indra
minsait

Fundamentos del aprendizaje automático

(Machine learning)

Joaquín Luque

Contenido

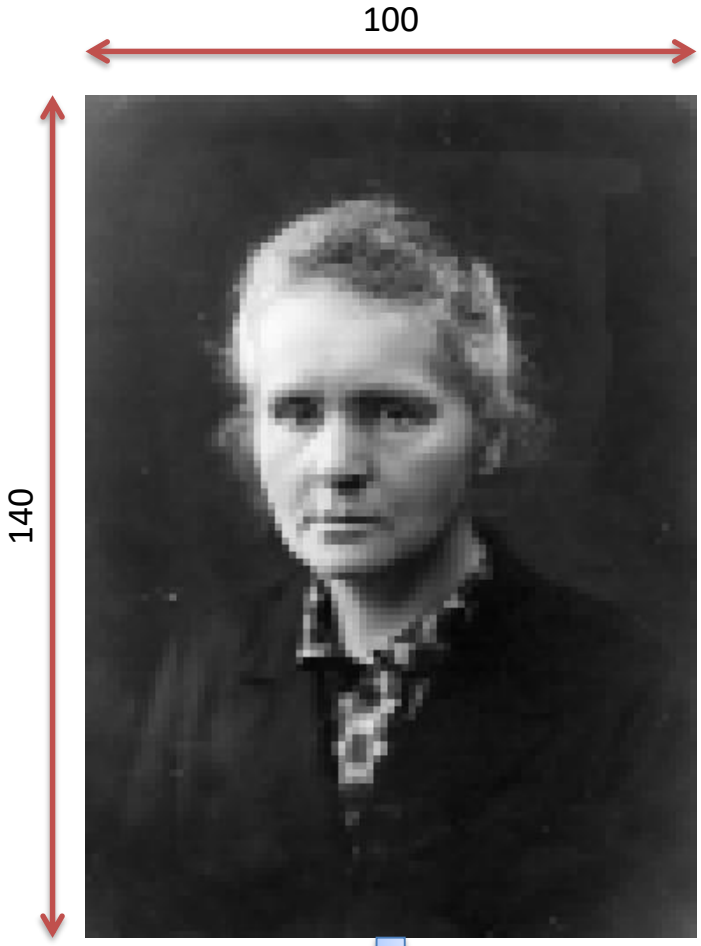
1. Introducción
2. Regresión
   a) Regresión univariable
   b) Regresión multivariable
3. Clasificación
   a) Regresión logística
   b) Máquinas de vectores soporte (SVM)
      - Forma dual de la optimización (regresión y SVM)
   c) Funciones Kernel
   d) Clasificación multiclase
4. Segmentación
5. Reducción de dimensionalidad
6. Deep learning (introducción)

Deep Learning

- Conceptos generales
- Redes convolucionales
  - Conceptos generales
    - Número de parámetros
    - Tensores
  - Filtros
  - Tipos de capas
    - Convolucional
    - Agrupación (pooling)
    - Completamente conectada
  - Ejemplos
    - LeNet-5
    - AlexNet
  - Transfer learning
- Redes recurrentes

Convolutional Neural Networks

Número de parámetros



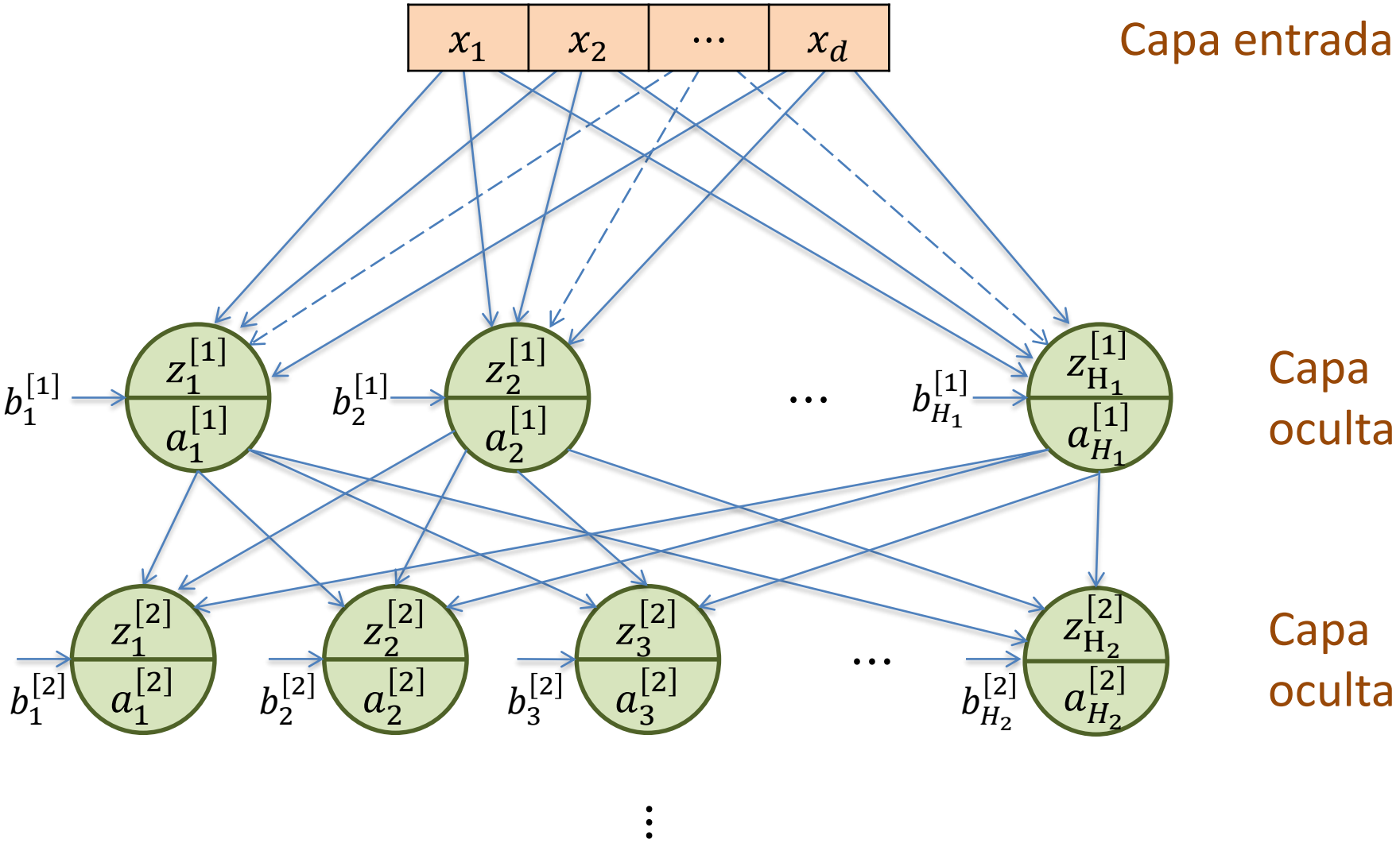
$$d = 140 \times 100 = 14{,}000$$

Vector de características

$x_{1,1}^{(i)}$	$x_{1,2}^{(i)}$	$\cdots$	$x_{1,100}^{(i)}$	$x_{2,1}^{(i)}$	$\cdots$	$x_{2,100}^{(i)}$	$\cdots$	$x_{140,1}^{(i)}$	$\cdots$	$x_{140,100}^{(i)}$

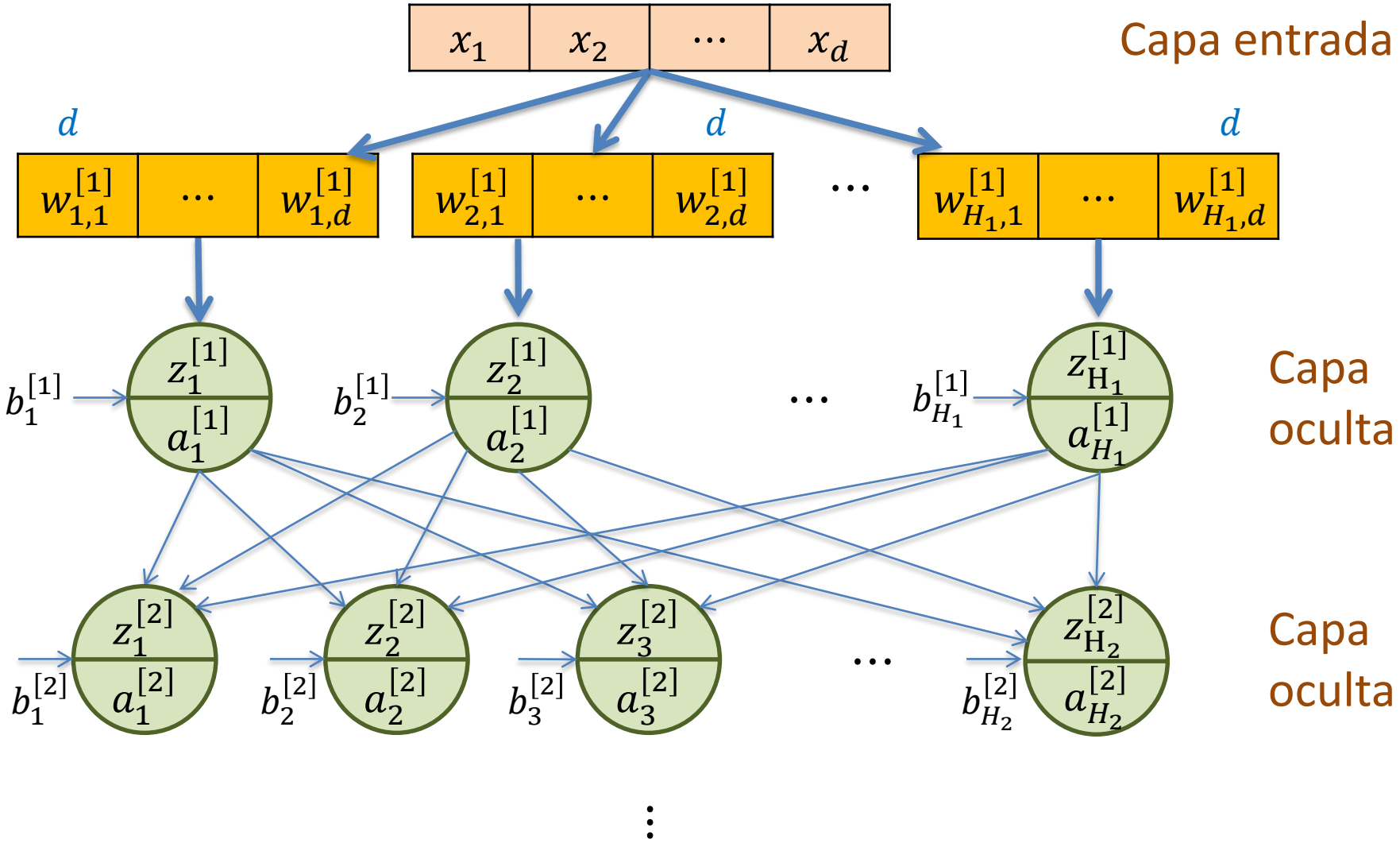
Convolutional Neural Networks

Número de parámetros



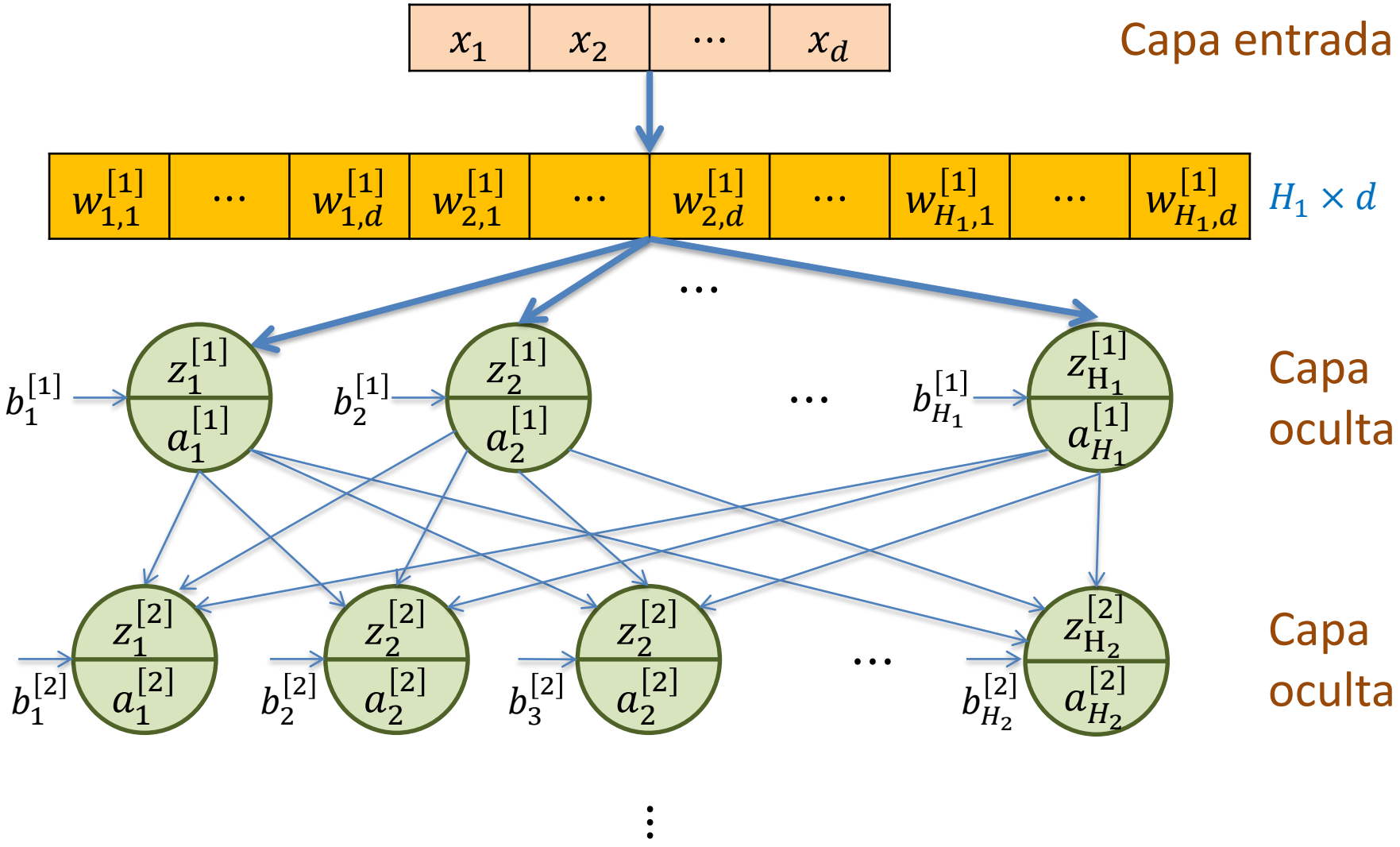
Convolutional Neural Networks

Número de parámetros



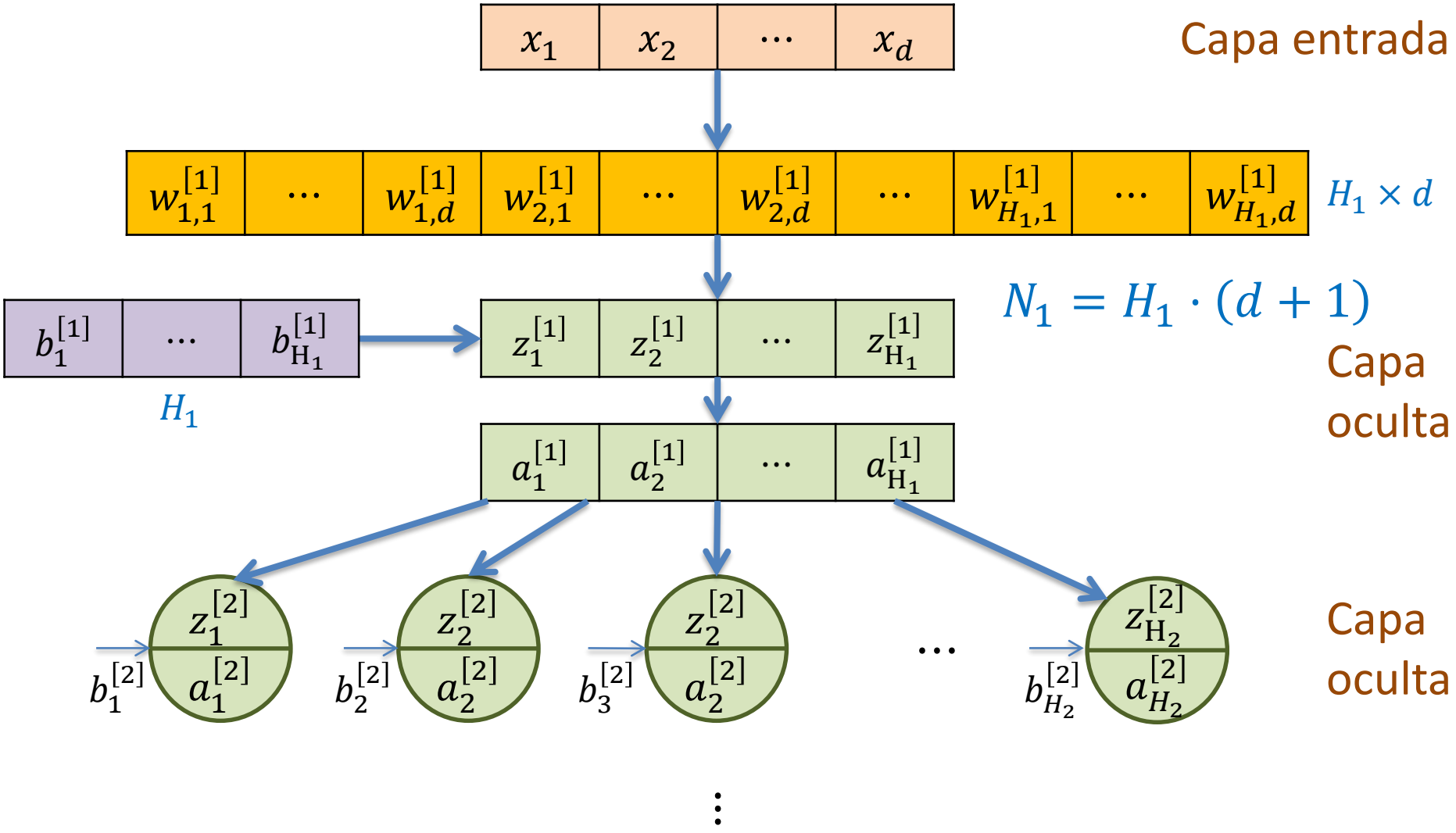
Convolutional Neural Networks

Número de parámetros



Convolutional Neural Networks

Número de parámetros



Convolutional Neural Networks

Número de parámetros

Número de parámetros de la red neuronal:

$$N = \sum_{l=1}^{L} H_l \cdot (H_{l-1} + 1)$$

Si $d \gg 1; H_L = 1; \forall l, H_l \approx d$:

$$N = H_L \cdot (H_{L-1} + 1) + \sum_{l=1}^{L-1} H_l \cdot (H_{l-1} + 1)$$

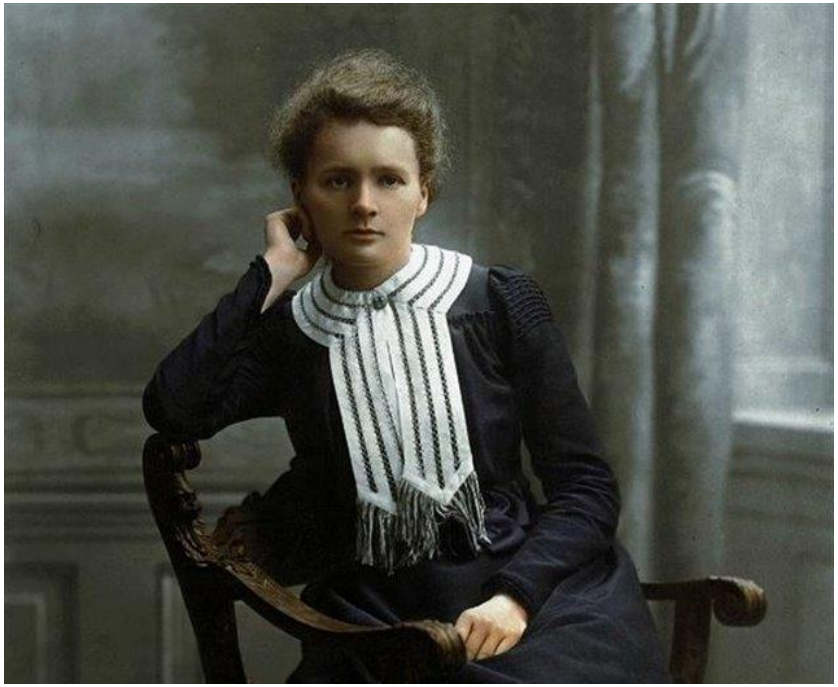
$$N \approx 1 \cdot (d + 1) + \sum_{l=1}^{L-1} d \cdot (d + 1) \approx d + \sum_{l=1}^{L-1} d^2$$

$$N \approx d^{2(L-1)} + d$$

Si $d \gg 1; H_L = 1; \forall l\ , H_l \approx d : N \approx d^{2L}$

Convolutional Neural Networks

Número de parámetros



$$d = 1000 \times 1000 \times 3 = 3 \cdot 10^6$$

Si $L = 2$

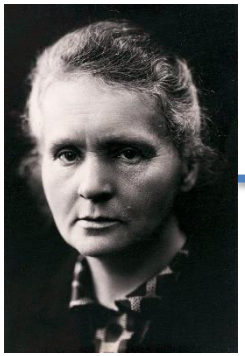
$$N \approx d^{2L} = (3 \cdot 10^6)^4 = 3^4 \cdot 10^{24} \approx 10^{26}$$

Convolutional Neural Networks
Número de parámetros

- El número de parámetros de una red neuronal convencional es inmanejable
- Intuitivamente
  - Los parámetros correspondientes a un pixel deben ser similares a la de los pixeles vecinos
  - La vecindad entre pixeles no se da en una línea (1D), sino en:
    - Un plano (2D) para las imágenes en B/N
    - Un cubo (3D) para las imágenes en colores
    - Un hipercubo (4D) para las imágenes en colores en movimiento

Convolutional Neural Networks

Tensores

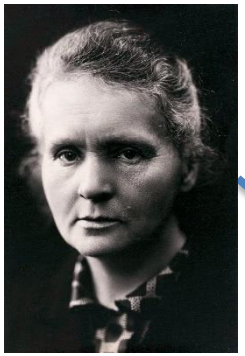


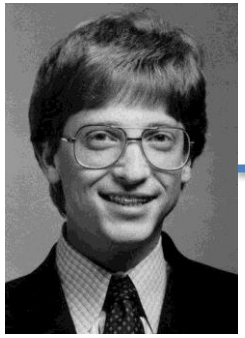
Vector de características

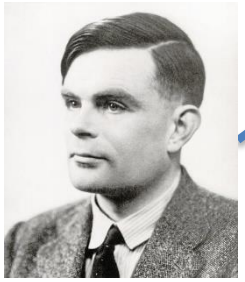
$x_{1,1}^{(1)}$	$x_{1,2}^{(1)}$	$\cdots$	$x_{1,100}^{(1)}$	$x_{2,1}^{(1)}$	$\cdots$	$x_{2,100}^{(1)}$	$\cdots$	$x_{140,1}^{(1)}$	$\cdots$	$x_{140,100}^{(1)}$

Convolutional Neural Networks

Tensores







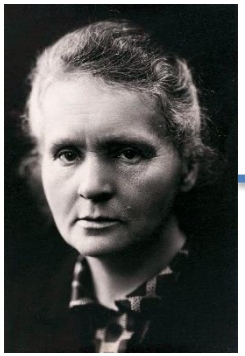
$\vdots$

Matriz de diseño

$x_{1,1}^{(1)}$	$x_{1,2}^{(1)}$	$\cdots$	$x_{1,100}^{(1)}$	$x_{2,1}^{(1)}$	$\cdots$	$x_{2,100}^{(1)}$	$\cdots$	$x_{140,1}^{(1)}$	$\cdots$	$x_{140,100}^{(1)}$
$x_{1,1}^{(2)}$	$x_{1,2}^{(2)}$	$\cdots$	$x_{1,100}^{(2)}$	$x_{2,1}^{(2)}$	$\cdots$	$x_{2,100}^{(2)}$	$\cdots$	$x_{140,1}^{(2)}$	$\cdots$	$x_{140,100}^{(2)}$
$x_{1,1}^{(3)}$	$x_{1,2}^{(3)}$	$\cdots$	$x_{1,100}^{(3)}$	$x_{2,1}^{(3)}$	$\cdots$	$x_{2,100}^{(3)}$	$\cdots$	$x_{140,1}^{(3)}$	$\cdots$	$x_{140,100}^{(3)}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

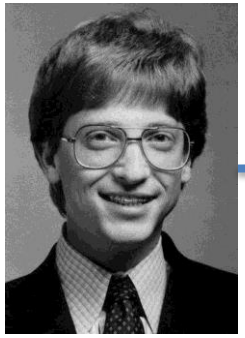
Convolutional Neural Networks

Tensores



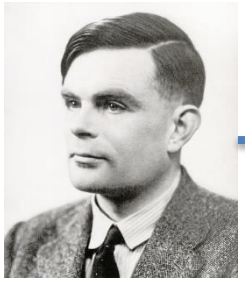
$x_{1,1}^{(1)}$	$x_{1,2}^{(1)}$	$\cdots$	$x_{1,100}^{(1)}$
$x_{2,1}^{(1)}$	$x_{2,2}^{(1)}$	$\cdots$	$x_{2,100}^{(1)}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$x_{140,1}^{(1)}$	$x_{140,2}^{(1)}$	$\cdots$	$x_{140,100}^{(1)}$

Las características se organizan en 2D para aprovechar similitudes entre píxeles vecinos



$x_{1,1}^{(2)}$	$x_{1,2}^{(2)}$	$\cdots$	$x_{1,100}^{(2)}$
$x_{2,1}^{(2)}$	$x_{2,2}^{(2)}$	$\cdots$	$x_{2,100}^{(2)}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$x_{140,1}^{(2)}$	$x_{140,2}^{(2)}$	$\cdots$	$x_{140,100}^{(2)}$

Matriz de características



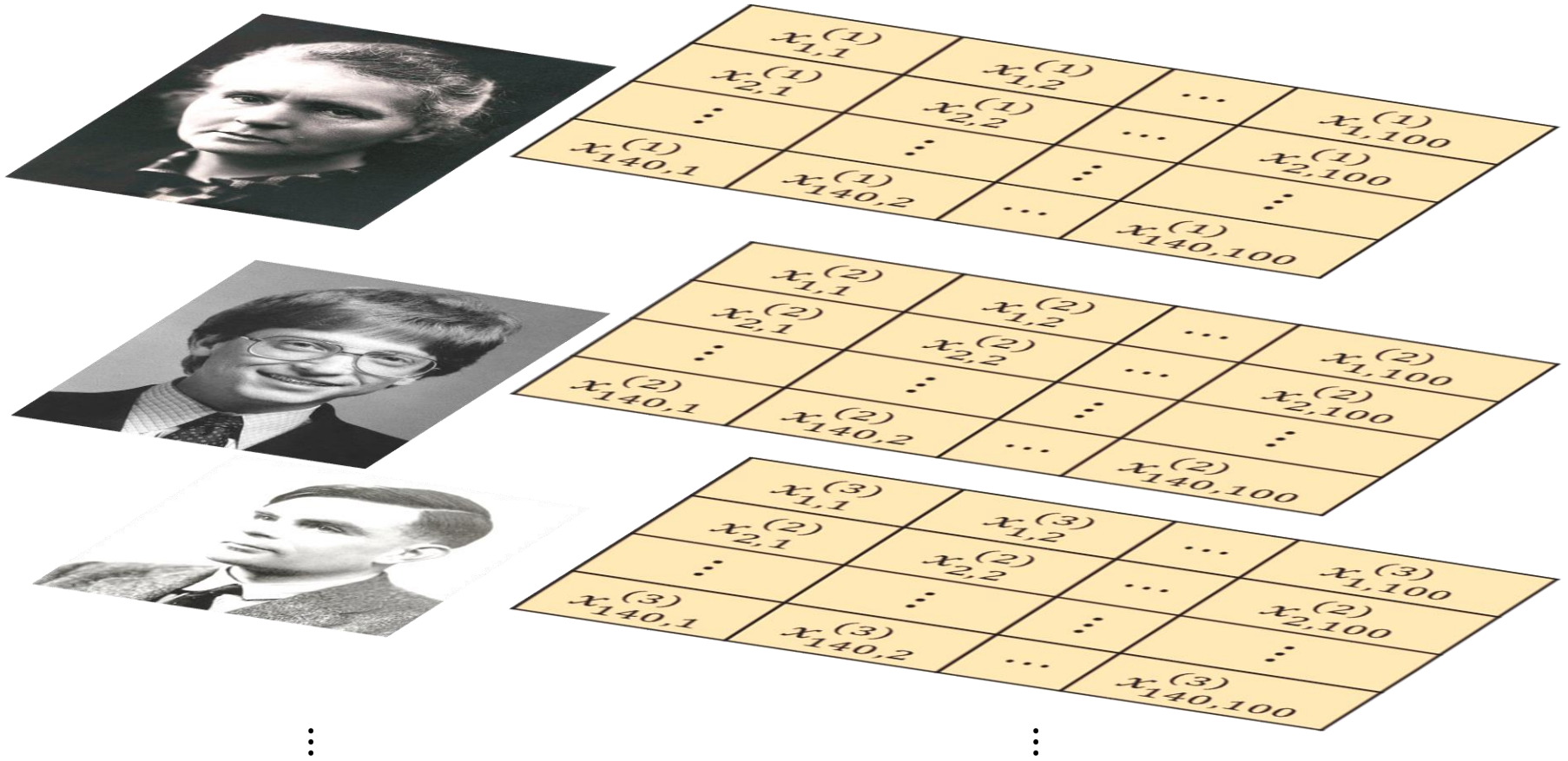
$x_{1,1}^{(3)}$	$x_{1,2}^{(3)}$	$\cdots$	$x_{1,100}^{(3)}$
$x_{2,1}^{(2)}$	$x_{2,2}^{(2)}$	$\cdots$	$x_{2,100}^{(2)}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$x_{140,1}^{(3)}$	$x_{140,2}^{(3)}$	$\cdots$	$x_{140,100}^{(3)}$

$\vdots$

Convolutional Neural Networks

Tensores

Tensor de diseño (3D)



Convolutional Neural Networks

Tensores

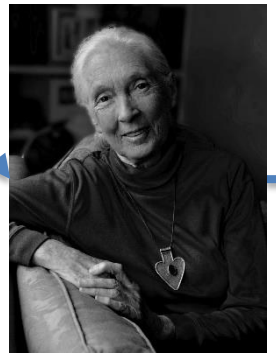
R

$$\begin{bmatrix} x_{1,1}^{(i_r)} & x_{1,2}^{(i_r)} & \cdots & x_{1,100}^{(i_r)} \\ x_{2,1}^{(i_r)} & x_{2,2}^{(i_r)} & \cdots & x_{2,100}^{(i_r)} \\ \vdots & \vdots & \vdots & \vdots \\ x_{140,1}^{(i_r)} & x_{140,2}^{(i_r)} & \cdots & x_{140,100}^{(i_r)} \end{bmatrix}$$

G

$$\begin{bmatrix} x_{1,1}^{(i_g)} & x_{1,2}^{(i_g)} & \cdots & x_{1,100}^{(i_g)} \\ x_{2,1}^{(i_g)} & x_{2,2}^{(i_g)} & \cdots & x_{2,100}^{(i_g)} \\ \vdots & \vdots & \vdots & \vdots \\ x_{140,1}^{(i_g)} & x_{140,2}^{(i_g)} & \cdots & x_{140,100}^{(i_g)} \end{bmatrix}$$

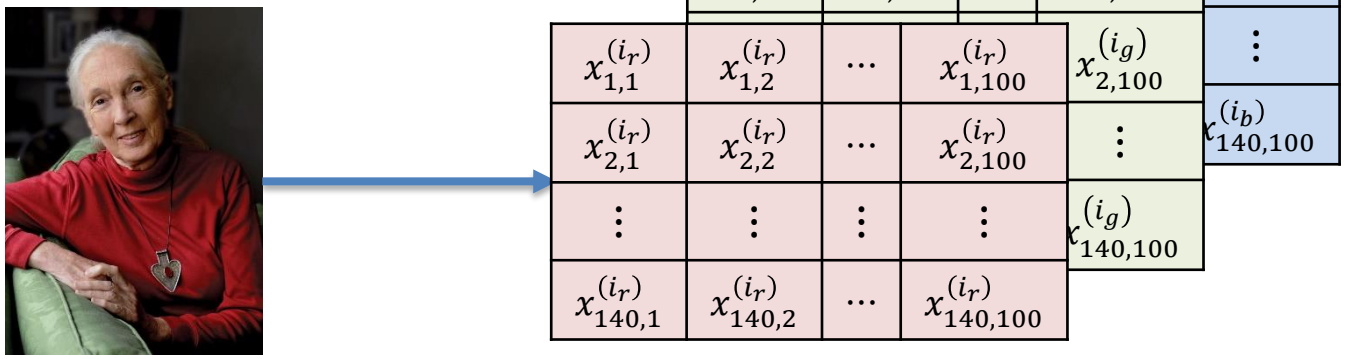
B



$$\begin{bmatrix} x_{1,1}^{(i_b)} & x_{1,2}^{(i_b)} & \cdots & x_{1,100}^{(i_b)} \\ x_{2,1}^{(i_b)} & x_{2,2}^{(i_b)} & \cdots & x_{2,100}^{(i_b)} \\ \vdots & \vdots & \vdots & \vdots \\ x_{140,1}^{(i_b)} & x_{140,2}^{(i_b)} & \cdots & x_{140,100}^{(i_b)} \end{bmatrix}$$

Convolutional Neural Networks
Tensores

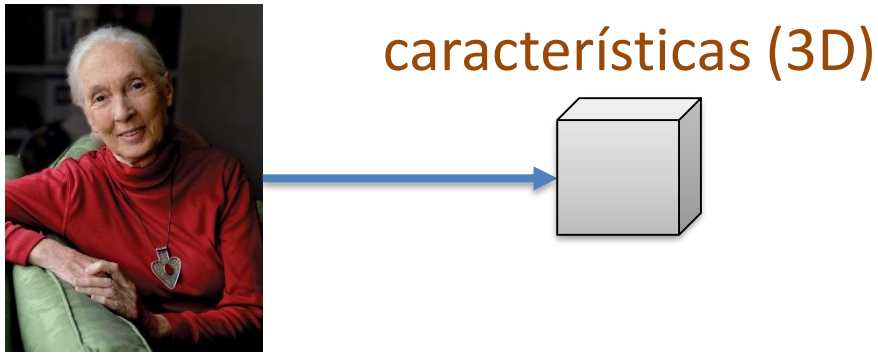
Tensor de características (3D)



Las características se organizan en 3D para aprovechar similitudes entre píxeles vecinos, incluso de colores distintos

Convolutional Neural Networks

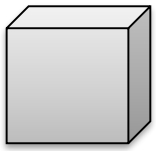
Tensores

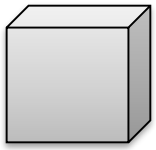


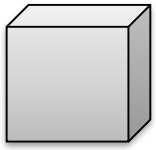
Convolutional Neural Networks

Tensores

Tensor de diseño (4D)







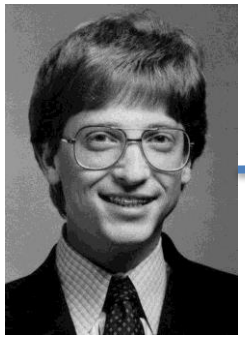
⋮ ⋮

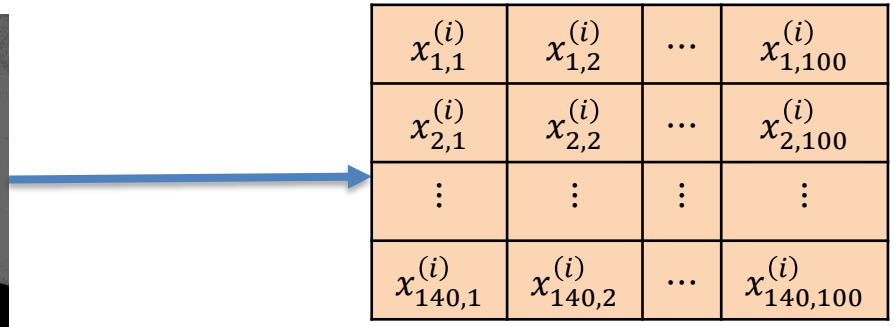
Deep Learning

- Conceptos generales
- Redes convolucionales
  - Conceptos generales
    - Número de parámetros
    - Tensores
  - Filtros
  - Tipos de capas
    - Convolucional
    - Agrupación (pooling)
    - Completamente conectada
  - Ejemplos
    - LeNet-5
    - AlexNet
  - Transfer learning
- Redes recurrentes

Convolutional Neural Networks

Filtros



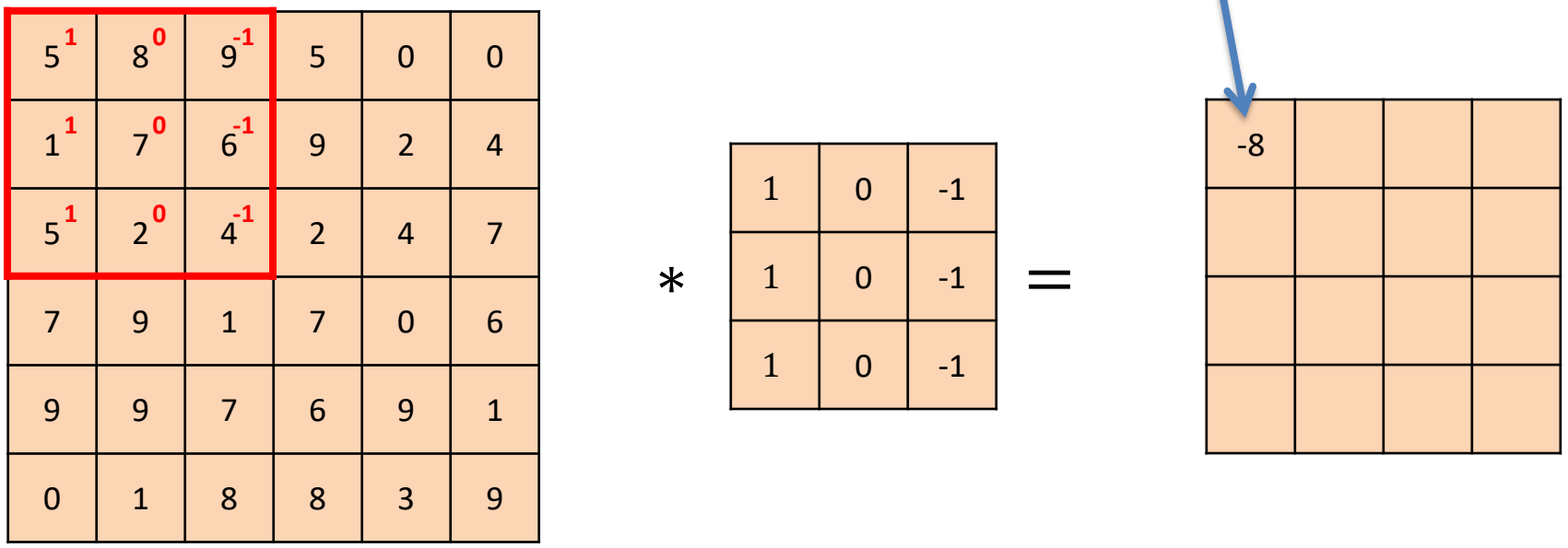


$x_{1,1}^{(i)}$	$x_{1,2}^{(i)}$	$\cdots$	$x_{1,100}^{(i)}$
$x_{2,1}^{(i)}$	$x_{2,2}^{(i)}$	$\cdots$	$x_{2,100}^{(i)}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$x_{140,1}^{(i)}$	$x_{140,2}^{(i)}$	$\cdots$	$x_{140,100}^{(i)}$

Convolutional Neural Networks

Filtros

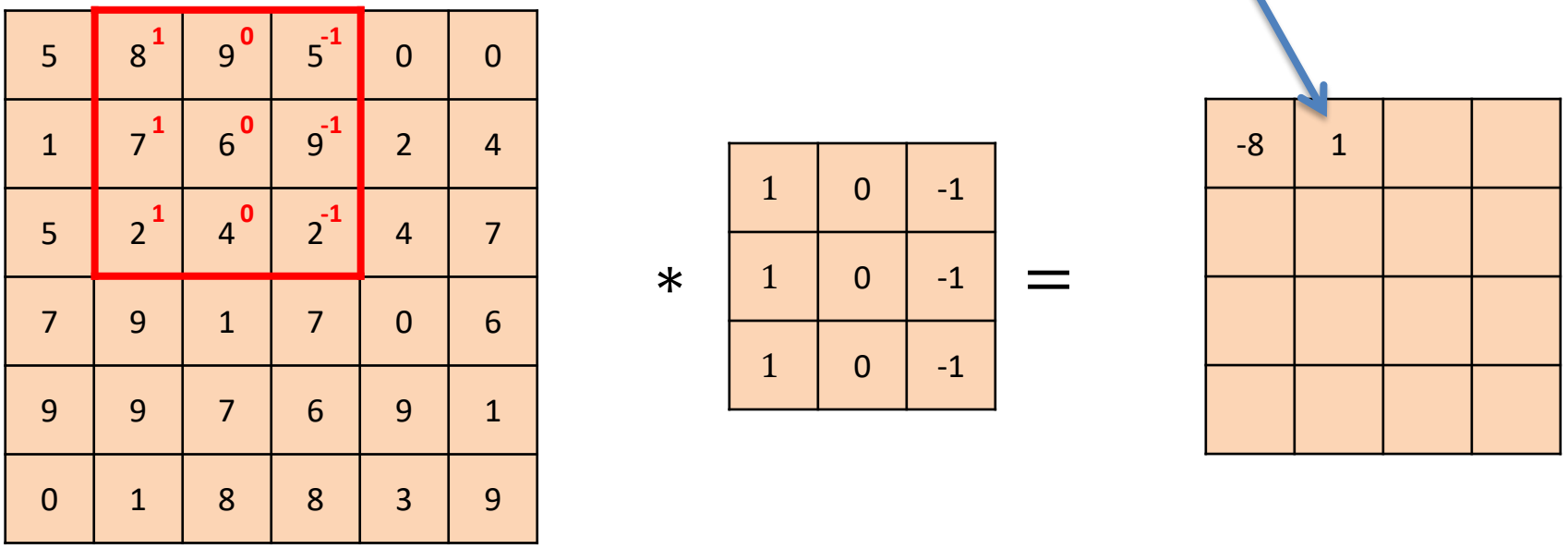
$$5 + 0 - 9 + 1 + 0 - 6 + 5 + 0 - 4 = -8$$



Convolutional Neural Networks

Filtros

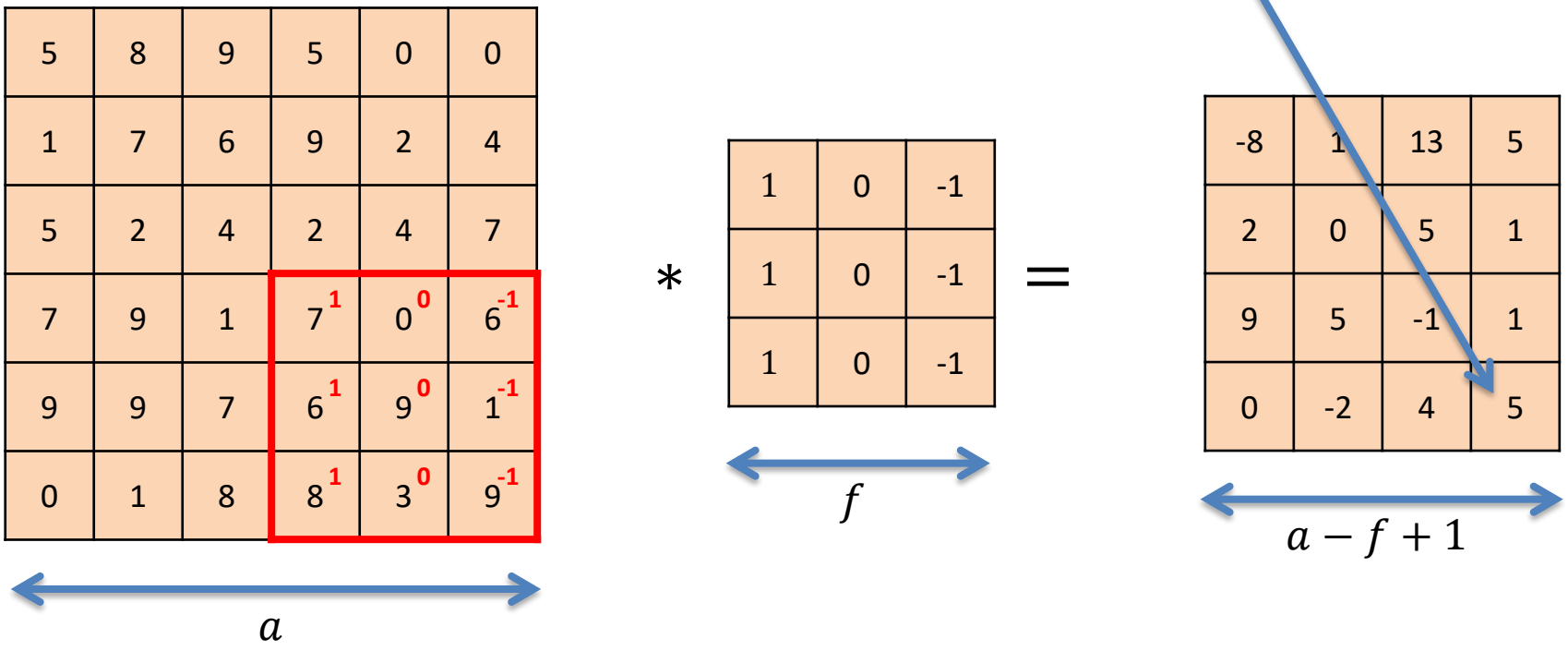
$$8 + 0 - 5 + 7 + 0 - 9 + 2 + 0 - 2 = 1$$



Convolutional Neural Networks

Filtros

$$7 + 0 - 6 + 6 + 0 - 1 + 8 + 0 - 9 = 5$$



Convolutional Neural Networks

Filtros

0 (1)	0 (0)	0 (-1)	0	0	0	0	0
0 (1)	5 (0)	8 (-1)	9	5	0	0	0
0 (1)	1 (0)	7 (-1)	6	9	2	4	0
0	5	2	4	2	4	7	0
0	7	9	1	7	0	6	0
0	9	9	7	6	9	1	0
0	0	1	8	8	3	9	0
0	0	0	0	0	0	0	0

*

1	0	-1
1	0	-1
1	0	-1

=

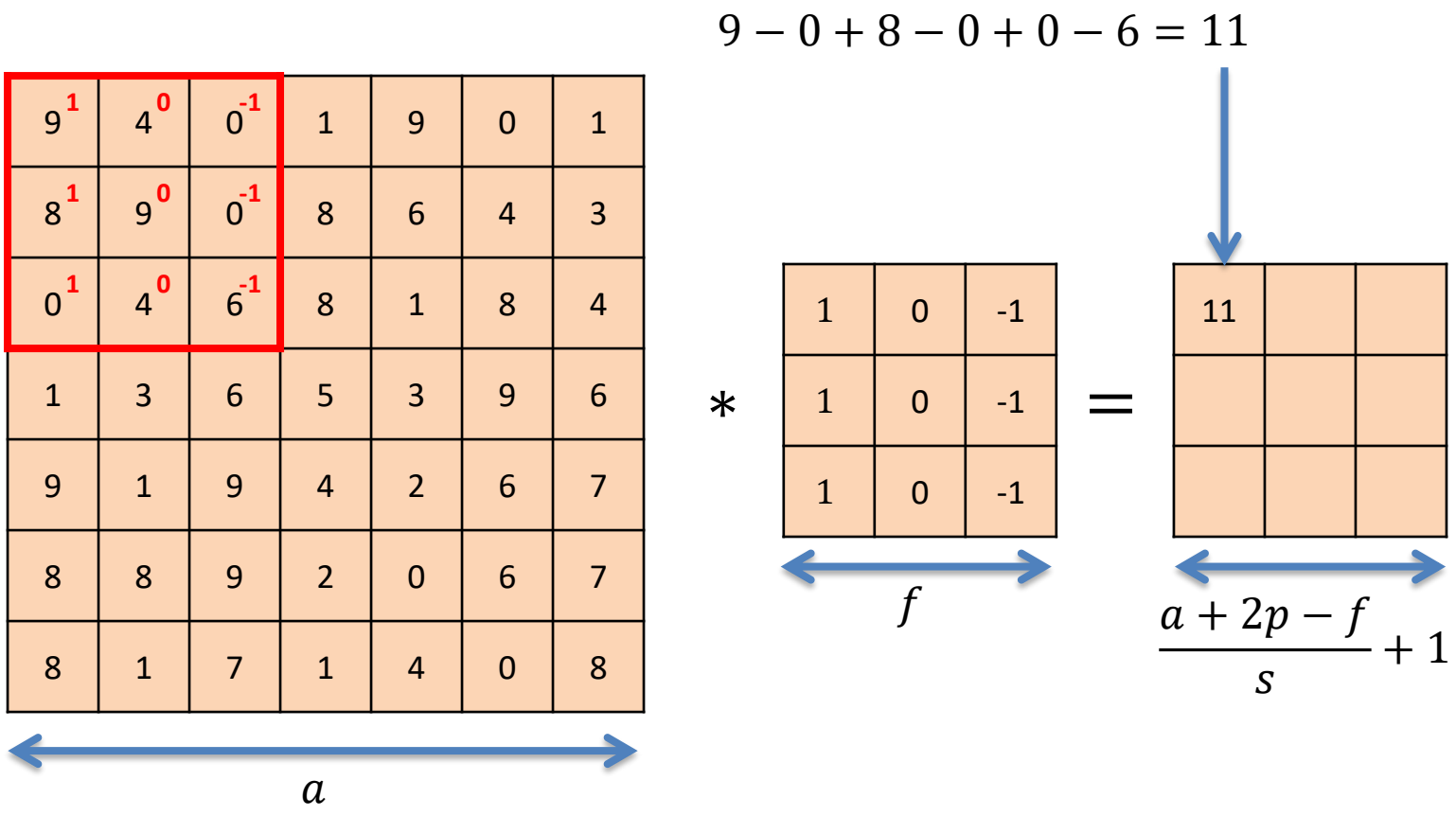
-15	-9	1	13	10	2
-17	-8	1	13	5	6
-18	2	0	5	1	6
-20	9	5	-1	1	13
-19	0	-2	4	5	12
-10	-6	-4	3	4	12

Padding (relleno)

"same" → salida y entrada del mismo tamaño

Convolutional Neural Networks

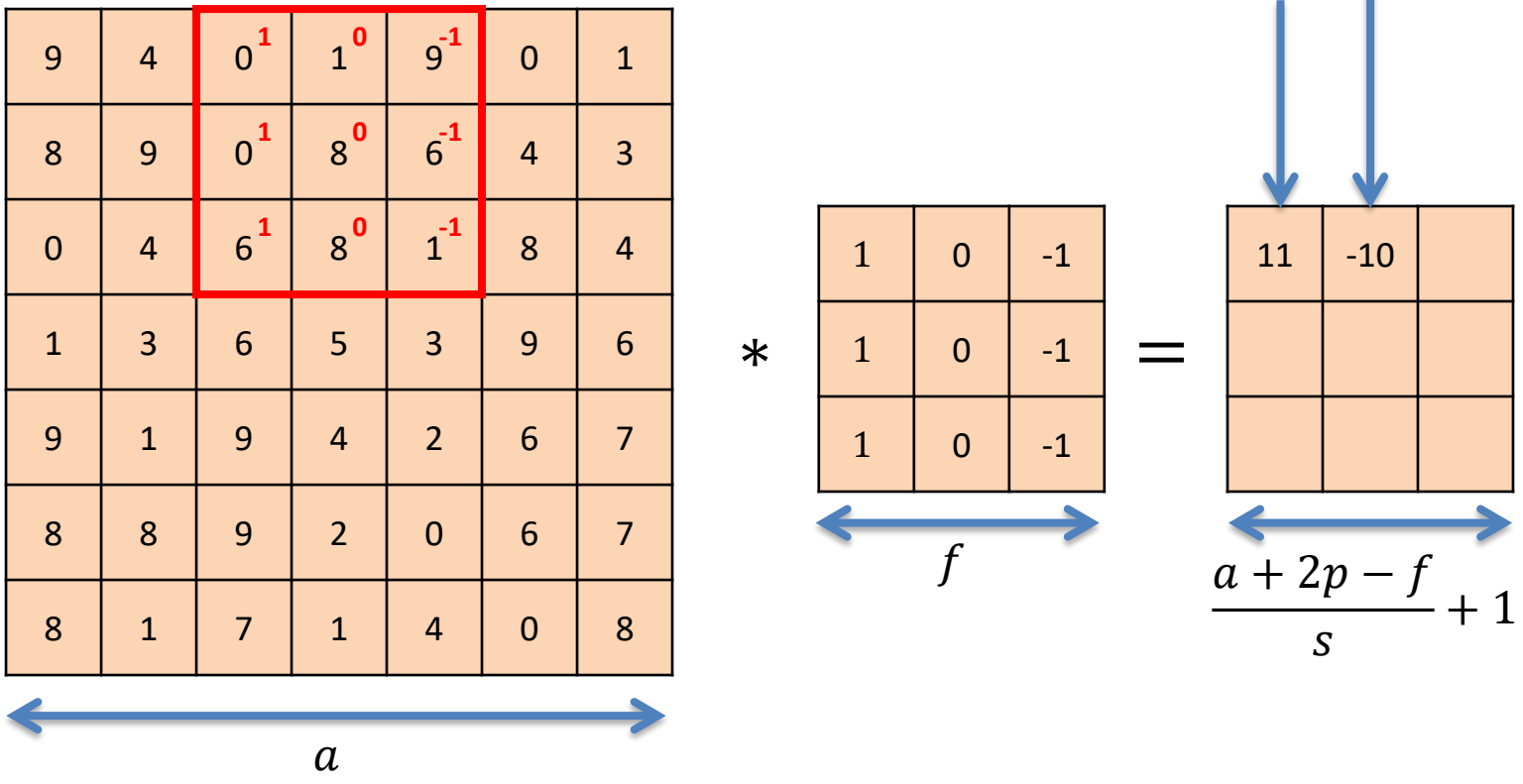
Filtros



Stride (avance)

Convolutional Neural Networks

Filtros



Stride (avance)

Convolutional Neural Networks

Filtros

$$9 - 1 + 6 - 3 + 1 - 4 = 8$$
$$0 - 9 + 0 - 6 + 6 - 1 = -10$$
$$9 - 0 + 8 - 0 + 0 - 6 = 11$$

9	4	0	1	9^{1}	0^{0}	1^{-1}
8	9	0	8	6^{1}	4^{0}	3^{-1}
0	4	6	8	1^{1}	8^{0}	4^{-1}
1	3	6	5	3	9	6
9	1	9	4	2	6	7
8	8	9	2	0	6	7
8	1	7	1	4	0	8

a

$*$

1	0	-1
1	0	-1
1	0	-1

f

$=$

11	-10	8
-11	15	-11
0	19	-16

$$\frac{a + 2p - f}{s} + 1$$

Stride (avance)

Convolutional Neural Networks

Filtros

Detector de bordes verticales

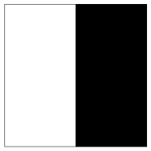
10	**10**	**10**	**10**	0	0	0	0
10	**10**	**10**	**10**	0	0	0	0
10	**10**	**10**	**10**	0	0	0	0
10	**10**	**10**	**10**	0	0	0	0
10	**10**	**10**	**10**	0	0	0	0
10	**10**	**10**	**10**	0	0	0	0
10	**10**	**10**	**10**	0	0	0	0
10	**10**	**10**	**10**	0	0	0	0

$*$

1	0	-1
1	0	-1
1	0	-1

$=$

0	0	**30**	**30**	0	0
0	0	**30**	**30**	0	0
0	0	**30**	**30**	0	0
0	0	**30**	**30**	0	0
0	0	**30**	**30**	0	0
0	0	**30**	**30**	0	0



Convolutional Neural Networks

Filtros

Detector de bordes horizontales

10	**10**	**10**	**10**	**10**	**10**	**10**	**10**
10	**10**	**10**	**10**	**10**	**10**	**10**	**10**
10	**10**	**10**	**10**	**10**	**10**	**10**	**10**
10	**10**	**10**	**10**	**10**	**10**	**10**	**10**
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

*

1	1	1
0	0	0
-1	-1	-1

=

0	0	0	0	0	0
0	0	0	0	0	0
30	**30**	**30**	**30**	**30**	**30**
30	**30**	**30**	**30**	**30**	**30**
0	0	0	0	0	0
0	0	0	0	0	0

Convolutional Neural Networks

Filtros

Detector de bordes diagonales (+45º)

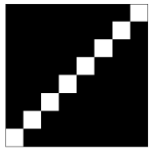
0	0	0	0	0	0	0	**10**
0	0	0	0	0	0	**10**	0
0	0	0	0	0	**10**	0	0
0	0	0	0	**10**	0	0	0
0	0	0	**10**	0	0	0	0
0	0	**10**	0	0	0	0	0
0	**10**	0	0	0	0	0	0
10	0	0	0	0	0	0	0

*

-1	0	0
0	2	0
0	0	-1

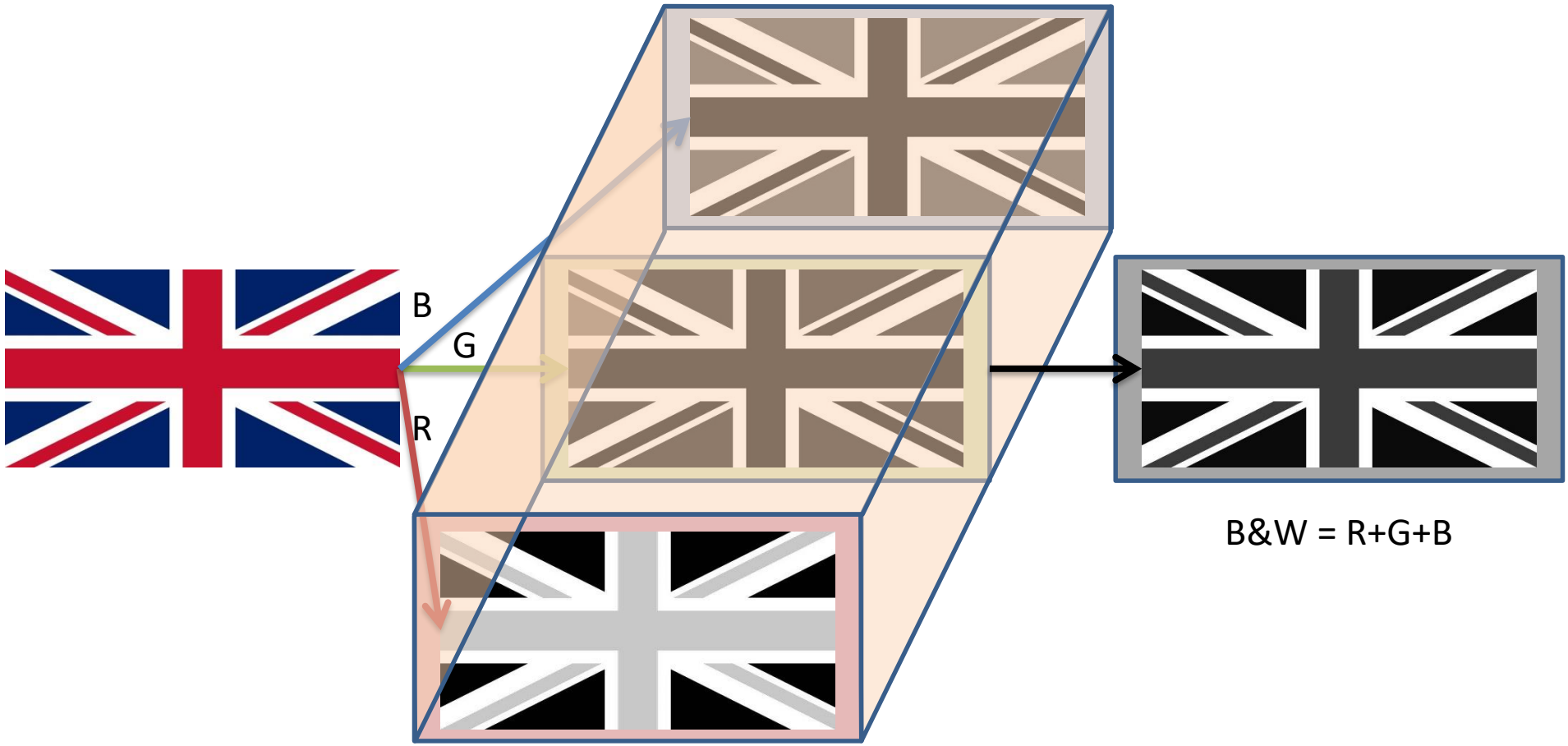
=

0	0	0	**-10**	0	**20**
0	0	**-10**	0	**20**	0
0	**-10**	0	**20**	0	**-10**
-10	0	**20**	0	**-10**	0
0	**20**	0	**-10**	0	0
20	0	**-10**	0	0	0



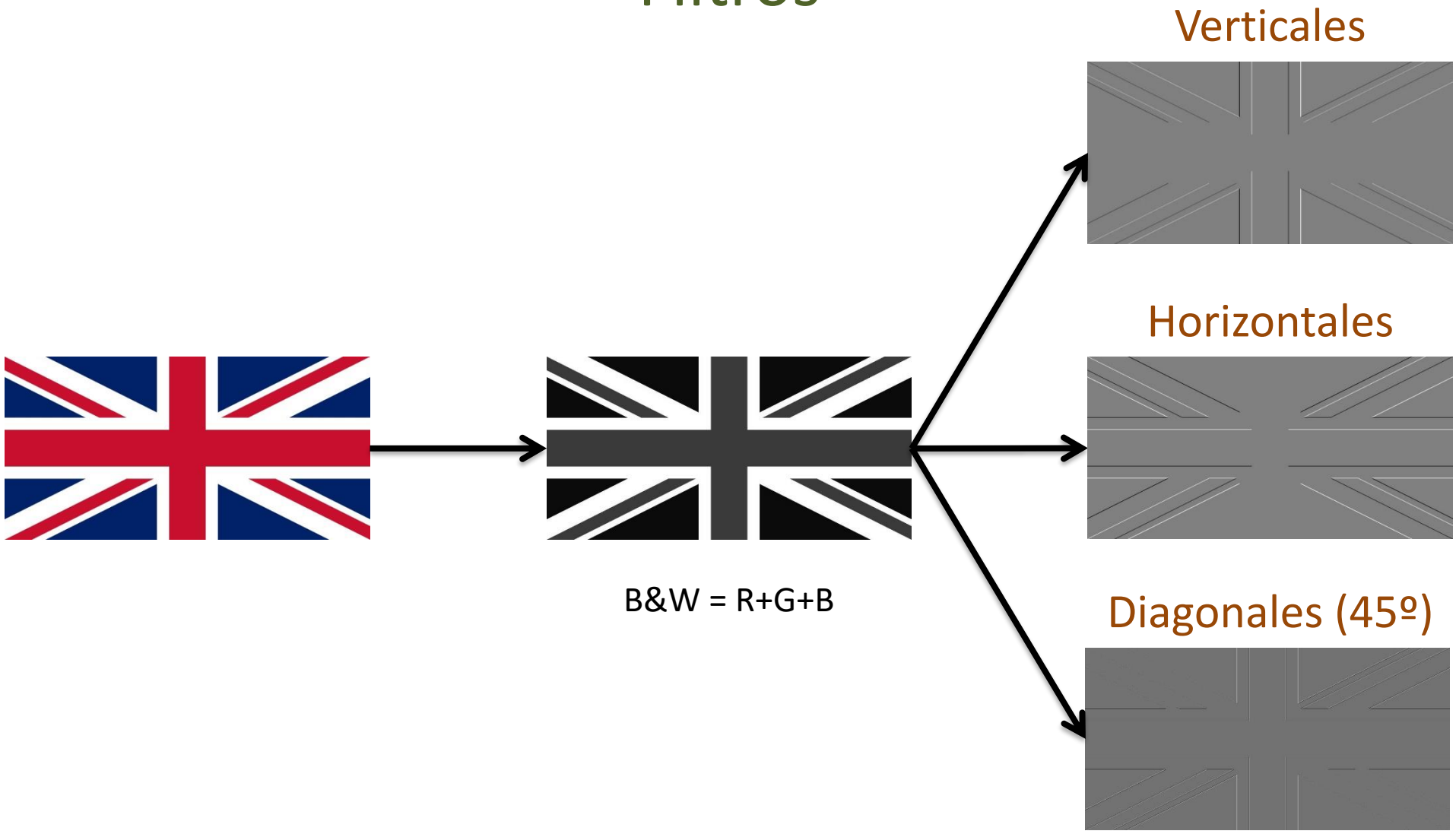
Convolutional Neural Networks

Filtros



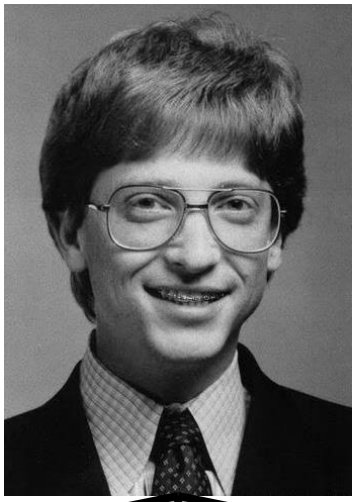
Convolutional Neural Networks

Filtros

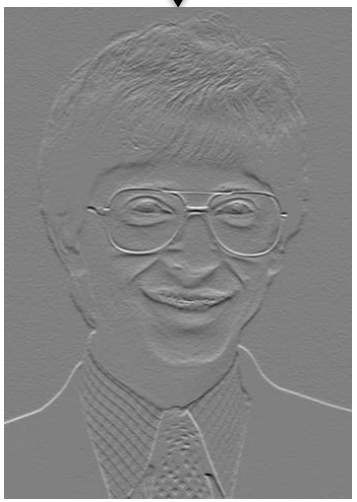


Convolutional Neural Networks

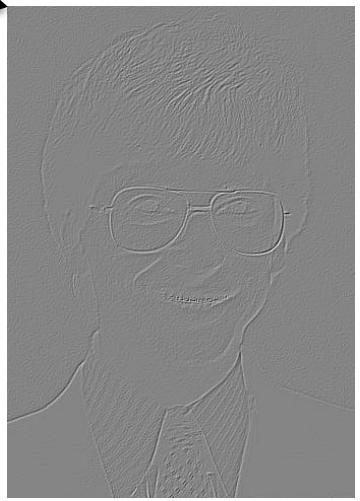
Filtros



Verticales



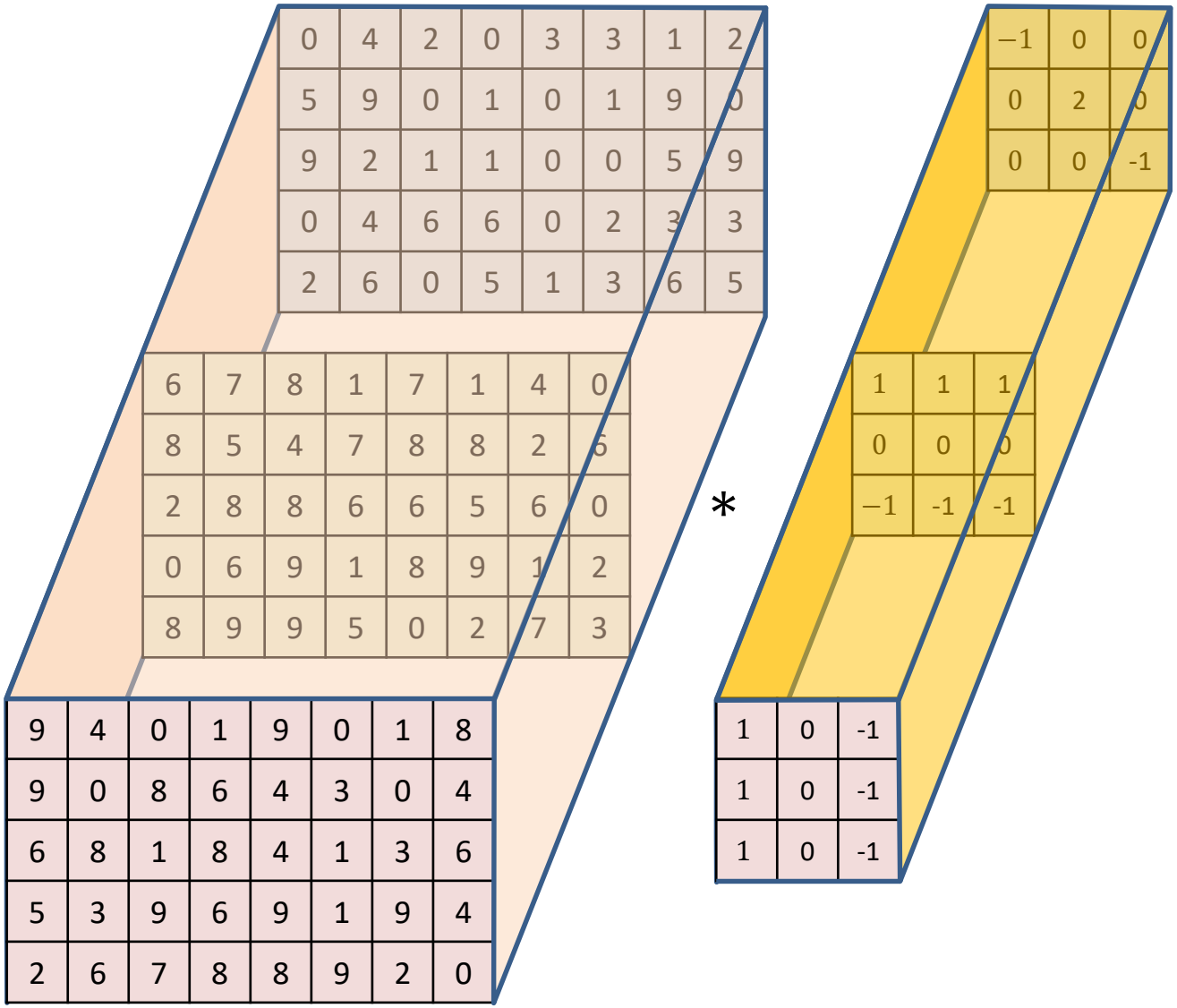
Horizontales



Diagonales (45º)

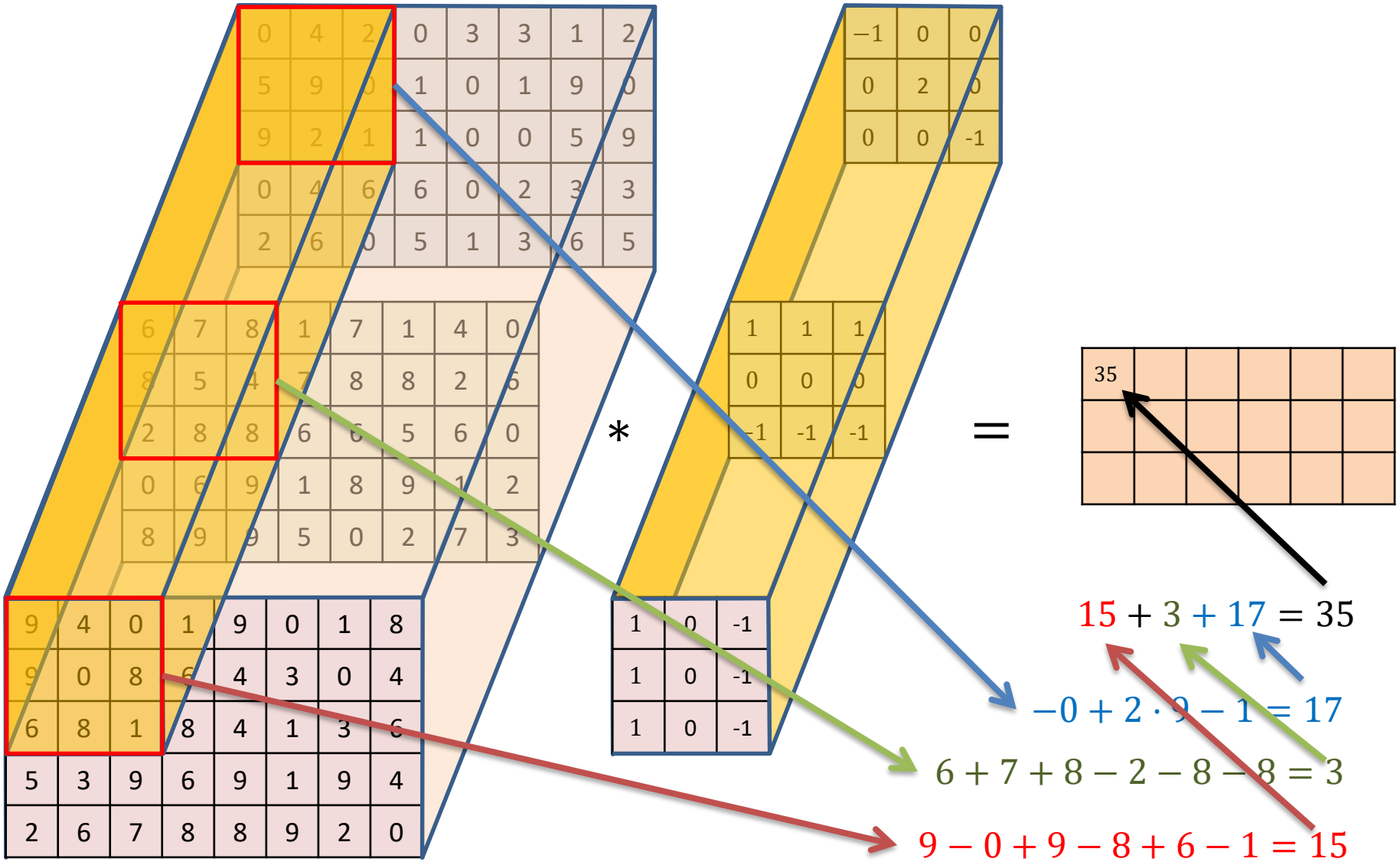
Convolutional Neural Networks

Filtros



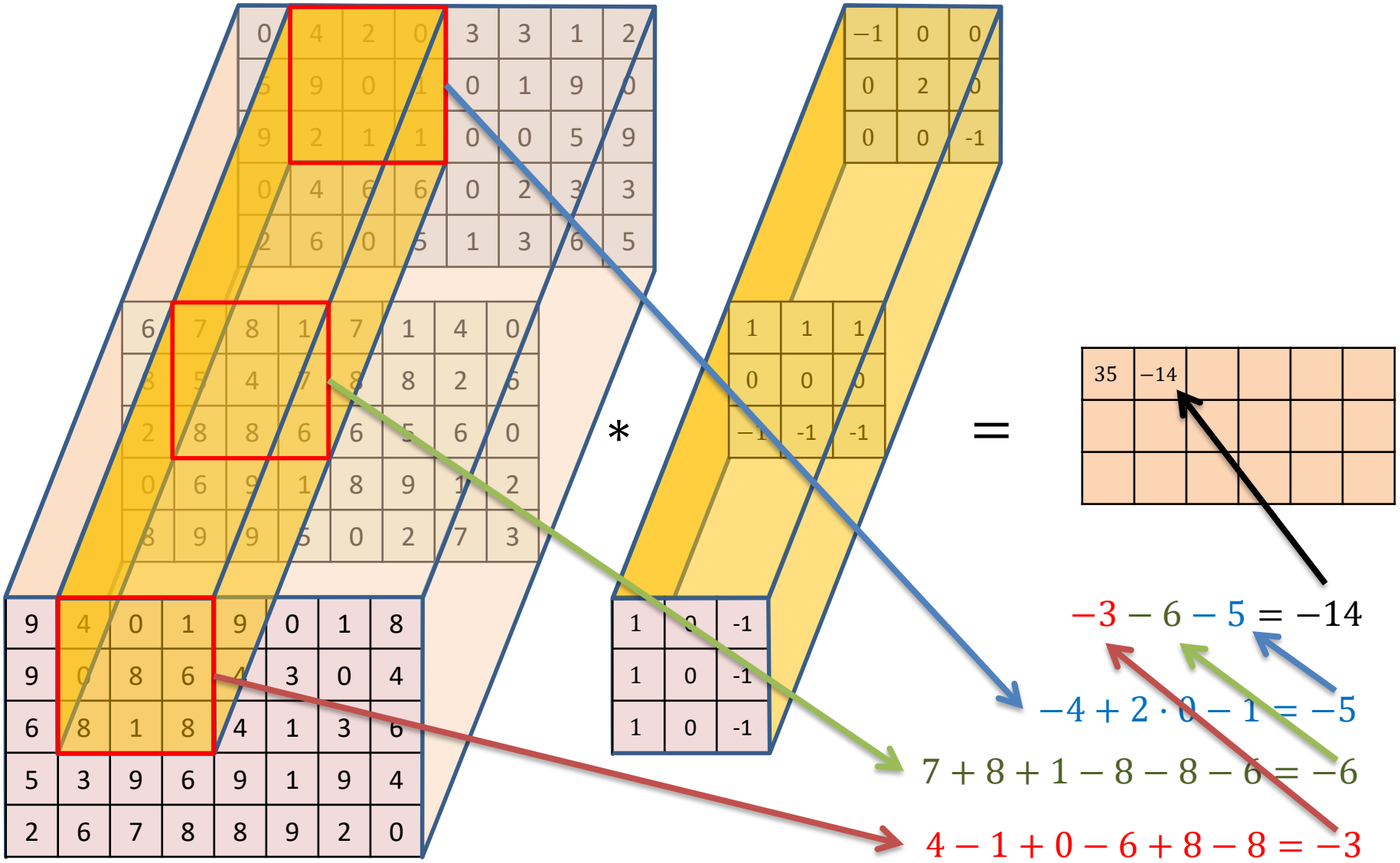
Convolutional Neural Networks

Filtros



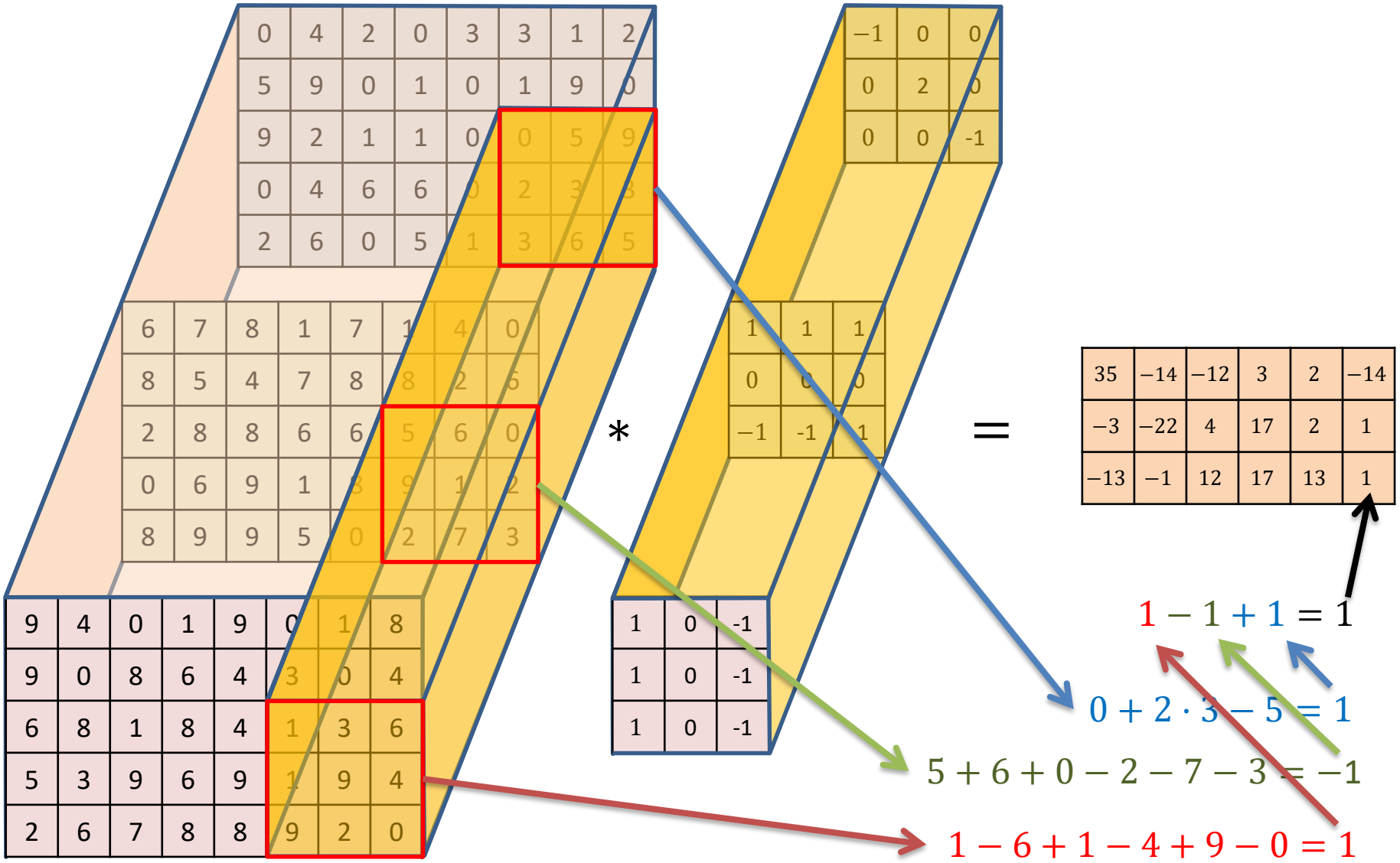
Convolutional Neural Networks

Filtros



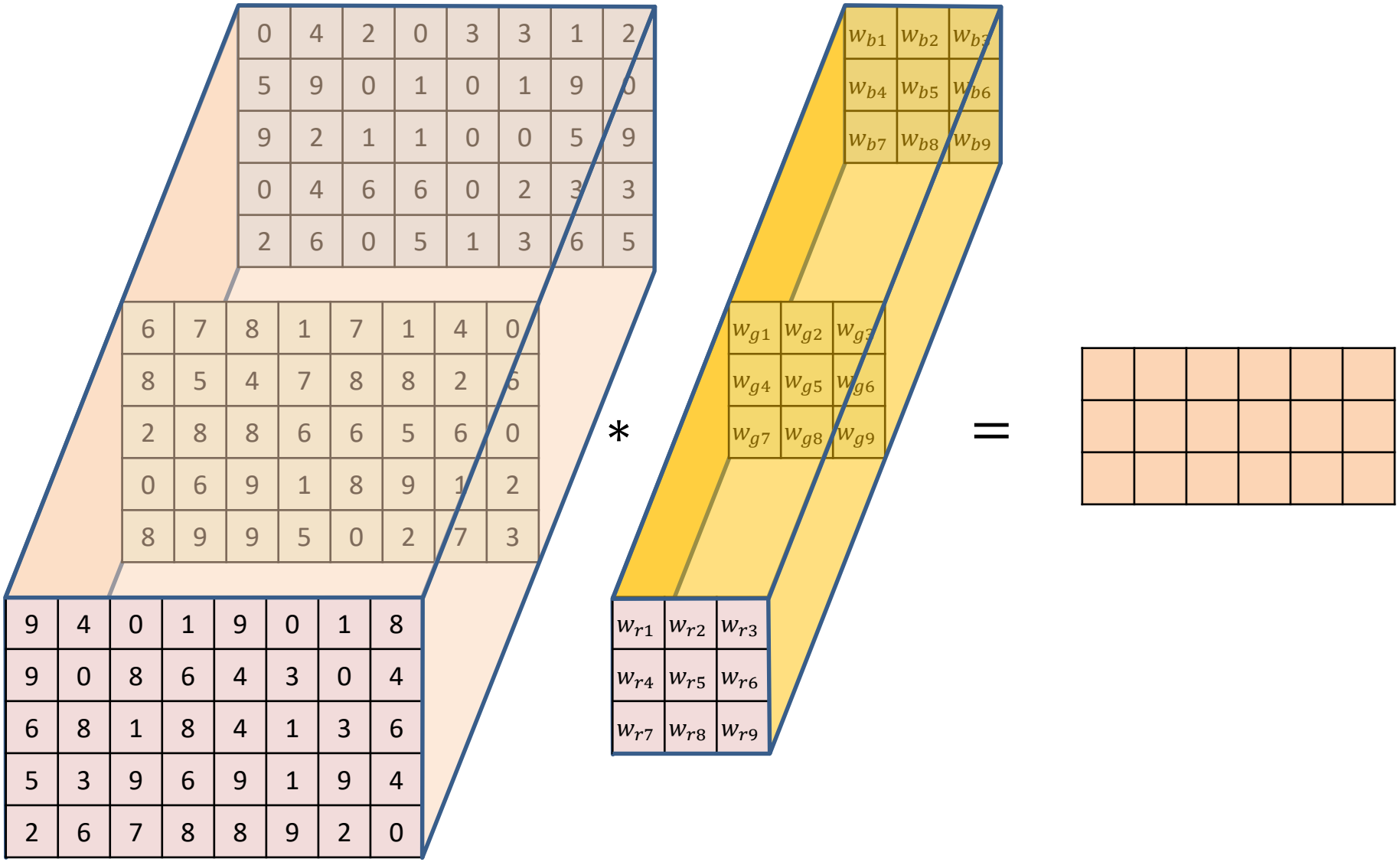
Convolutional Neural Networks

Filtros



Convolutional Neural Networks

Filtros

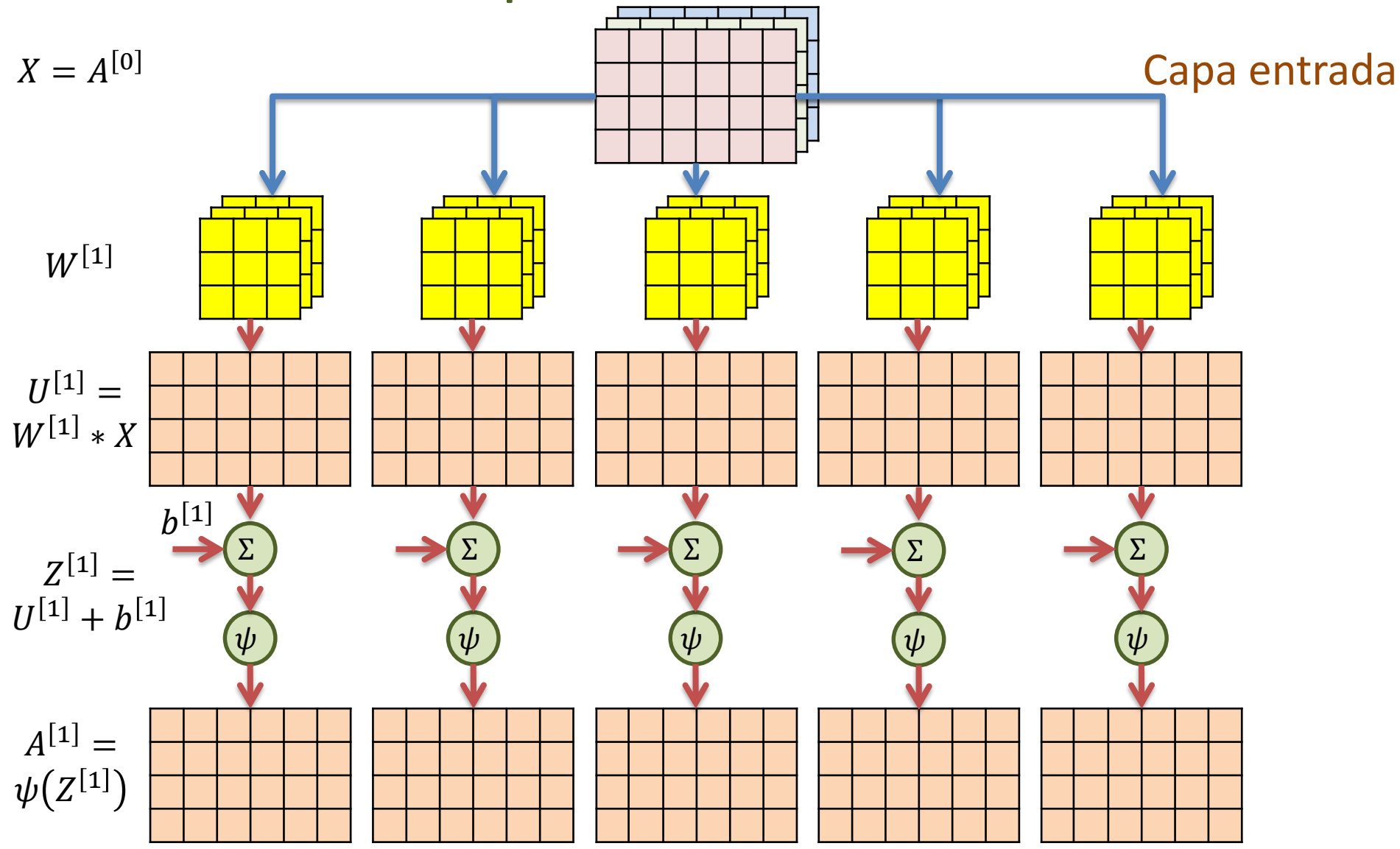


Deep Learning

- Conceptos generales
- Redes convolucionales
  - Conceptos generales
    - Número de parámetros
    - Tensores
  - Filtros
  - Tipos de capas
    - Convolucional
    - Agrupación (pooling)
    - Completamente conectada
  - Ejemplos
    - LeNet-5
    - AlexNet
  - Transfer learning
- Redes recurrentes

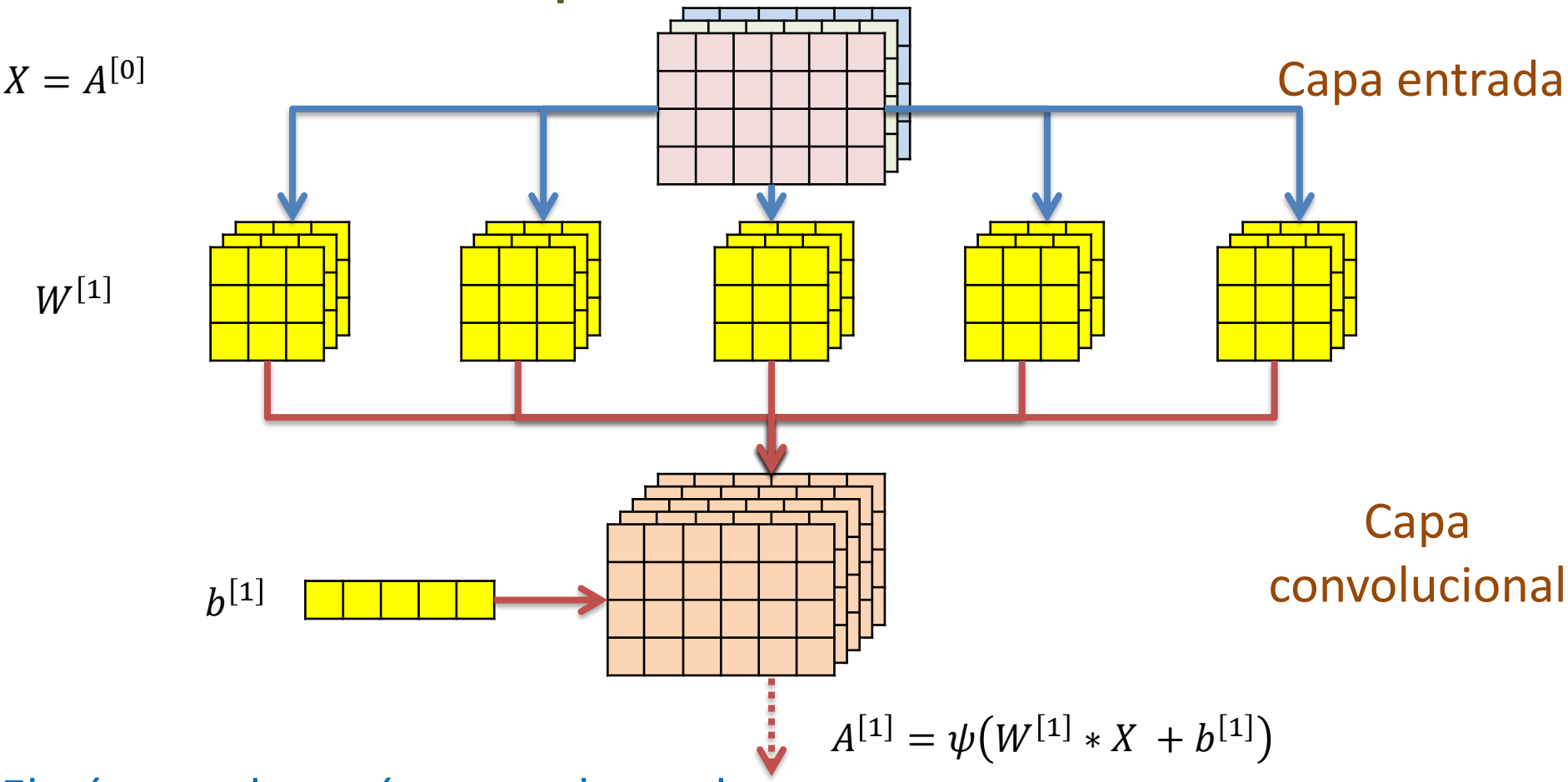
Convolutional Neural Networks

Capa convolucional



Convolutional Neural Networks

Capa convolucional



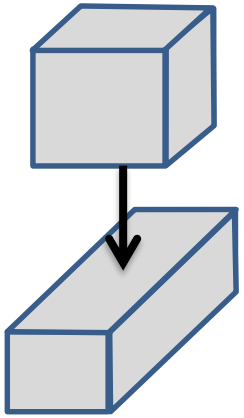
El número de parámetros depende

- del número y tamaño de los filtros
- no del tamaño de la entrada

Convolutional Neural Networks

Capa convolucional

$X = A^{[0]}$



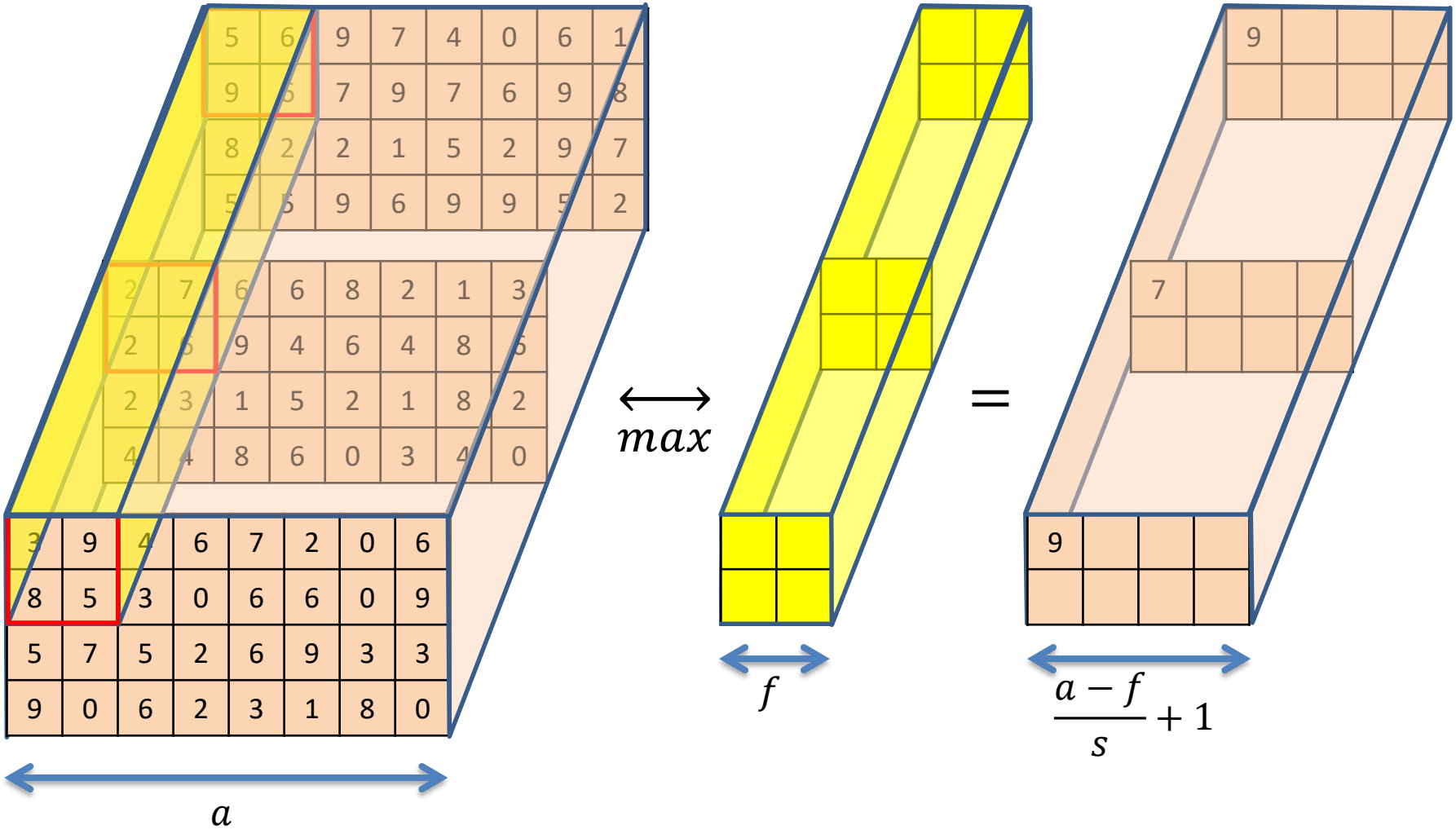
Input

Convolutional

$A^{[1]}$

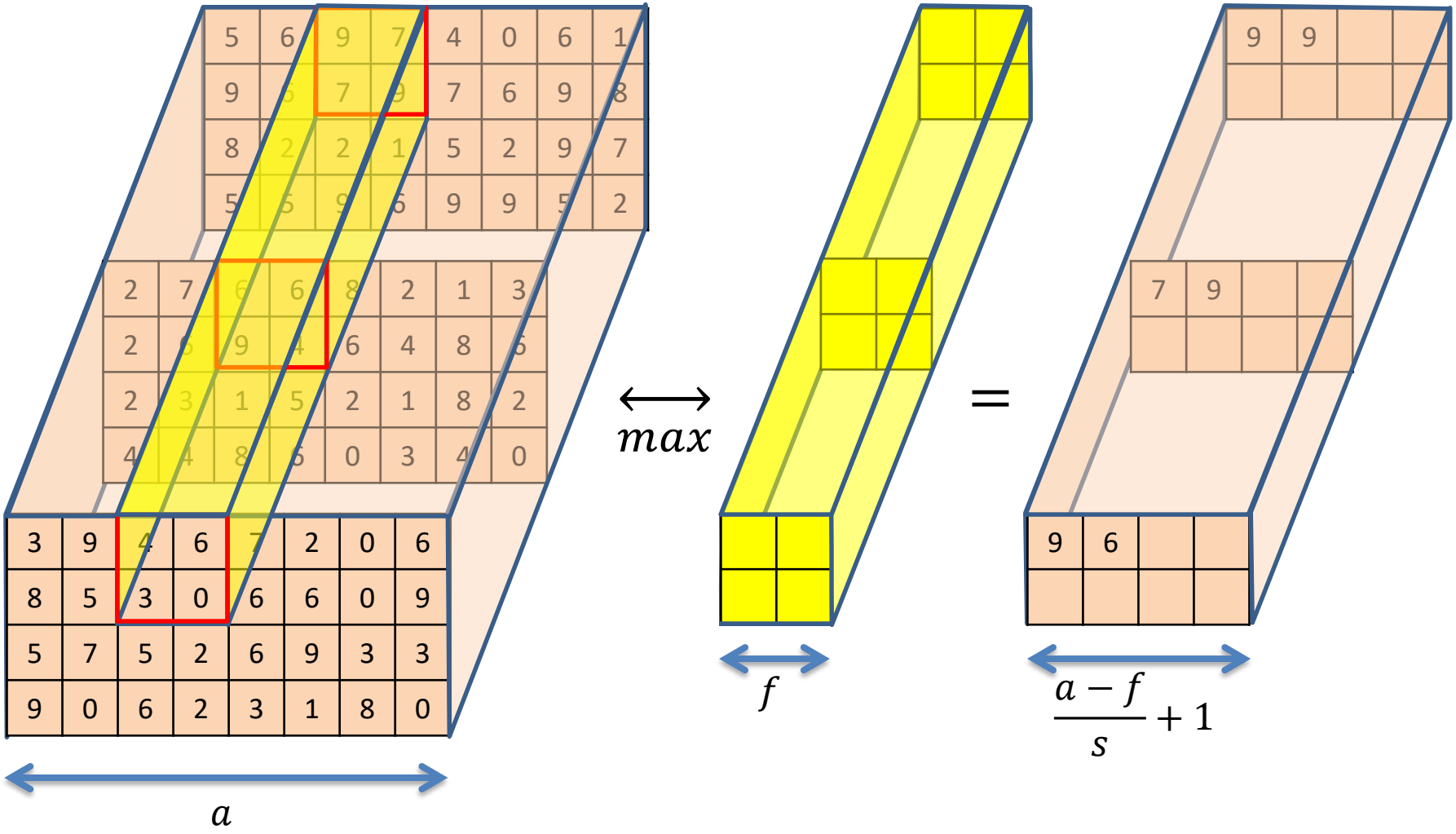
Convolutional Neural Networks

Capa de agrupación (pooling)



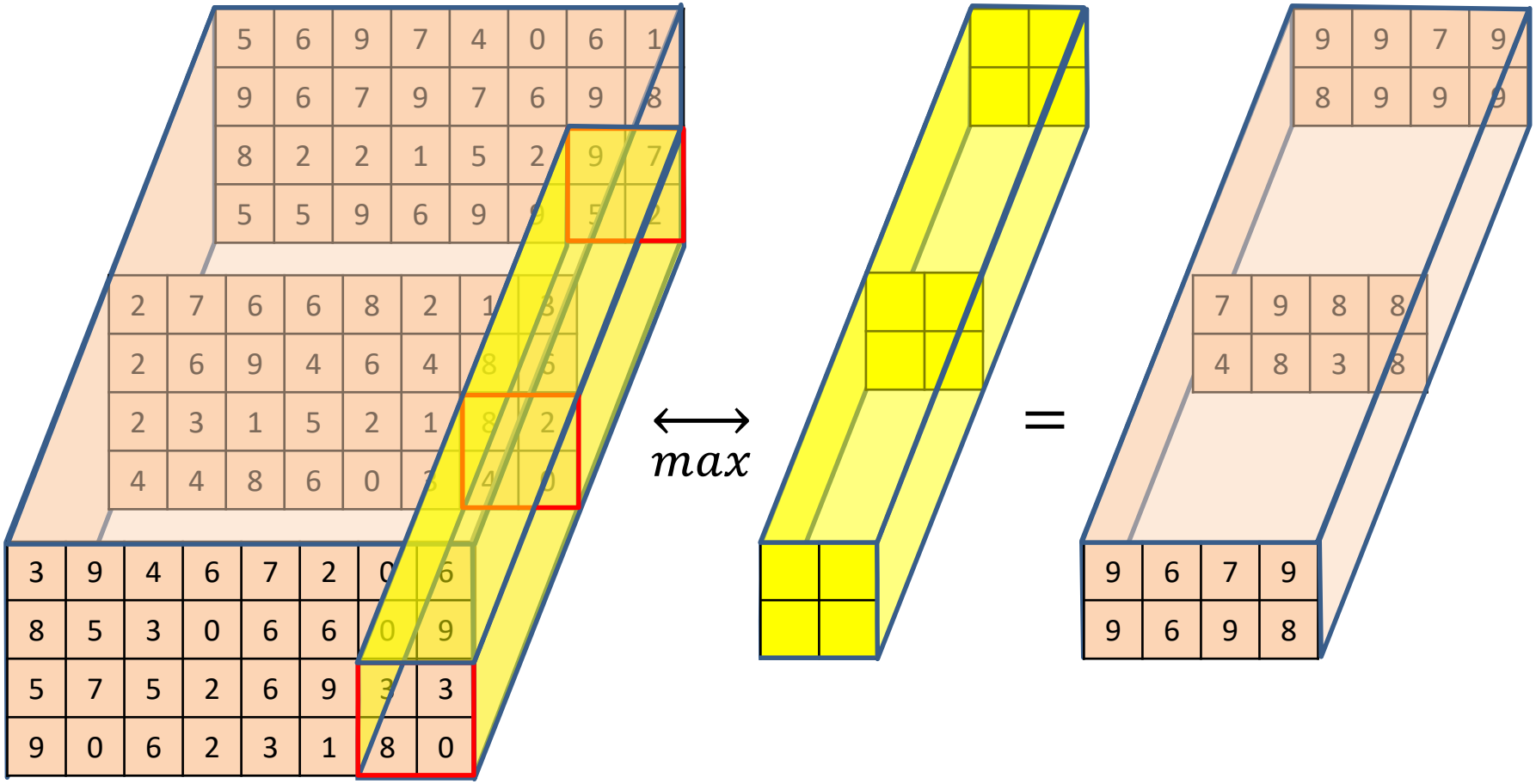
Convolutional Neural Networks

Capa de agrupación (pooling)



Convolutional Neural Networks

Capa de agrupación (pooling)

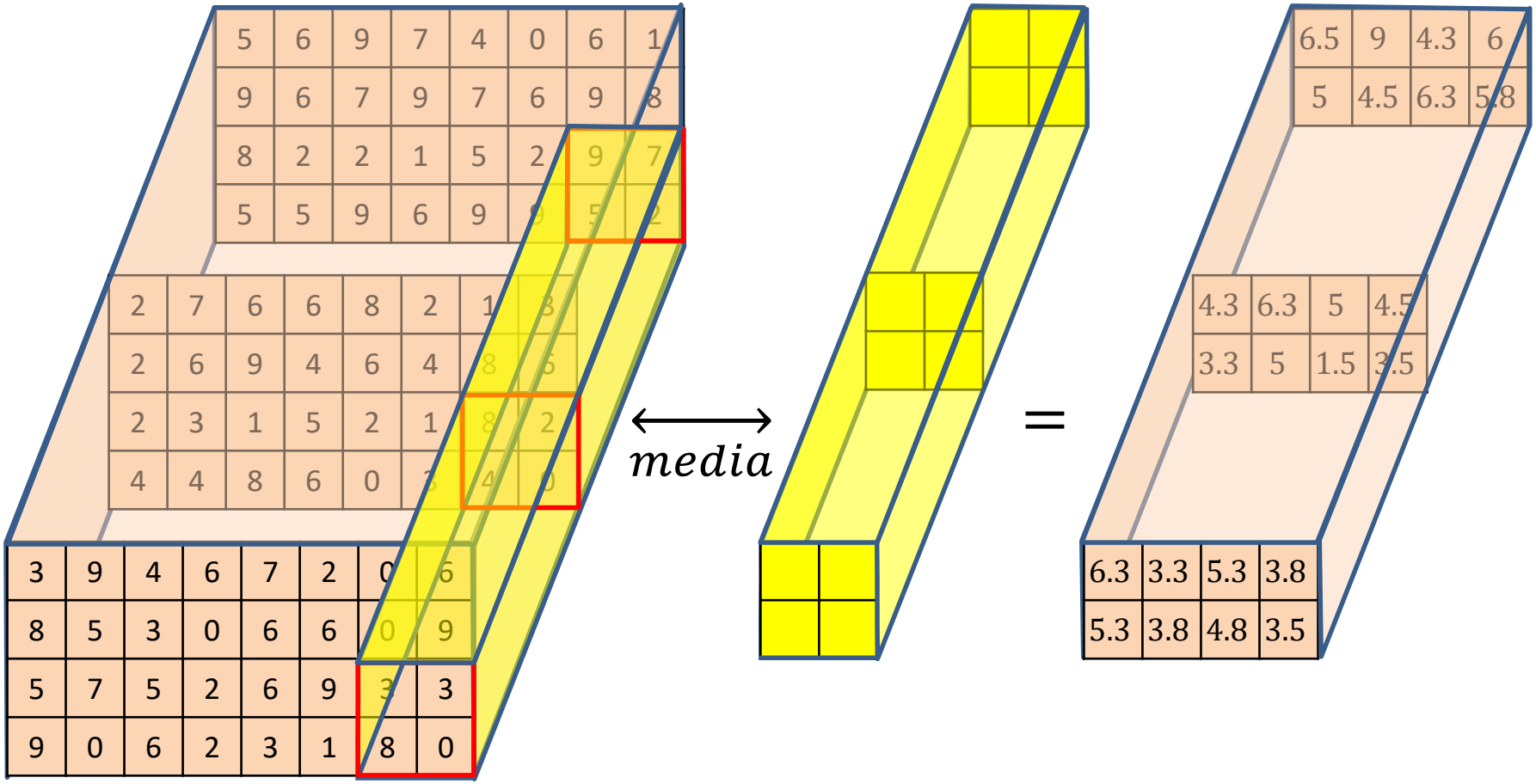


Reducción del tamaño (subsampling)

Resumen de características de una zona

Convolutional Neural Networks

Capa de agrupación (pooling)



Reducción del tamaño (subsampling)

Resumen de características de una zona

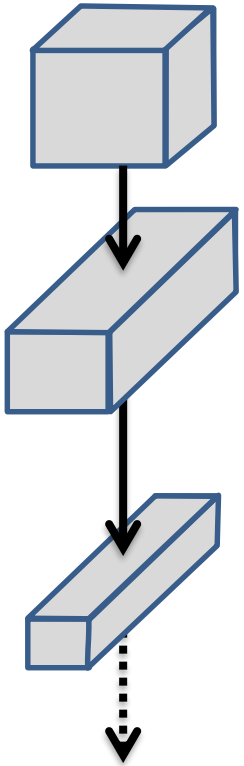
Convolutional Neural Networks

Capa de agrupación (pooling)

$X = A^{[0]}$

$A^{[1]}$

$A^{[2]}$



Input

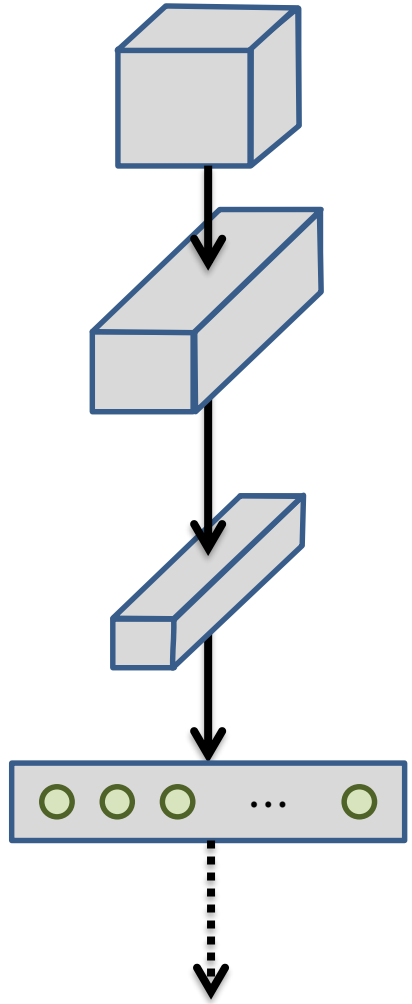
Convolutional

Pooling

Convolutional Neural Networks

Capa completamente conectada

$X = A^{[0]}$



Input

$A^{[1]}$

Convolutional

$A^{[2]}$

Pooling

$A^{[3]}$

Fully connected

Deep Learning

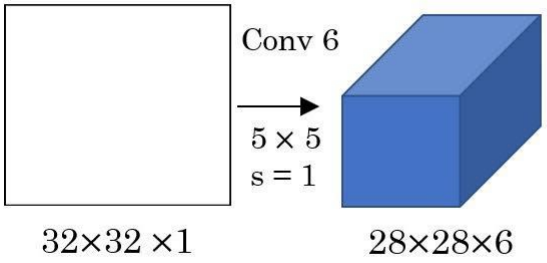
- Conceptos generales
- Redes convolucionales
  - Conceptos generales
    - Número de parámetros
    - Tensores
  - Filtros
  - Tipos de capas
    - Convolucional
    - Agrupación (pooling)
    - Completamente conectada
  - Ejemplos
    - LeNet-5
    - AlexNet
  - Transfer learning
- Redes recurrentes

Convolutional Neural Networks

LeNet-5

60k parámetros

$$5 \cdot 5 \cdot 6 + 6 = 156$$



$$\frac{a + 2p - f}{s} + 1 = \frac{32 - 5}{1} + 1 = 28$$

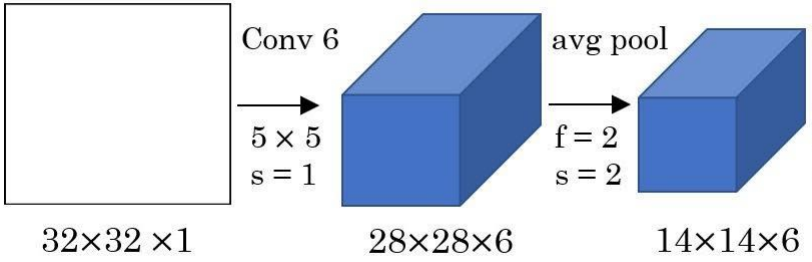
https://harangdev.github.io/deep-learning/convolutional-neural-networks/25/

LeCun et al., 1998. Gradient-based learning applied to document recognition

Convolutional Neural Networks

LeNet-5

~60k parámetros



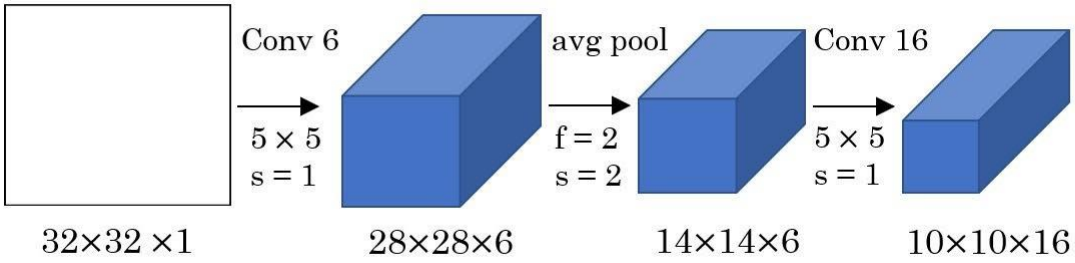
$$\frac{a-f}{s}+1=\frac{28-2}{2}+1=14$$

Convolutional Neural Networks

LeNet-5

~60k parámetros

$$5 \cdot 5 \cdot 16 + 16 = 416$$

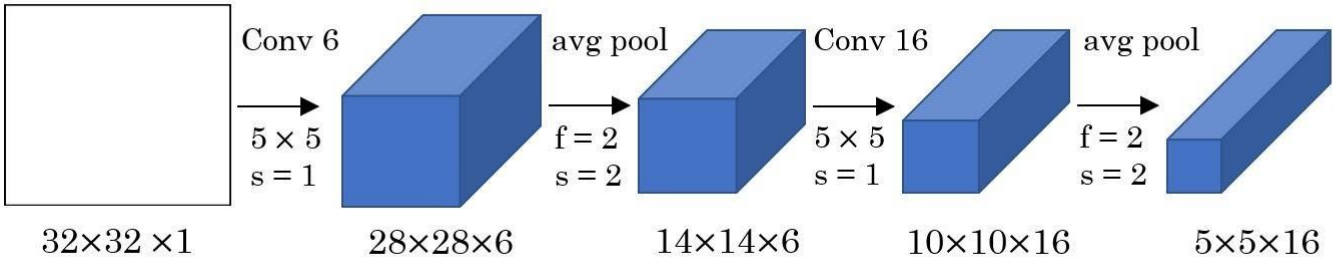


$$\frac{a + 2p - f}{s} + 1 = \frac{14 - 5}{1} + 1 = 10$$

Convolutional Neural Networks

LeNet-5

~60k parámetros



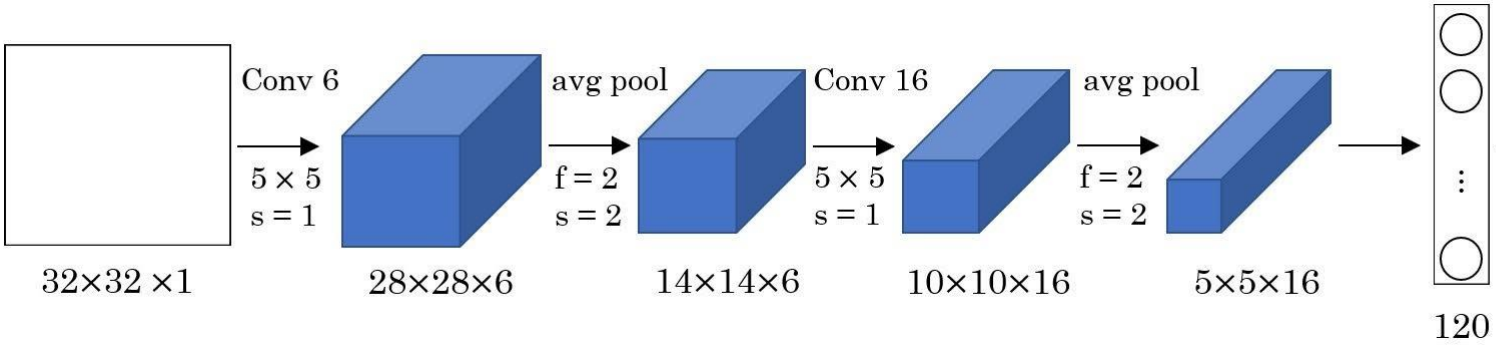
$$\frac{a-f}{s}+1=\frac{10-2}{2}+1=5$$

Convolutional Neural Networks

LeNet-5

~60k parámetros

$$120 \cdot 400 + 120 = 48{,}120$$



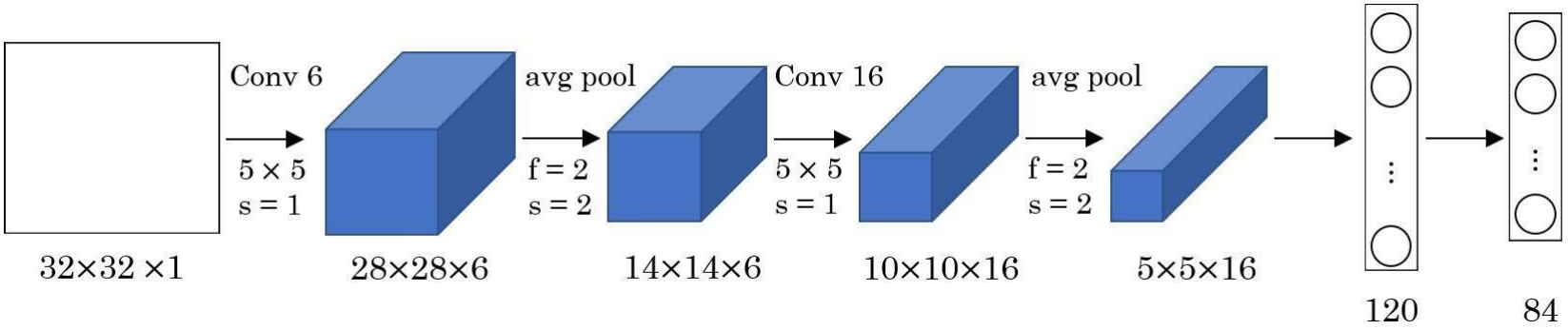
$$5 \times 5 \times 16 = 400$$

Convolutional Neural Networks

LeNet-5

~60k parámetros

$$84 \cdot 120 + 84 = 10{,}164$$

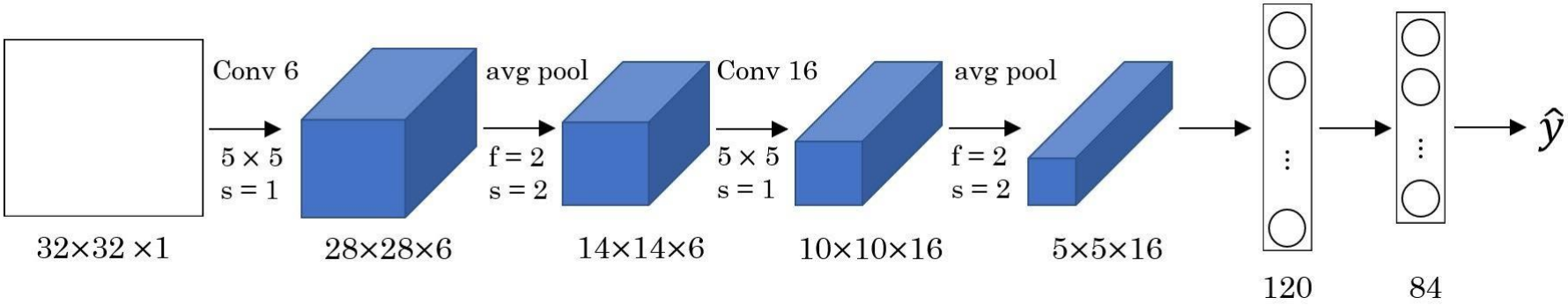


Convolutional Neural Networks

LeNet-5

~60k parámetros

$$10 \cdot 84 + 10 = 850$$



Parámetros:

- Convolutional: 572
- Pooling: 0
- Fully connected: 58,284
- Sotmax: 850

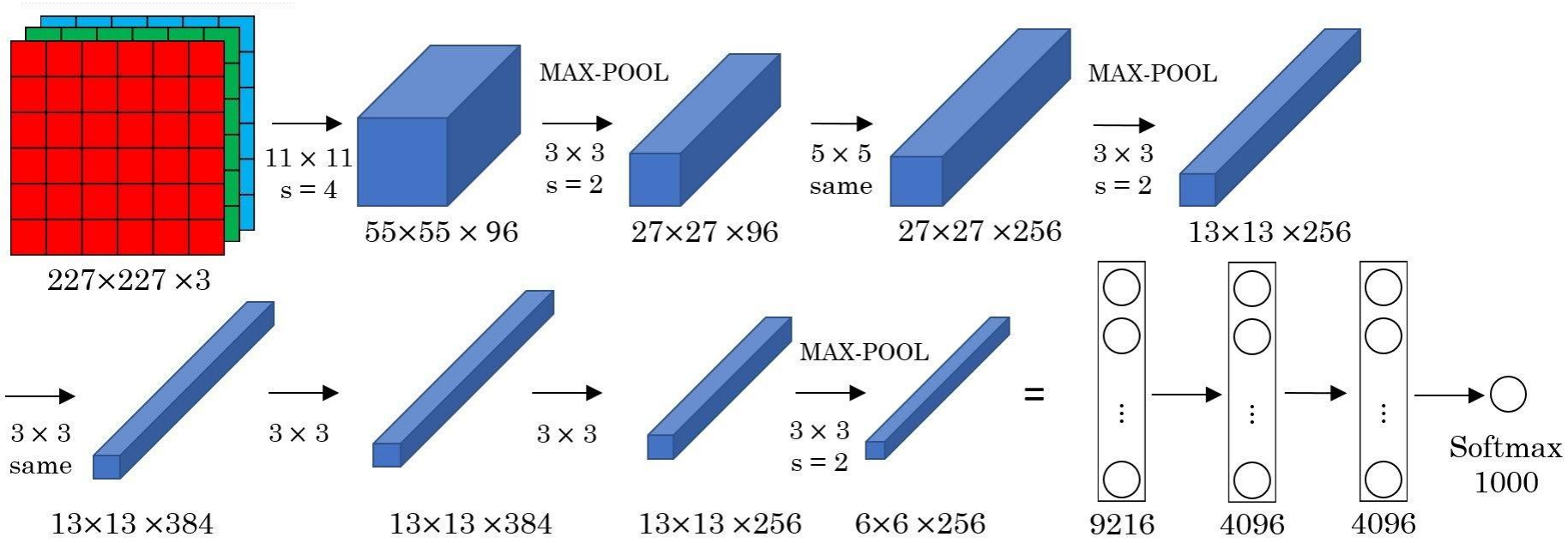
TOTAL: 59,706

Softmax-10

Convolutional Neural Networks

AlexNet

~60M parámetros

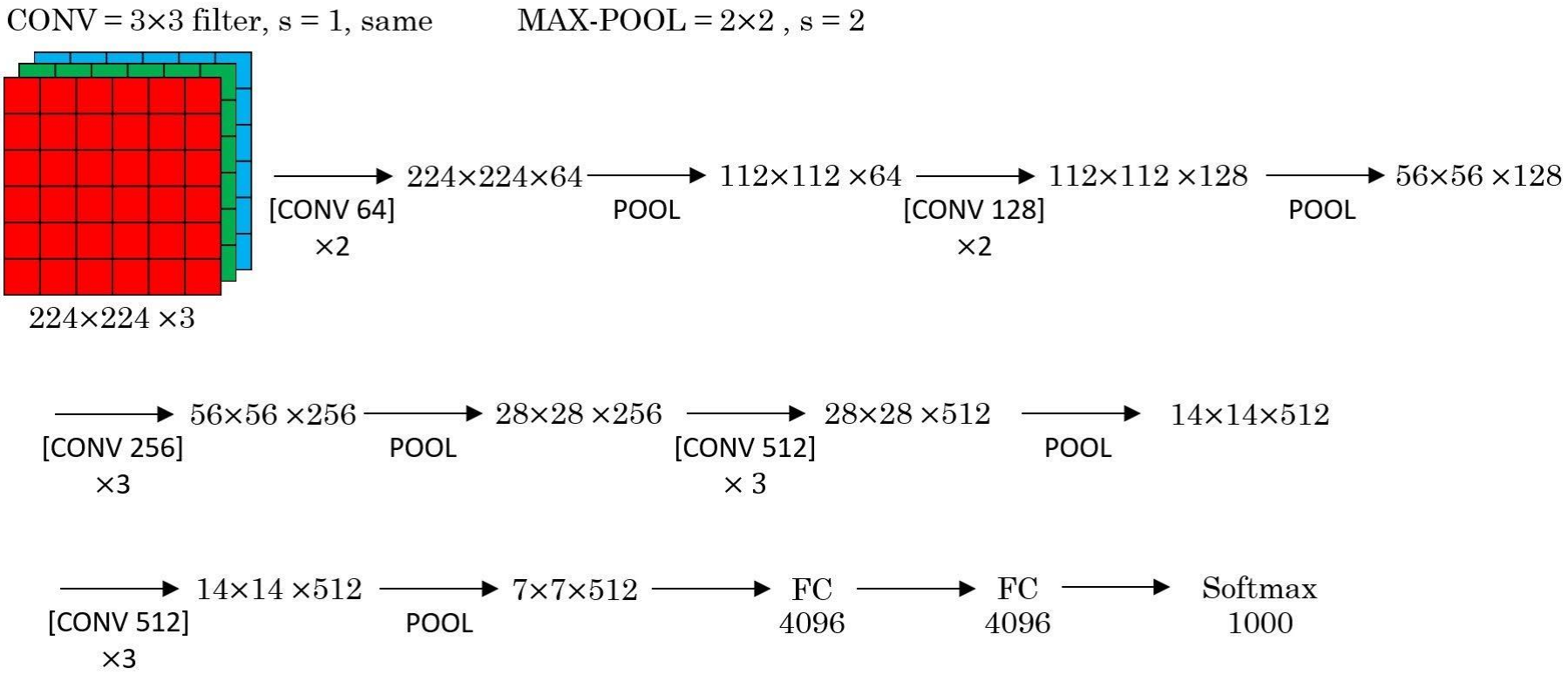


Krizhevsky et al., 2012. ImageNet classification with deep convolutional neural networks

Convolutional Neural Networks

AlexNet

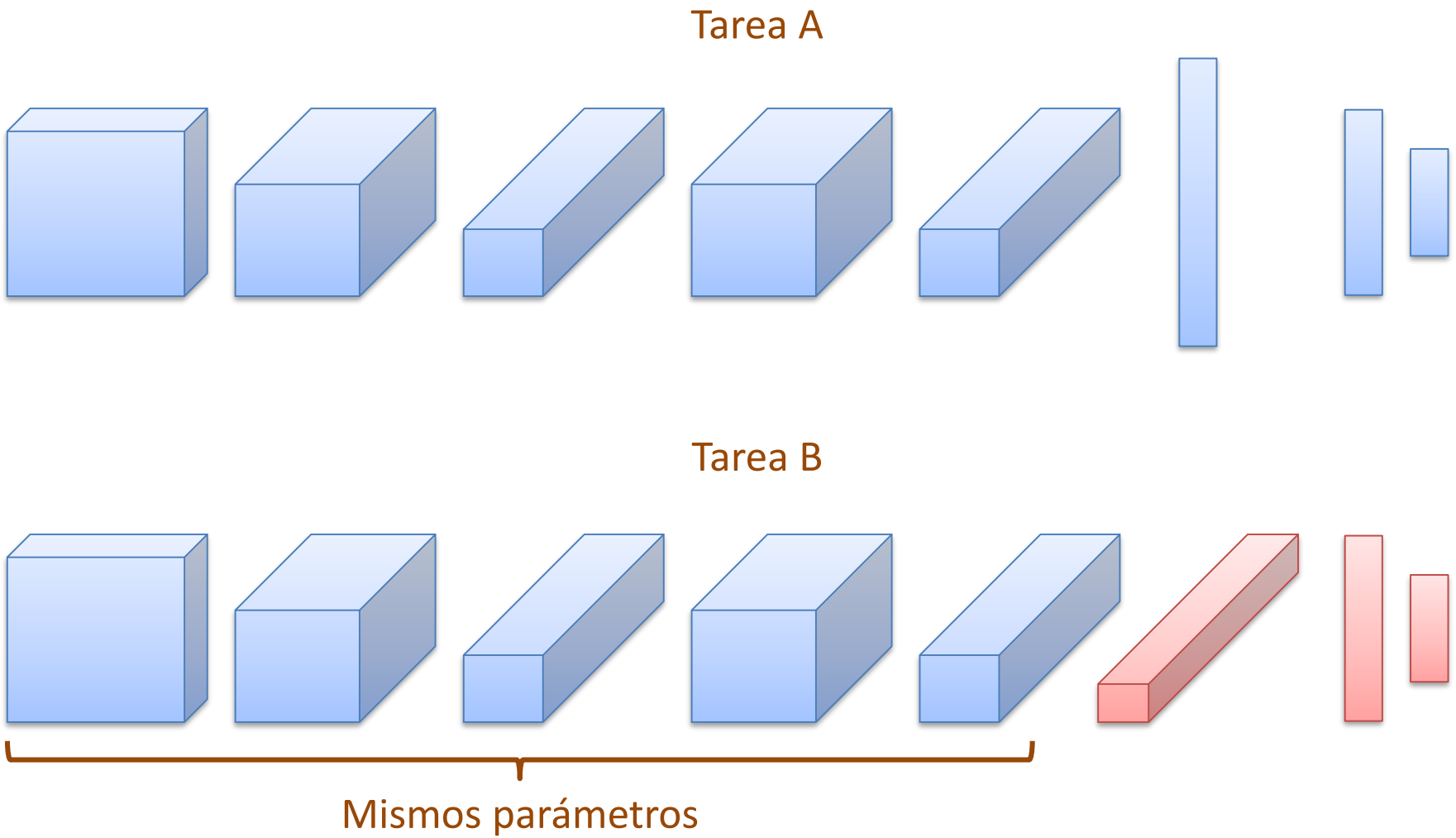
~138M parámetros



Simonyan & Zisserman 2015. Very deep convolutional networks for large-scale image recognition

Convolutional Neural Networks

Transfer Learning



Deep Learning

- Conceptos generales
- Redes convolucionales
- Redes recurrentes
  - Conceptos generales
    - Datos secuenciales
    - FCN para secuencias
  - Representación simplificada
  - Natural Language Processing (NLP)
    - One-hot encoding
  - Tipos de RNN
  - Backpropagation en el tiempo
    - Gated Recurrent Unit (GRU)
    - Long Short Term Memory (LSTM)

Recurrent Neural Networks
Datos secuenciales (ejemplos)

- Series temporales
  - p.e.: datos de demanda de energía, …
    - Clasificación, predicción,…
- Natural Language Processing (NLP)
  - Identificación de nombres propios
  - Análisis de sentimiento (opinión)
  - Traducción automática
- Otras aplicaciones
  - Reconocimiento de voz
  - Generación de música
  - Análisis de secuencias de ADN

Recurrent Neural Networks
Datos secuenciales (ejemplos)

- En una cierta empresa eléctrica existe un call-center dedicado a la resolución de incidencias.
- Cuando se produce una incidencia en la red, son varias las llamadas que se realizan cuya secuencia permite realizar un mejor diagnóstico del problema.
- Cada llamada está definida por un conjunto de características que incluyen nombre del cliente, localización, tipo de incidencia,… .
- Para la resolución de la incidencia el call-center genera una secuencia de órdenes de trabajo, de un catálogo de posibles actuaciones protocolizadas.
- El objetivo es automatizar la producción de las secuencias de órdenes de trabajo.

Recurrent Neural Networks

Datos secuenciales (ejemplos)

Secuencia de llamadas del i-ésimo día

$$x^{(i)}: x^{(i)}_{\langle 1\rangle}, x^{(i)}_{\langle 2\rangle}, \cdots, x^{(i)}_{\langle L_i\rangle}$$

Secuencia de palabras de la i-ésima frase

t-ésima llamada del i-ésimo día

$$x^{(i)}_{\langle t\rangle} = \left(x^{(i)}_{\langle t\rangle 1}, x^{(i)}_{\langle t\rangle 2}, \cdots, x^{(i)}_{\langle t\rangle d}\right)$$

t-ésima palabra de la i-ésima frase

j-ésima característica de la t-ésima llamada del i-ésimo día

$$x^{(i)}_{\langle t\rangle j}$$

j-ésima característica de la t-ésima palabra de la i-ésima frase

Secuencia de órdenes de trabajo del i-ésimo día

$$y^{(i)}: y^{(i)}_{\langle 1\rangle}, y^{(i)}_{\langle 2\rangle}, \cdots, y^{(i)}_{\langle O_i\rangle}$$

Secuencia de palabras de salida correspondiente a la i-ésima frase

$$x^{(i)}: x^{(i)}_{\langle 1\rangle}, x^{(i)}_{\langle 2\rangle}, \cdots, x^{(i)}_{\langle L_i\rangle} \rightarrow y^{(i)}: y^{(i)}_{\langle 1\rangle}, y^{(i)}_{\langle 2\rangle}, \cdots, y^{(i)}_{\langle O_i\rangle}$$

Recurrent Neural Networks

Datos secuenciales (ejemplos)

$$x^{(i)}: x_{\langle 1 \rangle}^{(i)}, x_{\langle 2 \rangle}^{(i)}, \cdots, x_{\langle L_i \rangle}^{(i)} \rightarrow y^{(i)}: y_{\langle 1 \rangle}^{(i)}, y_{\langle 2 \rangle}^{(i)}, \cdots, y_{\langle O_i \rangle}^{(i)}$$

Matriz de (secuencias de) diseño

$$x^{(1)}: x_{\langle 1 \rangle}^{(1)}, x_{\langle 2 \rangle}^{(1)}, \cdots, x_{\langle 43 \rangle}^{(1)}$$
$$x^{(2)}: x_{\langle 1 \rangle}^{(2)}, x_{\langle 2 \rangle}^{(2)}, \cdots, x_{\langle 25 \rangle}^{(2)}$$
$$\vdots$$
$$x^{(n)}: x_{\langle 1 \rangle}^{(n)}, x_{\langle 2 \rangle}^{(n)}, \cdots, x_{\langle 61 \rangle}^{(n)}$$

Vector de (secuencias de) objetivos

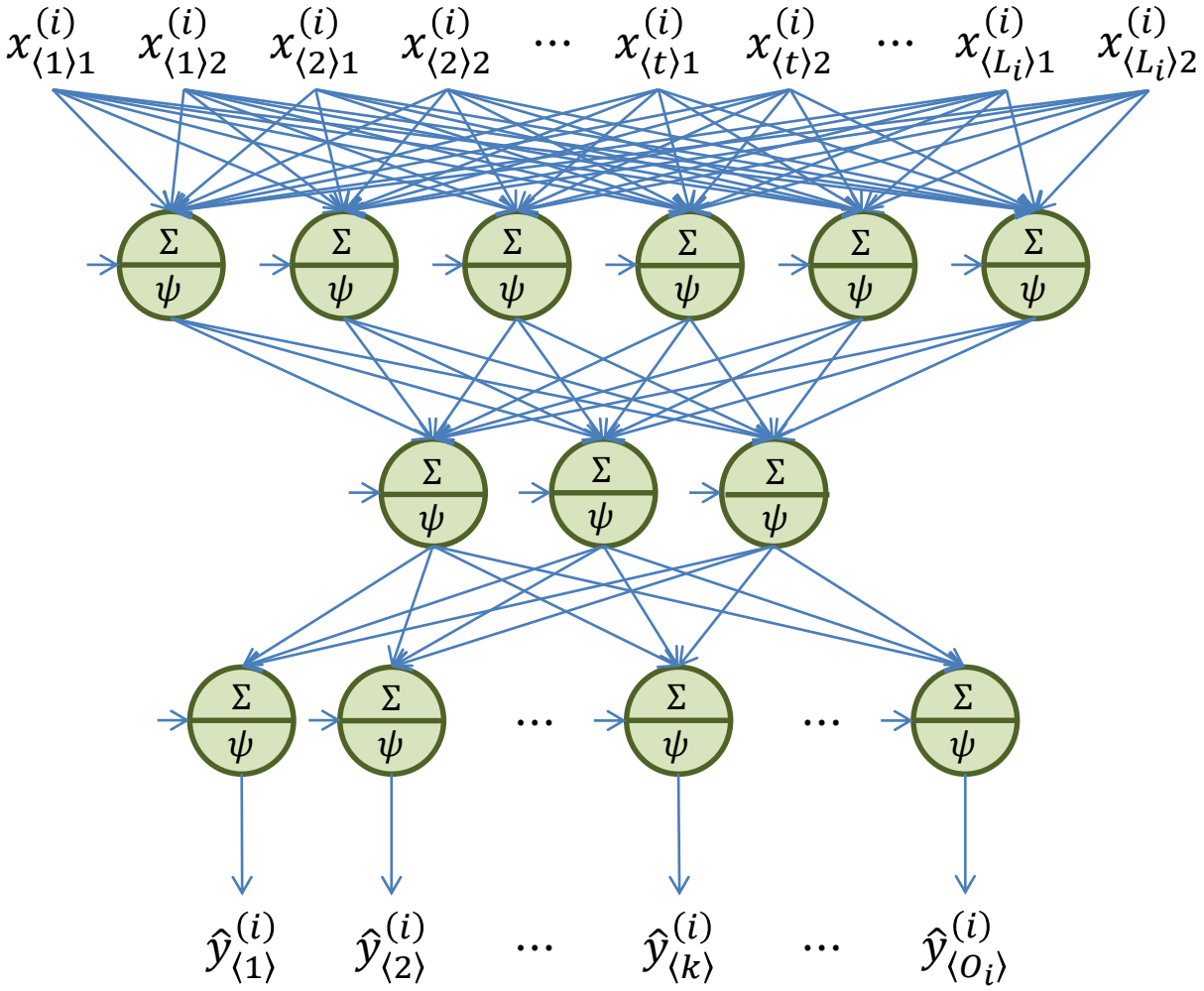
$$y^{(1)}: y_{\langle 1 \rangle}^{(1)}, y_{\langle 2 \rangle}^{(1)}, \cdots, y_{\langle 12 \rangle}^{(1)}$$
$$y^{(2)}: y_{\langle 1 \rangle}^{(2)}, y_{\langle 2 \rangle}^{(2)}, \cdots, y_{\langle 7 \rangle}^{(2)}$$
$$\vdots$$
$$y^{(n)}: y_{\langle 1 \rangle}^{(n)}, y_{\langle 2 \rangle}^{(n)}, \cdots, y_{\langle 23 \rangle}^{(n)}$$

$$x_{\langle t \rangle}^{(i)} = \left(x_{\langle t \rangle 1}^{(i)}, x_{\langle t \rangle 2}^{(i)}, \cdots, x_{\langle t \rangle d}^{(i)} \right)$$

$y_{\langle k \rangle}^{(i)}$: tipo de actuación

Recurrent Neural Networks

Fully Connected Networks para secuencias



- Razones por las que las redes completamente conectadas no son adecuadas para procesar secuencias
  - Cada secuencia de entrada (y de salida) puede tener diferente longitud
  - No comparten parámetros aprendidos a lo largo de diferentes partes de las secuencias

Recurrent Neural Networks
Datos secuenciales (ejemplos)

- En una cierta empresa eléctrica se dispone de los consumos diarios de una ciudad.
- Cada uno de los consumos viene caracterizado por una serie valores tales como potencia media, pico, valle, …
- A partir del consumo se pretende deducir la temperatura media en la ciudad durante ese día.
- Dado que los consumos son diferentes para cada día de la semana, el predictor de temperatura deberá tener en cuenta, no sólo el consumo de ese día, sino también la secuencia de valores de consumo anteriores.
- La longitud de las secuencias de entrada (consumos) y salida (temperatura) son iguales.

Recurrent Neural Networks

Datos secuenciales (ejemplos)

Secuencia de consumos de la i-ésima ciudad $x^{(i)}: x^{(i)}_{\langle 1\rangle}, x^{(i)}_{\langle 2\rangle}, \cdots, x^{(i)}_{\langle T\rangle}$

Consumo del t-ésimo día de la i-ésima ciudad $x^{(i)}_{\langle t\rangle} = \left(x^{(i)}_{\langle t\rangle 1}, x^{(i)}_{\langle t\rangle 2}, \cdots, x^{(i)}_{\langle t\rangle d}\right)$

j-ésima característica del consumo del t-ésimo día de la i-ésima ciudad $x^{(i)}_{\langle t\rangle j}$

Secuencia de temperaturas de la i-ésima ciudad $y^{(i)}: y^{(i)}_{\langle 1\rangle}, y^{(i)}_{\langle 2\rangle}, \cdots, y^{(i)}_{\langle T\rangle}$

Temperatura del t-ésimo día de la i-ésima ciudad $y^{(i)}_{\langle t\rangle}$

$$x^{(i)}: x^{(i)}_{\langle 1\rangle}, x^{(i)}_{\langle 2\rangle}, \cdots, x^{(i)}_{\langle T\rangle} \rightarrow y^{(i)}: y^{(i)}_{\langle 1\rangle}, y^{(i)}_{\langle 2\rangle}, \cdots, y^{(i)}_{\langle T\rangle}$$

Recurrent Neural Networks

Datos secuenciales (ejemplos)

Secuencia de palabras de la i-ésima frase $\quad x^{(i)}: x^{(i)}_{\langle 1\rangle}, x^{(i)}_{\langle 2\rangle}, \cdots, x^{(i)}_{\langle T\rangle}$

t-ésima palabra de la i-ésima frase $\quad x^{(i)}_{\langle t\rangle} = \left(x^{(i)}_{\langle t\rangle 1}, x^{(i)}_{\langle t\rangle 2}, \cdots, x^{(i)}_{\langle t\rangle d}\right)$

j-ésima característica de la t-ésima palabra de la i-ésima frase $\quad x^{(i)}_{\langle t\rangle j}$

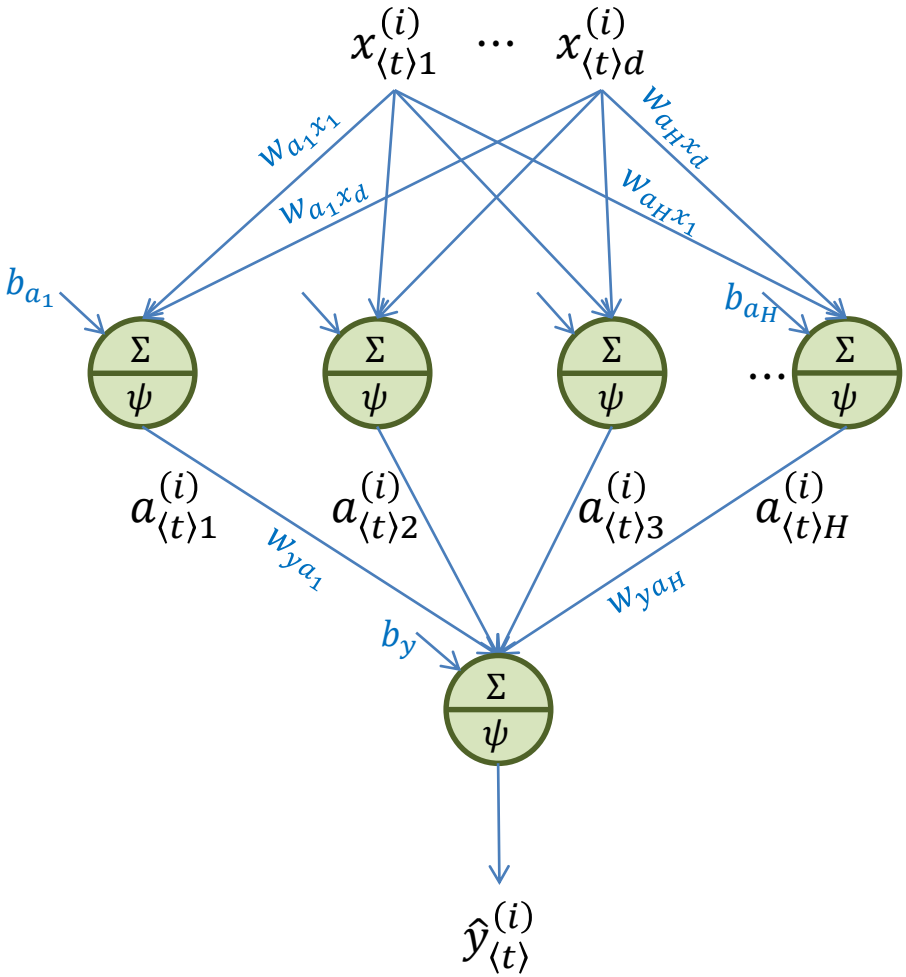
Secuencia de indicadores de nombre de la i-ésima frase $\quad y^{(i)}: y^{(i)}_{\langle 1\rangle}, y^{(i)}_{\langle 2\rangle}, \cdots, y^{(i)}_{\langle T\rangle}$

Indicador de nombre de la t-ésima palabra de la i-ésima frase $\quad y^{(i)}_{\langle t\rangle}$

$$x^{(i)}: x^{(i)}_{\langle 1\rangle}, x^{(i)}_{\langle 2\rangle}, \cdots, x^{(i)}_{\langle T\rangle} \rightarrow y^{(i)}: y^{(i)}_{\langle 1\rangle}, y^{(i)}_{\langle 2\rangle}, \cdots, y^{(i)}_{\langle T\rangle}$$

Recurrent Neural Networks

Fully Connected Networks para secuencias

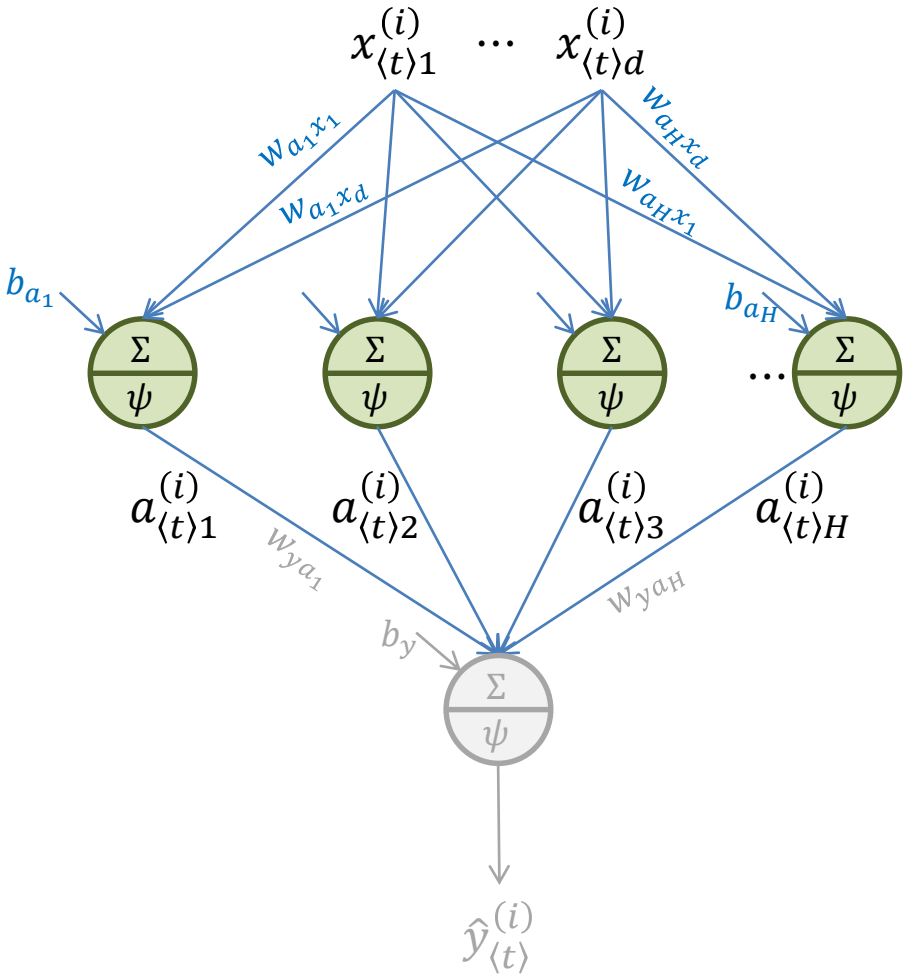


Deep Learning

- Conceptos generales
- Redes convolucionales
- Redes recurrentes
  - Conceptos generales
    - Datos secuenciales
    - FCN para secuencias
  - Representación simplificada
  - Natural Language Processing (NLP)
    - One-hot encoding
  - Tipos de RNN
  - Backpropagation en el tiempo
    - Gated Recurrent Unit (GRU)
    - Long Short Term Memory (LSTM)

Recurrent Neural Networks

Representación simplificada



$$a^{(i)}_{\langle t\rangle k} = \psi\left(b_{a_k} + \sum_{j=1}^{d} w_{a_k x_j} x^{(i)}_{\langle t\rangle j}\right)$$

$$x^{(i)}_{\langle t\rangle} = \begin{bmatrix} x^{(i)}_{\langle t\rangle 1} & \cdots & x^{(i)}_{\langle t\rangle d} \end{bmatrix}$$

$$w_{a_k x} = \begin{bmatrix} w_{a_k x_1} & \cdots & w_{a_k x_d} \end{bmatrix}$$

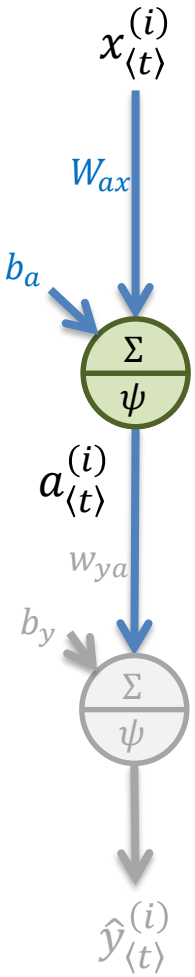
$$a^{(i)}_{\langle t\rangle k} = \psi\left(x^{(i)}_{\langle t\rangle} w^T_{a_k x} + b_{a_k}\right)$$

$$W_{ax} = \begin{bmatrix} w_{a_1 x_1} & \cdots & w_{a_1 x_d} \\ \vdots & \ddots & \vdots \\ w_{a_H x_1} & \cdots & w_{a_H x_d} \end{bmatrix}$$

$$b_a = \begin{bmatrix} b_{a_1} & \cdots & b_{a_H} \end{bmatrix}$$

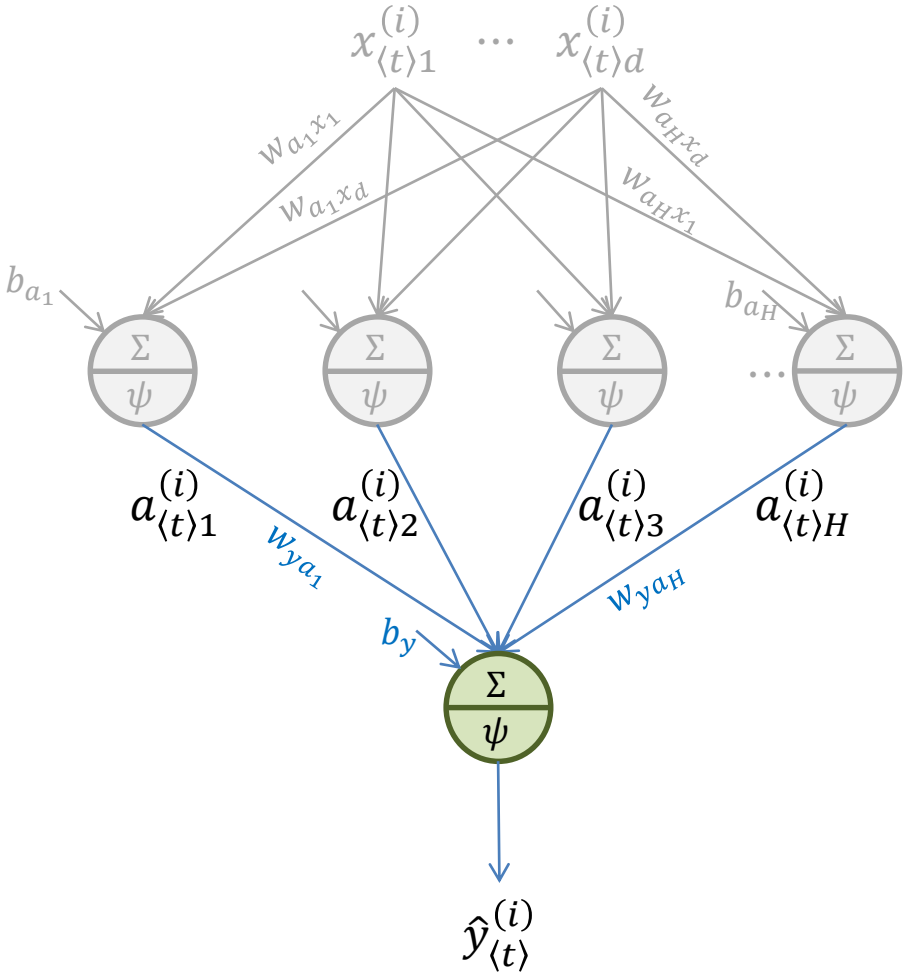
$$a^{(i)}_{\langle t\rangle} = \begin{bmatrix} a^{(i)}_{\langle t\rangle 1} & \cdots & a^{(i)}_{\langle t\rangle H} \end{bmatrix}$$

$$\boxed{a^{(i)}_{\langle t\rangle} = \psi\left(x^{(i)}_{\langle t\rangle} W^T_{ax} + b_a\right)}$$



Recurrent Neural Networks

Representación simplificada

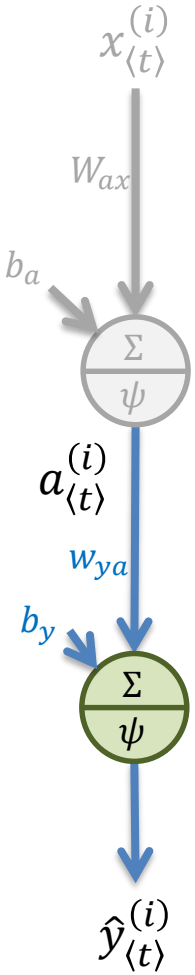


$$\hat{y}^{(i)}_{\langle t\rangle} = \psi\left(b_y + \sum_{k=1}^{H} w_{ya_k} a^{(i)}_{\langle t\rangle k}\right)$$

$$a^{(i)}_{\langle t\rangle} = \left[a^{(i)}_{\langle t\rangle 1} \quad \cdots \quad a^{(i)}_{\langle t\rangle H}\right]$$

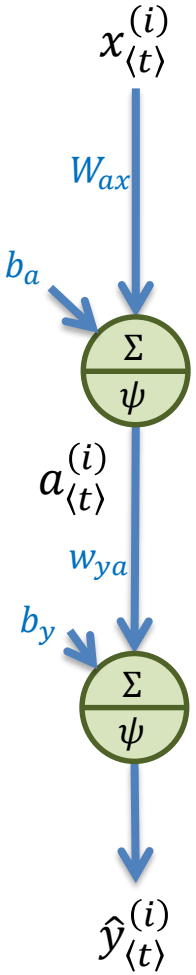
$$w_{ya} = [w_{ya_1} \quad \cdots \quad w_{ya_H}]$$

$$\boxed{y^{(i)}_{\langle t\rangle} = \psi\left(a^{(i)}_{\langle t\rangle} w^T_{ya} + b_y\right)}$$



Recurrent Neural Networks

Representación simplificada

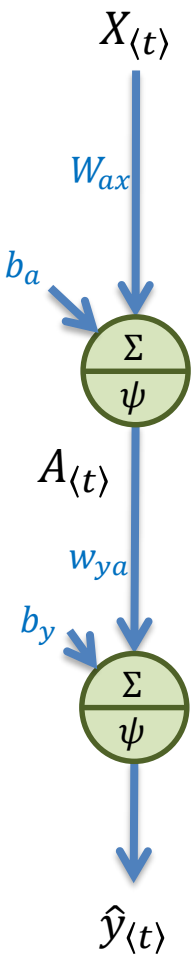


$$a_{\langle t \rangle}^{(i)} = \psi\left(x_{\langle t \rangle}^{(i)} W_{ax}^T + b_a\right) \qquad y_{\langle t \rangle}^{(i)} = \psi\left(a_{\langle t \rangle}^{(i)} w_{ya}^T + b_y\right)$$

$$X_{\langle t \rangle} = \begin{bmatrix} x_{\langle t \rangle}^{(1)} \\ x_{\langle t \rangle}^{(2)} \\ \vdots \\ x_{\langle t \rangle}^{(n)} \end{bmatrix} = \begin{bmatrix} x_{\langle t \rangle 1}^{(1)} & x_{\langle t \rangle 2}^{(1)} & \cdots & x_{\langle t \rangle d}^{(1)} \\ x_{\langle t \rangle 1}^{(2)} & x_{\langle t \rangle 2}^{(2)} & \cdots & x_{\langle t \rangle d}^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ x_{\langle t \rangle 1}^{(n)} & x_{\langle t \rangle 2}^{(n)} & \cdots & x_{\langle t \rangle d}^{(n)} \end{bmatrix} \qquad \hat{y}_{\langle t \rangle} = \begin{bmatrix} y_{\langle t \rangle}^{(1)} \\ y_{\langle t \rangle}^{(2)} \\ \vdots \\ y_{\langle t \rangle}^{(n)} \end{bmatrix}$$

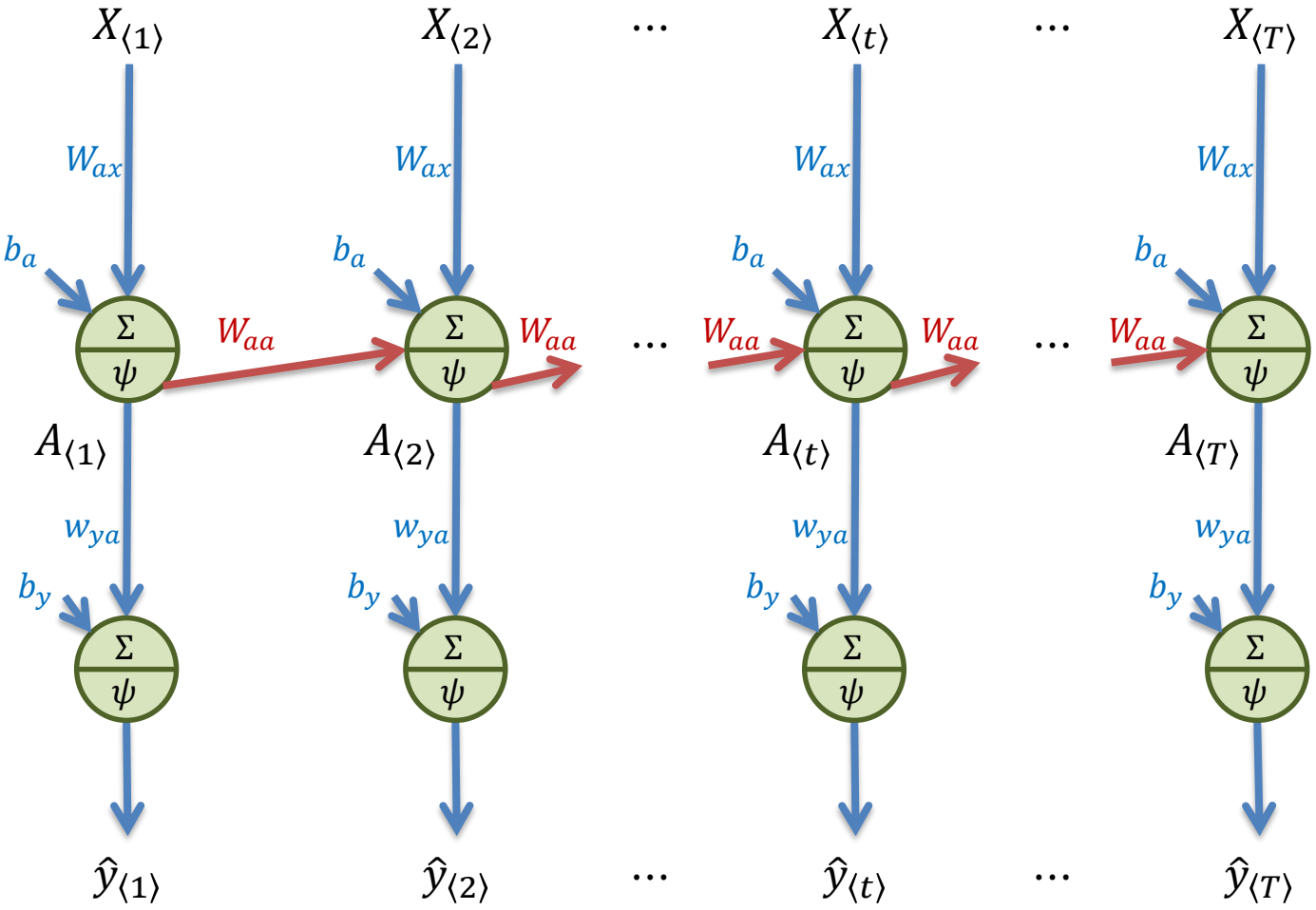
$$A_{\langle t \rangle} = \begin{bmatrix} a_{\langle t \rangle}^{(1)} \\ a_{\langle t \rangle}^{(2)} \\ \vdots \\ a_{\langle t \rangle}^{(n)} \end{bmatrix} = \begin{bmatrix} a_{\langle t \rangle 1}^{(1)} & a_{\langle t \rangle 2}^{(1)} & \cdots & a_{\langle t \rangle H}^{(1)} \\ a_{\langle t \rangle 1}^{(2)} & a_{\langle t \rangle 2}^{(2)} & \cdots & a_{\langle t \rangle H}^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ a_{\langle t \rangle 1}^{(n)} & a_{\langle t \rangle 2}^{(n)} & \cdots & a_{\langle t \rangle H}^{(n)} \end{bmatrix}$$

$$A_{\langle t \rangle} = \psi\left(X_{\langle t \rangle} W_{ax}^T + b_a\right) \qquad \hat{y}_{\langle t \rangle} = \psi\left(A_{\langle t \rangle} w_{ya}^T + b_y\right)$$



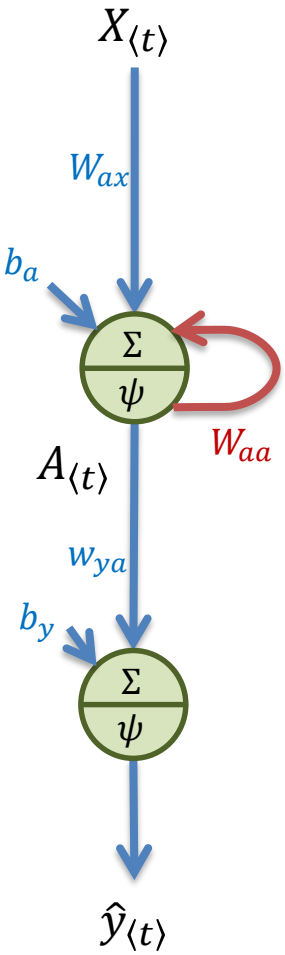
Recurrent Neural Networks

Representación simplificada



Recurrent Neural Networks

Representación simplificada



$$a_{\langle t \rangle k}^{(i)} = \psi\left(b_{a_k} + \sum_{j=1}^{d} w_{a_k x_j} x_{\langle t \rangle j}^{(i)} + \sum_{u=1}^{d} w_{a_k a_u} a_{\langle t-1 \rangle u}^{(i)}\right)$$

$$w_{a_k a} = \begin{bmatrix} w_{a_k a_1} & \cdots & w_{a_k a_H} \end{bmatrix}$$

$$a_{\langle t \rangle k}^{(i)} = \psi\left(x_{\langle t \rangle}^{(i)} w_{a_k x}^T + a_{\langle t-1 \rangle}^{(i)} w_{a_k a}^T + b_{a_k}\right)$$

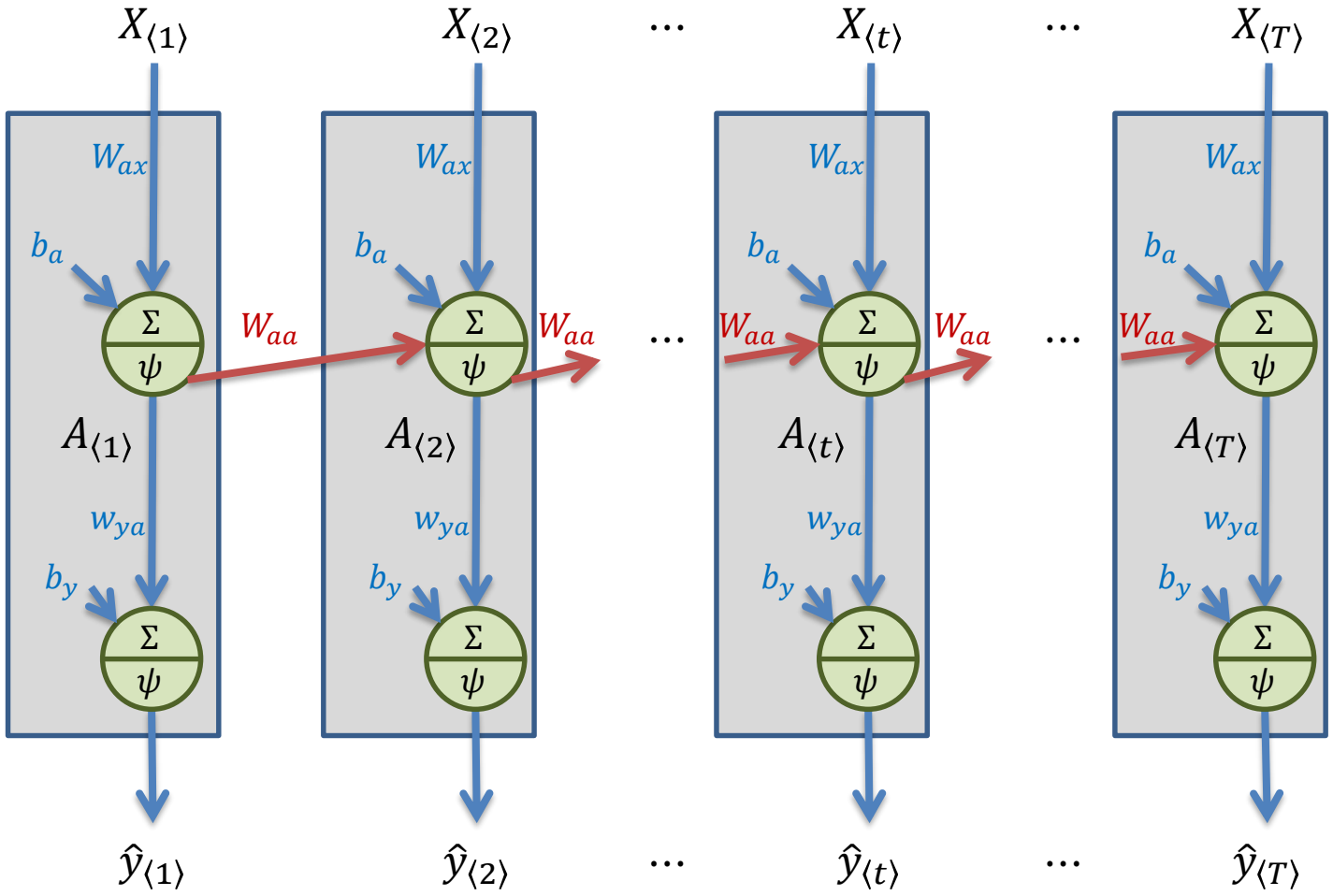
$$W_{aa} = \begin{bmatrix} w_{a_1 a_1} & \cdots & w_{a_1 a_H} \\ \vdots & \ddots & \vdots \\ w_{a_H a_1} & \cdots & w_{a_H a_H} \end{bmatrix}$$

$$a_{\langle t \rangle}^{(i)} = \psi\left(x_{\langle t \rangle}^{(i)} W_{ax}^T + a_{\langle t-1 \rangle}^{(i)} W_{aa}^T + b_a\right)$$

$$\boxed{A_{\langle t \rangle} = \psi\left(X_{\langle t \rangle} W_{ax}^T + A_{\langle t-1 \rangle} W_{aa}^T + b_a\right)}$$

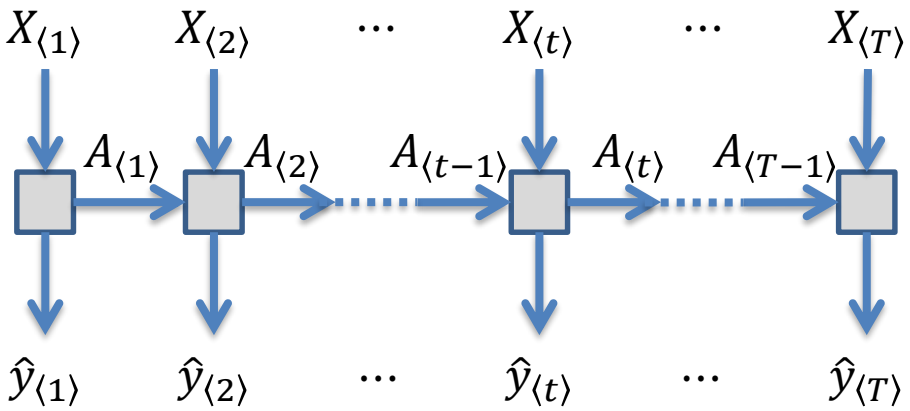
Recurrent Neural Networks

Representación simplificada



Recurrent Neural Networks

Representación simplificada

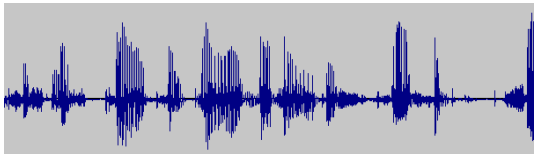


Deep Learning

- Conceptos generales
- Redes convolucionales
- Redes recurrentes
  - Conceptos generales
    - Datos secuenciales
    - FCN para secuencias
  - Representación simplificada
  - Natural Language Processing (NLP)
    - One-hot encoding
  - Tipos de RNN
  - Backpropagation en el tiempo
    - Gated Recurrent Unit (GRU)
    - Long Short Term Memory (LSTM)

Recurrent Neural Networks

Natural Language Processing (NLP)

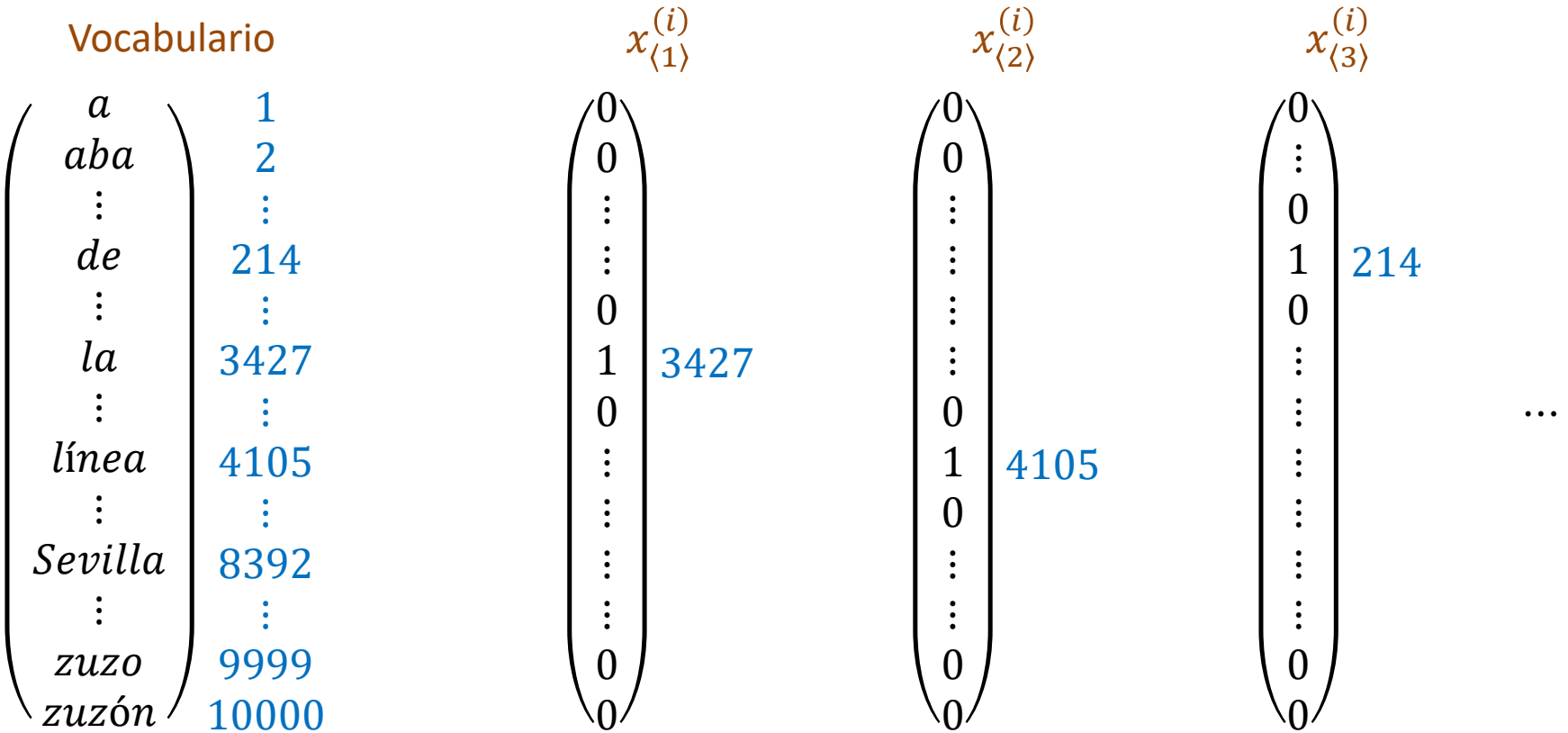
Identificación de nombres propios	La línea de Sevilla a Málaga está sobrecargada	→ La línea de Sevilla a Málaga está sobrecargada
Reconocimiento de voz		→ Tengo una avería
Clasificación de sentimientos	El servicio de atención al cliente es horrible	→ ★☆☆☆☆
Traducción automática	El servicio de atención al cliente es horrible	→ Customer service is horrible.
Generación de textos	∅	→ Estimado cliente, es un placer dirigirme a usted para ….

Recurrent Neural Networks

NLP: One-hot coding

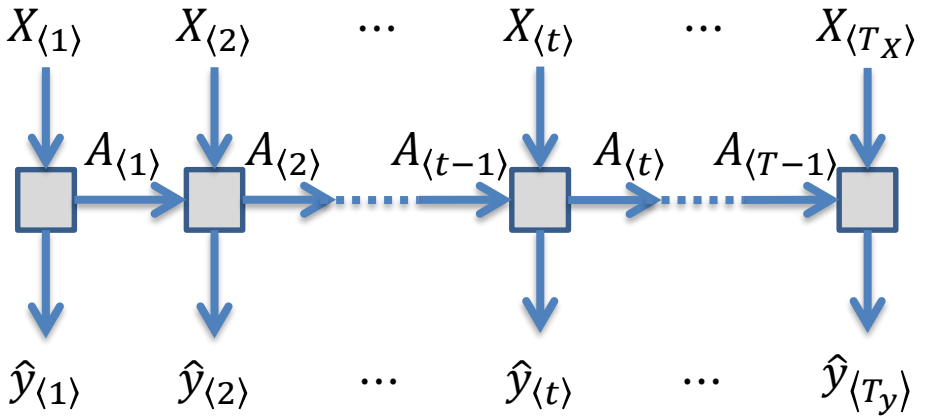
$x^{(i)}$: La línea de Sevilla a Málaga está sobrecargada

$x^{(i)}_{\langle 1\rangle}$ $x^{(i)}_{\langle 2\rangle}$ $x^{(i)}_{\langle 3\rangle}$ $x^{(i)}_{\langle 4\rangle}$ $x^{(i)}_{\langle 5\rangle}$ $x^{(i)}_{\langle 6\rangle}$ $x^{(i)}_{\langle 7\rangle}$ $x^{(i)}_{\langle 8\rangle}$



Recurrent Neural Networks

Tipos de RNN

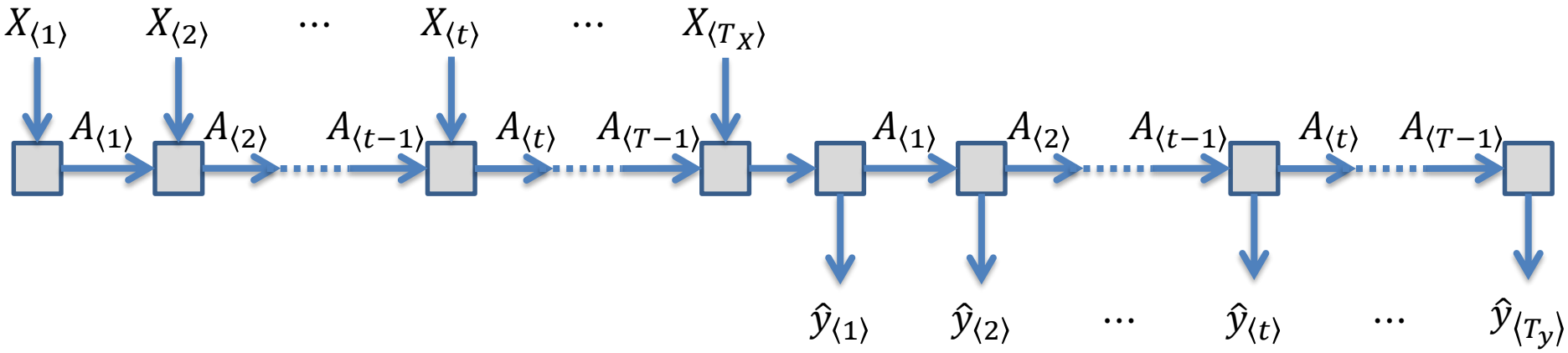


Muchos (elementos) a muchos (objetivos): $T_x = T_y$

Ej.: Identificación de nombres propios

Recurrent Neural Networks

Tipos de RNN

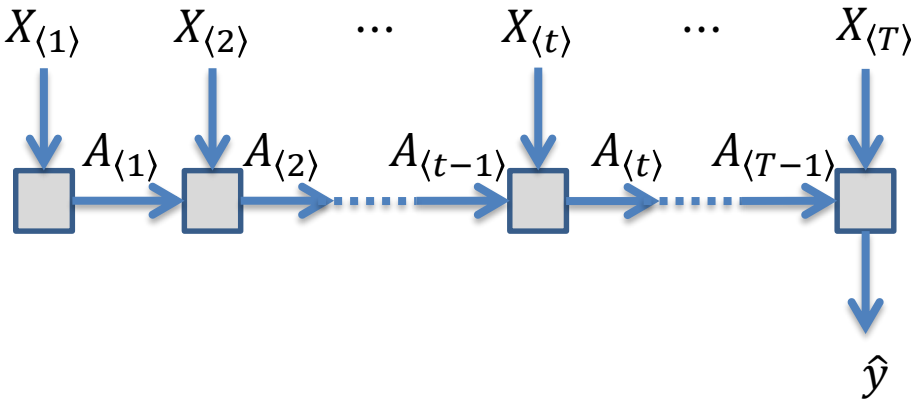


Muchos a muchos: $T_x \neq T_y$

Ej.: Traducción automática

Recurrent Neural Networks

Tipos de RNN

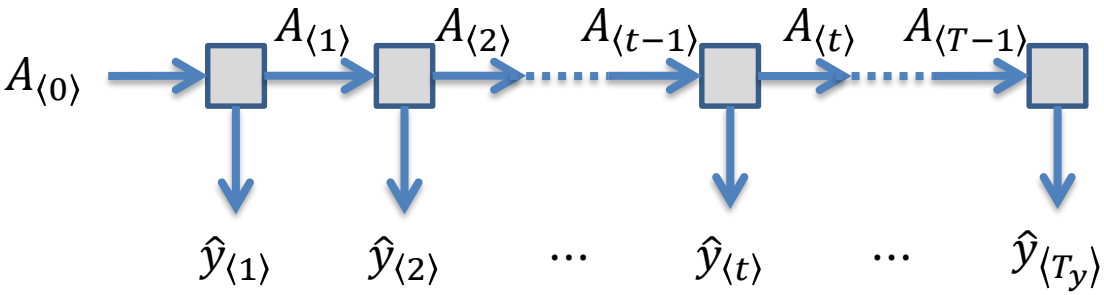


Muchos a uno

Ej.: Clasificación de sentimientos

Recurrent Neural Networks

Tipos de RNN



Uno (o ninguno) a muchos: $T_x = T_y$

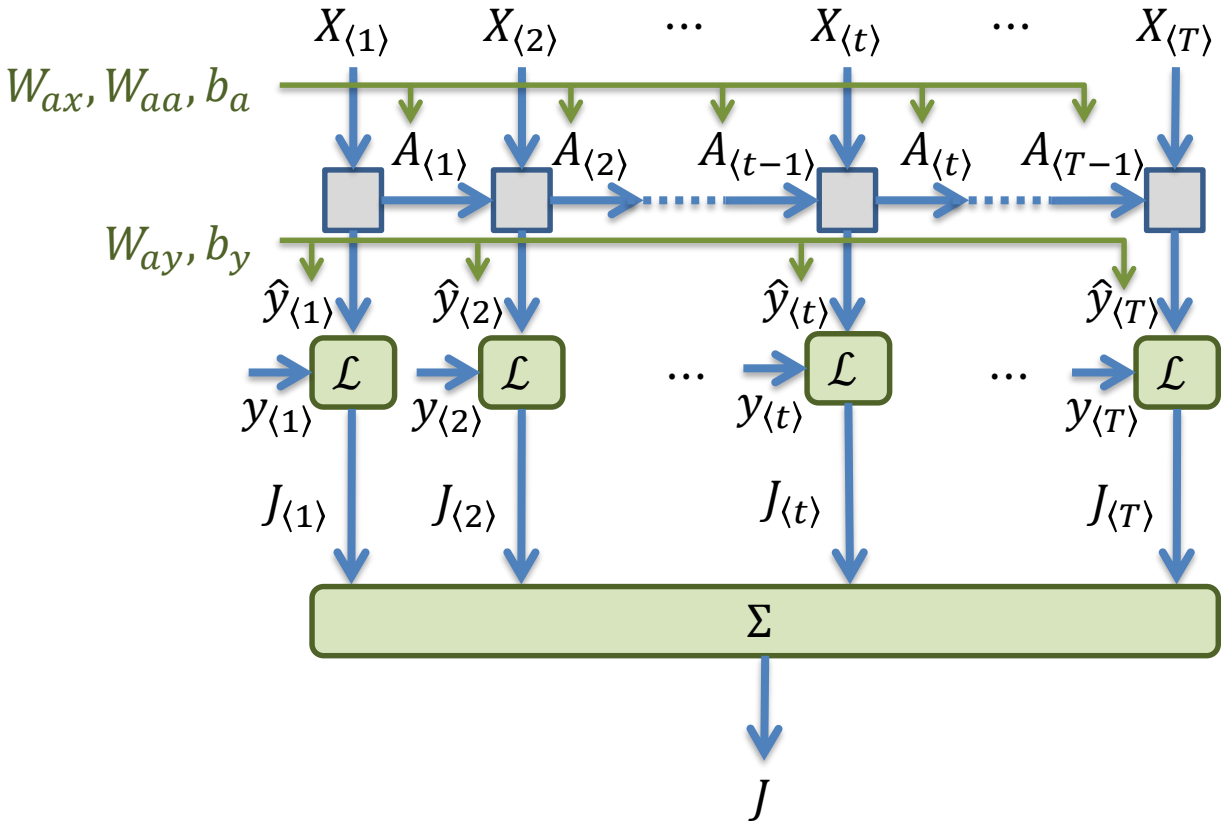
Ej.: Generación automática de textos (o música)

Deep Learning

- Conceptos generales
- Redes convolucionales
- Redes recurrentes
  - Conceptos generales
    - Datos secuenciales
    - FCN para secuencias
  - Representación simplificada
  - Natural Language Processing (NLP)
    - One-hot encoding
  - Tipos de RNN
  - Backpropagation en el tiempo
    - Gated Recurrent Unit (GRU)
    - Long Short Term Memory (LSTM)

Recurrent Neural Networks

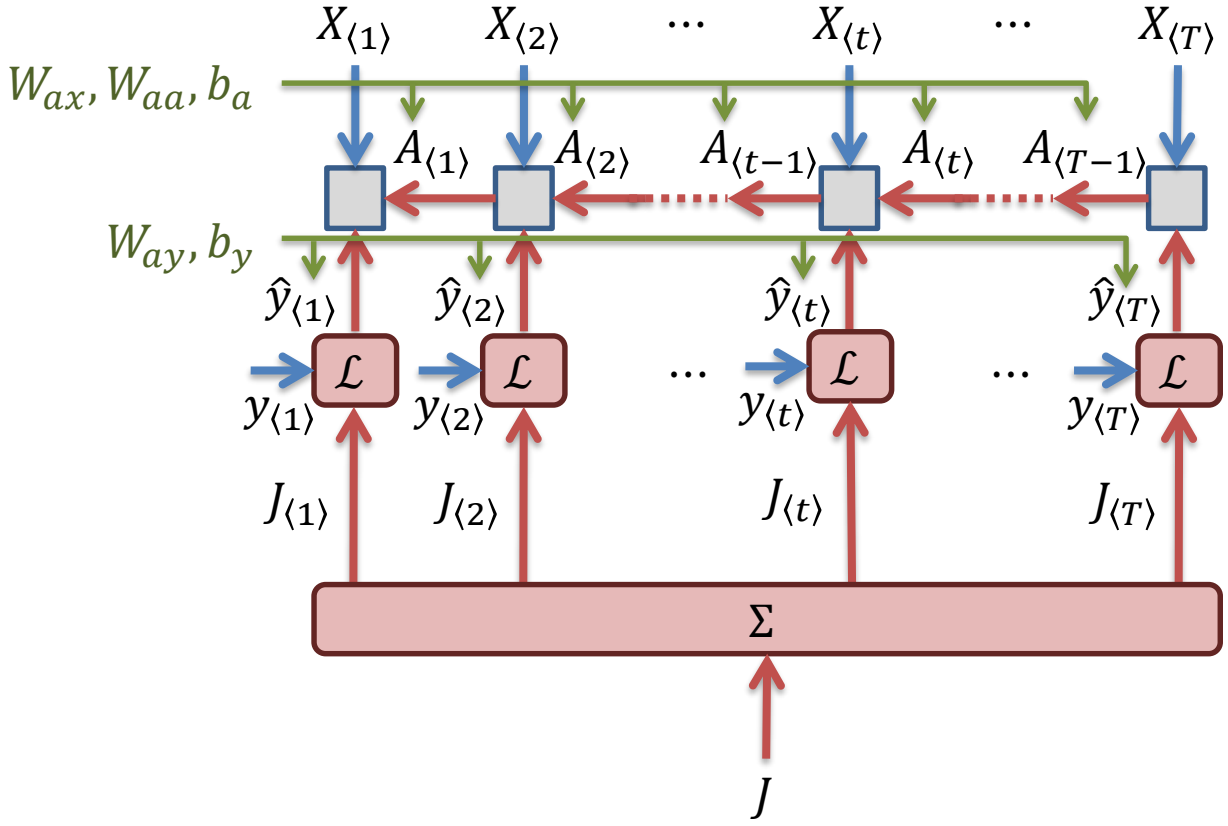
Backpropagation en el tiempo



$$J_{\langle t\rangle} = \mathcal{L}\left(\hat{y}_{\langle t\rangle}, y_{\langle t\rangle}\right) \qquad J = \sum_{t=1}^{T} J_{\langle t\rangle}$$

Recurrent Neural Networks

Backpropagation en el tiempo



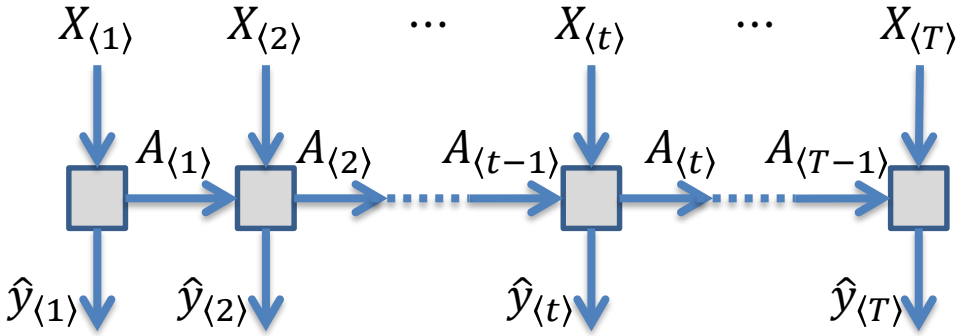
$$\frac{\partial J}{\partial W} \propto \prod_{t=1}^{T} \psi'^{[t]}$$

La "profundidad" de la red (número de neuronas que se "atraviesan" en el cálculo de la predicción o del gradiente), depende también del tiempo (de la longitud de la secuencia):

Potenciales problemas de desvanecimiento de gradiente

Recurrent Neural Networks

Backpropagation en el tiempo



$X_{\langle 1\rangle}$ $X_{\langle 2\rangle}$ $X_{\langle 3\rangle}$ $X_{\langle 27\rangle}$ $X_{\langle 30\rangle}$

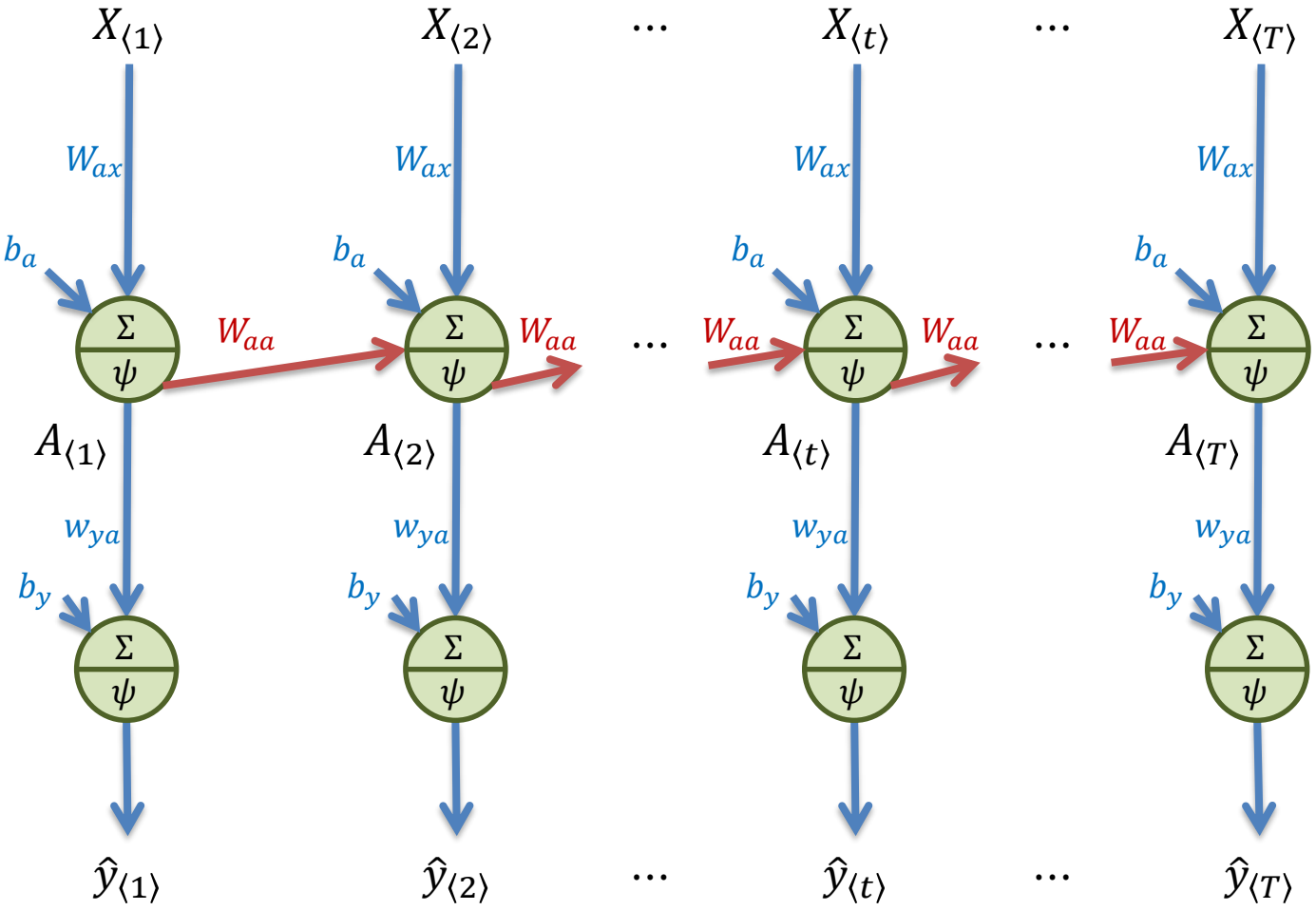
La(s) línea(s), considerando la temperatura y …., está(n) sobrecargada(s) un 40%

La influencia de un elemento de la secuencia sobre otros elementos muy alejados está limitada por el problema del desvanecimiento del gradiente.

Solución: arquitecturas de red mejoradas (GRU, LSTM, …)

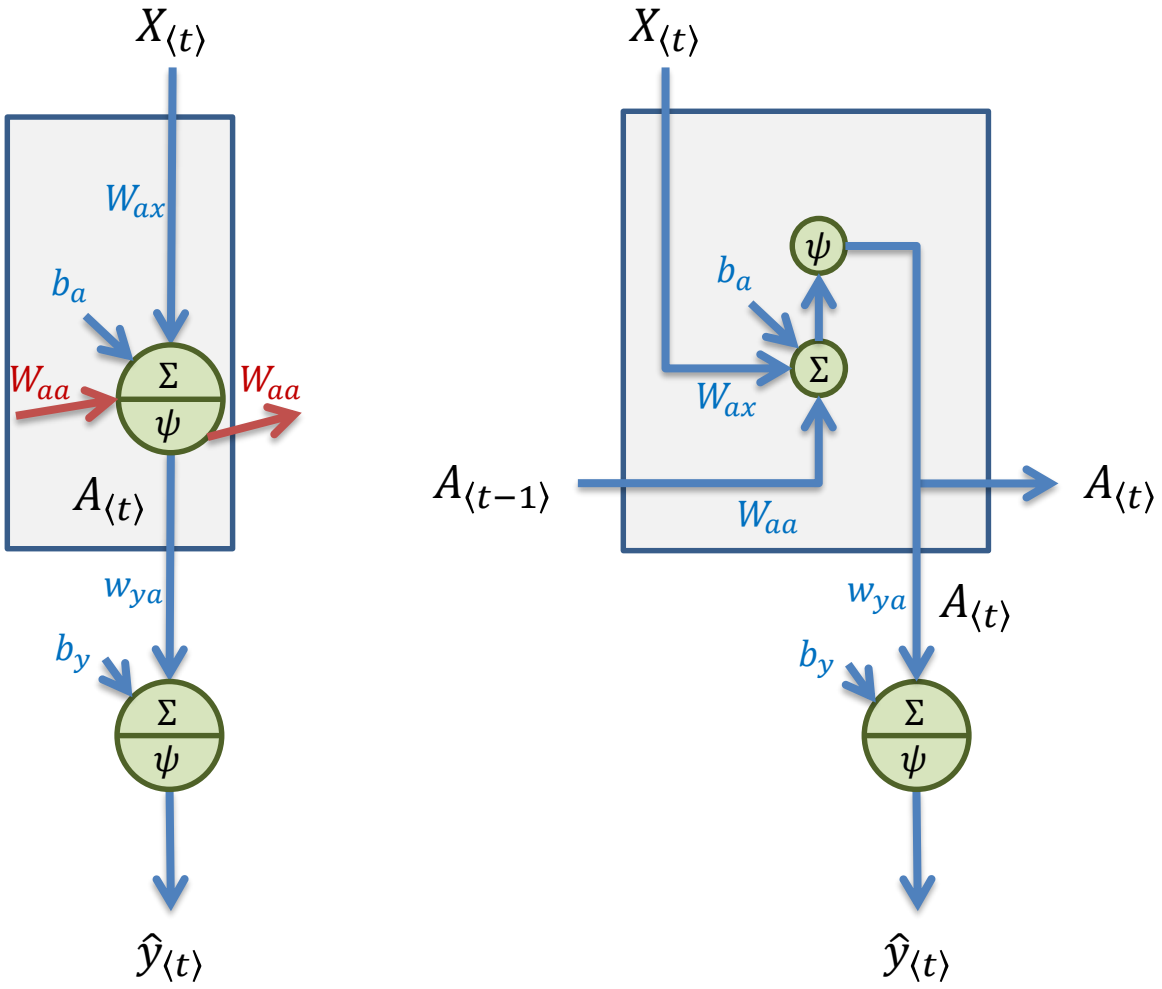
Recurrent Neural Networks

Gated Recurrent Unit (GRU)



Recurrent Neural Networks

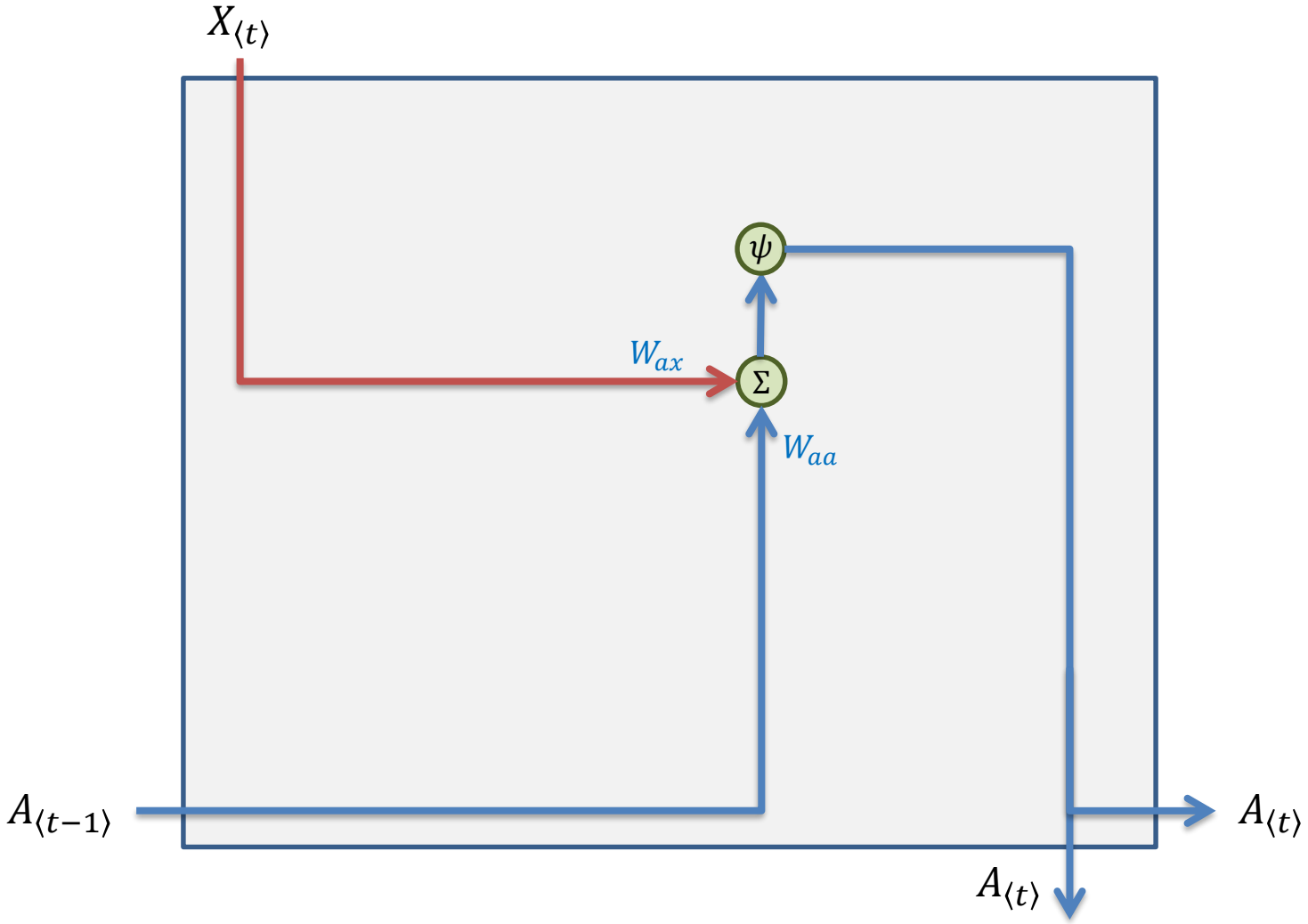
Gated Recurrent Unit (GRU)



$$A_{\langle t \rangle} = \psi\left(X_{\langle t \rangle} W_{ax}^T + A_{\langle t-1 \rangle} W_{aa}^T + b_a\right) \qquad \hat{y}_{\langle t \rangle} = \psi\left(A_{\langle t \rangle} w_{ya}^T + b_y\right)$$

Recurrent Neural Networks

Gated Recurrent Unit (GRU)



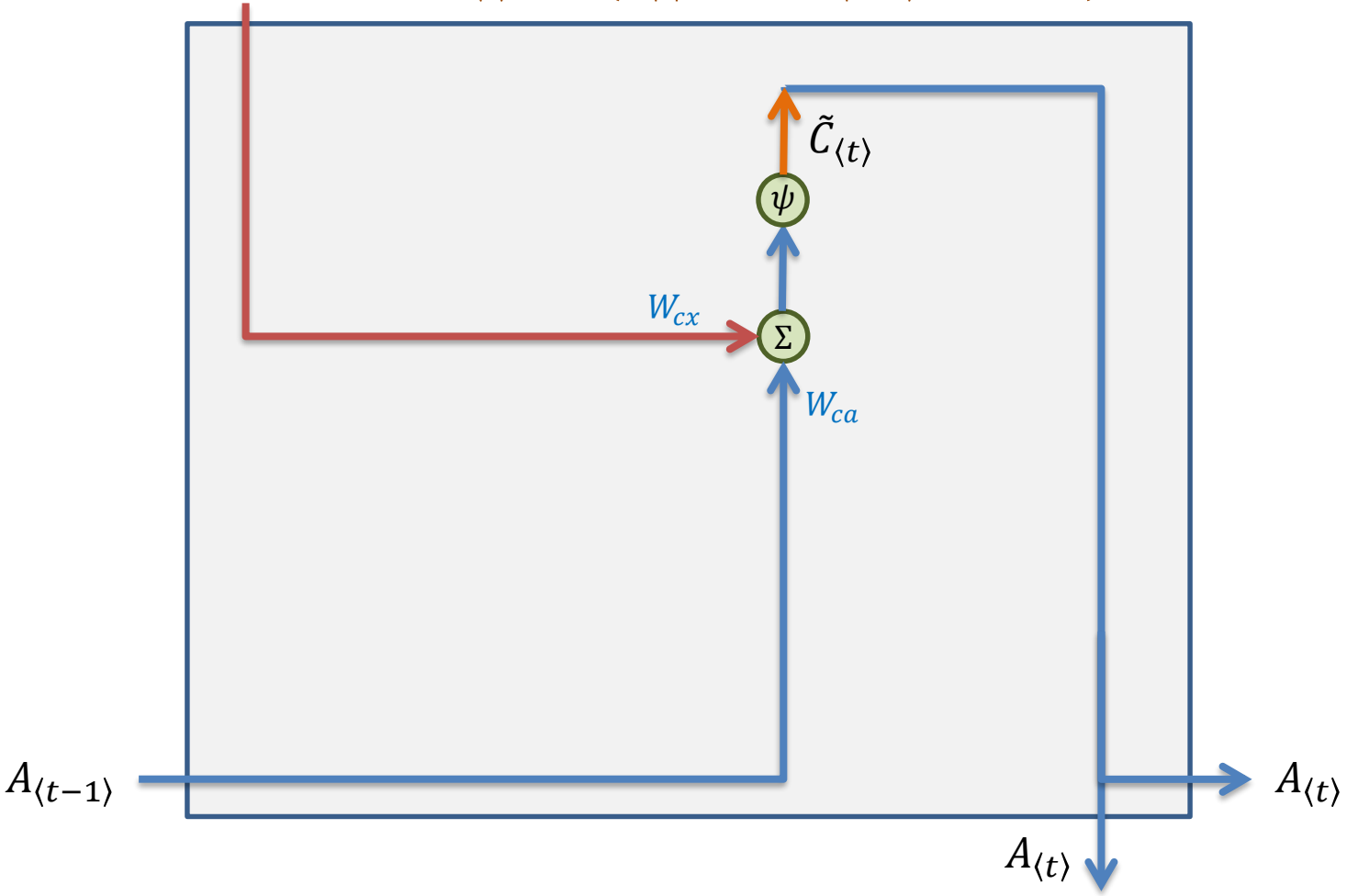
Cell output: $A_{\langle t \rangle} = \psi\left(X_{\langle t \rangle} W_{ax}^T + A_{\langle t-1 \rangle} W_{aa}^T + b_a\right)$

Recurrent Neural Networks

Gated Recurrent Unit (GRU)

$X_{\langle t \rangle}$

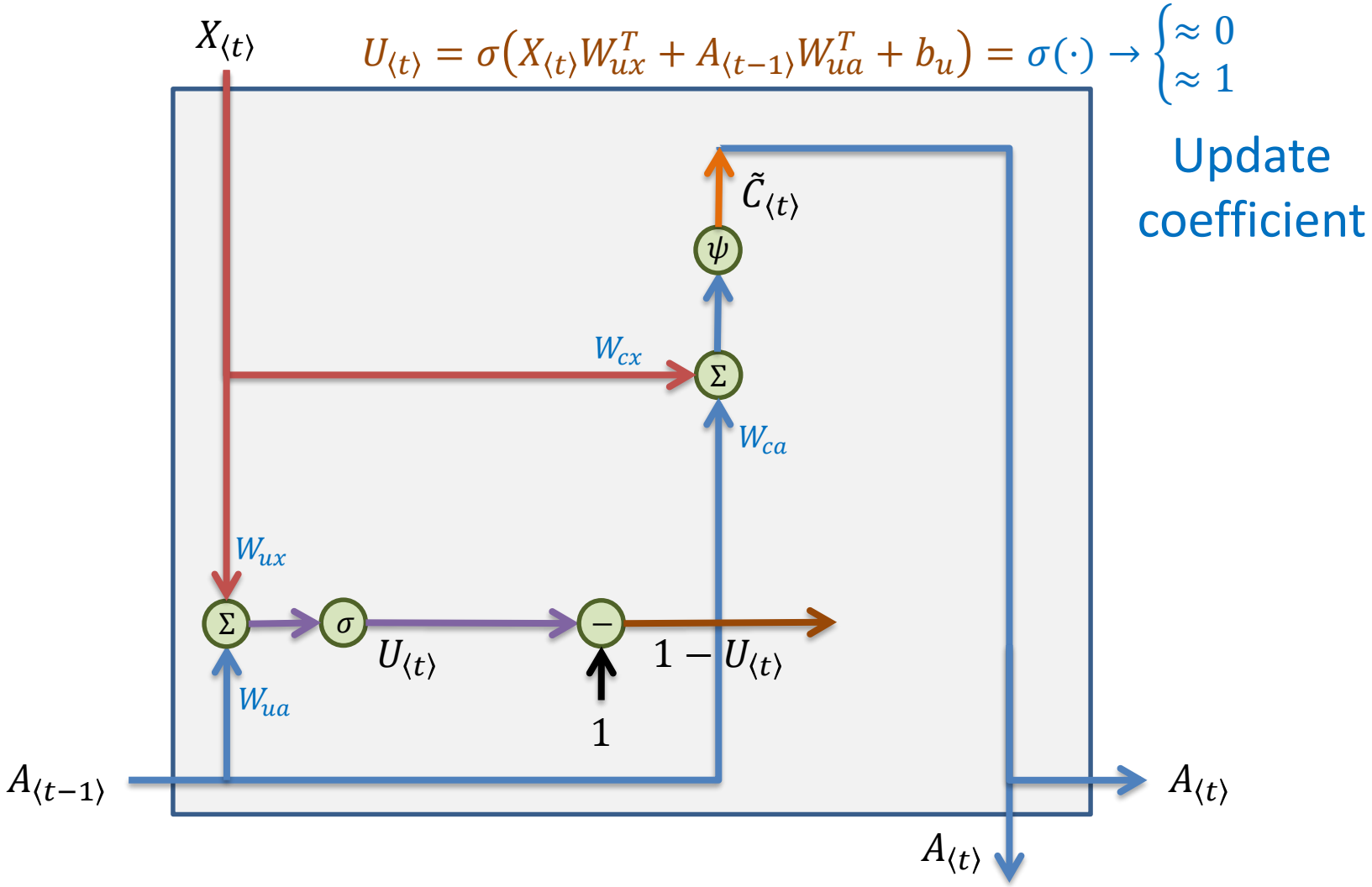
$\tilde{C}_{\langle t \rangle} = \psi\left(X_{\langle t \rangle} W_{cx}^T + A_{\langle t-1 \rangle} W_{ca}^T + b_c\right)$ Cell computation



$A_{\langle t \rangle} = \tilde{C}_{\langle t \rangle}$

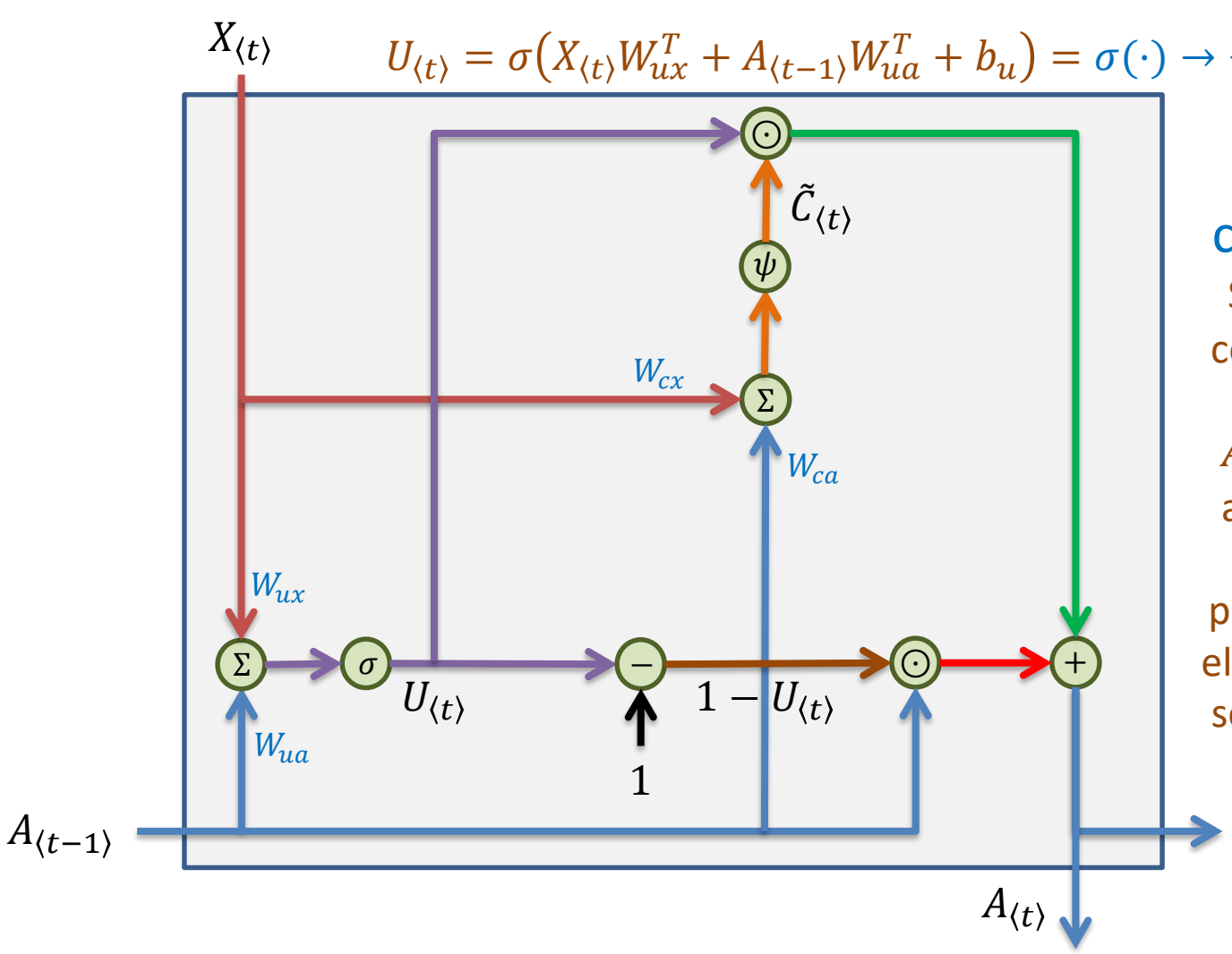
Recurrent Neural Networks

Gated Recurrent Unit (GRU)



Recurrent Neural Networks

Gated Recurrent Unit (GRU)



Update coefficient

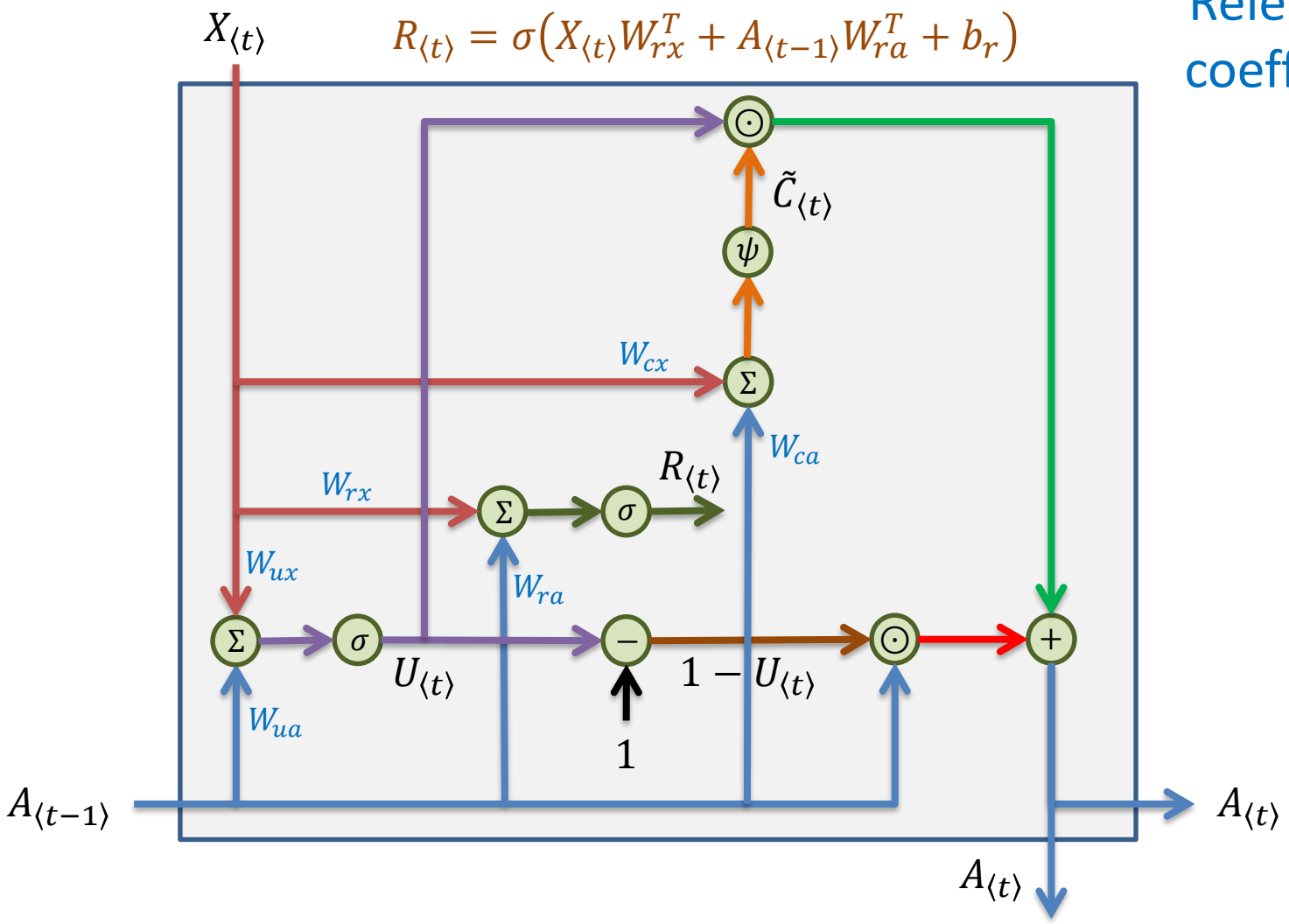
Si $U_{\langle t \rangle} \approx 0$ la celda "pasa" la información $A_{\langle t-1 \rangle}$ sin casi alterarla. Esta información puede alcanzar elementos de la secuencia muy alejados

Cell output: $A_{\langle t \rangle} = U_{\langle t \rangle} \odot \tilde{C}_{\langle t \rangle} + \left(1 - U_{\langle t \rangle}\right) \odot A_{\langle t-1 \rangle}$

Recurrent Neural Networks

Gated Recurrent Unit (GRU)

Relevance coefficient



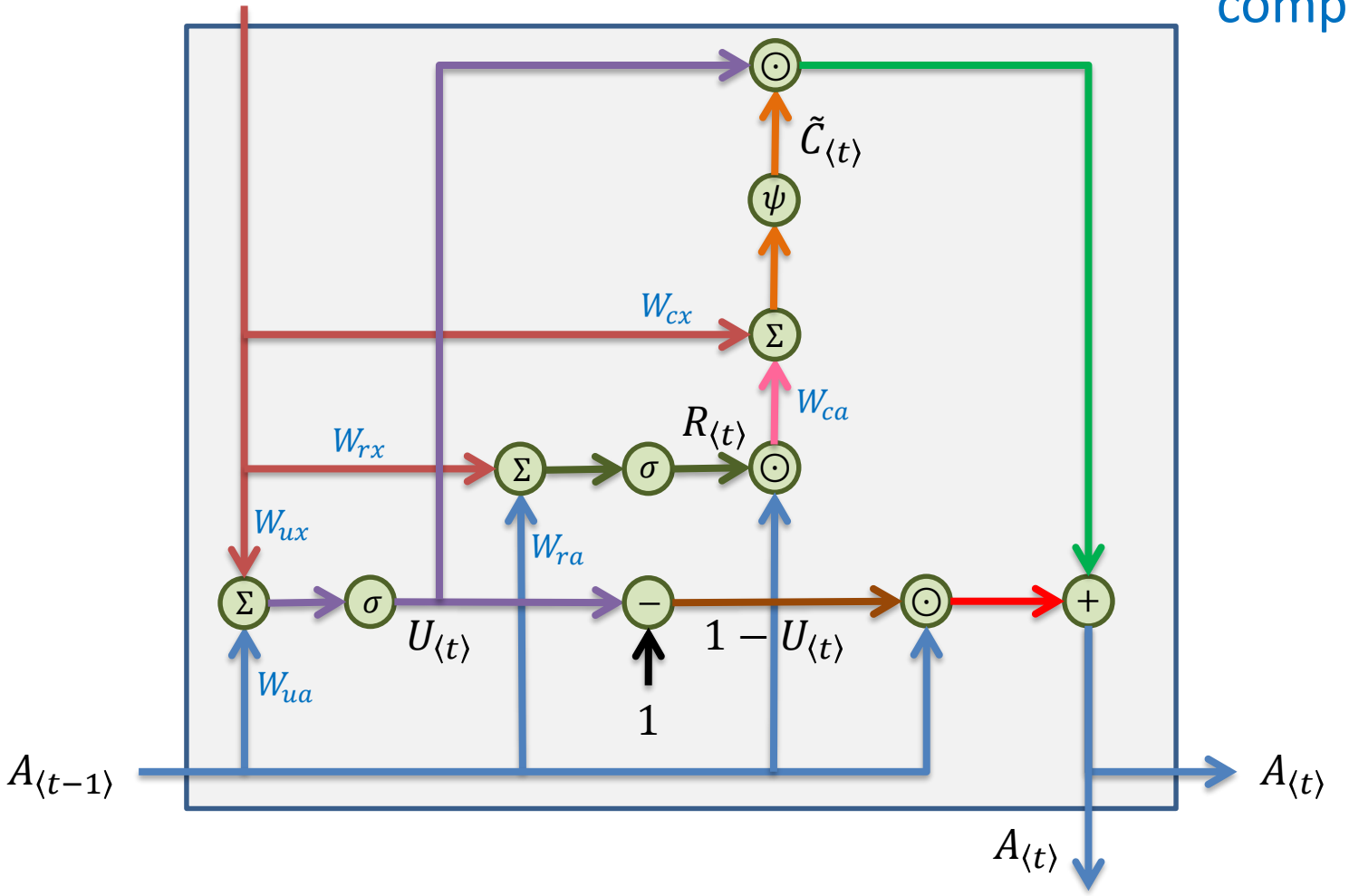
Cell output: $A_{\langle t \rangle} = U_{\langle t \rangle} \odot \tilde{C}_{\langle t \rangle} + \left(1 - U_{\langle t \rangle}\right) \odot A_{\langle t-1 \rangle}$

Recurrent Neural Networks

Gated Recurrent Unit (GRU)

Cell computation

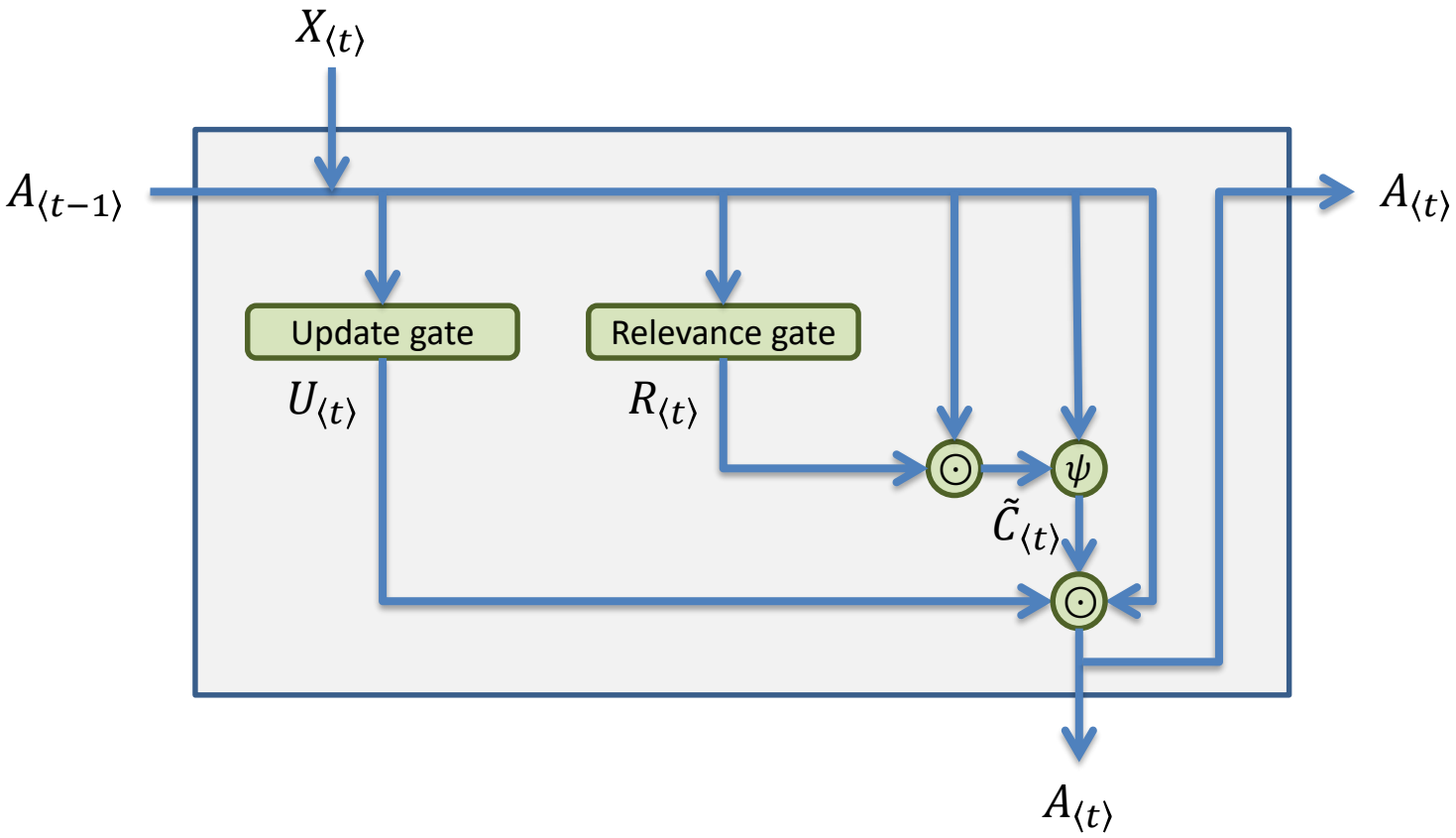
$$X_{\langle t \rangle} \quad \tilde{C}_{\langle t \rangle} = \psi\left(X_{\langle t \rangle} W_{cx}^T + \left(R_{\langle t \rangle} \odot A_{\langle t-1 \rangle}\right) W_{ca}^T + b_c\right)$$



Cell output: $A_{\langle t \rangle} = U_{\langle t \rangle} \odot \tilde{C}_{\langle t \rangle} + \left(1 - U_{\langle t \rangle}\right) \odot A_{\langle t-1 \rangle}$

Recurrent Neural Networks

Gated Recurrent Unit (GRU)



$$U_{\langle t \rangle} = \sigma\left(X_{\langle t \rangle} W_{ux}^T + A_{\langle t-1 \rangle} W_{ua}^T + b_u\right)$$

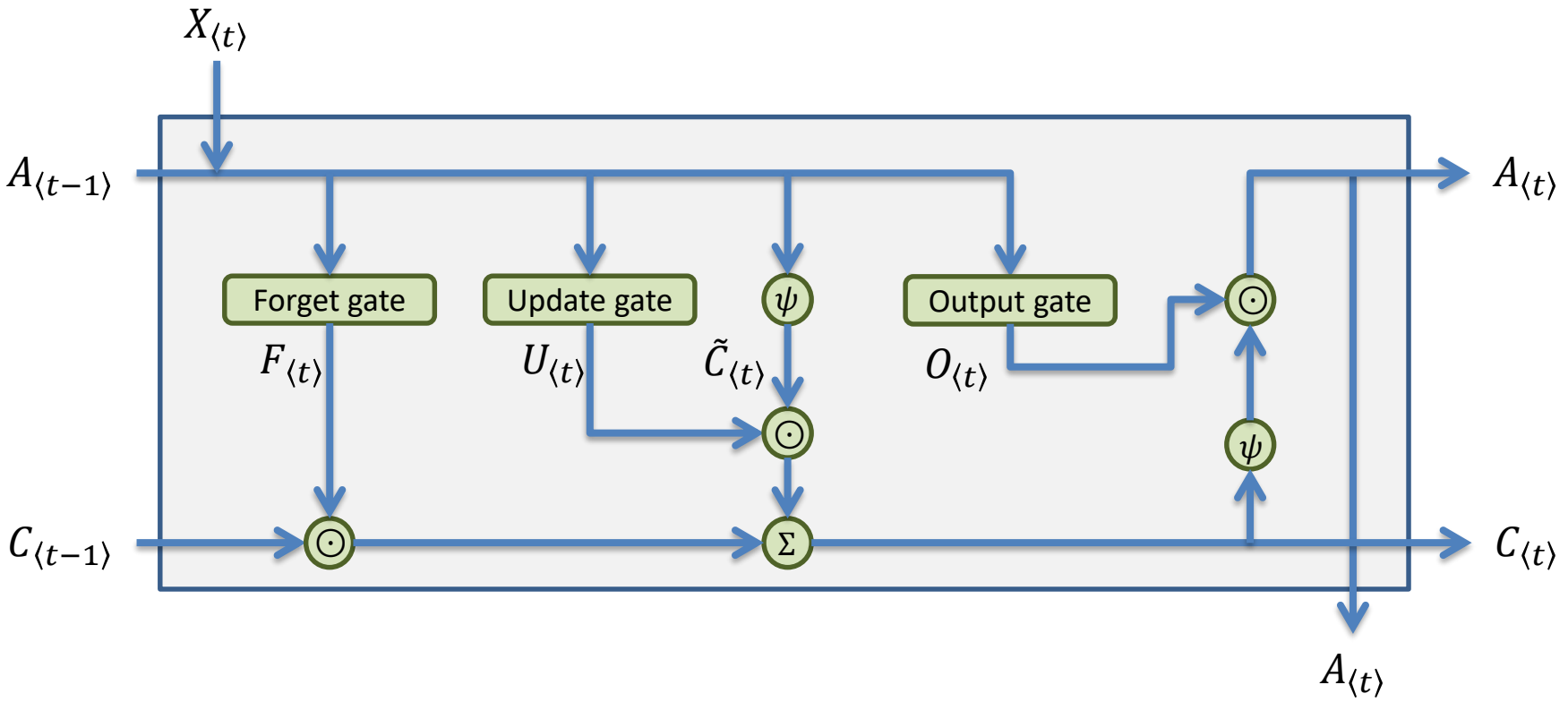
$$R_{\langle t \rangle} = \sigma\left(X_{\langle t \rangle} W_{rx}^T + A_{\langle t-1 \rangle} W_{ra}^T + b_r\right)$$

$$\tilde{C}_{\langle t \rangle} = \psi\left(X_{\langle t \rangle} W_{cx}^T + \left(R_{\langle t \rangle} \odot A_{\langle t-1 \rangle}\right) W_{ca}^T + b_c\right)$$

$$A_{\langle t \rangle} = U_{\langle t \rangle} \odot \tilde{C}_{\langle t \rangle} + \left(1 - U_{\langle t \rangle}\right) \odot A_{\langle t-1 \rangle}$$

Recurrent Neural Networks

Long Short Term Memory (LSTM)



$$F_{\langle t \rangle} = \sigma\left(X_{\langle t \rangle} W_{fx}^T + A_{\langle t-1 \rangle} W_{fa}^T + b_f\right)$$

$$U_{\langle t \rangle} = \sigma\left(X_{\langle t \rangle} W_{ux}^T + A_{\langle t-1 \rangle} W_{ua}^T + b_u\right)$$

$$O_{\langle t \rangle} = \sigma\left(X_{\langle t \rangle} W_{ox}^T + A_{\langle t-1 \rangle} W_{oa}^T + b_o\right)$$

$$\tilde{C}_{\langle t \rangle} = \psi\left(X_{\langle t \rangle} W_{cx}^T + A_{\langle t-1 \rangle} W_{ca}^T + b_c\right)$$

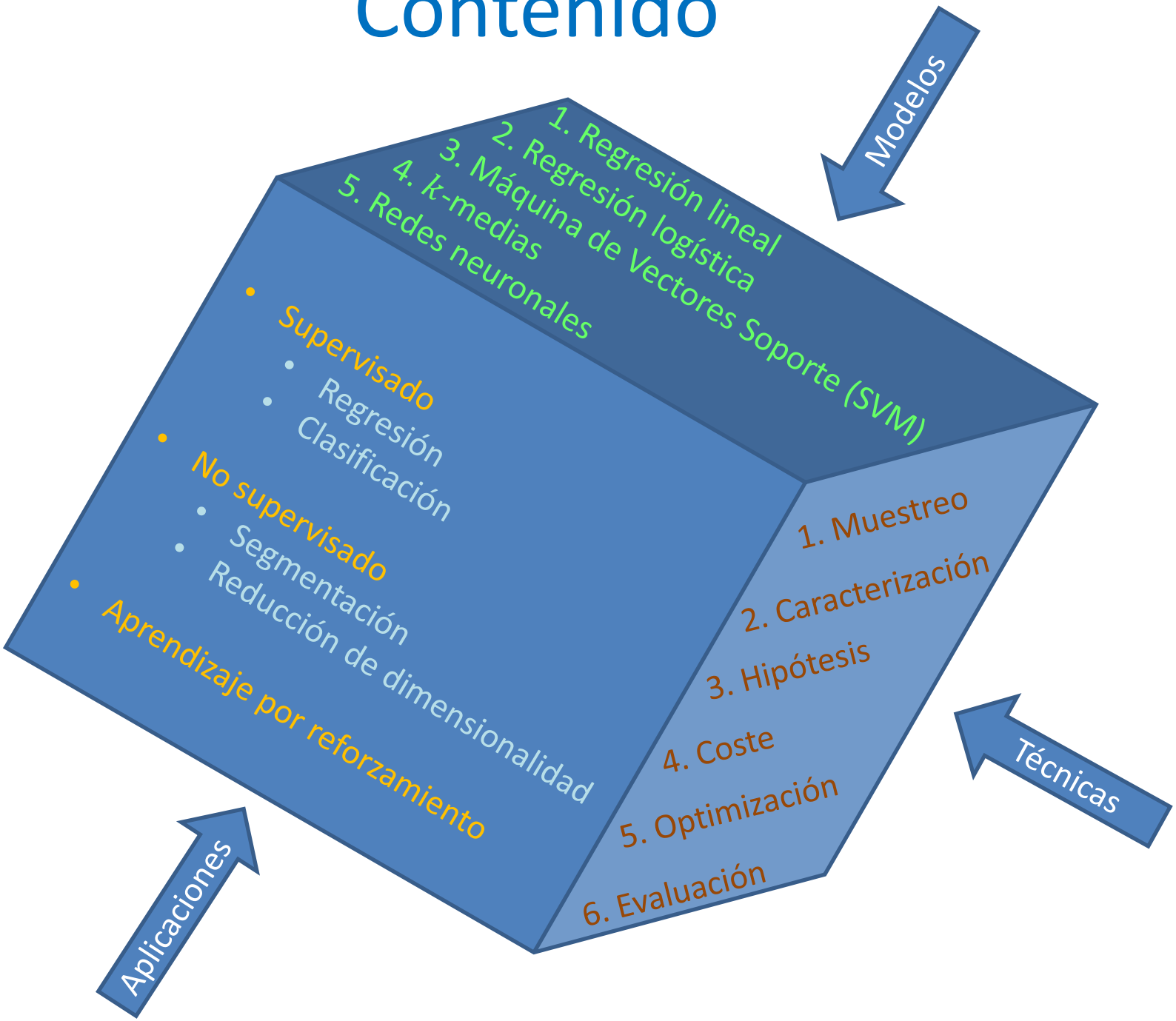
$$C_{\langle t \rangle} = U_{\langle t \rangle} \odot \tilde{C}_{\langle t \rangle} + F_{\langle t \rangle} \odot C_{\langle t-1 \rangle}$$

$$A_{\langle t \rangle} = O_{\langle t \rangle} \odot \psi\left(C_{\langle t \rangle}\right)$$

Contenido

1. Introducción
2. Regresión
   a) Regresión univariable
   b) Regresión multivariable
3. Clasificación
   a) Regresión logística
   b) Máquinas de vectores soporte (SVM)
      - Forma dual de la optimización (regresión y SVM)
   c) Funciones Kernel
   d) Clasificación multiclase
4. Segmentación
5. Reducción de dimensionalidad
6. Deep learning (introducción)

Contenido


Modelos
1. Regresión lineal
2. Regresión logística
3. Máquina de Vectores Soporte (SVM)
4. k-medias
5. Redes neuronales
Supervisado
Regresión
Clasificación
No supervisado
Segmentación
Reducción de dimensionalidad
Aprendizaje por reforzamiento
Técnicas
1. Muestreo
2. Caracterización
3. Hipótesis
4. Coste
5. Optimización
6. Evaluación
Aplicaciones

Niveles del *machine learning*

			f	φ	K	$\mathbf{X}$
Rule-based systems		$y = f(X)$	Manual	-	-	Manual
Machine Learning	Basic ML	$y = f(X)$	Automatic	-	-	Manual
	Feature-based ML	$y = f[\varphi(X)]$	Automatic	Manual	-	Raw
	Kernel-based ML	$y = f[K(X)]$	Automatic	Automatic	Manual	Raw
	Deep Learning	$y = f\big(\varphi(X)\big)$	Automatic	Automatic	-	Raw

Contenido

Técnicas

- Aprendizaje (por técnicas)
  - Muestreo
    - Simple
    - Hold-out
    - Cros-validación
  - Determinación de características (features)
  - Formulación de hipótesis
  - Función de coste
  - Optimización
  - Evaluación

Contenido

Técnicas

- Aprendizaje (por técnicas)
  - Muestreo
  - Determinación de características (features)
    - Representación multidimensional
    - Codificación 1-hot
    - Normalización de variables
    - Kernels
    - Análisis de componentes principales
  - Formulación de hipótesis
  - Función de coste
  - Optimización
  - Evaluación

Contenido

Técnicas

- Aprendizaje (por técnicas)
  - Muestreo
  - Determinación de características (features)
  - Formulación de hipótesis
    - Lineal
    - Polinómica
    - Logística
  - Función de coste
  - Optimización
  - Evaluación

Contenido

Técnicas

- Aprendizaje (por técnicas)
  - Muestreo
  - Determinación de características (features)
  - Formulación de hipótesis
  - Función de coste
    - Cuadrática
    - Logística
    - Bisagra
    - Distorsión
    - Regularización (regresión de arista)
  - Optimización
  - Evaluación

Contenido

Técnicas

- Aprendizaje (por técnicas)
  - Muestreo
  - Determinación de características (features)
  - Formulación de hipótesis
  - Función de coste
  - Optimización
    - Ecuación normal
      - Simple
      - Descomposición en Valores Singulares
    - Gradiente descendiente
      - Simple
      - Estocástico
      - Lotes
    - Optimización dual
    - Método del codo
  - Evaluación

Contenido

Técnicas

- Aprendizaje (por técnicas)
  - Muestreo
  - Determinación de características (features)
  - Formulación de hipótesis
  - Función de coste
  - Optimización
  - Evaluación
    - Bootstrap
    - Compromiso sesgo-varianza
    - Curva de aprendizaje
    - Matriz de confusión
    - Medidas de prestaciones
    - Silueta
    - Índice de Rand

Líneas de continuación

- Otros algoritmos de predicción
  - Random forest (árboles de decisión)
  - Gaussian process
  - …
- Otras arquitecturas
  - Self-Organizing Map (SOM)
  - Extreme Learning Machine (ELM)
  - Generative Adversarial Network (GAN)
  - …
- Evaluación
  - Robustness & stability
  - Fairness
  - Sensibilidad a features
  - Explicación
  - …

indra
minsait

¡¡¡¡ GRACIAS !!!!

Fundamentos del aprendizaje automático

(Machine learning)

Joaquín Luque