Modelos de clasificación de imágenes

Visión por Computador II

Contenido

A. Batch Normalization

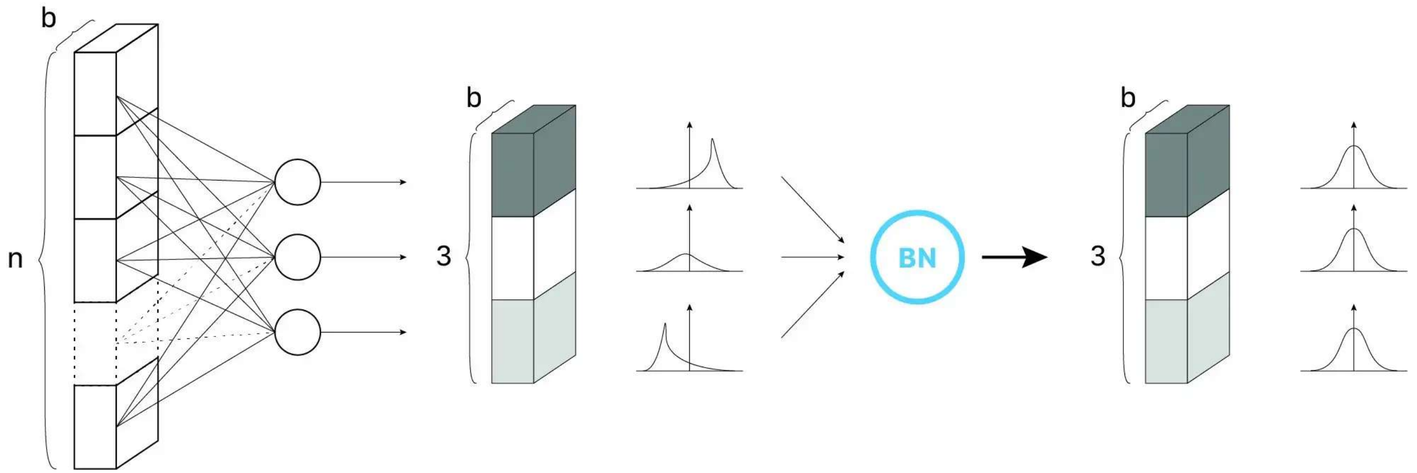
B. Función de activación (Softmax)

C. Cross-Entropy Loss

D. Correlación y convoluciones

Batch Normalization (BN)

- Es un método que hace que la formación sea más rápida y estable
- Normalizar los vectores de activación de las capas ocultas utilizando la media y la varianza del bache actual
- Aplicado justo antes (o justo después) de la función no lineal.
- Se utiliza en casi todas las arquitecturas CNN



Primer paso de la normalización por baches. Ejemplo de una capa oculta de 3 neuronas, con un bache de tamaño b. Cada neurona sigue una distribución normal estándar. | Crédito : autor - Diseño : Lou HD

Batch Normalization

BN se calcula de forma diferente durante la fase de entrenamiento y la de prueba

Entrenamiento

- Restar la media del bache
- Dividir por la desviación estándar del bache
- Aplicar una transformación lineal con dos parámetros entrenables (γ, β)
- *Bias* ignorado

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

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

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

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{// normalize}$$

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

Algorithm 1: Batch Normalizing Transform, applied to activation x over a mini-batch.

2015 - Batch Normalization: Accelerating Deep Network Training By Reducing Covariate Shift

Batch Normalization

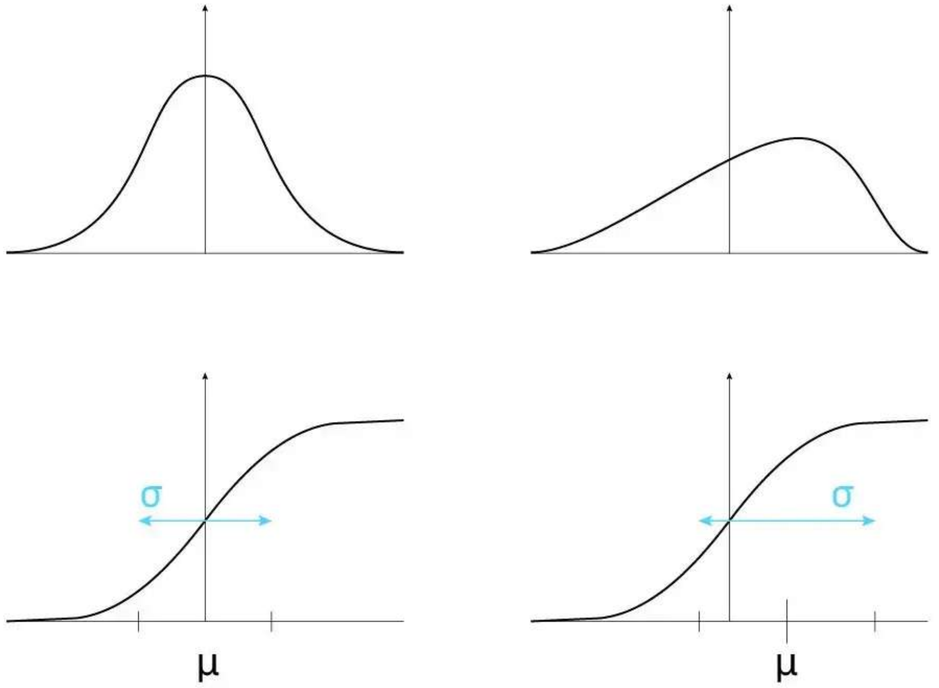
Se entrenan dos parámetros adicionales:

γ : Controla la desviación

β : Controla la media

Detalles

- Normalizar la media y la desviación de una unidad puede reducir su poder expresivo
- $\gamma x_i' + \beta$ mantiene el poder expresivo de la red
- Permiten que la nueva activación tenga cualquier media y desviación
- En esta nueva parametrización solo se depende de (γ, β) y no de la complicada relación de las activaciones lo que facilita el entrenamiento



Aplicación de γ y β. Funciones de activación (Inferior). Resultado después de la normalización (Superior)

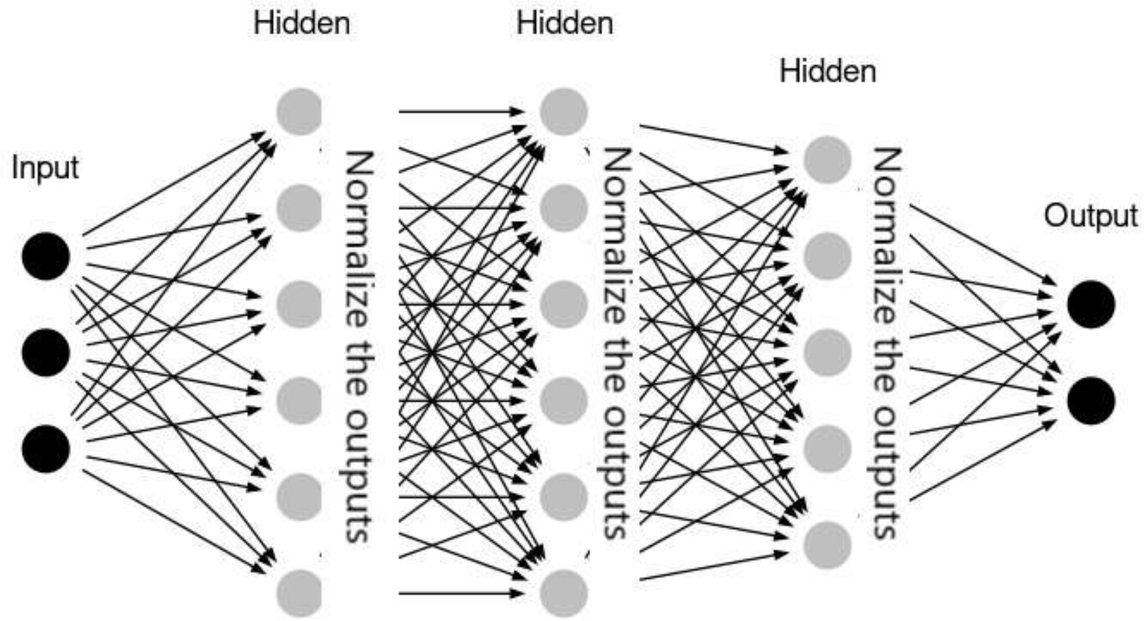
https://towardsdatascience.com/batch-normalization-in-3-levels-of-understanding-14c2da90a338

Batch Normalization

- Durante la evaluación y despliegue del modelo se usa el promedio de (σ, μ) en todo el dataset de entrenamiento

- **En la práctica, consideramos la normalización por baches como una capa estándar**, como un oculta, una capa convolucional, una función de activación

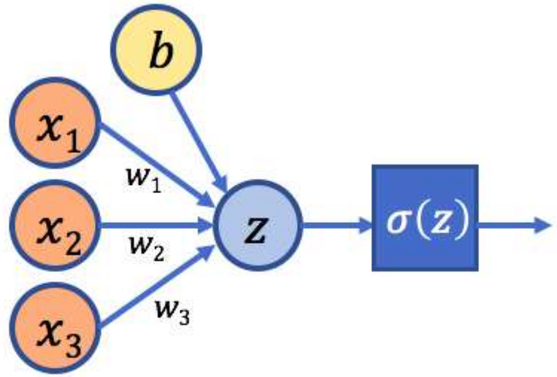
- Cada uno de los frameworks más populares ya tiene implementada una capa de Normalización por baches.



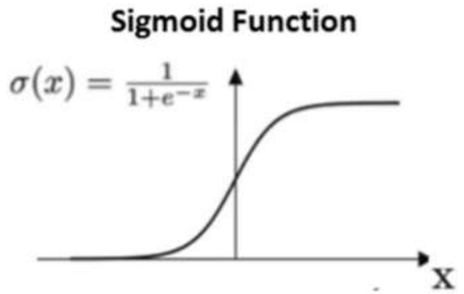
https://deepai.org/machine-learning-glossary-and-terms/batch-normalization

Función de activación (Softmax)

- Recordemos que la regresión logística produce un decimal entre 0 y 1,0
- Por ejemplo, un resultado de regresión logística de 0,8 de un clasificador de correo electrónico sugiere un 80% de posibilidades de que un correo electrónico sea spam y un 20% de que no lo sea.
- La suma de las probabilidades de que un correo electrónico sea o no spam es 100%



Regresión logística

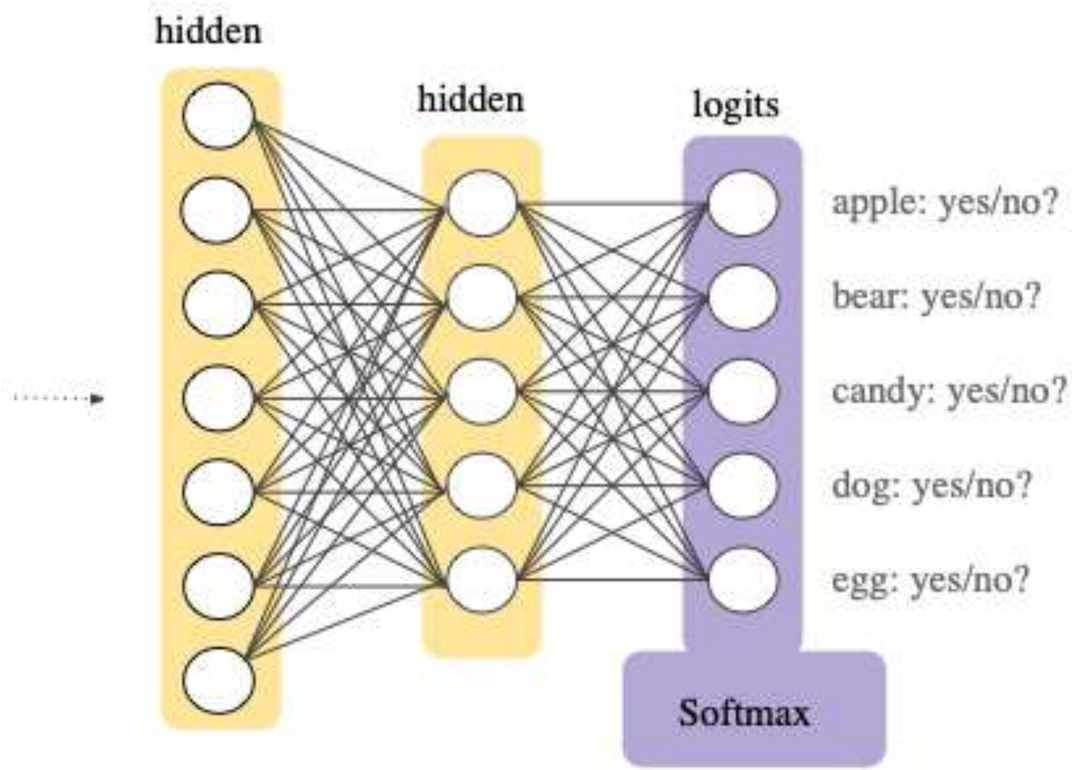


Función de activación sigmoid

https://medium.com/@melodious/understanding-deep-neural-networks-from-first-principles-logistic-regression-bd2f01c9e263
https://machinelearninggeek.com/activation-functions/

Softmax

- Softmax amplía esta idea al problema multiclase.
- Asigna probabilidades decimales a cada clase en un problema multiclase.
- Esas probabilidades deben sumar 1. Esta restricción adicional ayuda a que el entrenamiento converja más rápido.
- Softmax se implementa mediante una capa antes de la capa de salida.
- La capa Softmax debe tener el mismo número de nodos que la capa de salida.



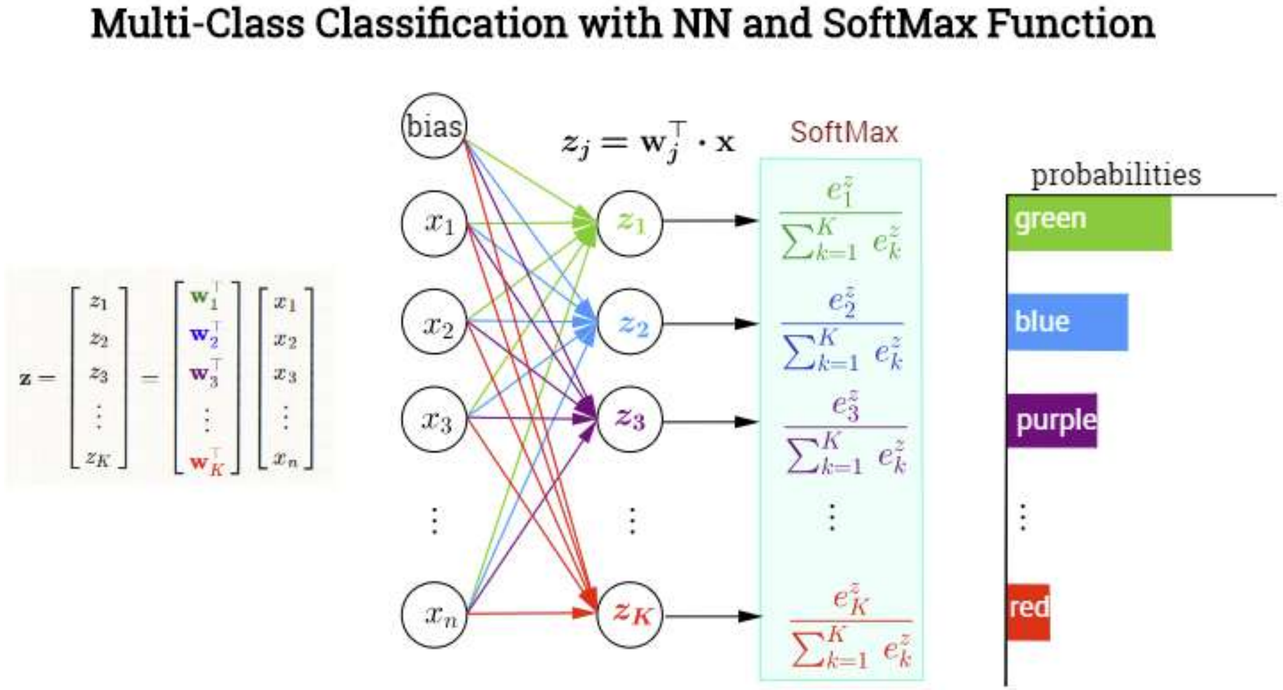
https://developers.google.com/machine-learning/crash-course/multi-class-neural-networks/softmax

Softmax

Función:

$$\phi(z)_i = \frac{e_i^z}{\sum_{j=1}^{k} e_j^z}$$

- Aplica la función exponencial a cada elemento del vector z y normaliza estos valores dividiéndolos por la suma de todos los exponenciales
- Garantiza que la suma de los componentes del vector de salida sea 1.
- Softmax es bastante barato cuando el número de clases es pequeño, pero resulta muy costos cuando el número de clases aumenta.

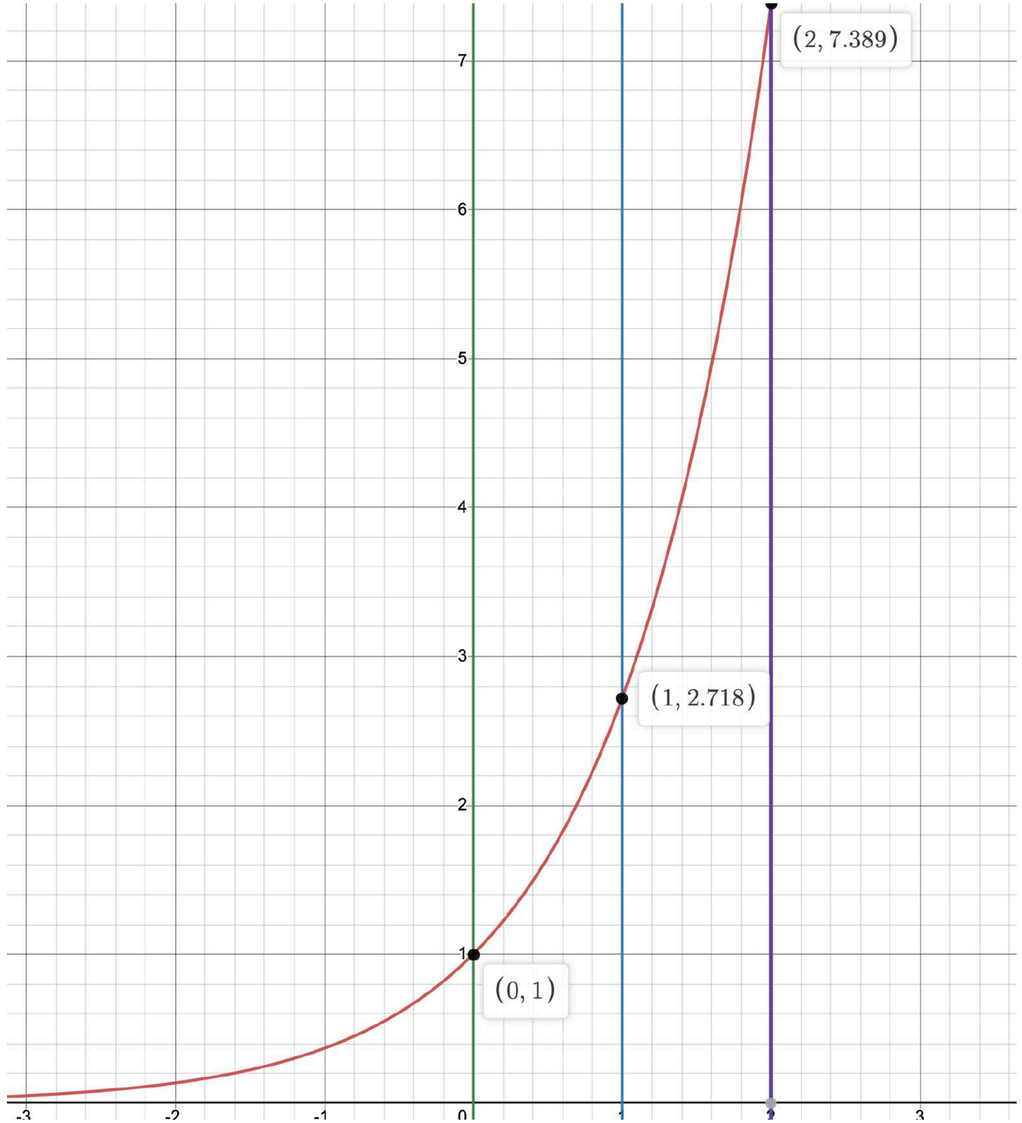


https://rinterested.github.io/statistics/softmax.html

Softmax

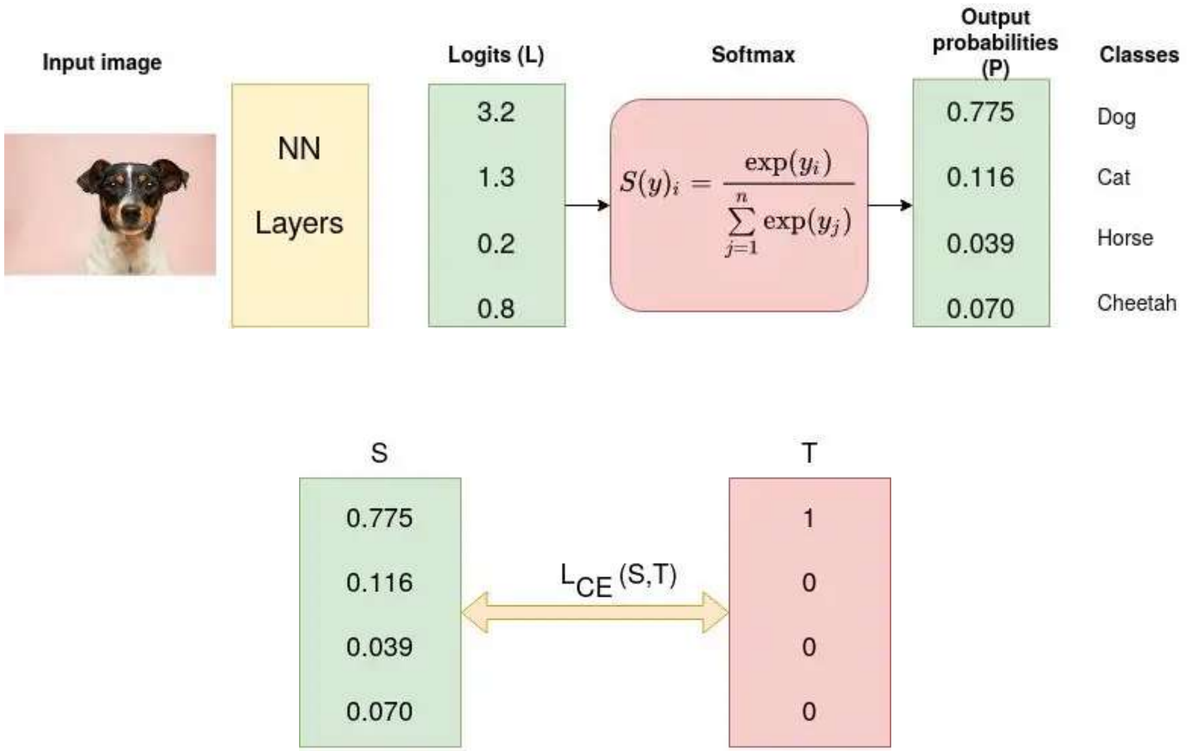
Propiedades

- Normaliza los datos (genera una distribución de probabilidad adecuada).
- Es diferenciable. Una función "*hardmax*" (es decir, argmax) no es diferenciable.
- Aumentamos enormemente la probabilidad de la puntuación más alta y disminuimos la probabilidad de las puntuaciones más bajas



Cross-Entropy loss

- *Cross-Entropy loss* es una función de costo muy importante utilizada en los modelos de clasificación múltiple
- La función está altamente relacionada con el uso de la activación *Softmax*
- La finalidad es tomar las probabilidades de salida P y medir la distancia respecto a los valores de verdad T



https://towardsdatascience.com/cross-entropy-loss-function-f38c4ec8643e

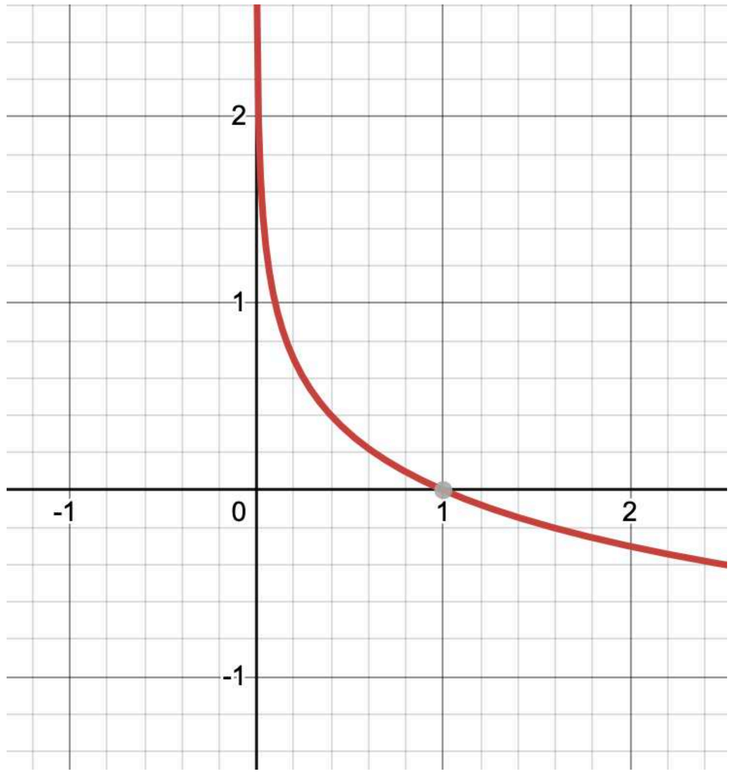
Cross-Entropy

La entropía de una variable aleatoria es el nivel medio de "información", **"sorpresa" o "incertidumbre"** inherente a los posibles resultados de la variable.

$$H(X) = -\sum_{x} p(x) \log p(x)$$

Suma ponderada por la función logarítmica de la probabilidad de cada clase por la probabilidad de cada clase

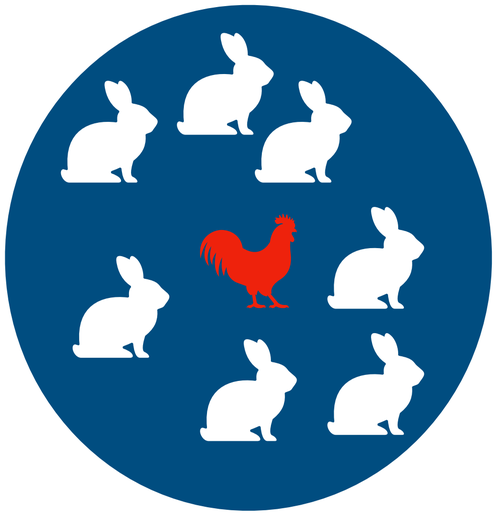
Penaliza probabilidades bajas intermedias



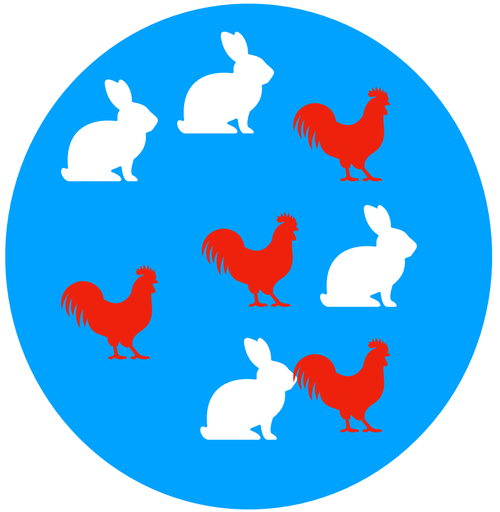
$$y = -\log(x)$$

Cross-Entropy

- Incertidumbre de la posible clase (resultado) al tomar una muestra de una distribución
- Cuanto mayor sea el valor de la entropía, mayor será la incertidumbre de la distribución de probabilidad y cuanto menor sea el valor, menor será la incertidumbre.



D1



D2

Calcular el Cross-Entropy de las dos distribuciones de la imagen

Cross-Entropy

D1

$$H(X) = -\sum_{x} p(x)\log(p(x))$$

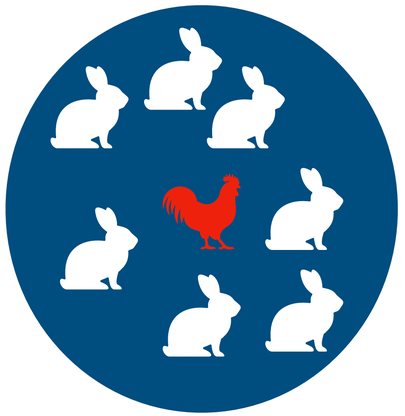
$$H(X) = -\left[\frac{1}{8}\log\left(\frac{1}{8}\right) + \frac{7}{8}\log\left(\frac{7}{8}\right)\right]$$

$$H(X) = 0.06$$

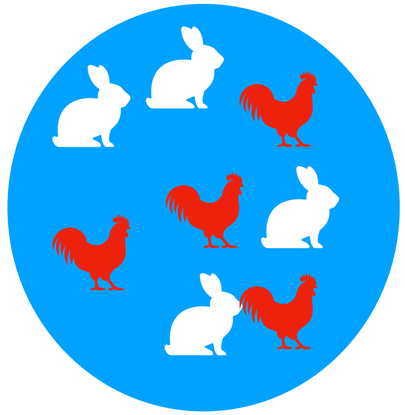
D2

$$H(X) = -\left[\frac{1}{2}\log\left(\frac{1}{2}\right) + \frac{1}{2}\log\left(\frac{1}{2}\right)\right]$$

$$H(X) = 0.3$$



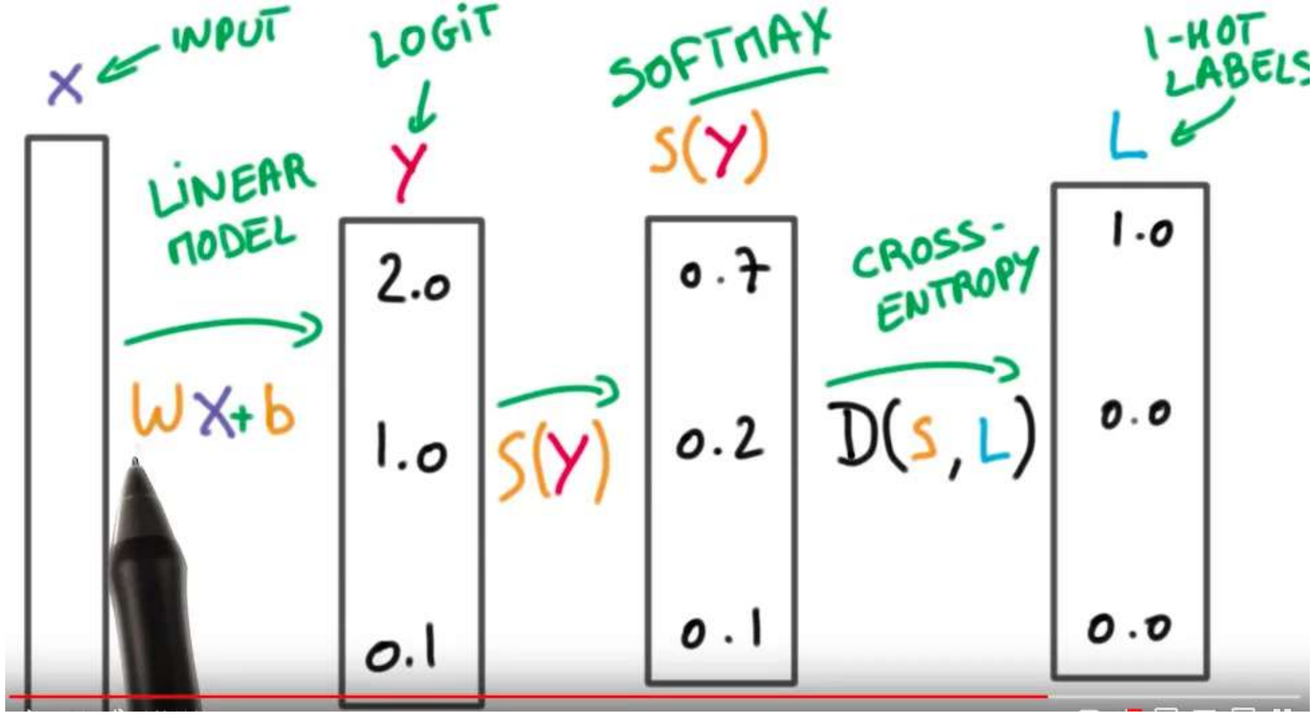
Baja cross-entropía



Alta cross-entropía

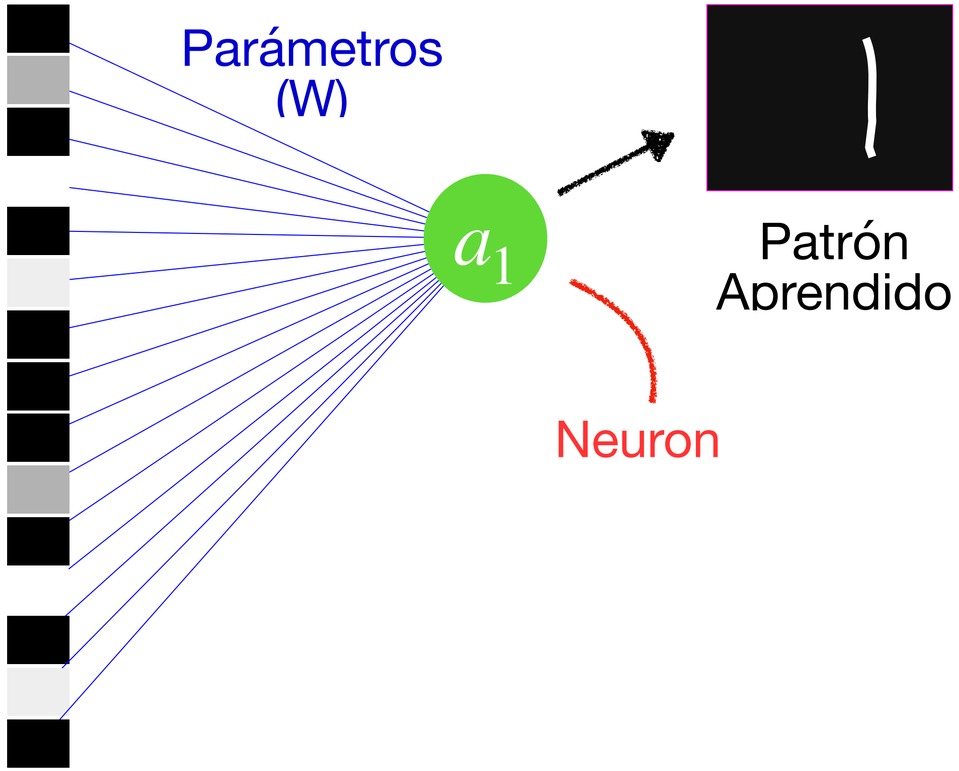
Cross-Entropy loss

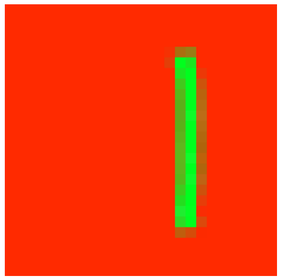
- Cada probabilidad de clase predicha se compara con la salida real de clase deseada 0 o 1
- Se calcula una costo que penaliza la probabilidad en función de lo lejos que esté del valor real esperado.
- La penalización es de naturaleza logarítmica, lo que da una puntuación grande para las diferencias grandes cercanas a 1 y pequeña para las diferencias pequeñas que tienden a 0.
- $L_{CE} = -\sum_{i}^{n} y \log(\hat{y})$



https://majianglin2003.medium.com/loss-functions-in-pytorch-5c06fdcaf0ff

Introducción a Redes Neuronales Convolucionales (CNN)





$$\Theta$$

$$\phi(\theta_1 a1 + \theta_2 a_2 + \theta_3 a_3 \ldots + \theta_n a_n + b)$$

Qué deberíamos hacer si en una Red Neuronal quisiéramos encontrar el mismo patrón en diferentes partes de la imagen?

Filtros lineales

Inicial

Box blur

- Colección de valores de píxeles en las proximidades de un píxel dado para determinar su valor de salida final
- Se utiliza para filtrar imágenes con el fin de añadir un desenfoque, acentuar los bordes o eliminar el ruido.
- Suma ponderada de píxeles circundantes

Cross-Correlation

La correlación es el proceso de mover una máscara de filtro, a menudo denominada kernel, sobre la imagen y calcular la suma de productos en cada posición.

$$g = f \otimes h$$

$$g(i) = \sum_k f(i+k)h(k)$$

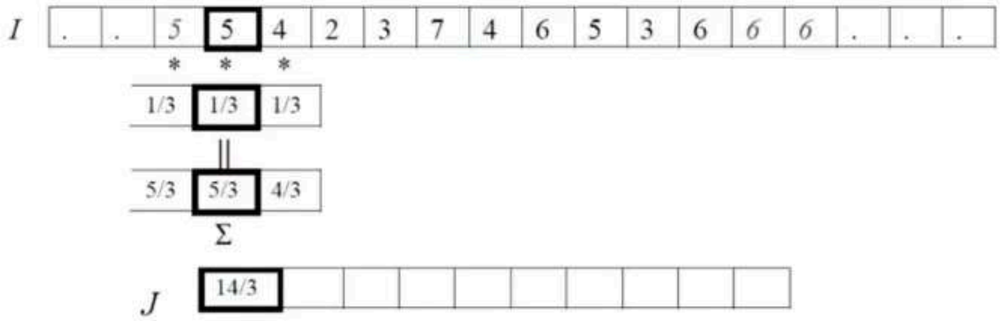


Figure 1.Cross-Correlation in 1-D

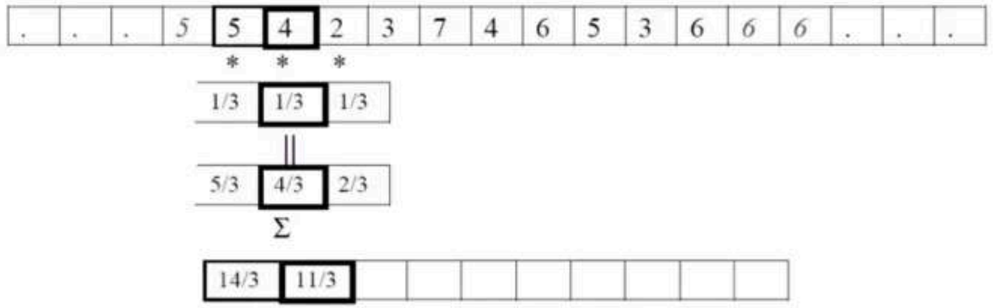


Figure 2.Cross-Correlation in 1-D

https://towardsdatascience.com/convolution-vs-correlation-af868b6b4fb5

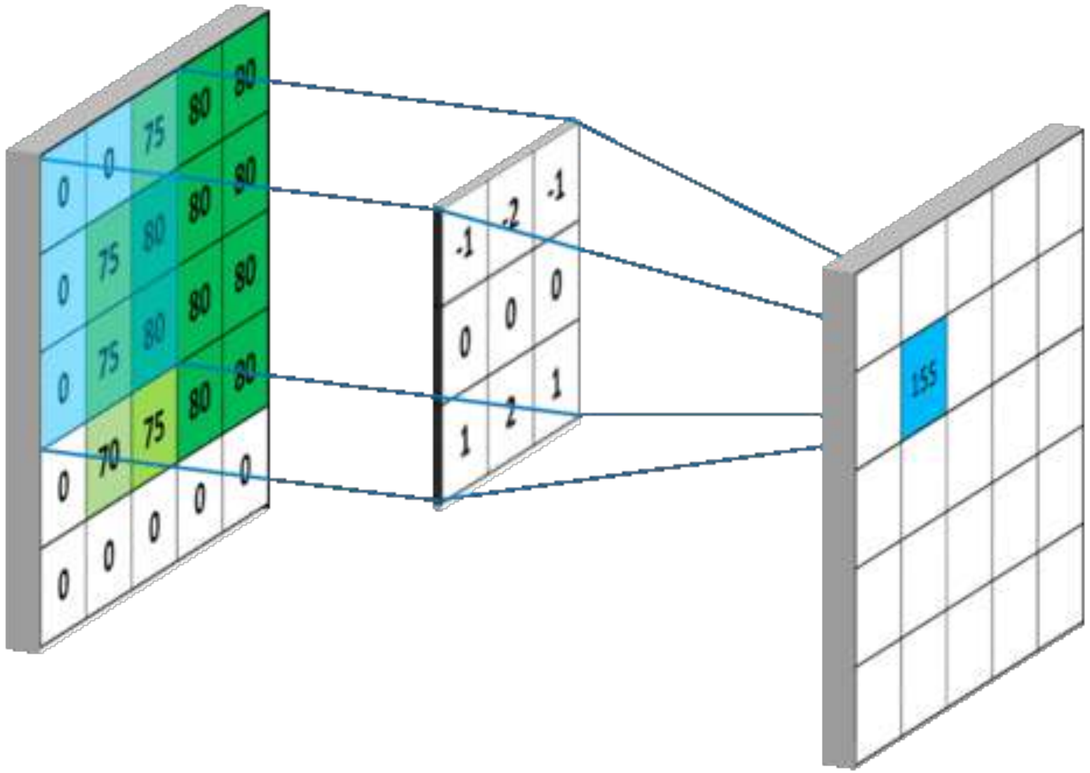
Filtros lineales

Correlación 2D

- Filtro lineal, en el que el valor de un píxel de salida se determina como una suma ponderada de los valores de los píxeles de entrada

$$g(i,j) = \sum_{k,l} f(i+k, j+l)h(k,l)$$

- Los valores del kernel o máscara $h(k,l)$ son llamados coeficientes del filtro



https://mlnotebook.github.io/post/CNN1/

Filtros lineales

$$g = f \otimes h$$

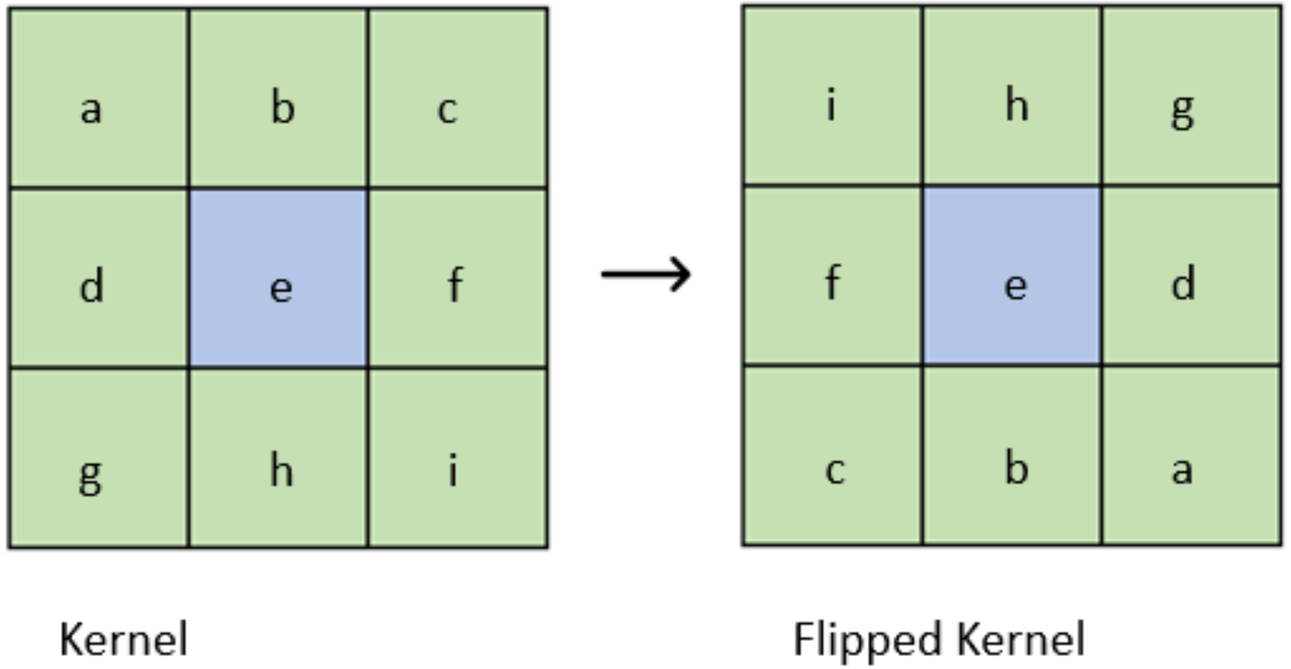
$$h = \frac{1}{25}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Box filter kernel

https://docs.nvidia.com/vpi/algo_box_filter.html

Convolution

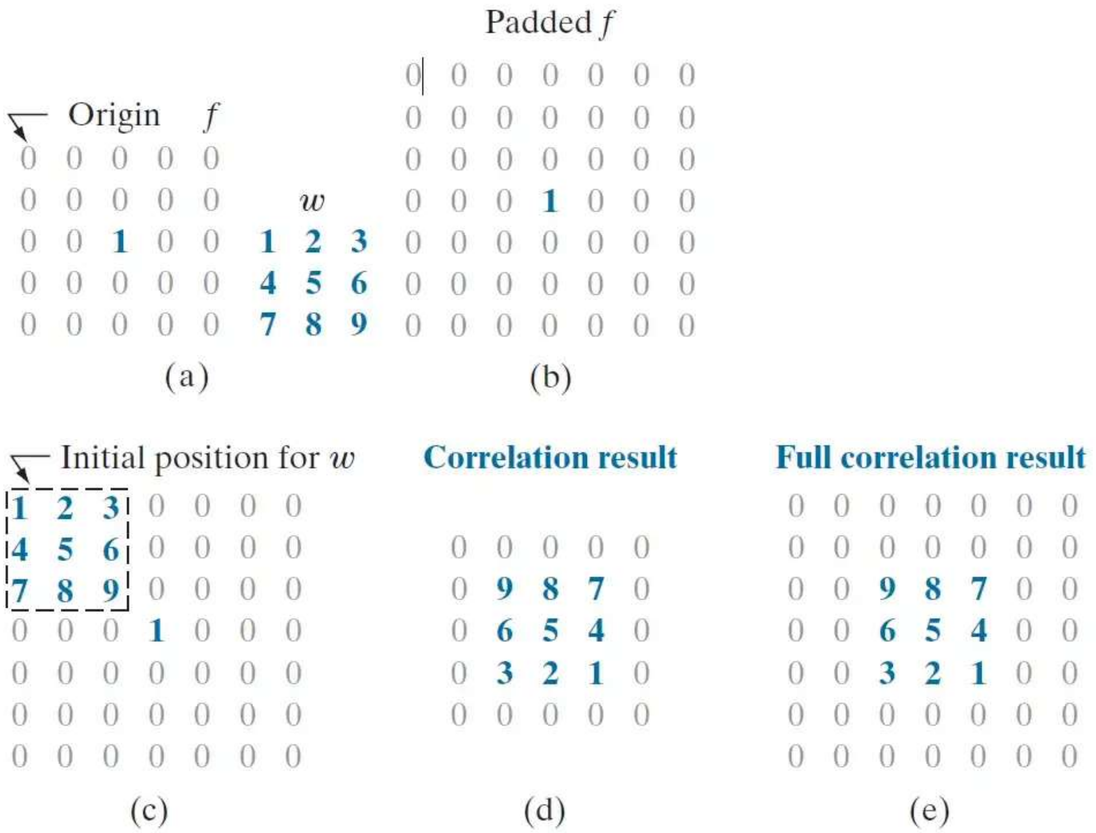
- Una variación común a la correlación es la *convolución*
- La convolución es igual que la correlación, solo que antes de correlacionar volteamos el filtro.
- En la operación de convolución, primero se voltea el núcleo un ángulo de 180 grados y luego se aplica a la imagen



https://medium.com/@timothy_terati/image-convolution-filtering-a54dce7c786b

Convolution vs Correlation

- Cuando se aplica sobre un impulso unitario discreto (una matriz 2D de todos ceros y un solo 1)
- Produce un resultado que es una copia del filtro pero girado un ángulo de 180 grados.



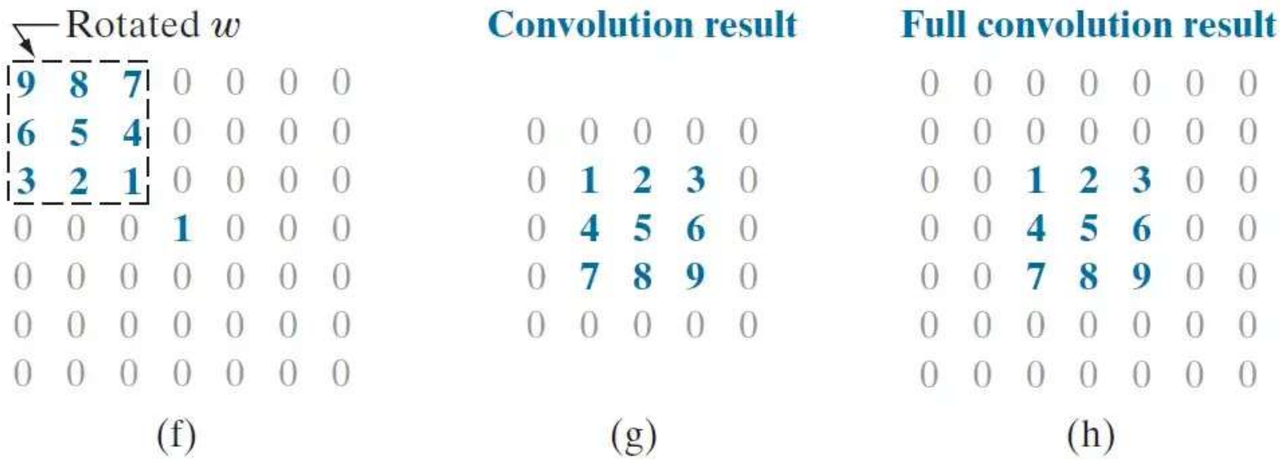
https://towardsdatascience.com/convolution-vs-correlation-af868b6b4fb5

Convolution vs Correlation

Si giramos previamente el filtro y realizamos la misma operación deslizante de suma de productos, obtenemos el resultado deseado.

$$g = f * h$$

$$g(i,j) = \sum_{k,l} f(k,l)h(i-k, j-l)$$

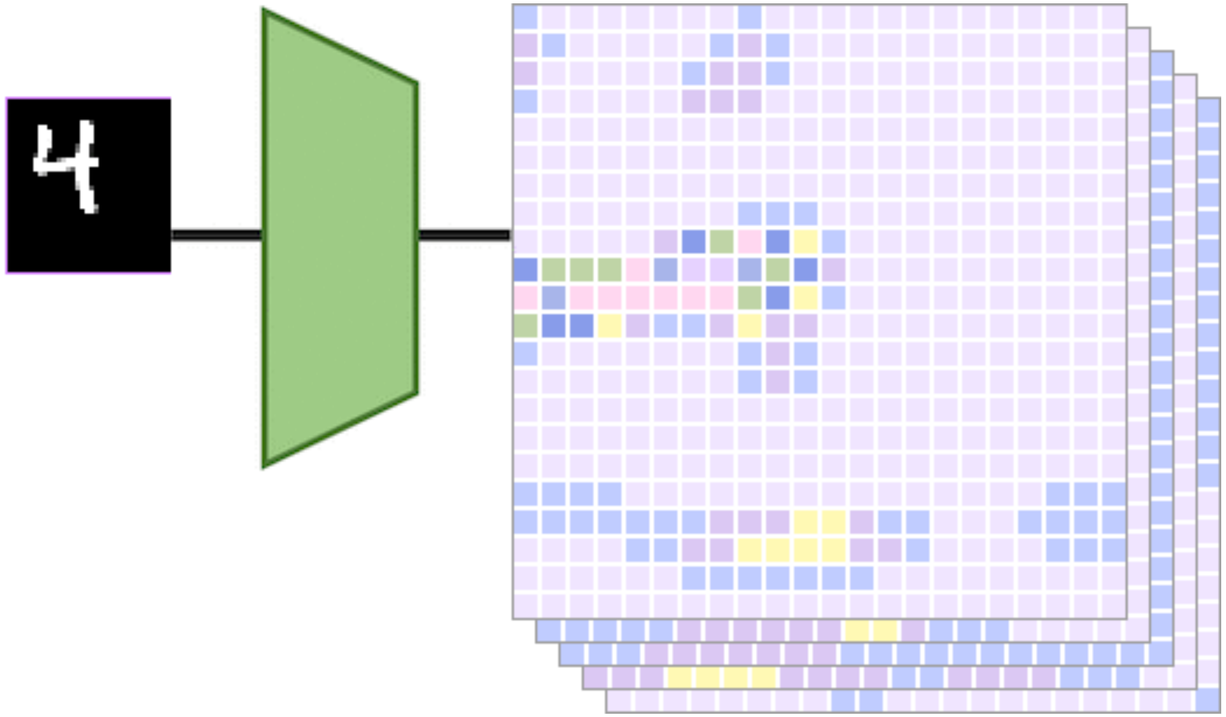


https://towardsdatascience.com/convolution-vs-correlation-af868b6b4fb5

Convolution

Características

- Presenta una propiedad de *“translation equivariant”* (invariancia de cambio)
- Preserva la estructura algebraica de una transformación. Cuando se traslada la entrada, produce un mapeo trasladado
- Presenta propiedades conmutativa y asociativa.



https://chriswolfvision.medium.com/what-is-translation-equivariance-and-why-do-we-use-convolutions-to-get-it-6f18139d4c59

Convolution

Original

Gaussian blur

0	0	0	5	0	0	0
0	5	18	32	18	5	0
0	18	64	100	64	18	0
5	32	100	100	100	32	5
0	18	64	100	64	18	0
0	5	18	32	18	5	0
0	0	0	5	0	0	0

Line detection

-1	-1	-1
2	2	2
-1	-1	-1

Horizontal lines

-1	2	-1
-1	2	-1
-1	2	-1

Vertical lines

-1	-1	2
-1	2	-1
2	-1	-1

45 degree lines

2	-1	-1
-1	2	-1
-1	-1	2

135 degree lines

Edge detection

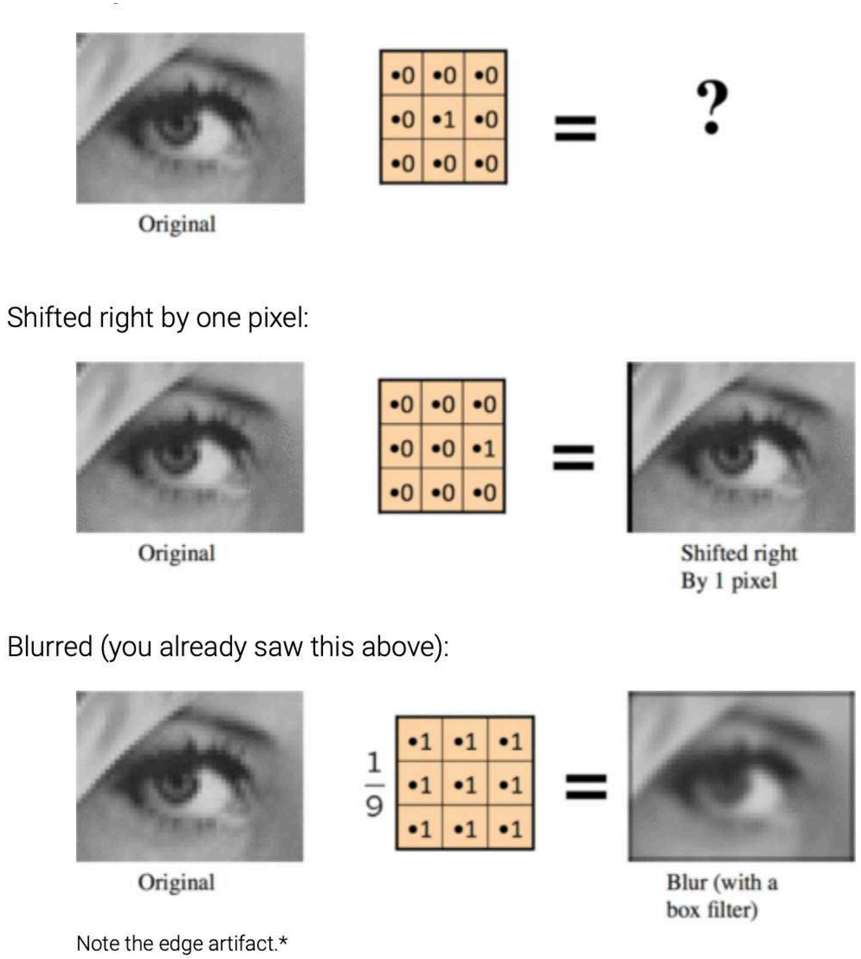
-1	-1	-1
-1	8	-1
-1	-1	-1

https://aishack.in/tutorials/image-convolution-examples/

2D Convolutions

Resumen

- Muchos filtros pueden expresarse como convolución 2D como el suavizado y la nitidez de imágenes y la detección de bordes.
- La convolución en 2D opera sobre dos matrices, una de ellas es la imagen de entrada y la otra como filtro
- **Expresa la cantidad de superposición de una función cuando se desplaza sobre otra función**, ya que la imagen de salida se produce deslizando el núcleo sobre la imagen de entrada.

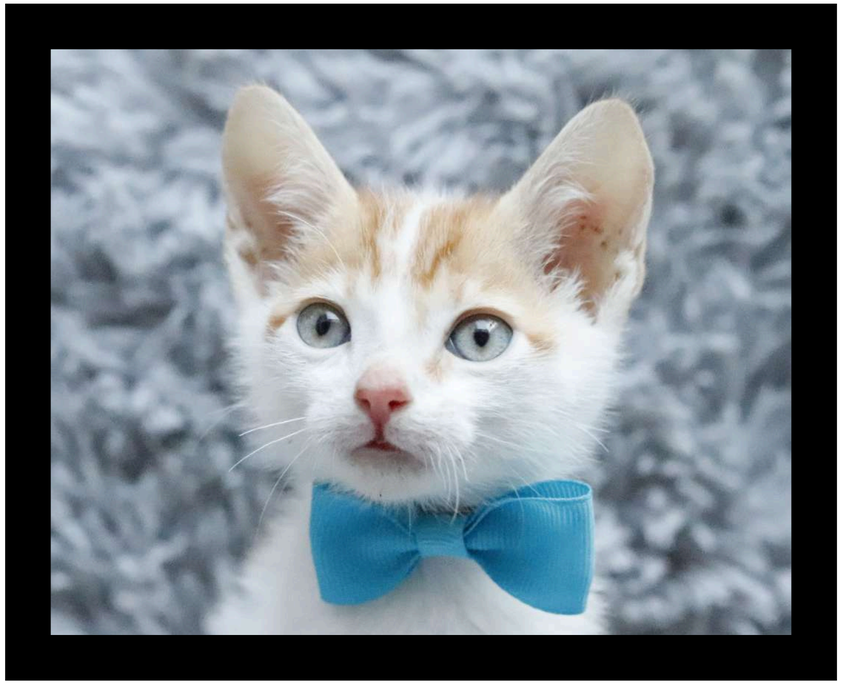


https://ai.stanford.edu/~syyeung/cvweb/tutorial1.html

Parámetros de la Convolución

Padding (Efectos del borde)

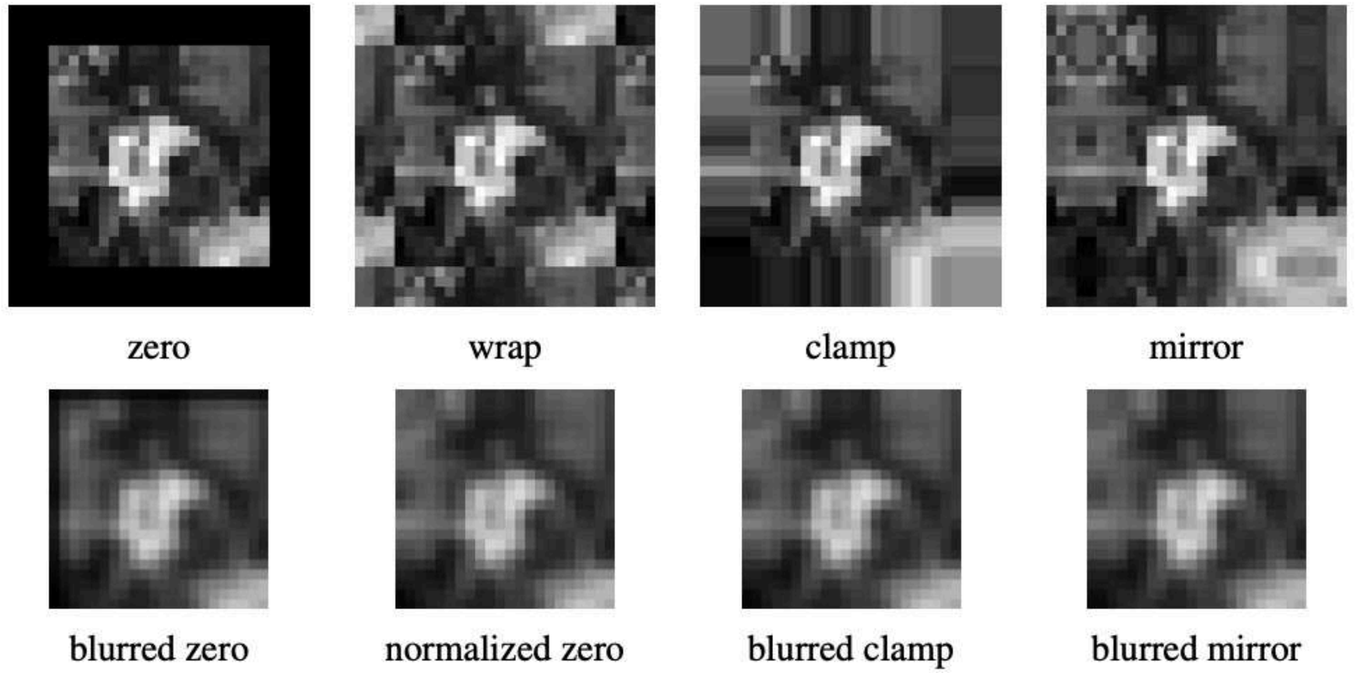
- Después de cada operación de convolución, el tamaño de la imagen original se reduce.
- Cuando el kernel se mueve sobre las imágenes originales, toca el borde de la imagen menos veces y toca el centro de la imagen más veces y se solapa también en el centro.
- Por lo tanto, las características de las esquinas de cualquier imagen o en los bordes no se utilizan mucho en la salida.



Ejemplo de zero padding

Parámetros de la Convolución

- **Cero:** establece todos los píxeles fuera de la imagen de origen a 0 (una buena opción para imágenes de recorte con malla alfa);
- **Constante (color de borde):** establece todos los píxeles fuera de la imagen de origen a un valor de borde especificado;
- **Clamp (replicar o sujetar al borde):** repite indefinidamente los píxeles del borde;
- **Reflejo:** reflejar píxeles a través del borde de la imagen;
- **Extensión:** extender la señal restando la versión reflejada de la señal del valor del píxel del borde.



Parámetros de la Convolución

Tamaño del filtro

- El tamaño del kernel define el campo de visión de la convolución.
- Impares para ser simétricos (3x3, 5x5) (comúnmente)
- Una elección común para 2D es de 3x3 píxeles

0	13	78	60	7	74
89	100	81	84	78	9
49	49	0	1	2	8
8	58	23	0	68	135
28	80	4	59	167	236
24	2	8	39	34	5

5x5 Kernel

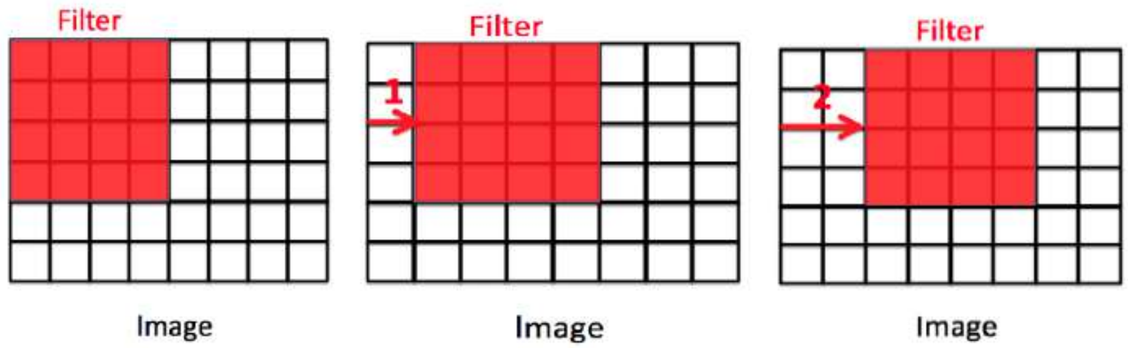
0	13	78	60	7	74
89	100	81	84	78	9
49	49	0	1	2	8
8	58	23	0	68	135
28	80	4	59	167	236
24	2	8	39	34	5

3x3 Kernel

Parámetros de la Convolución

Stride

- Usado para la compresión de imágenes
- Modifica la cantidad de movimiento sobre la imagen
- Por ejemplo, si el *stride* de una red neuronal se fija en 1, el filtro se moverá un píxel.



Kernel 4x4 **Stride: 1** **Stride: 2**

https://www.codingninjas.com/codestudio/library/convolution-layer-padding-stride-and-pooling-in-cnn

Ejercicio

Crear un script para encontrar a Waldo