Ciencia y Tecnología
Secretaría de Ciencia, Humanidades, Tecnología e Innovación

CIMAT
UNIDAD MÉRIDA

INTRO. AL DEEP LEARNING (REDES NEURONALES MULTICAPA)

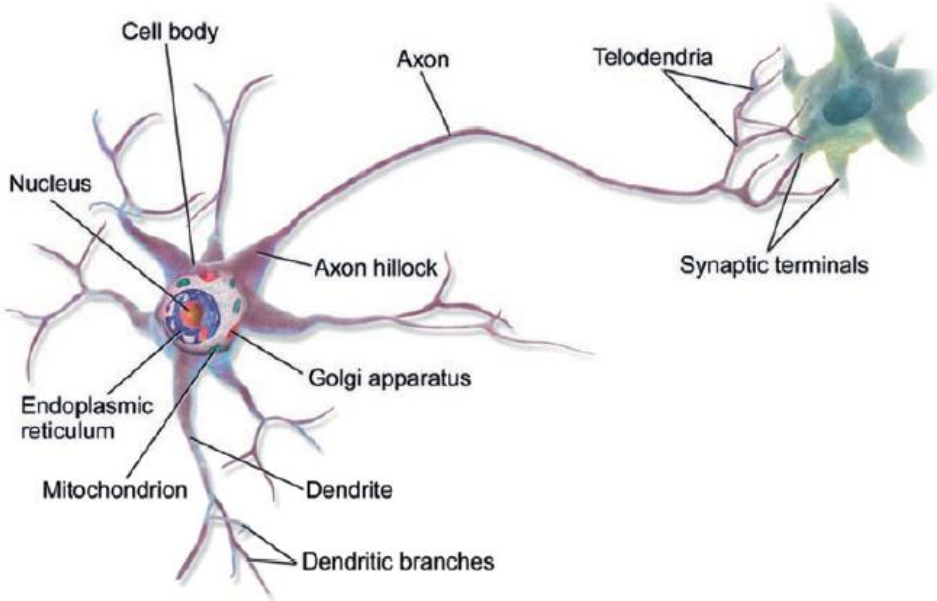
Dr. Francisco J. Hernández López
SECIHTI – CIMAT-Mérida
fcoj23@cimat.mx, www.cimat.mx/~fcoj23

MODELO DE DEEP LEARNING

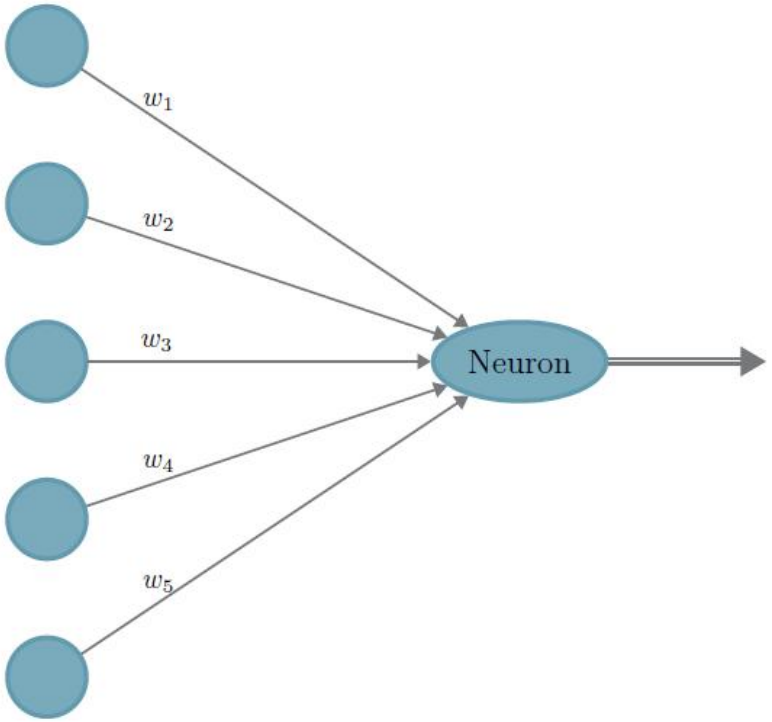
- Un modelo de aprendizaje profundo es una función $y = f_\theta(x)$
  - θ son los parámetros en dimensiones altas.
  - x representa los datos en dimensiones altas.
  - y representa la salida de la función que normalm. es de dim. baja.
- El problema es encontrar los parámetros θ tales que $f_\theta(\cdot)$ satisface aprox. la relación entre x y y.
- Entrenamiento: es el proceso de estimar los parámetros θ.
- Pruebas o inferencia: es el proceso de aplicar la función $f_\theta(\cdot)$ a datos de entrada x no utilizados durante el entrenamiento.

Liquet, B., Moka, S., & Nazarathy, Y. (2024). Mathematical Engineering of Deep Learning (1st ed.). Chapman and Hall/CRC.

UNA NEURONA



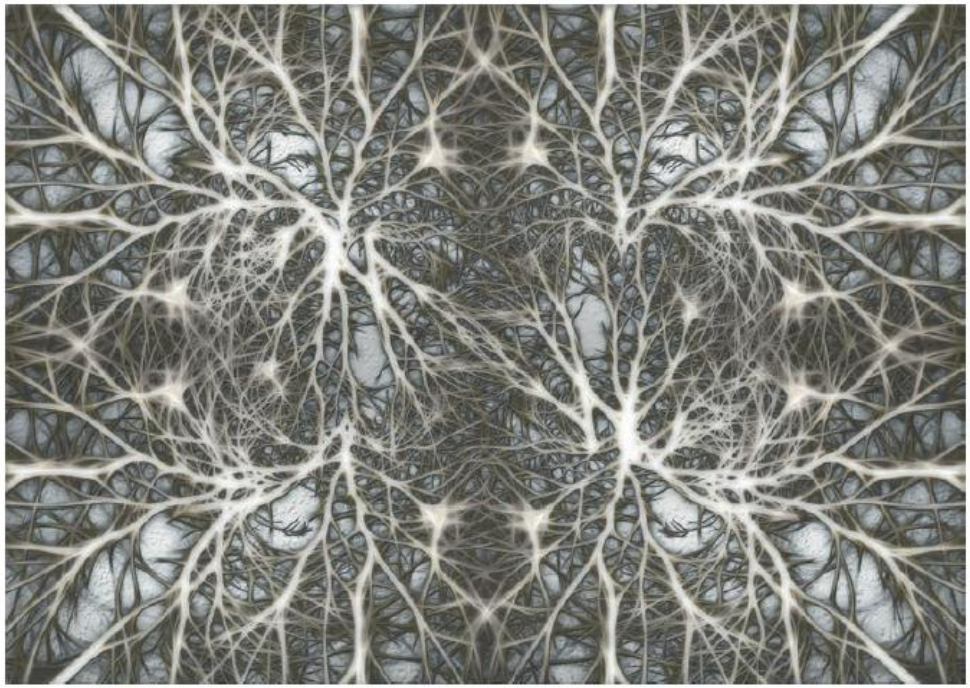
Neurona biológica



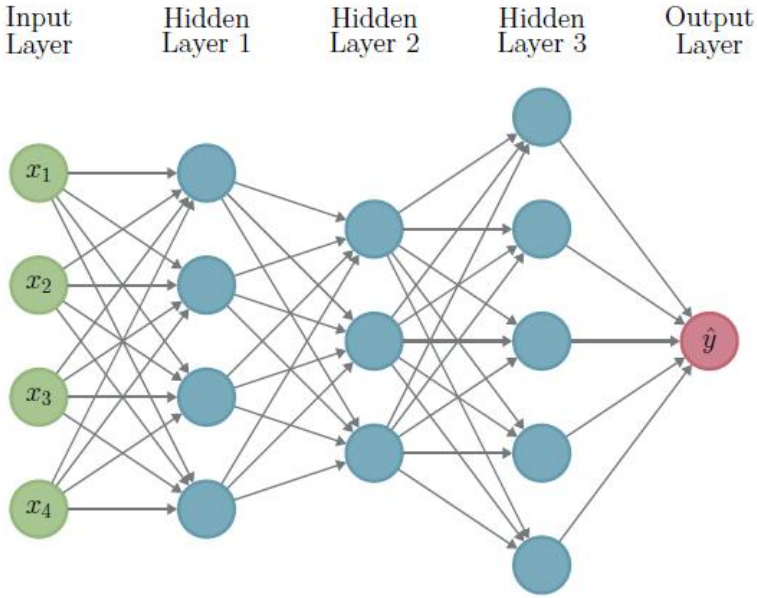
Neurona artificial

Liquet, B., Moka, S., & Nazarathy, Y. (2024). Mathematical Engineering of Deep Learning (1st ed.). Chapman and Hall/CRC.

RED NEURONAL



Conexión de múltiples neuronas biológicas en el cerebro



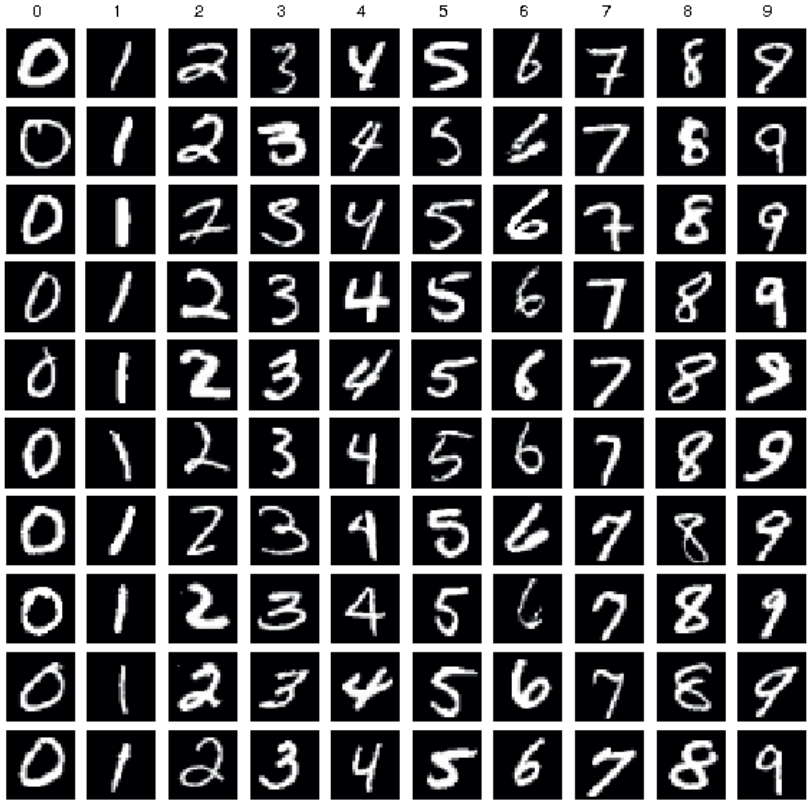
Red neuronal artificial conectando múltiples neuronas

Liquet, B., Moka, S., & Nazarathy, Y. (2024). Mathematical Engineering of Deep Learning (1st ed.). Chapman and Hall/CRC.

CONJUNTO DE DATOS POPULARES

- **MNIST**: Imágenes en blanco y negro de dígitos escritos a mano de tamaño 28×28 pixeles.
- Cada etiqueta y es un elemento entre $\{0,1,\ldots,9\}$ indicando el digito. La distribución de los dígitos esta balanceado (hay aprox. el mismo núm. de imágenes para c/digito).
- El conjunto está dividido en:
  - 60000 imágenes para entrenamiento.
  - 10000 imágenes para pruebas.

https://yann.lecun.com/exdb/mnist/

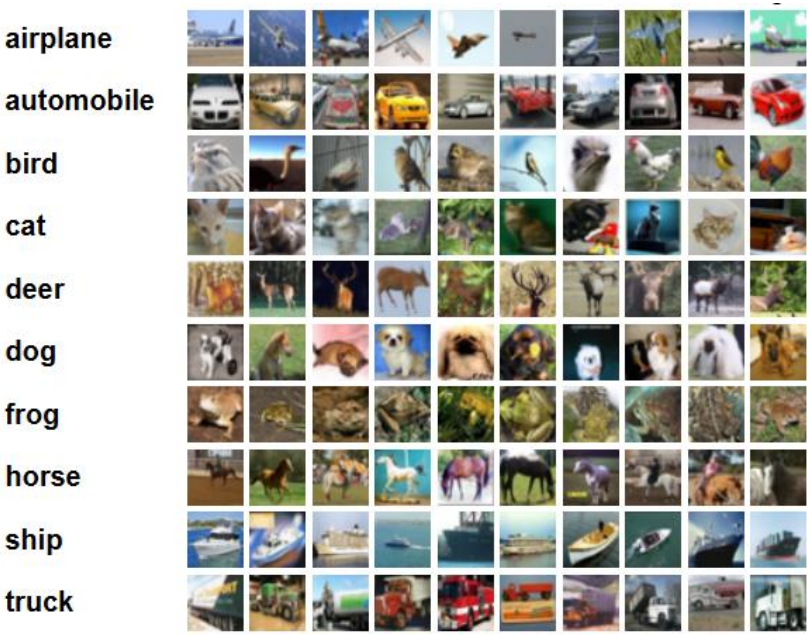


Liquet, B., Moka, S., & Nazarathy, Y. (2024). Mathematical Engineering of Deep Learning (1st ed.). Chapman and Hall/CRC.

CIFAR-10

- Consiste en imágenes a color de 32×32 pixeles de 10 clases.
- El conjunto está balanceado.
- El conjunto está dividido en:
  - 50000 imágenes para entrenamiento.
  - 10000 imágenes para pruebas.

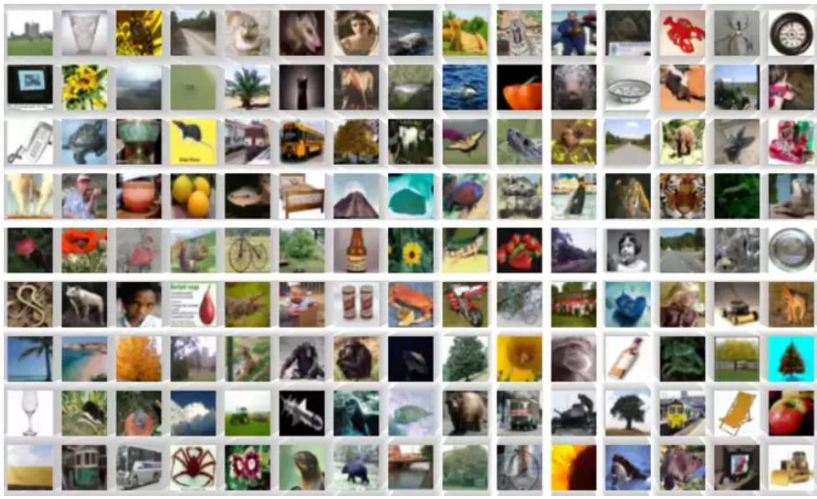
https://www.cs.toronto.edu/~kriz/cifar.html



CIFAR-100

https://www.cs.toronto.edu/~kriz/cifar.html

- Consiste en imágenes a color de 32×32 pixeles de 100 clases.
- El conjunto está balanceado.
- El conjunto está dividido en:
  - 50000 imágenes para entrenamiento.
  - 10000 imágenes para pruebas.

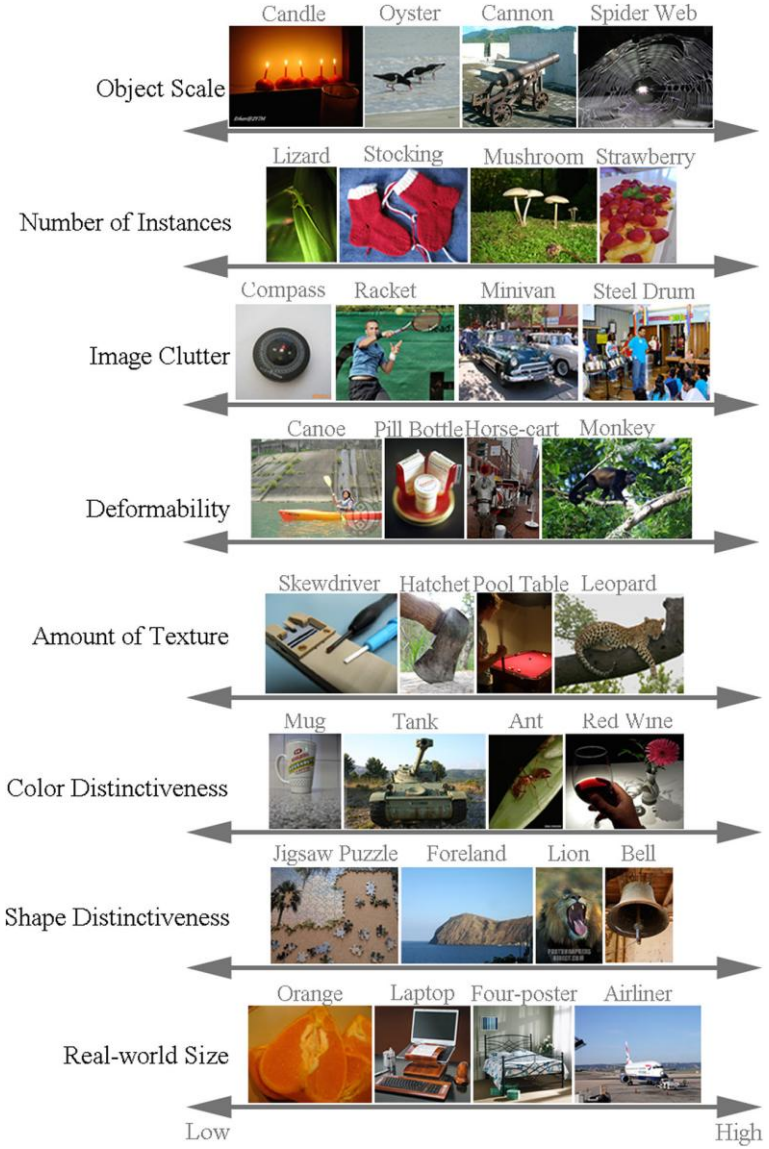


Superclass	Classes
aquatic mammals	beaver, dolphin, otter, seal, whale
fish	aquarium fish, flatfish, ray, shark, trout
flowers	orchids, poppies, roses, sunflowers, tulips
food containers	bottles, bowls, cans, cups, plates
fruit and vegetables	apples, mushrooms, oranges, pears, sweet peppers
household electrical devices	clock, computer keyboard, lamp, telephone, television
household furniture	bed, chair, couch, table, wardrobe
insects	bee, beetle, butterfly, caterpillar, cockroach
large carnivores	bear, leopard, lion, tiger, wolf
large man-made outdoor things	bridge, castle, house, road, skyscraper
large natural outdoor scenes	cloud, forest, mountain, plain, sea
large omnivores and herbivores	camel, cattle, chimpanzee, elephant, kangaroo
medium-sized mammals	fox, porcupine, possum, raccoon, skunk
non-insect invertebrates	crab, lobster, snail, spider, worm
people	baby, boy, girl, man, woman
reptiles	crocodile, dinosaur, lizard, snake, turtle
small mammals	hamster, mouse, rabbit, shrew, squirrel
trees	maple, oak, palm, pine, willow
vehicles 1	bicycle, bus, motorcycle, pickup truck, train
vehicles 2	lawn-mower, rocket, streetcar, tank, tractor

IMAGENET

https://www.image-net.org/

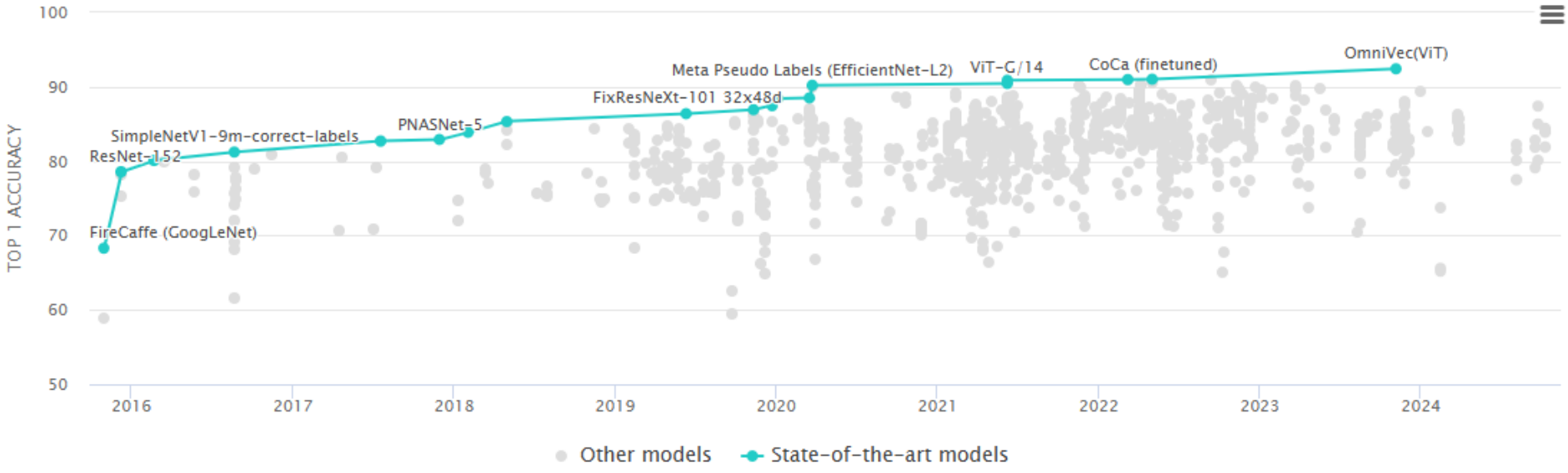
- Es un *benchmark* en la clasificación y detección de categorías de objetos.
- 22000 categorías y 15 millones de imágenes de 470×385 pixels en promedio.
- El ILSVRC usa un subconjunto del ImageNet:
  - 1.2 millones de imágenes para entrenamiento.
  - 50 mil imágenes para validación.
  - 150 mil imágenes para pruebas.



Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 25.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... & Fei-Fei, L. (2015). Imagenet large scale visual recognition challenge (**ILSVRC**). International journal of computer vision, 115, 211-252.

IMAGENET [CLASIFICACIÓN]



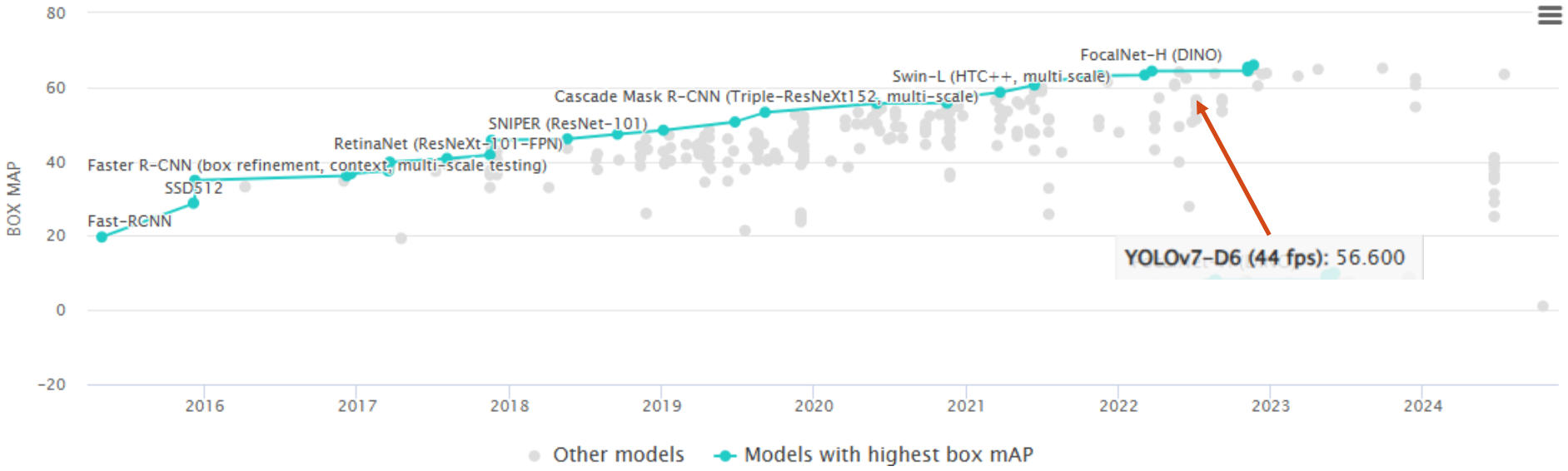
https://paperswithcode.com/sota/image-classification-on-imagenet

https://cocodataset.org/#home

- Microsoft COCO contiene 91 categorías. 82 tienen más de 5000 instancias etiquetadas.
- En total son 2500000 instancias etiquetadas en 328000 imágenes.
- 2014:
  - 82783 imágenes para entrenam. (1/2).
  - 40504 imágenes para validación (1/4).
  - 40775 imágenes para pruebas (1/4).
- 2015:
  - 165482 imágenes para entrenam. (1/2).
  - 81208 imágenes para validación (1/4).
  - 81434 imágenes para pruebas (1/4).

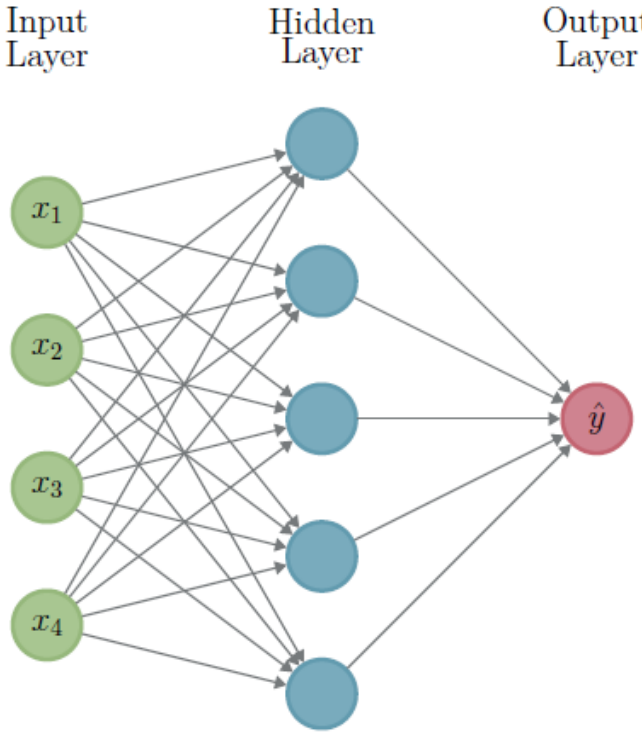
Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., ... & Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13* (pp. 740-755). Springer International Publishing.

[DETECCIÓN DE OBJETOS]

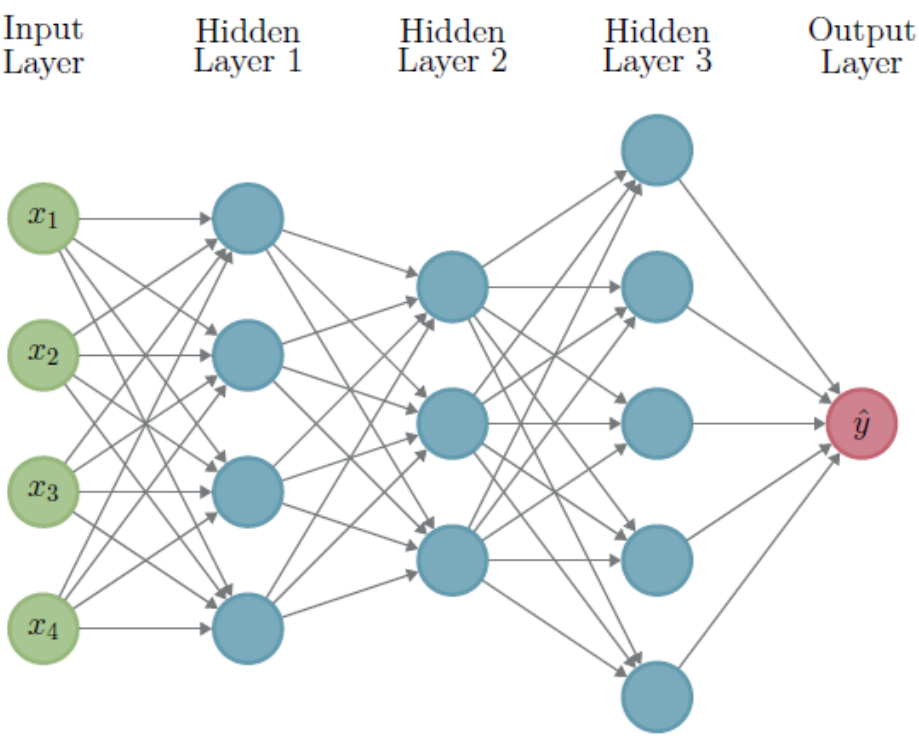


https://paperswithcode.com/sota/object-detection-on-coco

REDES NEURONALES COMPLETAMENTE CONECTADAS



Con una capa oculta



Con múltiples capas ocultas

Liquet, B., Moka, S., & Nazarathy, Y. (2024). Mathematical Engineering of Deep Learning (1st ed.). Chapman and Hall/CRC.

COMPOSICIÓN DE FUNCIONES

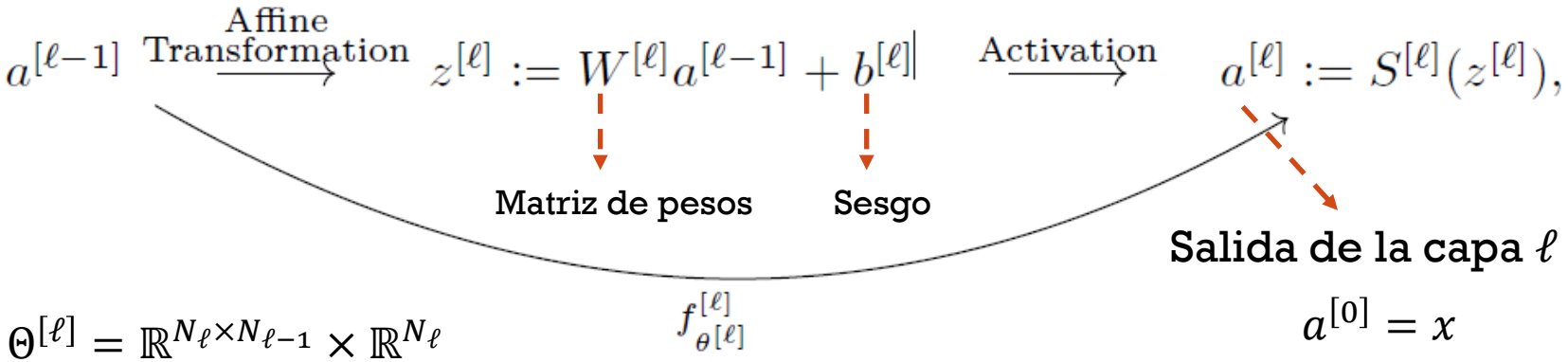
- El objetivo de una red de propagación hacia adelante (*feedforward*) es aproximar alguna función $f^*: \mathbb{R}^p \to \mathbb{R}^q$ y aprender los valores de los parámetros θ de tal forma que
$$f_\theta(x) \approx f^*(x).$$
- La función f_θ se compone de forma recursiva por una cadena de funciones
$$f_\theta(x) = f^{[L]}_{\theta^{[L]}}\left(f^{[L-1]}_{\theta^{[L-1]}}\left(\dots\left(f^{[1]}_{\theta^{[1]}}(x)\right)\dots\right)\right),$$

donde $f^{[\ell]}_{\theta^{[\ell]}}: \mathbb{R}^{N_{\ell-1}} \to \mathbb{R}^{N_\ell}$ está asociado con la capa ℓ-ésima la cual depende de los parámetros $\theta^{[\ell]} \in \Theta^{[\ell]}$, donde $\Theta^{[\ell]}$ es el espacio de parámetros para la capa ℓ-ésima.

- La profundidad de la red es L.
- $N_0 = p$ es el número de características de los datos de entrada.
- $N_L = q$ es el número de las variables de salida.
- $\sum_{\ell=1}^{L} N_\ell$ es el número de neuronas en la red (sin N_0).

TRANSFORMACIONES AFINES SEGUIDAS POR ACTIVACIONES

- La función $f_{\theta^{[\ell]}}^{[\ell]}$ se define mediante una transformación afín seguida de una función de activación.
- Las funciones de activación introducen la no-linealidad en el modelo.



$\Theta^{[\ell]} = \mathbb{R}^{N_\ell \times N_{\ell-1}} \times \mathbb{R}^{N_\ell}$

$a^{[0]} = x$

$S^{[\ell]}(z) = \left[\sigma^{[\ell]}(z_1) \dots \sigma^{[\ell]}(z_{N_\ell})\right]^T$ función de activación para $\ell = 1, \dots, L-1$.

$\sigma^{[\ell]}: \mathbb{R} \to \mathbb{R}$ es una función de activación común a todas las capas ocultas.

Para $\ell = L$ puede tener una forma diferente dependiendo del problema.

Liquet, B., Moka, S., & Nazarathy, Y. (2024). Mathematical Engineering of Deep Learning (1st ed.). Chapman and Hall/CRC.

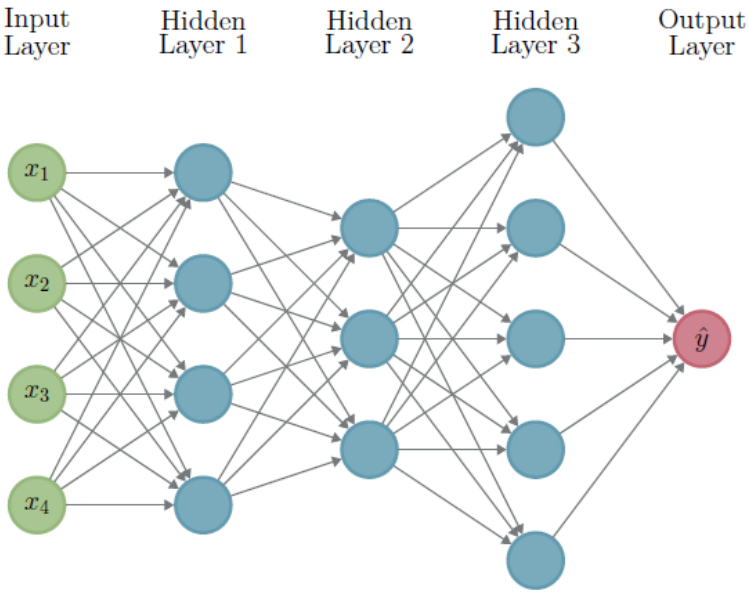
RETROALIMENTACIÓN HACIA ADELANTE

$$\text{Layer 1} \begin{cases} z^{[1]} = W^{[1]} \overbrace{x}^{\text{Input}} + b^{[1]} \\ a^{[1]} = S^{[1]}(z^{[1]}) \end{cases}$$

$$\text{Layer 2} \begin{cases} z^{[2]} = W^{[2]} a^{[1]} + b^{[2]} \\ a^{[2]} = S^{[2]}(z^{[2]}) \end{cases}$$

$$\vdots$$

$$\text{Layer L} \begin{cases} z^{[L]} = W^{[L]} a^{[L-1]} + b^{[L]} & N_\ell \times N_{\ell-1} + N_\ell \\ \underbrace{\hat{y}}_{\text{output}} = a^{[L]} = S^{[L]}(z^{[L]}). & N_\ell \end{cases}$$



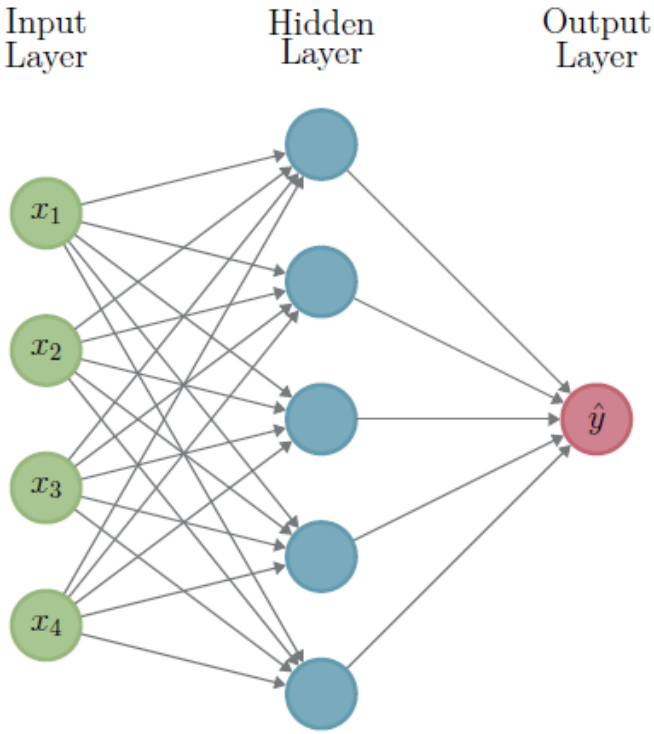
Costo Computacional:

Número total de pesos que hay que estimar: $\sum_{\ell=1}^{L} N_\ell(N_{\ell-1}+2) \approx \sum_{\ell=1}^{L} N_\ell N_{\ell-1}$

Liquet, B., Moka, S., & Nazarathy, Y. (2024). Mathematical Engineering of Deep Learning (1st ed.). Chapman and Hall/CRC.

EJEMPLO

- $N_0 = p = 4, N_1 = 5$ y $N_2 = q = 1$
- $W^{[1]}$ es de 5×4
- $b^{[1]}$ es de tamaño 5
- $W^{[2]}$ es de 1×5
- $b^{[2]}$ es de tamaño 1
- θ es de tam. $5 \times 4 + 5 + 1 \times 5 + 1 = 31$



$$f_\theta(x) = S^{[2]}\left(W^{[2]}S^{[1]}\left(W^{[1]}x + b^{[1]}\right) + b^{[2]}\right)$$

$z^{[2]}$

$z^{[1]}$

$a^{[1]}$

$a^{[2]}$

VISTA DEL MODELO EN OPERACIONES

$$\underset{\text{Transformation}}{\text{Affine}}: \begin{cases} z_1^{[\ell]} &= w_{(1)}^{[\ell]\top} a^{[\ell-1]} + b_1^{[\ell]} \\ z_2^{[\ell]} &= w_{(2)}^{[\ell]\top} a^{[\ell-1]} + b_2^{[\ell]} \\ &\vdots \\ z_{N_\ell}^{[\ell]} &= w_{(N_\ell)}^{[\ell]\top} a^{[\ell-1]} + b_{N_\ell}^{[\ell]} \end{cases} \Rightarrow \underset{\text{Step}}{\text{Activation}}: \begin{cases} a_1^{[\ell]} &= \sigma\left(z_1^{[\ell]}\right) \\ a_2^{[\ell]} &= \sigma\left(z_2^{[\ell]}\right) \\ &\vdots \\ a_{N_\ell}^{[\ell]} &= \sigma\left(z_{N_\ell}^{[\ell]}\right) \end{cases}$$

$$w_{(j)}^{[\ell]\top} = \begin{bmatrix} w_{j,1}^{[\ell]} & \dots & w_{j,N_{\ell-1}}^{[\ell]} \end{bmatrix}, \qquad \text{for} \qquad j = 1, \dots, N_\ell,$$

Liquet, B., Moka, S., & Nazarathy, Y. (2024). Mathematical Engineering of Deep Learning (1st ed.). Chapman and Hall/CRC.

VECTORIZACIÓN

- Supongamos que tenemos n_b muestras para el entrenamiento

$$x^{(1)}, \dots, x^{(n_b)}$$

- Para cada muestra $x^{(i)}$ tenemos una predicción

$$x^{(i)} \rightarrow a^{[L](i)} = \hat{y}^{(i)} \quad i = 1, \dots, n_b$$

- Podemos tener la siguiente representación:

$$X^T = \begin{bmatrix} | & | & & | \\ x^{(1)} & x^{(2)} & \dots & x^{(n_b)} \\ | & | & & | \end{bmatrix}$$

$$Z^{[\ell]} = \begin{bmatrix} | & | & & | \\ z^{[\ell](1)} & z^{[\ell](2)} & \dots & z^{[\ell](n_b)} \\ | & | & & | \end{bmatrix}$$

$$A^{[\ell]} = \begin{bmatrix} | & | & & | \\ a^{[\ell](1)} & a^{[\ell](2)} & \dots & a^{[\ell](n_b)} \\ | & | & & | \end{bmatrix}$$

$$\begin{aligned} Z^{[1]} &= W^{[1]} X^T + B^{[1]} \\ A^{[1]} &= \sigma^{[1]}\left(Z^{[1]}\right) \\ Z^{[2]} &= W^{[2]} A^{[1]} + B^{[2]} \\ A^{[2]} &= \sigma^{[2]}\left(Z^{[2]}\right) \\ &\vdots \\ Z^{[L]} &= W^{[L]} A^{[L-1]} + B^{[L]} \\ A^{[L]} &= S^{[L]}\left(Z^{[L]}\right) \end{aligned}$$

$$\left[\hat{y}^{(1)}, \dots, \hat{y}^{(n_b)}\right]$$

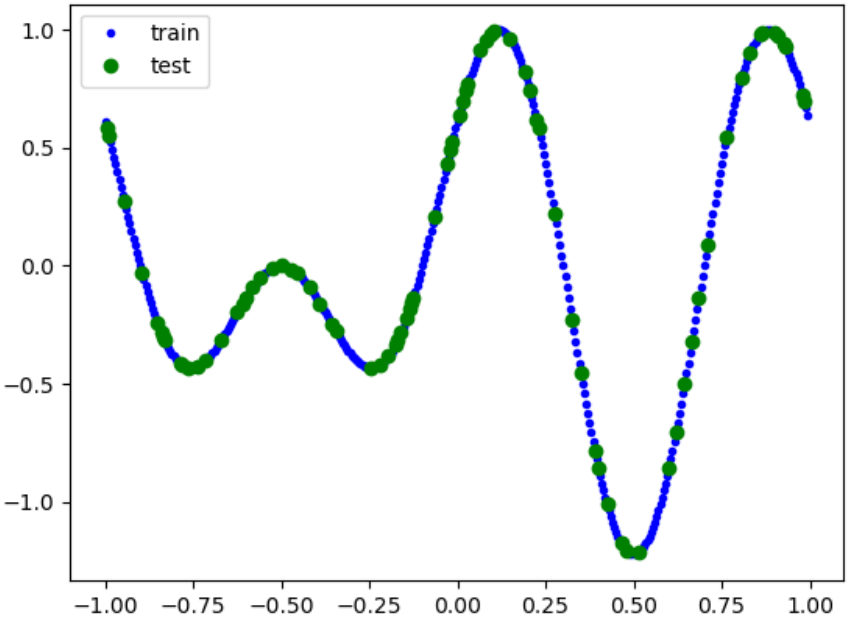
$$B^{[\ell]} = \begin{bmatrix} | & | & & | \\ b^{[\ell]} & b^{[\ell]} & \dots & b^{[\ell]} \\ | & | & & | \end{bmatrix}$$

EJEMPLO: APROXIMACIÓN A UNA FUNCIÓN

- Dada una función $f^*: [\underline{l}, \bar{l}] \rightarrow \mathbb{R}$

$$f^*(x) = sen(3\pi x) + \cos(2\pi x) \text{ para } x \in [-1,1]$$

- Dados los valores de la función $v_i = f^*(u_i)$ para $u_1, \ldots, u_r$ con $\underline{l} \leq u_1 < u_2 < \cdots < u_r \leq \bar{l}$
- Hay que estimar f_θ que aproxima a f^*.




```
7   #Definimos la función
8   def y(x):
9       return (np.sin(3*np.pi*x) + np.cos(2*np.pi*x))
10
11  #Definimos los valores (-1,+1,discretización)
12  x_vals = np.arange(-1,1,0.005)
13  y_vals = y(x_vals)
```

```
15  #Normalizamos los datos
16  y_max = y_vals.max()
17  y_vals /= y_max
```

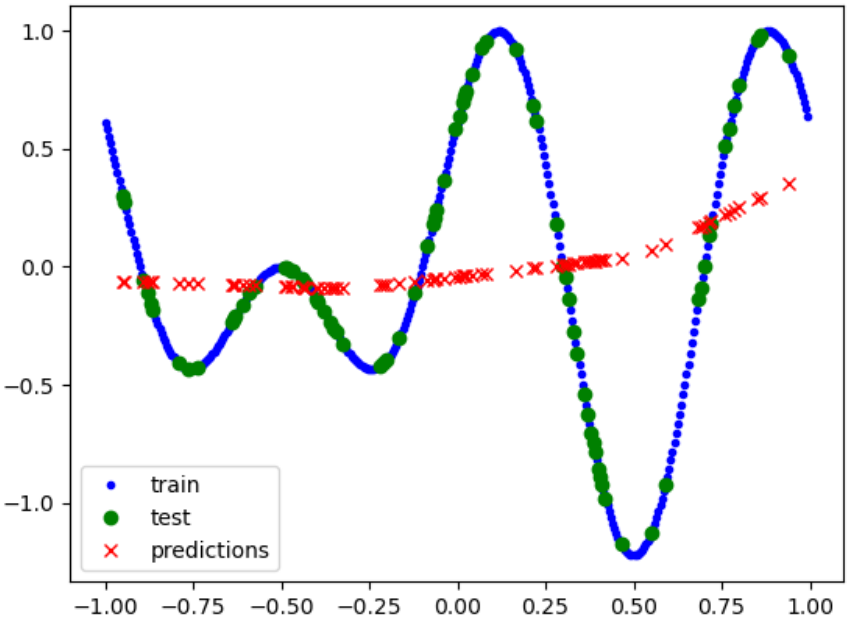
AJUSTE CON MODELO DE RN - [10]

```
#Dividimos el conjunto de datos en train y test
x_train, x_test, y_train, y_test = \
train_test_split(x_vals, y_vals, test_size=0.20)
print("Puntos para Entrenar: ",x_train.shape,
      "Puntos para Pruebas: ",x_test.shape)
```

```
#Modificamos el shape de las variables
#para que sean de la forma: (#datos,1)
x_train = x_train.reshape(-1,1)
x_test = x_test.reshape(-1,1)
print("Puntos para Entrenar: ",x_train.shape,
      "Puntos para Pruebas: ",x_test.shape)
```

```
#Creamos la Red Neuronal
mlp = MLPRegressor(
    hidden_layer_sizes=[10],
    max_iter=1000,
)
#Estimamos los parámetros
mlp.fit(x_train,y_train)

#Realizamos las predicciones
predictions = mlp.predict(x_test)
```

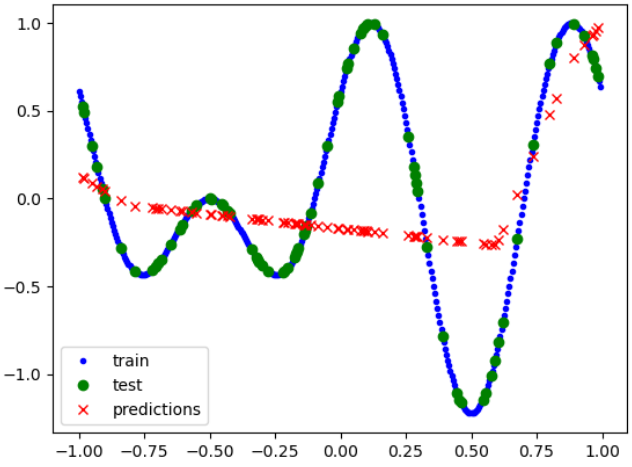



```
#Mostramos la función
plt.figure()
plt.plot(x_train, y_train, 'b.',label="train")
plt.plot(x_test, y_test, 'go',label="test")
#Mostramos las predicciones
plt.plot(x_test, predictions, 'rx',label="predictions")
plt.legend()
plt.show()
```

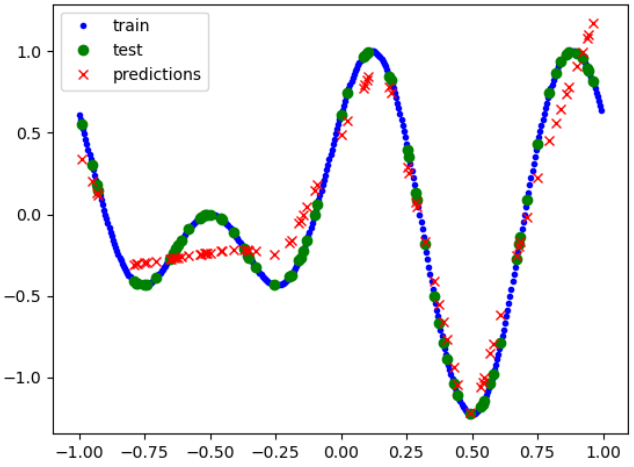
CAMBIANDO EL MODELO

¿Cuál será la mejor arquit. que aprox. bien a los datos, que sea eficiente y que además haga buenas predic. sobre datos nuevos (generalice)?

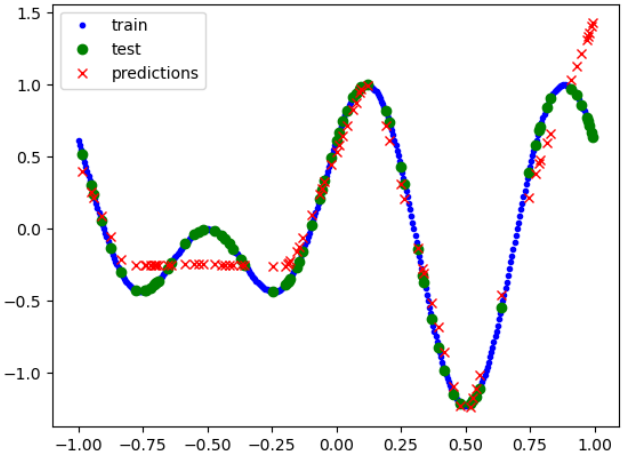
[10,10]



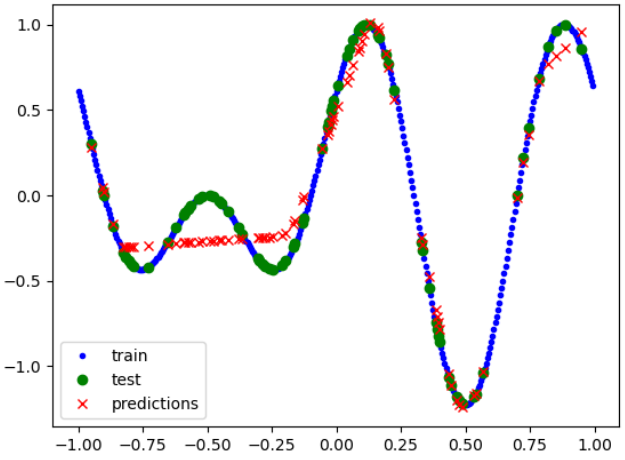
[20,20]



[40,40]



[40,40,40]



GRACIAS POR SU ATENCIÓN

Francisco J. Hernández López

fcoj23@cimat.mx

WebPage:

www.cimat.mx/~fcoj23