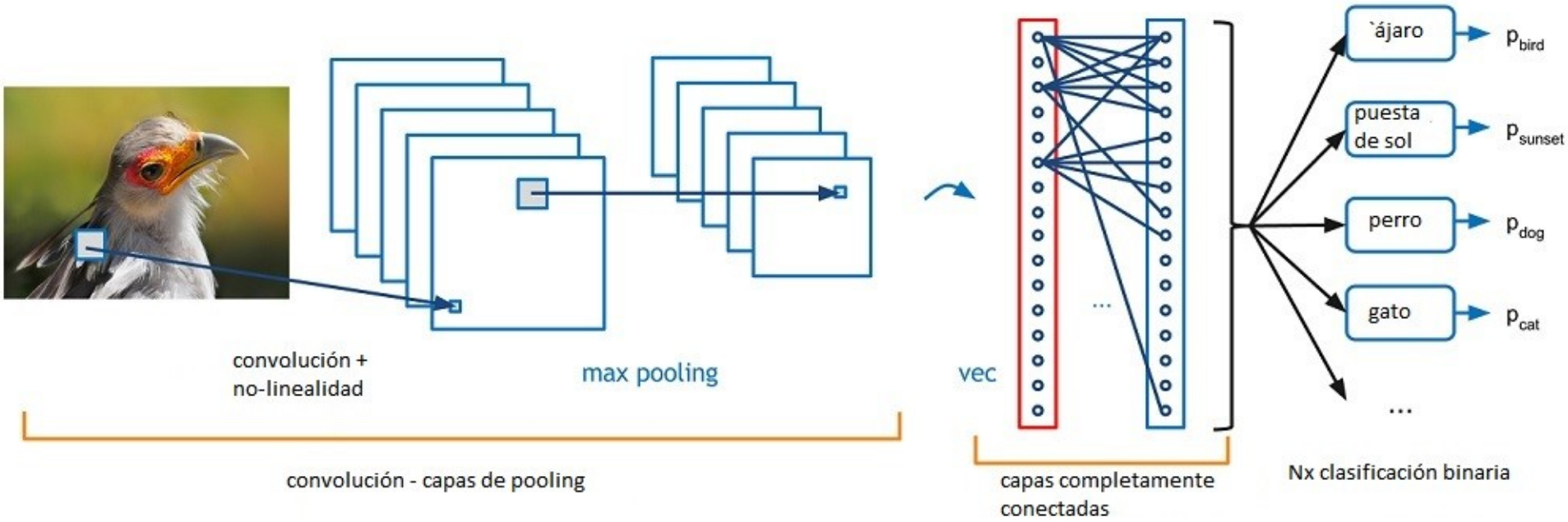
REDES CONVOLUCIONALES

Primer Cuatrimestre 2025

REDES CONVOLUCIONALES

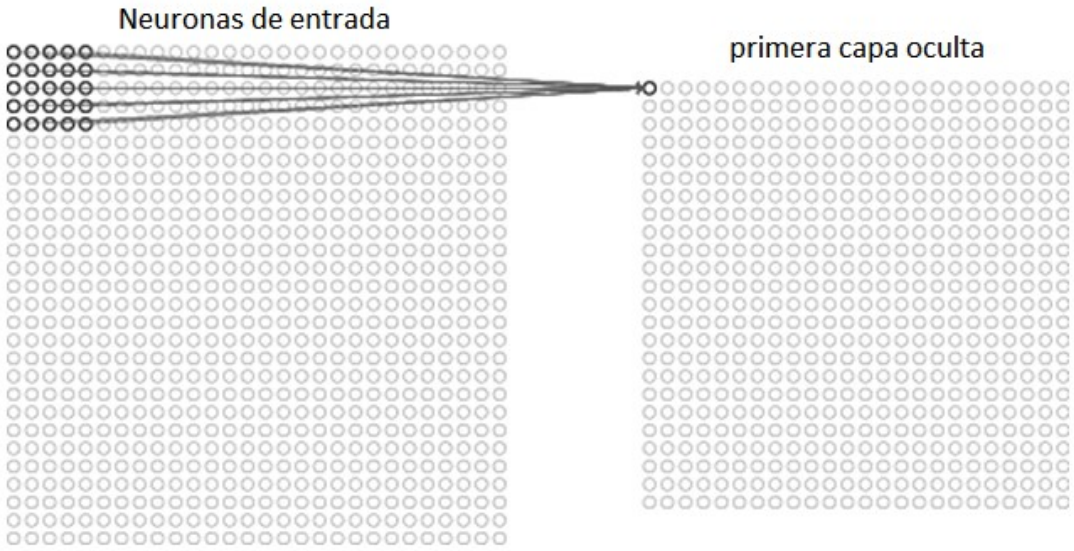


REDES CONVOLUCIONALES PARA CLASIFICACION

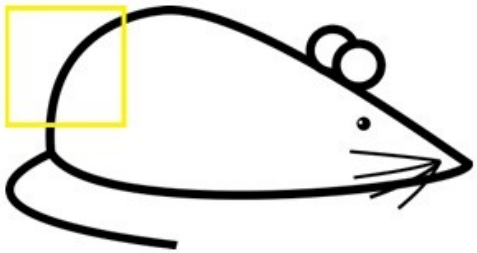
Entrada: imagen digitalizada
Salida: vector con distribución de probabilidades de pertenencia a distintas clases

Inspiración biológica: experimento de Hubel y Wiesel (1962)
- La corteza visual posee regiones de células sensitivas a regiones específicas del campo visual.
- Algunas neuronas individuales se activaban sólo en presencia de bordes de una cierta orientación (e.g. verticales, horizontales, diagonales)
- Neuronas organizadas en arquitectura por columnas.

Primera capa: *convolucional*



Filtro de 5 x 5 convolucionado sobre la entrada para producir un mapa de activación



filtro aplicado a la imagen

campo receptivo

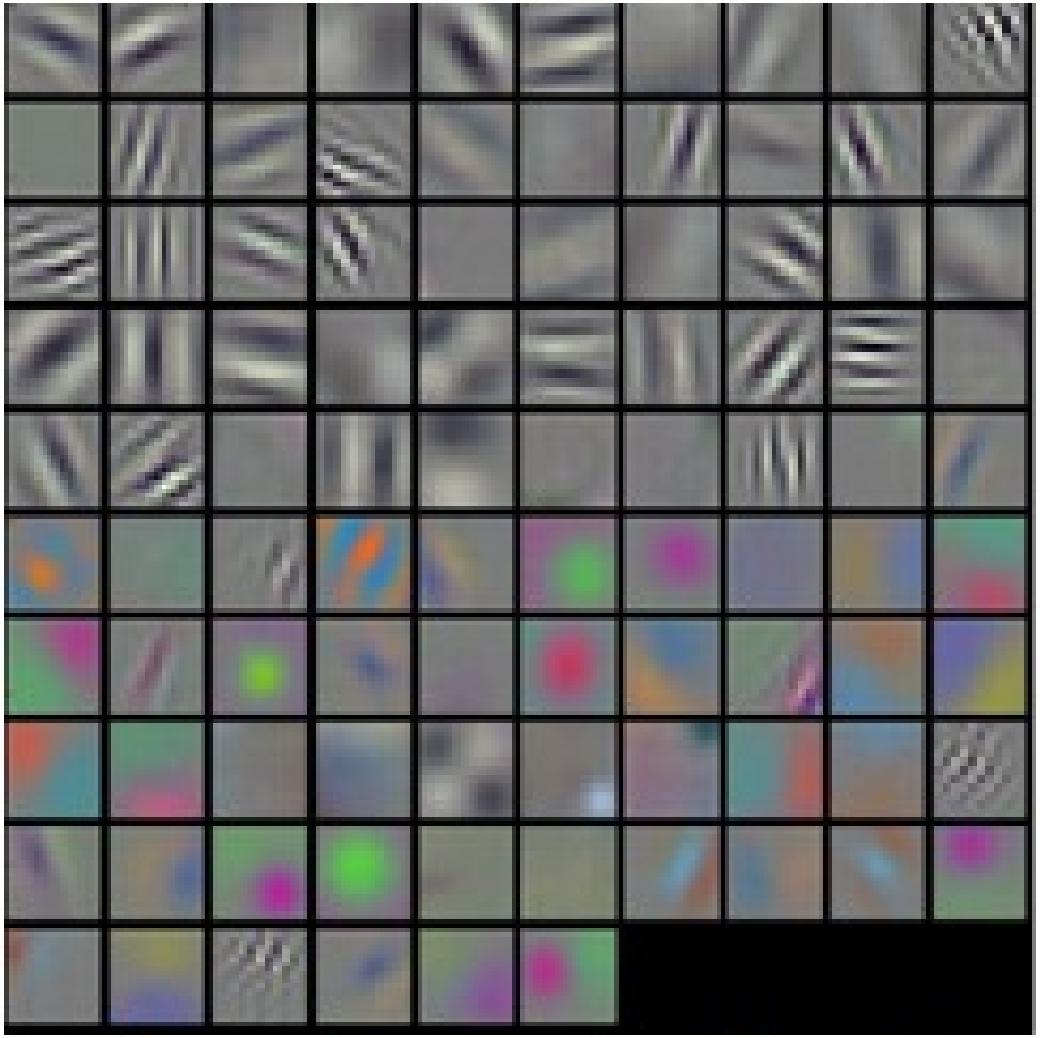
0	0	0	0	0	0	30
0	0	0	0	50	50	50
0	0	0	20	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0

representación en pixels

*

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Filtro



Filtros de una aplicación real

Capas ReLU (*rectified linear units*): después de cada capa convolucional

Activación: $f(x) = \max(0,x)$

Función: introducir no-linealidad sin afectar los campos receptivos de las capas convolucionales → Sustituyen a las clásicas *Tanh* o *sigmoidea logística*

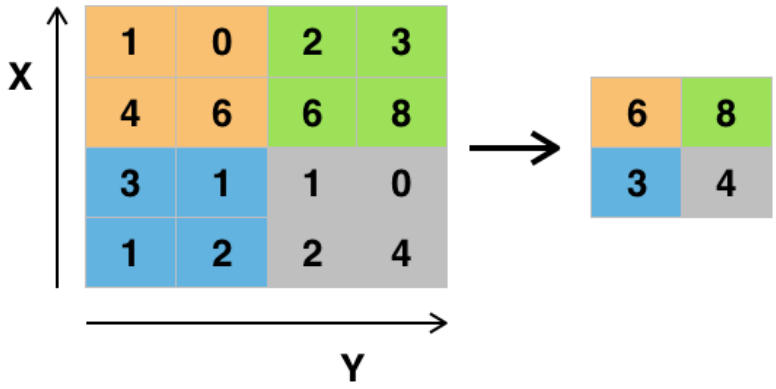
Ventajas:

- Eficiencia computacional (velocidad de aprendizaje)

- Evitar el problema de la anulación del gradiente (*vanishing gradients,* decrecimiento exponencial a lo largo de las capas)

Capas de pooling o subsampleo: filtro (2 x 2 en general)

Opción clásica: *maxpooling*



Otras posibilidades: Promedio, Norma L2

Pérdida de precisión en la ubicación exacta de cada feature, sin perder información sobre la posición relativa entre ellos.

Objetivos:
- Reducción de complejidad y costo computacional (**± 75%)**
- Controlar el sobreentrenamiento (*overfitting*)

Capas de descarte (“dropout”)

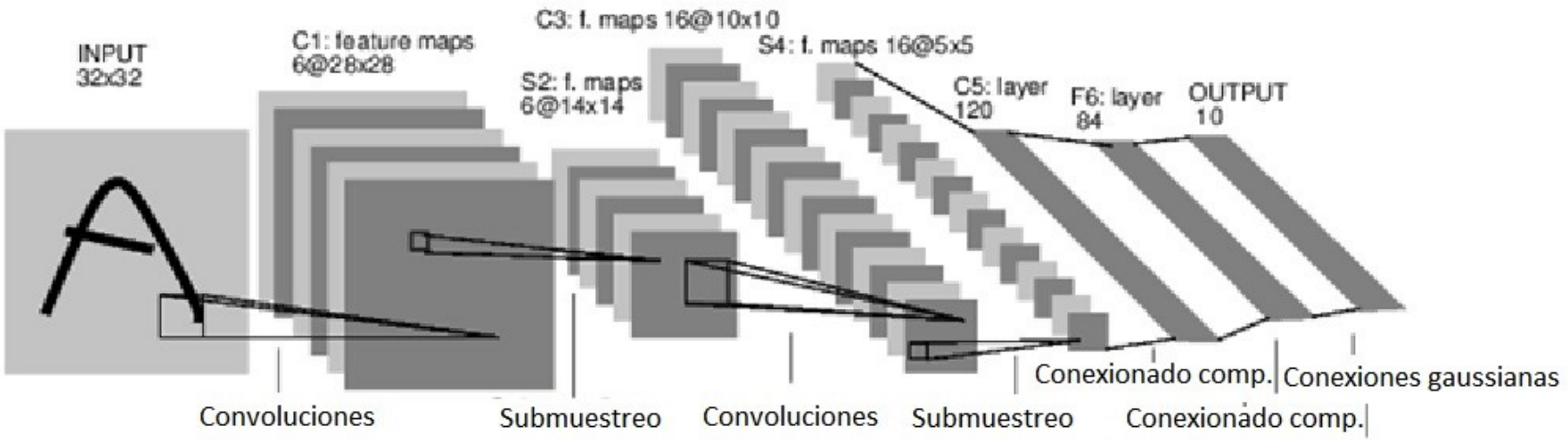
- Sólo etapa forward
- Descarta un porcentaje de activaciones al azar
- Fuerza a la red a ser redundante: deberá clasificar correctamente un ejemplo específico incluso si algunos pesos son anulados al azar.
- Controla el *overfitting*

Se usa sólo para entrenamiento, no para testeo

Capa final: completamente conectada.

- Entrada: capa que representa las características de más alto nivel
- Salida: vector de probabilidades según la correlación entre esas características y sus pesos, que representan las caractrísticas asociadas a cada clase almacenada.

REDES CONVOLUCIONALES: ESTRUCTURA



Una red convolucional completa (LeNet)

Estructura general (tipica):

Entrada→Conv→ReLU→Conv→ReLu→Pool→ReLU→Conv→ReLu→Pool→Compl.Conect.

Otras estrategias de mejora

-*Transferencia de conocimiento*: para reducir las demandas de gran cantidad de datos.

Tomar un modelo pre-entrenado y ajustarlo (*fine-tuning*) con los datos propios.
→ se conservan las capas de extracción de características

- Se sustituye la última capa (de clasificación) por la adecuada al problema a resolver. → sólo se entrenan esos pesos.

Justificacion: se basa en la universalidad de las características de más bajo nivel (bordes, curvas). Si el modelo original fue entrenado con una gran base de datos y el nuevo problema no es excesivamente específico, pueden llagar a servir todas o casi todas las capas.

- *Técnicas de aumento de datos*
Para incrementar la robustez: escalas de gris, translaciones, rotaciones, etc.

EN SINTESIS

Capa de Entrada [sup. 32x32x3]: valores de los pixels de la imagen, ancho 32, altura 32, y tres canales de color R,G,B.

Capa Convolucional: calculará la salida de las neuronas que están conectadas a regiones locales en la entrada, cada una computando un producto escalar entre sus pesos y una pequeña región a la que están conectados en el volumen de entrada. Esto puede producir un volumen de, por ejemplo, [32x32x12] (si se usan 12 filtros).

Capa RELU: aplicará una función de activación elemento a elemento, tal como *max(0,x)* umbralando a cero. El volumen no cambia.

Capa de Pooling: submuestreo sobre las dimensiones espaciales (ancho, altura), reduciendo el volumen, por ejemplo, a [16x16x12].

Capa FC (totalmente conectada): calculará los puntajes de las clases, produciendo un volumen de tamaño [1x1xK], donde cada uno de los K números corresponde al puntaje de una clase. Cada neurona conectada a todos los elementos del volumen anterior.

Características generales

- *Estructura feed-forward*: salida de la n-esima capa = entrada de la n+1-esima

- *Mapa de activaciones de una capa = entrada a la siguiente*

- *Capas inferiores* (más cercanas a la entrada): describen las ubicaciones de las características (features) de bajo nivel

- *Capas superiores*: sus salidas representan características de más alto nivel
  Ej.: semicírculos (combinación de curva y línea recta)
  Cuadrado (combinación de varias líneas rectas)

- *A mayor profundidad, mayor área de los campos receptivos*

REDES CONVOLUCIONALES: ENTRENAMIENTO

Supervisado con backpropagation:

- Etapa forward: cada patrón (imagen) como entrada
  Salida deseada : rótulo según su clase

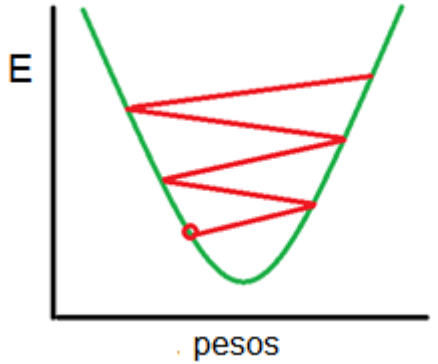
- Etapa backward

$$E = \sum \frac{1}{2}\left(\text{salida esperada - salida actual} \right)^2$$

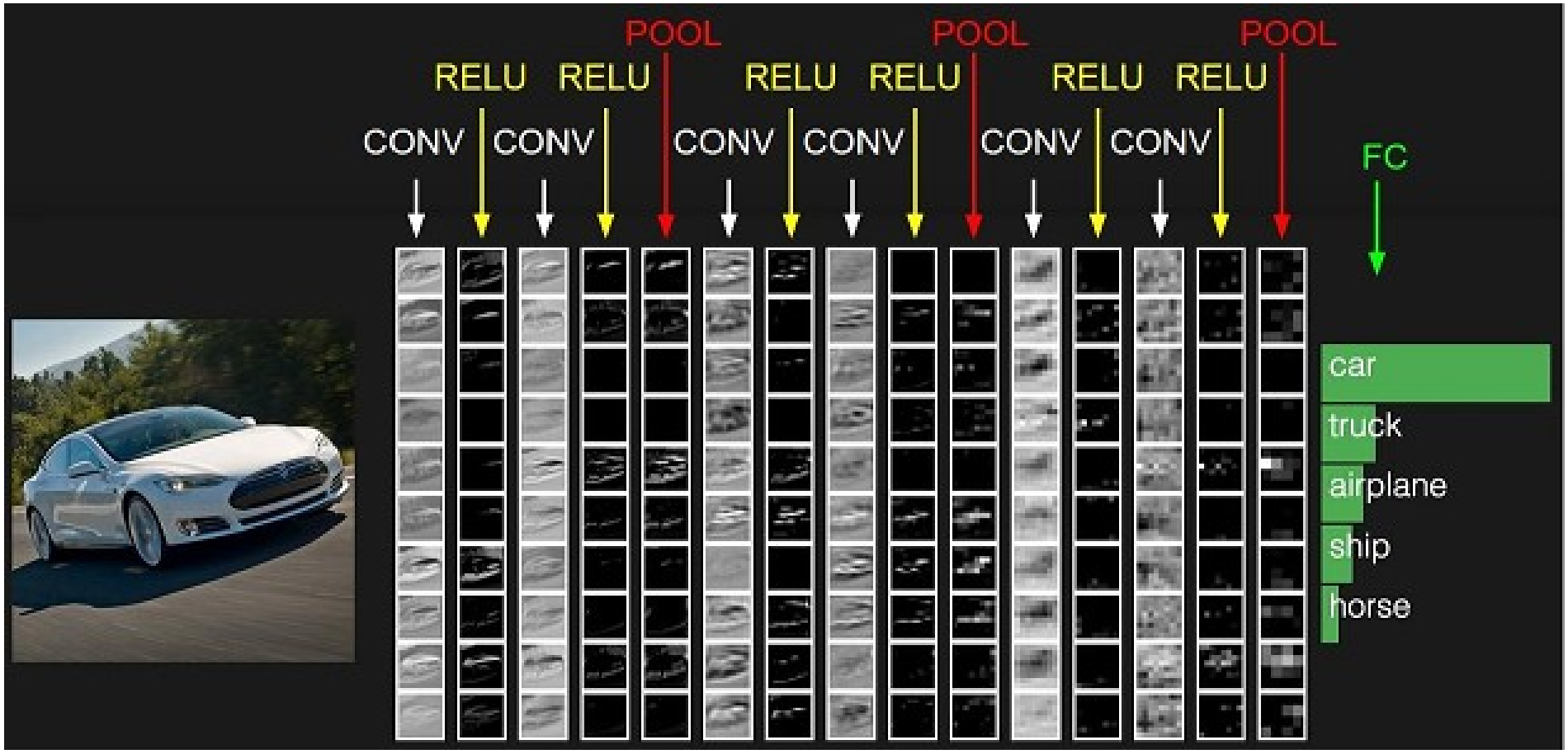
$w = w_i - \eta \, dE/dw$

Se actualizan los pesos de todas las capas, filtros incluidos

Atención: alta sensibilidad a tasa de entrenamiento η



consecuencia típica de una velocidad de aprendizaje excesiva


POOL
POOL
POOL
RELU
RELU
RELU
RELU
RELU
RELU
CONV
CONV
CONV
CONV
CONV
CONV
FC
car
truck
airplane
ship
horse