

Transferencia de aprendizaje

Visión por computador II

Juan Carlos Arbeláez

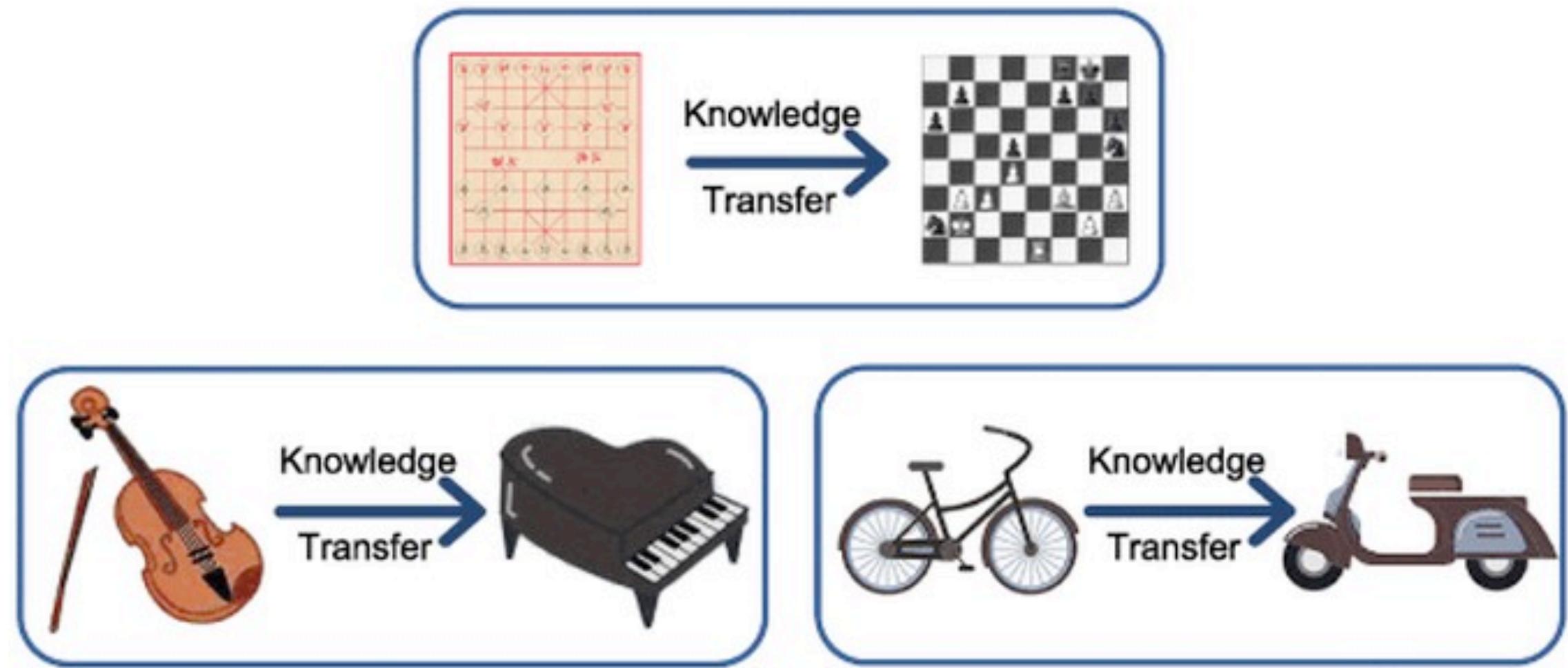
Contenido

- A. Transferencia de aprendizaje
- B. Recap Neuronal Network y Convoluciones
- C. Modelos pre-entrenados:
 - VGG16
 - MobileNet
 - ResNet

Transfer Learning

Transferencia de aprendizaje

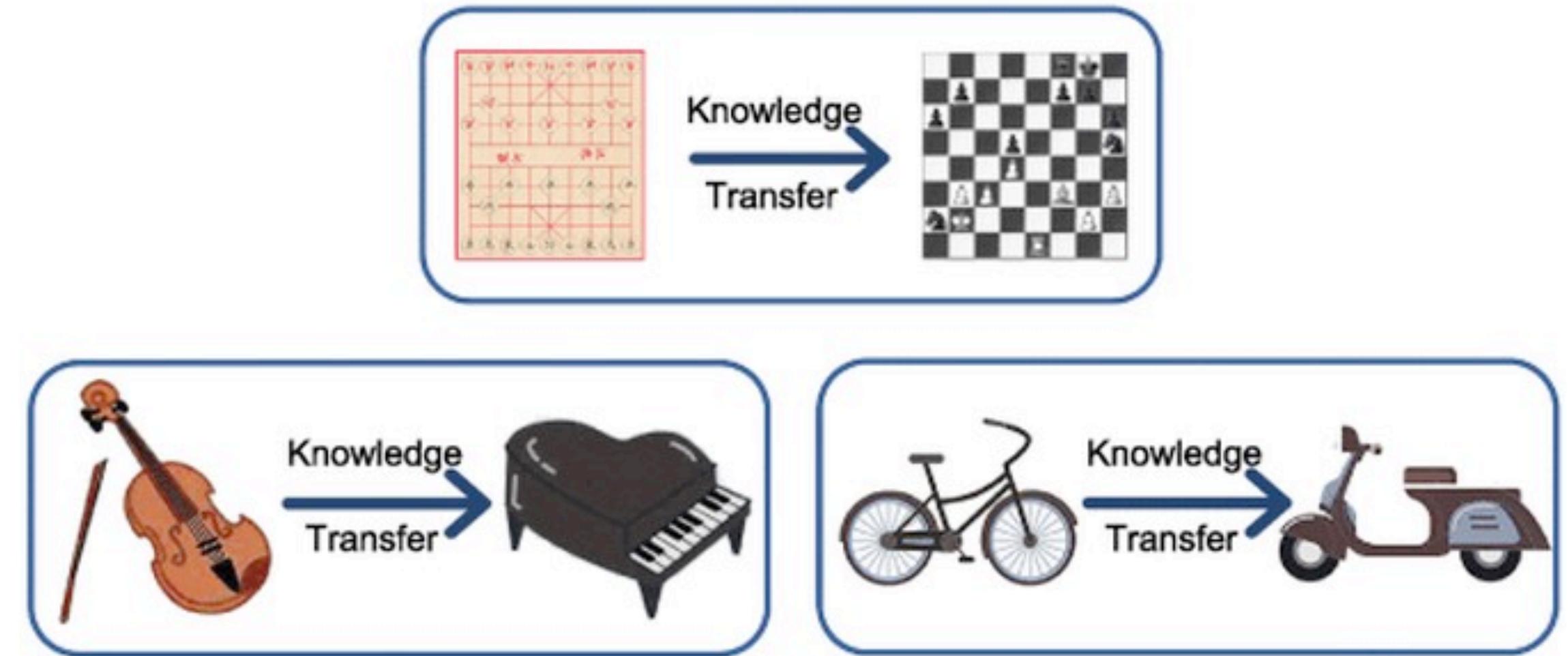
- Los seres humanos tenemos una capacidad inherente para transferir conocimientos entre tareas.
- Lo que adquirimos como conocimiento mientras aprendemos sobre una tarea, lo utilizamos del mismo modo para resolver tareas relacionadas.
- Cuanto más relacionadas estén las tareas, más fácil nos resultará transferir el conocimiento



2021 - A Comprehensive Survey on Transfer Learning

Introducción

- En cada uno de los escenarios anteriores, no aprendemos todo desde cero cuando intentamos aprender nuevos aspectos o temas.
- Transferimos y aprovechamos nuestros conocimientos de lo que hemos aprendido en el pasado.

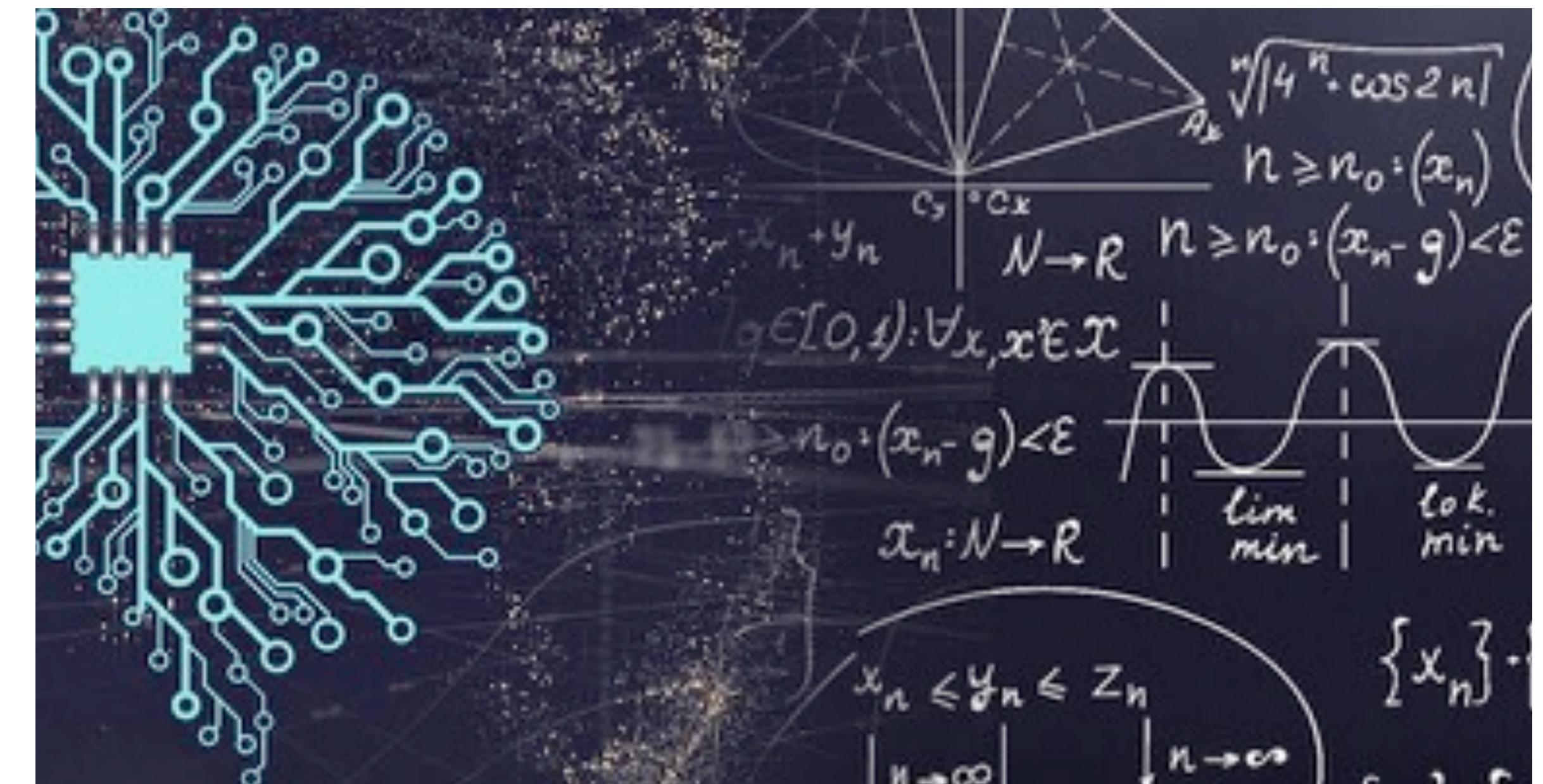


2021 - A Comprehensive Survey on Transfer Learning

Introducción

- Los algoritmos convencionales de ML, hasta ahora, los hemos diseñado para trabajar de forma aislada
- Se entrena para resolver tareas específicas
- Los modelos tienen que reconstruirse desde cero cuando cambia la distribución del espacio de características o la tarea

El aprendizaje por transferencia es la idea de superar el paradigma del aprendizaje aislado y utilizar los conocimientos adquiridos para una tarea para resolver otras relacionadas



Cada vez que aprendemos algo no hacemos un “*inicialización aleatoria*”

Motivación

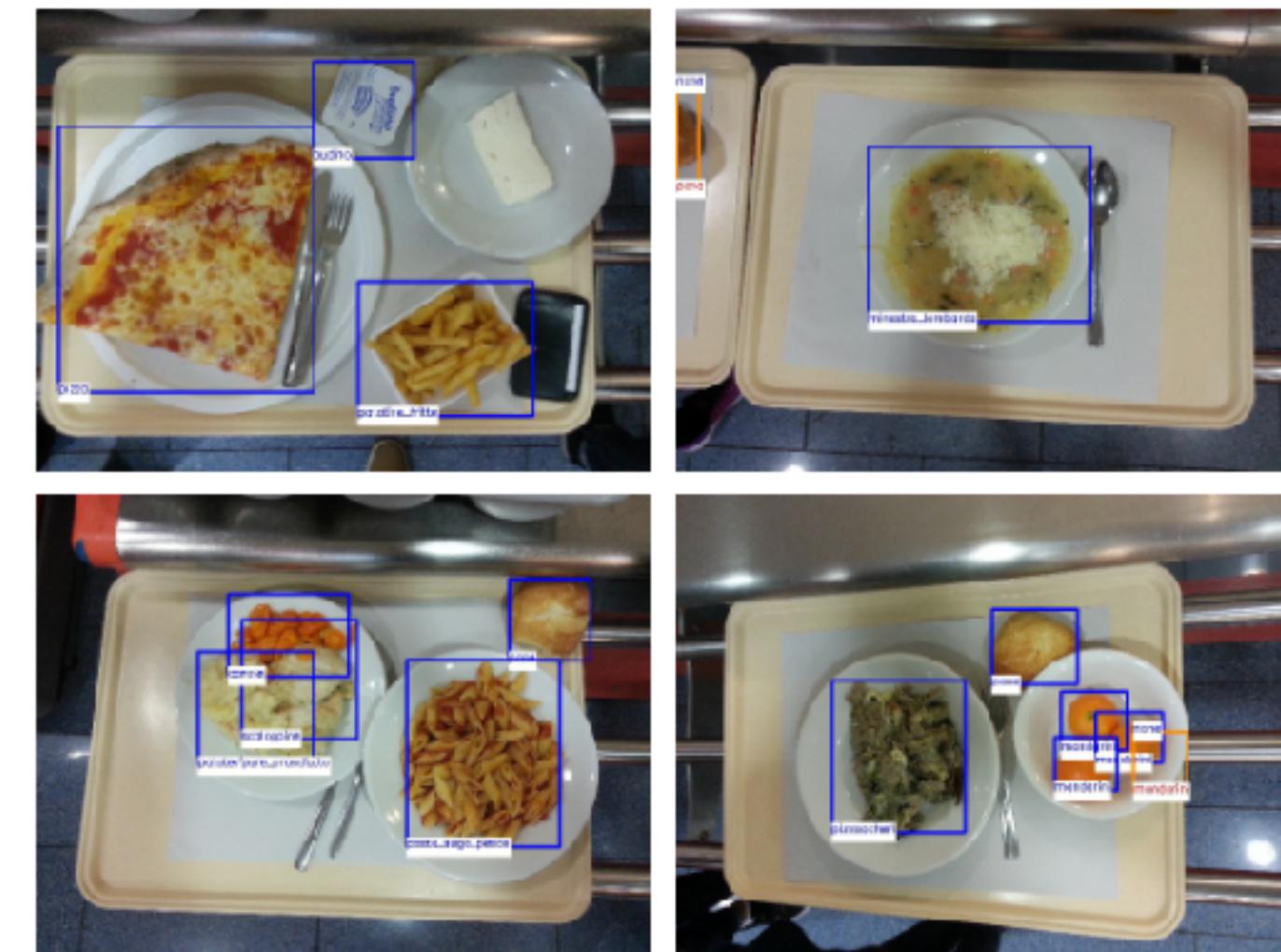
- La mayoría de los modelos que resuelven problemas complejos necesitan una gran cantidad de datos
- Obtener grandes cantidades de datos etiquetados para modelos supervisados puede ser difícil (tiempo y el esfuerzo que lleva etiquetar puntos de datos)
- Un ejemplo es ImageNet, que contiene millones de imágenes pertenecientes a diferentes categorías



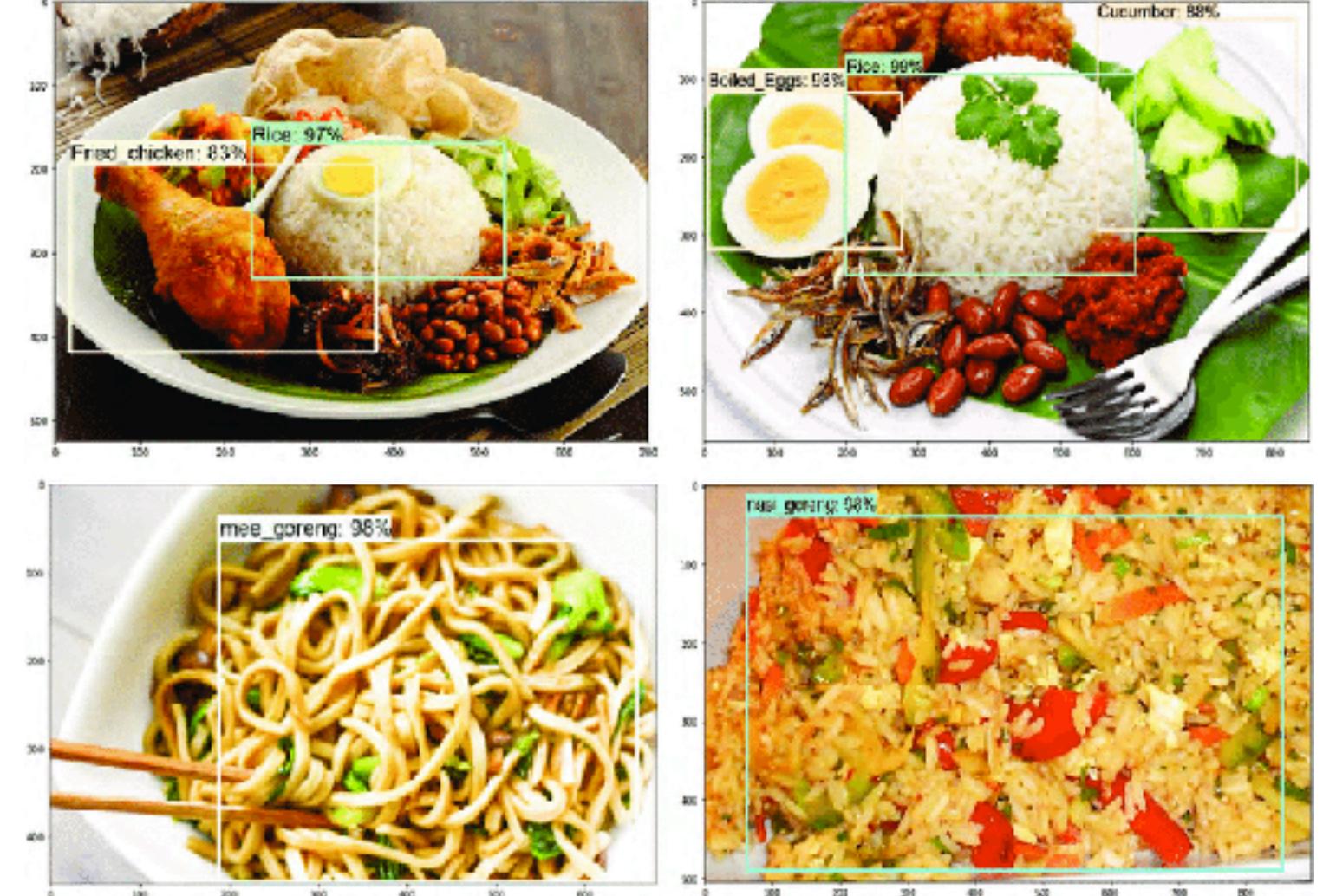
ImageNet: más de 100.000 categorías, cada categoría con ~1.000 imágenes

Motivación

- (Tarea T_1) tenemos que identificar comidas de un restaurante
- Entrenamos un modelo y lo ajustamos para que funcione bien (generalice) en el conjunto de pruebas
- (Tarea T_2) ahora debemos detectar objetos de una cafetería
- Idealmente, deberíamos poder aplicar el modelo entrenado para T_1 , pero en realidad, **nos enfrentamos a una degradación del rendimiento y a modelos que no generalizan bien**



Tarea T_2



Tarea T_1

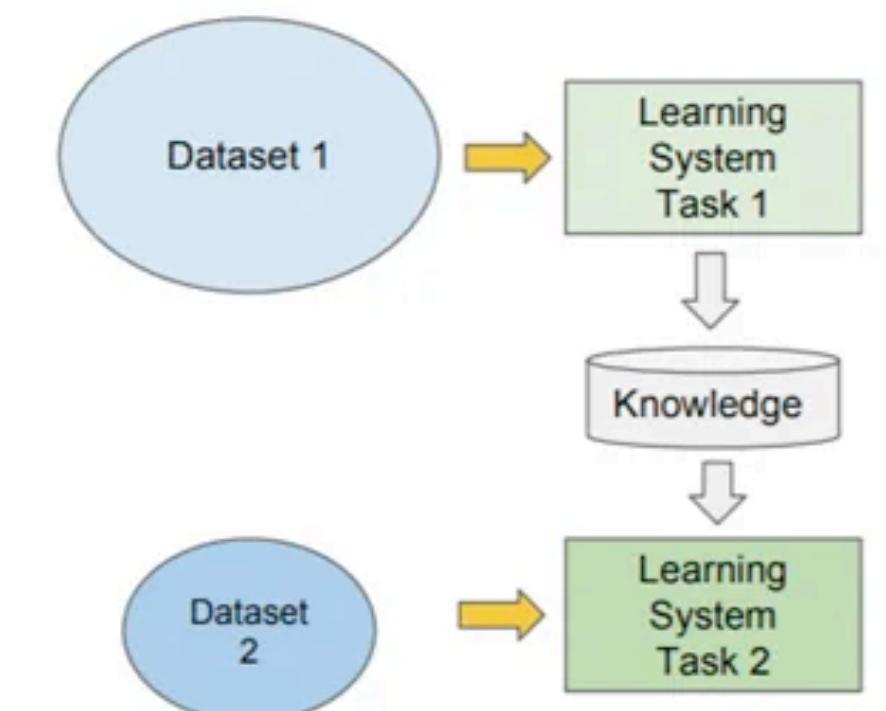
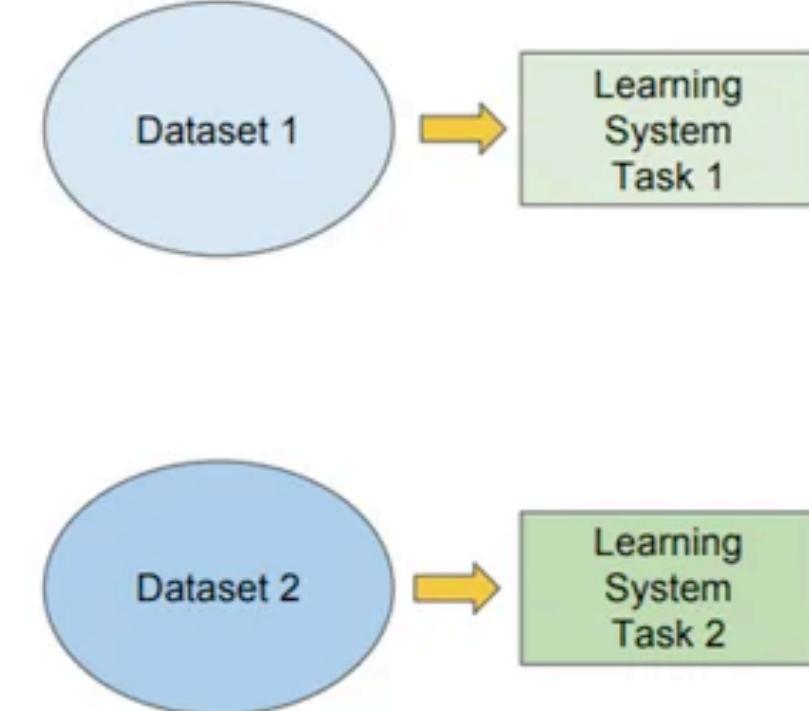
Motivación

- La transferencia de aprendizaje permite utilizar los conocimientos de tareas aprendidas anteriormente y aplicarlos a otras nuevas y relacionadas.
- Si tenemos muchos más datos para la tarea T_1 , podemos utilizar su aprendizaje y generalizar este conocimiento (características, pesos) para la tarea T_2 (que tiene muchos menos datos).

En visión: las características de bajo nivel, como los bordes, las formas, las esquinas y la intensidad, pueden compartirse entre tareas

Traditional ML vs Transfer Learning

- Isolated, single task learning:
 - Knowledge is not retained or accumulated. Learning is performed w.o. considering past learned knowledge in other tasks
- Learning of a new tasks relies on the previous learned tasks:
 - Learning process can be faster, more accurate and/or need less training data



Definición formal

Dominio D

- A. Espacio de características \mathcal{X}
- B. Distribución marginal $P(X), X = \{x_1, x_2 \dots x_n\} \in \mathcal{X}$

Un dominio son estos dos componentes $D = \{\mathcal{X}, P(X)\}$ definen un conjunto de datos

Tarea T

- C. Espacio de etiquetas \mathcal{Y}
- D. Función predictiva η , donde por cada característica predice una etiqueta $\eta(x_i) = y_i$

La tarea son dos componentes $\{\mathcal{Y}, \eta\}$

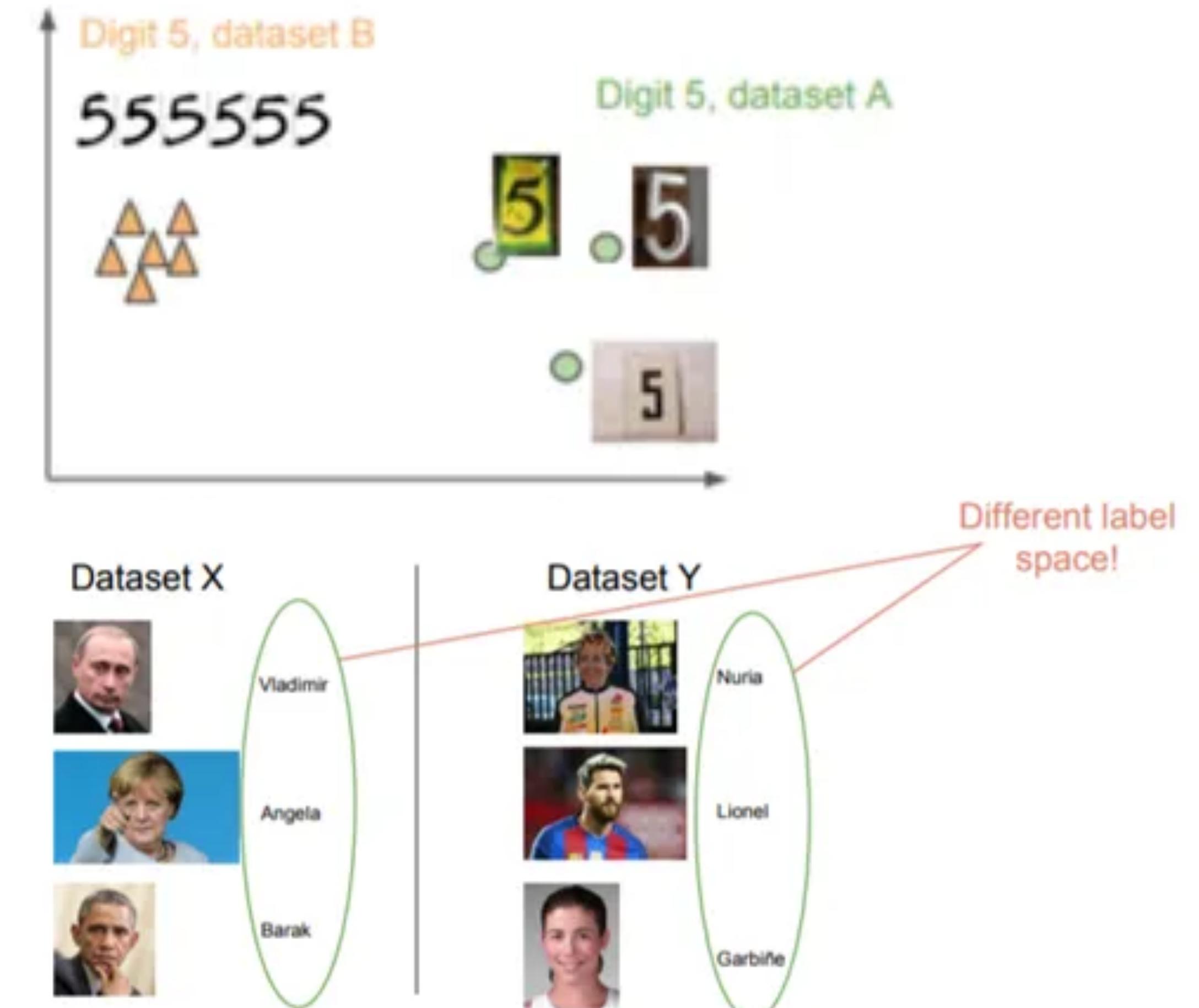
Definición formal

A. Dado un dominio fuente D_s con una tarea T_s (Source)

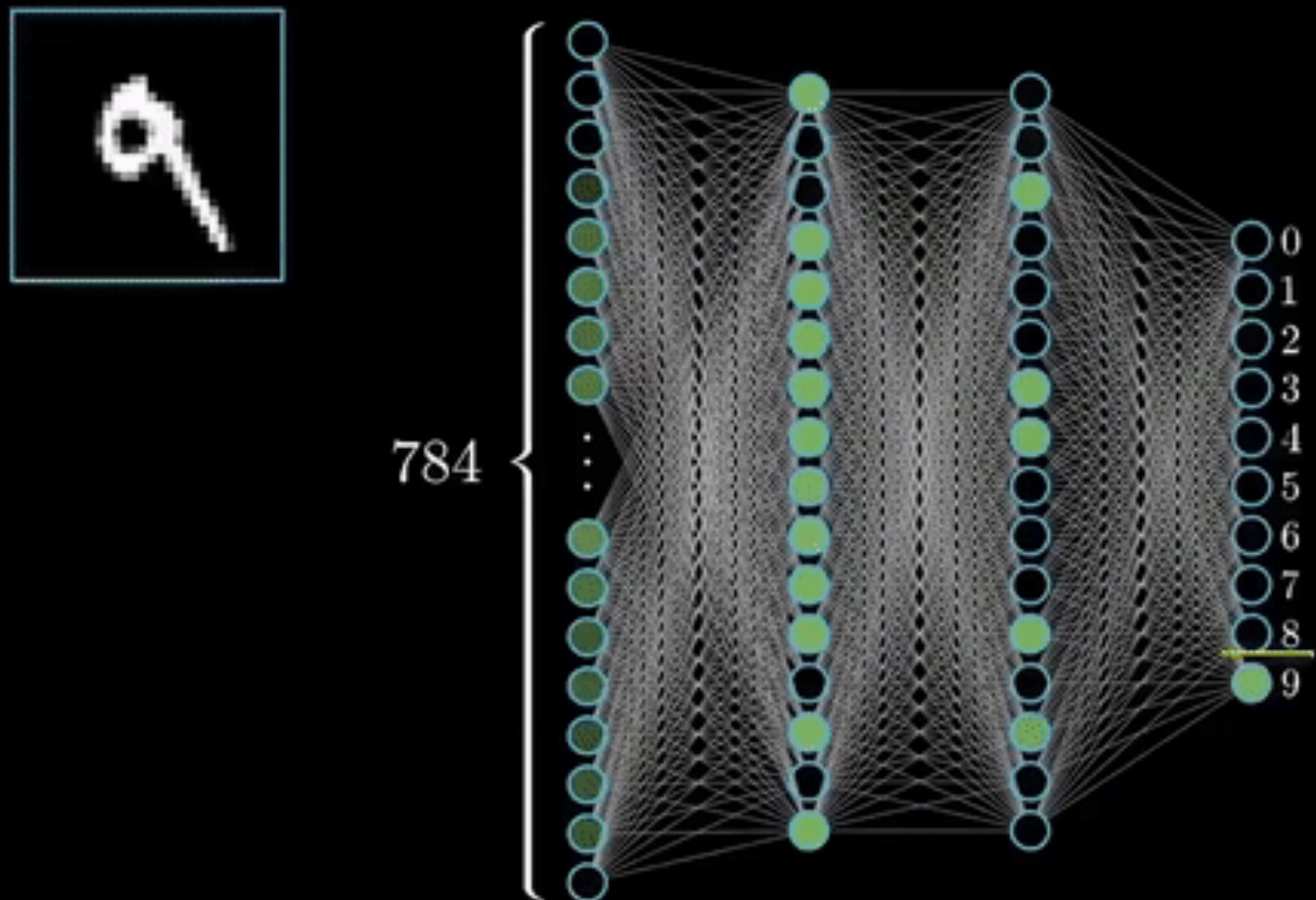
B. Un dominio objetivo D_T con una tarea objetivo T_T (Target)

El objetivo de la transferencia es:

- Aprender la función objetivo $\eta_T = P(Y_T | X_T)$ en el dominio D_T
- Con la información obtenida de D_S y T_S
- En donde $D_S \neq D_T$ y $T_S \neq T_T$



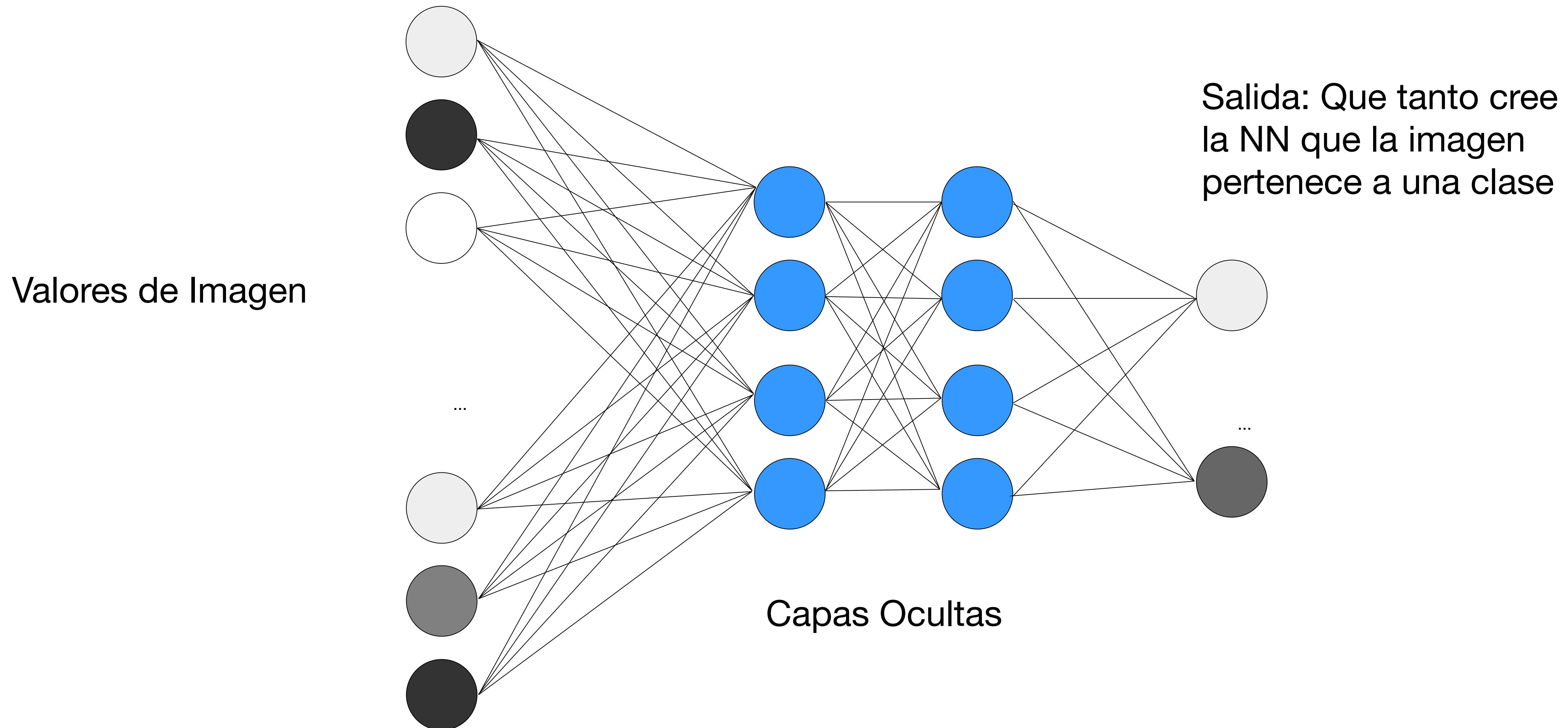
Recap Red Neuronal



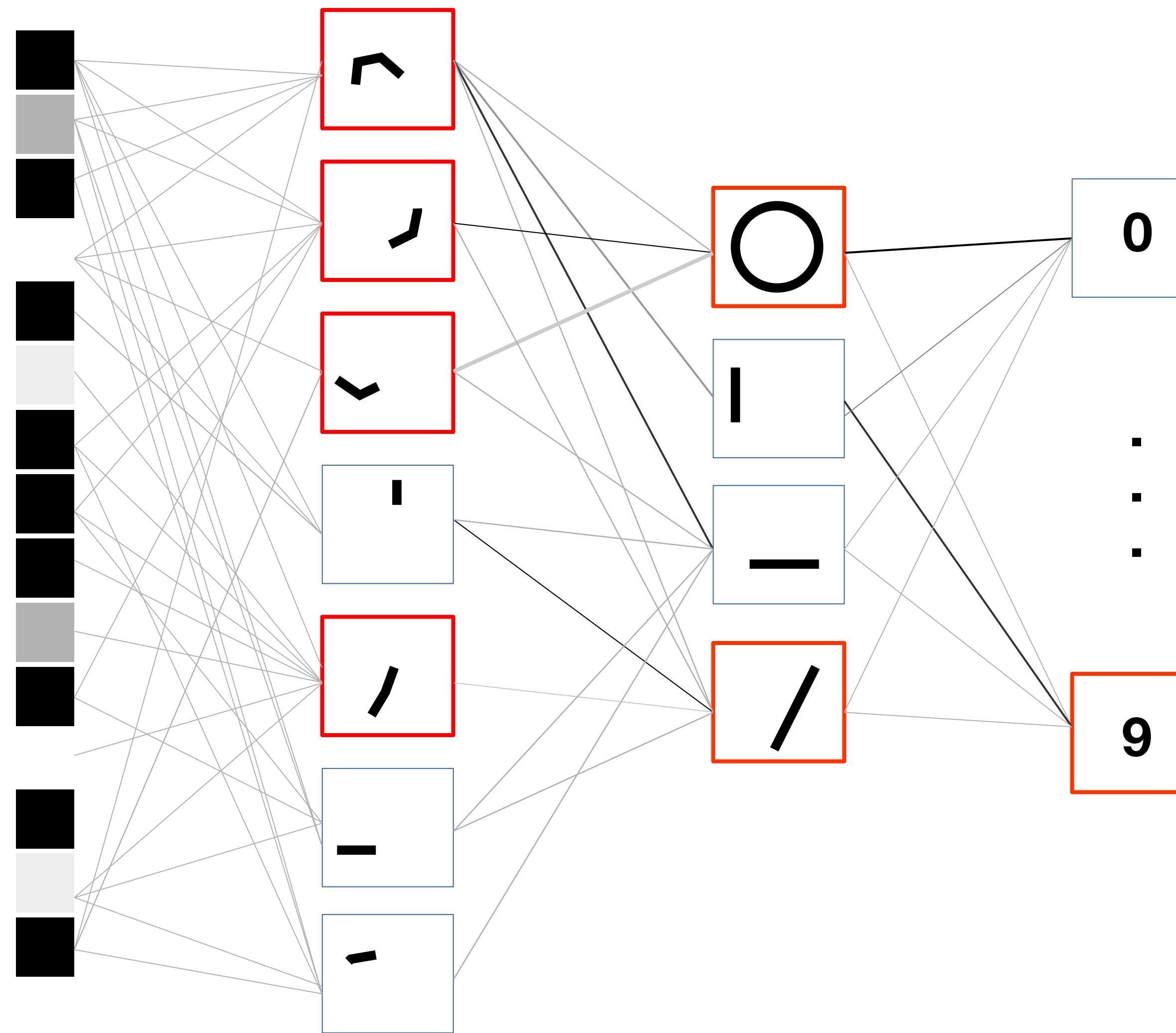
Patrón de activación de una capa causa una patrón específico en la siguiente

youtube.com/watch?v=aircAruvnKk

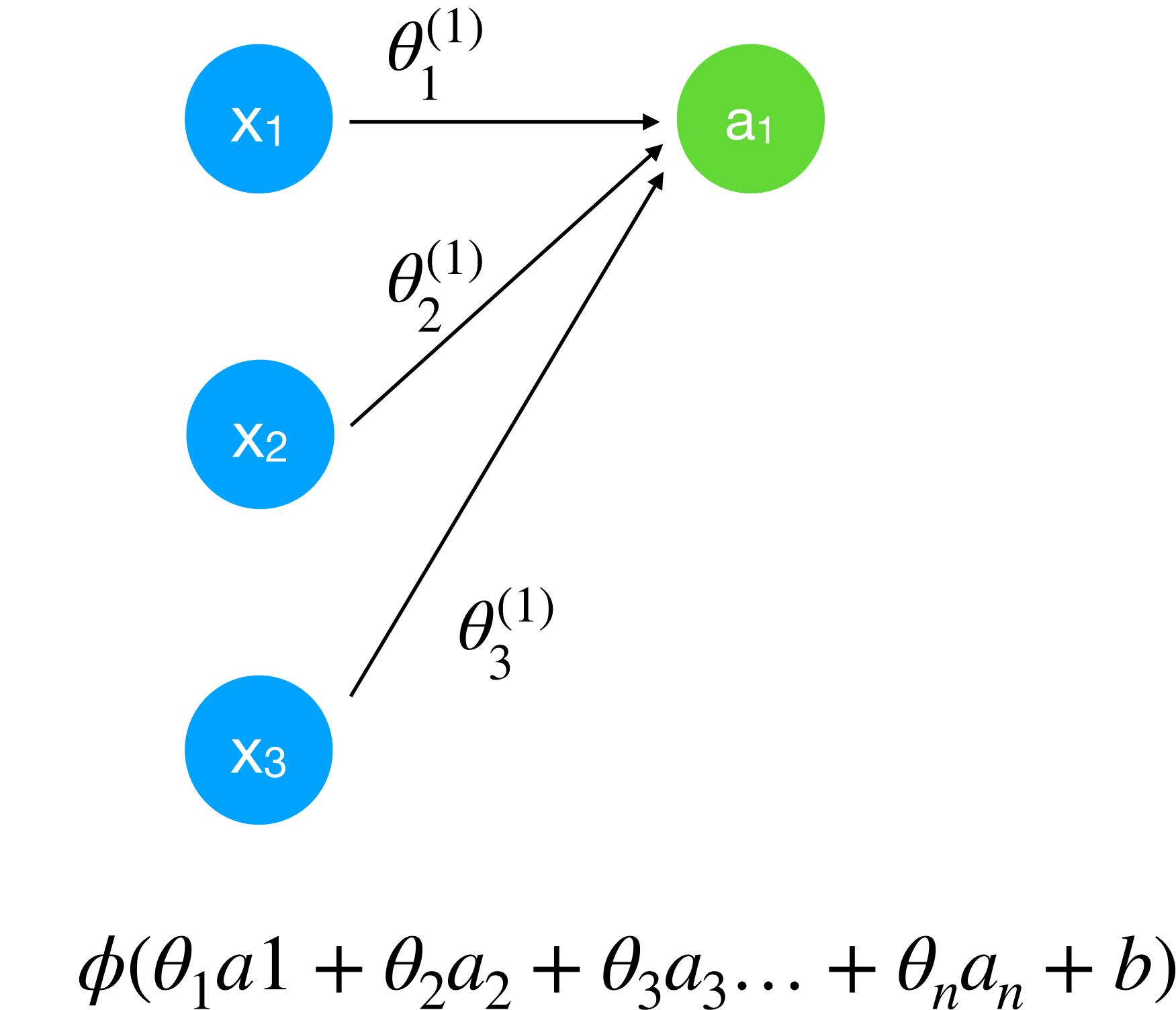
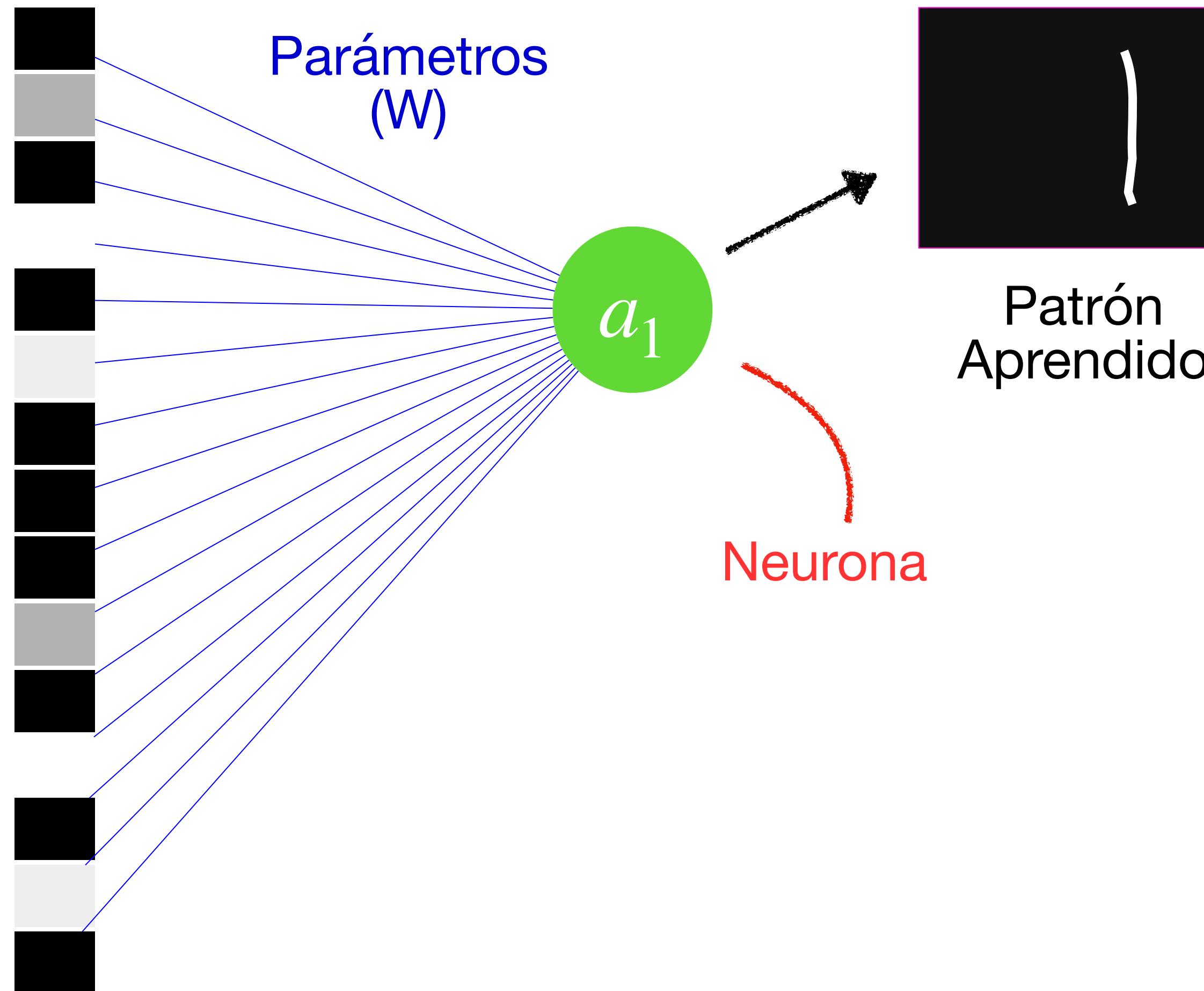
Arquitectura Neurona Artificial



Comportamiento Hipotético



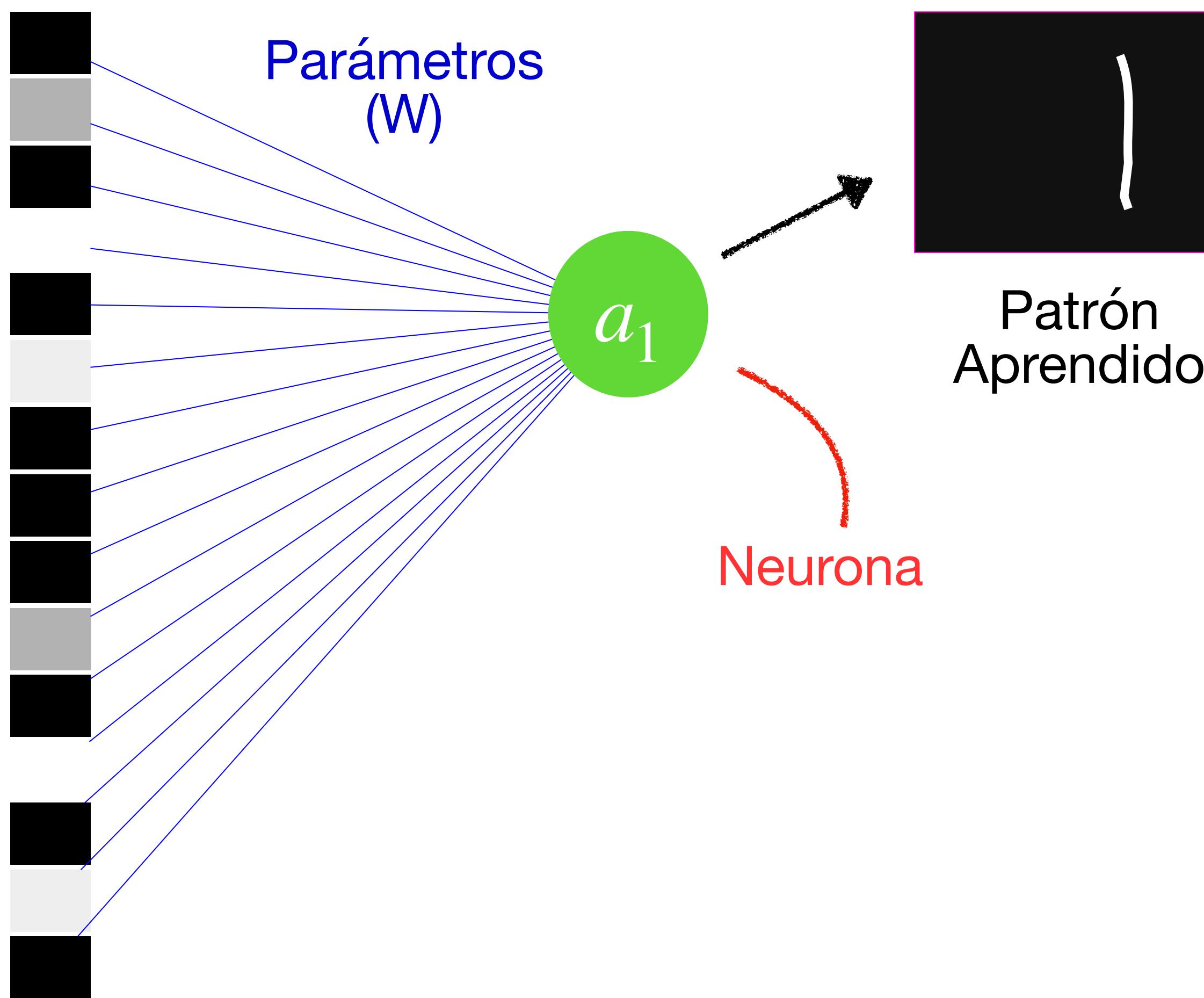
Comportamiento Hipotético



Reconocimiento de elementos básicos

Comportamiento Hipotético

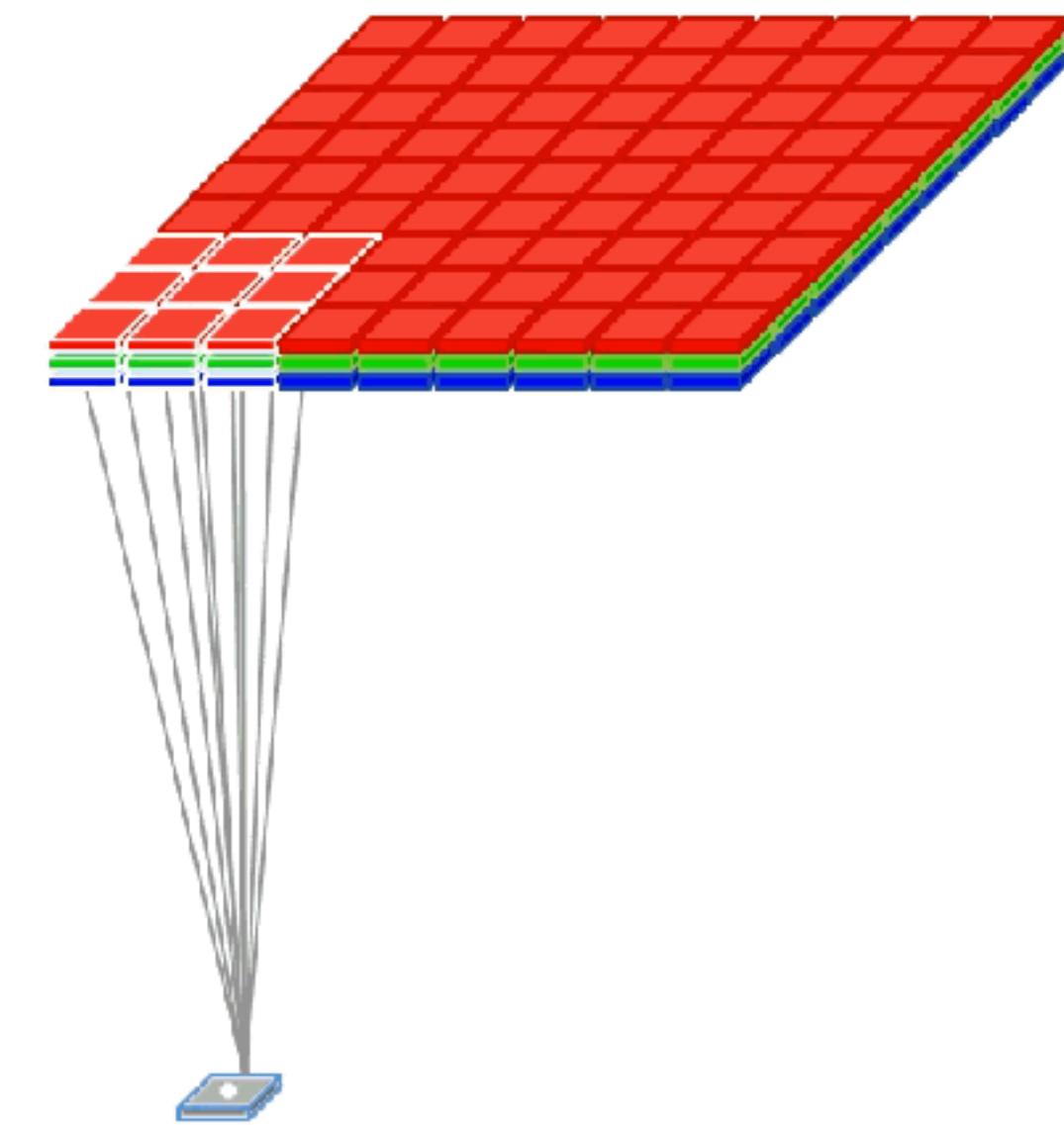
Reconocimiento de elementos básicos



The diagram shows a red square divided into four quadrants. The bottom-right quadrant contains a green vertical bar with a yellow outline. Above this is a mathematical expression for a neural activation function: $\phi(\theta_1 a_1 + \theta_2 a_2 + \theta_3 a_3 \dots + \theta_n a_n + b)$. Above the expression is a symbol resembling a Greek letter theta (Θ) with dashed lines extending downwards to the terms in the equation.

Convolución

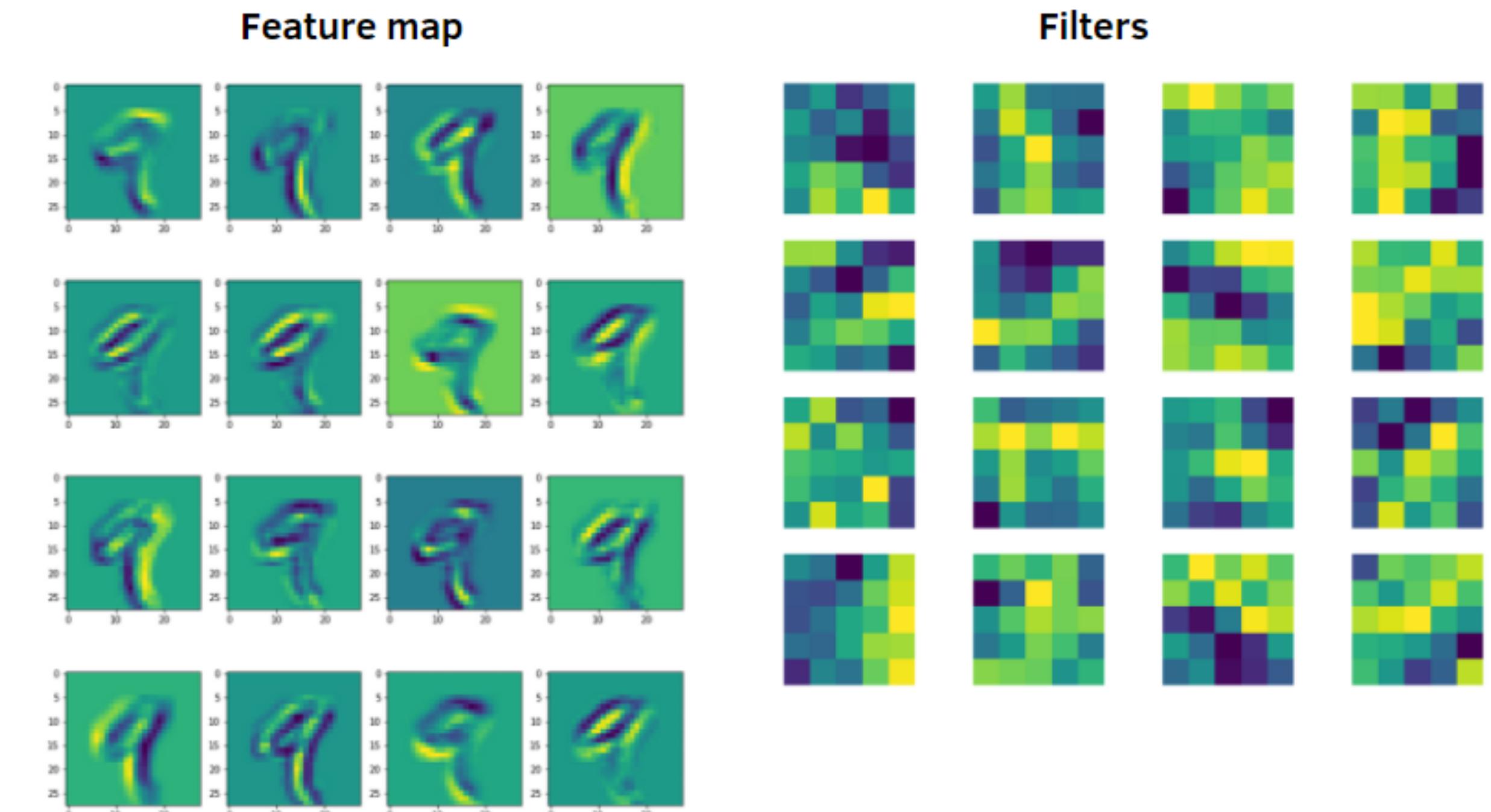
- El valor de un píxel de salida se determina como una suma ponderada de los valores de los píxeles de entrada
- Los valores del kernel o máscara $h(k, l)$ son llamados coeficientes del filtro
- **Encontrar el mismo patrón en diferentes partes de la imagen**



https://medium.com/@adityaraj_64455/it-all-started-with-cnns-alexnet-3023b21bb891

Convolutional Neural Network (CNN)

- La convolución es una operación utilizada para extraer características de una imagen
- Se define mediante un *kernel*
- El kernel es una matriz con menor dimensión que la imagen
- Dos tamaños típicos son de 3x3 y 5x5



Convolutional Neuronal Networks

- Aprender *Kernels* → Extraer características útiles
- Las convoluciones hacen el trabajo pesado
- Existen diferentes **tipos de capas:**
 - Convolutional
 - Pooling
 - Normalization
 - Fully Connected

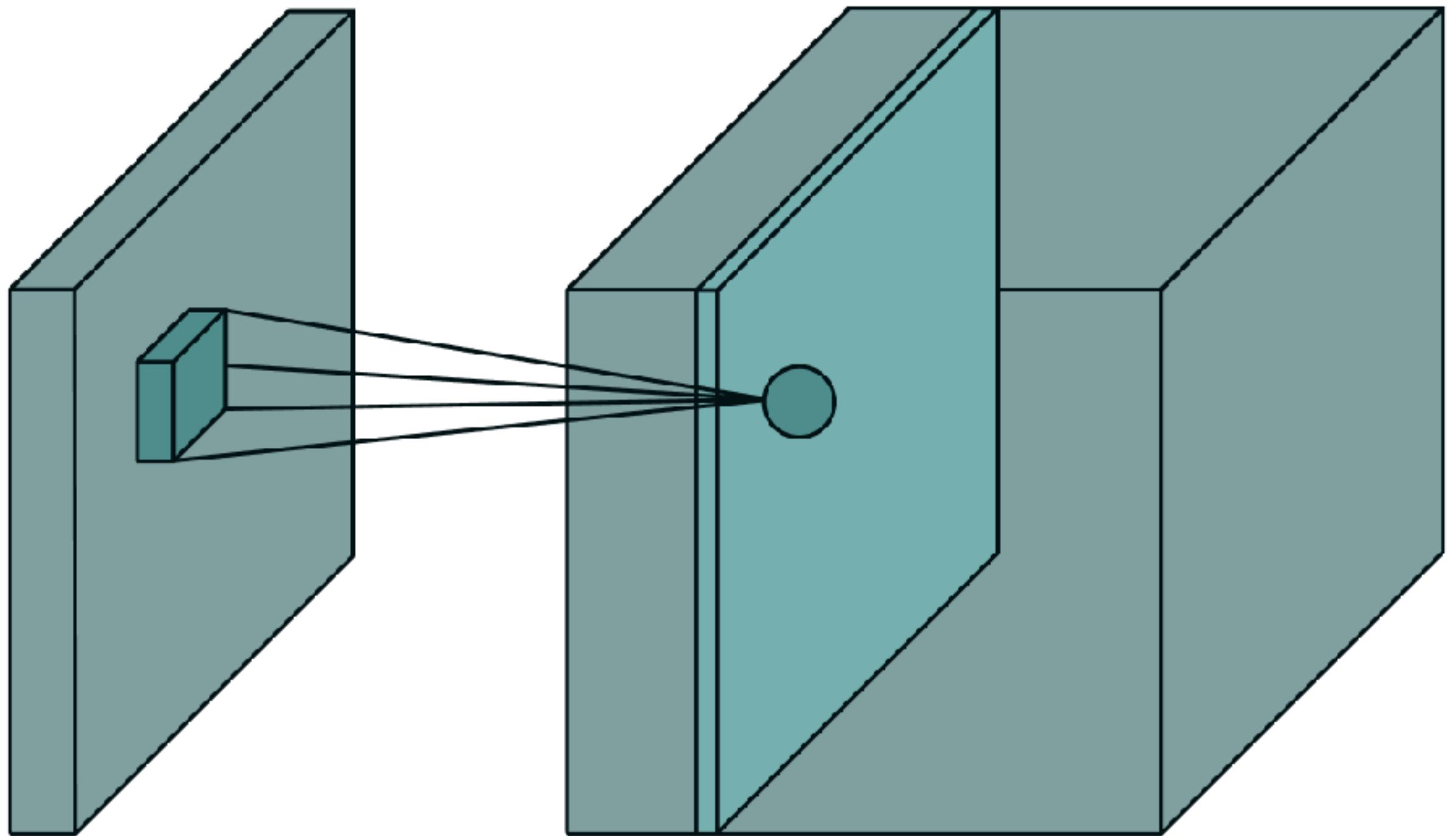


Arquitectura de VGG16

Convolutional Neuronal Networks

Capa Convolucional

- Compuesta de varios *kernels* cuyos valores son aprendidos:
 - i. Realiza las convocaciones
 - ii. Se apilan las activaciones
- Tienen la propiedad de que el filtro es compartido para toda la imagen
- Tamaño del filtro: [W, H, Depth (full)].
Ejemplo: [5,5,3] (RGB)

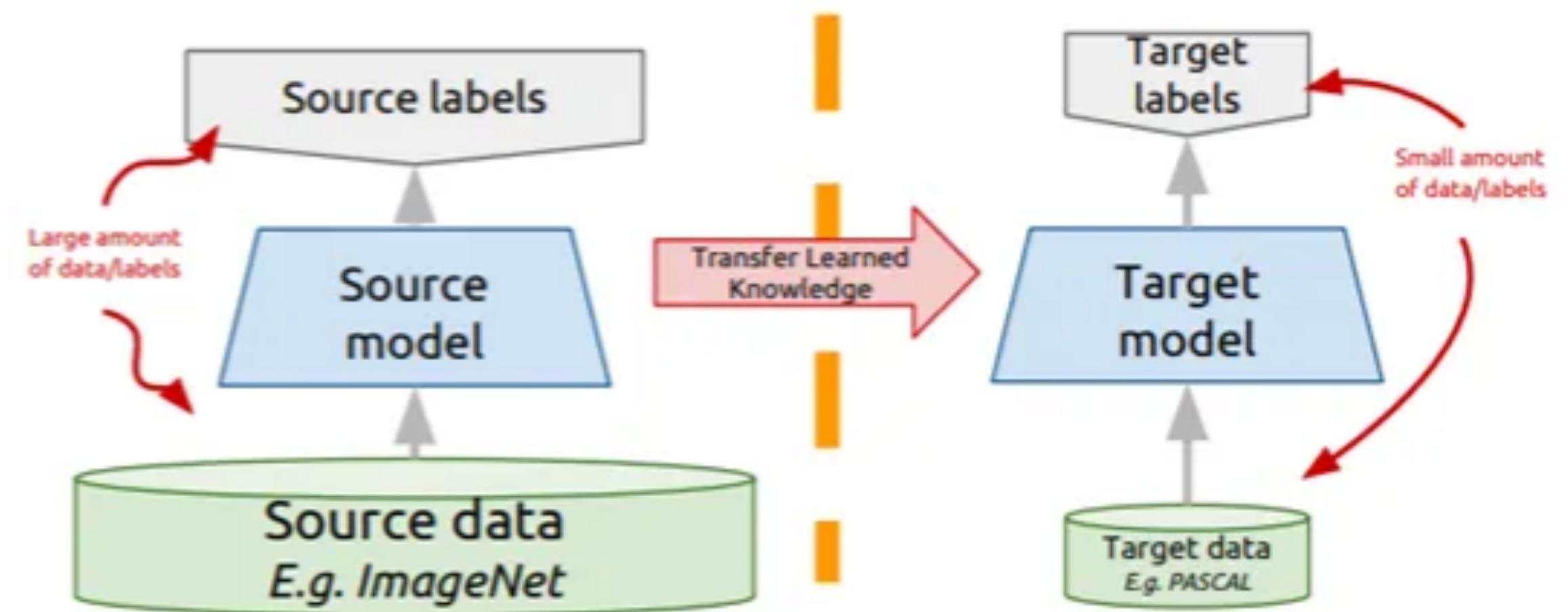


Transfer Learning for Deep Learning

- No entrenar un modelo desde 0
- Tomar una red entrenada en un dominio y tarea diferente
- Adaptarlo al dominio y tarea objetivos

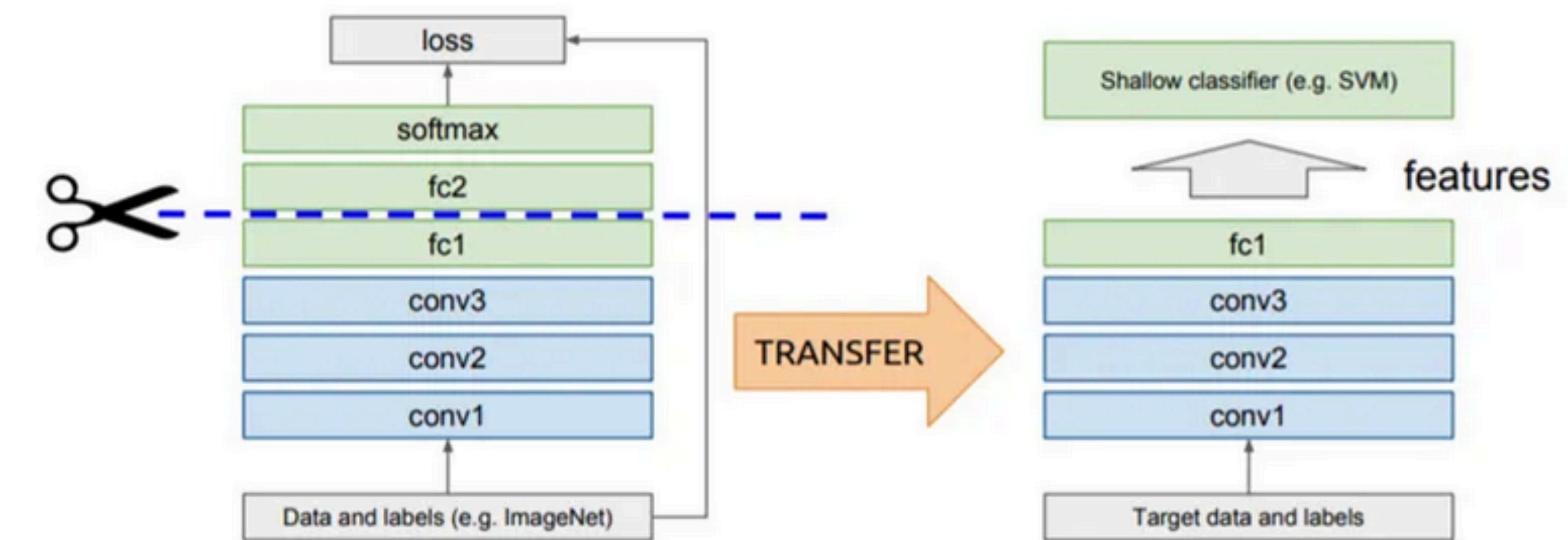
Variaciones:

- A. Mismo dominio diferentes tareas
- B. Diferentes dominios con tareas iguales



Extractores de características

- Los modelos de DL son arquitecturas en capas que aprenden diferentes características en cada capa
- Estas capas se conectan finalmente a una última capa (normalmente FC) para obtener el resultado final.
- Esta arquitectura en capas nos permite utilizar una red pre-entrenada (como VGG) sin su capa final como extractor de características



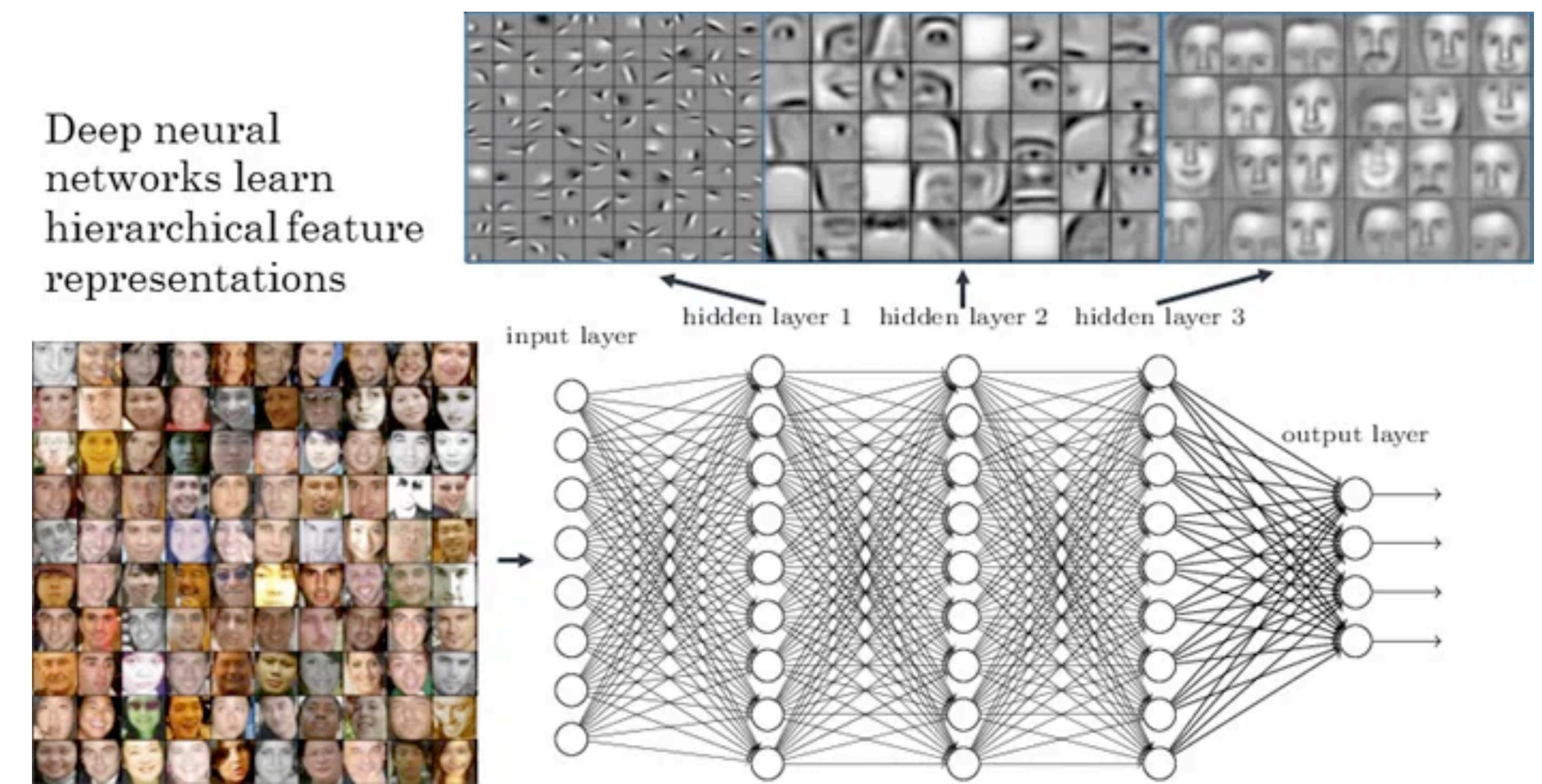
Extractores de características

- La idea principal es aprovechar las capas del modelo pre-entrenado para extraer características
- No actualizar los pesos de las capas del modelo per-entrenado (Extractor de características) durante el entrenamiento con los nuevos datos para la nueva tarea.

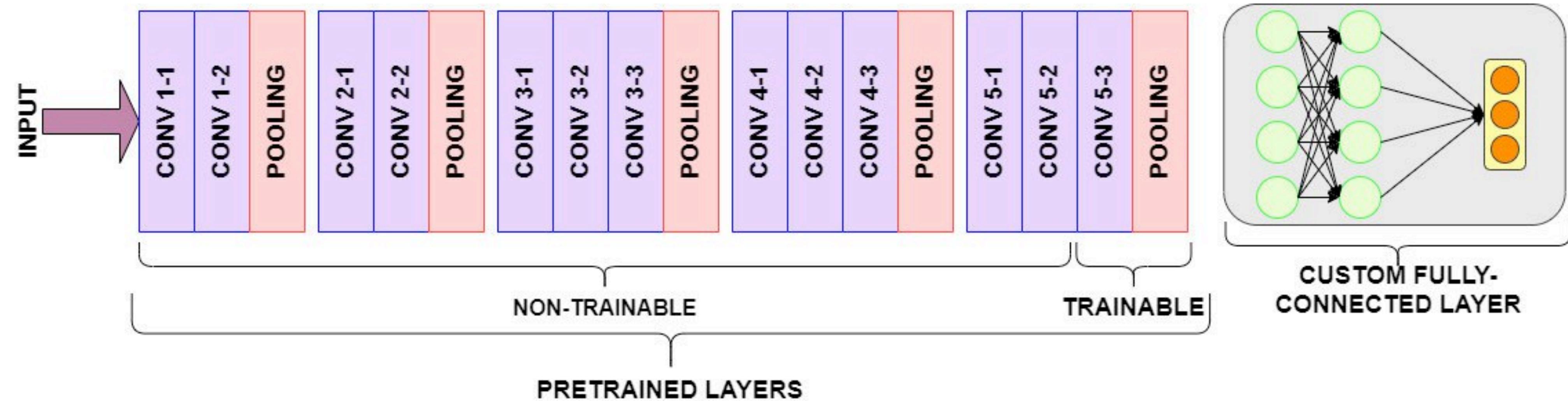


Fine-tuning

- Se ha visto que las capas iniciales captan características genéricas, mientras que las posteriores se centran más en la tarea específica
- Podemos congelar (fijar pesos) ciertas capas mientras re-entrenamos y afinar el resto de ellas para adaptarlas a nuestras necesidades.
- En este caso, aprovechamos la arquitectura general de la red y utilizamos sus estados (pesos) como punto de partida para el re-entrenamiento.



Fine-tuning

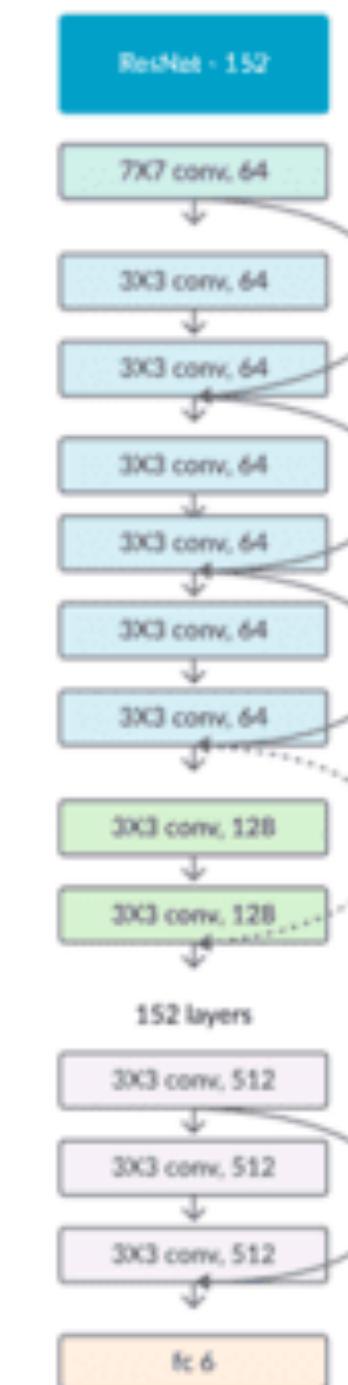


Las capas del modelo fuente (Extractor de características) pueden ser:

- A. Congeladas (No se actualizan los pesos) - Hay pocos datos etiquetados (evitar overfitting)
- B. Finetune: (Actualizar con un LR suave) - Hay suficientes datos de entrenamiento

Modelos Pre-entrenados

- El requisito para el TL es la presencia de modelos (generales) que funcionen bien en sus tareas
- Muchas de las arquitecturas del estado del arte han sido compartidas por sus autores
- Los modelos entrenados están disponibles a través de diferentes medios, e.g: *Tensorflow*, proporciona una interfaz para descargar algunos modelos populares.



Modelos Pre-entrenados

Para visión se puede aprovechar algunos modelos como:

- VGG-16
- MobileNet
- Inception V3
- Xception
- ResNet-50

Extract features with VGG16

```
from tensorflow.keras.applications.vgg16 import VGG16
from tensorflow.keras.preprocessing import image
from tensorflow.keras.applications.vgg16 import preprocess_input
import numpy as np

model = VGG16(weights='imagenet', include_top=False)

img_path = 'elephant.jpg'
img = image.load_img(img_path, target_size=(224, 224))
x = image.img_to_array(img)
x = np.expand_dims(x, axis=0)
x = preprocess_input(x)

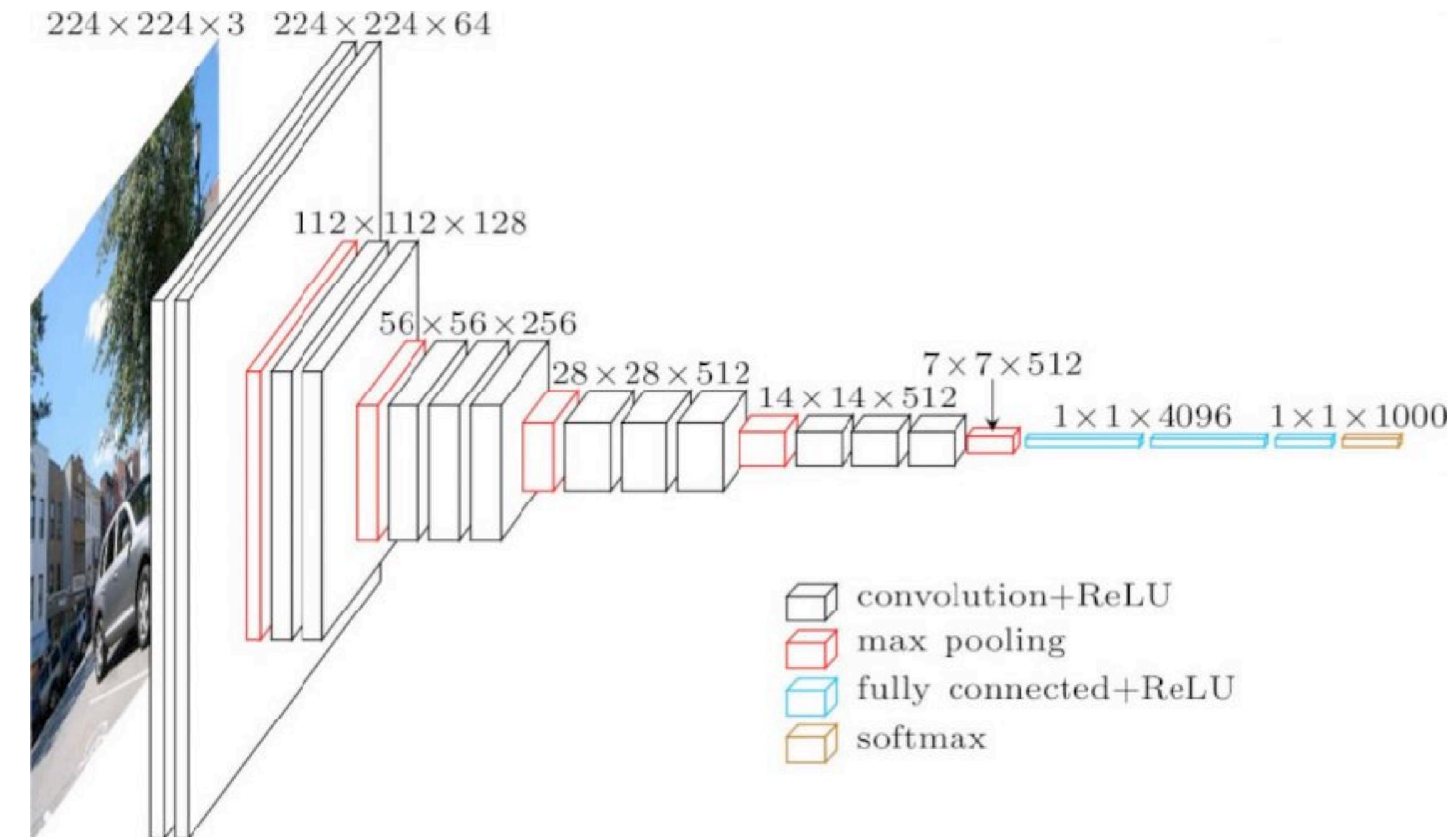
features = model.predict(x)
```

Más arquitecturas en: <https://keras.io/api/applications/>

<https://keras.io/api/applications/>

VGG16

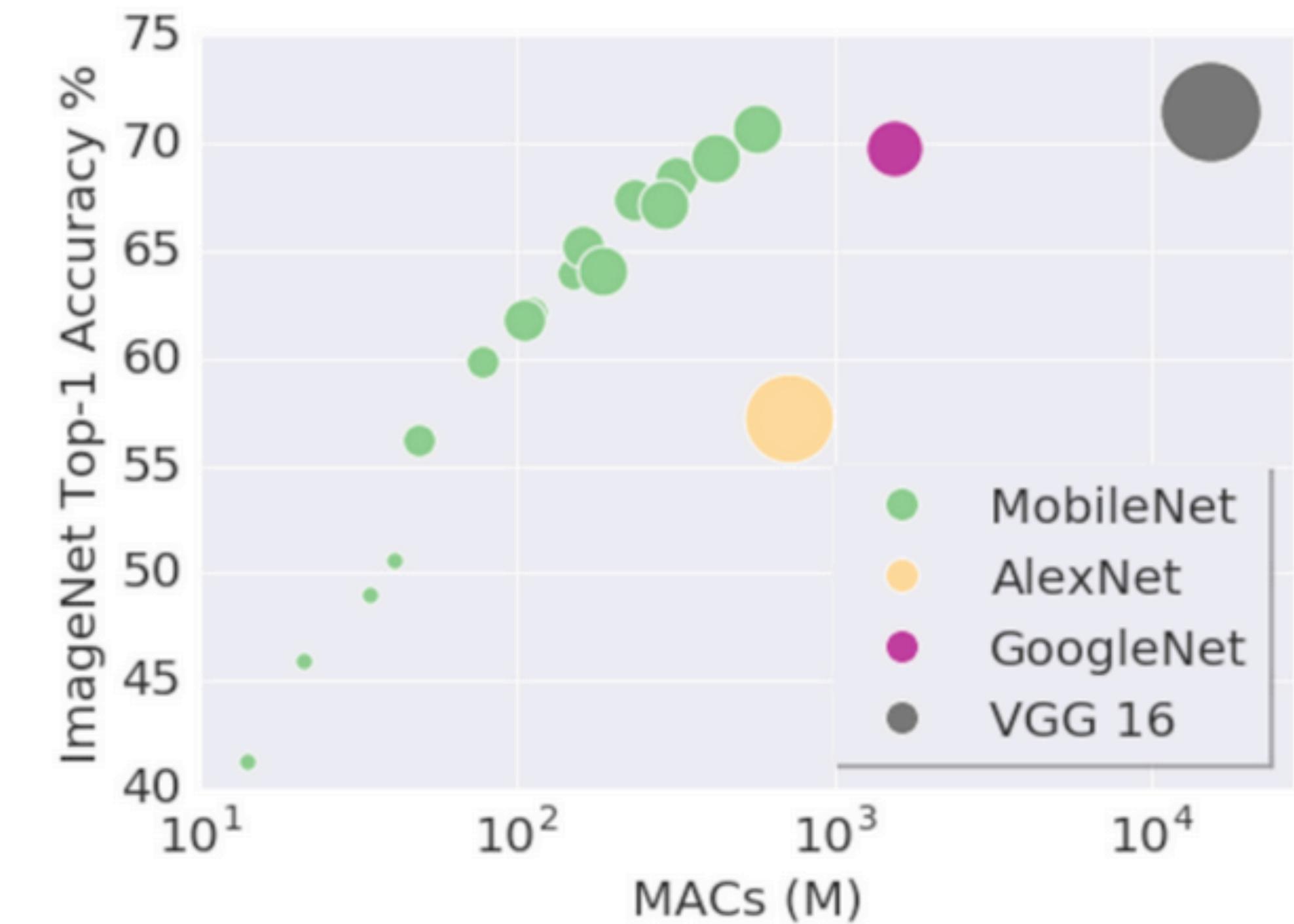
- El 16 se refiere a 16 capas entrenables:
 - 13 capas convolucionales
 - 5 capas Max Pooling
 - 3 capas densas
- VGG16 toma una imagen de entrada de 224, 244 con 3 canales RGB.
- Patrón de la arquitectura (mínimo de hiperparámetros)
 - Capas de convolución de filtro 3x3 con stride 1 con el mismo padding
 - Capa maxpool de filtro 2x2 de stride 2.



2015 - Very Deep Convolutional Networks for Large-Scale Image Recognition

MobileNet

- Una CNN sencilla pero eficiente y poco intensiva en recursos
- *MobileNet* es una clase de CNN de código abierto desarrollada por Google.
- **Convolución separable:** Reduce significativamente el número de parámetros en comparación con la red con convoluciones regulares con la misma profundidad en las redes.

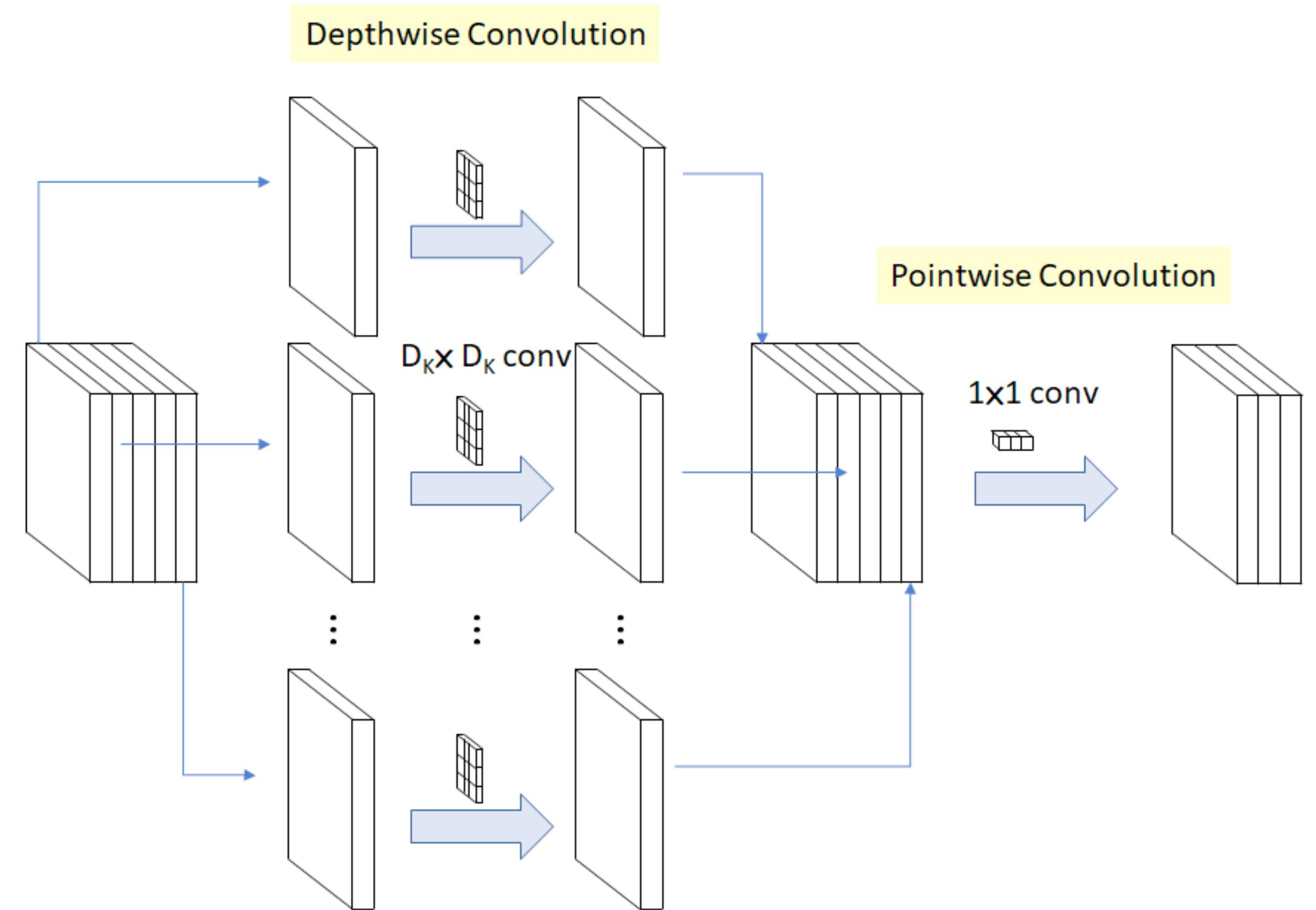


2017 - MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications

MobileNet

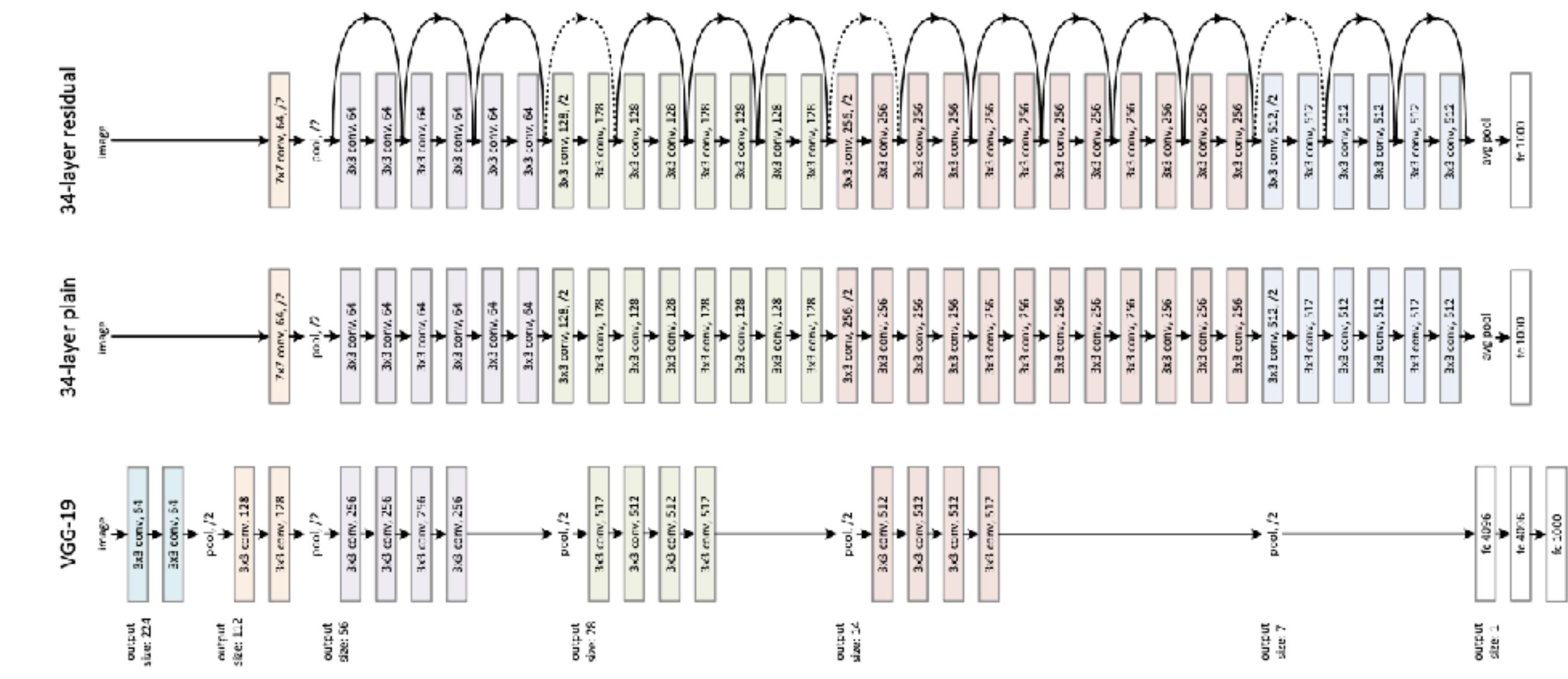
Convolución separable:

- **Convolución de profundidad:** Una convolución por canal (profundidad 1) del tamaño espacial $(D_K \times D_K)$
- **Convolución de punto:** Es una consolidación de (1×1) para reducir el número de canales



ResNet

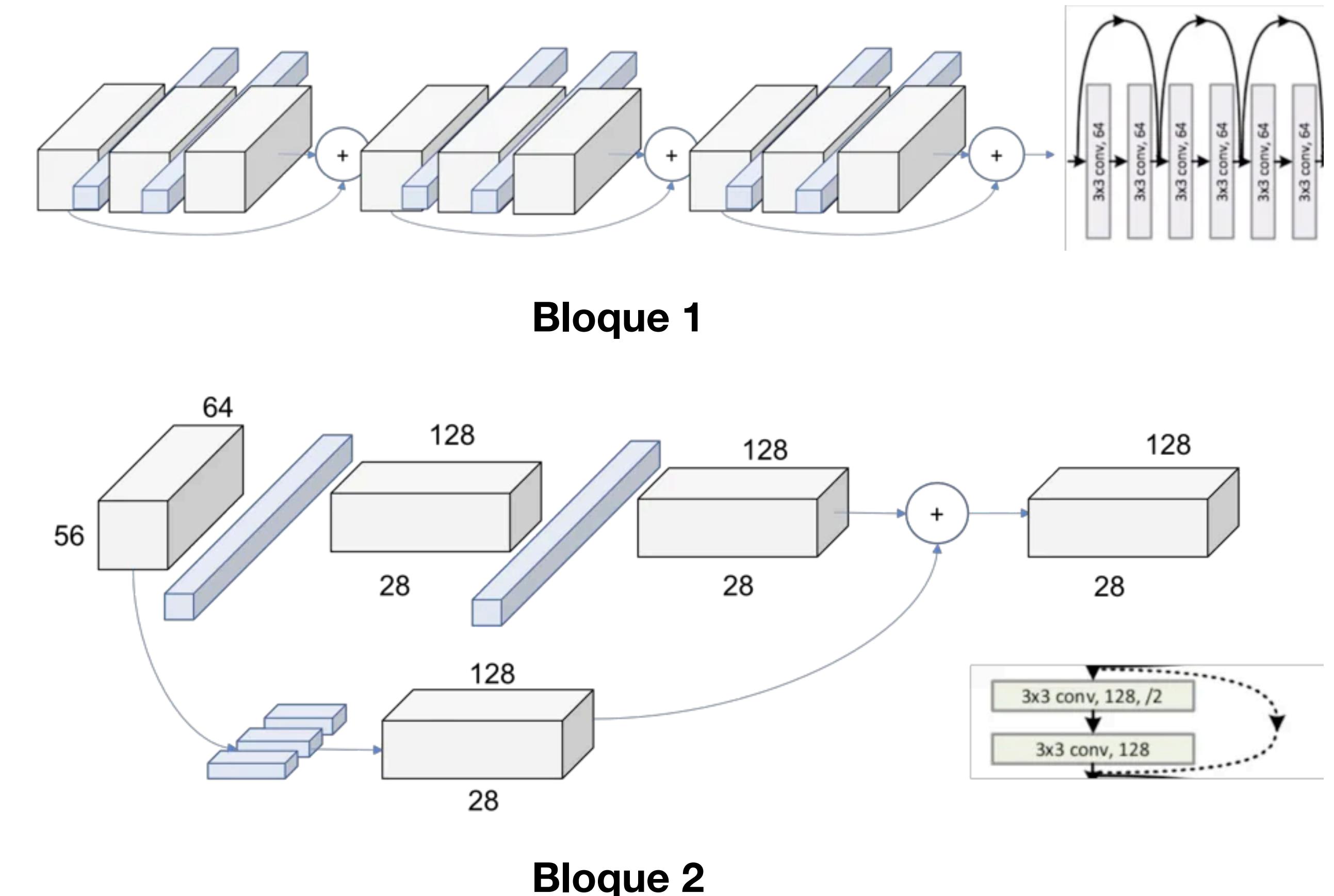
- Uno de los problemas que resuelven las ResNets es el “*gradient-vanish*”
- Esto se debe a que, cuando la red es demasiado profunda, los gradientes desde la función de costo tienden a 0 en las operaciones iniciales tras varias aplicaciones de la regla de la cadena.
- Con *ResNets*, los gradientes pueden fluir directamente a través de “*skip connections*” hacia atrás desde las capas posteriores a los filtros iniciales.



2015 - Deep Residual Learning for Image Recognition

ResNet

- Red Residual: Para resolver el problema del desvanecimiento de la gradiente, esta arquitectura introdujo el concepto llamado Bloques Residuales.
- En esta red, utilizamos una técnica denominada conexiones de salto. La conexión de salto conecta las activaciones de una capa con otras capas saltándose algunas capas intermedias.

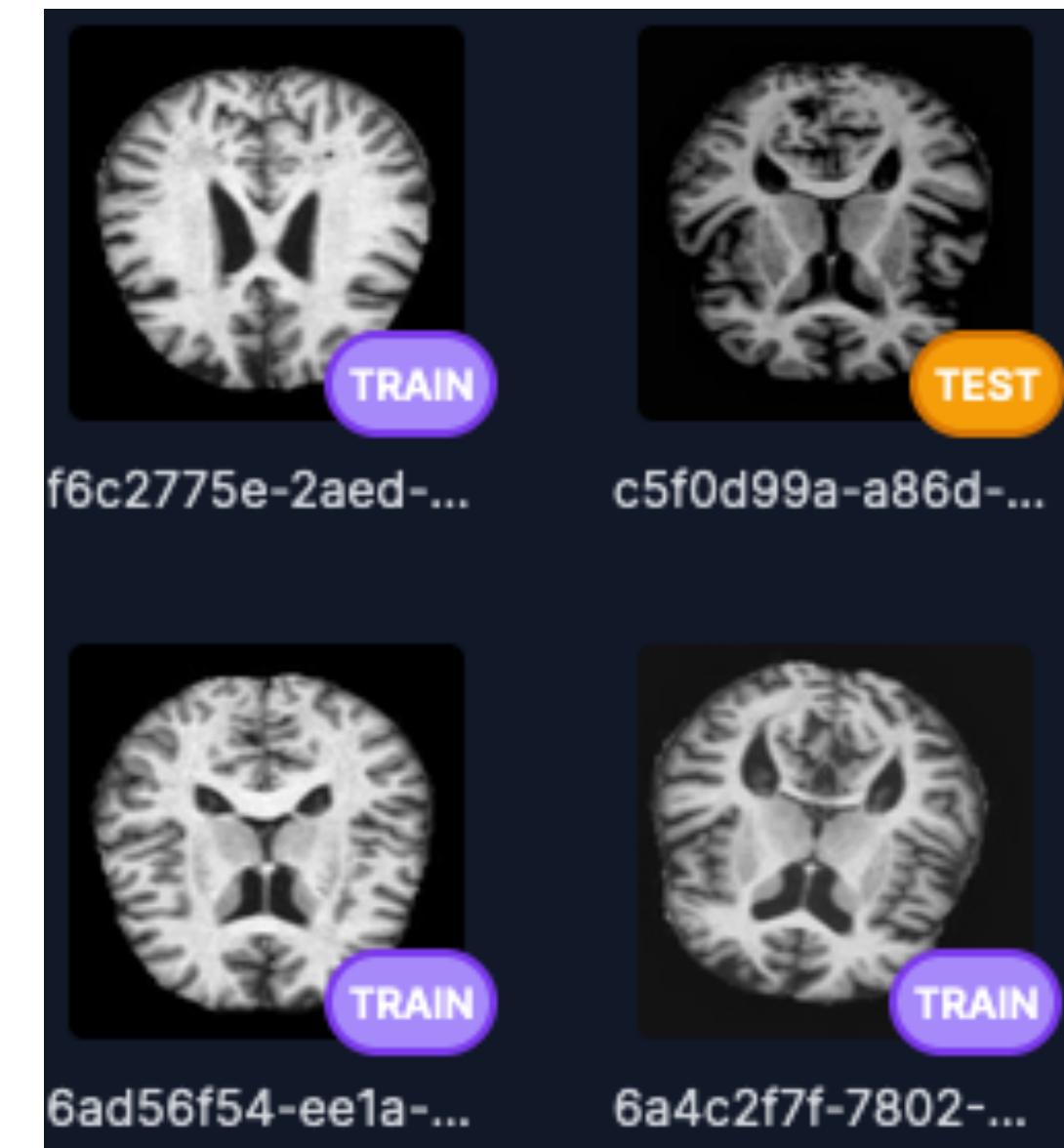


Ejercicio

- Entrenar el modelo del Parcial II usando una red pre-entrenada
- Comprar resultados de ambos modelos

Links:

- [Transfer Learning and Finetune](#)



Se debe Implementar un modelo de aprendizaje de máquinas tipo Convolutional Neural Network (CNN) que ayude a determinar el nivel de Alzheimer de un paciente basado sus resonancias cerebrales.