Taller de Aprendizaje Automático

Redes Neuronales Convolucionales Profundas

Instituto de Ingeniería Eléctrica
Facultad de Ingeniería

UNIVERSIDAD
DE LA REPÚBLICA
URUGUAY

1. Introducción: redes *fully connected* y limitaciones, redes convolucionales
2. Conexiones con la percepción visual
3. Convolución de imágenes
4. Redes convolucionales
   - Capa de convolución
   - Capa de *pooling*
5. Arquitecturas de redes convolucionales profundas
6. Implementación de redes convolucionales en Keras y Tensorflow

El problema de clasificación de Imágenes

Problemas de visión son muy difíciles: se requieren invarianzas a distintas transformaciones (punto de vista, iluminación,..)

Dos grandes caminos:

1. **Aprender** las invarianzas a partir de un (enorme) conjunto de un entrenamiento (*let the data talk*)
2. **Construir** las invarianzas imponiendo un modelo en la representación

Francesco Peri, Hidden Cat - Penny

Christina Gandolfo, Cat in the box

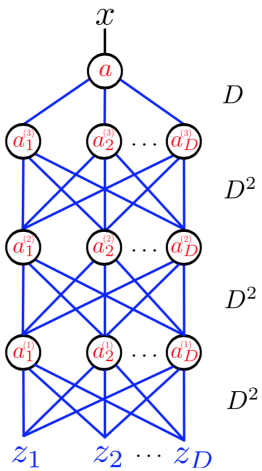
Matteo, hiding

Grahford, Hidden Cat

¿Cuál es el problema de las redes *fully connected*?

- ¿Cuántos parámetros tiene esta red?
  - $3D^2 + D$
- Si tenemos una imagen pequeña 32x32
  - $3 \times (32^2)^2 + 32^2 \approx 3 \times 10^6$

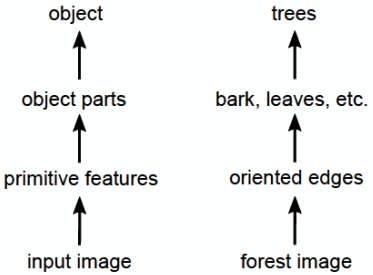
- **Difícil de entrenar**: sobreajuste, no escala bien
- **Redes de Convolución:** permiten disminuir número de parámetros



Slides adaptadas de Richard E. Turner

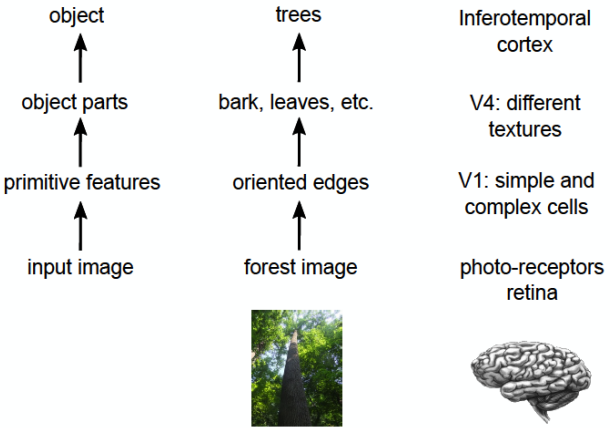
¿Por qué usar redes profundas?

Los datos (en general) tienen una organización jerárquica



¿Por qué usar redes profundas?

Nuestra visión tiene una organización jerárquica



Hubel y Wiesel - 1961*

Bases neurológicas de la percepción visual

106

J. Physiol. (1962), **160**, *pp.* 106–154
With 2 plates and 20 text-figures
Printed in Great Britain

RECEPTIVE FIELDS, BINOCULAR INTERACTION AND FUNCTIONAL ARCHITECTURE IN THE CAT'S VISUAL CORTEX

By D. H. HUBEL and T. N. WIESEL

From the Neurophysiology Laboratory, Department of Pharmacology Harvard Medical School, Boston, Massachusetts, U.S.A.

(*Received* 31 *July* 1961)

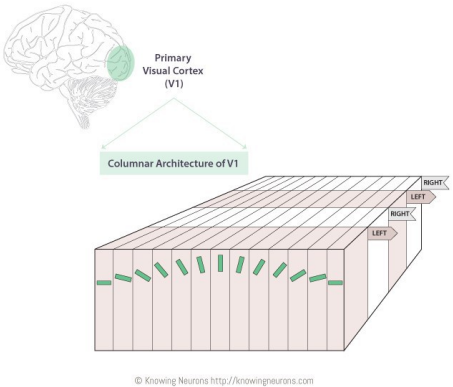
What chiefly distinguishes cerebral cortex from other parts of the central nervous system is the great diversity of its cell types and interconnexions. It would be astonishing if such a structure did not profoundly modify the response patterns of fibres coming into it. In the cat's visual cortex, the receptive field arrangements of single cells suggest that there is indeed a degree of complexity far exceeding anything yet seen at lower levels in the visual system.

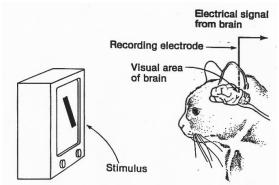
* D. Hubel and T. Wiesel, "Receptive fields, binocular interaction, and functional architecture in the cat's visual cortex," *Journal of Physiology*, vol. 160, 1962

Hubel y Wiesel - 1961*

Bases neurológicas de la percepción visual

El experimento del gato





* D. Hubel and T. Wiesel, "Receptive fields, binocular interaction, and functional architecture in the cat's visual cortex," *Journal of Physiology*, vol. 160, 1962

Hubel y Wiesel - 1961*

Bases neurológicas de la percepción visual

* D. Hubel and T. Wiesel, “Receptive fields, binocular interaction, and functional architecture in the cat’s visual cortex,” *Journal of Physiology*, vol. 160, 1962

Sobre como percibimos: un poco de cultura general

- Norbert Wiener. *Cybernetics: Or Control and Communication in the Animal and the Machine.* Paris, (Hermann & Cie) & Camb. Mass. (MIT Press); 1948, 2nd revised ed. 1961. Acceso abierto.

  En particular el capítulo 2 *(Gestalts and Univesals):* describe los procesos de percepción visual y luego presenta dispositivos para prótesis sensoriales similares a las utilizadas en el cerebro para la detección de Gestalt visual.

- Sugerencia recreativa: No es el fin del mundo - Pablo Casacuberta.

Convolución de imágenes

- **Convolución en imágenes:** Operación **lineal** entre imagen y filtro, que produce una nueva imagen

- Cada píxel se calcula como suma ponderada píxeles de imagen entrada trasladada y núcleo de convolución:

$$(u * h)(i,j) = \sum_{k,l} u(i-k, j-l) h(k,l)$$

- Es muy parecido a la correlación

$$(u * h)(i,j) = \sum_{k,l} u(i+k, j+l) h(k,l)$$

$$\underbrace{\begin{bmatrix} 130 & 136 & 53 & 44 & 231 & 67 & 108 \\ 130 & 89 & 77 & 58 & 250 & 154 & 130 \\ 208 & 239 & 120 & 111 & 112 & 181 & 22 \\ 203 & 223 & 59 & 79 & 28 & 57 & 67 \\ 164 & 140 & 215 & 235 & 66 & 30 & 204 \\ 97 & 159 & 50 & 110 & 104 & 76 & 7 \\ 207 & 150 & 58 & 47 & 152 & 81 & 237 \end{bmatrix}}_{u} * \underbrace{\frac{1}{5}\begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}}_{h} = \underbrace{\begin{bmatrix} 79.2 & 81.6 & 62.0 & 77.2 & 118.4 & 112.0 & 61.0 \\ 111.4 & 134.2 & 79.4 & 108.0 & 161.0 & 156.4 & 82.8 \\ 156.0 & 175.8 & 121.2 & 96.0 & 136.4 & 105.2 & 80.0 \\ 159.6 & 172.8 & 139.2 & 102.4 & 68.4 & 72.6 & 70.0 \\ 120.8 & 180.2 & 139.8 & 141.0 & 92.6 & 86.6 & 61.6 \\ 125.4 & 119.2 & 118.4 & 109.2 & 101.6 & 59.6 & 104.8 \\ 90.8 & 114.8 & 61.0 & 73.4 & 76.8 & 109.2 & 65.0 \end{bmatrix}}_{u*h}$$

Convolución de imágenes

Image Kernels, Explained Visually *por Victor Powell* (https://setosa.io/ev/image-kernels/)

Image Kernels

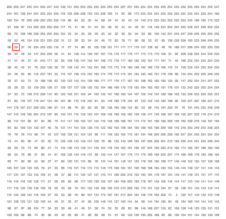
Explained Visually

Tweet | Me gusta 687 | Con

By Victor Powell

An image kernel is a small matrix used to apply effects like the ones you might find in Photoshop or Gimp, such as blurring, sharpening, outlining or embossing. They're also used in machine learning for 'feature extraction', a technique for determining the most important portions of an image. In this context the process is referred to more generally as "convolution" (see: convolutional neural networks.)

To see how they work, let's start by inspecting a black and white image. The matrix on the left contains numbers, between 0 and 255, which each correspond to the brightness of one pixel in a picture of a face. The large, granulated picture has been blown up to make it easier to see; the last image is the "real" size.



Redes Convolucionales: Motivación

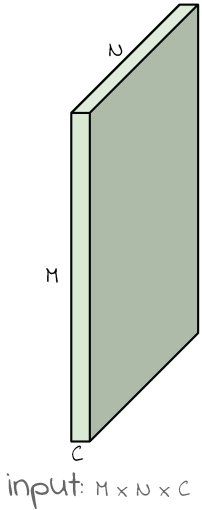
1. Características de bajo nivel son locales (detector de blobs, bordes)
   - Imponer localidad en el modelo: conectividad local (soporte del filtro)
   - Baja el número de parámetros: Núcleos pequeños

2. Estadísticas en imágenes son invariantes a traslaciones
   - Imponer invarianza a traslación en el modelo (en lugar de aprenderla)
   - Baja el número de parámetros: Se comparten pesos

3. Se espera que características de alto nivel sean gruesas (biología)
   - Se puede submuestrear a medida que aumenta la profundidad en la red
   - Baja (considerablemente) el costo computacional de correr la red

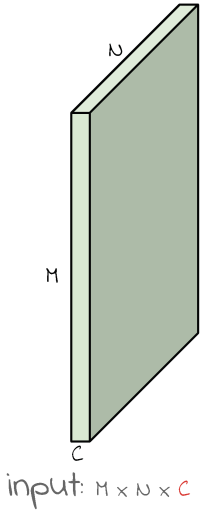
Capa de Convolución

Capa de Convolución



Convolución de imagen de entrada (tensor) y filtro (productos internos en cada posición de la imagen)

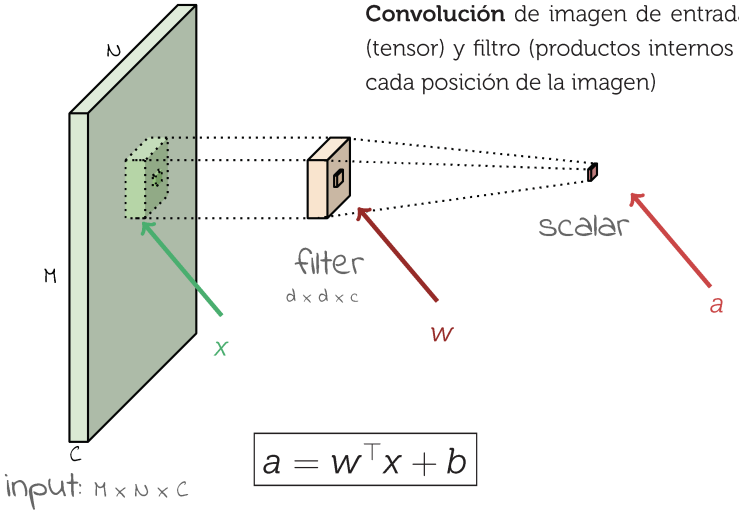
Capa de Convolución



Convolución de imagen de entrada (tensor) y filtro (productos internos en cada posición de la imagen)

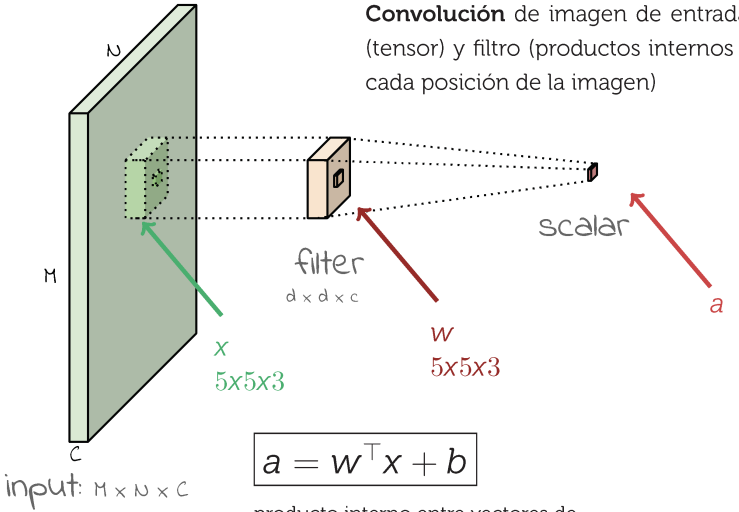
Capa de Convolución

Convolución de imagen de entrada (tensor) y filtro (productos internos en cada posición de la imagen)



$$a = w^\top x + b$$

Capa de Convolución

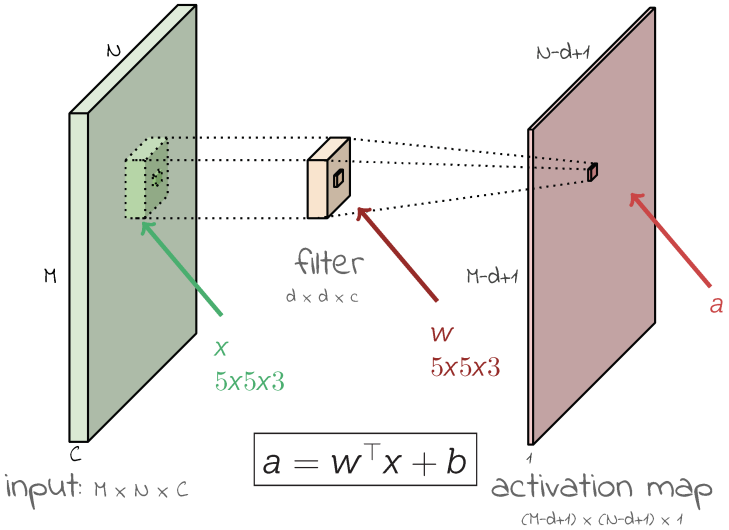


Convolución de imagen de entrada (tensor) y filtro (productos internos en cada posición de la imagen)

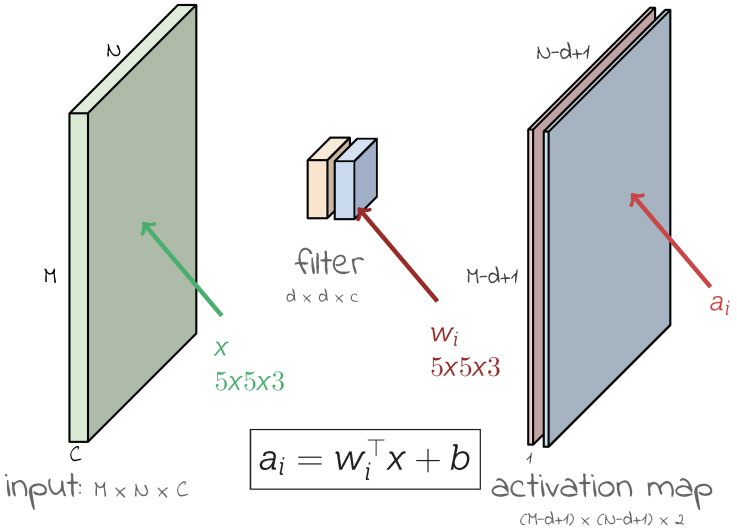
$$a = w^\top x + b$$

producto interno entre vectores de dimensión 75 $(5 \times 5 \times 3)$+ bias $\rightarrow$ **escalar**

Capa de Convolución

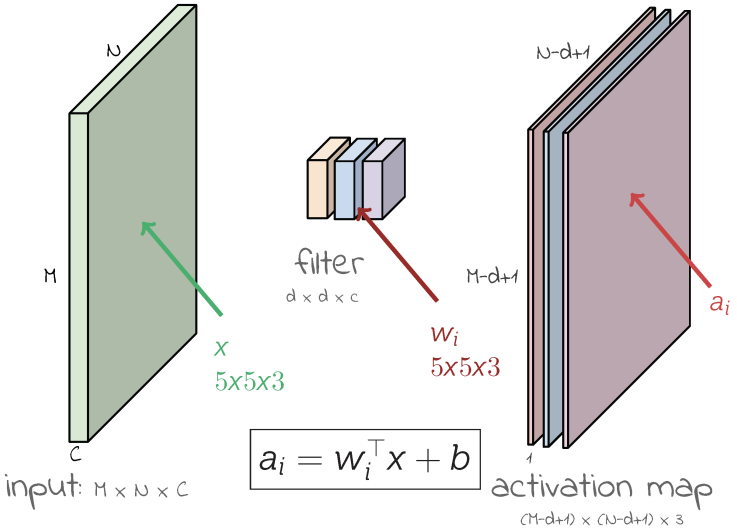


Capa de Convolución



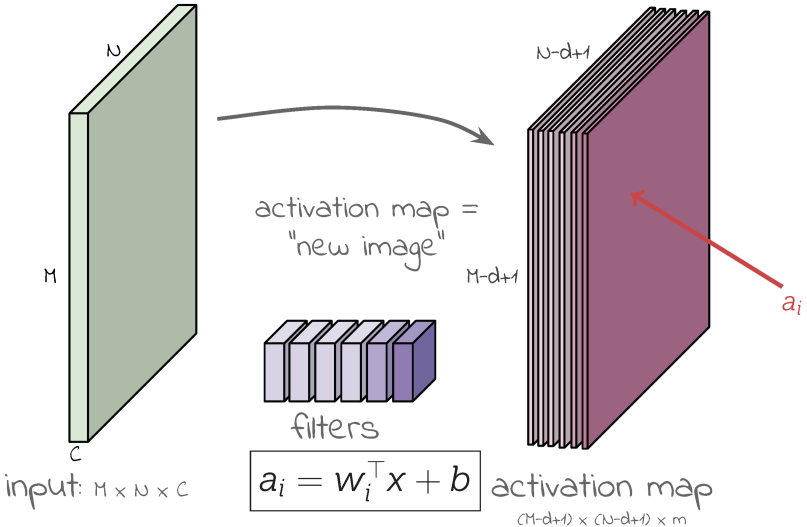
$$a_i = w_i^\top x + b$$

Capa de Convolución

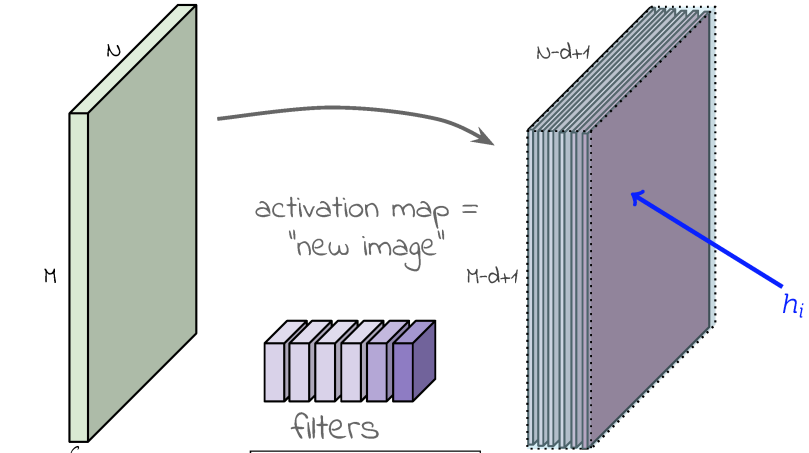


$$a_i = w_i^\top x + b$$

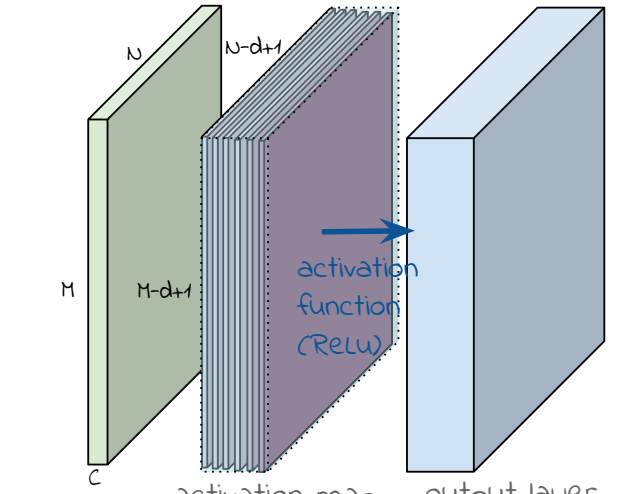
Capa de Convolución



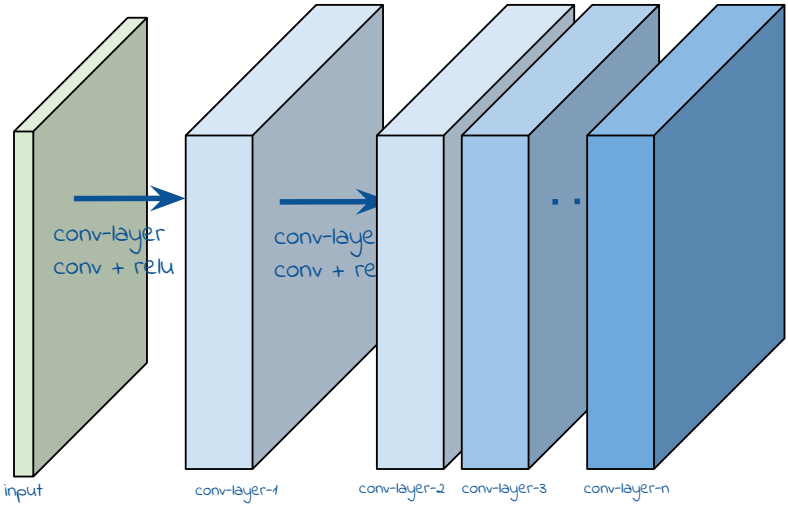
Capa de Convolución



Capa de Convolución: Convolución + Activación

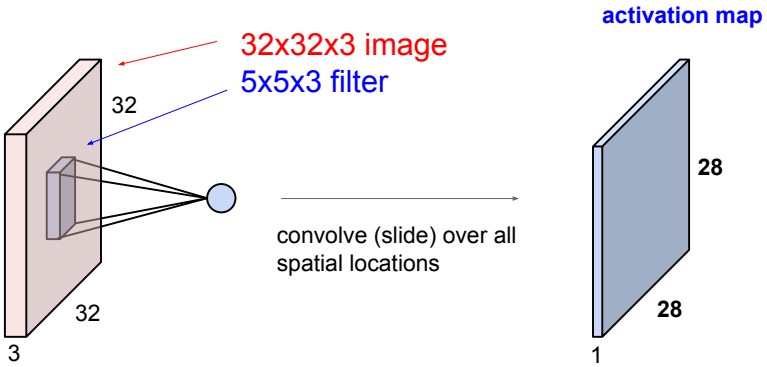


Capa de Convolución: Convolución + Activación



Capa de Convolución

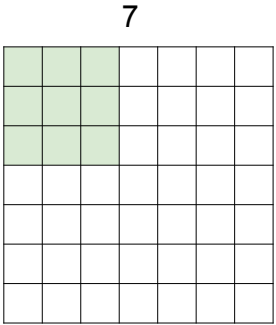
A closer look at spatial dimensions:



Slides tomadas de cs231n (Stanford) - Fei-Fei Li & Justin Johnson & Serena Yeung

Capa de Convolución

A closer look at spatial dimensions:

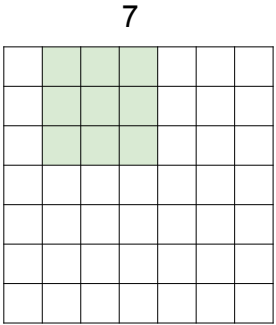


7x7 input (spatially)
assume 3x3 filter

Slides tomadas de cs231n (Stanford) - Fei-Fei Li & Justin Johnson & Serena Yeung

Capa de Convolución

A closer look at spatial dimensions:

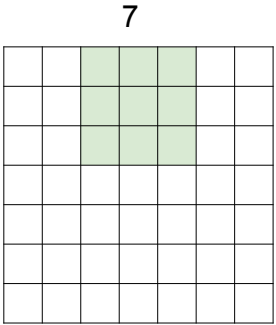


7x7 input (spatially)
assume 3x3 filter

Slides tomadas de cs231n (Stanford) - Fei-Fei Li & Justin Johnson & Serena Yeung

Capa de Convolución

A closer look at spatial dimensions:

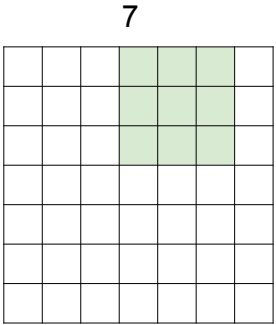


7x7 input (spatially)
assume 3x3 filter

Slides tomadas de cs231n (Stanford) - Fei-Fei Li & Justin Johnson & Serena Yeung

Capa de Convolución

A closer look at spatial dimensions:

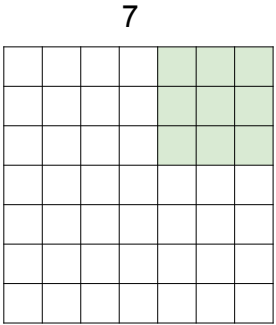


7x7 input (spatially)
assume 3x3 filter

Slides tomadas de cs231n (Stanford) - Fei-Fei Li & Justin Johnson & Serena Yeung

Capa de Convolución

A closer look at spatial dimensions:



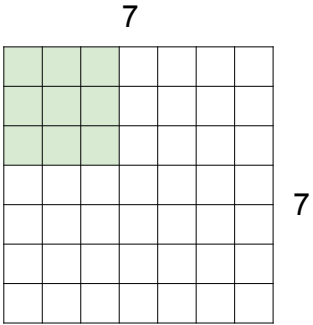
7x7 input (spatially)
assume 3x3 filter

=> 5x5 output

Slides tomadas de cs231n (Stanford) - Fei-Fei Li & Justin Johnson & Serena Yeung

Capa de Convolución

A closer look at spatial dimensions:

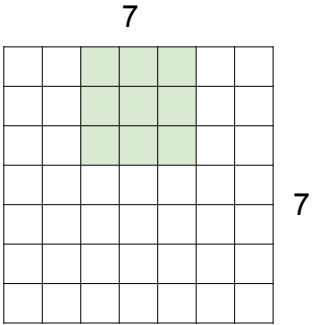


7x7 input (spatially)
assume 3x3 filter
applied **with stride 2**

Slides tomadas de cs231n (Stanford) - Fei-Fei Li & Justin Johnson & Serena Yeung

Capa de Convolución

A closer look at spatial dimensions:

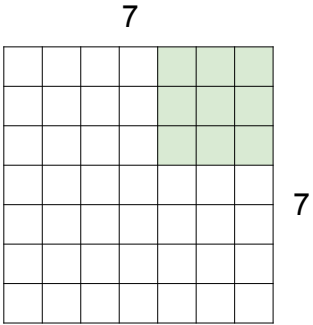


7x7 input (spatially)
assume 3x3 filter
applied **with stride 2**

Slides tomadas de cs231n (Stanford) - Fei-Fei Li & Justin Johnson & Serena Yeung

Capa de Convolución

A closer look at spatial dimensions:

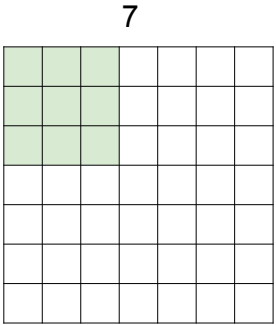


7x7 input (spatially)
assume 3x3 filter
applied **with stride 2**
=> 3x3 output!

Slides tomadas de cs231n (Stanford) - Fei-Fei Li & Justin Johnson & Serena Yeung

Capa de Convolución

A closer look at spatial dimensions:

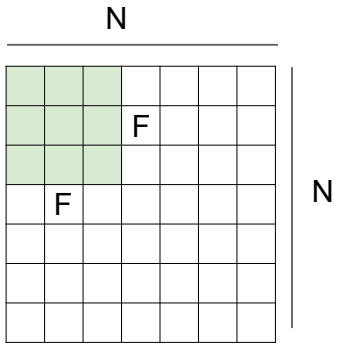


7x7 input (spatially)
assume 3x3 filter
applied **with stride 3?**

doesn't fit!
cannot apply 3x3 filter on
7x7 input with stride 3.

Slides tomadas de cs231n (Stanford) - Fei-Fei Li & Justin Johnson & Serena Yeung

Capa de Convolución



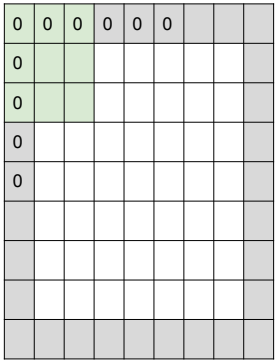
Output size:
(N - F) / stride + 1

e.g. N = 7, F = 3:
stride 1 => (7 - 3)/1 + 1 = 5
stride 2 => (7 - 3)/2 + 1 = 3
stride 3 => (7 - 3)/3 + 1 = 2.33 :\

Slides tomadas de cs231n (Stanford) - Fei-Fei Li & Justin Johnson & Serena Yeung

Capa de Convolución

In practice: Common to zero pad the border



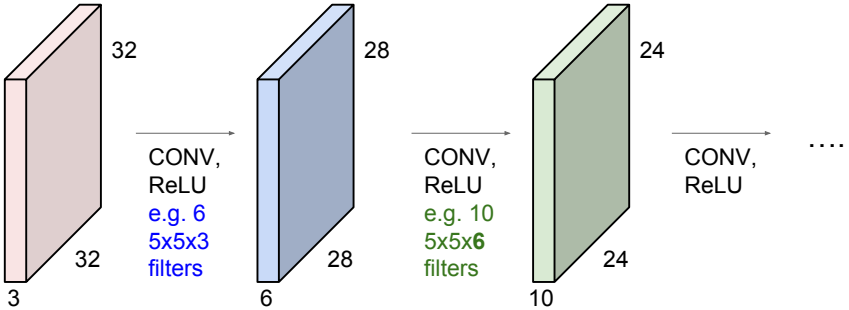
e.g. input 7x7
3x3 filter, applied with **stride 1**
pad with 1 pixel border => what is the output?

7x7 output!
in general, common to see CONV layers with stride 1, filters of size FxF, and zero-padding with (F-1)/2. (will preserve size spatially)
e.g. F = 3 => zero pad with 1
F = 5 => zero pad with 2
F = 7 => zero pad with 3

Slides tomadas de cs231n (Stanford) - Fei-Fei Li & Justin Johnson & Serena Yeung

Capa de Convolución

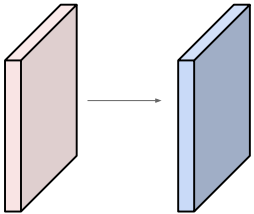
Remember back to…
E.g. 32x32 input convolved repeatedly with 5x5 filters shrinks volumes spatially!
(32 -> 28 -> 24 ...). Shrinking too fast is not good, doesn't work well.



Slides tomadas de cs231n (Stanford) - Fei-Fei Li & Justin Johnson & Serena Yeung

Capa de Convolución

Examples time:



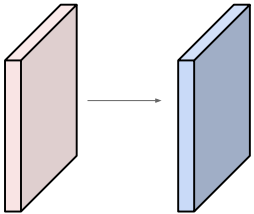
Input volume: **32x32x3**
10 5x5 filters with stride 1, pad 2

Output volume size:
(32+2*2-5)/1+1 = 32 spatially, so
32x32x10

Slides tomadas de cs231n (Stanford) - Fei-Fei Li & Justin Johnson & Serena Yeung

Capa de Convolución

Examples time:

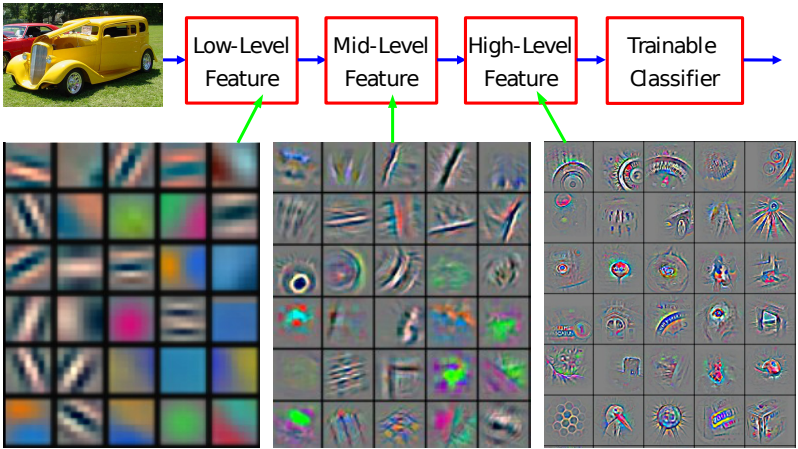


Input volume: **32x32x3**
10 5x5 filters with stride 1, pad 2

Number of parameters in this layer?
each filter has 5*5*3 + 1 = 76 params (+1 for bias)
=> 76*10 = **760**

Slides tomadas de cs231n (Stanford) - Fei-Fei Li & Justin Johnson & Serena Yeung

Representaciones jerárquicas adaptivas



Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

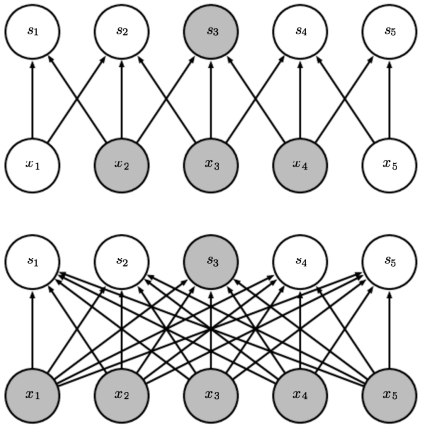
Ejemplo tomado de charlas de Y. Lecun.

Capa de Convolución: Observaciones

- **Conexiones Esparsas:** Conectividad de un elemento en la salida está dada por el soporte del filtro (en general pequeño: 3×3 o 5×5).
- **Pesos compartidos:** A diferencia de capas *totalmente conectadas*, los pesos de las capas de convolución (filtros) se reutilizan en varios elementos de la entrada.
- **Equivarianza:** Si se traslada la entrada, se traslada la salida.
- **Representaciones:** Capas de convolución permiten manejar datos de diferente tamaño (e.g., imágenes), sin necesidad de cambiar la arquitectura

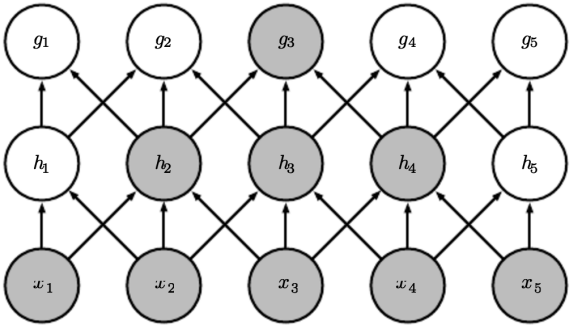
Capa de convolución: Observaciones

Campo Receptivo (*receptive field*): Varias capas de convolución con filtros pequeños $\rightarrow$ Aumenta el campo receptivo).



Capa de convolución: Observaciones

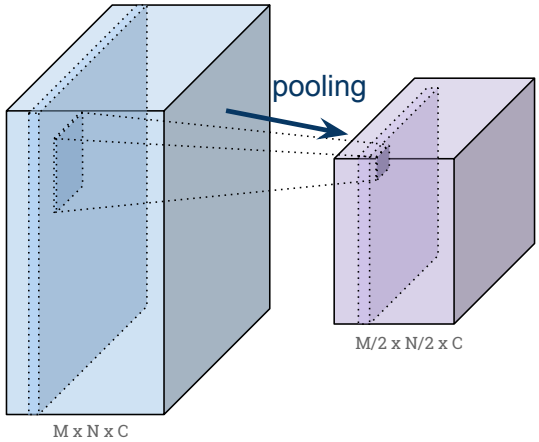
Campo Receptivo (*receptive field*): Varias capas de convolución con filtros pequeños $\rightarrow$ Aumenta el campo receptivo).



campo receptivo

Capa de *Pooling*

- Comprime (sub-muestreo) de la representación
- Opera en cada mapa de activación (canal) por separado



Capa de *Pooling*

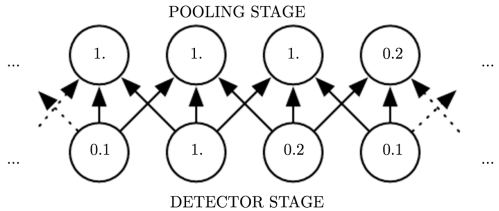
- Comprime (sub-muestreo) de la representación
- Opera en cada mapa de activación (canal) por separado

Capa de *Pooling*

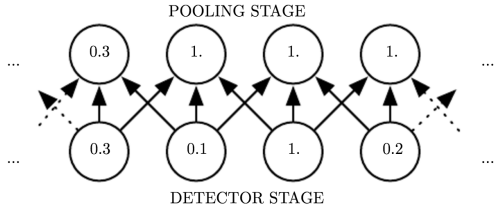
- Comprime (sub-muestreo) de la representación
- Opera en cada mapa de activación (canal) por separado

Pooling: Invarianza

El pooling introduce **Invarianza**.



Abajo: Salida de capa de conv.,
Arriba: salida de capa max-pooling
(*stride=1*, *width=3*)



Misma vista, cuando la entrada es trasladada 1 pixel. Capa de conv. cambia toda, mientras que capa del pooling sólo cambia la mitad

Convolución + Pooling como Prior

- **Convolución** puede verse como caso particular de una capa totalmente conectada
- Ciertos pesos **se comparten** (son los mismos) y otros están fijados a cero
- Esto puede verse como un ***prior*** (duro) en la distribución de pesos de una capa totalmente conectada
- ***Pooling*** puede verse como un *prior*, donde la salida debe ser invariante a pequeñas traslaciones
- **Atención:** Convolución + Pooling puede generar **subajuste**
  (si es necesario preservar información espacial de manera precisa)

Lenet-5 - 1998 (poniendo todo junto)Prueba *

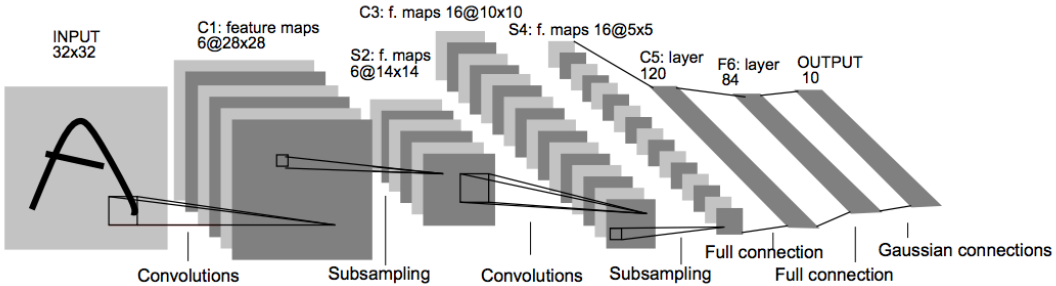


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

+info: https://yohanes.gultom.me/understanding-lenet-lecun-1998

* Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in *Proc. of the IEEE*, vol. 86(11), pp. 2278–2324, 1998

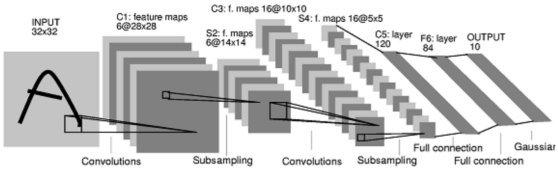
ConvnetJS

- (conv-relu-conv-relu-pool)x3-FC-softmax
- 7 capas 7000 parámetros, filtros 3×3, pooling 2×2
- ConvNetJS: http://cs.stanford.edu/people/karpathy/convnetjs/demo/cifar10.html

Evolución de las arquitecturas

1998

LeCun et al.



# of transistors

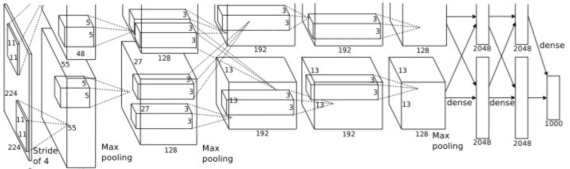
10^6

# of pixels used in training

10^7 NIST

2012

Krizhevsky et al.



# of transistors GPUs

10^9

# of pixels used in training

10^{14} IMAGENET

Evolución de las arquitecturas

2011

NEC-UIUC

Dense grid descriptor: HOG, LBP

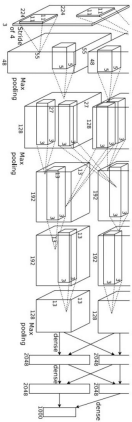
Coding: local coordinate, super-vector

Pooling, SPM

Linear SVM

2012

SuperVision



2014

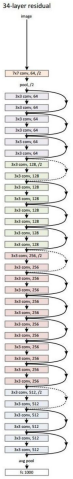
GoogLeNet

2014

VGG

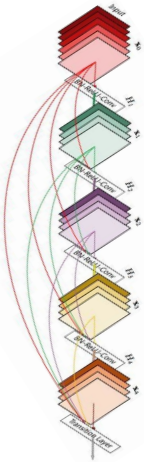
2015

Res-nets



2017

DenseNet

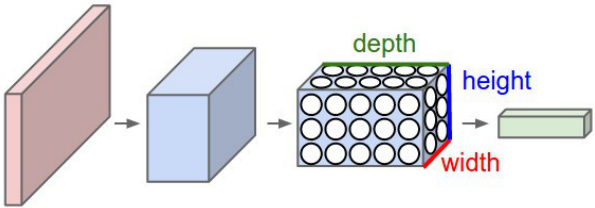


Capa de convolución: recordatorio

Tres hiperparámetros controlan el tamaño del volumen generado en una capa de convolución:

- *Depth*: Cantidad de filtros (hiperparámetro)
- *Stride*: paso de la convolución S (típicamente 1 o 2, rara vez mayor)
- *Zero-padding*: Se agranda el volumen para controlar el volumen de salida Volumen de salida es : $W_o = (W - F + 2P)/S + 1$ (similar para H_o)

Además debemos especificar el tamaño del filtro $F \times F$.



Arquitectura CNN típica en Keras

```
model = keras.models.Sequential([
    keras.layers.Conv2D(64, 7, activation="relu", padding="same",
                        input_shape=[28, 28, 1]),
    keras.layers.MaxPooling2D(2),
    keras.layers.Conv2D(128, 3, activation="relu", padding="same"),
    keras.layers.Conv2D(128, 3, activation="relu", padding="same"),
    keras.layers.MaxPooling2D(2),
    keras.layers.Conv2D(256, 3, activation="relu", padding="same"),
    keras.layers.Conv2D(256, 3, activation="relu", padding="same"),
    keras.layers.MaxPooling2D(2),
    keras.layers.Flatten(),
    keras.layers.Dense(128, activation="relu"),
    keras.layers.Dropout(0.5),
    keras.layers.Dense(64, activation="relu"),
    keras.layers.Dropout(0.5),
    keras.layers.Dense(10, activation="softmax")
])
```

- Se repiten algunas veces: stacks de conv-relu, luego pooling
- Luego un *flattening* para adaptar los datos a las capas densas
- Capas densas con relu, eventualmente intercaladas con capas de dropout para reducir el sobre-ajuste.

- Esta arquitectura simple alcanza una precisión de 92% (testing) para Fashion MNIST.
- Obs.: avanzando en profundidad, la cantidad de filtros por capa crece. Razonable ya que cuanto más de alto nivel son los features, más cantidad hay.
- Es usual multiplicar por N la cantidad de filtros luego de un pooling que reduce dimensiones espaciales por N (costo computacional controlado).

Referencias I


D. Hubel and T. Wiesel, “Receptive fields, binocular interaction, and functional architecture in the cat’s visual cortex,” *Journal of Physiology*, vol. 160, 1962.


Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” in *Proc. of the IEEE*, vol. 86(11), pp. 2278–2324, 1998.


A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition*.
O’Reilly Media, Inc., 2019.


A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems*, vol. 25, 2012.


M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in *Computer Vision – ECCV 2014*, pp. 818–833, Springer, 2014.