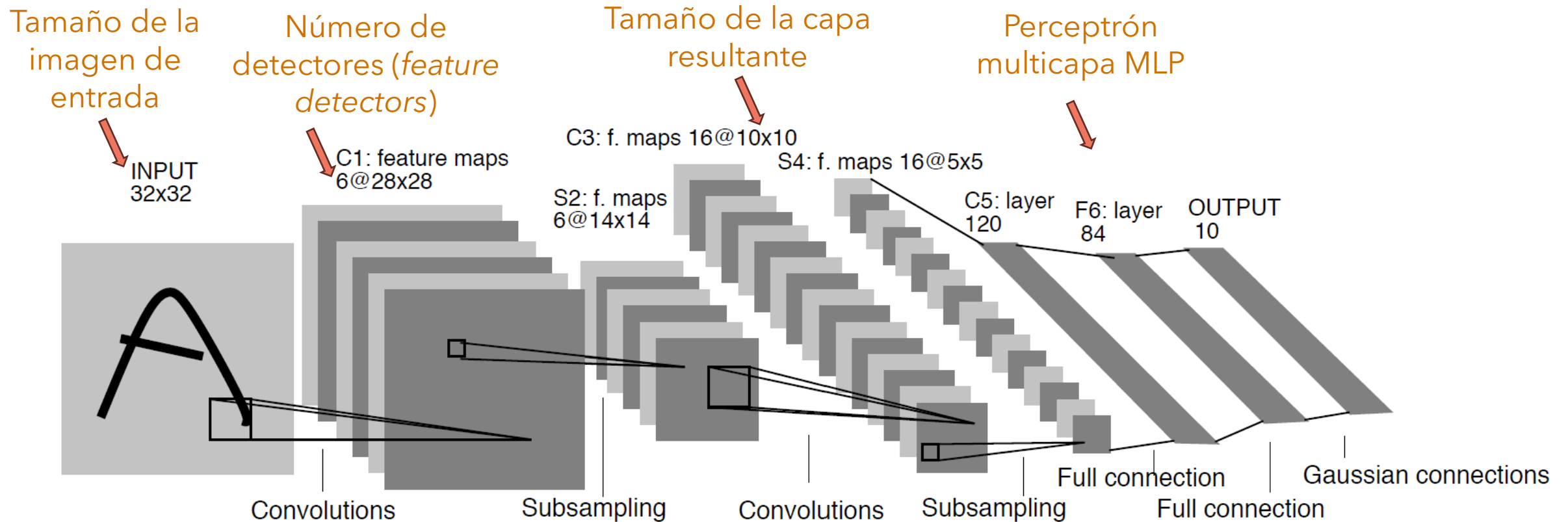
Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

Extracción de características
automático

Clasificador tradicional

Yann LeCun, Léon Bottou, Yoshua Bengio and Patrick Haffner: [Gradient Based Learning Applied to Document Recognition](#), Proceedings of IEEE, 86(11):2278–2324, 1998.

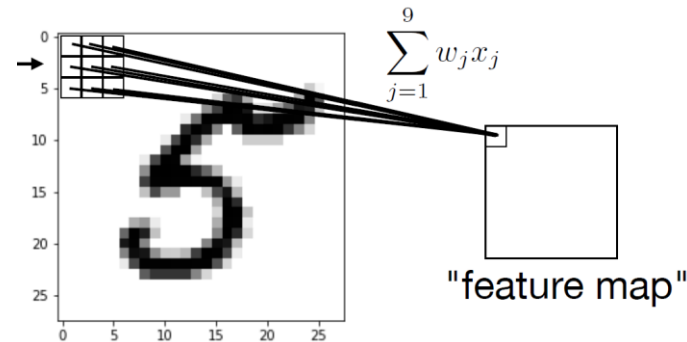
CNN – Convolutional Neural Network



- ❑ **Convolución:** aplicación de los detectores (matrices de pesos) también denominados filtros o kernels
- ❑ **Submuestreo - subsampling:** hoy en día denominado "pooling"
- ❑ **Conección Gausiana:** una capa completamente conectada con pérdida MSE

CNN – Convolutional Neural Network

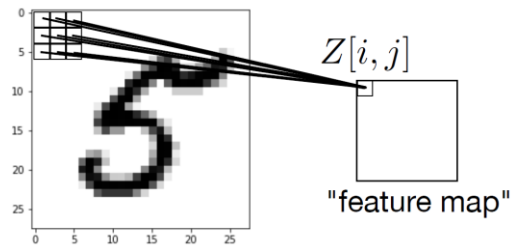
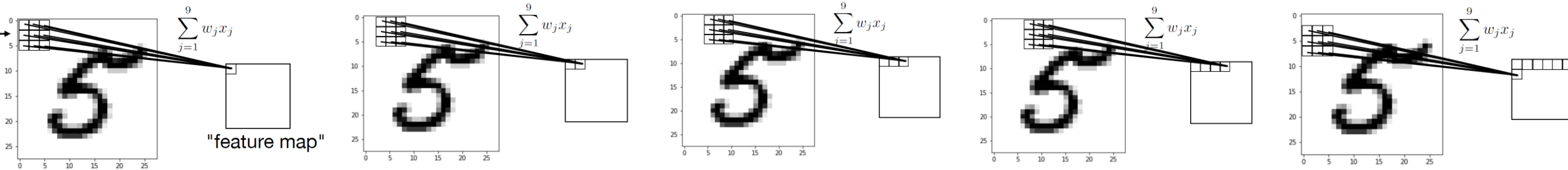
- ❑ **Conectividad esparcida:** un elemento del mapa de características está conectado a una pequeña porción de píxeles. Esto es muy diferente a conectar toda la capa de entrada como en el MLP
- ❑ **Parámetros compartidos:** los mismos pesos son usados por diferentes segmentos de la imagen de entrada
 - Un detector de características es un filtro que pasa por encima de la entrada generando un mapa de características



- ❑ **Múltiples capas:** combinan patrones locales con patrones globales de la imagen

Convolución

❑ Extracción de características espaciales



Cross-correlación:

$$Z[i,j] = \sum_{u=-k}^k \sum_{v=-k}^k K[u,v]A[i+u,j+v] = K \otimes A$$

Usada en DL por su fácil implementación

Convolución:

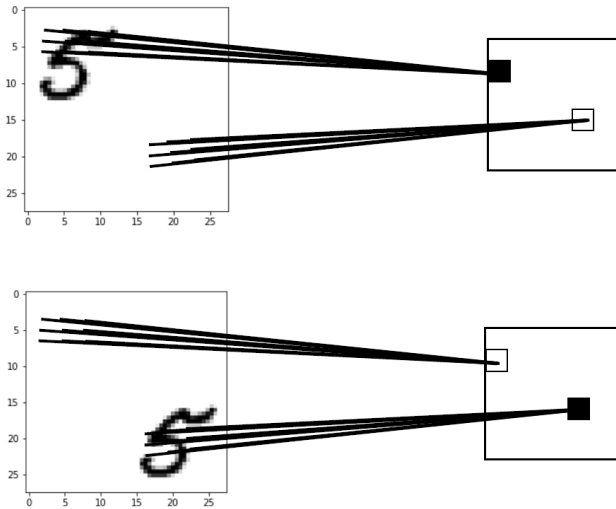
$$Z[i,j] = \sum_{u=-k}^k \sum_{v=-k}^k K[u,v]A[i-u,j-v] = K * A$$

Se aprenden durante el entrenamiento

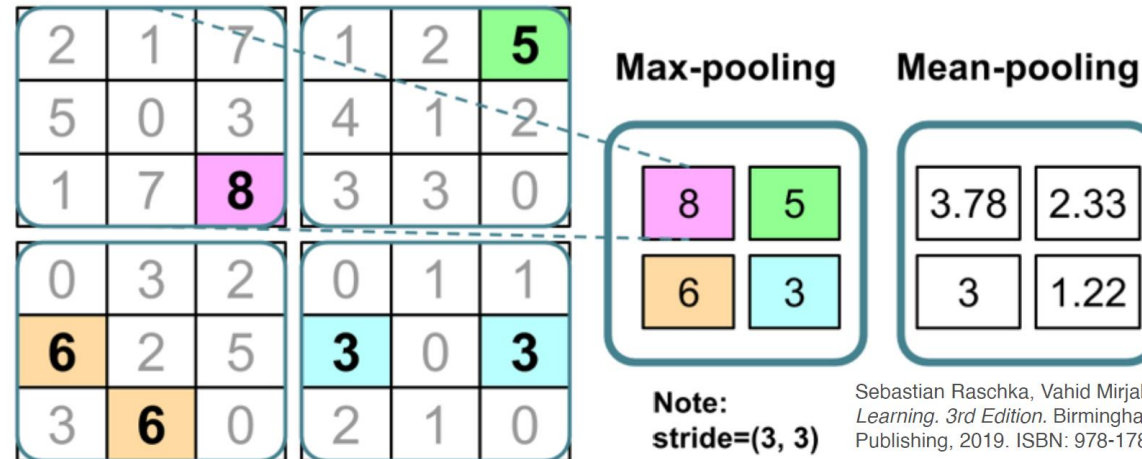
1) -1,-1	2) -1,0	3) -1,1
4) 0,-1	5) 0,0	6) 0,1
7) 1,-1	8) 1,0	9) 1,1

Invarianza a la traslación, rotación y escala

- ❑ La extracción de características espaciales usando la correlación no es invariante a la escala, rotación y traslación



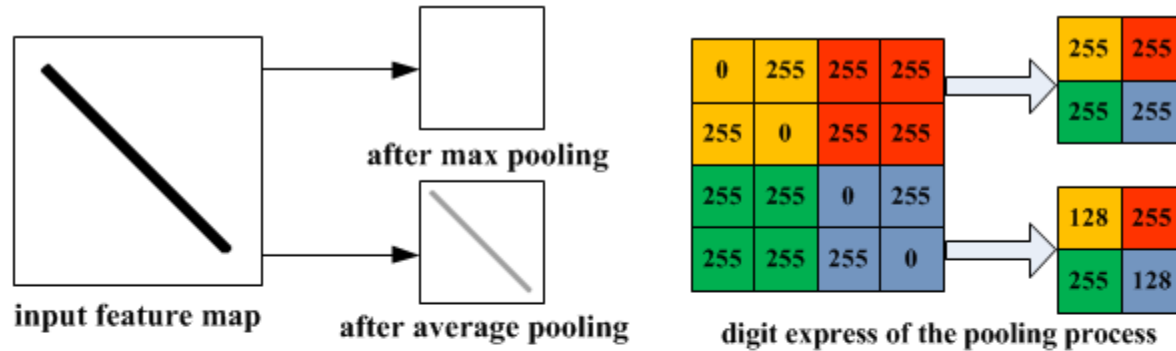
- ❑ Para ayudar con la invarianza local se incluyen las capas de "pooling", inicialmente denominadas de sub-muestreo



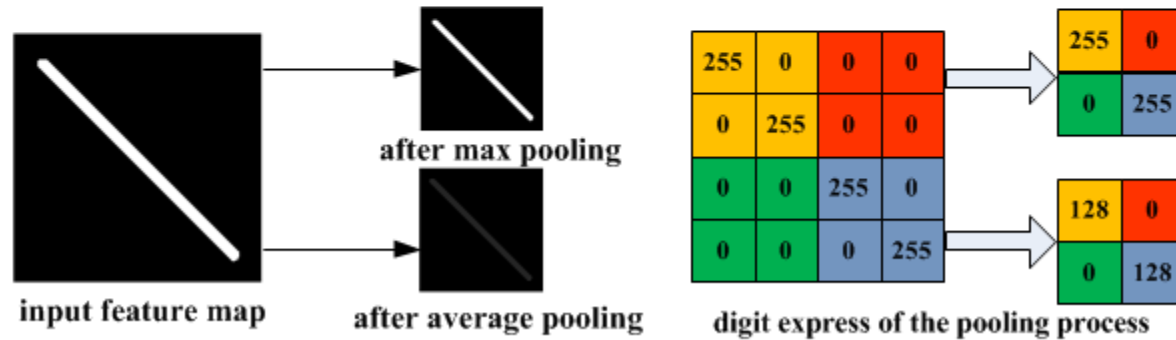
Sebastian Raschka, Vahid Mirjalili. *Python Machine Learning, 3rd Edition*. Birmingham, UK: Packt Publishing, 2019. ISBN: 978-1789955750

- ❑ Esta operación reduce el tamaño de la capa, lo que implica pérdida de información
- ❑ Típicamente, esta capa no tiene ningún parámetro que requiera ser entrenado

Pooling



(a) Illustration of max pooling drawback

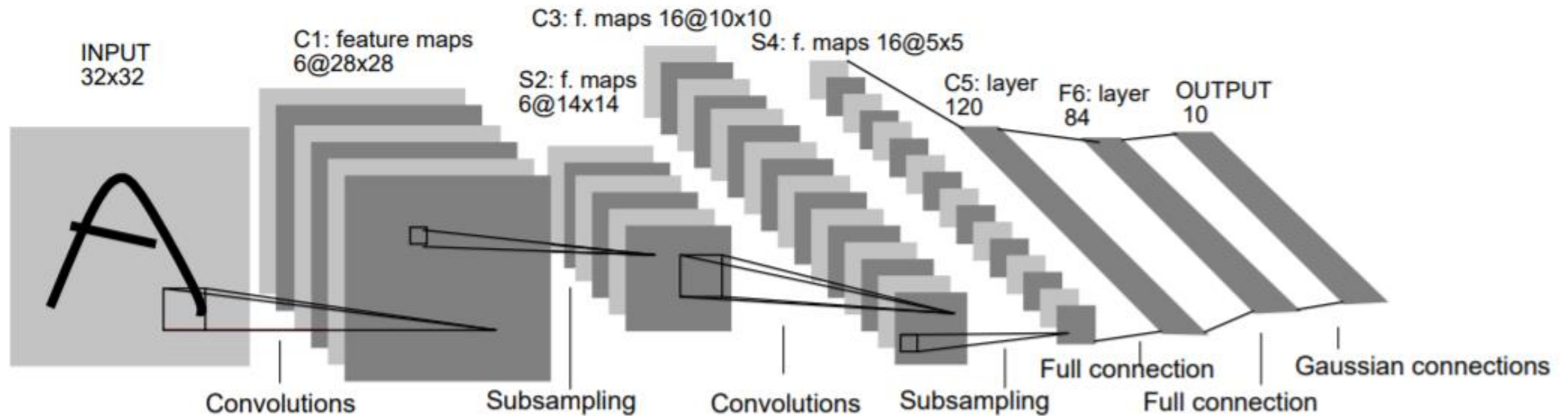


(b) Illustration of average pooling drawback

- ❑ Si la mayoría de los elementos son de magnitudes elevadas, el rasgo distintivo desaparece en el max pooling

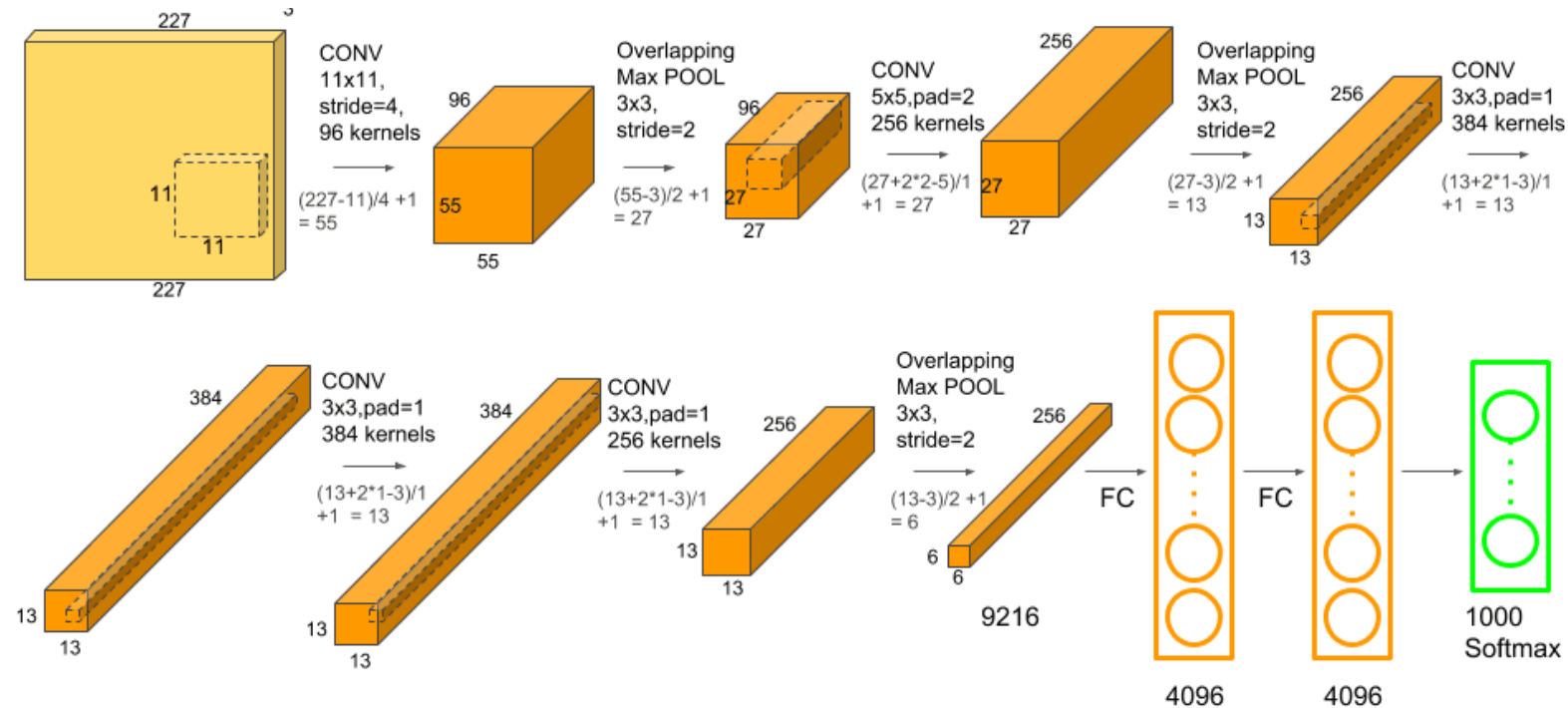
- ❑ Por su parte, el mean pooling tiene en cuenta todas las magnitudes bajas, reduciendo el contraste. Incluso, si hay muchos elementos en cero, la característica del mapa se reducirá en gran medida

LeNet

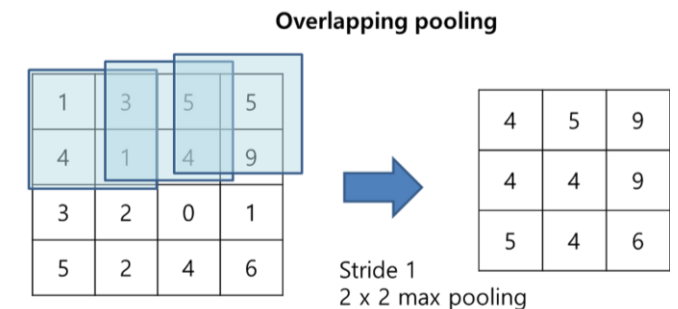


- ❑ 1998: Primera aplicación exitosa de una CNN. Desarrollada por Yann LeCun, Corinna Cortes, y Christopher Burges
- ❑ 5 capas alternando convolución y pooling
- ❑ Usa las funciones de activación tanh/sigmoid
- ❑ Se aplicó al reconocimiento de caracteres

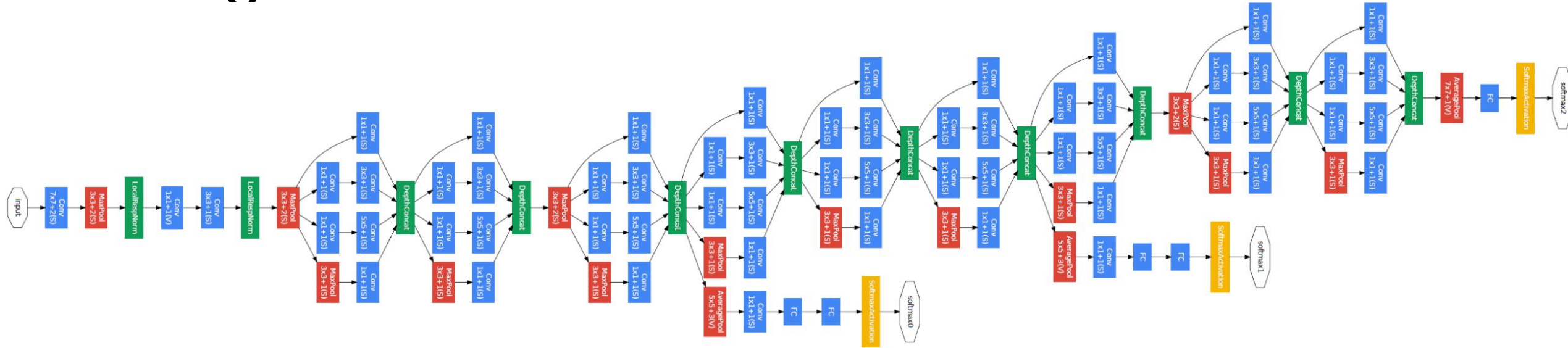
AlexNet



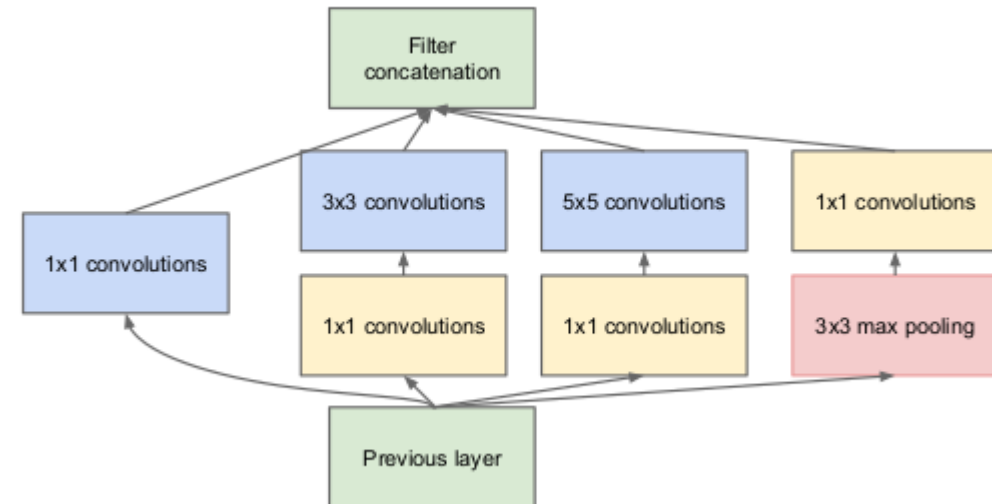
- ❑ 2012: Red más profunda que LeNet. Desarrollada por Alex Krizhevsky, Ilya Sutskever, t Geoff Hinton.
- ❑ Usa la función de activación ReLU
- ❑ Implementa el dropout
- ❑ Emplea GPU para su entrenamiento
- ❑ Usada en el reconocimiento de imágenes a gran escala
- ❑ Número de parámetros aprox.: 60 millones



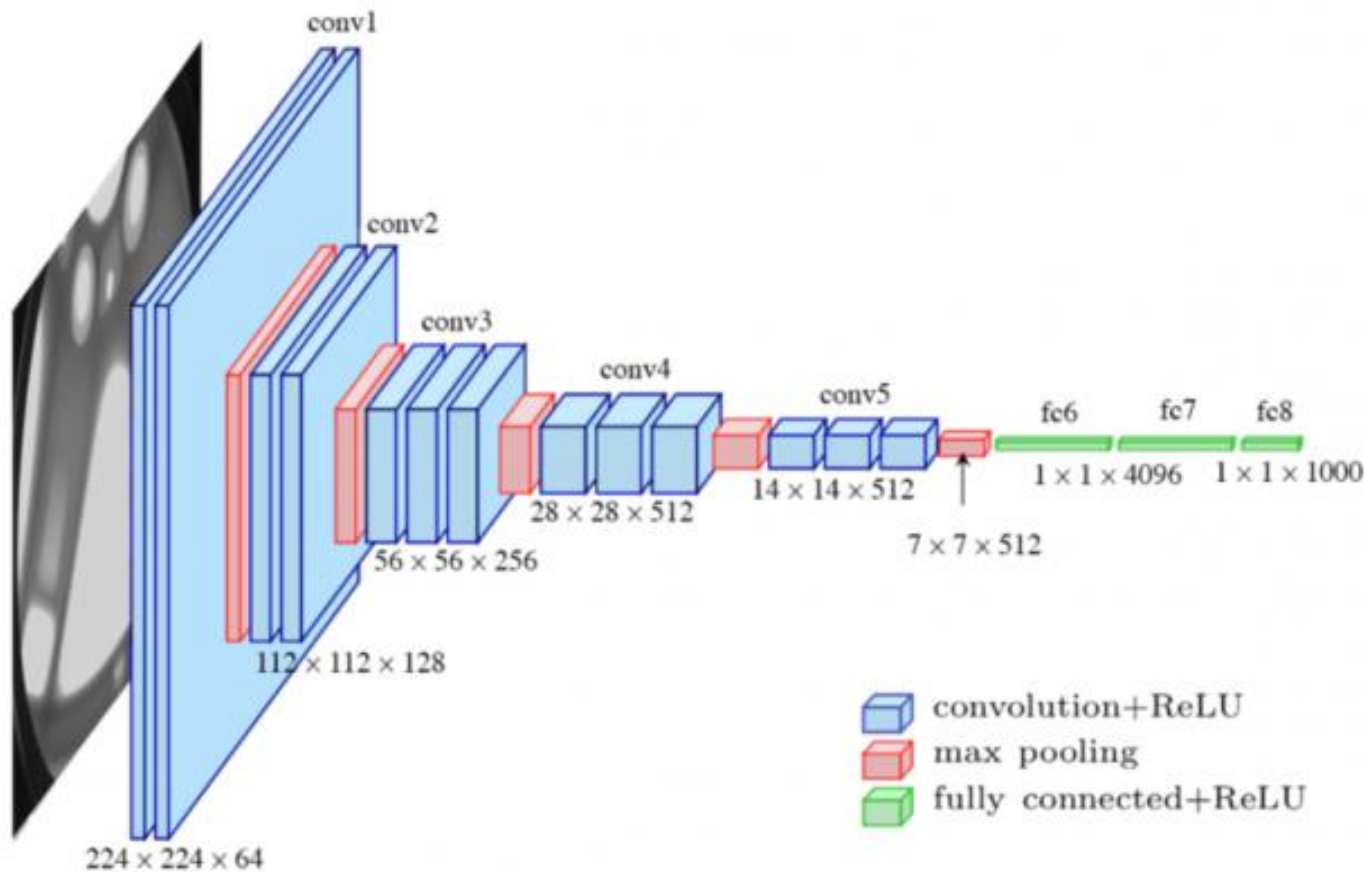
GoogLeNet



- ❑ 2014: Desarrollada en Google por Jeff Dean, Christian Szegedy, Alexandro Szegedy et al.
- ❑ 22 Capas
- ❑ Utiliza capas "Inception"
- ❑ Se utiliza para clasificación de imágenes y detección de objetos

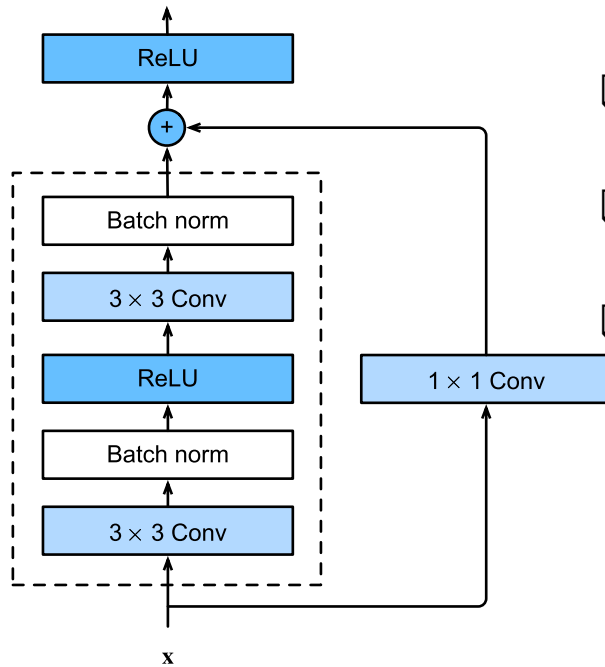
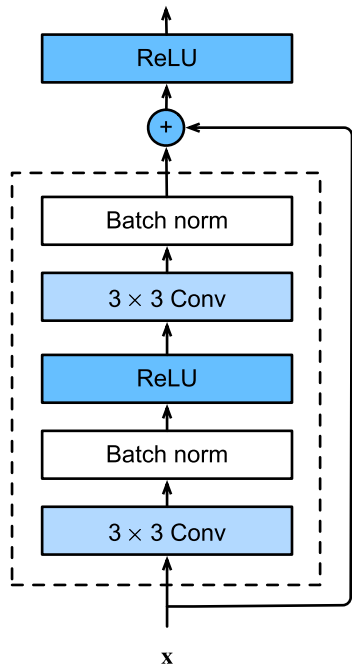


VGGNet



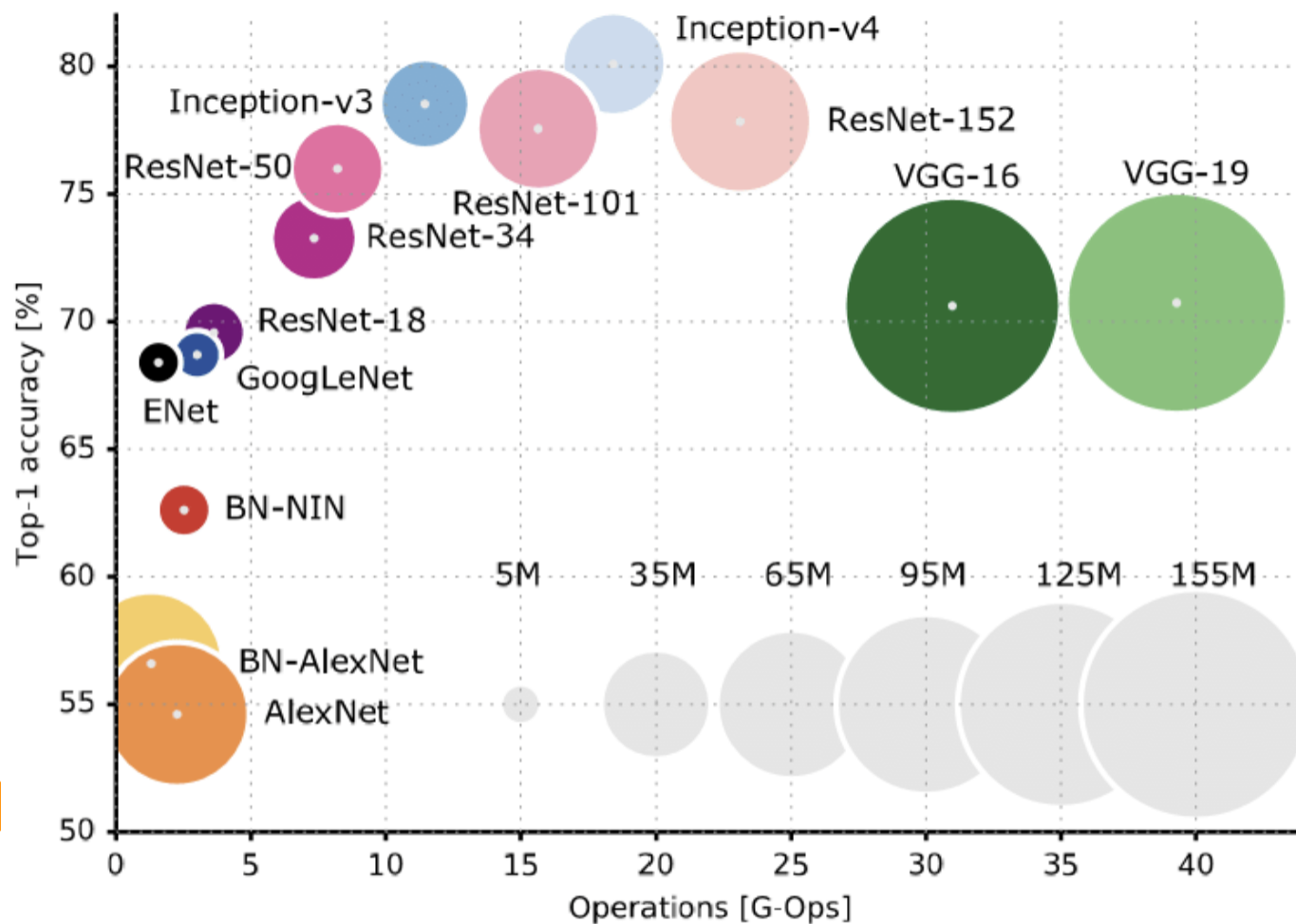
- ❑ Desarrollada por Karen Simonyan, Andrew Zisserman, et. al. en la Universidad de Oxford.
- ❑ VGG16 significa «Visual Geometry Group 16». Este nombre procede del Grupo de Geometría Visual de la Universidad
- ❑ El «16» del nombre indica que el modelo contiene 16 capas: incluye capas convolucionales y capas totalmente conectadas.
- ❑ VGGNet es una CNN de 16 capas con hasta 95 millones de parámetros y entrenada con más de mil millones de imágenes (1000 clases).

ResNet



- ❑ Desarrollada por Kaiming He et al. para ganar el concurso de clasificación ILSVRC 2015
- ❑ La red tiene 152 capas y más de un millón de parámetros
- ❑ Tomó más de 40 días en 32 GPU entrenar la red en el conjunto de datos ILSVRC 2015
- ❑ Las CNN se utilizan principalmente para tareas de clasificación de imágenes con 1000 clases, pero también pueden emplearse con éxito para resolver problemas de procesamiento del lenguaje natural como completar frases o comprensión automática (demostrado por el equipo de Microsoft Research Asia en 2016 y 2017).

CNN



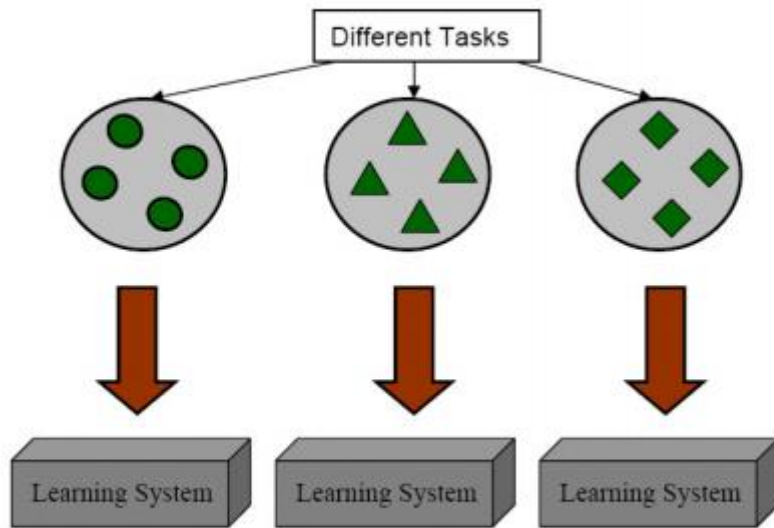
Comparison of popular CNN architectures. The vertical axis shows top 1 accuracy on ImageNet classification. The horizontal axis shows the number of operations needed to classify an image. Circle size is proportional to the number of parameters in the network.

¿Como elegir la arquitectura?

- ☐ Si los datos de entrada son pequeños y sencillos, como imágenes de baja resolución, puede bastar con una arquitectura CNN más pequeña, como LeNet o AlexNet.
- ☐ Si los datos de entrada son grandes y complejos, como imágenes o vídeos de alta resolución, puede ser necesaria una arquitectura CNN más grande y compleja, como VGG, Inception o ResNet.
- ☐ Si la tarea implica la detección o segmentación de objetos, pueden ser adecuadas arquitecturas como YOLO, RCNN o Mask R-CNN.

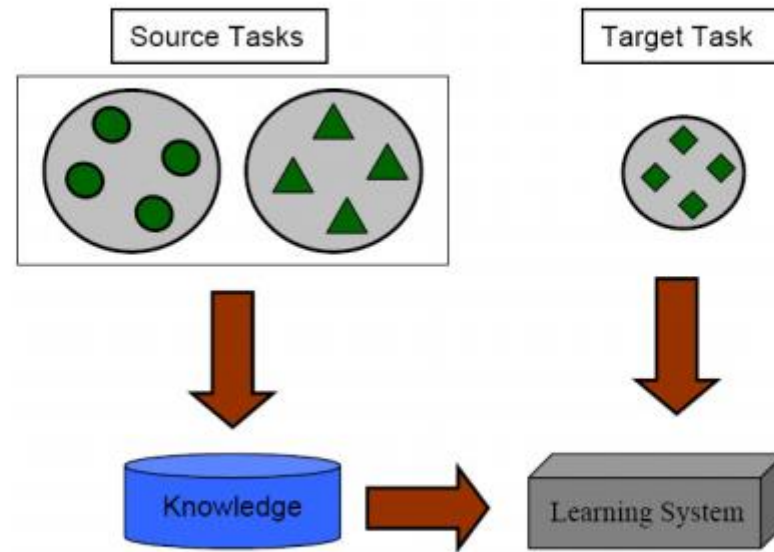
Transfer Learning

Learning Process of Traditional Machine Learning



(a) Traditional Machine Learning

Learning Process of Transfer Learning



(b) Transfer Learning

❑ Busca transferir el conocimiento de un modelo hacia una nueva tarea: “adaptación de dominio”

❑ Motivación

- Entrenar una red requiere muchos datos, tiempo y recursos
 - ImageNet puede tomar semanas para entrenar desde cero
- Es una forma rápida y económica de adaptar una red explotando sus propiedades de generalización

Transfer Learning

Tipos

- ❑ **Inductivo:** adapta un modelo entrenado supervisado en un nuevo conjunto de datos etiquetado (clasificación, regresión)
- ❑ **Transductivo:** adapta un modelo entrenado supervisado en un conjunto de datos no etiquetado (clasificación, regresión)
- ❑ **No supervisado:** adapta un modelo entrenado no supervisado en un nuevo conjunto de datos no etiquetado (clustering, reducción de dimensiones)

Aplicaciones

- ❑ Clasificación de imágenes (más común): aprender nuevas clases
- ❑ Clasificación de sentimientos en texto
- ❑ Traducción de textos a nuevos lenguajes
- ❑ Respuesta a preguntas

Proceso

- Inicie con la red pre-entrenada
- Seleccione
 - Las capas que desea mantener (usualmente la extracción de características)
 - Las capas que desea reemplazar (usualmente el clasificador)
- Entrene las capas seleccionadas
- Sintonice toda red (con los pesos congelados)