CIS

Redes Neuronales 2

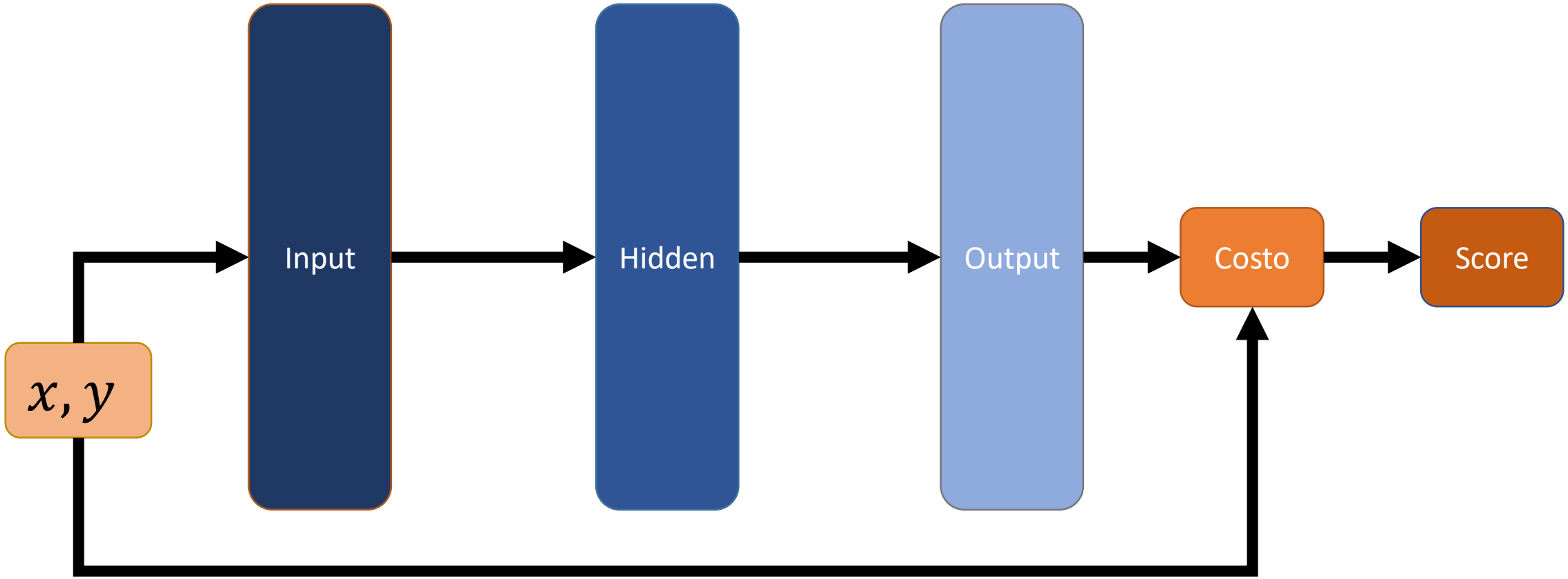
Machine Learning
Fernando Teshima
2021-2

UNIVERSIDAD DE LIMA

Redes Neuronales

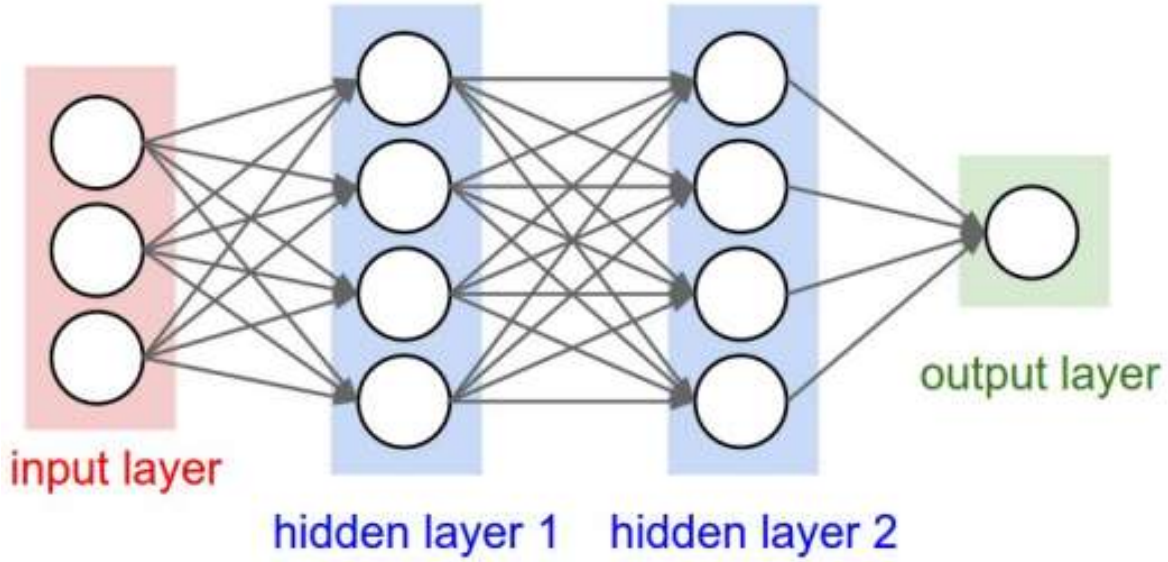
Parte 2

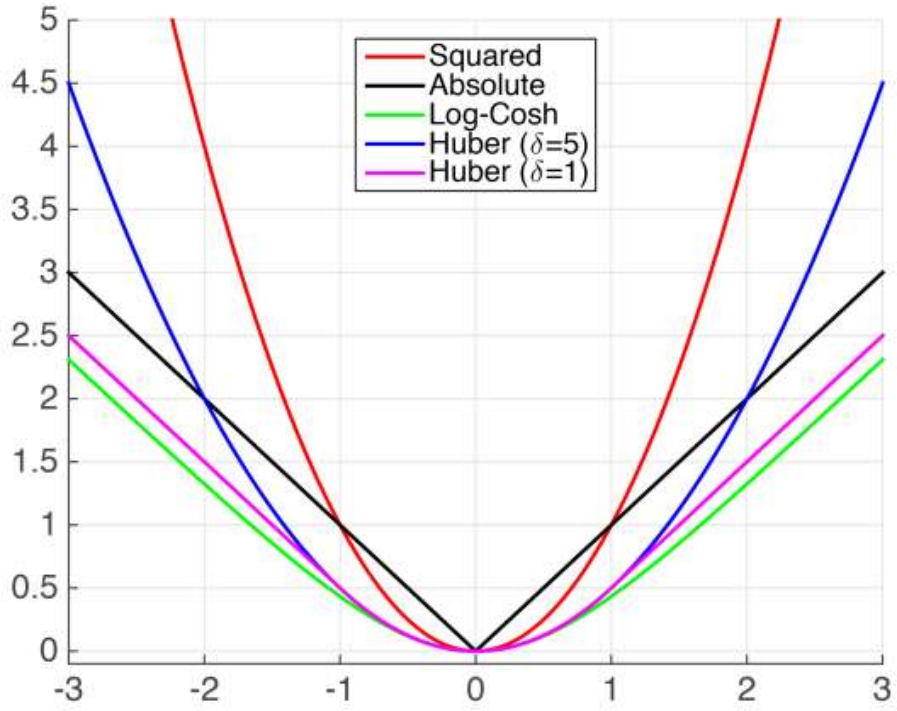
Estructura General



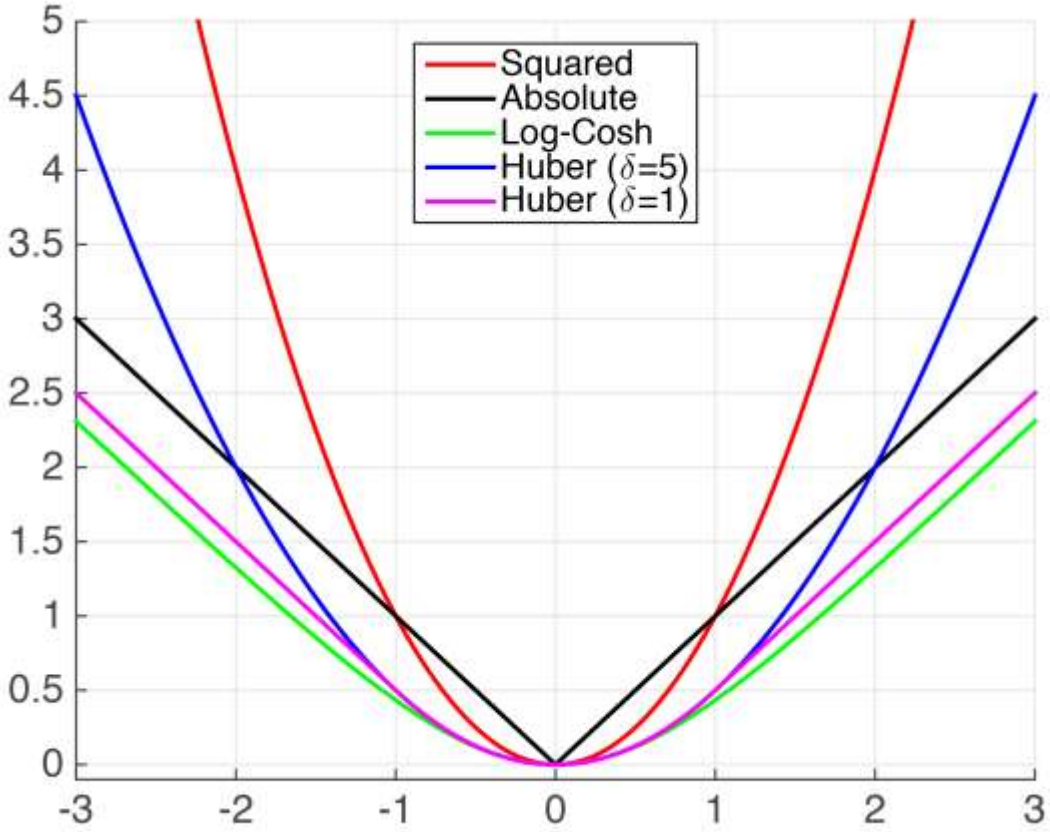
Personalizando la red neuronal

- Principalmente se modifican la función de costo y las capas ocultas.





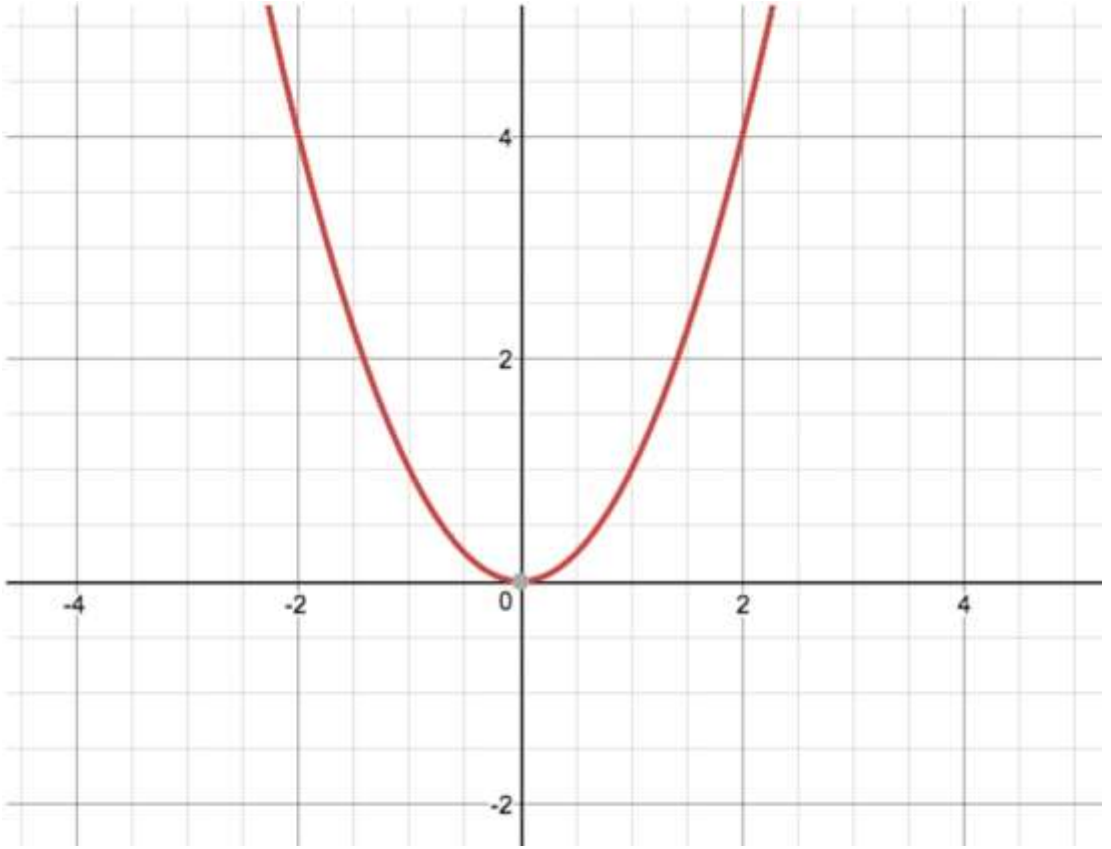
Funciones de Costo



MSE (Mean Squared Error)

$$\sum_{k=1}^{m} \left(y_k - a_k^{(L)}\right)^2$$

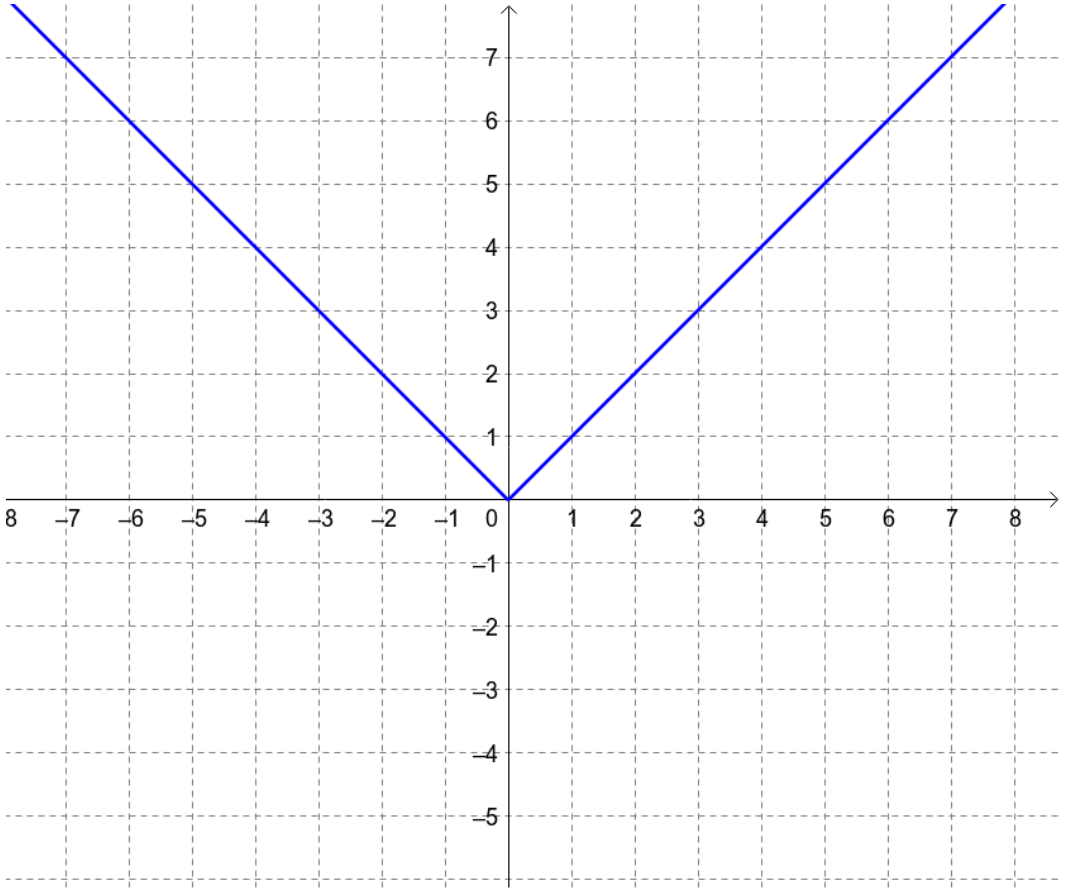
- Penaliza las grandes diferencias
- Susceptible a outliers



MAE (Mean Absolute Error)

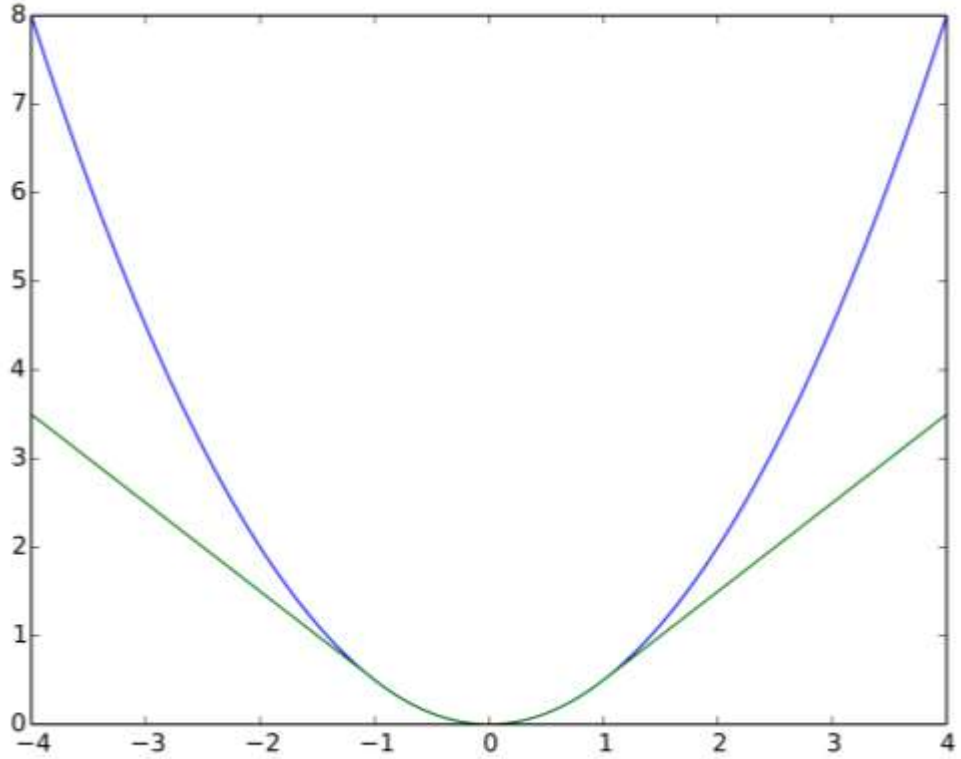
$$\sum_{k=1}^{m} \left| y_k - a_k^{(L)} \right|$$

- Mejor manejo de outlier que MSE
- Puede tener mínimos locales
- Computacionalmente costoso



Huber

$$C_\delta \begin{cases} \frac{1}{2}a^2 & si\ |a| \leq \delta \\ \delta|a| - \frac{1}{2}\delta & de\ lo\ contrario \end{cases}$$



Maneja bien los outlier y mínimos locales

Se tiene δ como hiperparámetro

Más funciones de costo

Clasificación

- Log
- Focal
- Cross-Entropy
- Hinge

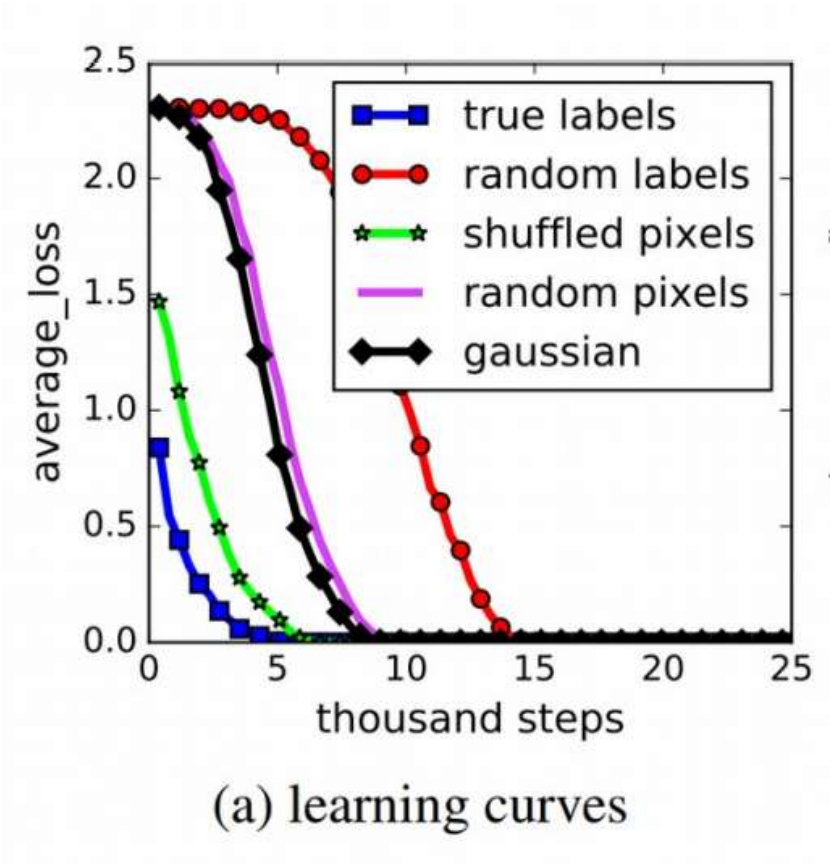
Regresión

- MSE
- MAE
- Huber
- Log-cosh

El poder de la profundidad

Redes neuronales profundas pueden fácilmente ajustarse a etiquetas aleatorias

La capacidad de la red puede ser suficiente para memorizar completamente el conjunto de datos.



(a) learning curves

Regularización

- Weight regularization
- Weight clipping
- Dropout
- Incremento de data
- Early Stopping

Weight regularization ⚖️

$$C(w; x, y) = C(w; x, y) + \alpha\Omega(w)$$

Donde

C es la función de costo

w los coeficientes del modelo

x, y la data de entrenamiento

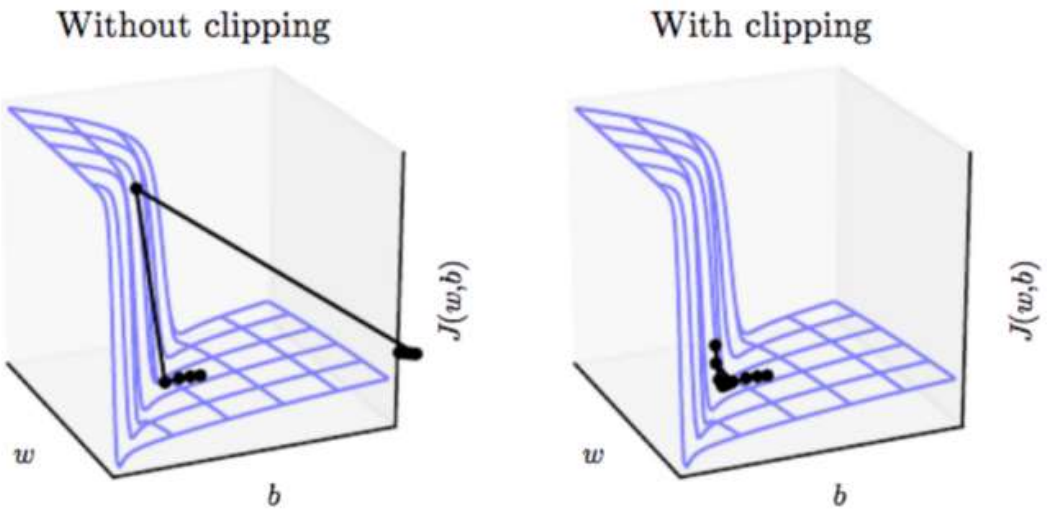
Ω la función de regularización, que penaliza modelos complejos

α el hiperparámetro que controla el grado de regularización

Ω puede ser por ejemplo w^2 o $|w|$

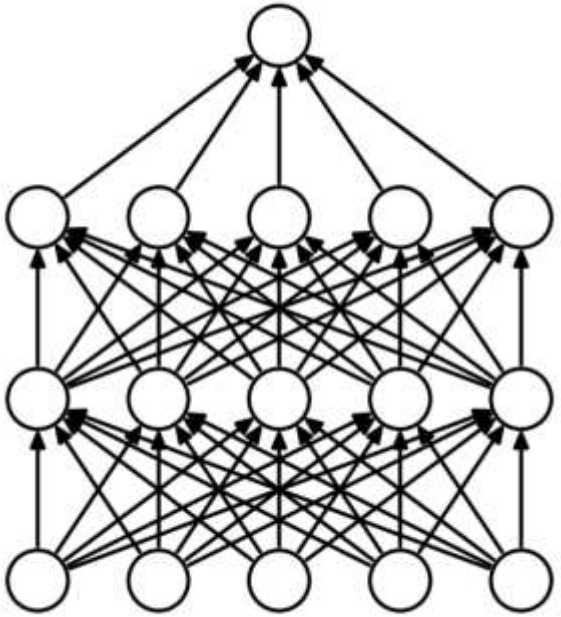
Weight clipping

- Método de limitar el tamaño de los coeficientes
- Si el coeficiente (o la norma del vector) supera un valor predeterminado, aplicar la reducción.

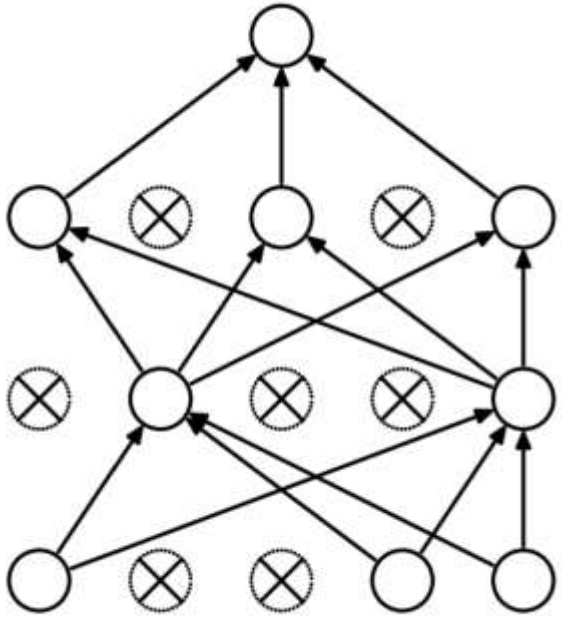


Dropout

- Durante el entrenamiento, ciertos nodos son ignorados (dropped out) de forma aleatoria
- Evita que la red dependa de nodos en particular.



(a) Standard Neural Net



(b) After applying dropout.

Dropout

$$a_j^l = \phi(z_j^l)$$

Cambia a:

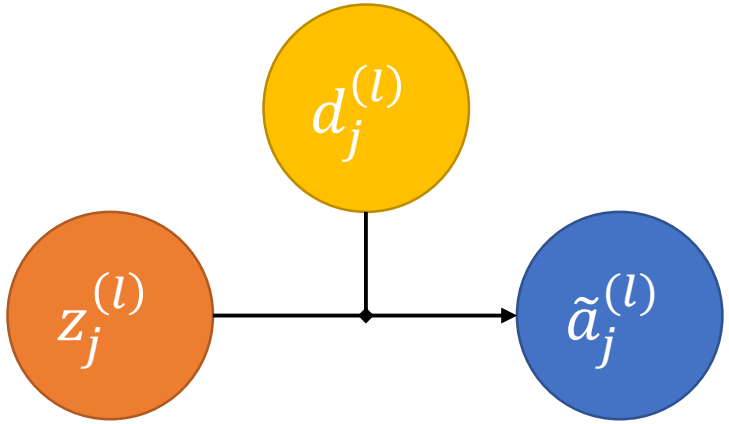
$$\tilde{a}_j^l = \frac{1}{1-p} \cdot d_j^l \cdot \phi(z_j^l)$$

Donde

d_j^l es $Bernoulli(1-p)$

El factor $\frac{1}{1-p}$ se añade para que el nodo siga teniendo el mismo valor esperado $\phi(z_j^l)$

$Bernoulli(1-p)$ es
0 con probabilidad p;
1 de lo contrario



Dropout

- Derivadas parciales:

$$\frac{\partial C}{\partial w_{j,k}^{(l)}} = \delta_j^l \cdot \tilde{a}_k^{l-1} \qquad \frac{\partial C}{\partial b_j^{(l)}} = \delta_j^l$$

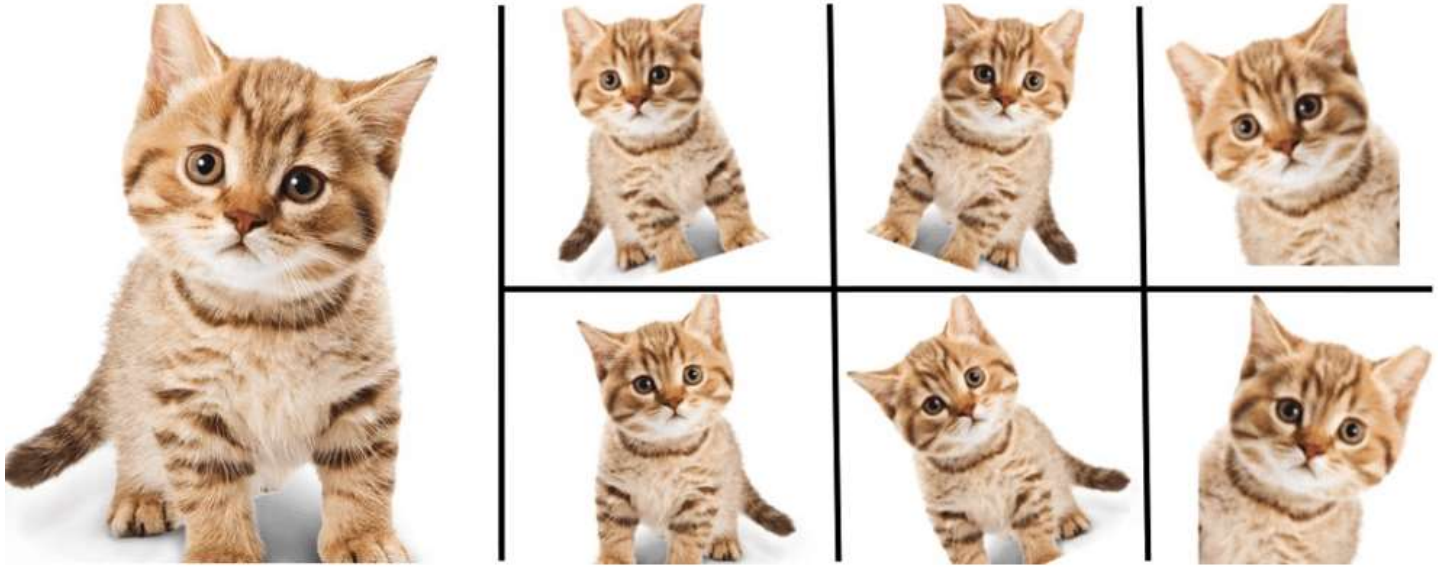
- Gradiente local

$$\delta_j^l = \frac{1}{1-p} \cdot d_j^l \cdot \phi'\left(z_j^l\right) \cdot \sum_k \delta_k^{l+1} \cdot w_{k,j}^{l+1}$$

Incremento de data

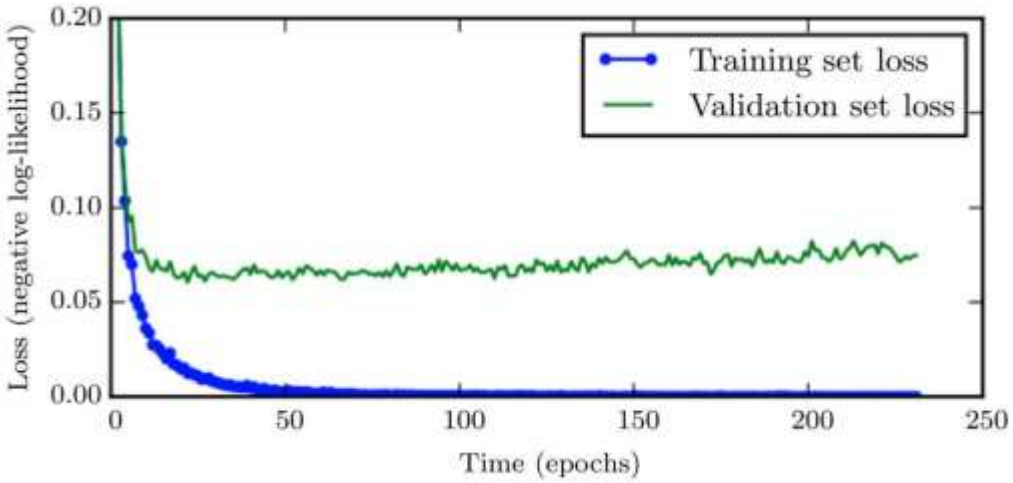
Usar la data existente y generar nuevas observaciones

- Transformaciones
  - Espejo
  - Translación
  - Escalar
  - Rotación
  - Ruido



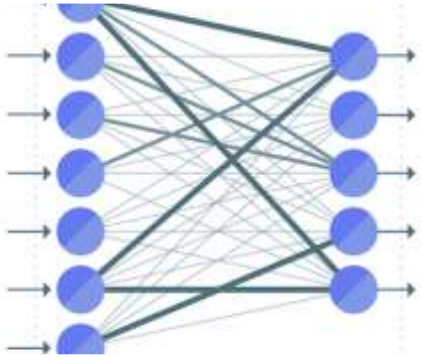
Early Stopping

- Entrenar la red usando el training set.
- Validar el modelo cada n número de iteraciones.
- Detener el entrenamiento cuando:
  - El error de validación se incrementa
  - No hay mejora del error de validación después de x iteraciones
- El modelo final es aquél con menor error de validación.

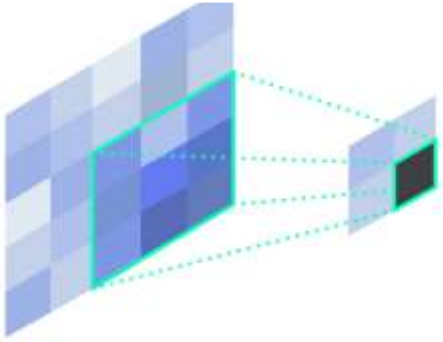


Capas Ocultas

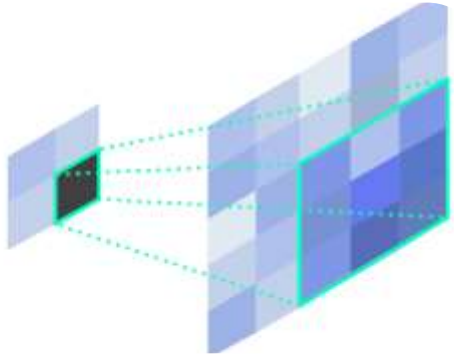
Capas Ocultas



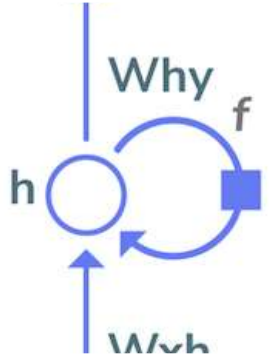
Fully Connected



Convolución



Deconvolución

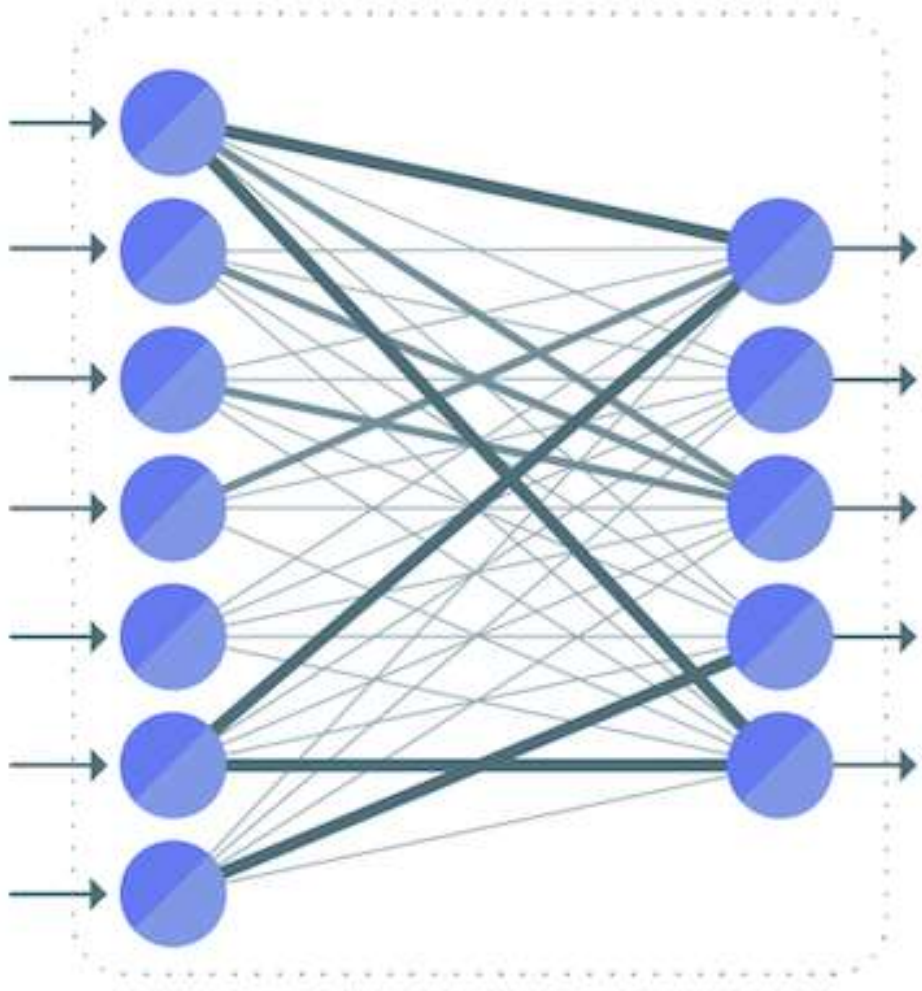


Recurrente

Fully Connected

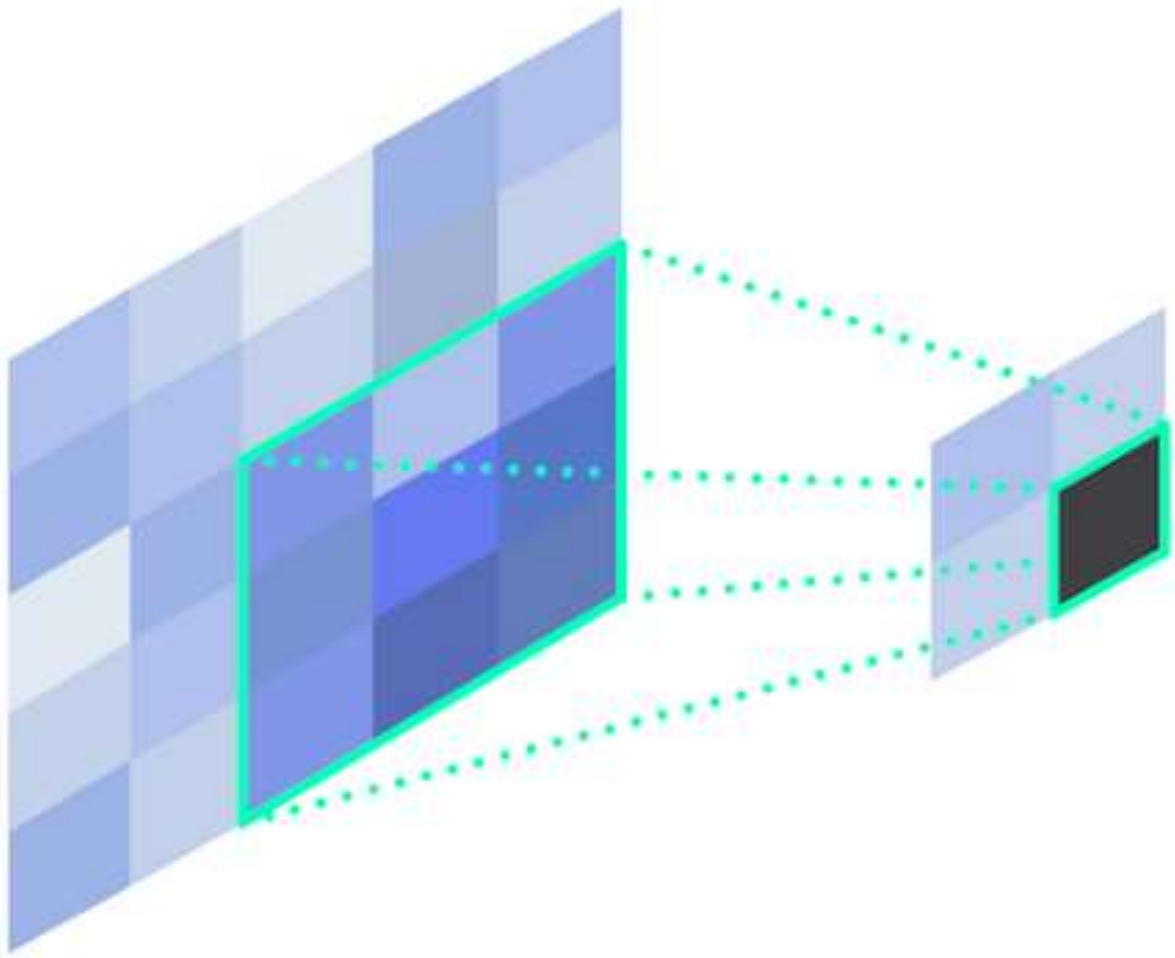
- Cada neurona se conecta con todas las otras neuronas de la siguiente capa
- Perceptrón multicapa

- Hiperparámetros asociados
  - Función de Activación
  - Número de Neuronas
  - Dropout



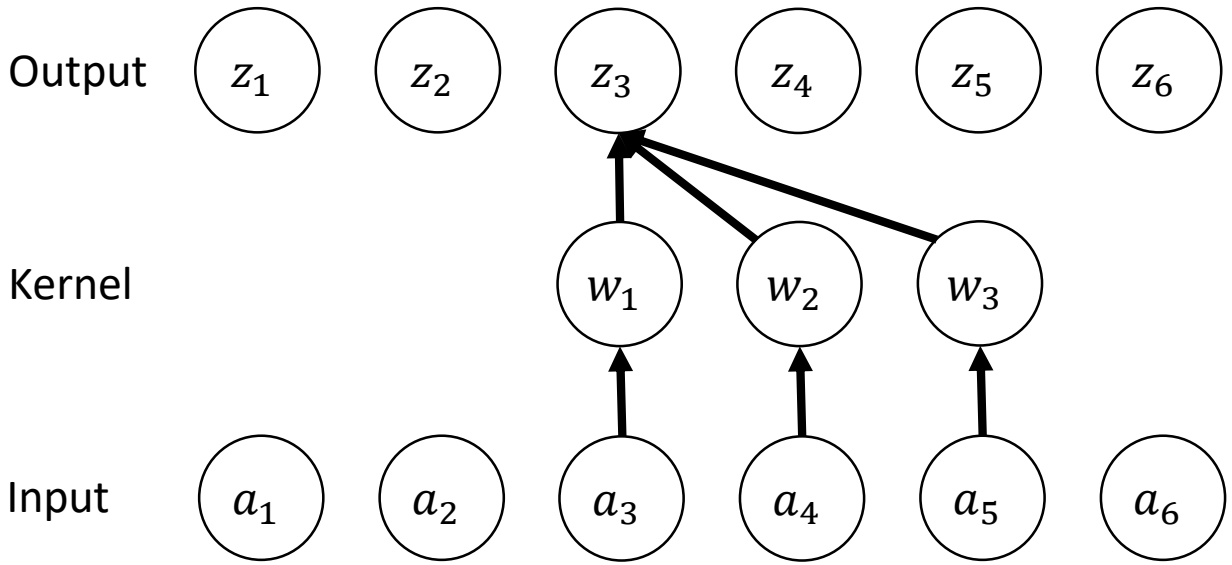
Convolución

- Detectar características en imágenes
- El filtro también es llamado kernel



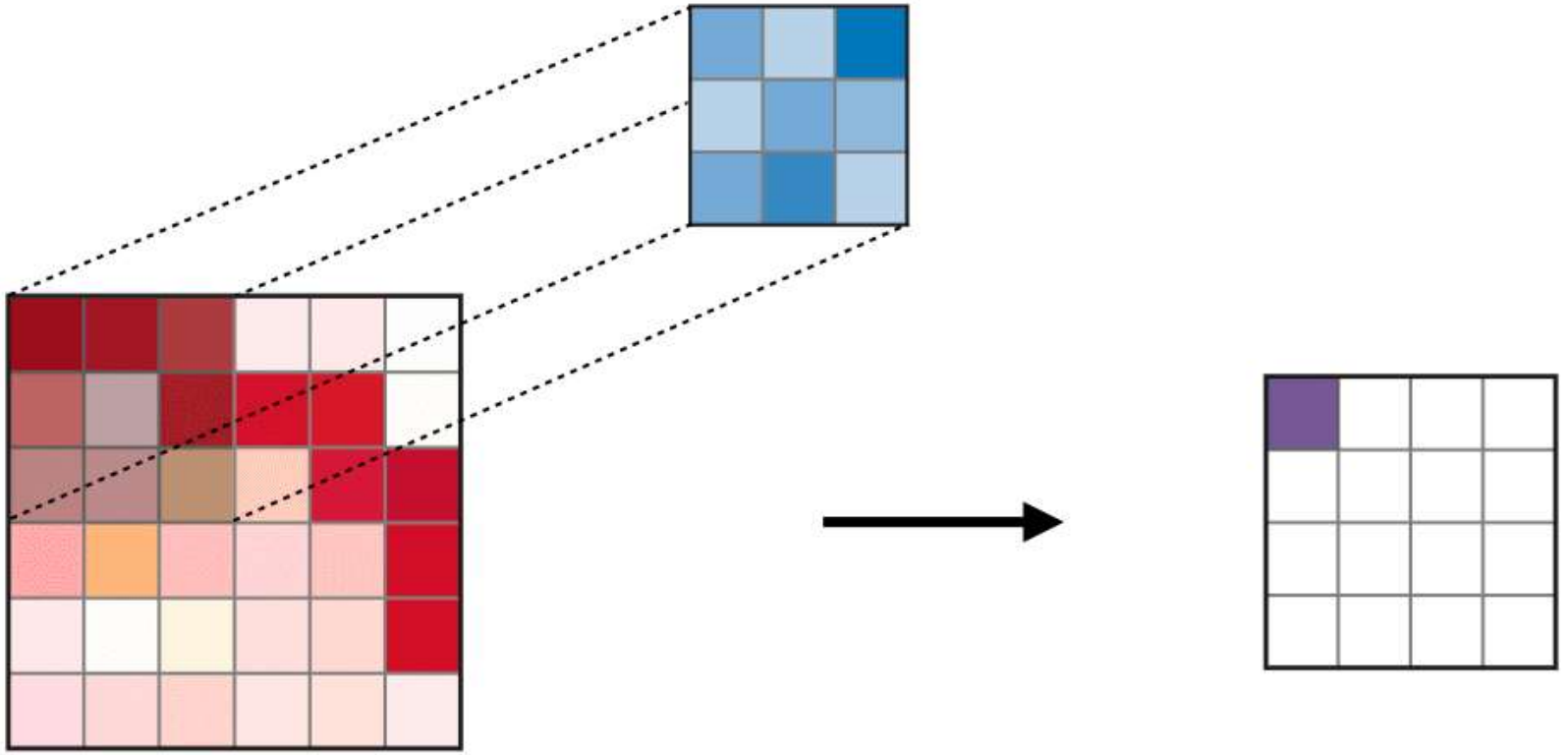
Convolución 1D

$$z_j = (a * w)_j$$



$$z_1 = a_1 w_1 + a_2 w_2 + a_3 w_3$$
$$z_2 = a_2 w_1 + a_3 w_2 + a_4 w_3$$
$$z_3 = a_3 w_1 + a_4 w_2 + a_5 w_3$$
$$\vdots$$

Convolución 2D



https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks

Convolución 2D: Ejemplo

- $\begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix}$

https://docs.gimp.org/2.2/en/plug-in-convmatrix.html

Convolutional Neural Network (CNN)

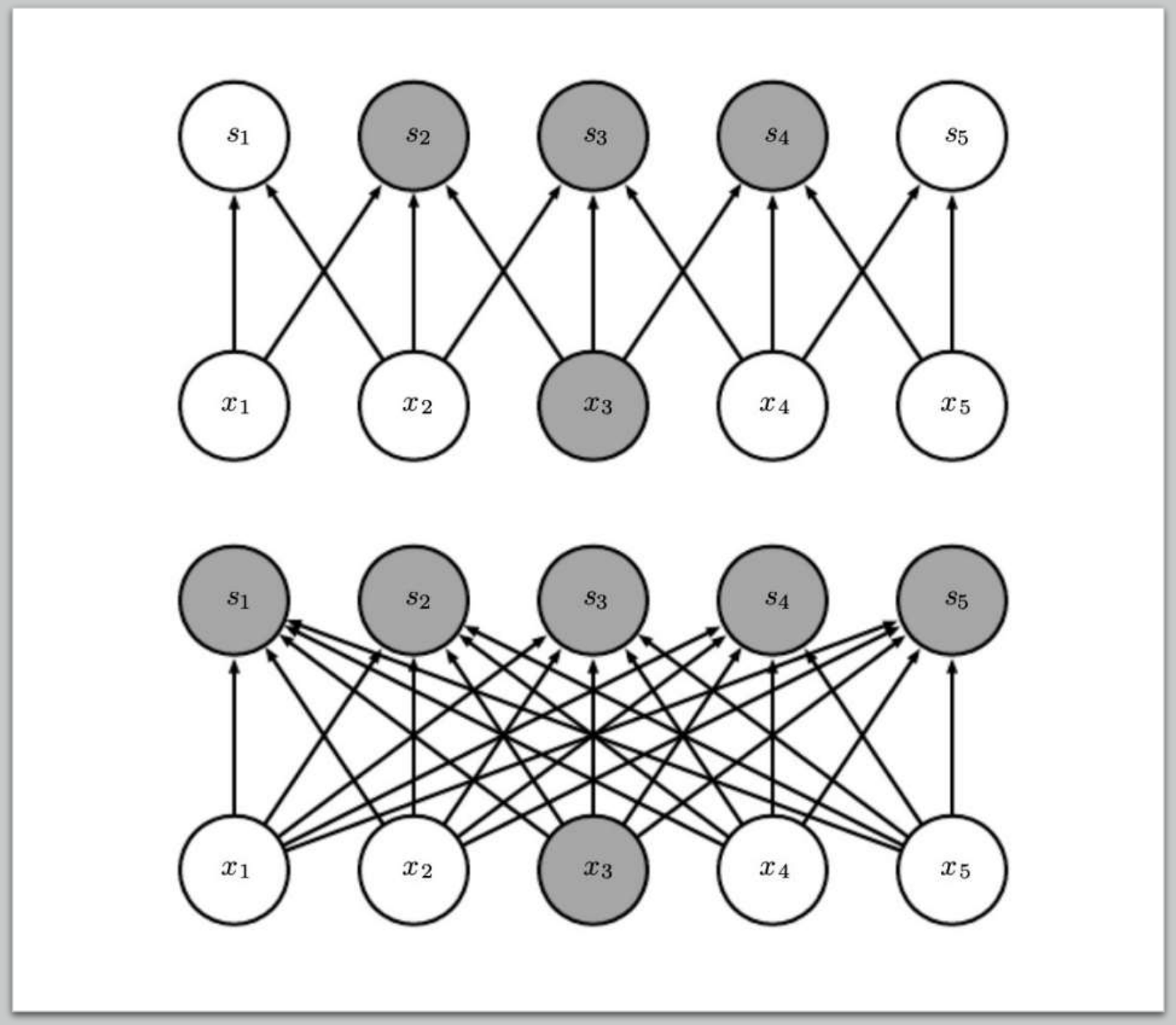
Una CNN aplica al menos una capa convolucional en al menos una capa.

Cada capa convolucional tiene múltiples kernels

Los kernels son entrenados por backpropagation y GD

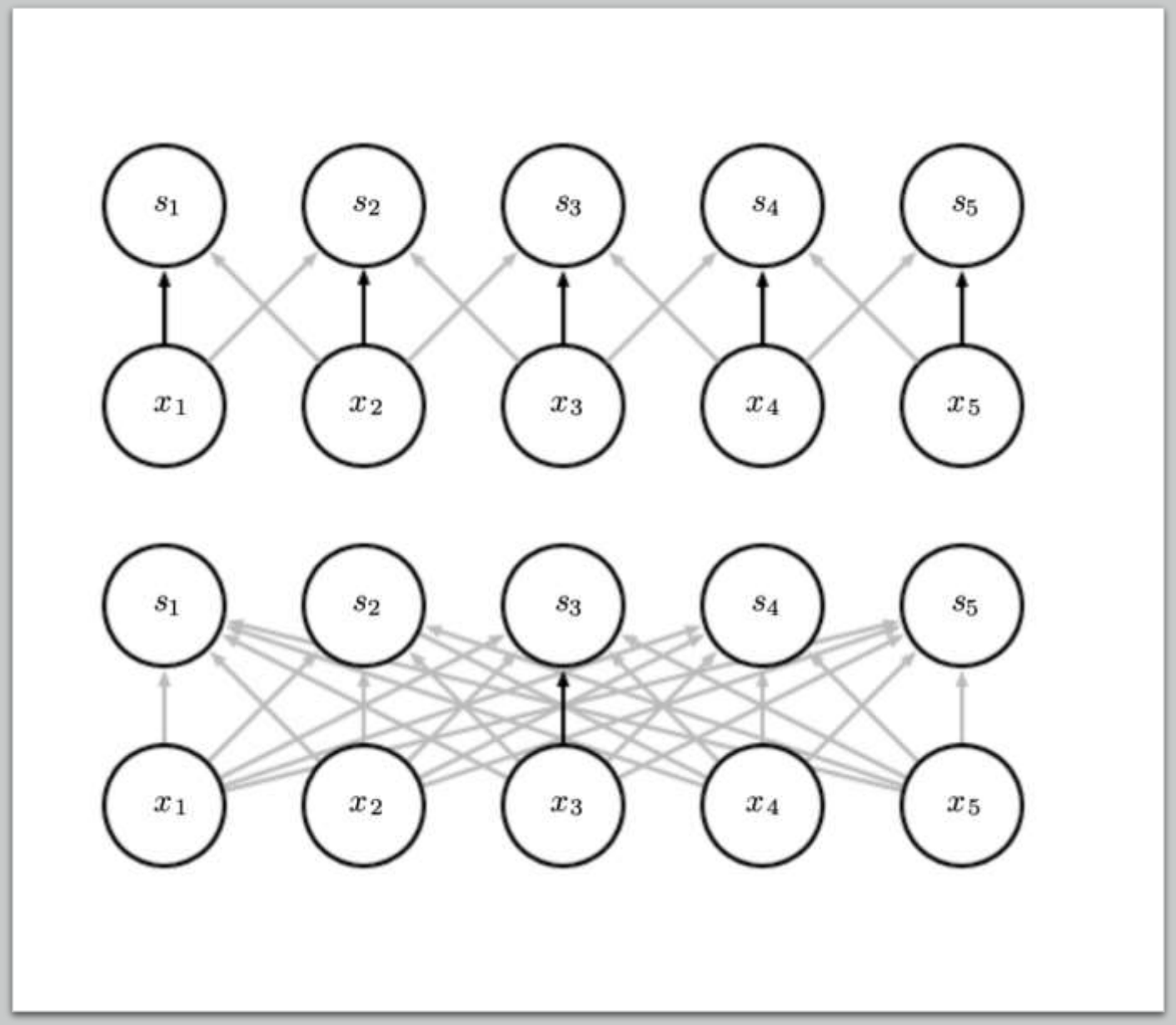
Propiedades del CNN

- Menor interacción entre unidades vecinas



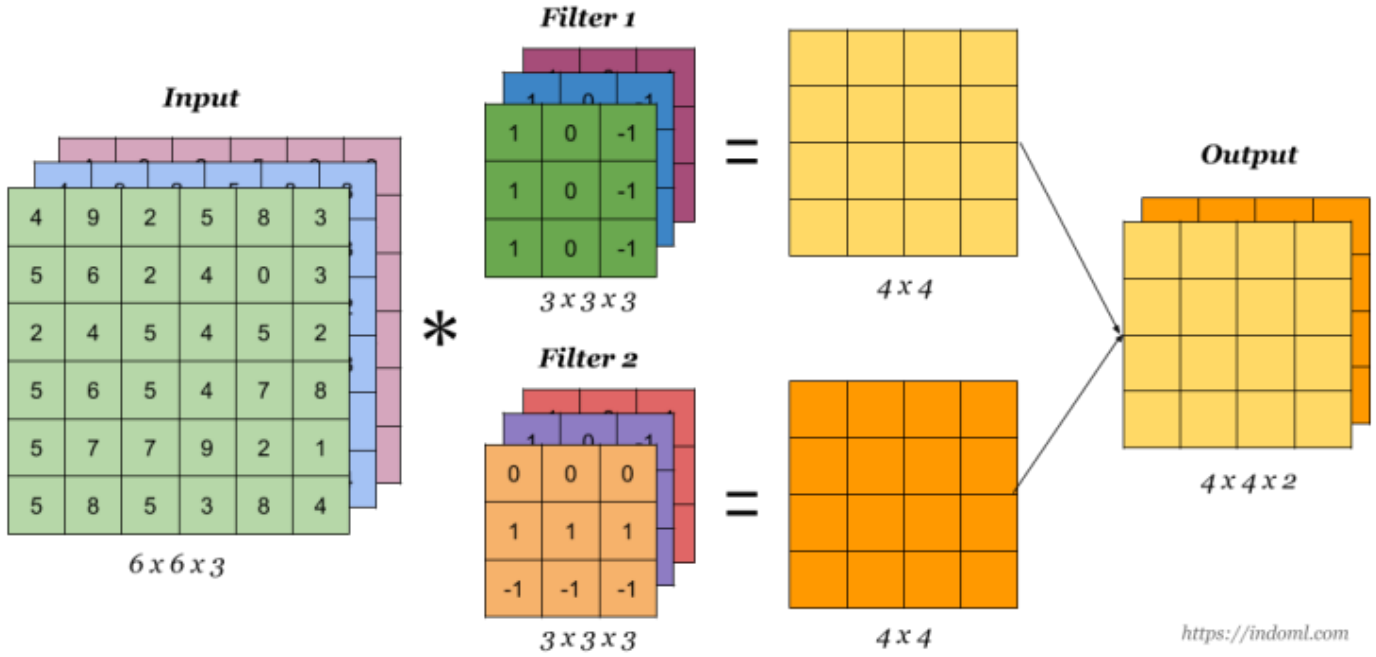
Propiedades del CNN

- Uso del mismo parámetro para más de una sola función en la red
- Reducción significativa en el número de parámetros (pesos)



Capa convolucional

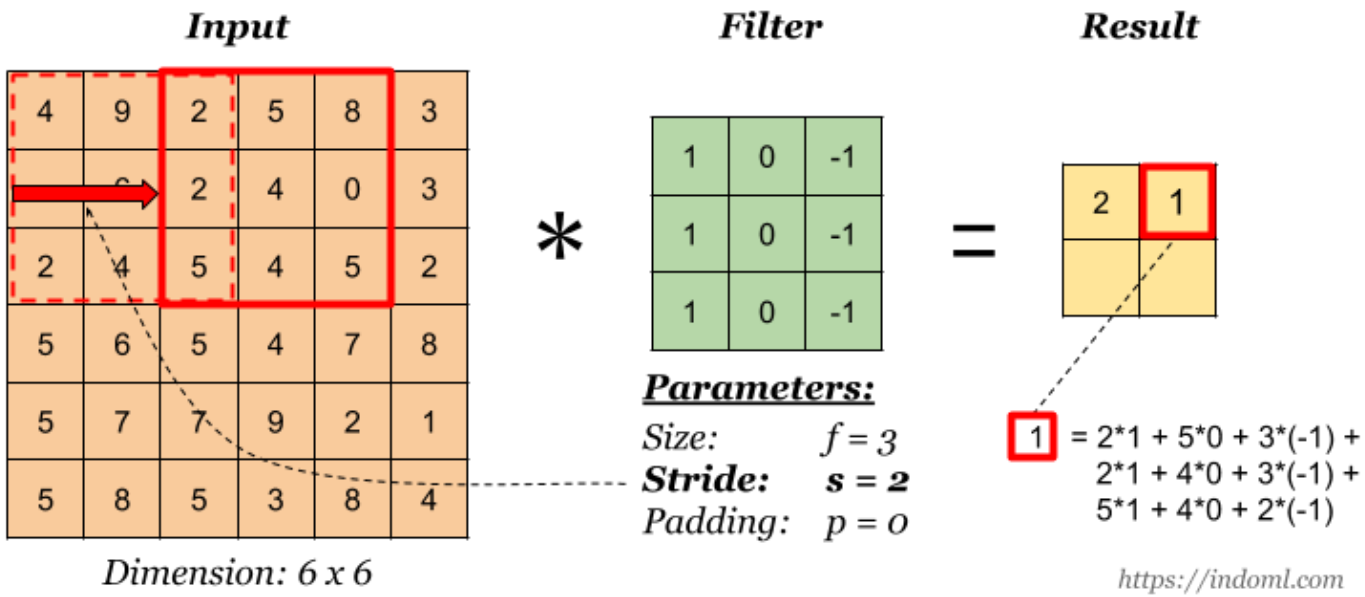
- Número de filtros
- Canales
- Tamaño
- Paso
- Padding
- Activación



https://indoml.com/2018/03/07/student-notes-convolutional-neural-networks-cnn-introduction/

Capa convolucional

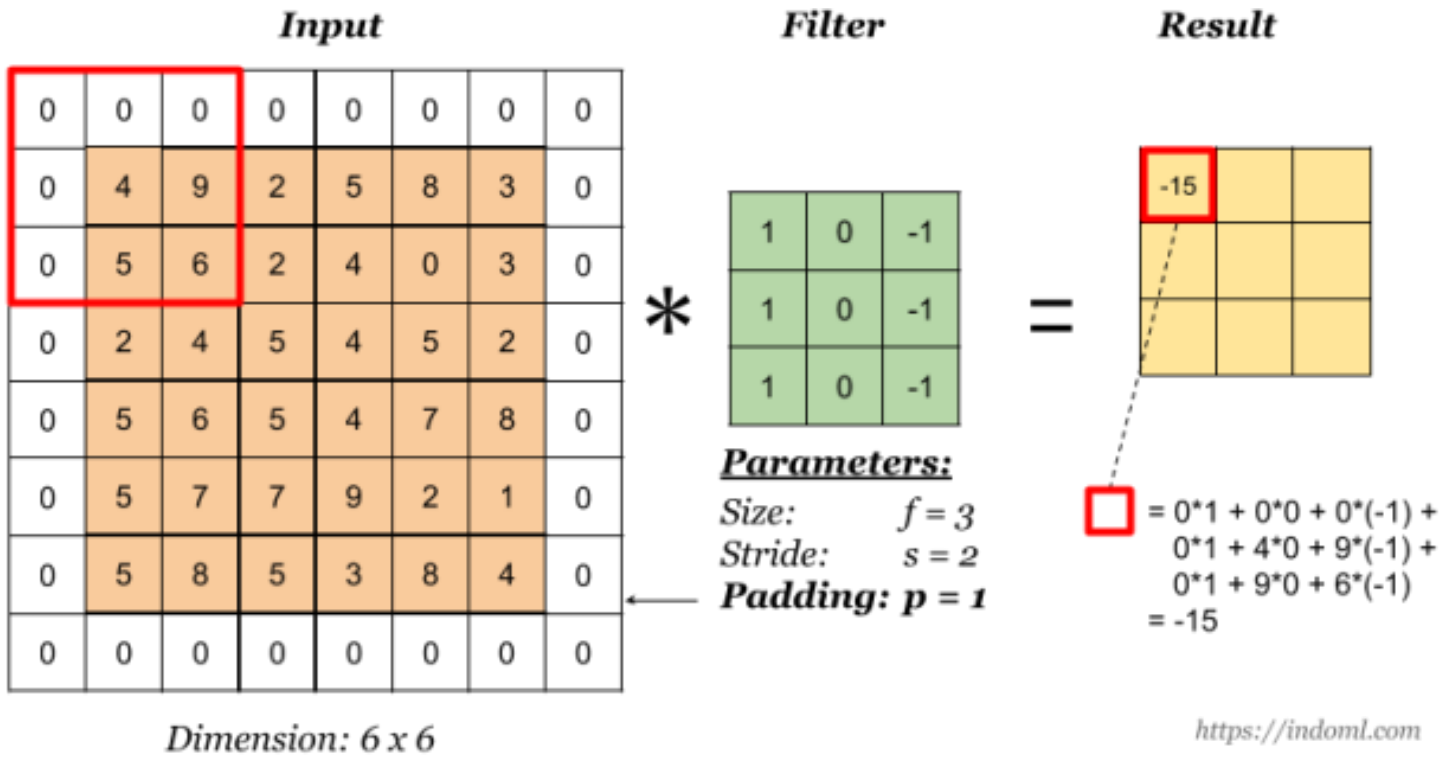
- Número de filtros
- Canales
- Tamaño
- Paso
- Padding
- Activación



https://indoml.com/2018/03/07/student-notes-convolutional-neural-networks-cnn-introduction/

Capa convolucional

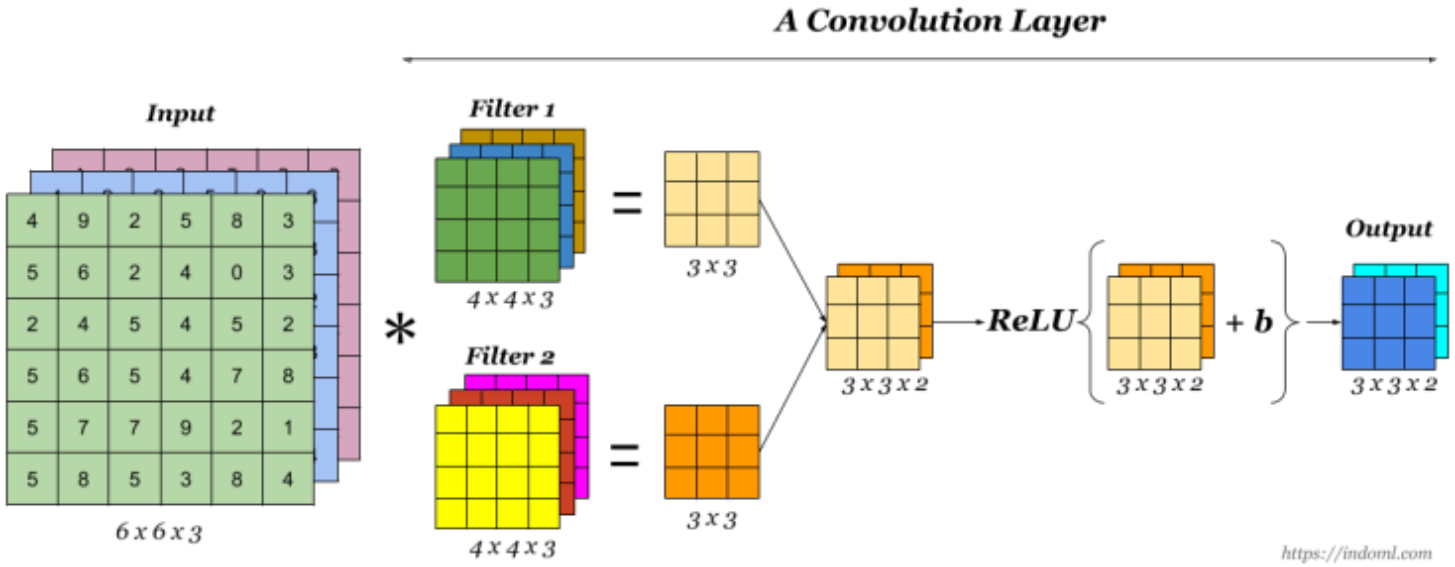
- Número de filtros
- Canales
- Tamaño
- Paso
- Padding
- Activación



https://indoml.com/2018/03/07/student-notes-convolutional-neural-networks-cnn-introduction/

Capa convolucional

- Número de filtros
- Canales
- Tamaño
- Paso
- Padding
- Activación

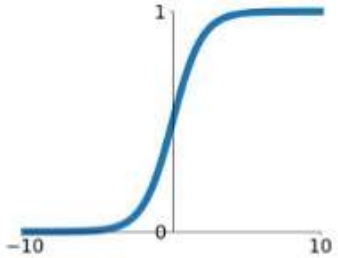


https://indoml.com/2018/03/07/student-notes-convolutional-neural-networks-cnn-introduction/

Funciones de Activación

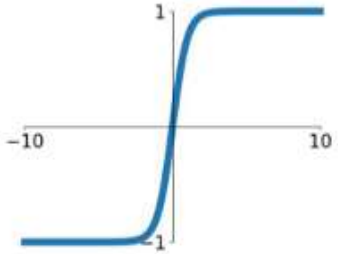
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



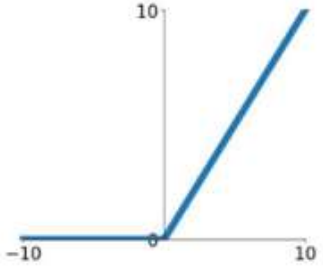
tanh

$$\tanh(x)$$



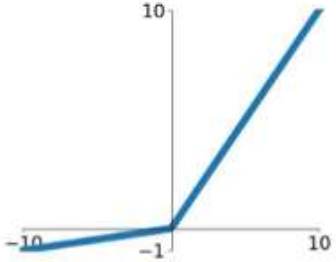
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$

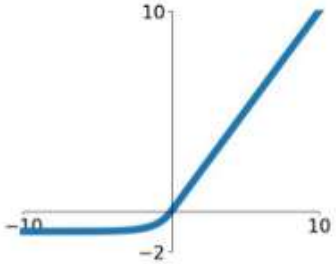


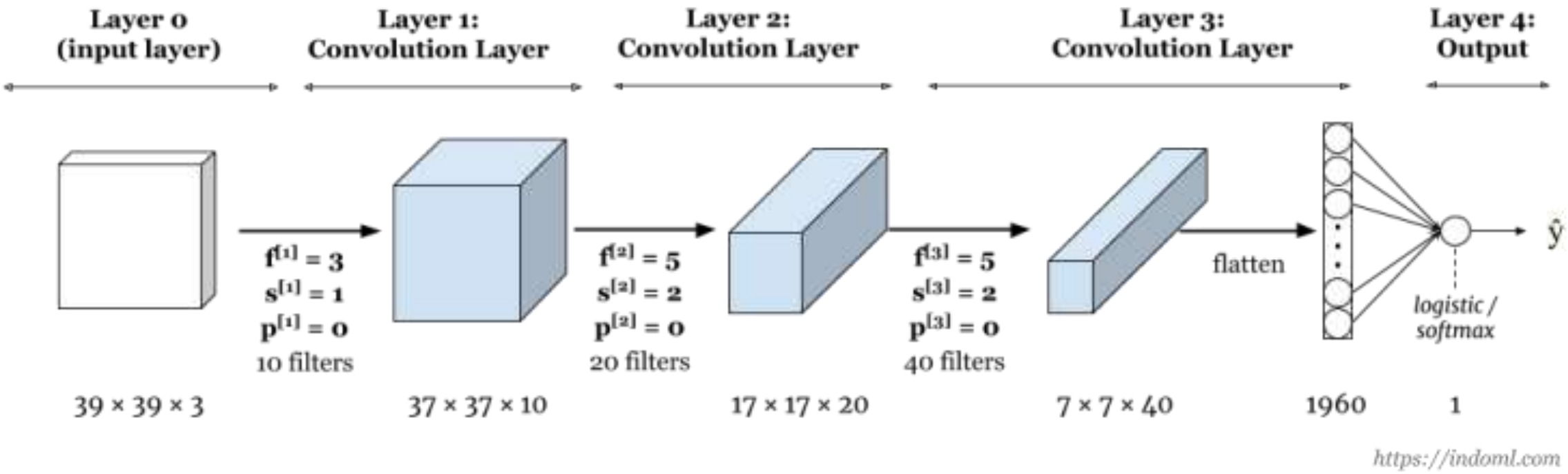
Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$

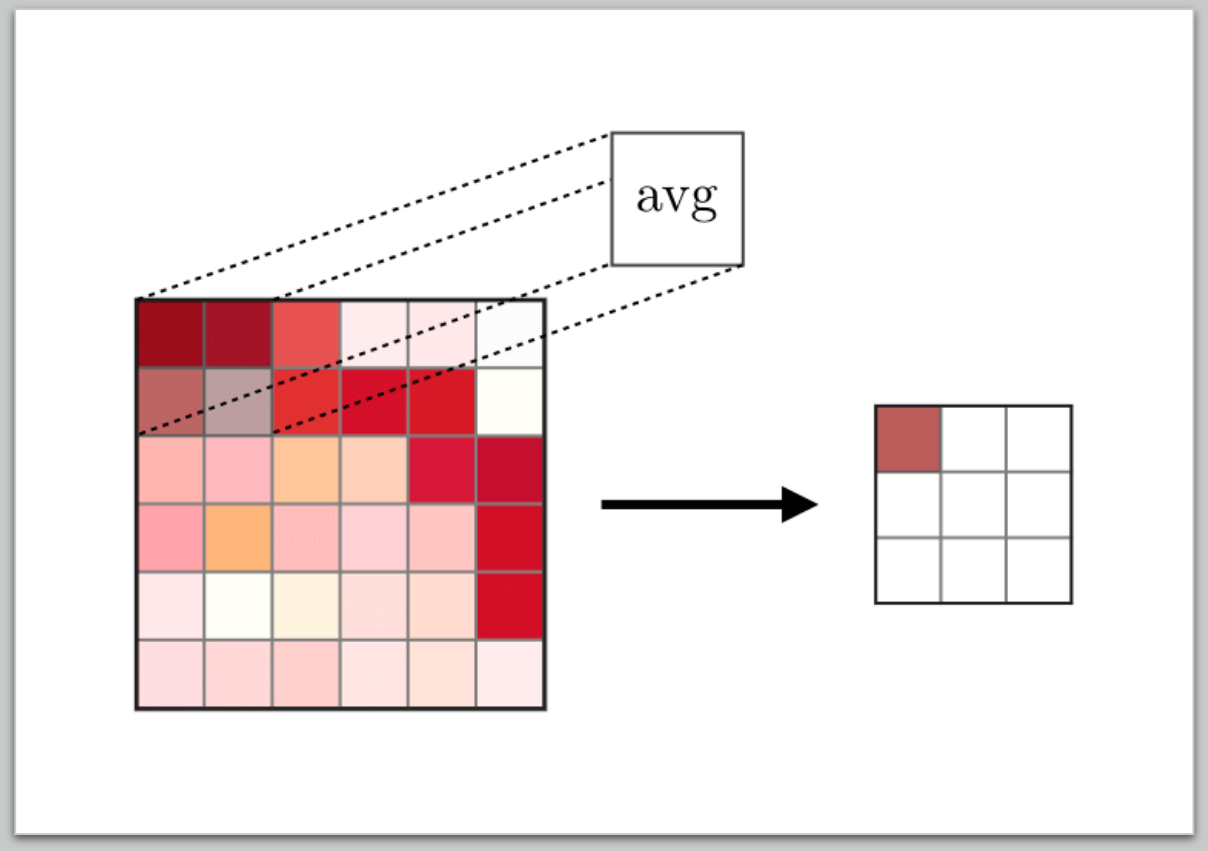


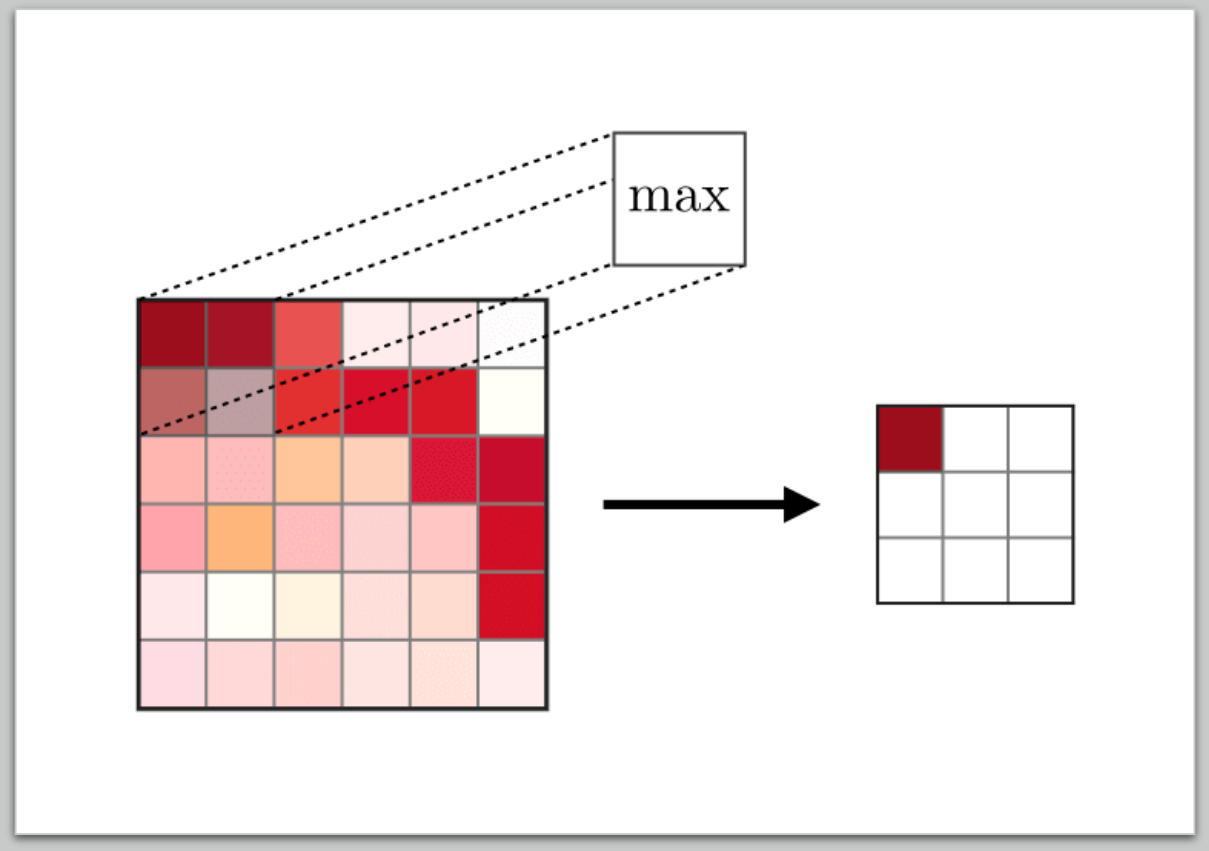


https://indoml.com/2018/03/07/student-notes-convolutional-neural-networks-cnn-introduction/

Pooling

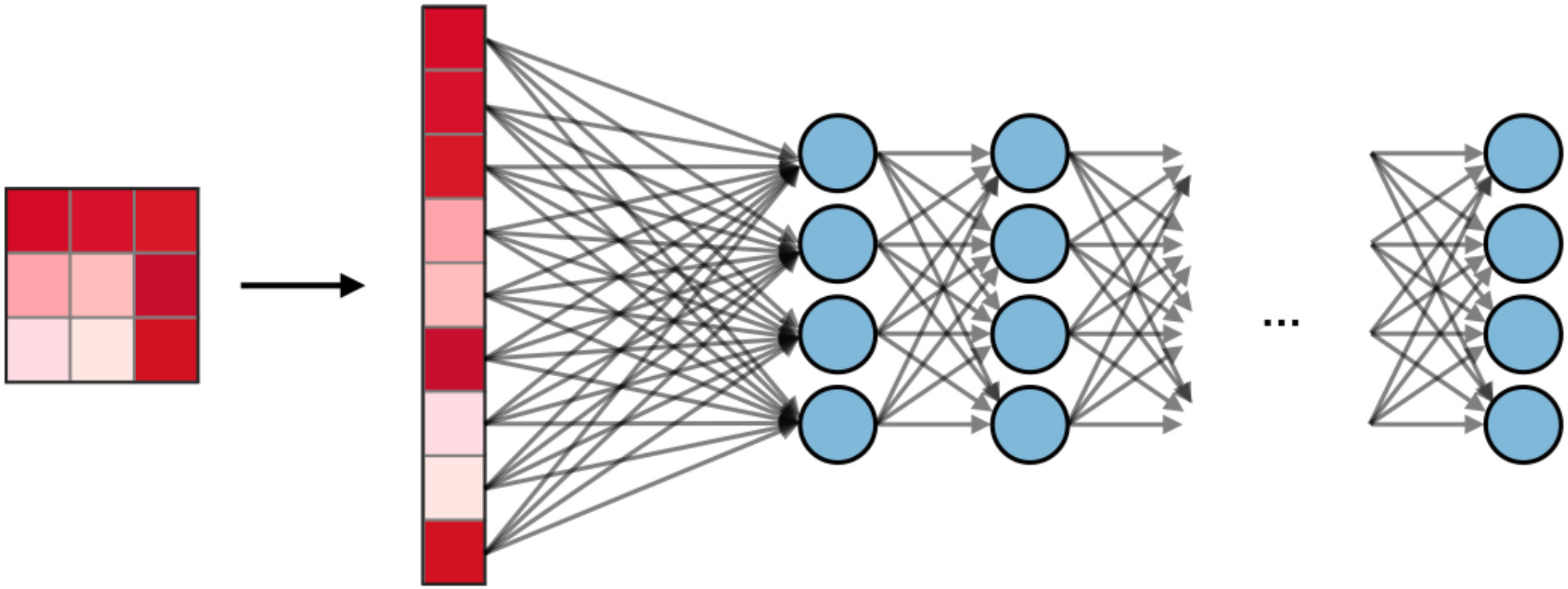
- Resume estadísticamente las salidas cercanas, reduciendo las dimensiones
- Por lo general se usa el promedio o el máximo (maxpooling)
- Tamaño y paso





https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks

Fully connected



https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks