

Redes Neuronales Convolucionales

Juan Bernardo Gómez Mendoza

Profesor Asociado - DIIEEyC

Universidad Nacional de Colombia Sede
Manizales

Contenido

El perceptrón de Rosenblatt.

- Algoritmo de entrenamiento del perceptrón.
- Perceptrones multicapa (MLP).
- Limitaciones del MLP en tareas visión de máquinas.

Redes neuronales convolucionales (CNN).

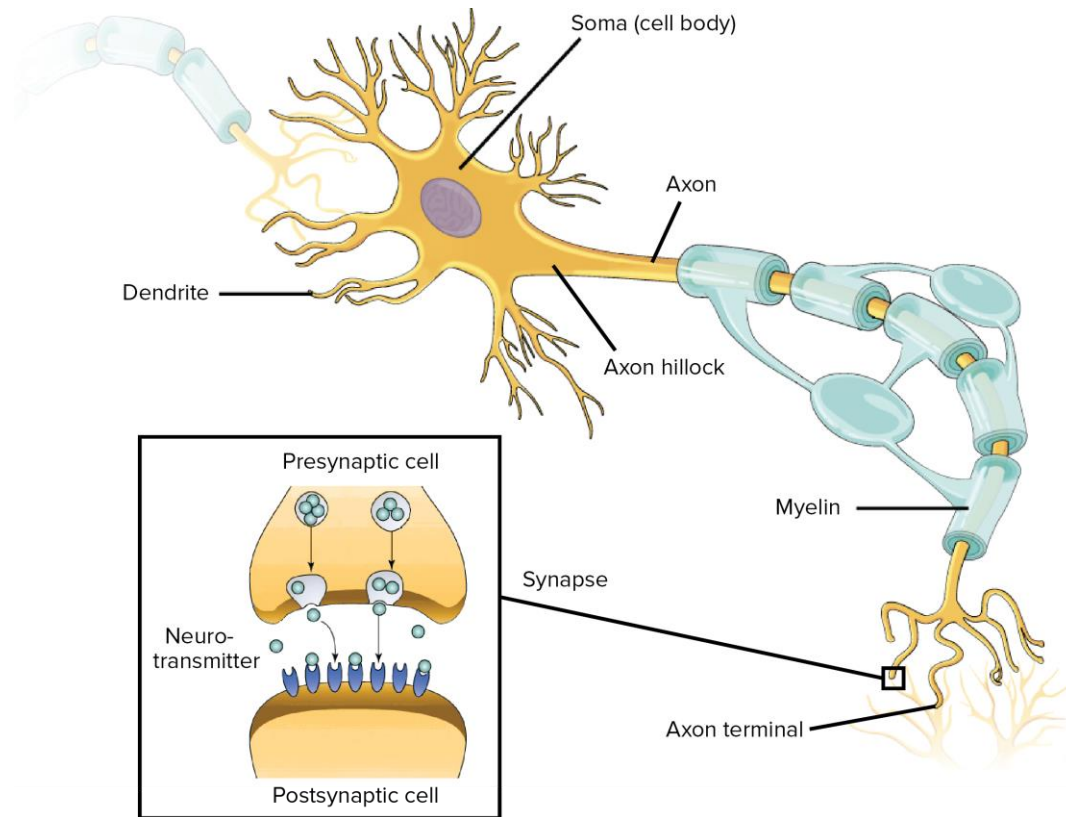
- Arquitectura de una CNN.
- Campo perceptual de una neurona. Efecto convolucional.
- Efecto del pooling en la selección de características.
- Ventajas de las CNNs en tareas de visión de máquinas.

Ejemplos

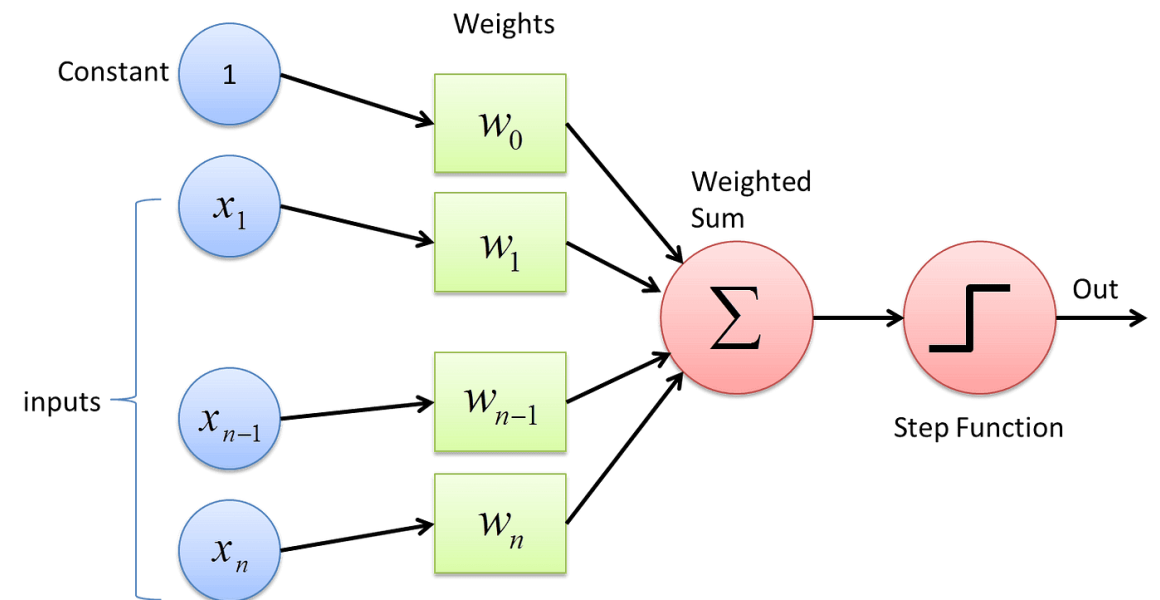
- MobileNet.
- YOLO (You only look once).

Discusión acerca del uso de redes neuronales.

El perceptrón de Rosenblatt



Tomada de [Overview of neuron structure and function \(article\) | Khan Academy](#)



Tomada de [What the Hell is Perceptron?. The Fundamentals of Neural Networks | by SAGAR SHARMA | Towards Data Science](#)

Diapositiva dejada intencionalmente en blanco

Algoritmo de entrenamiento del perceptrón

Salida del perceptrón

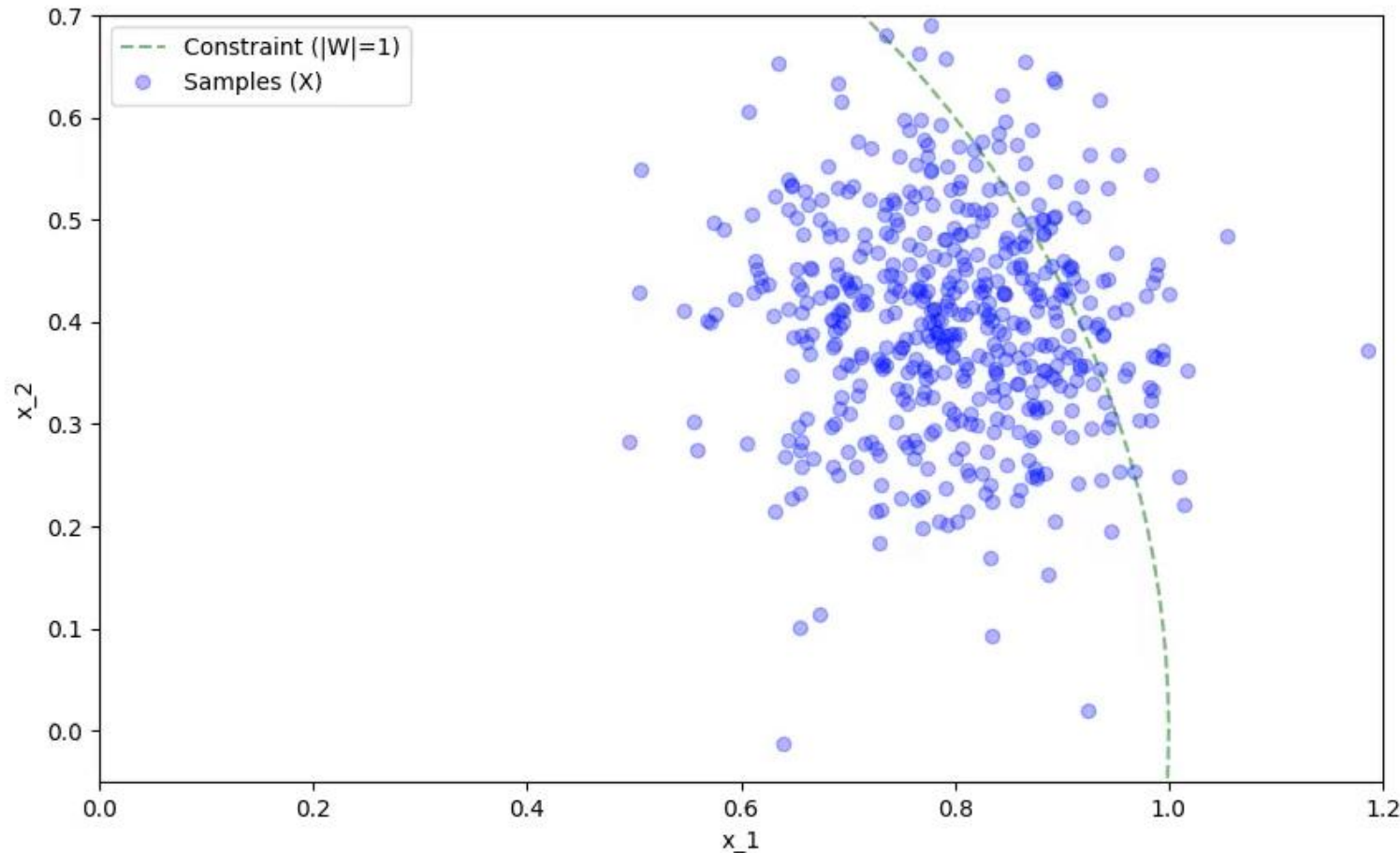
$$y_i = \text{signo}(\mathbf{W}^T \mathbf{X}_i)$$

Cuando no tenemos etiquetas (regresión)

$$\begin{aligned}\mathbf{W}_{k+1} &= \mathbf{W}_k + \Delta \mathbf{W}_k \\ \Delta \mathbf{W}_k &= \eta(\mathbf{X}_i - \mathbf{W}_k) \\ &\text{con } 0 < \eta < 1\end{aligned}$$

Cuando tenemos etiquetas (clasificación)

$$\begin{aligned}\mathbf{W}_{k+1} &= \mathbf{W}_k + \Delta \mathbf{W}_k \\ \Delta \mathbf{W}_k &= \eta e \mathbf{X}_i \\ &\text{con } 0 < \eta < 1; \\ &y \ e = y_j - \hat{y}_j\end{aligned}$$

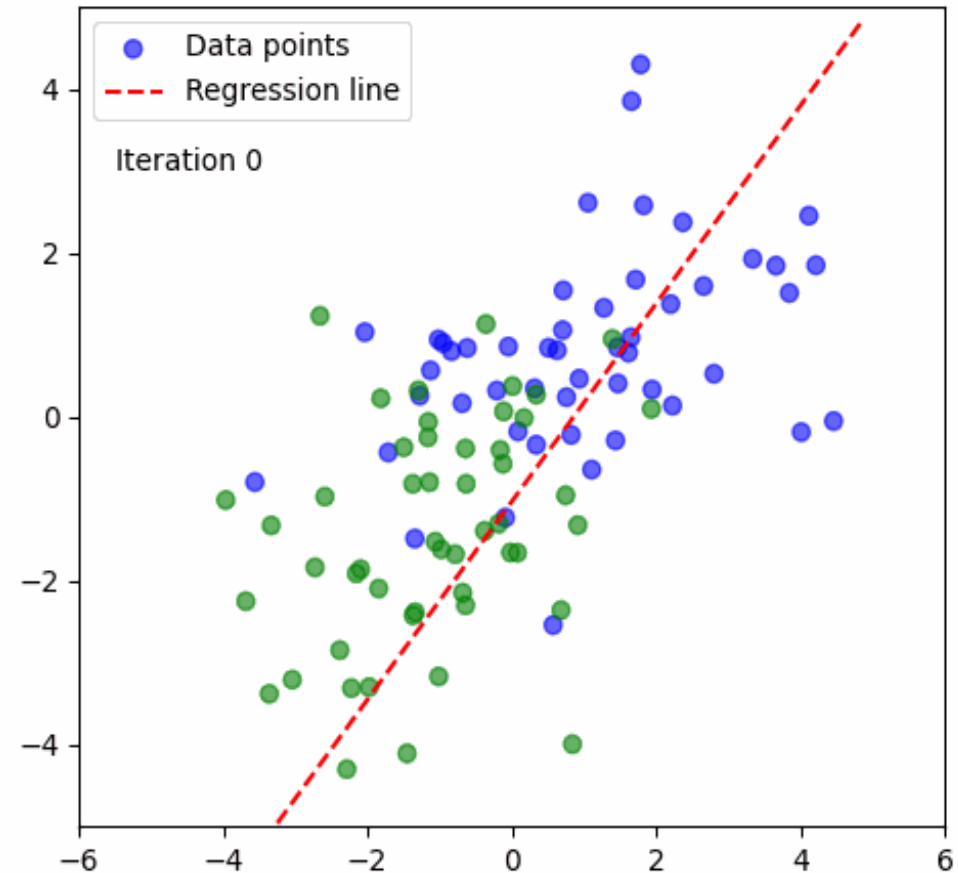
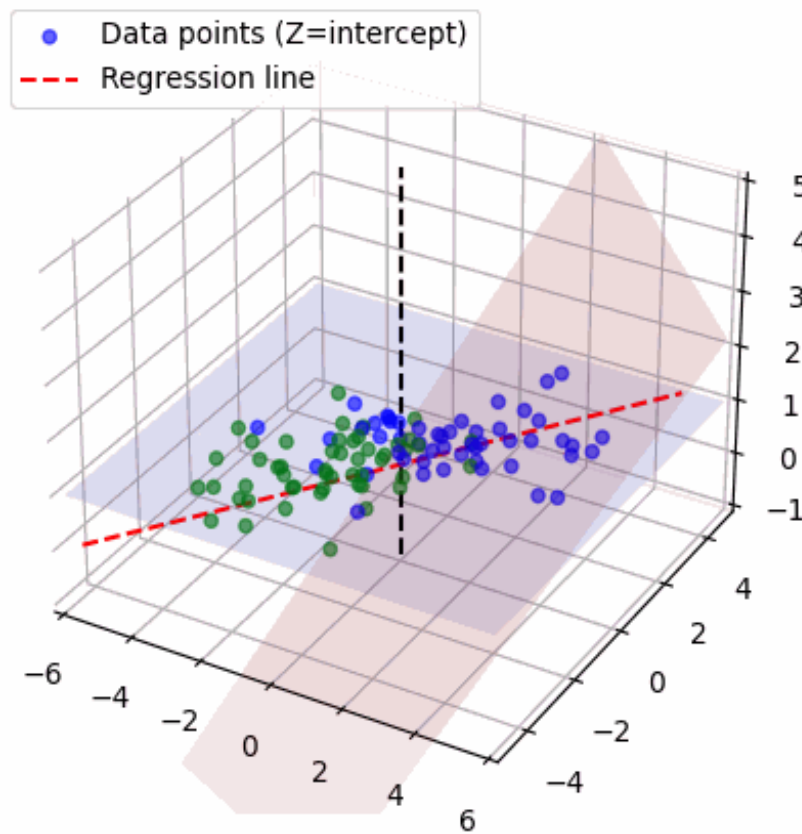


Esta simulación fue realizada usando una versión modificada de la regla de entrenamiento sin etiquetas:

$$\Delta \mathbf{W}_k = \eta \left(\frac{\mathbf{X}_i}{|\mathbf{X}_i|} - \mathbf{W}_i \right)$$

Algoritmo de entrenamiento del perceptrón

Algoritmo de entrenamiento del perceptrón



Algoritmo de entrenamiento del perceptrón

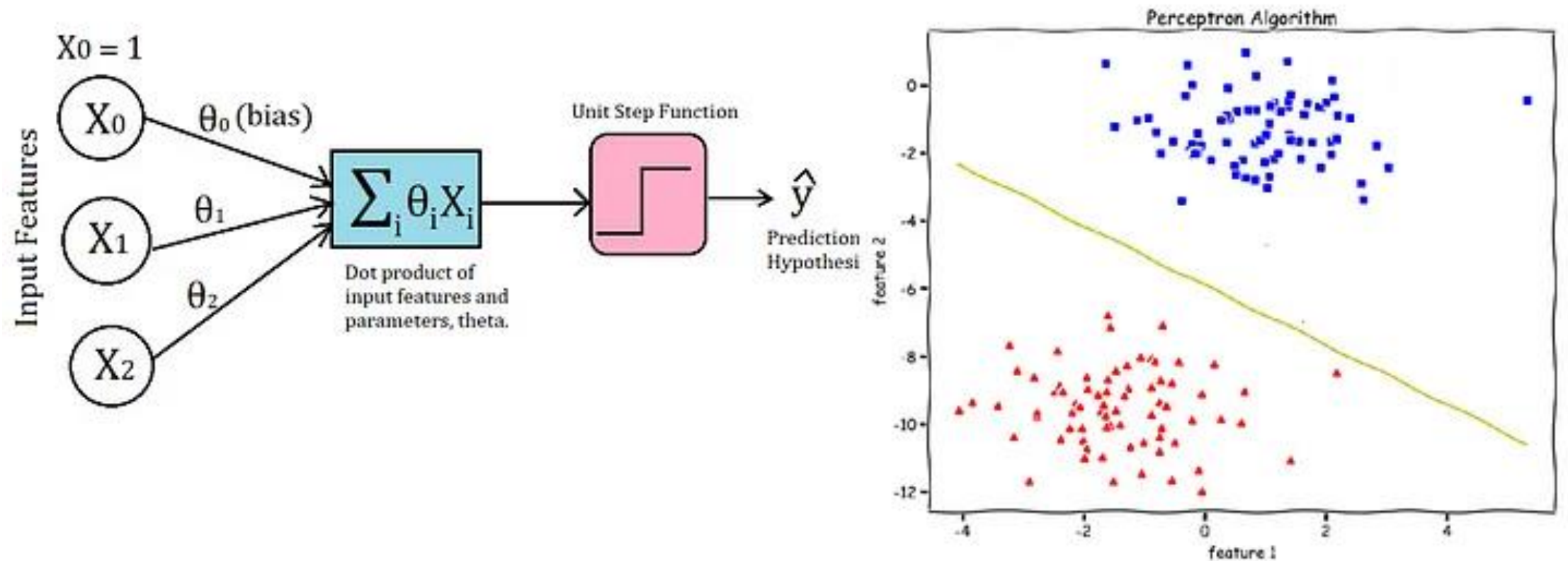


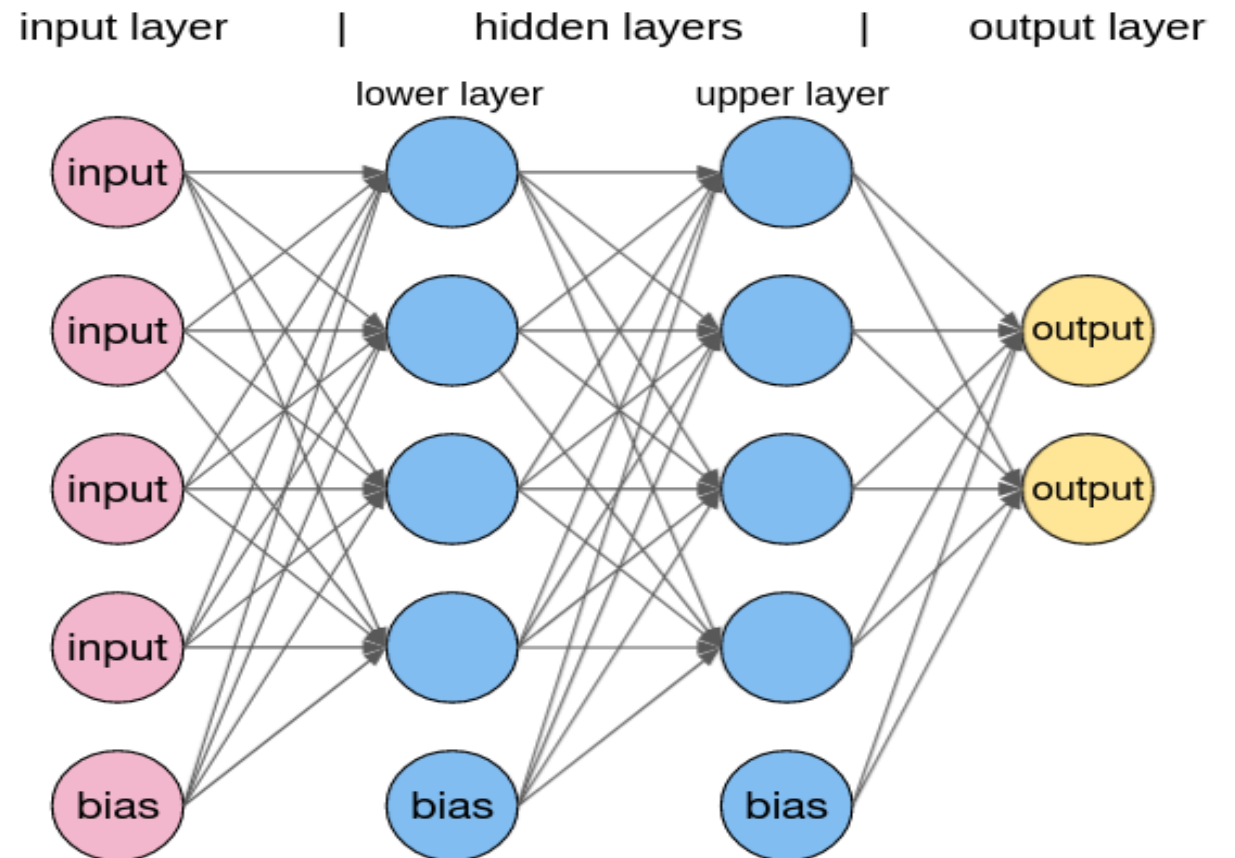
Imagen tomada de [Implementing the Perceptron Algorithm in Python | by Suraj Verma | Towards Data Science](#)

Perceptrón multicapa (MLP)

El perceptrón multicapa es un regresor universal.

Tiene una estructura completamente conectada hacia adelante.

Puede aproximar relaciones de entrada-salida no dinámicas*.



Tomada de [Introduction to how an Multilayer Perceptron works but without complicated math | by SangGyu An | CodeX | Medium](#)

Perceptrón multicapa (MLP)

- Los perceptrones multicapa con funciones de activación no lineales sirven como aproximadores universales.

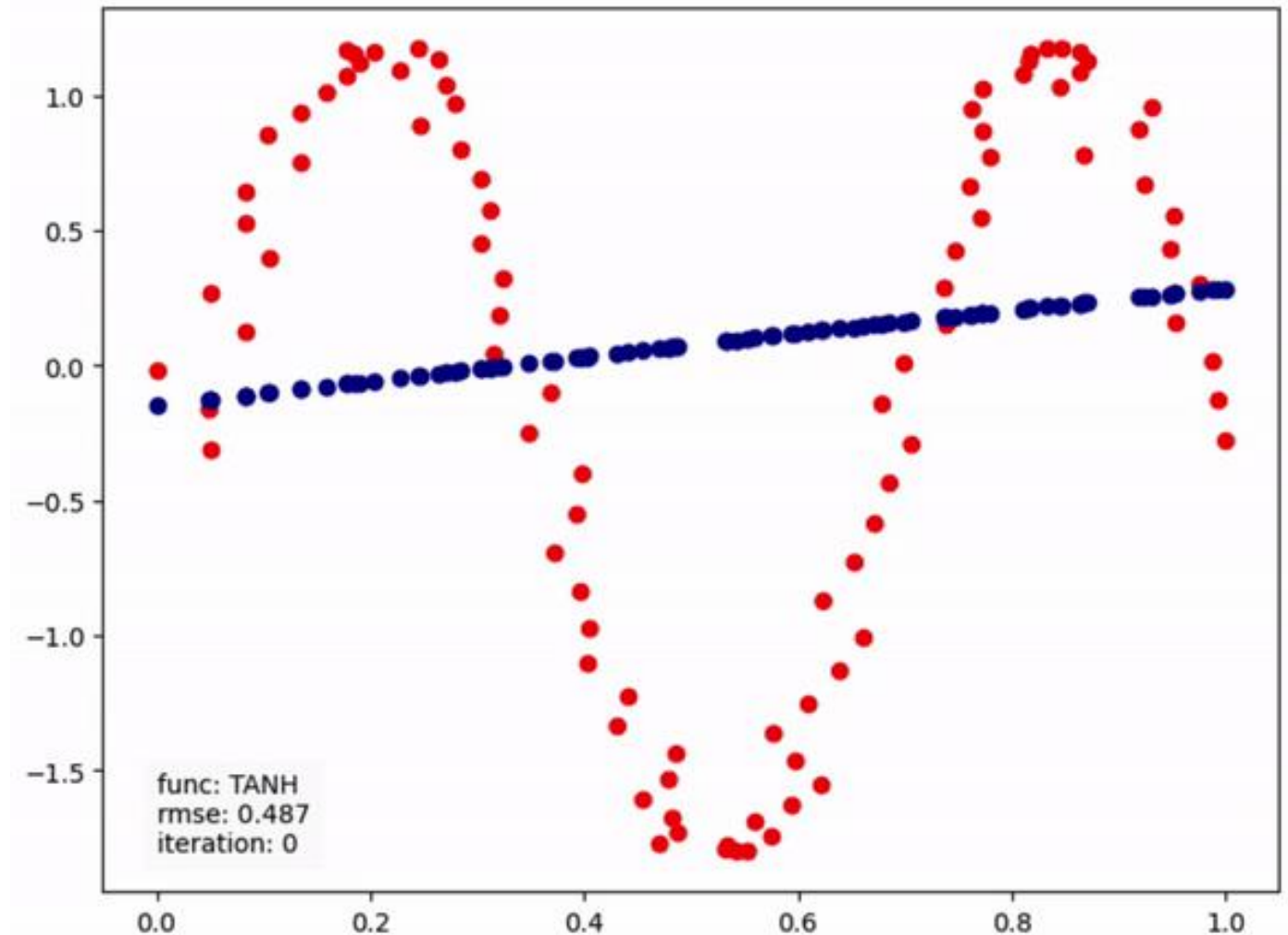


Imagen tomada de [Regression neural-network from scratch | by Razsh | Medium](#)

Perceptrón multicapa (MLP)

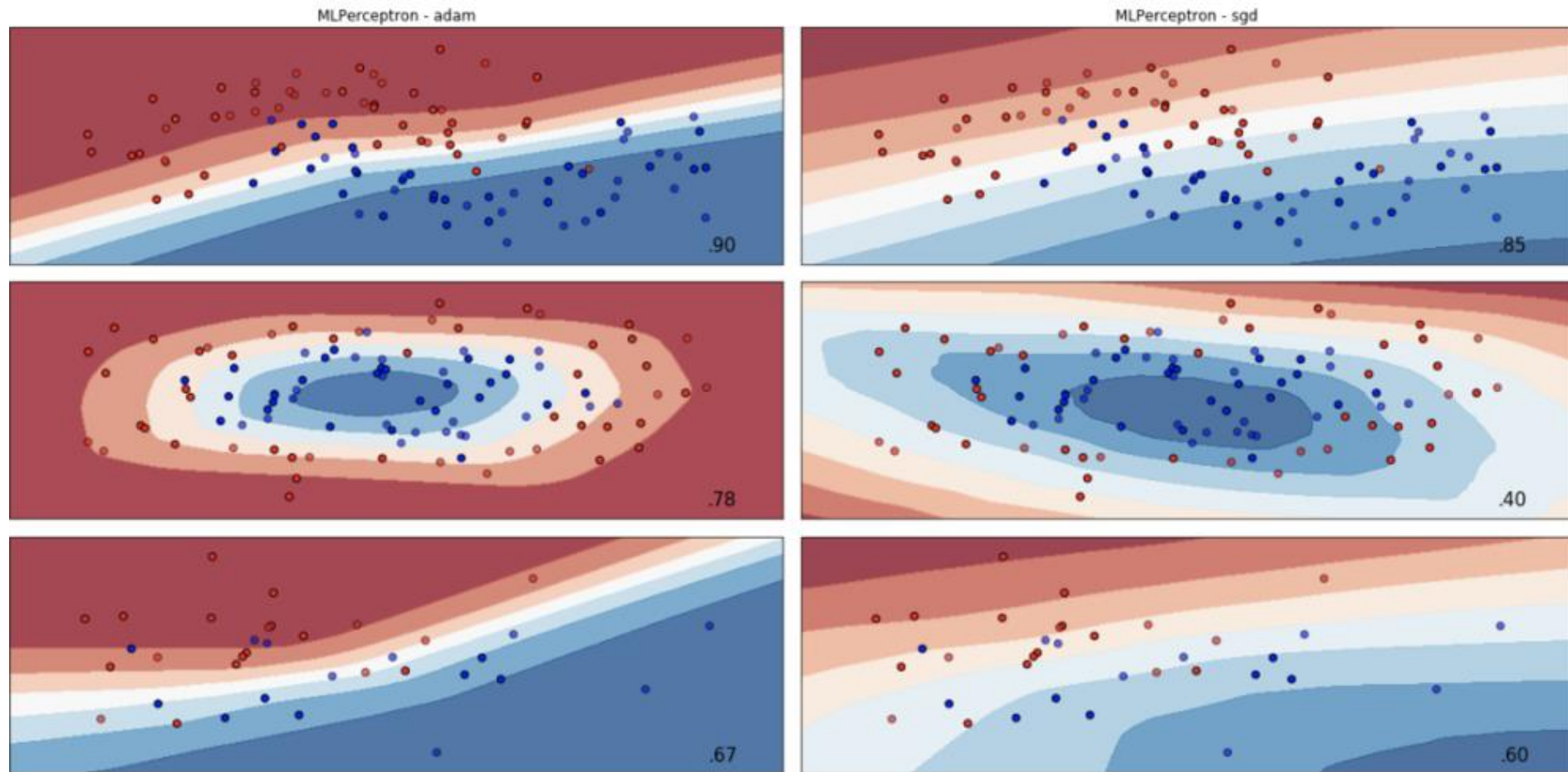
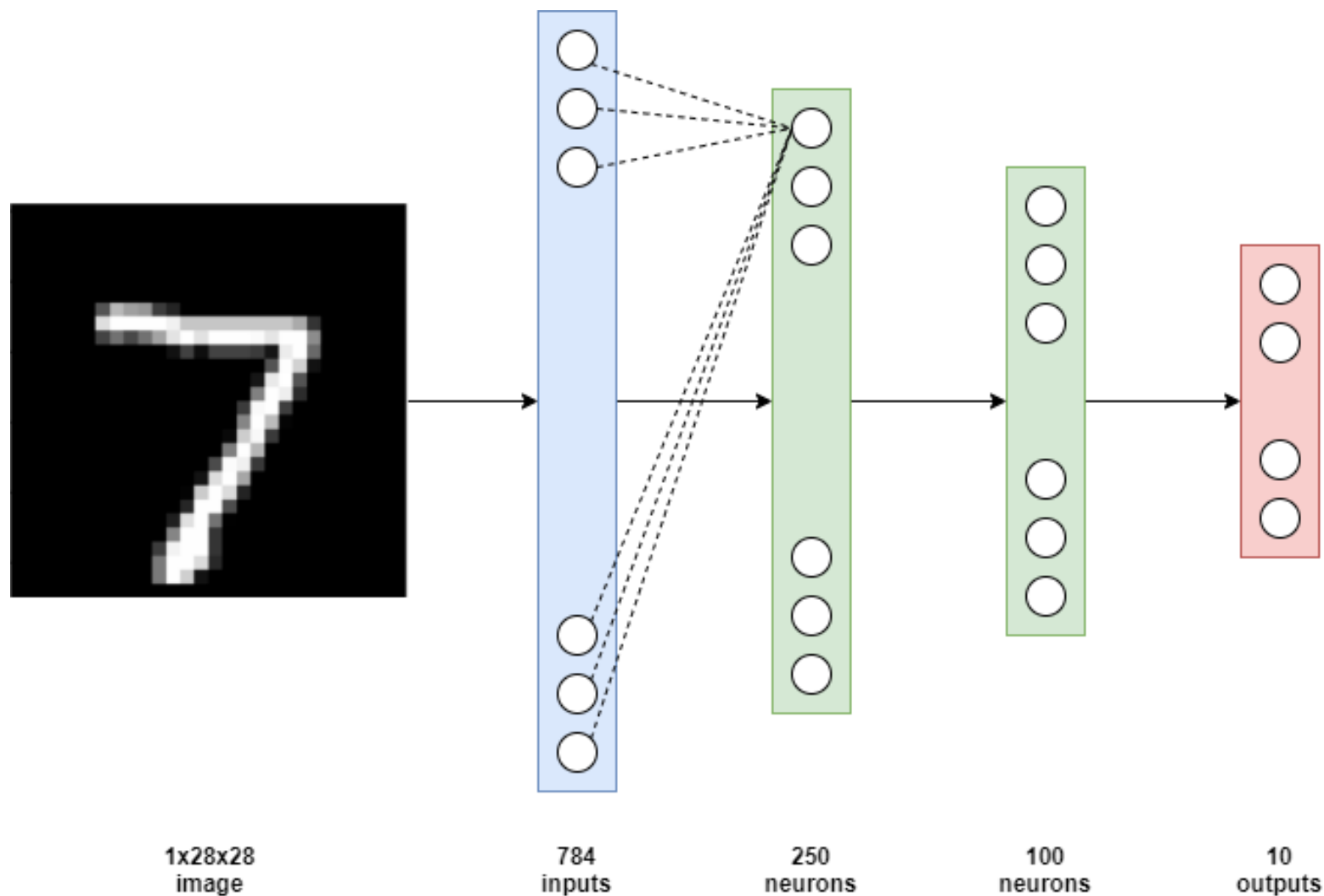
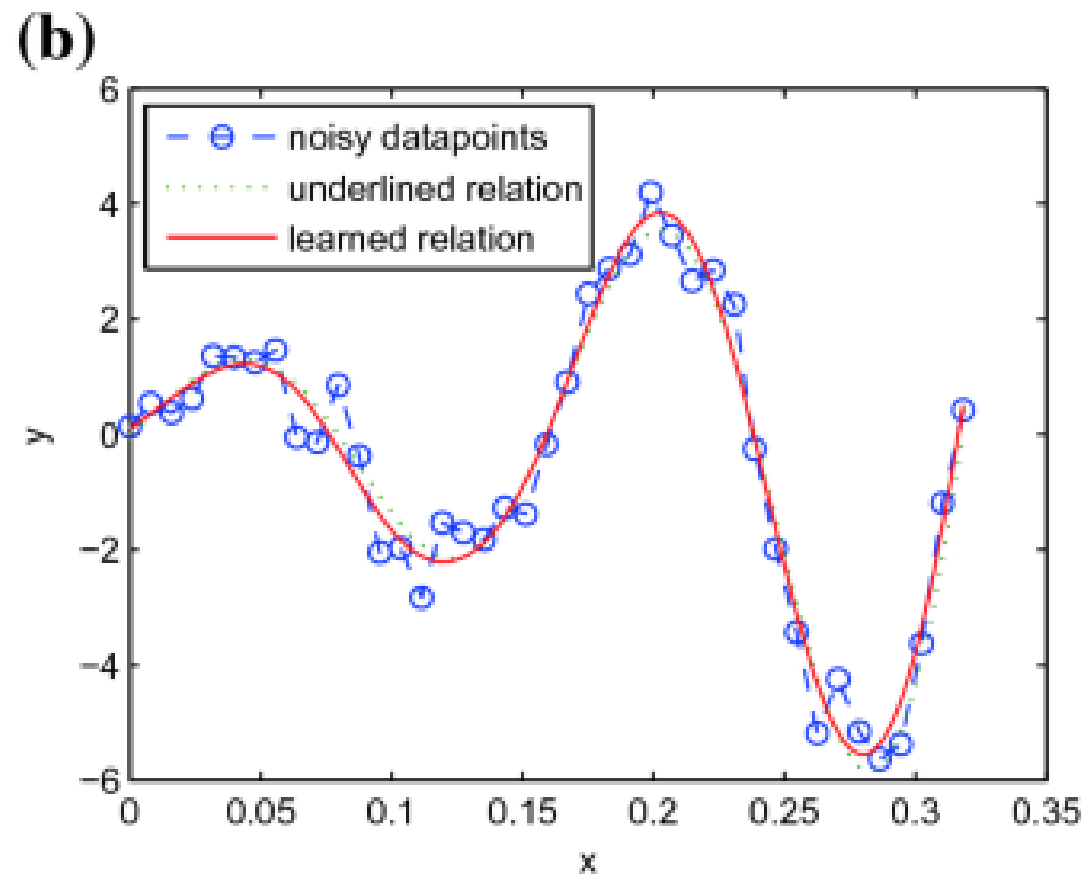
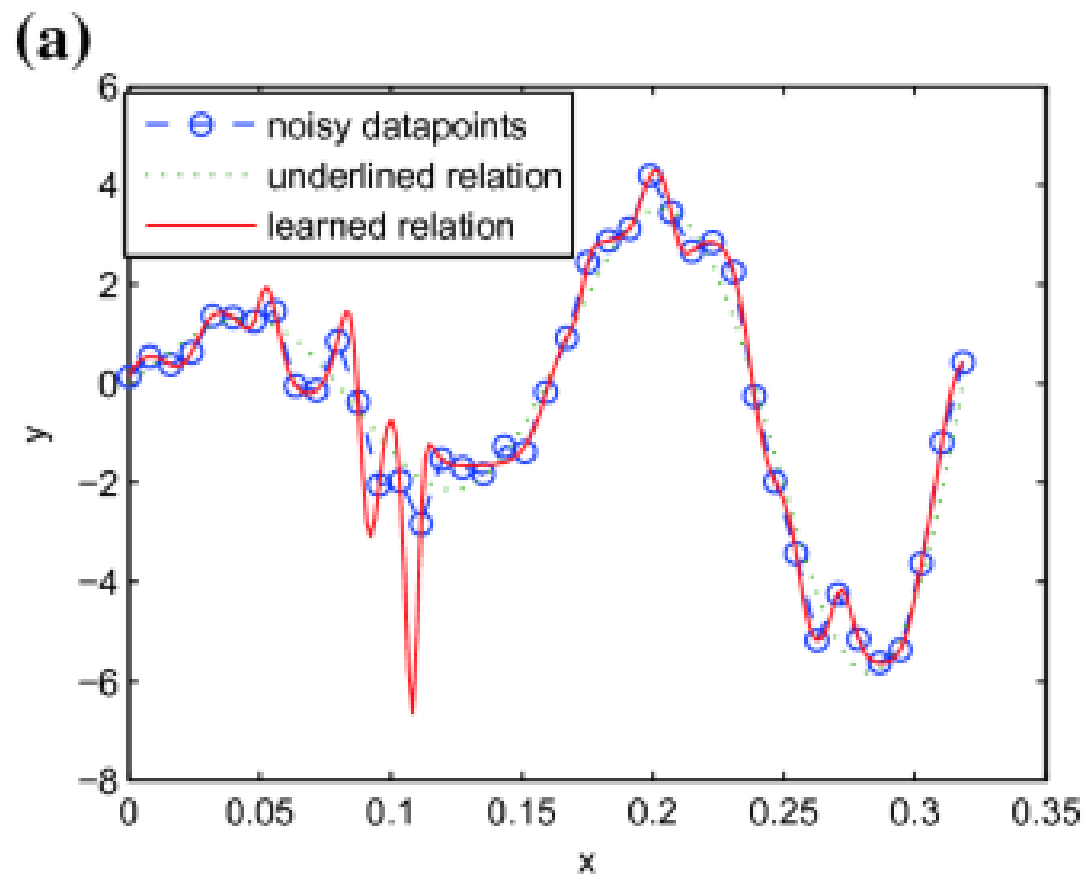


Imagen tomada de [An Introduction to Multi-layer Perceptron and Artificial Neural Networks with Python — DataSklr](#)

Perceptrón multicapa (MLP)

Como regresor universal, el perceptrón puede ser entrenado para resolver problemas de clasificación no lineal.

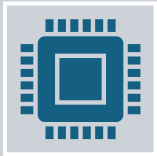




Limitaciones del MLP en tareas de visión de máquinas

Imagen tomada de [Multilayer Perceptrons: Architecture and Error Backpropagation | SpringerLink](#)

Limitaciones del MLP en tareas de visión de máquinas



El rendimiento de los algoritmos de aprendizaje de máquinas depende de la cantidad y la calidad de datos que se tengan para entrenamiento.



Muy pocos datos, o de poca calidad, conllevan a modelos sobreajustados y poco robustos.

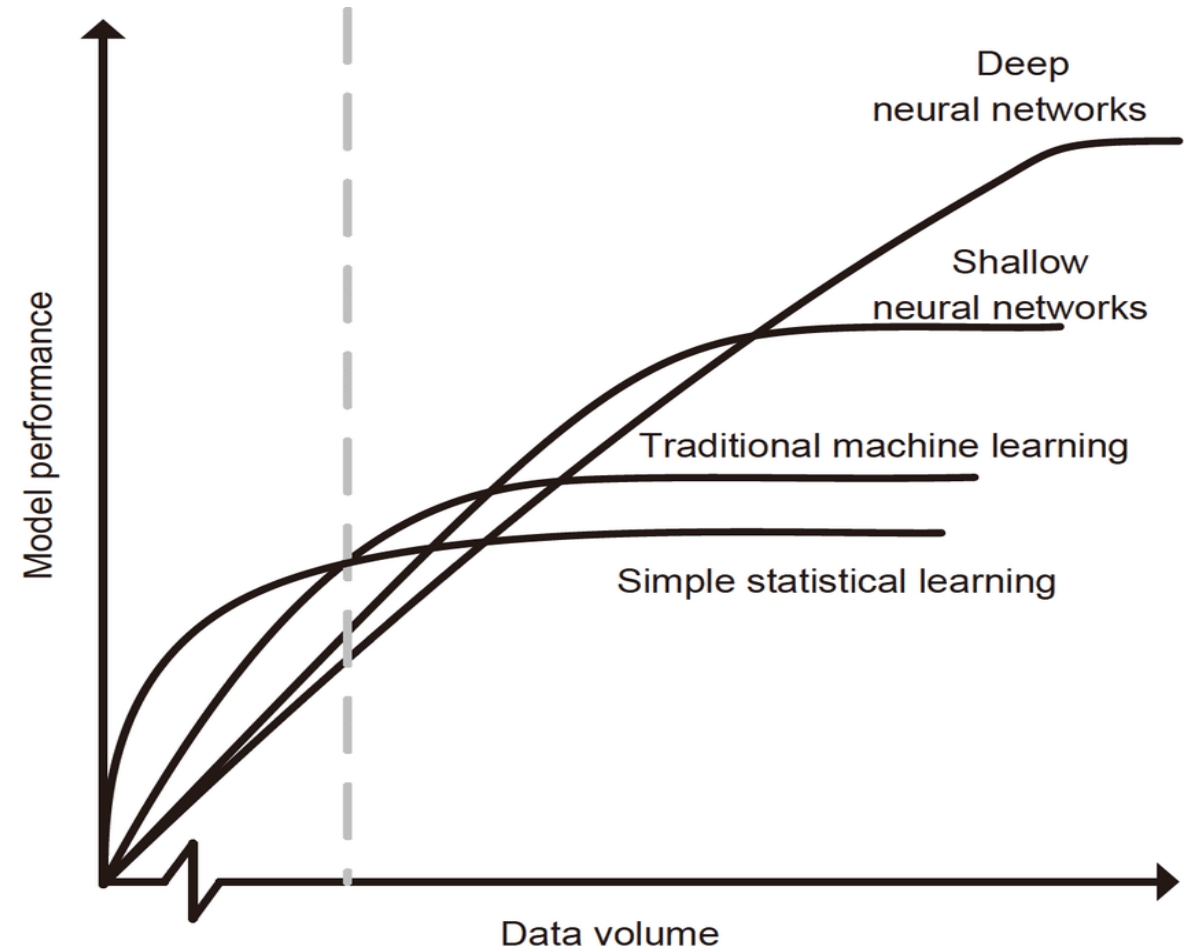
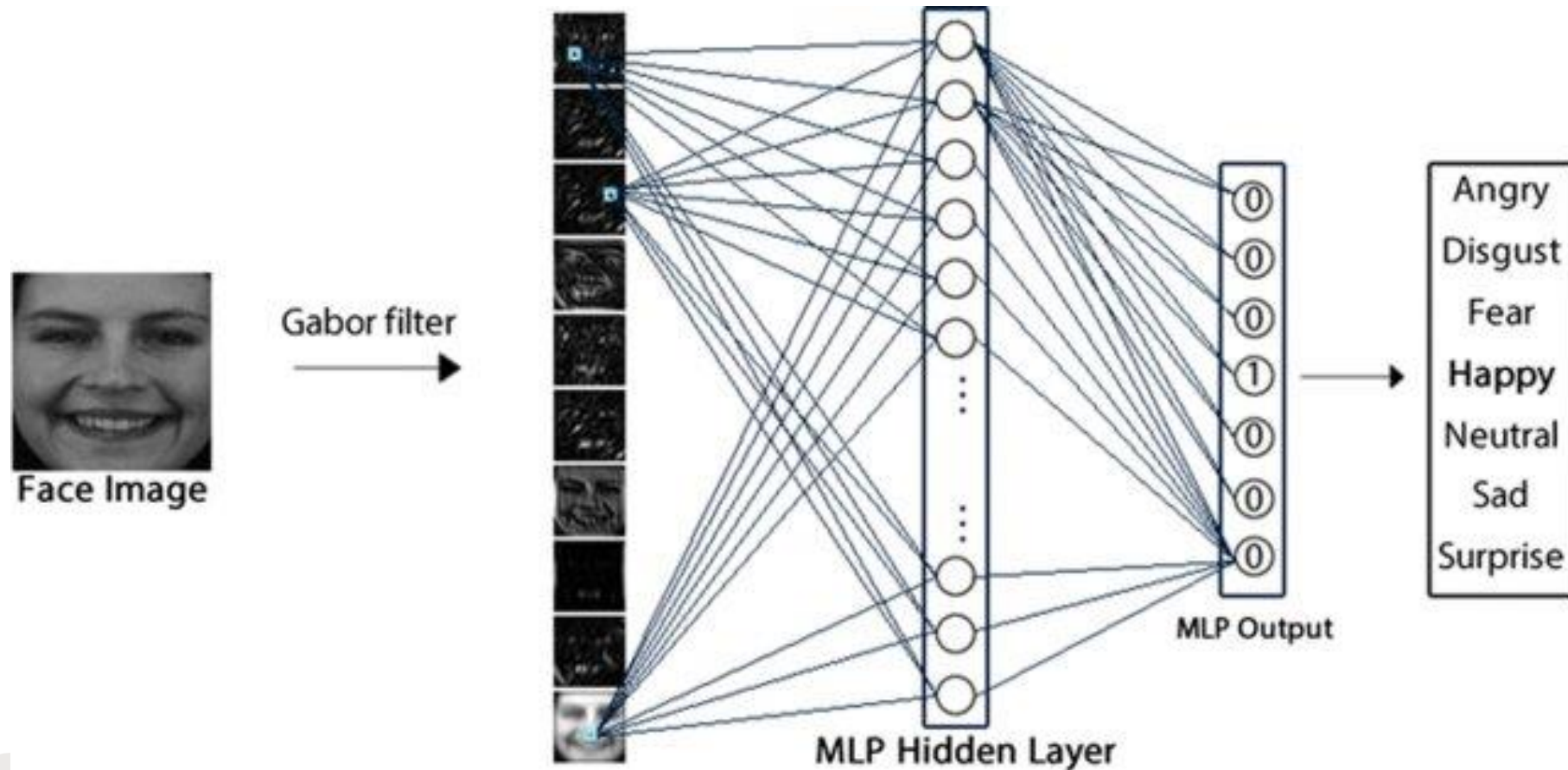


Imagen tomada de [\(PDF\) Application of deep learning in ecological resource research: Theories, methods, and challenges](#)

Imagen tomada de [\(PDF\) A hybrid deep learning neural approach for emotion recognition from facial expressions for socially assistive robots](#)



Limitaciones del MLP en tareas de visión de máquinas

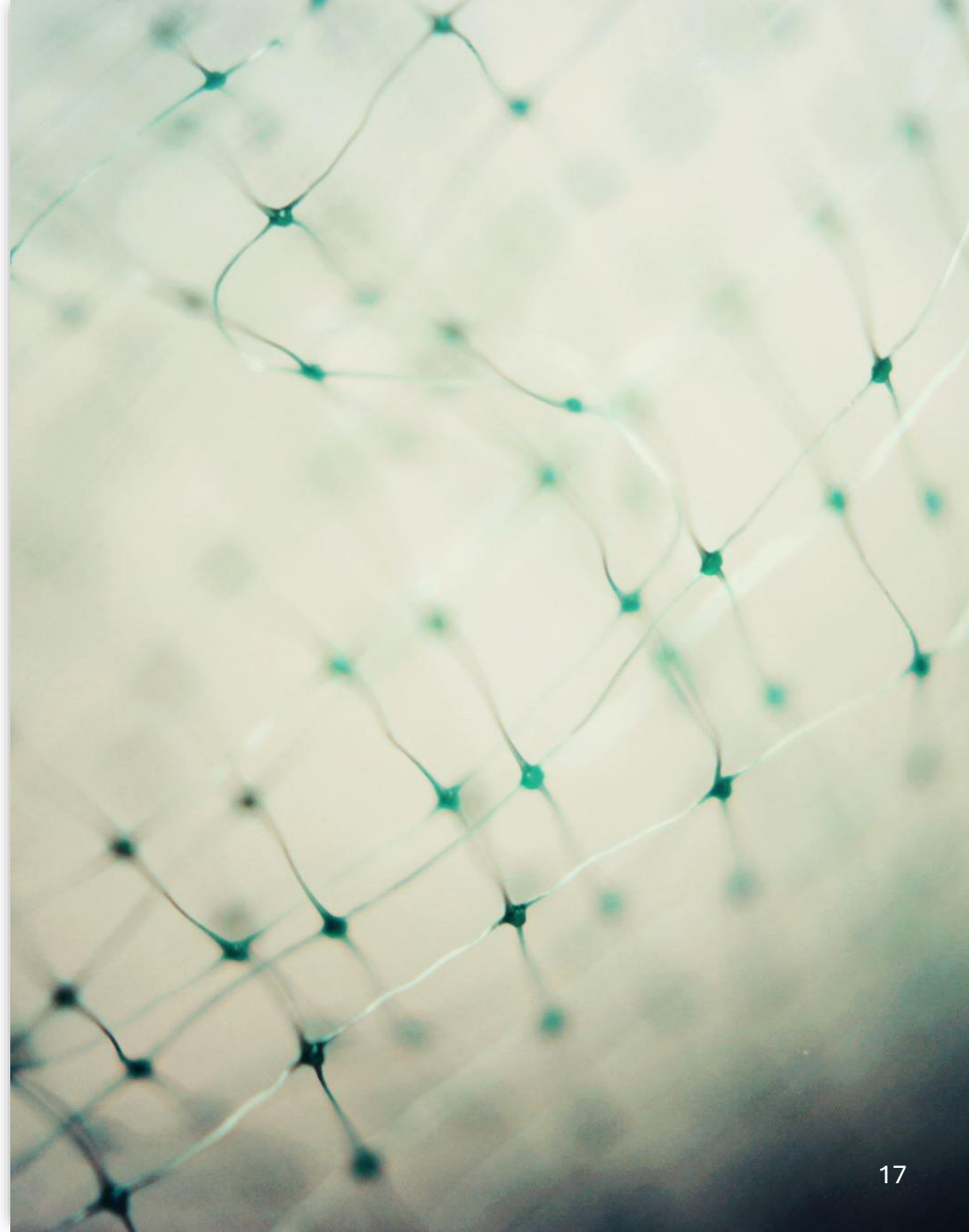
Takeaways

- Los MLPs, como aproximadores universales, tienen la capacidad de aproximar casi cualquier relación entrada-salida entre los datos.
- Sin embargo, la cantidad de parámetros requeridos para la aproximación crece de manera significativa en la medida que se incrementa el número de neuronas, sobre todo en las capas ocultas. Eso afecta el número de muestras necesarias para la aproximación.
- Los MLPs no son robustos frente a operaciones típicas en imágenes, como las transformaciones rígidas y de intensidad/color.

Redes neuronales convolucionales (CNNs)

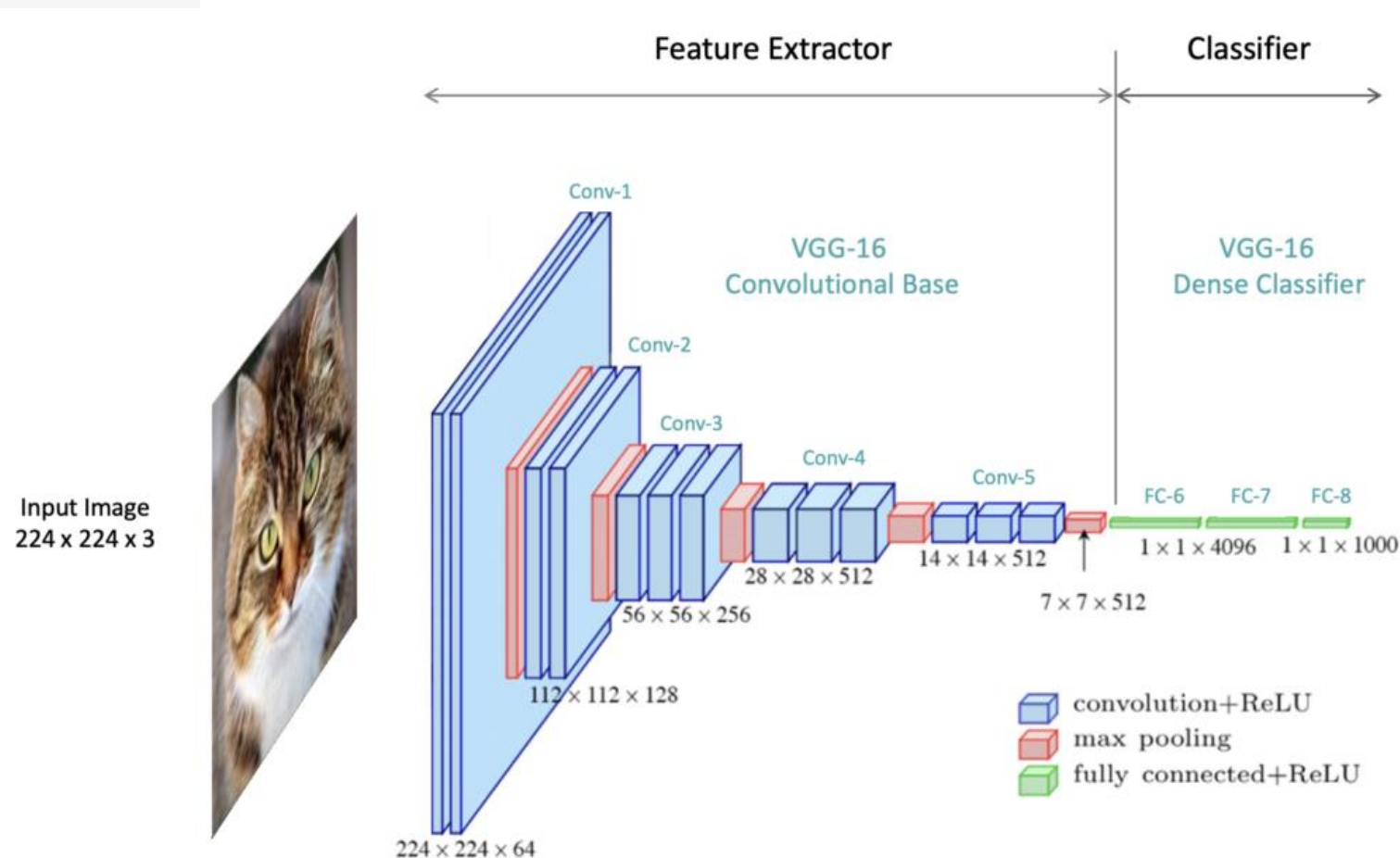
Una red neuronal que trabaja con imágenes puede beneficiarse de incluir, de algún modo, la naturaleza propia de las imágenes. Esto es:

- El hecho de que es una rejilla o tensor;
- La información de vecindad entre los píxeles da un sentido de estructura;
- El hecho de que la información contenida se obtiene a partir de procesos de sensado (habitualmente);
- Ciertas características propias de la imagen están relacionadas con el tipo de contenido.





Redes neuronales convolucionales (CNNs)

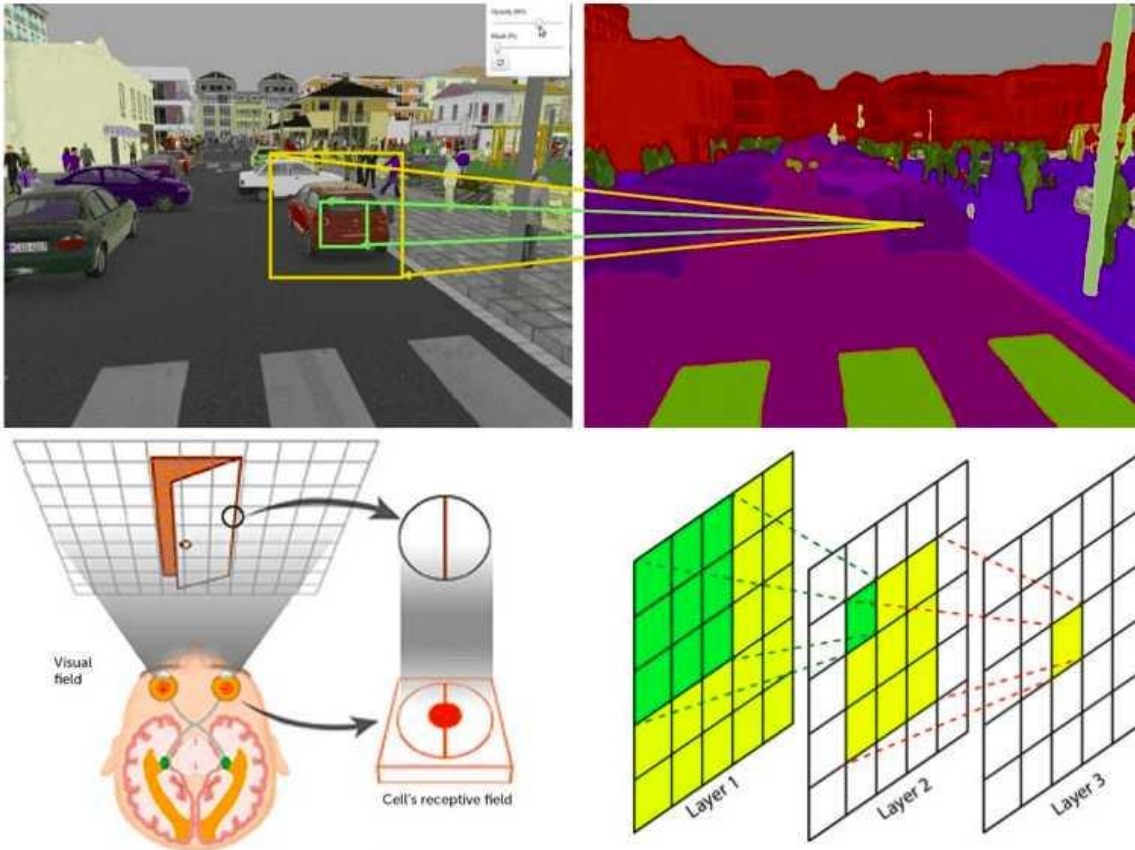


Arquitectura de una CNN

Las redes convolucionales tienen varios tipos de operaciones apilados en capas:

- Convoluciones;
- Reducciones de dimensión del espacio de entrada;
- Decisión, generalmente realizada a través de un MLP.

Arquitectura de una CNN



- Una de las claves de las CNNs es el poder enfocar ciertas partes de la imagen (o ciertas características) en una o en grupos de neuronas.
- La operación que se realiza en cada neurona de las capas que reciben la imagen es una convolución.

Campo perceptual de una neurona

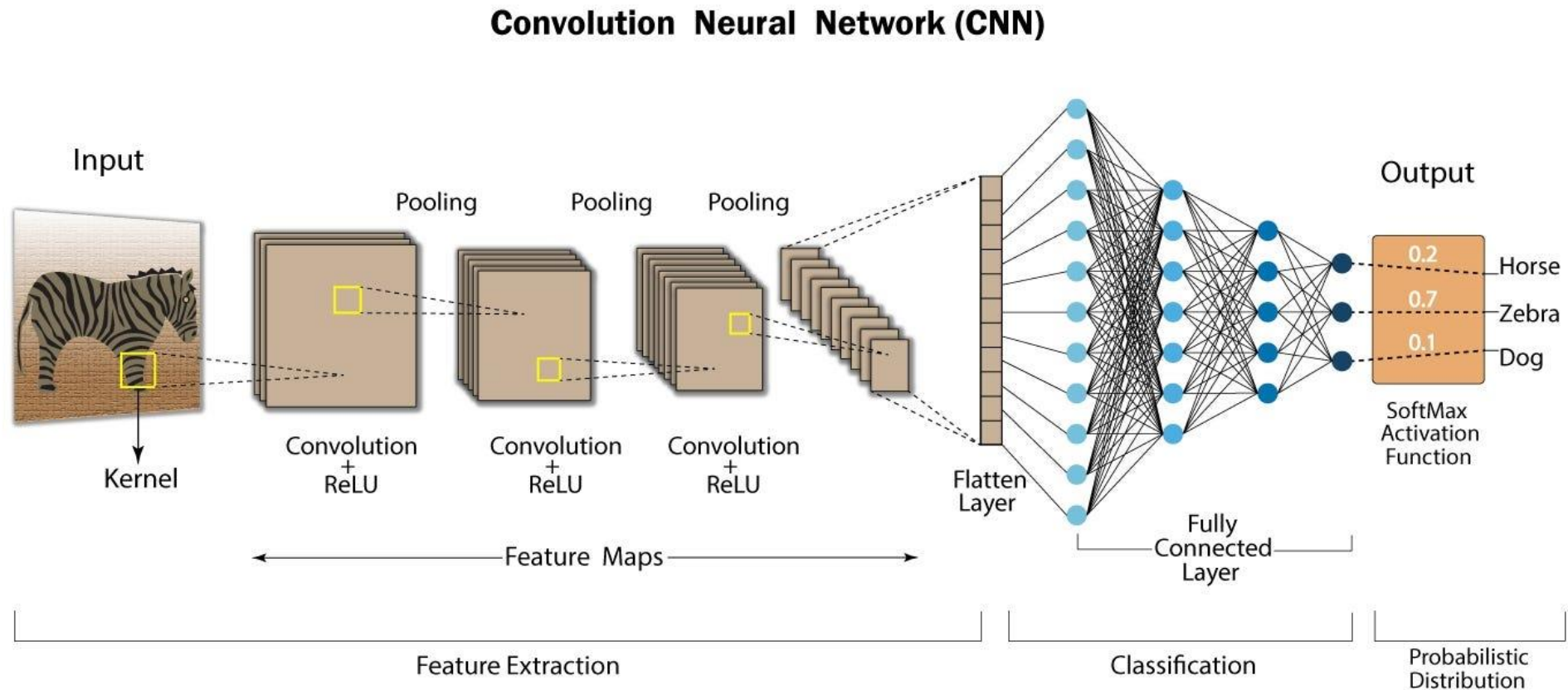
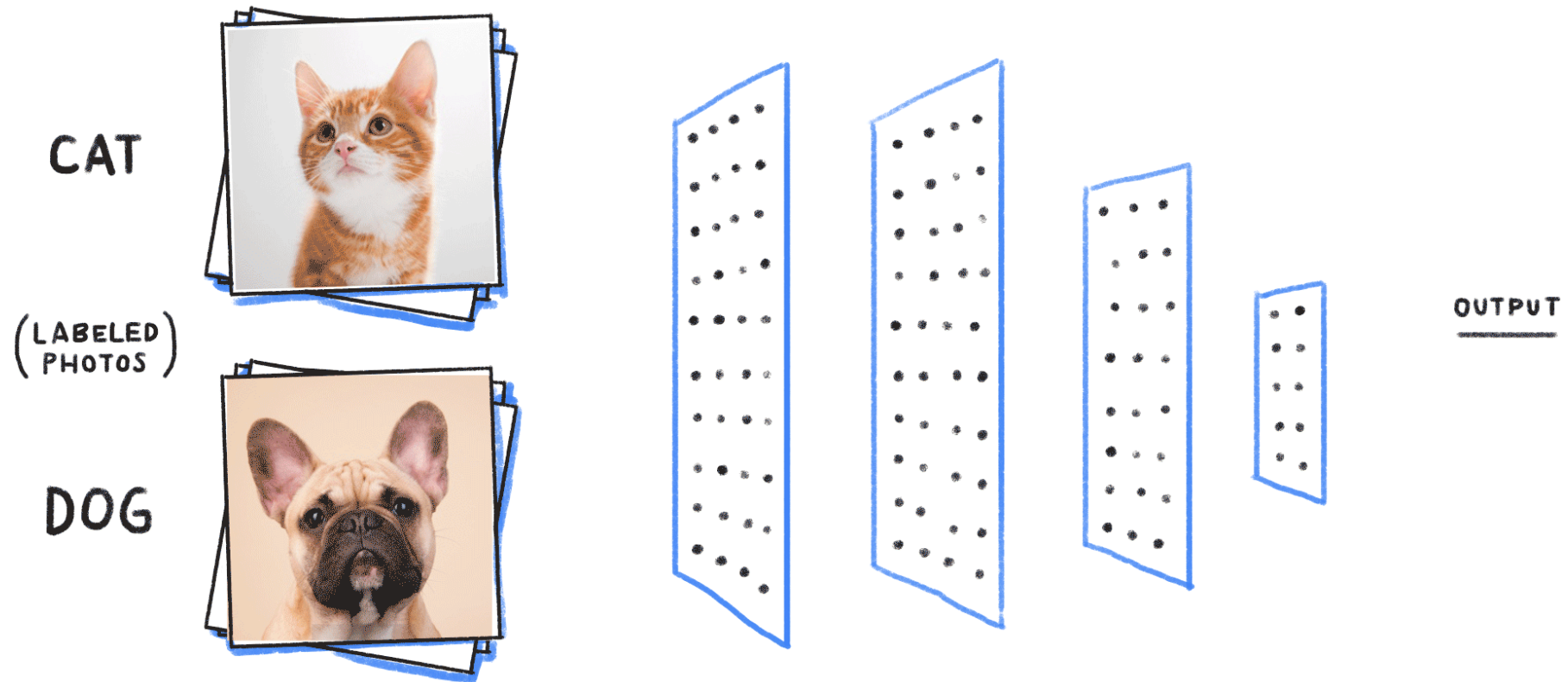


Imagen tomada de [Convolutional Neural Networks. Hello, and thank you for taking an... | by Nishad Ahamed | Dev Genius](#)

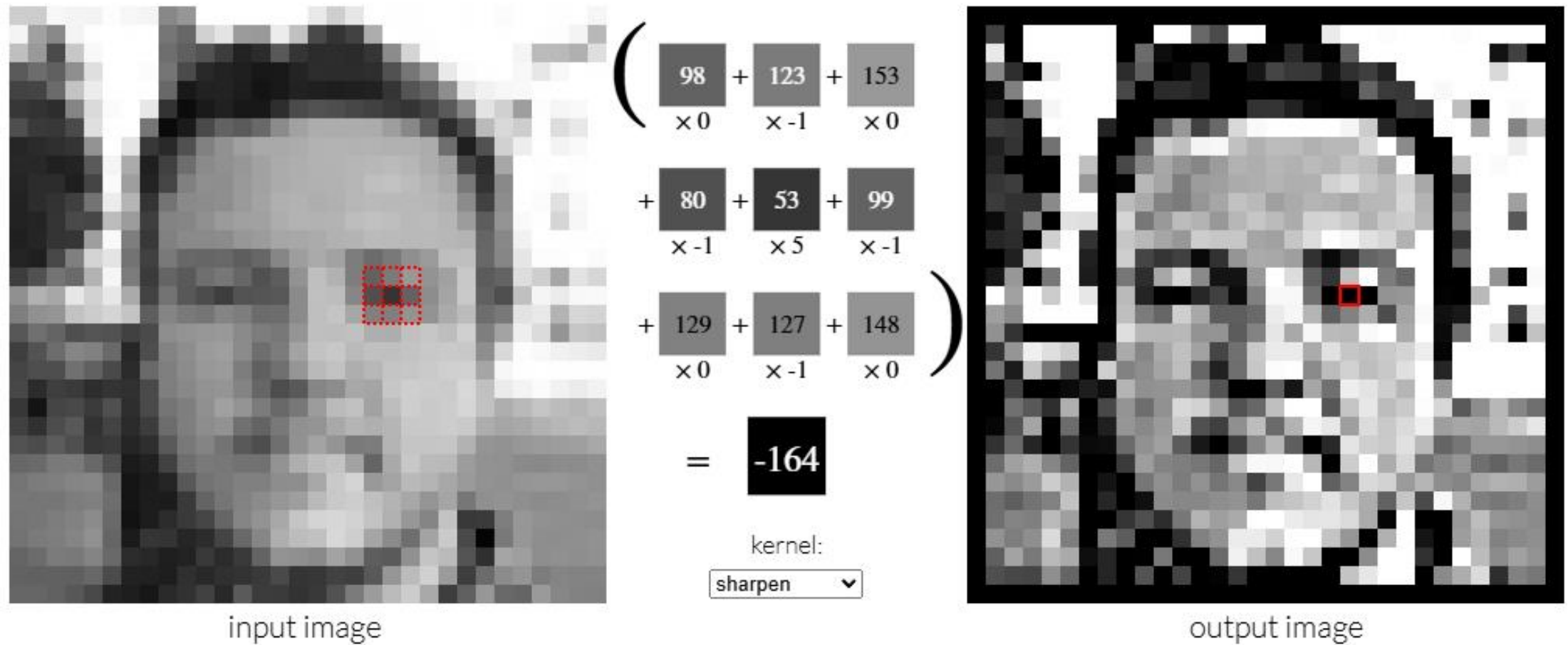
Campo perceptual



Tomada de [What the Hell is Perceptron?. The Fundamentals of Neural Networks | by SAGAR SHARMA | Towards Data Science](#)

Efecto convolucional

Imagen tomada de [Image Kernels explained visually](#)

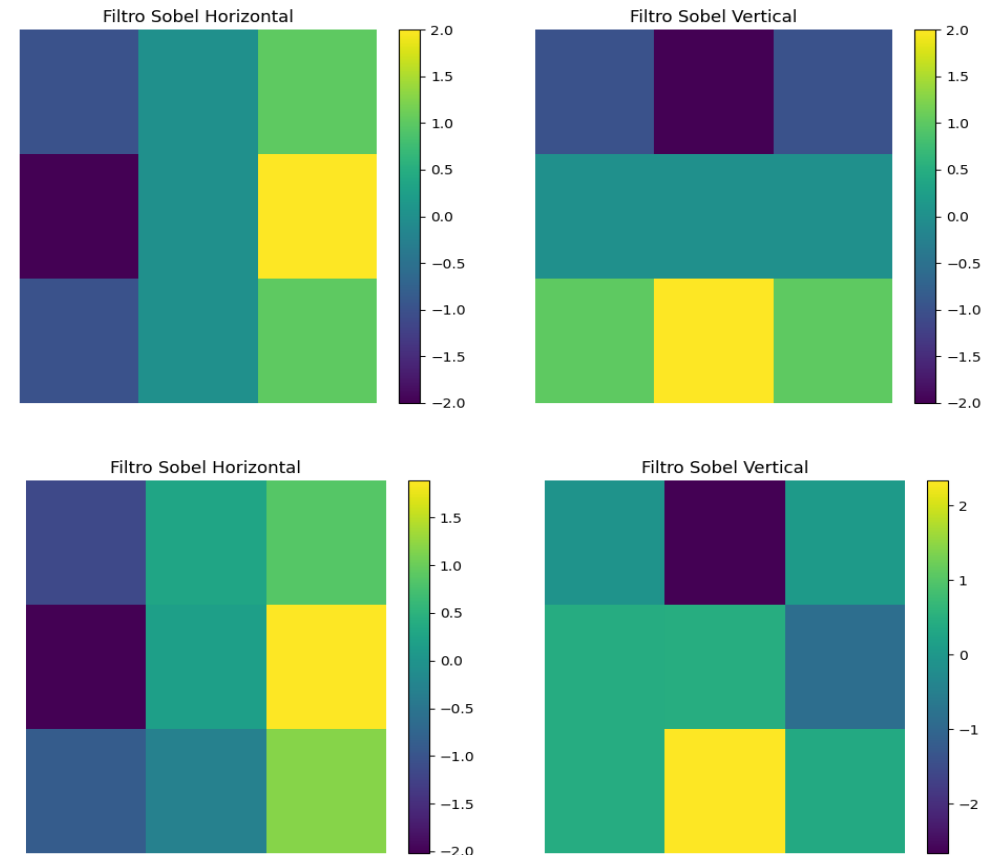
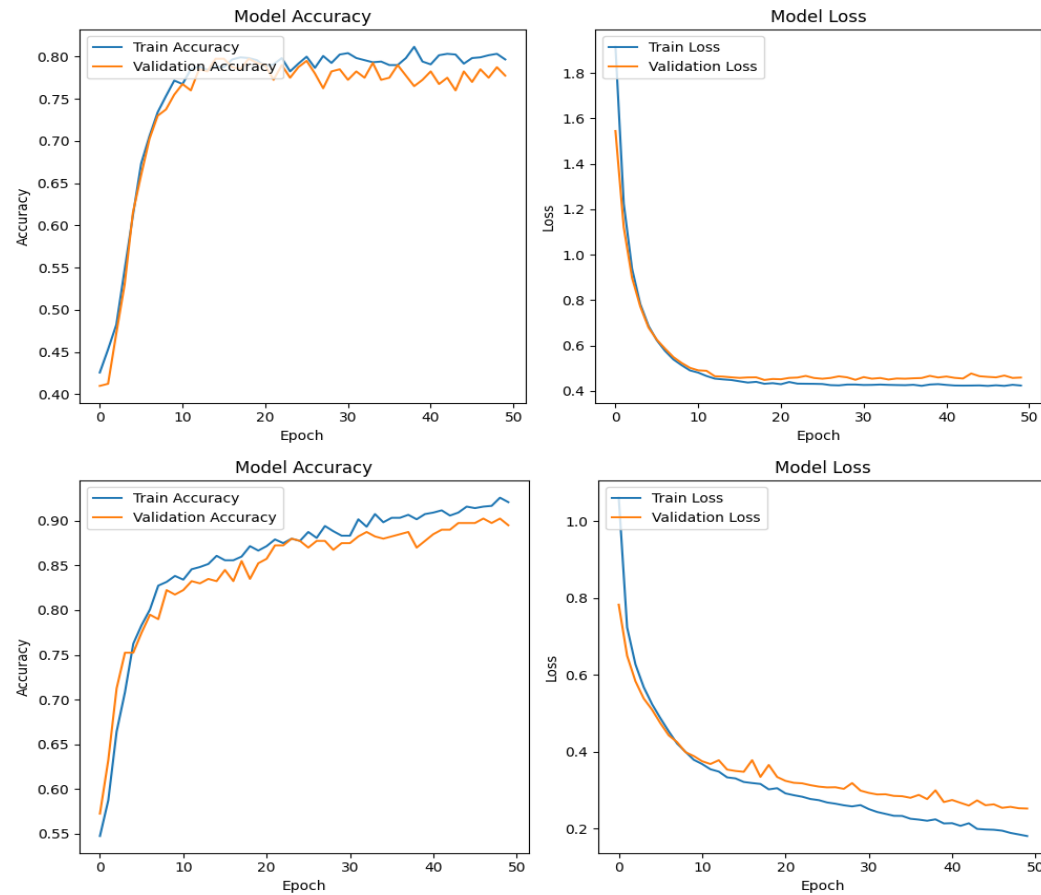


Efecto convolucional



- Los filtros de la primera capa convolucional son aplicados directamente sobre las imágenes de entrada.
- Si la imagen está representada en RGB, cada filtro tendrá también las tres componentes respectivas.

Efecto convolucional



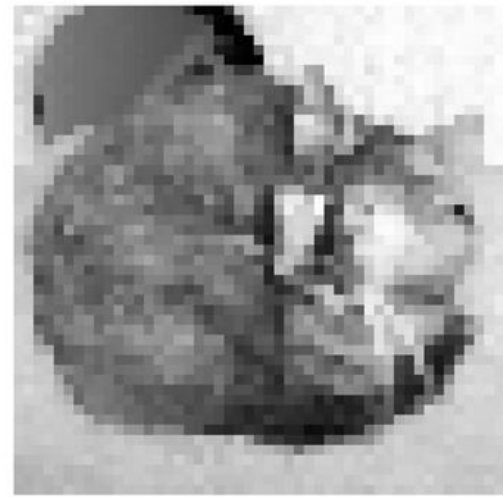
(446, 450)



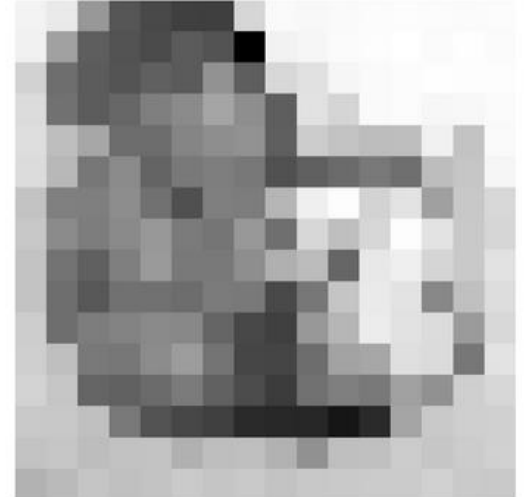
(148, 150)



(49, 50)



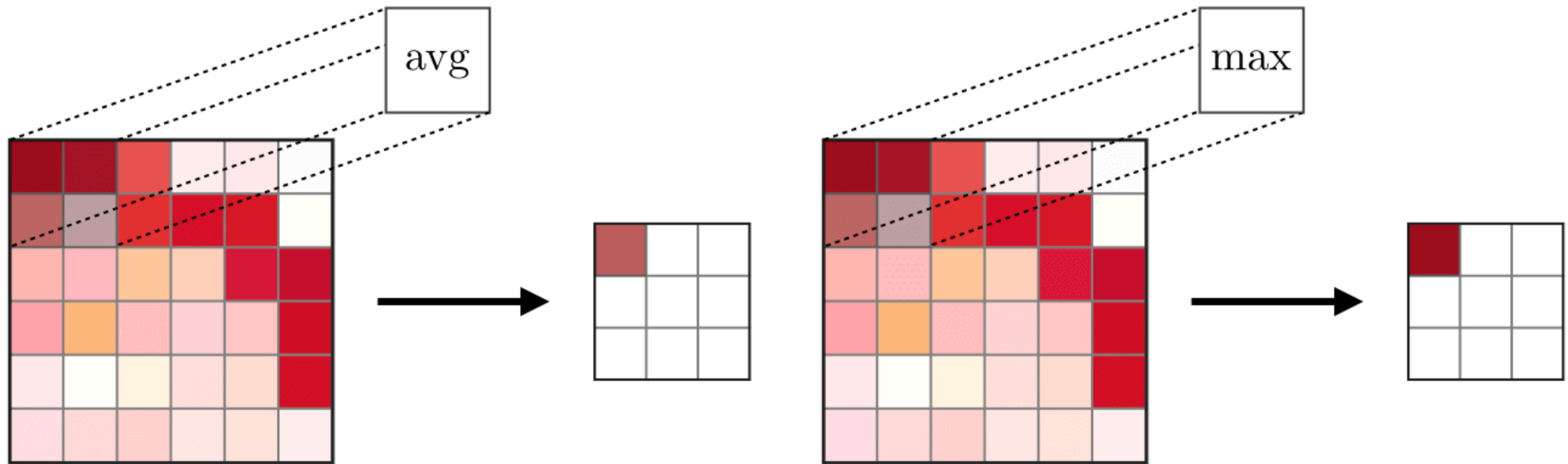
(16, 16)



Efecto del pooling en la selección de características

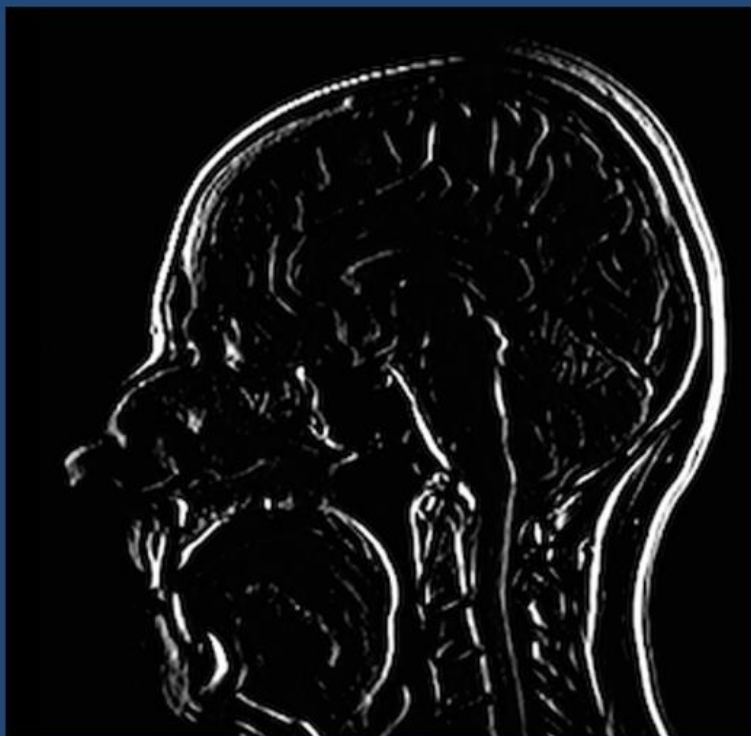
Imagen tomada de [Pooling In Convolutional Neural Networks | DigitalOcean](#)

Efecto del pooling en la selección de características

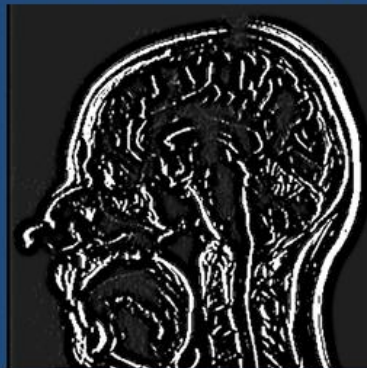


Imágenes tomadas de [CS 230 - Convolutional Neural Networks Cheatsheet](#)

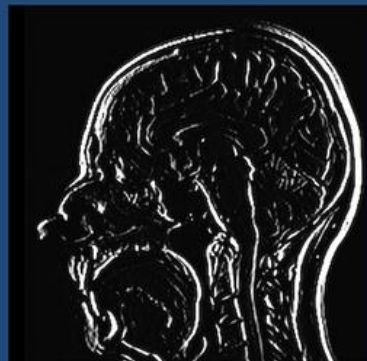
Rectified Feature Map



2x2 Max Pooling

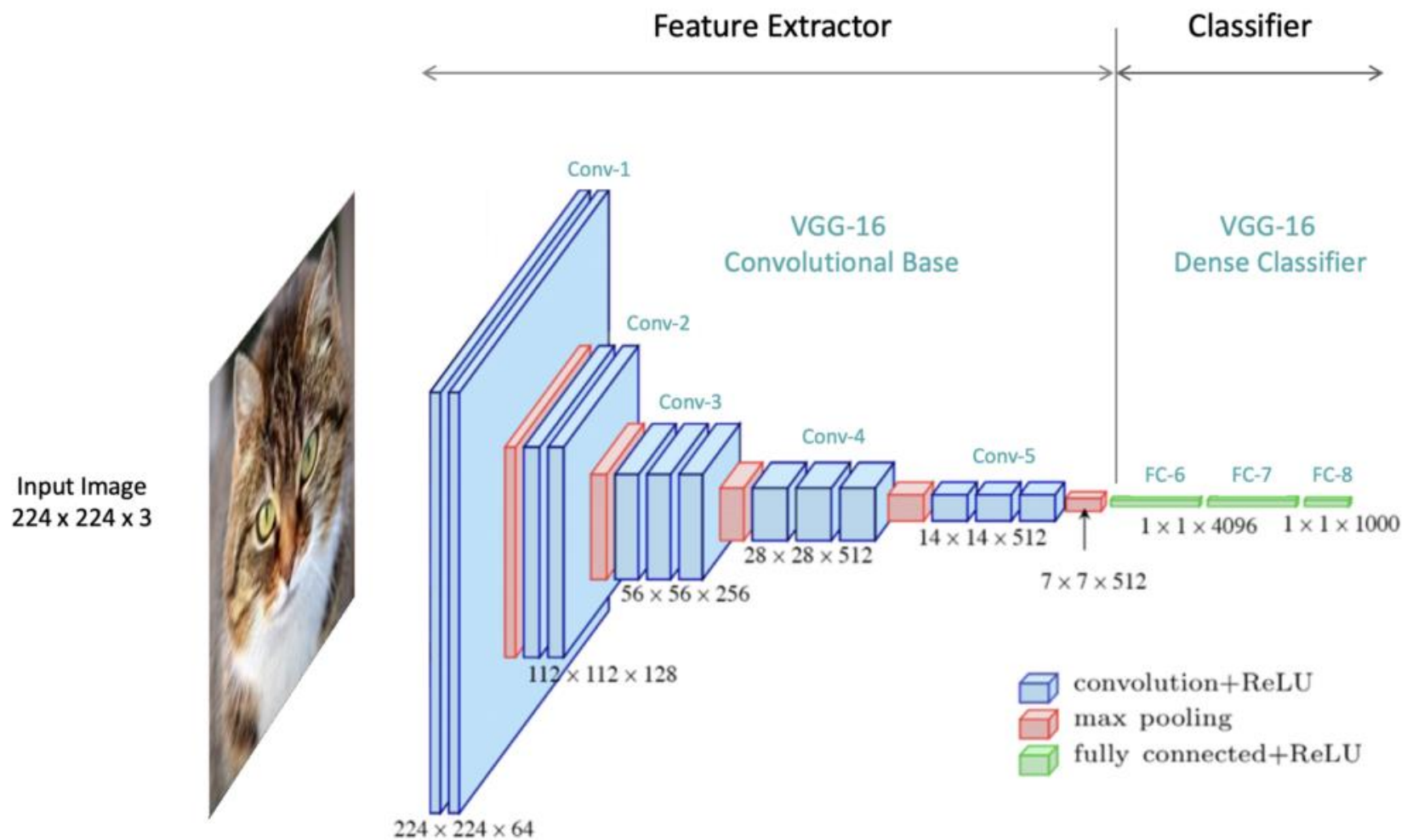


2x2 Avg Pooling




Efecto del pooling en la selección de características

Imagen tomada de [Convolutional Neural Network - Questions and Answers in MRI](#)



Arquitectura de una CNN



Ventajas de las CNNs en tareas de visión de máquinas

- Las CNNs tienen la capacidad de enfocarse en ciertas características presentes en las imágenes, teniendo en cuenta elementos como la estructura, intensidad o color.
- El hecho de que las capas convolucionales compartan un mismo filtro para todas las neuronas de la capa reduce la cantidad de parámetros del modelo. Adicionalmente, los núcleos de convolución (kernel) son interpretables.
- La combinación de capas convolucionales y de pooling hacen que la estructura de la CNN sea hasta cierto punto robusta frente a transformaciones de la información de entrada.

MobileNet

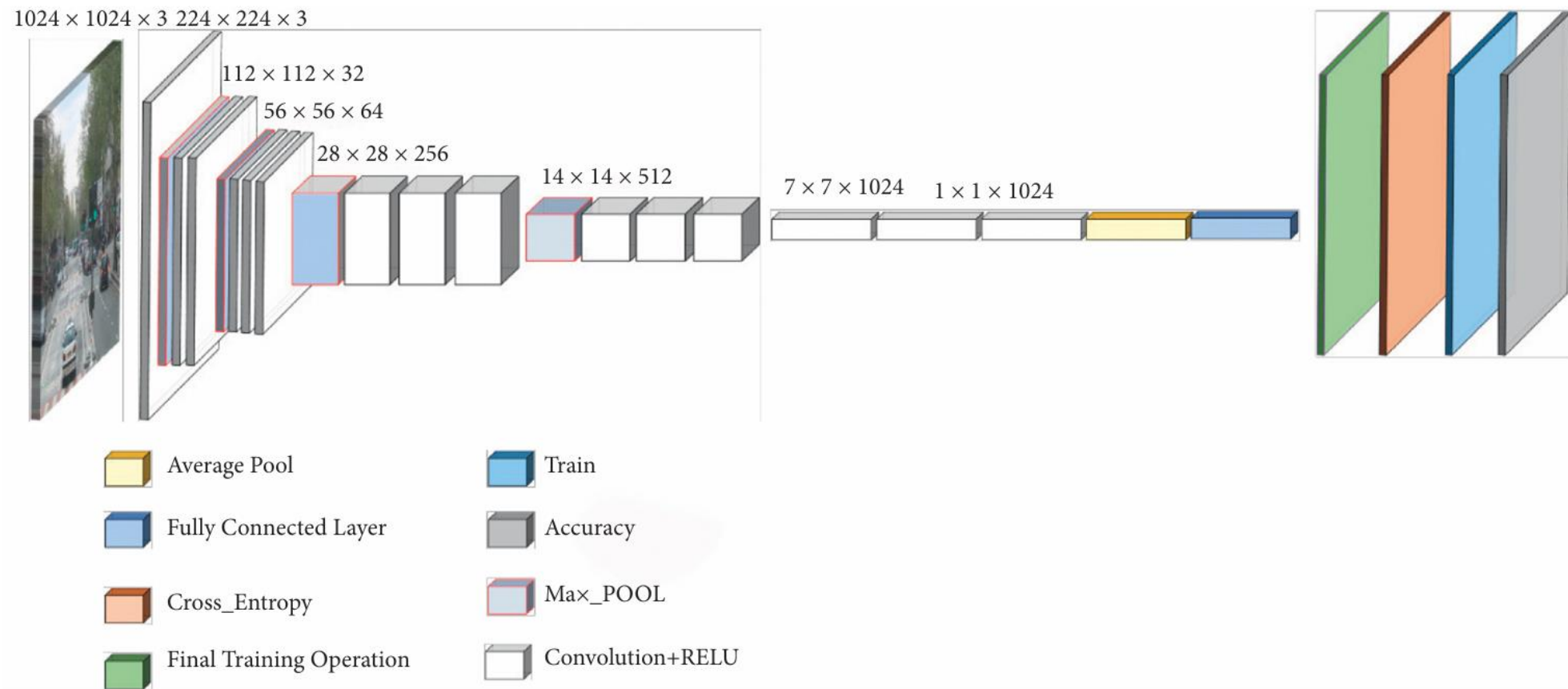
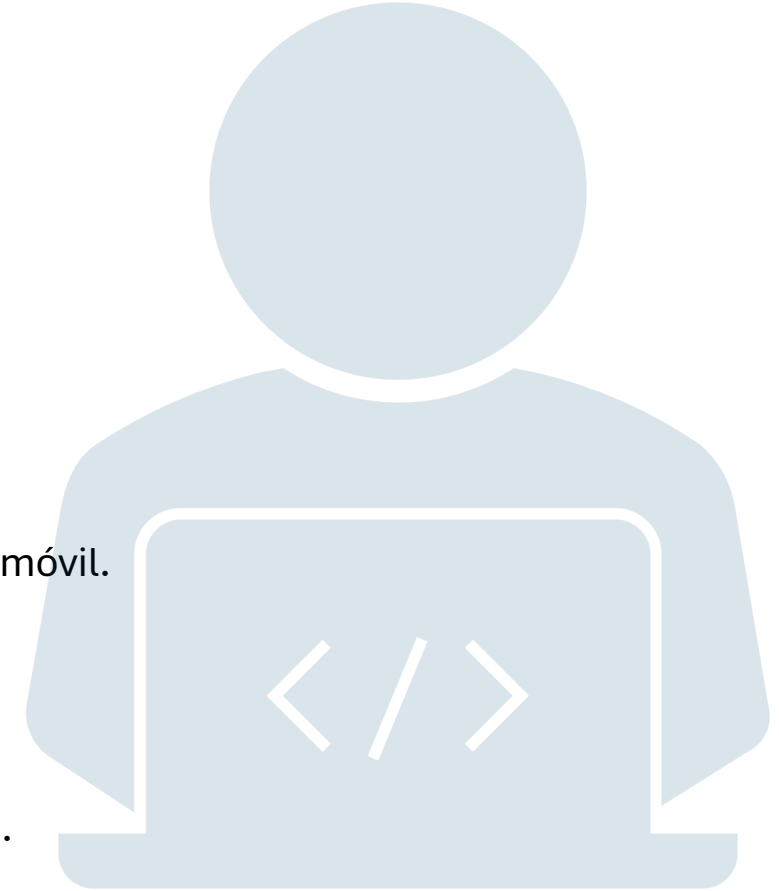


Imagem tomada de [\(PDF\) Efficient Approach towards Detection and Identification of Copy Move and Image Splicing Forgeries Using Mask R-CNN with MobileNet V1](#)

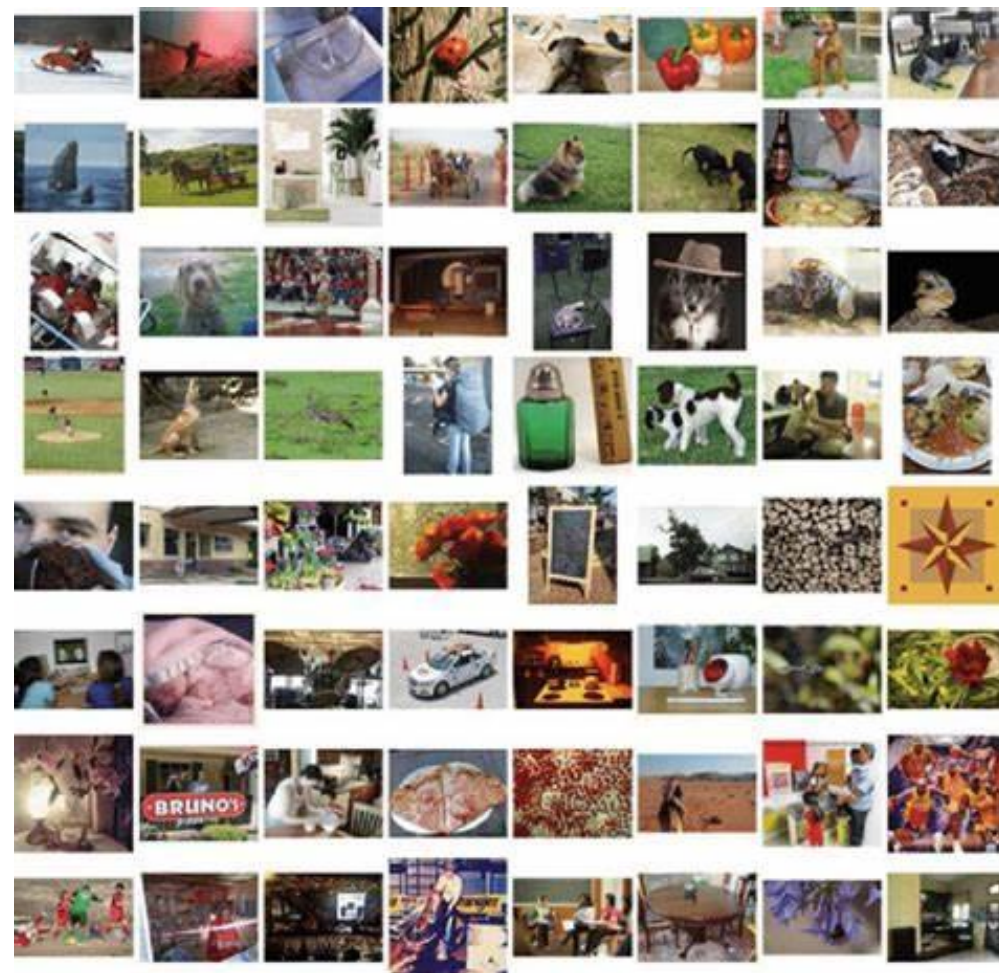


MobileNet

- Fue el primer modelo de TensorFlow para computación móvil.
- Usa convoluciones separables para simplificar las operaciones.
- Es propiedad de Google, y es de código abierto.
- Usable para computación en el borde (Edge computing).
- Está en el orden de millones de parámetros.



MobileNet



MobileNet

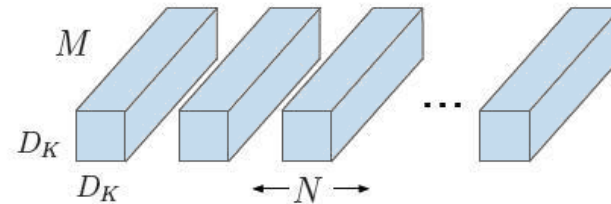
Table 1. MobileNet Body Architecture

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
5 ×	Conv dw / s1	$3 \times 3 \times 512$ dw
	Conv / s1	$1 \times 1 \times 512 \times 512$
	Conv dw / s2	$3 \times 3 \times 512$ dw
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

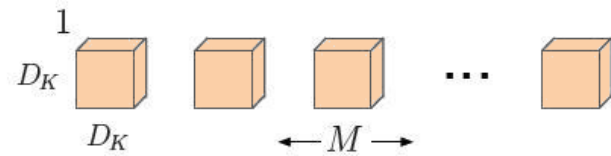
README

Pytorch MobileNet v1

A Pytorch Implementation of the 2017 paper ["MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications"](#) from Google.

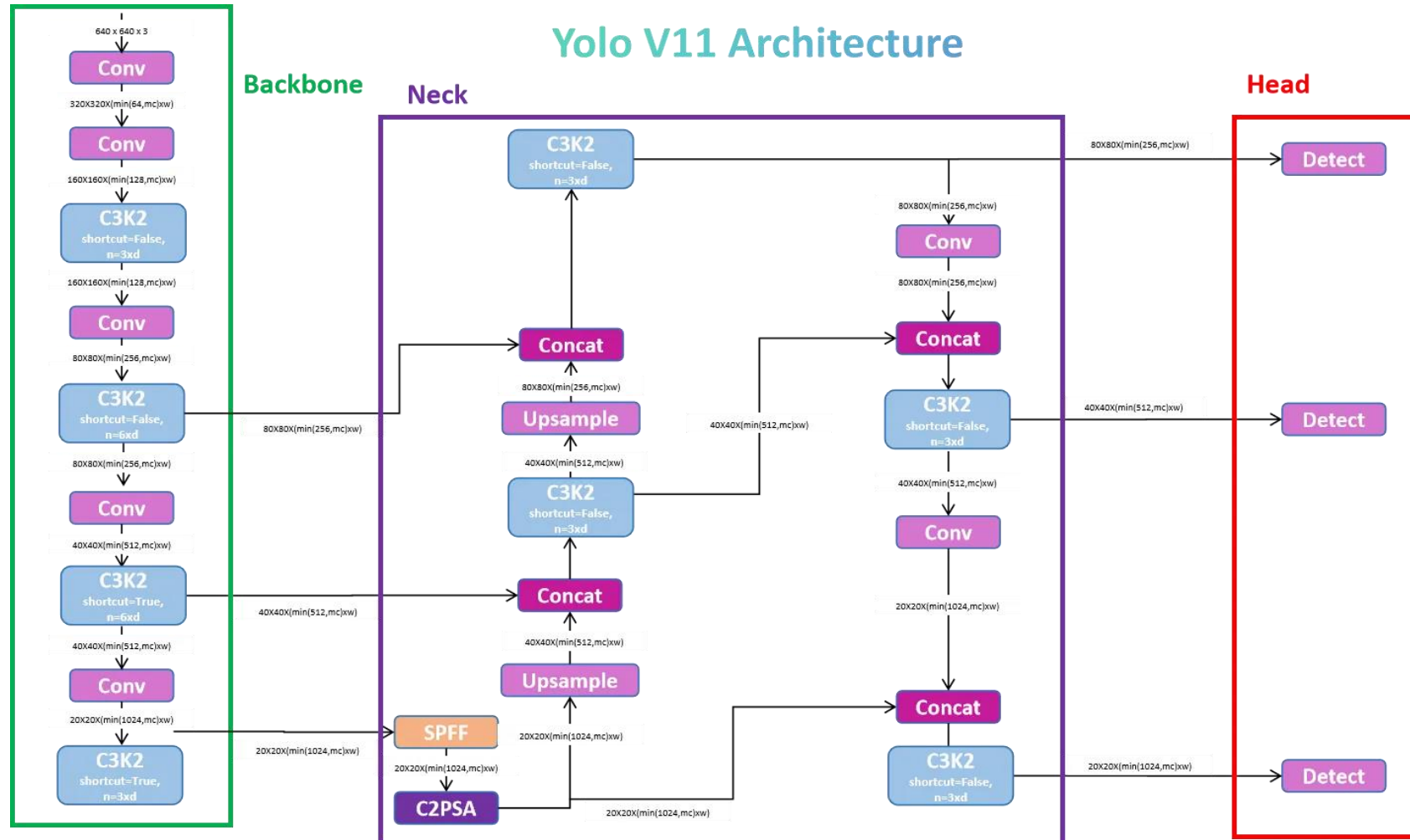


(a) Standard Convolution Filters



(b) Depthwise Convolutional Filters

YOLO (You only look once)



Tomada de [YOLOv11 Architecture Explained: Next-Level Object Detection with Enhanced Speed and Accuracy](#) | by S Nikhileswara Rao | Oct, 2024 | Medium

YOLO (You only look once)

YOLO es una estructura que permite la detección de objetos en imágenes, proveyendo sus cajas delimitadoras, y las probabilidades de pertenencia a las clases correspondientes.

La primera versión fue publicada en 2016.

Ya van 11 versiones de la estructura, con mejoras incrementales.

Usado en múltiples implementaciones/aplicaciones comerciales y libres, así como [playgrounds en línea](#).

YOLO (You only look once)



A pesar de ser una arquitectura relativamente simple (en cuanto al número de parámetros), resulta ser más costosa que MobileNet.



Dado a que está muy difundida, es fácil encontrar información acerca de cómo integrarla en aplicaciones propietarias.



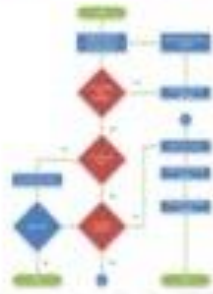
Se puede usar como base para hacer aprendizaje por transferencia y destilado de modelos.

Discusión acerca del uso de redes neuronales

- Las redes neuronales son aproximadores universales que pueden ser usados en diversos tipos de problemas.
- No son la mejor alternativa cuando el problema puede ser claramente formalizable.

The Three Waves of AI

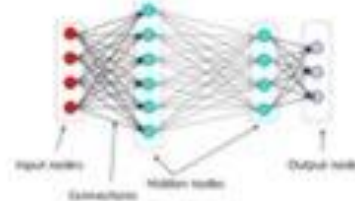
First Wave – Traditional Programming



Wave 1

Hand-Crafted
Programming/ Logic
GOFAI

Second Wave – Neural Nets - Big Data



Wave 2

Statistical
Machine Learning
Generative AI

Third Wave – Cognitive Architecture



Wave 3

Adaptive. Contextual
Autonomous
Cognitive AI

Discusión acerca del uso de redes neuronales

Imagen tomada de [\(PDF\) Why We Don't Have AGI Yet](#)

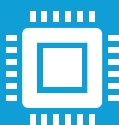
Discusión acerca del uso de redes neuronales



Es importante dedicar algo de tiempo a estudiar la naturaleza de los datos y del problema a resolver antes de escoger una arquitectura.



El entrenamiento de las redes neuronales profundas (CNNs, residuales, Transformers, generativas, etc.) suele ser muy costoso desde el punto de vista computacional y energético.



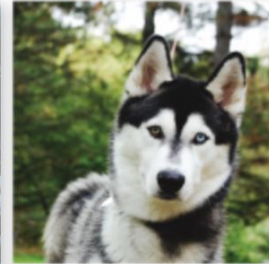
Los modelos grandes no se pueden implementar en el borde, y a veces ni siquiera en PCs de escritorio.

Discusión

Explain the Prediction



Predicted: **Wolf**
True: **Wolf**



Predicted: **Husky**
True: **Husky**



Predicted: **Husky**
True: **Husky**



Predicted: **Wolf**
True: **Wolf**



Predicted: **Wolf**
True: **Wolf**



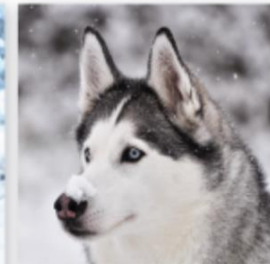
Predicted: **Wolf**
True: **Wolf**



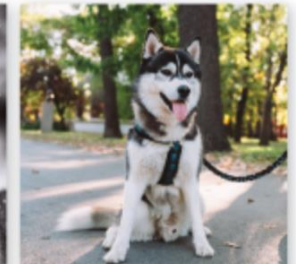
Predicted: **Husky**
True: **Wolf**



Predicted: **Wolf**
True: **Wolf**



Predicted: **Wolf**
True: **Husky**



Predicted: **Husky**
True: **Husky**

Discusión

input image



input image



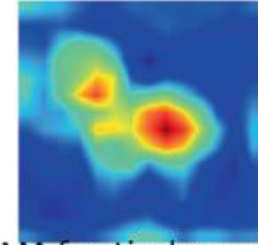
input image



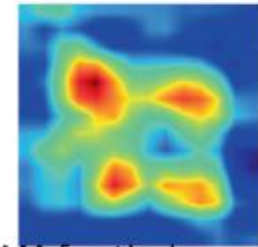
input image



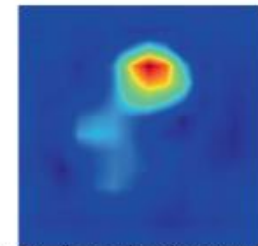
Grad-CAM for coyote (272)



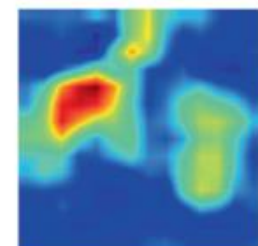
Grad-CAM for timber wolf (269)



Grad-CAM for timber wolf (269)



Grad-CAM for timber wolf (269)





Muchas gracias

Contacto: jbgomezm@unal.edu.co