Redes Convolucionales

Aprendizaje de Características

Diego R. Cabrera Mendieta

[1]Departamento de Ingeniería Mecánica
Universidad Politécnica Salesiana

[2]Departamento de Ciencias de la Computación e Inteligencia Artificial
Universidad de Sevilla

Charla de Aprendizaje Automático, 2016

Tabla de Contenidos

Metodología tradicional

- extracción manual de características
- experto o conjunto de expertos
- seleccionar la información relevante
- posterior proceso de aprendizaje

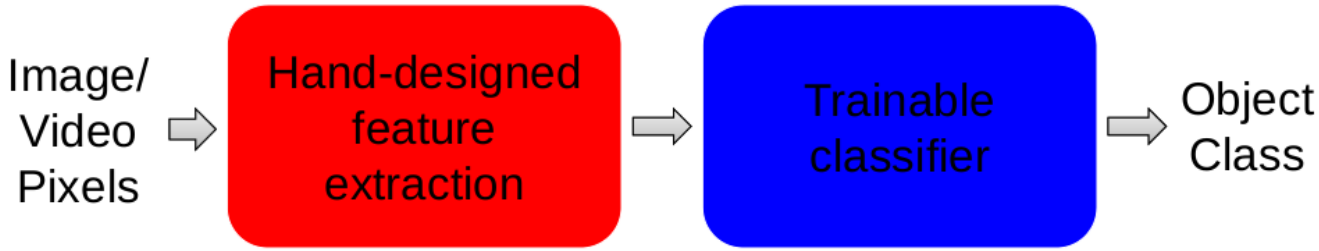


Figure : Metodología tradicional para un sistema de aprendizaje automático

Extracción de características

Proceso

- no es trivial.
- depende de la aplicación
- diseño de técnicas ad-hoc
- proceso es fundamental

Objetivos

- disminuir la cantidad de datos
- disminuir la variabilidad

Tabla de Contenidos

Inconvenientes con los datos-Tamaño

Elemento

- compuesto por bloques más pequeños que lo caracterizan.
- distinto tipo dependiendo del dominio
- ejm, elemento imagen, bloque pixel

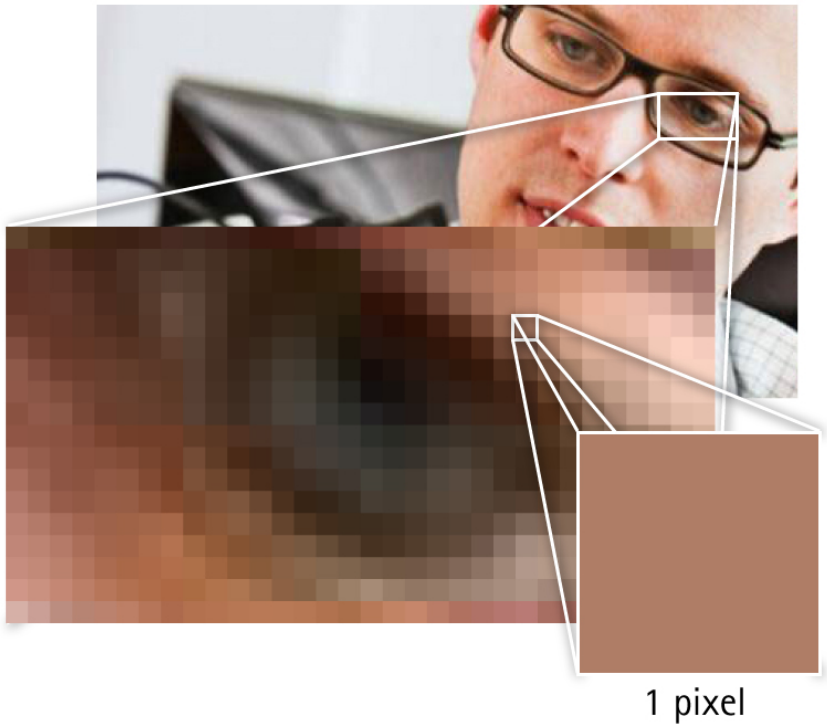


Figure : Imagen y un pixel

Inconvenientes con los datos-Tamaño

Imagen

- tantos pixeles como lo permita el dispositivo de captura
- 640 pixeles de ancho por 480 pixeles de alto
- 3 canales
- 921600 bloques que caracterizan la imagen
- demasiados datos.

Inconvenientes con los datos-Tamaño

Serie de tiempo

- eventos temporales
- conjunto de muestras tomadas una a continuación de otra
- uni-modales o multi-modales
- a 50kS/s
- cada segundo tendremos 50000 muestras.

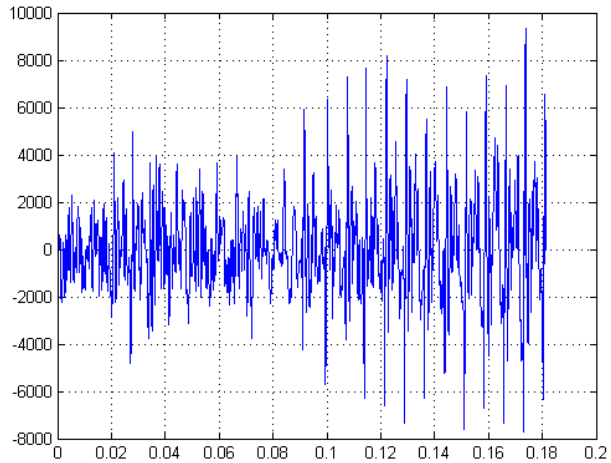


Figure : Serie de tiempo uni-modal proveniente de la medición de la aceleración instantánea en maquinaria industrial

Inconvenientes con los datos-Tamaño

Tradicionalmente

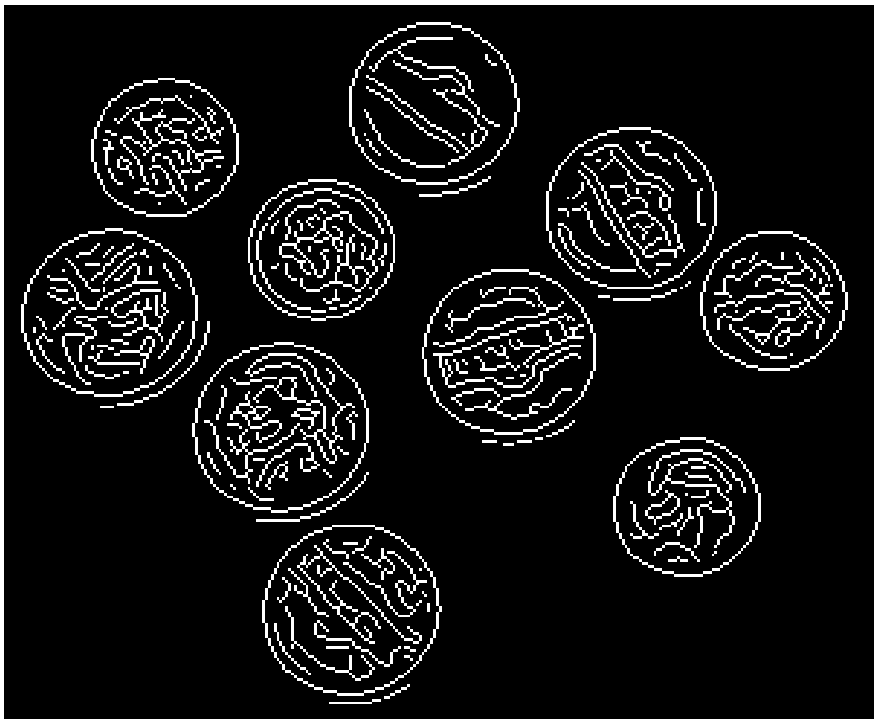
- intratable
- indispensable en la etapa de extracción de características
- ingresar al modelo de aprendizaje solamente información útil.

Ejm: Reconocer monedas-No interesa

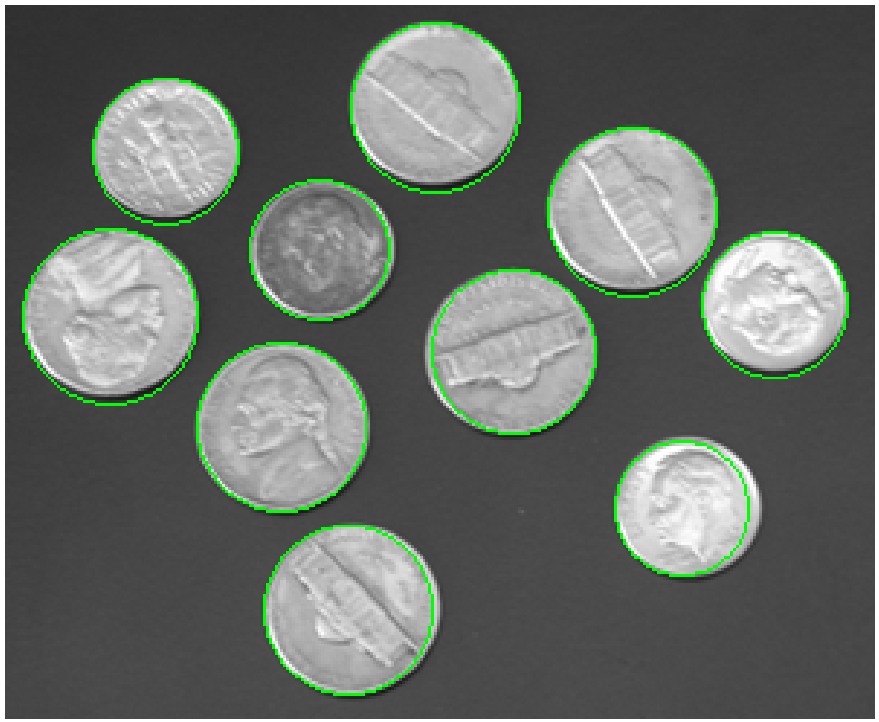
- el color de la imagen.
- el fondo.
- objetos de forma no circular.
- etc.

Inconvenientes con los datos-Tamaño

Solución



(a) Resultado de la aplicación del filtro de canny para la detección de bordes



(b) Resultado de la aplicación del algoritmo de hough luego de canny, superpuesto sobre la imagen original

Figure : Extracción y selección de características en una aplicación de detección de monedas

Inconvenientes con los datos-Variabilidad

Problema

- trasformación de la información
- variaciones en la adquisición de los datos
- variabilidad propia del proceso

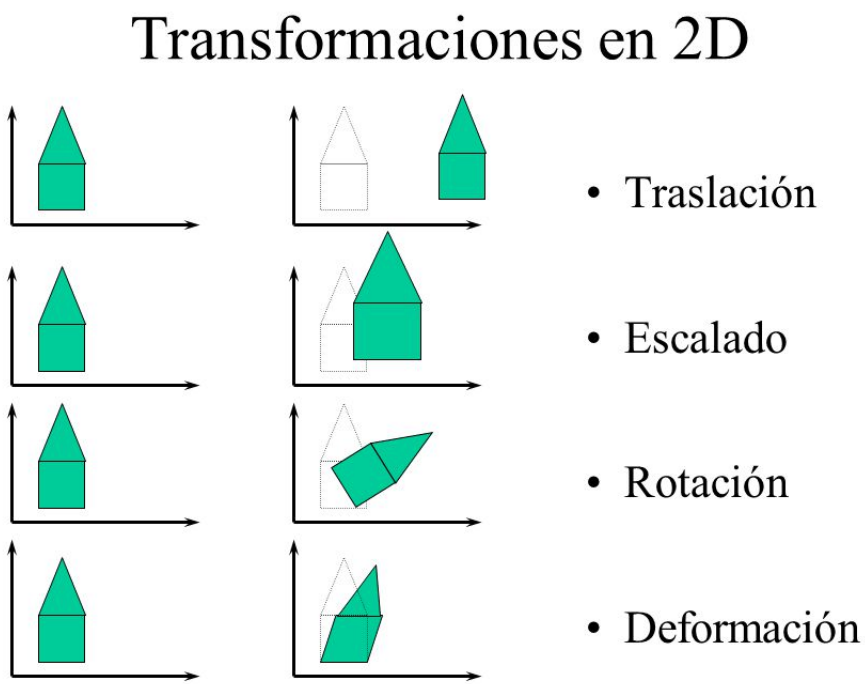


Figure : Transformaciones en 2D de una misma imagen

Inconvenientes con los datos-Variabilidad

Modelos tradicionales

- no son capaces de extraer patrones móviles
- propiedades invariantes a una o varias de las trasformaciones
- variabilidad propia del proceso
- Un ejemplo, algoritmo SIFT

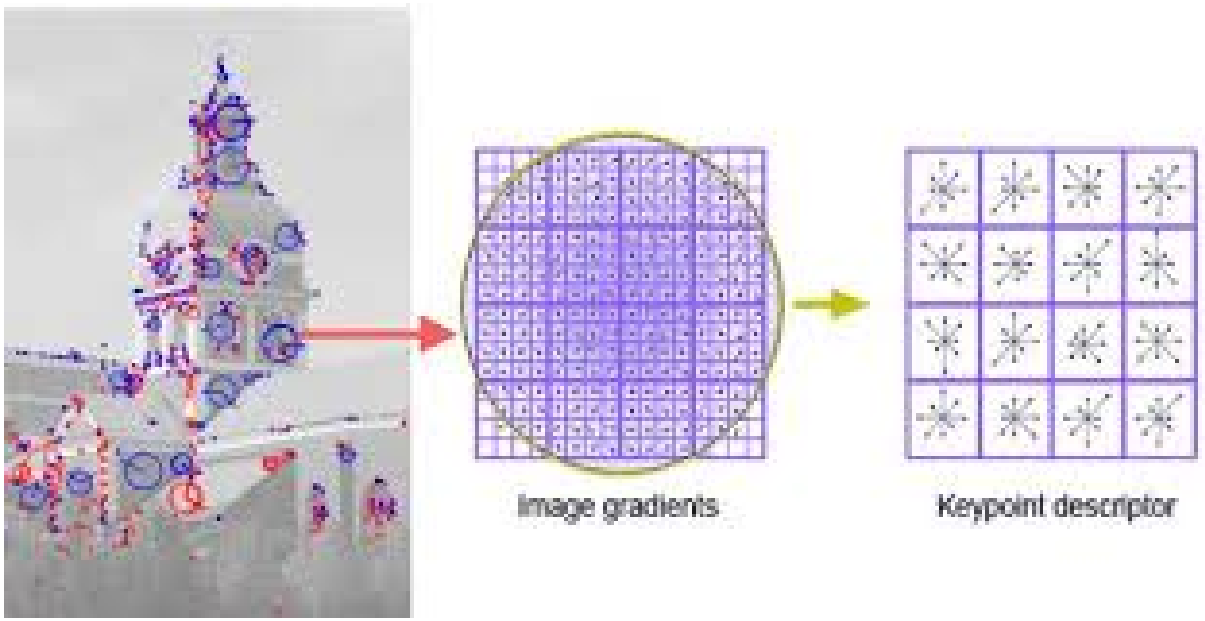


Figure : Ejemplo del algoritmo Scale Invariant Feature Transform, SIFT

Tabla de Contenidos

Aprendizaje de características

Preguntas:

- ¿Qué sucede cuando no se tiene ningún conocimiento a-priori en el área de aplicación?
- ¿Será posible disminuir o eliminar la dependencia del pre-procesamiento de los datos?
- ¿Será posible extraer las características de manera automática según la tarea de aprendizaje en cuestión?

Aprendizaje de características

Metodología propuesta

- enfocada al aprendizaje directo desde los datos
- uso mínimo de pre-procesamiento
- extraer o aprender las características
- reemplaza la fase de extracción manual de características

Figure : Proceso de aprendizaje jerárquico de características

Aprendizaje de características-Jerarquía

Cada bloque intermedio tendrá información mas concisa que describa la entrada

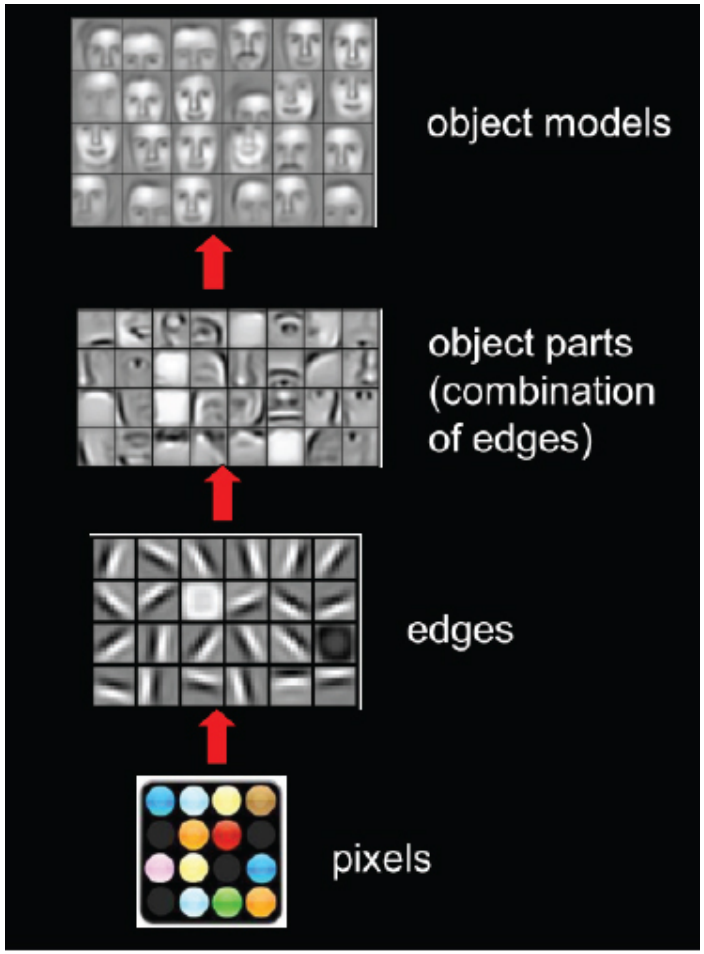


Figure : Ejemplo de un proceso de aprendizaje jerárquico de características

Tabla de Contenidos

Redes neuronales multi-capa

Un primer intento en la extracción automática de características

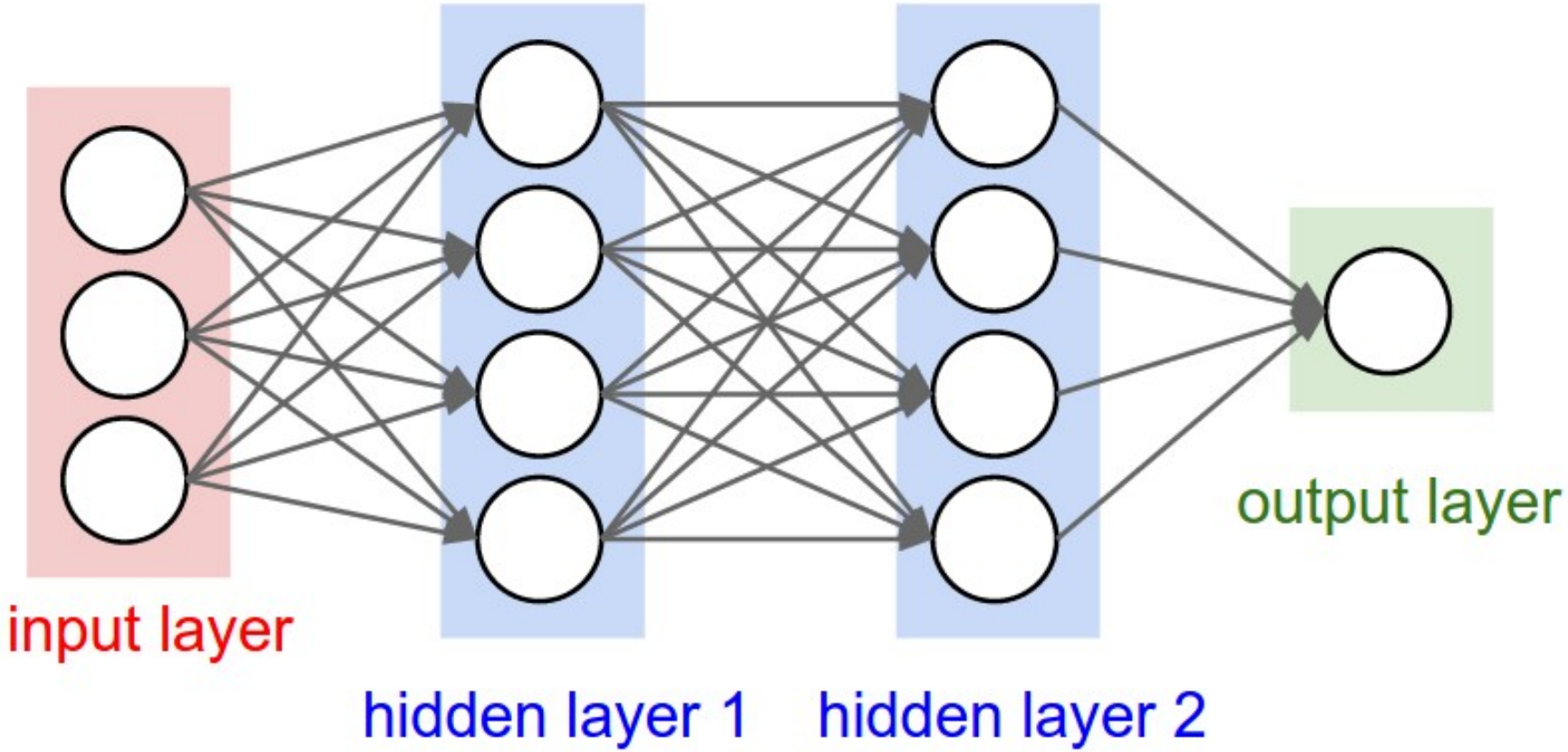


Figure : Arquitectura de una red neuronal profunda con dos capas ocultas

Tabla de Contenidos

Redes Neuronales Convolucionales

Características

- intenta ingresar un conocimiento a priori
- explota la localidad de los datos
- en un elemento solamente hace falta mirar su vecindario

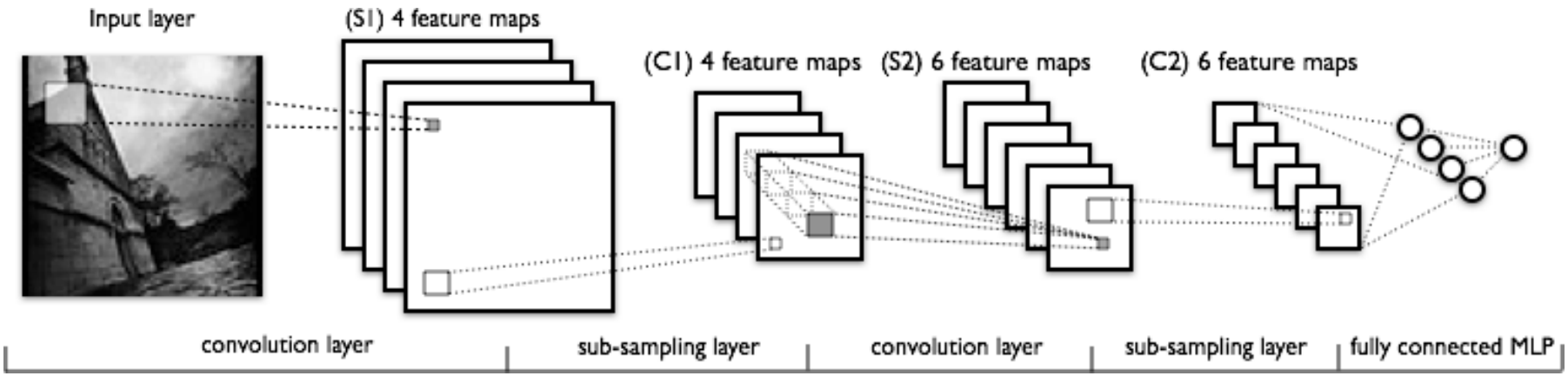


Figure : Arquitectura de una red neuronal convolucional (CNN).

Redes Neuronales Convolucionales

Extractores de características

- conjunto de extractores de características
- especialización en un tipo de característica
- localizados en cada región de la entrada

Apilamiento de capas

- capas convolucionales pueden ser apiladas
- capas más profundas encontrarán características más globales
- finalmente agregar un perceptrón multi-capa

Redes Neuronales Convolucionales-capa

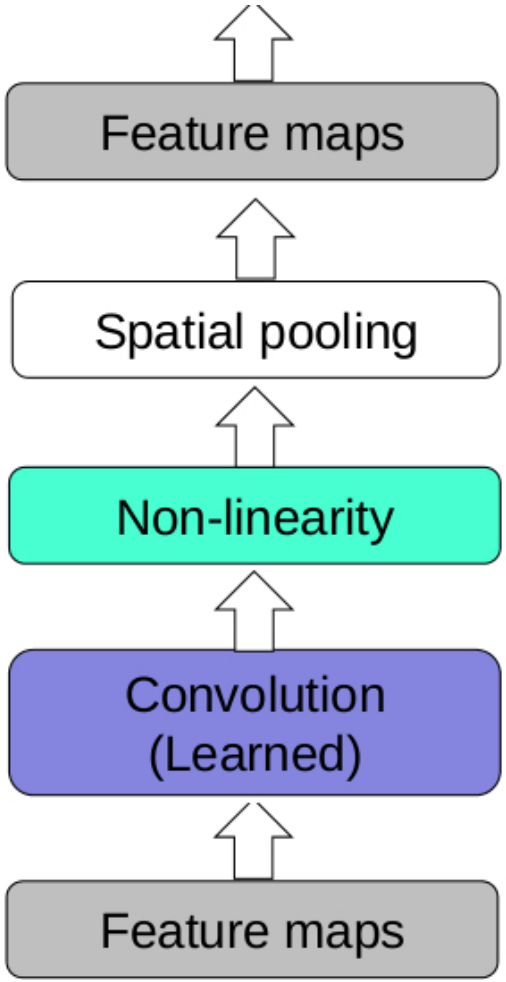


Figure : Procedimiento de cómputo en una capa convolucional standard.

Redes Neuronales Convolucionales-Convolución 1D

This operator merges an input signal $f(\cdot)$ with a pattern signal $g(\cdot)$ called kernel to obtain an output signal.
The output contains remarkable information in accordance with any criteria imposed by the kernel.
In 1D continuous domain the convolution is defined by:

$$(f * g)(t) = \int_{-\infty}^{\infty} f(t-\tau)\, g(\tau)\, d\tau \tag{1}$$

where time is the natural domain.
The same operator in a discrete space can expressed as:

$$(f * g)[n] = \sum_{k=-\infty}^{\infty} f[n-k]\, g[k] \tag{2}$$

This concept can be applied to 2D in the following way:

$$(f * g)(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x - \tau_1, y - \tau_2)\, g(\tau_1, \tau_2)\, d\tau_1\, d\tau_2 \qquad (3)$$

And its discrete expression:

$$(f * g)[x, y] = \sum_{k_1=-\infty}^{\infty} \sum_{k_2=-\infty}^{\infty} f[x - k_1, y - k_2]\, g[k_1, k_2] \qquad (4)$$

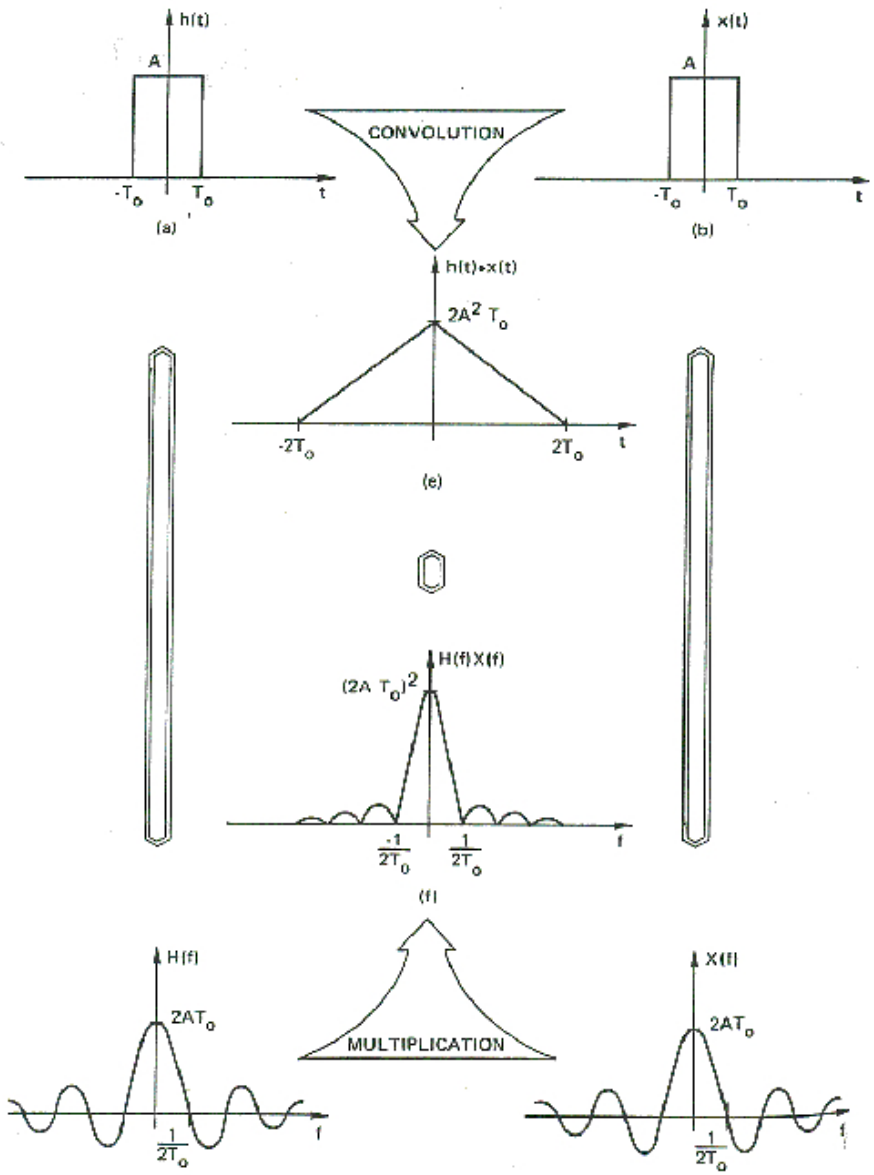
Redes Neuronales Convolucionales-Teorema de Convolución

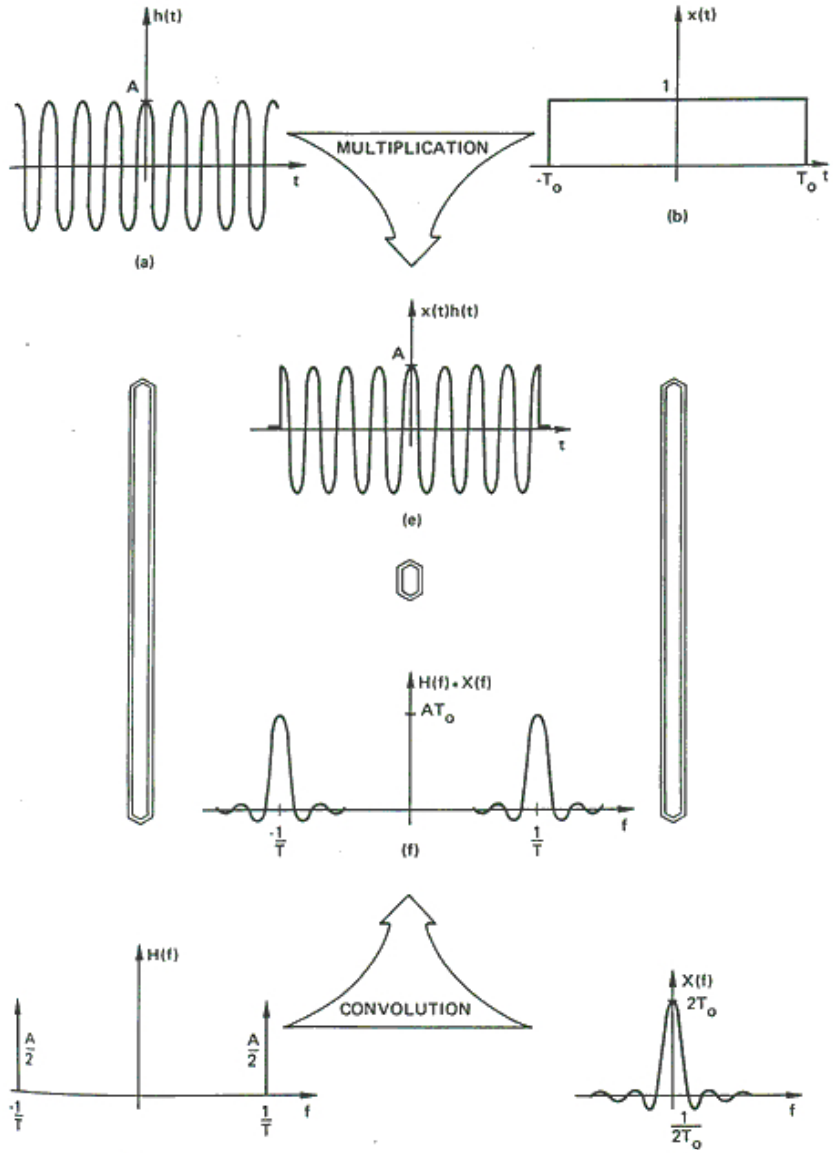
The convolution operation returns a filtered version of the input signal according to frequency component of the kernel.

$$\mathcal{F}\{f * g\} = F \cdot G \tag{5}$$

where $\mathcal{F}$ is the Fourier transform operator, F and G the Fourier transform of f and g respectively, and $F \cdot G$ denotes element-wise multiplication between F and G.

Redes Neuronales Convolucionales-Teorema de Convolución





Redes Neuronales Convolucionales-extractor

Un extractor de características viene determinado por un kernel de convolución

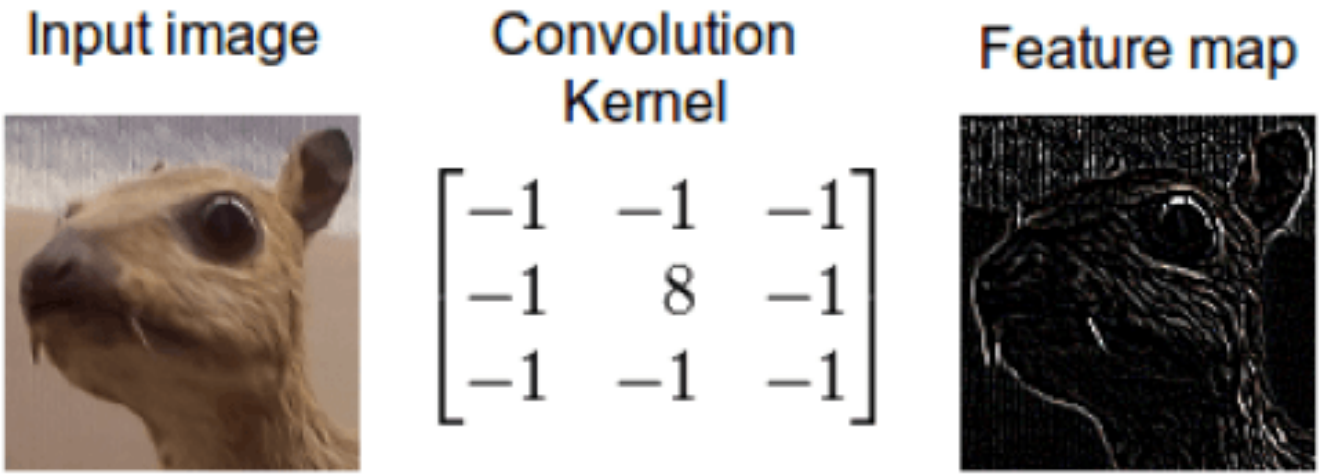


Figure : Aplicación de un kernel de convolución especializado en una característica a una imagen.

Las redes convolucionales intentan aprenden los mejores conjuntos de kernels.

Redes Neuronales Convolucionales-Nomenclatura

Un elemento de un canal de salida en la capa convolucional l viene determinado por:

$$o_n^l(x, y) \tag{6}$$

siendo n el n-ésimo canal del conjunto.
Un kernel en la capa l viene determinado por:

$$w_{n,m}^l(x, y) \tag{7}$$

siendo este el m-ésimo kernel que se va a aplicar al n-ésimo canal.
El tamaño del kernel viene dado por:

$$N_w \tag{8}$$

Redes Neuronales Convolucionales-Nomenclatura

Un elemento de bias está determinado por:

$$b_m^l(x, y) \tag{9}$$

función de activación dada por:

$$\sigma(x) \tag{10}$$

Error:

$$E(o^L) \tag{11}$$

Redes Neuronales Convolucionales-Propagación

Se aplica un conjunto de kernels $w^l_{,m}$ a cada elemento del conjunto de canales o^{l-1}, proveniente desde una capa anterior o desde la entrada de la red.

$$i^l_{n,m}(x,y) = \sum_{x_1,y_1} o^{l-1}_n(x + N_w - 1 - x_1, y + N_w - 1 - y_1)\, w^l_{n,m}(x_1,y_1) \quad (12)$$

Tomar en cuenta la adición del término $N_w - 1$ a cada coordenada de la entrada. Esto es necesario para permitir una concordancia entre las coordenadas (x,y) de o y (x,y) de i.

Redes Neuronales Convolucionales-Propagación

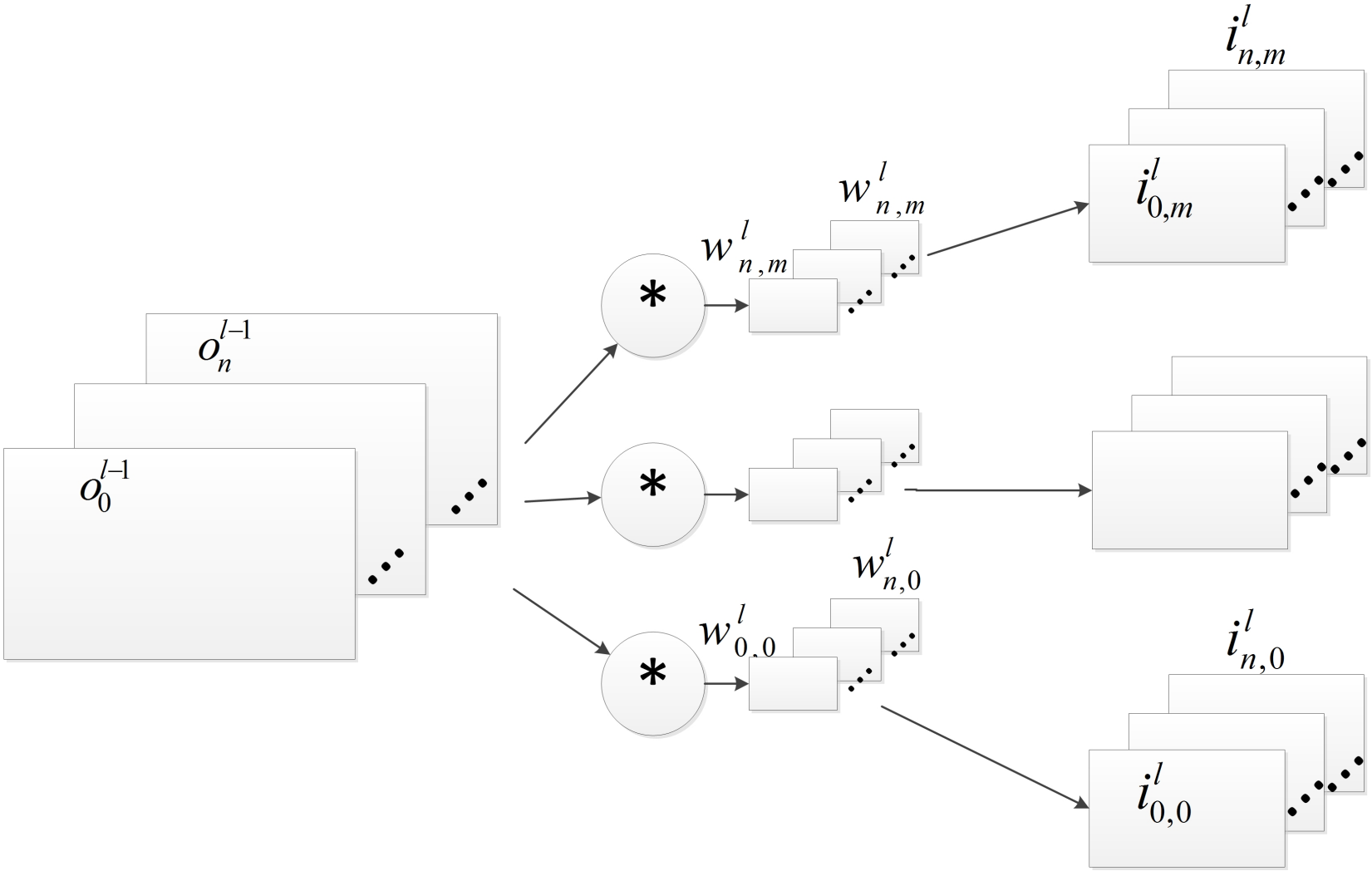


Figure : Proceso gráfico del cálculo de i.

Redes Neuronales Convolucionales-Propagación

Un término de bias b es sumado a cada resultante del agregado por canal de i^l. Este agregado genera los nuevos m canales.

$$c_m^l(x,y) = \sum_n i_{n,m}^l(x,y) + b_m^l \tag{13}$$

$$c_m^l(x,y) = \sum_{n,x_1,y_1} o_n^{l-1}(x + N_w - 1 - x_1, y + N_w - 1 - y_1)\, w_{n,m}^l(x_1,y_1) + b_m^l \tag{14}$$

Redes Neuronales Convolucionales-Propagación

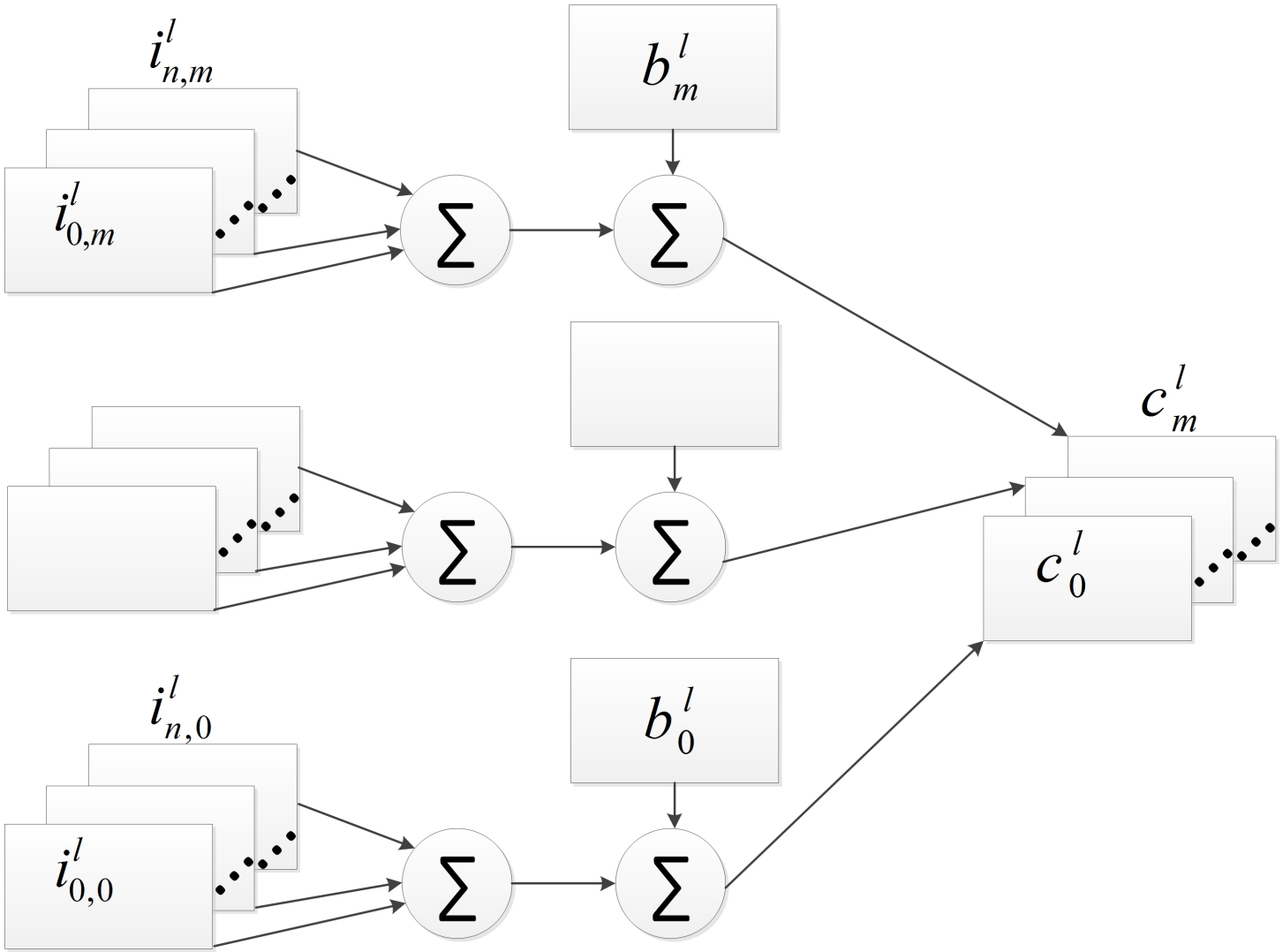


Figure : Proceso gráfico del cálculo de c.

Redes Neuronales Convolucionales-Propagación

Se aplica la función σ a cada elemento (x, y), de cada canal m en c^l. Como resultado la capa convolucional devuelve m canales provenientes de los m conjuntos de kernels w aplicados a la entrada.

$$o_m^l(x, y) = \sigma(c_m^l(x, y)) \tag{15}$$

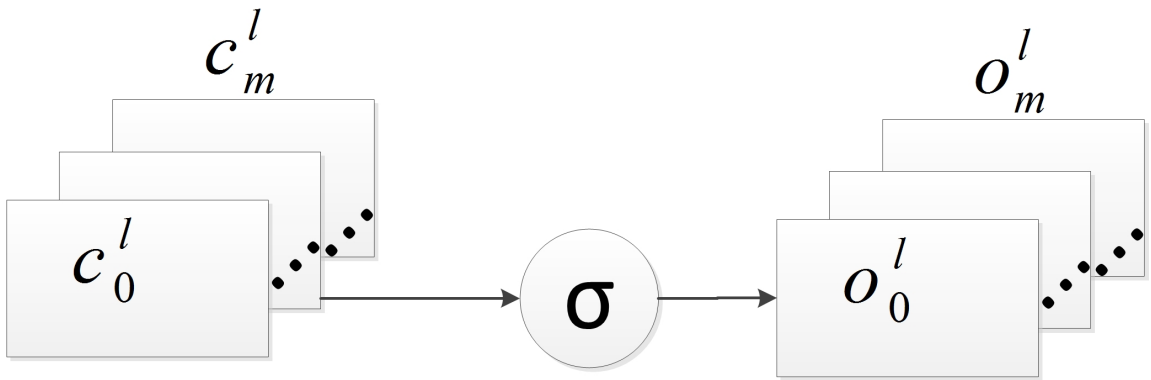


Figure : Proceso gráfico del cálculo de o.

Redes Neuronales Convolucionales - Retro-Propagación - Derivada del Error con respecto al kernel

Para optimizar una red neuronal, lo que se desea conocer es como influencia cada parámetro $w_{n,m}(x,y)$ en una capa l, al error E de la red completa. Esto es equivalente a la suma de la influencia de cada punto en c_m^l en E, ponderada por la influencia de $w_{n,m}(x,y)$ en cada punto concreto de c_m^l.

$$\frac{\partial E}{\partial w_{n,m}^l(x,y)} = \sum_{x_1,y_1} \frac{\partial E}{\partial c_m^l(x_1,y_1)} \frac{\partial c_m^l(x_1,y_1)}{\partial w_{n,m}^l(x,y)} \tag{16}$$

La influencia de cada punto en c_m^l en E, comúnmente se conoce como δ y es visto como el grado de cambio de E con un cambio en la salida de una capa.

$$\delta_m^l(x_1,y_1) = \frac{\partial E}{\partial c_m^l(x_1,y_1)} \tag{17}$$

Redes Neuronales Convolucionales - Retro-Propagación - Derivada del Error con respecto al kernel

La influencia de $w_{n,m}(x,y)$ en cada punto concreto de c_m^l es obtenida desarrollando la derivada de la ecuación 14.

$$\frac{\partial c_m^l(x_1,y_1)}{\partial w_{n,m}^l(x,y)} = \frac{\partial}{\partial w_{n,m}^l(x,y)}\Big(\sum_{n,x_2,y_2} o_n^{l-1}(x_1+N_w-1-x_2, y_1+N_w-1-y_2) \\ w_{n,m}^l(x_2,y_2) + b_m^l\Big) \quad (18)$$

$$\frac{\partial c_m^l(x_1,y_1)}{\partial w_{n,m}^l(x,y)} = o_n^{l-1}(x_1+N_w-1-x, y_1+N_w-1-y) \quad (19)$$

Reemplazando la ecuación 17 y 19 en 16:

$$\frac{\partial E}{\partial w_{n,m}^l(x,y)} = \sum_{x_1,y_1} o_n^{l-1}(x_1+N_w-1-x, y_1+N_w-1-y)\,\delta_m^l(x_1,y_1) \quad (20)$$

Redes Neuronales Convolucionales - Retro-Propagación - Derivada del Error con respecto al kernel

Por razones de implementación es necesario notar que la ecuación anterior puede ser reescrita como una operación de cross-correlación seguida por una rotación de 180° de la matriz resultante:

$$\frac{\partial E}{\partial w^l_{n,m}(x,y)} = rot_{180}\Big(\sum_{x_1,y_1} o^{l-1}_n(x+x_1, y+y_1)\,\delta^l_m(x_1,y_1)\Big) \tag{21}$$

Redes Neuronales Convolucionales - Retro-Propagación - Derivada del Error con respecto al bias

El mismo procedimiento se aplica para encontrar como afecta el parámetro b a E:

$$\frac{\partial E}{\partial b_m^l(x,y)} = \sum_{x_1,y_1} \delta_m^l(x_1,y_1)\, \frac{\partial c_m^l(x_1,y_1)}{\partial b_m^l(x,y)} \tag{22}$$

$$\frac{\partial c_m^l(x_1,y_1)}{\partial b_m^l(x,y)} = \frac{\partial}{\partial b_m^l(x,y)}\Big(\sum_{n,x_2,y_2} o_n^{l-1}(x_1+N_w-1-x_2, y_1+N_w-1-y_2)$$

$$w_{n,m}^l(x_2,y_2) + b_m^l\Big) \tag{23}$$

En este caso todos los valores dentro del sumatorio son independientes de b por lo tanto su derivada sera 0, entonces:

$$\frac{\partial c_m^l(x_1,y_1)}{\partial b_m^l(x,y)} = 1 \tag{24}$$

Redes Neuronales Convolucionales - Retro-Propagación - Derivada del Error con respecto al bias

Reemplazando la ecuación 17 y 24 en 23:

$$\frac{\partial E}{\partial b_m^l(x,y)} = \sum_{x_1,y_1} \delta_m^l(x_1,y_1) \tag{25}$$

Redes Neuronales Convolucionales - Cálculo de δ y Retro-propagación del Error

Para el cálculo de δ en una capa hacemos uso nuevamente de la regla de la cadena. Entonces se expresa que la variación del m-ésimo canal resultante c_m^l afecta a la variación de E, como la suma de las afecciones del m-ésimo canal de la capa l en el o-ésimo canal de la capa $l+1$, ponderado por las afecciones del o-ésimo canal de $l+1$ en la variación de E.

$$\delta_m^l(x,y) = \sum_o \sum_{x_1,y_1}^{N_w-1,N_w-1} \frac{\partial E}{\partial c_o^{l+1}(x+x_1, y+y_1)} \frac{\partial c_o^{l+1}(x+x_1, y+y_1)}{\partial c_m^l(x,y)} \tag{26}$$

Notese aquí que una coordenada (x,y) de c_m^l afecta en un rango de N_w coordenadas horizontales y verticales en su equivalente de la capa $l+1$, lo cual esta considerado en la ecuación dada.

Redes Neuronales Convolucionales - Cálculo de δ y Retro-propagación del Error

δ es tomado de la capa siguiente y es el término que retro-propaga el error hacia las capas anteriores. De esta manera el primer delta que se calcula es el de la capa final del red.

$$\delta_o^{l+1}(x + x_1, y + y_1) = \frac{\partial E}{\partial c_o^{l+1}(x + x_1, y + y_1)} \tag{27}$$

Redes Neuronales Convolucionales - Cálculo de δ y Retro-propagación del Error

Las afecciones del m-ésimo canal de la capa l en el o-ésimo canal de la capa $l+1$ vienen dadas por:

$$\frac{\partial c_o^{l+1}(x+x_1, y+y_1)}{\partial c_m^l(x,y)} = \frac{\partial}{\partial c_m^l(x,y)} \left(\sum_{m_1} \sum_{x_2,y_2}^{N_w-1,N_w-1} o_{m_1}^l(x+x_1-x_2, y+y_1-y_2) \right. \\ \left. w_{m_1,o}^{l+1}(x_2,y_2) + b_o^{l+1} \right) \quad (28)$$

$$\frac{\partial c_o^{l+1}(x+x_1, y+y_1)}{\partial c_m^l(x,y)} = \frac{\partial}{\partial c_m^l(x,y)} \left(\sum_{m_1} \sum_{x_2,y_2}^{N_w-1,N_w-1} \sigma(c_{m_1}^l(x+x_1-x_2, y+y_1-y_2)) \right. \\ \left. w_{m_1,o}^{l+1}(x_2,y_2) + b_o^{l+1} \right) \quad (29)$$

Redes Neuronales Convolucionales - Cálculo de δ y Retro-propagación del Error

$$\frac{\partial c_o^{l+1}(x+x_1, y+y_1)}{\partial c_m^l(x,y)} = \frac{\partial}{\partial c_m^l(x,y)} \left[\sigma(c_m^l(x,y))\, w_{m,o}^{l+1}(x_1,y_1) \right] \tag{30}$$

$$\frac{\partial c_o^{l+1}(x+x_1, y+y_1)}{\partial c_m^l(x,y)} = w_{m,o}^{l+1}(x_1,y_1) \sigma^{'}(c_m^l(x,y)) \tag{31}$$

Reemplazando la ecuación 27 y 31 en 26

$$\delta_m^l(x,y) = \left[\sum_o \sum_{x_1,y_1}^{N_w-1,N_w-1} \delta_o^{l+1}(x+x_1, y+y_1)\, w_{m,o}^{l+1}(x_1,y_1) \right] \sigma^{'}(c_m^l(x,y)) \tag{32}$$