

# Deep Learning II

## *redes neuronales convolucionales profundas*

---

MIGUEL ÁNGEL MARTÍNEZ DEL AMOR

DEPARTAMENTO CIENCIAS DE LA COMPUTACIÓN E INTELIGENCIA ARTIFICIAL

UNIVERSIDAD DE SEVILLA



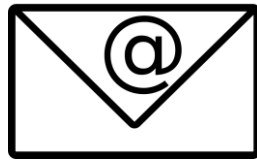
# About me

---

- **Miguel Ángel Martínez del Amor**
- Profesor Ayudante Doctor del Departamento de Ciencias de la Computación e Inteligencia Artificial



[www.cs.us.es/~mdelamor](http://www.cs.us.es/~mdelamor)



[mdelamor@us.es](mailto:mdelamor@us.es)



[@miguelamda](https://twitter.com/miguelamda)



[miguelamda](https://github.com/miguelamda)



[Research Group on Natural Computing](#)



[DeepKnowledge](#)



DEEP  
LEARNING  
INSTITUTE

[NVIDIA Deep Learning Institute](#)

# Warning!

---

- Si quieres reproducir el código que veremos al final, te aconsejo que:
  - O bien tengas abierta una sesión con tu **cuenta de Gmail** (si no tienes, hazte una).
  - O bien tengas instalado **Python 3** en local junto con *Jupyter, Keras 2.2.5, Tensorflow 1.15, sklearn, matplotlib, numpy...*



# Índice

---

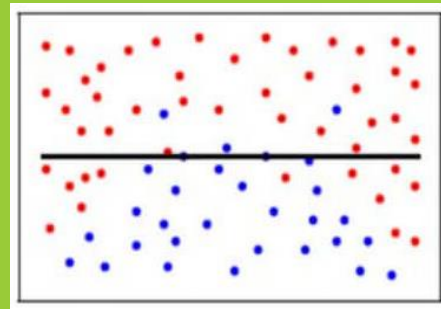
1. Introducción a la regularización.
2. Redes neuronales convolucionales.
3. Ejercicio 1: nuestra primera red neuronal convolucional con Keras.
4. Hardware para Deep Learning.
5. Algunas redes neuronales profundas y transferencia de aprendizaje.
6. Ejercicio 2: transfer learning con Keras.

# Índice

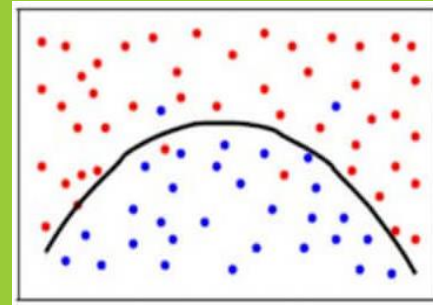
---

1. Introducción a la regularización.
2. Redes neuronales convolucionales.
3. Ejercicio 1: nuestra primera red neuronal convolucional con Keras.
4. Hardware para Deep Learning.
5. Algunas redes neuronales profundas y transferencia de aprendizaje.
6. Ejercicio 2: transfer learning con Keras.

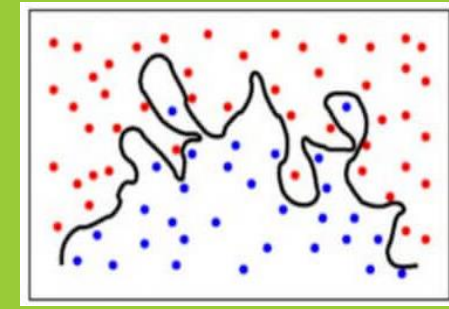
# Problemas de rendimiento



Underfitting



Correcto



Overfitting

Entrenamiento

Test

Malo

Malo

Bueno

Bueno

Muy bueno

Malo

# Problemas de rendimiento

---



# Necesidad de regularización

---

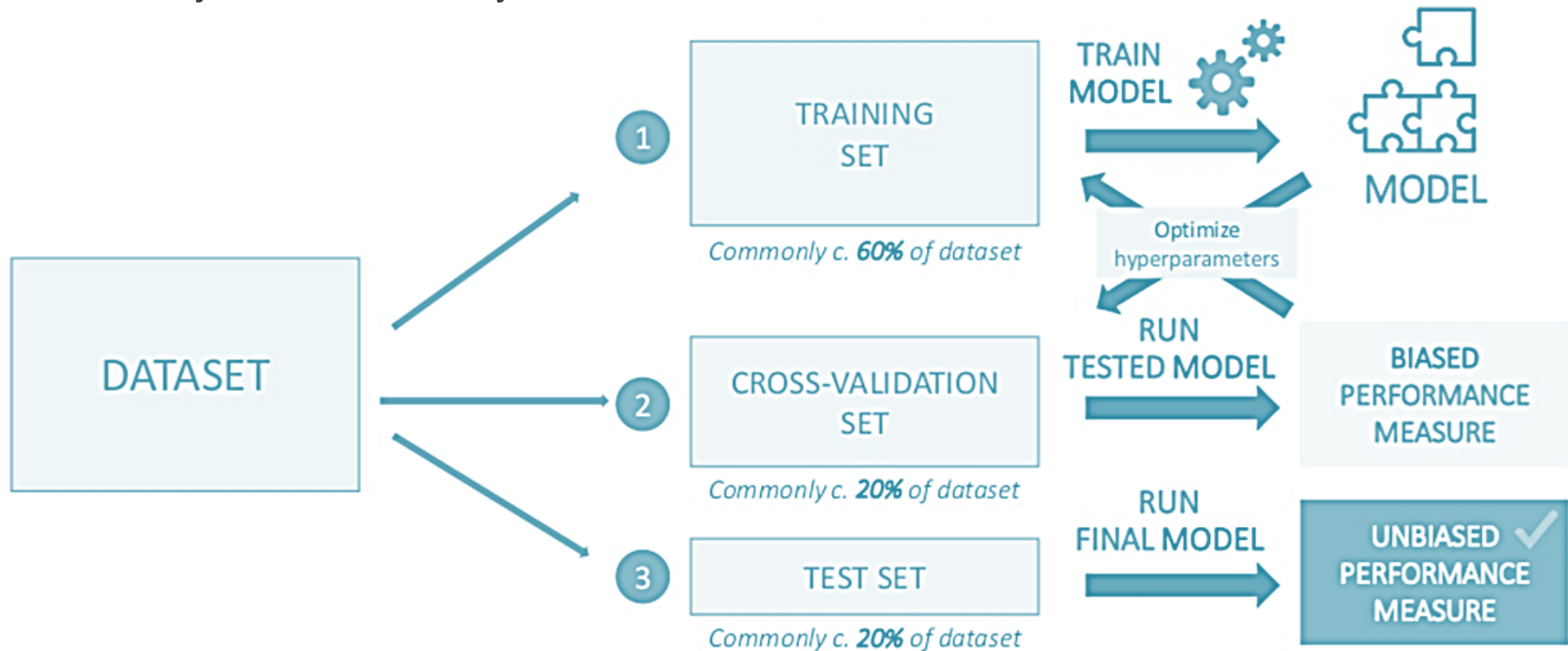
Técnicas para atacar el overfitting son:

- **Simplificar** los datos, reduciendo el número de características
  - Manualmente
  - Técnicas de reducción de dimensionalidad (PCA, t-SNE, ...)
- **Regularización**
  - Mantenemos todas las características
  - Reducimos los valores de los parámetros del modelo
  - Funciona bien cuando tenemos muchas características y todas aportan un poco a predecir



# Early stopping

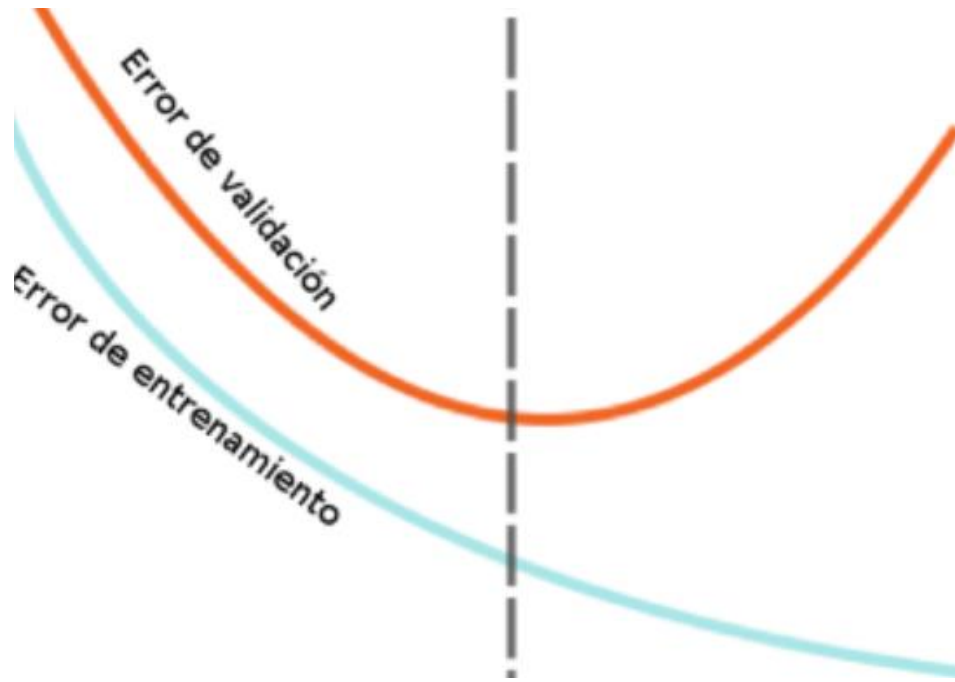
Partir el conjunto en 3 subconjuntos:



# Early stopping

---

**Idea:** detener el entrenamiento cuando el error cometido sobre el conjunto de validación crece.

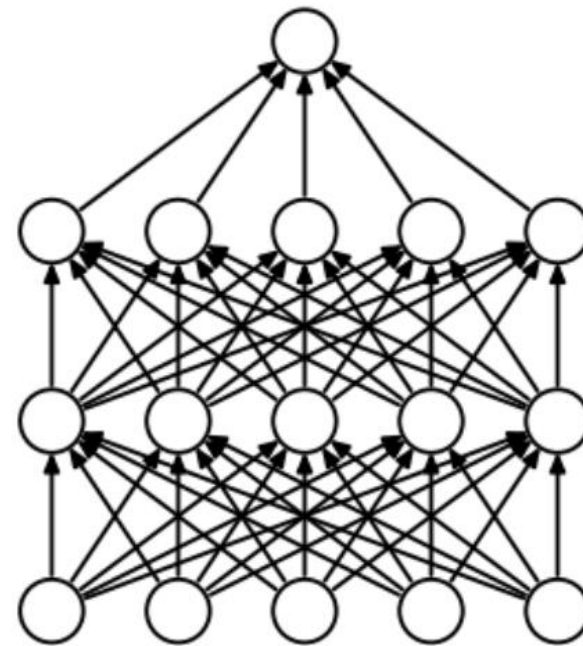


# Dropout

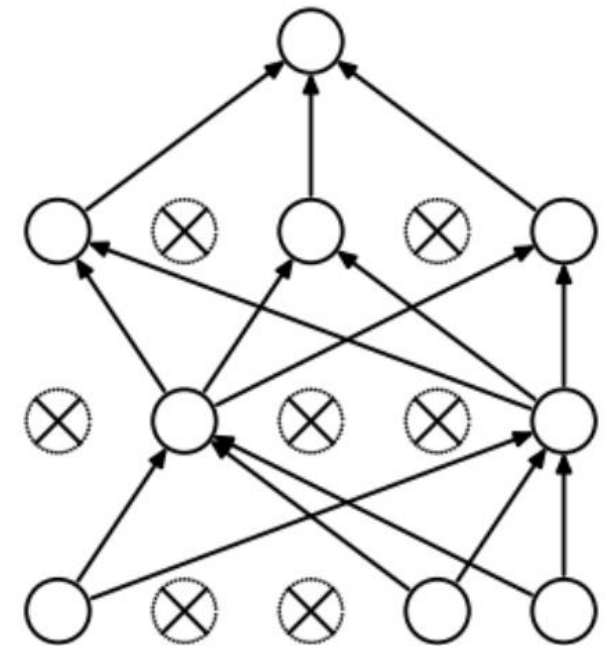
**Idea:** aleatoriamente **poner a cero** algunas neuronas en la propagación hacia adelante **Hiperparámetro:  $p$**

- Probabilidad de poner a cero

[[Srivastava et al 2014](#)]



(a) Standard Neural Net

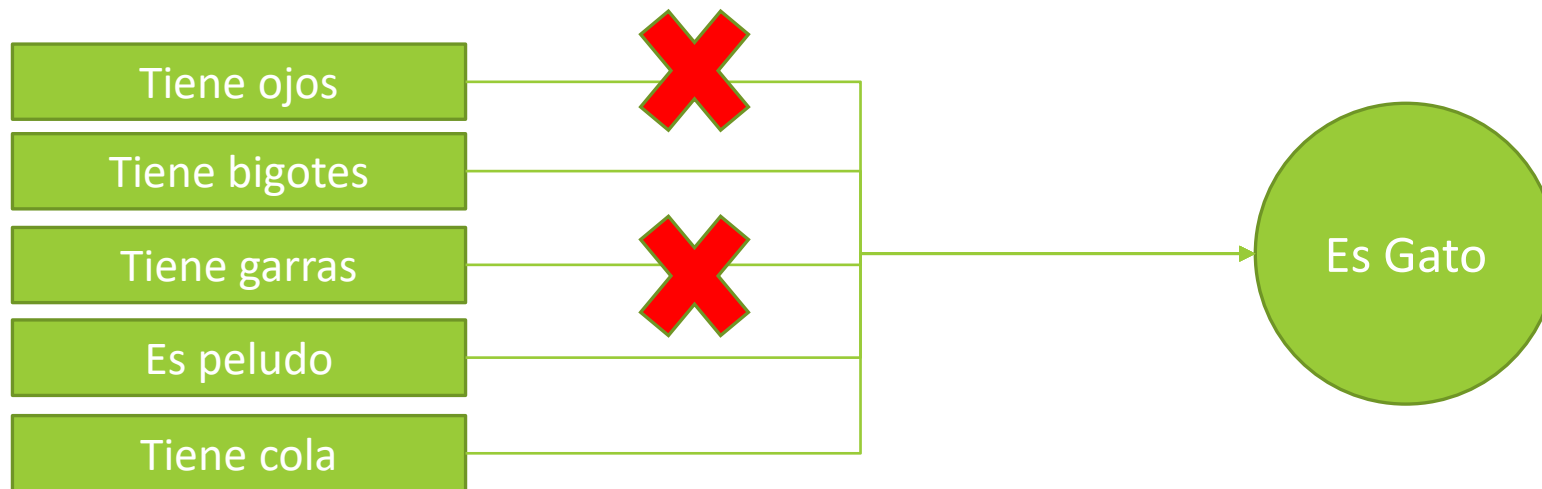


(b) After applying dropout.

# Dropout

## Fuerza a la red tener una representación más redundante

- La red encuentra otros “caminos” dentro de la red para llegar a la misma conclusión

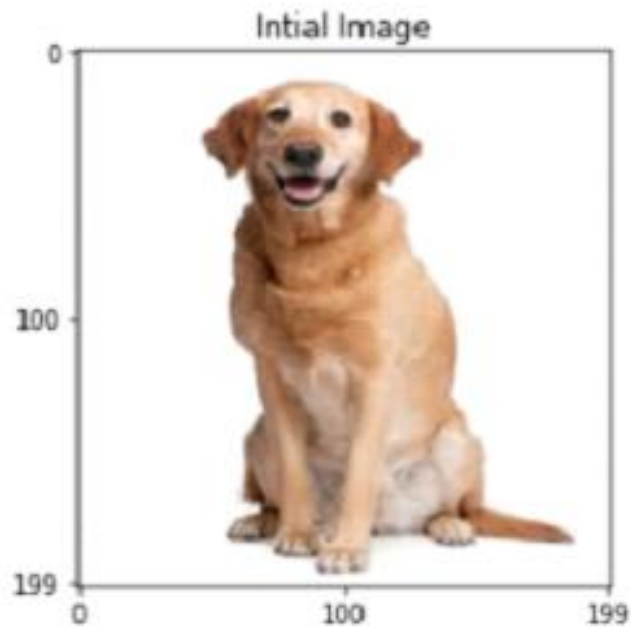


# Aumentado de datos

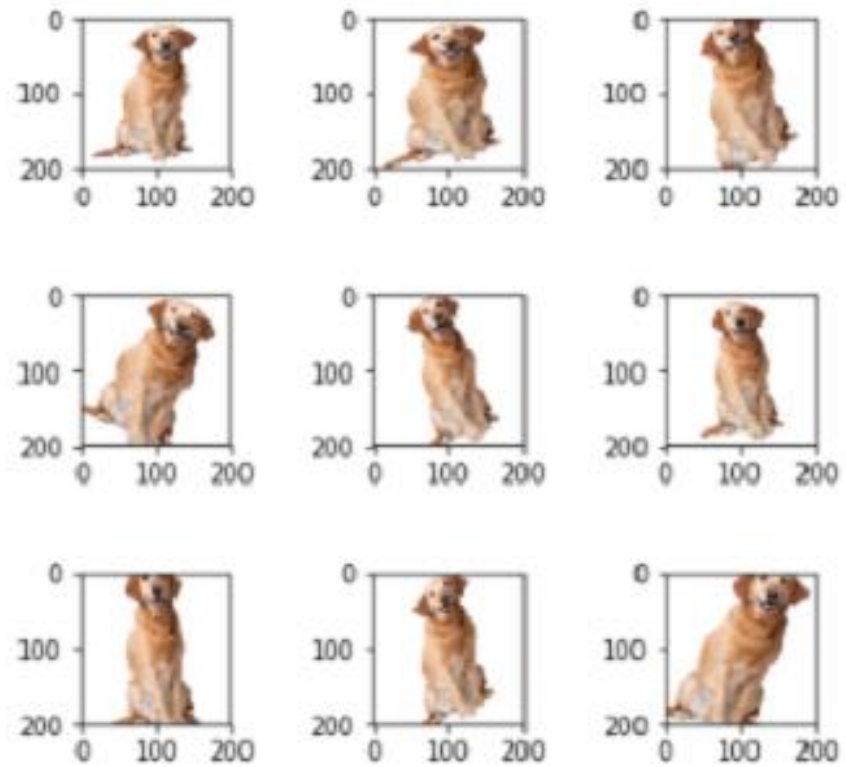
Transformaciones típicas: mirroring, cropping



# Data augmentation



Augmented Images



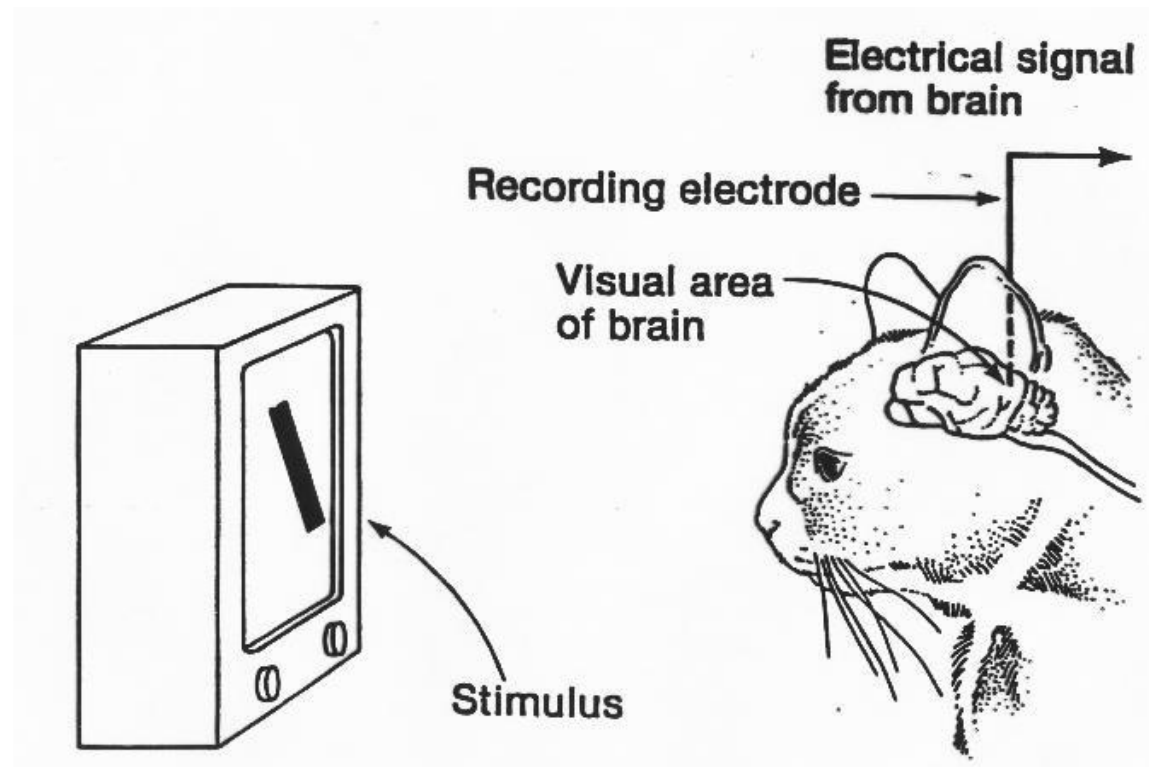
# Índice

---

1. Introducción a la regularización.
2. Redes neuronales convolucionales.
3. Ejercicio 1: nuestra primera red neuronal convolucional con Keras.
4. Hardware para Deep Learning.
5. Algunas redes neuronales profundas y transferencia de aprendizaje.
6. Ejercicio 2: transfer learning con Keras.

# El origen

El experimento [Hubel & Wiesel](#) 1959, 1962, 1968



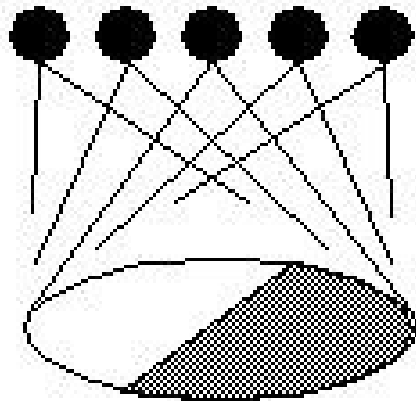


# El origen

El experimento [Hubel & Wiesel](#) 1959, 1962, 1968

## Hubel & Wiesel

topographical mapping

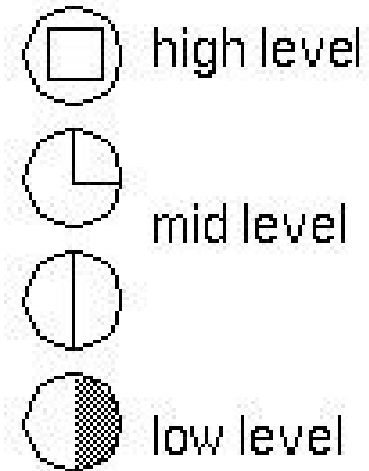
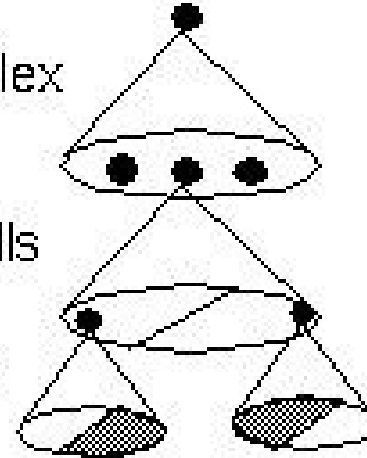


## featural hierarchy

hyper-complex cells

complex cells

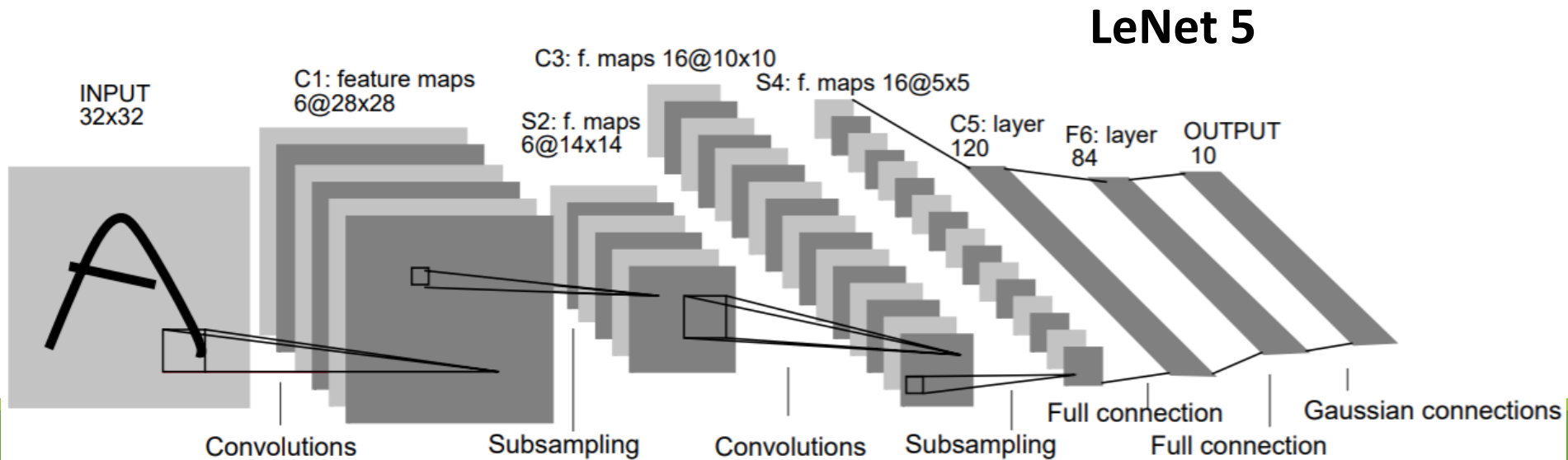
simple cells



# Primera red convolucional

*Gradient-based learning applied to document recognition* [[LeCun, Bottou, Bengio, Haffner 1998](#)]

- Reconocimiento de caracteres escritos a mano.
- Ya primeros experimentos en [1993](#).
- Entrenado con **backpropagation**.

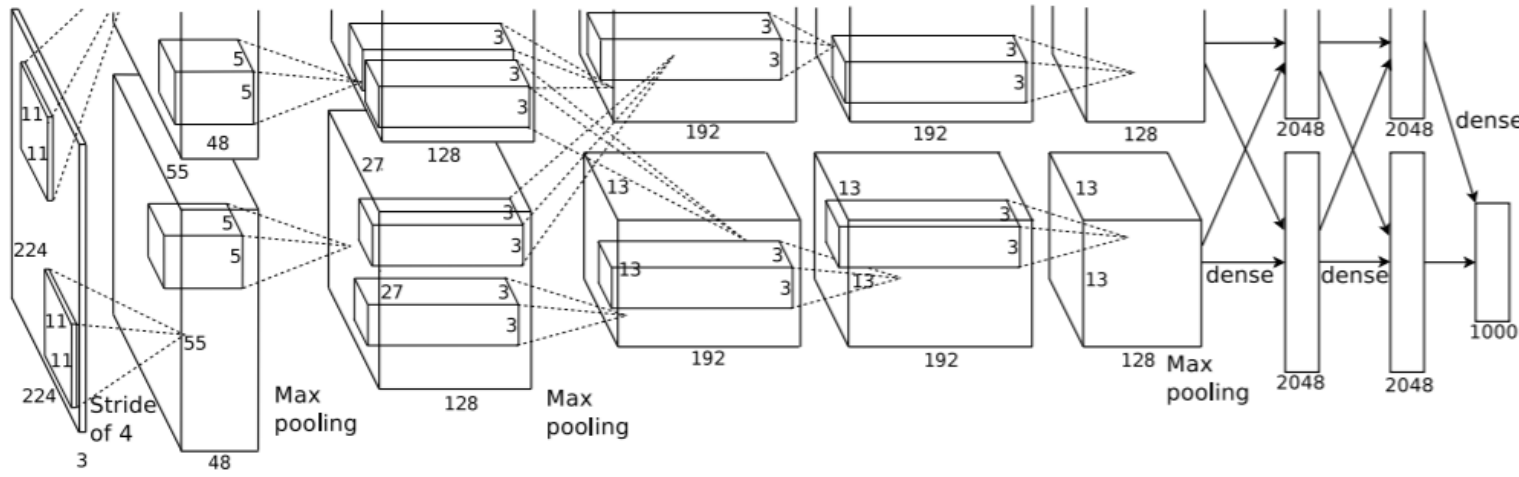


# El gran salto

*ImageNet Classification with Deep Convolutional Neural Networks* [[Krizhevsky, Sutskever, Hinton 2012](#)]. **Diferencias:**

- Entrenado sobre ImageNet (10<sup>6</sup> imágenes, 1000 categorías)
- Usa ReLU (LeNet 5 usa tanh) y dropout
- Más profundo
- Uso de GPUs (6 días)

## AlexNet



IMAGENET

## Tendencia

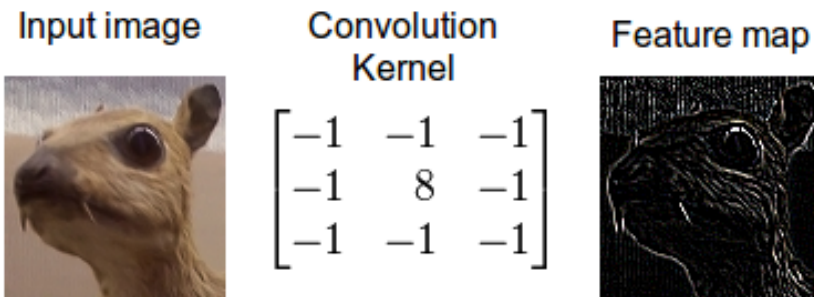
The diagram illustrates the progression of convolutional neural networks (CNNs) through several key architectures, each paired with a corresponding meme face representing the increasing complexity and 'madness' of the models.

- LeNet-5:** The first architecture shown, featuring a simple input image (a handwritten 'A') processed through convolutional layers (C1, C2), sub-sampling, and fully connected layers (F6, F7). The output is a Gaussian connection. The associated meme face is a simple, wide-eyed, open-mouthed expression.
- Convolution networks:** A more complex architecture with multiple convolutional layers (C1, C2, C3, C4, C5), sub-sampling, and fully connected layers (F6, F7). The output is a Gaussian connection. The associated meme face is a simple, wide-eyed, open-mouthed expression.
- AlexNet:** A more complex architecture with multiple convolutional layers (C1, C2, C3, C4, C5), sub-sampling, and fully connected layers (F6, F7). The output is a Gaussian connection. The associated meme face is a simple, wide-eyed, open-mouthed expression.
- VGG-19:** A more complex architecture with multiple convolutional layers (C1, C2, C3, C4, C5), sub-sampling, and fully connected layers (F6, F7). The output is a Gaussian connection. The associated meme face is a simple, wide-eyed, open-mouthed expression.
- Deep learning:** A more complex architecture with multiple convolutional layers (C1, C2, C3, C4, C5), sub-sampling, and fully connected layers (F6, F7). The output is a Gaussian connection. The associated meme face is a simple, wide-eyed, open-mouthed expression.
- ResNet-34:** The final architecture shown, featuring a complex residual network structure with multiple convolutional layers (C1, C2, C3, C4, C5), sub-sampling, and fully connected layers (F6, F7). The output is a Gaussian connection. The associated meme face is a simple, wide-eyed, open-mouthed expression.

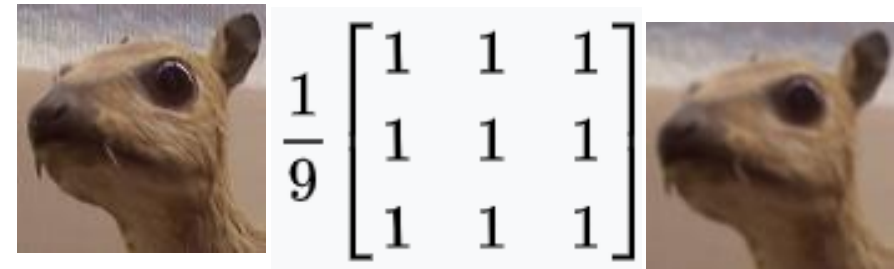
何だこりゃ！

# Convolución

## Ejemplos de convolución sobre imágenes



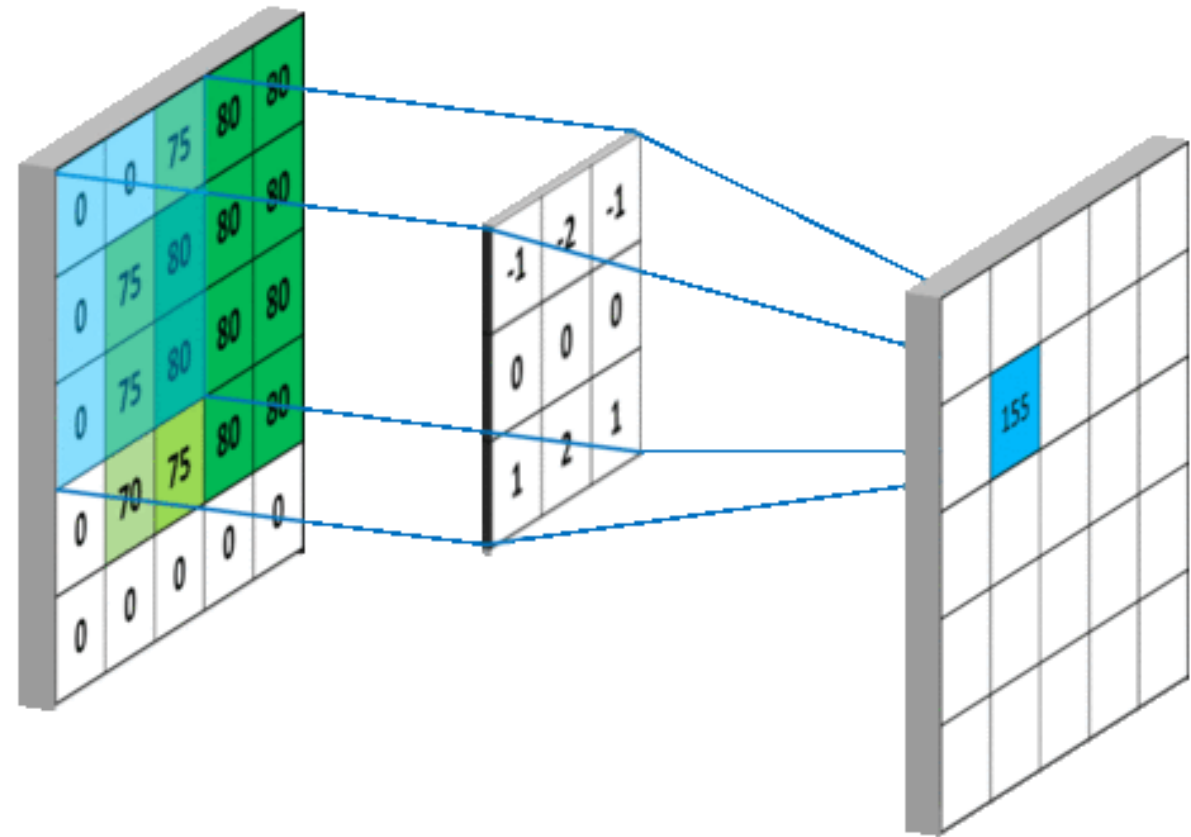
Detección bordes



Difuminado (Blur)

# Convolución

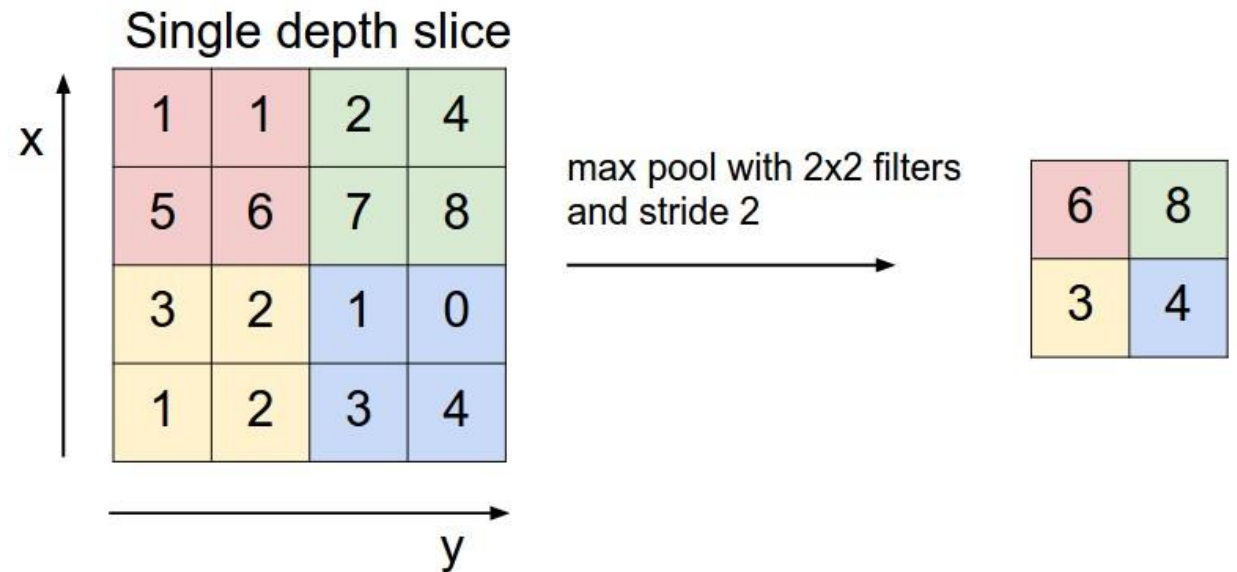
- Operación básica:
  - Multiplicación elemento a elemento
  - Suma de las multiplicaciones
- Para cada posible encaje del **kernel (filtro)** en la matriz de entrada:
  - Aplicar operación de convolución
  - Anotar el valor de salida en la matriz resultado
- Nota: esta operación es **derivable**



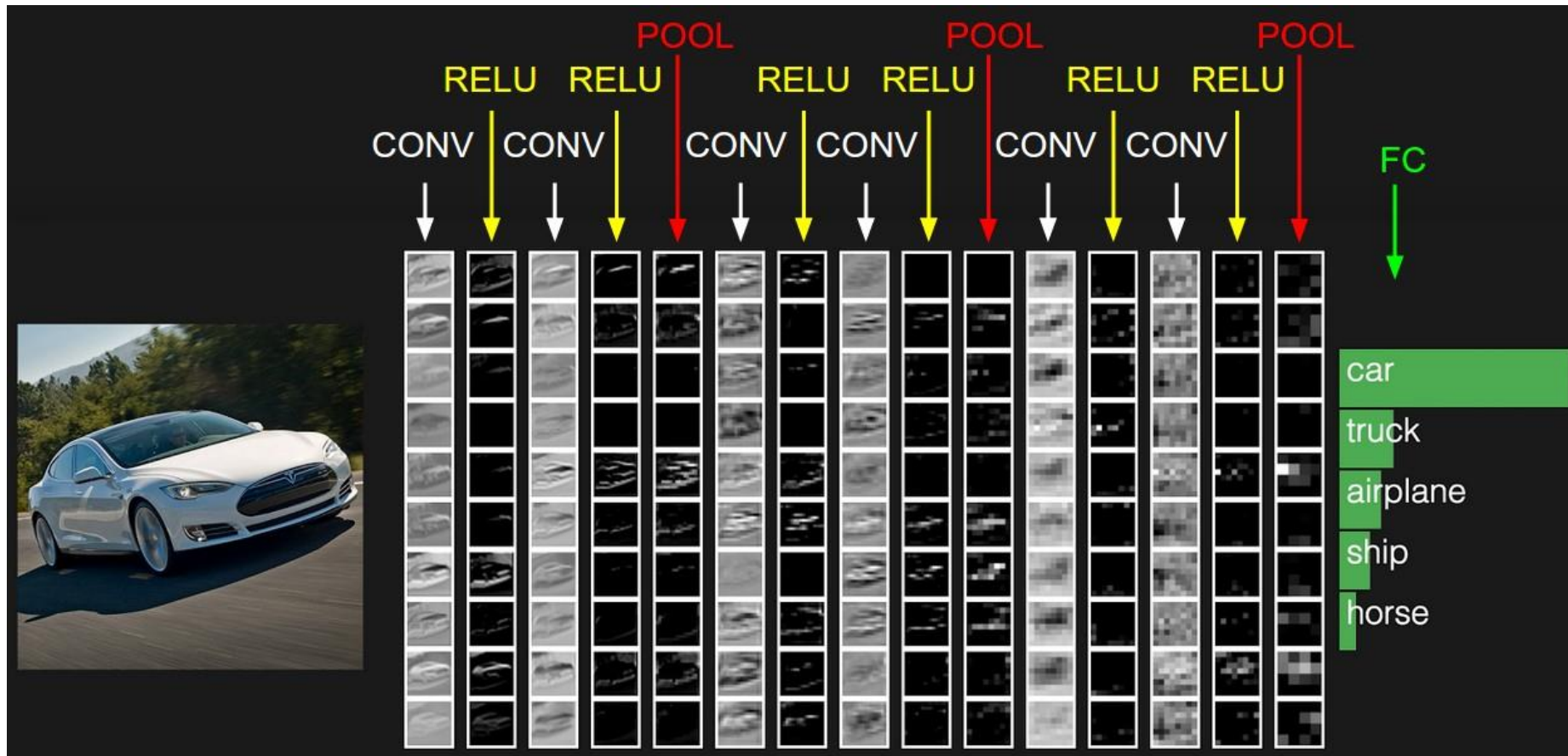
<https://analyticsindiamag.com/convolutional-neural-network-image-classification-overview/>

# Pooling (subsampling/downsampling)

- Capa que **redimensiona** espacialmente la representación
- Es común insertarla periódicamente entre capas convolutivas
- **Operaciones** típicas: MAX, AVG, SUM, L2, ...
- Es **derivable** para propagación de gradientes.



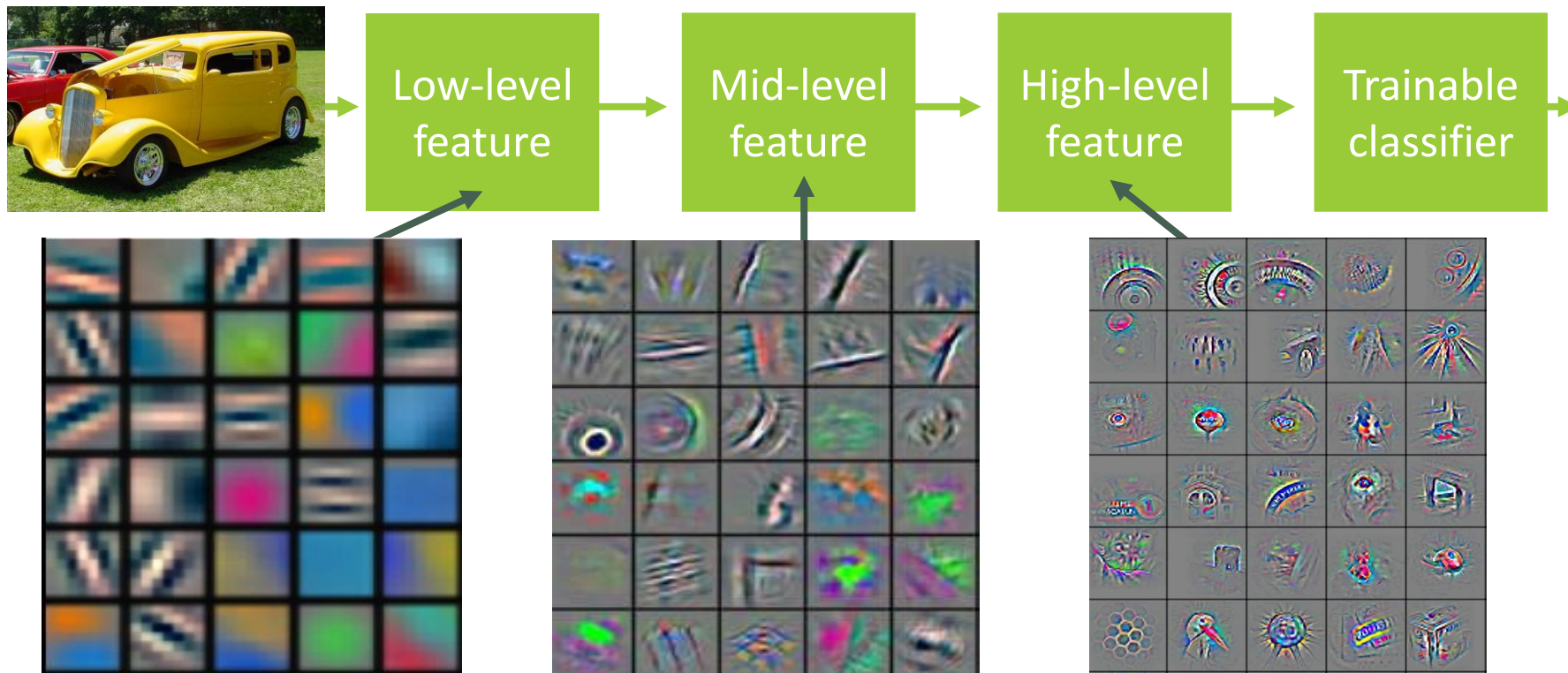
# Interpretando redes convolucionales





# Interpretando redes convolucionales

Visualización de los filtros aprendidos por una red ([ver aquí](#)).



# Índice

---

1. Introducción a la regularización.
2. Redes neuronales convolucionales.
3. Ejercicio 1: nuestra primera red neuronal convolucional con Keras.
4. Hardware para Deep Learning.
5. Algunas redes neuronales profundas y transferencia de aprendizaje.
6. Ejercicio 2: transfer learning con Keras.

# Índice

---

1. Introducción a la regularización.
2. Redes neuronales convolucionales.
3. Ejercicio 1: nuestra primera red neuronal convolucional con Keras.
4. **Hardware para Deep Learning.**
5. Algunas redes neuronales profundas y transferencia de aprendizaje.
6. Ejercicio 2: transfer learning con Keras.

# La trinidad del Deep Learning

---

## Big Data Availability

**facebook**

350 millions  
images uploaded  
per day

**Walmart** ✱

2.5 Petabytes of  
customer data  
hourly

**You Tube**

300 hours of video  
uploaded every  
minute

## New ML Techniques

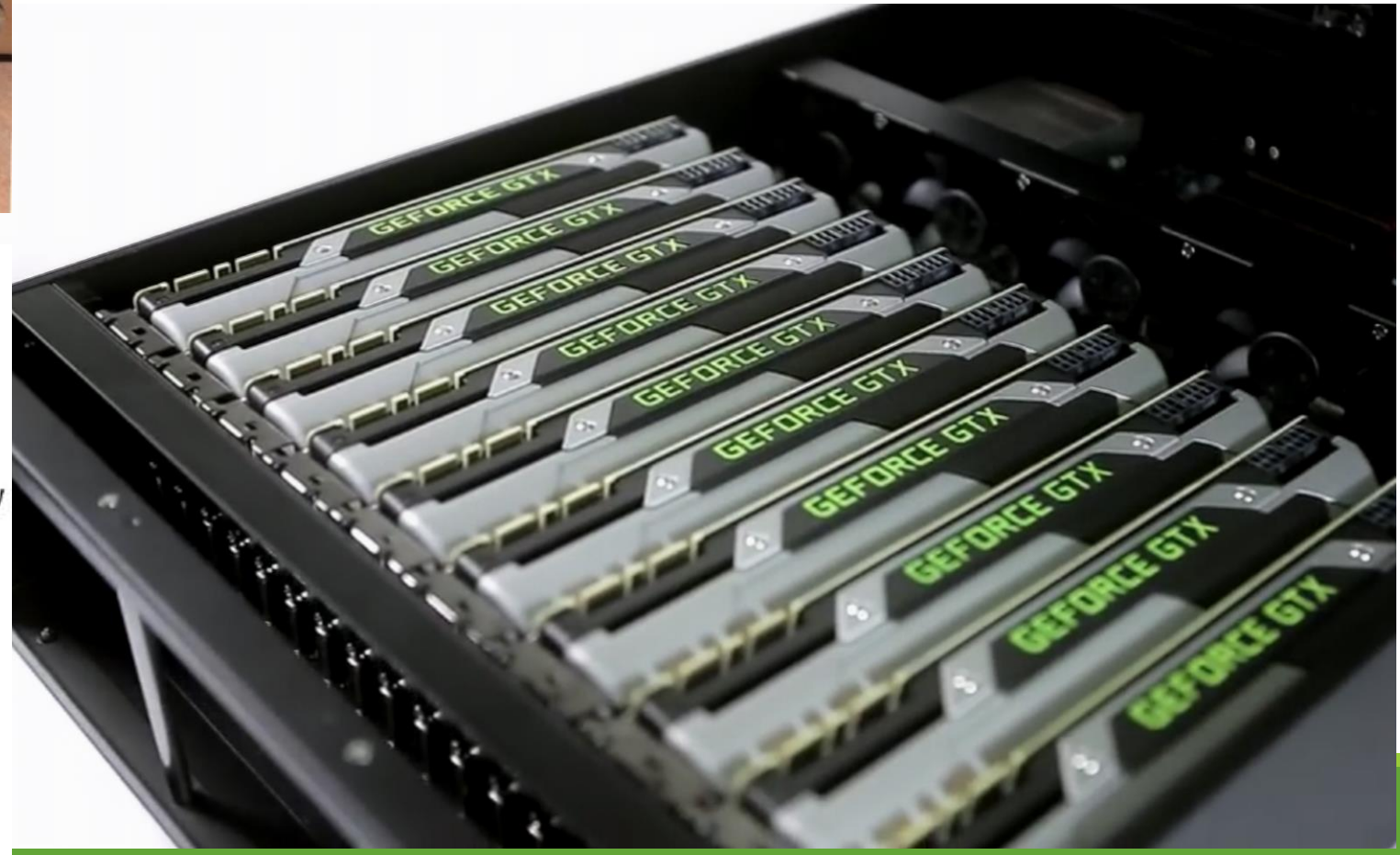
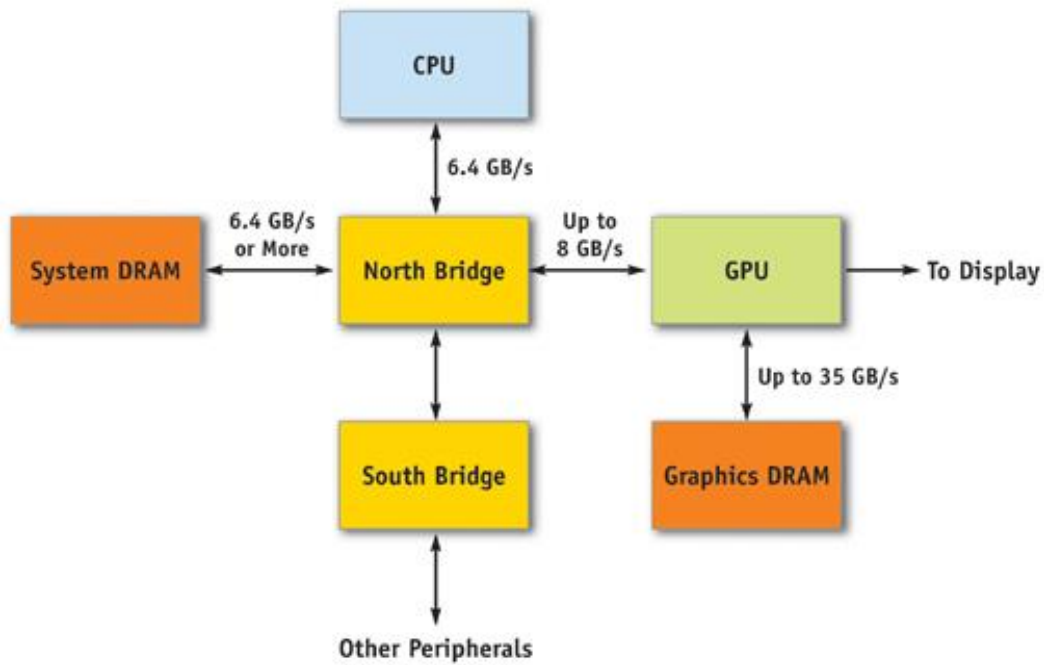


## GPU Acceleration



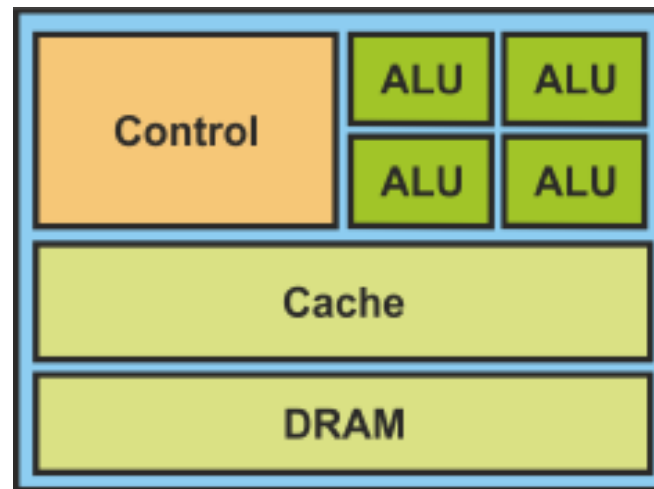


# Hardware paralelo: GPUs

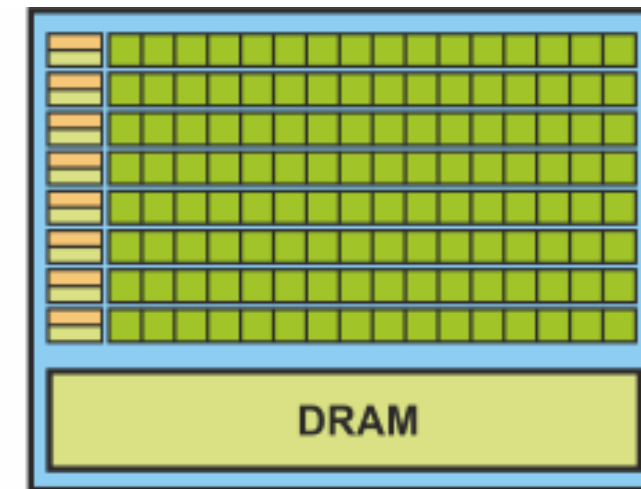


# Hardware paralelo: GPUs

- **GPU** = Graphics Processing Unit (núcleo tarjeta gráfica).
- Con el tiempo este procesador ha evolucionado y hoy en día se puede usar para cómputo paralelo, incluyendo del orden de cientos a miles de núcleos.
- **CUDA cores (Streaming Processors)**: Los núcleos son más básicos que los de una CPU, pero son muchos más!
- Muy buenos con cálculo matricial.
- Ahora incluyen núcleos especiales para Deep Learning: **Tensor cores**



**CPU**



**GPU**

# Hardware paralelo: GPUs

---

- Librerías para programar GPUs:
  - **CUDA** (de NVIDIA)
  - **OpenCL** (Chronos → NVIDIA, AMD, Intel...)
  - **ROCm** (de AMD), para soportar CUDA-like code
- **NVIDIA** invirtió en Deep Learning desde el principio, y ahora ofrece un **mayor soporte** (todos los frameworks soportan de forma nativa GPUs mediante CUDA): CuDNN, CuBLAS, ...
- OpenCL y ROCm (**AMD**) es soportado de manera **experimental** por algunos frameworks.



# Hardware paralelo: TPUs

---

**TPU** = Tensor Processing Unit

**Chip** introducido por Google en 2016 para Deep Learning.

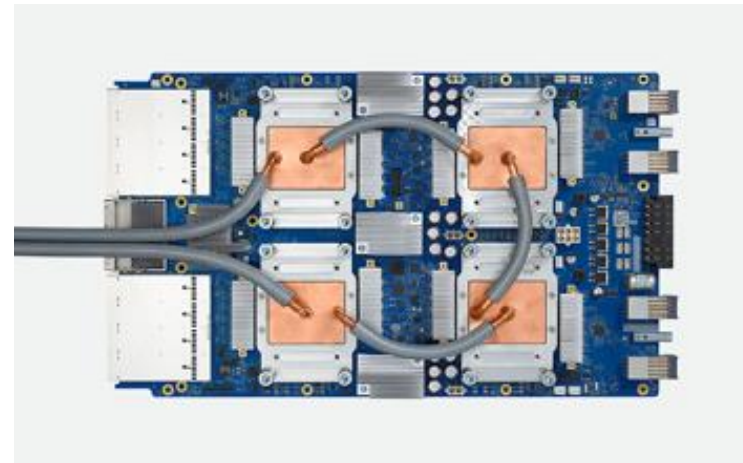
**Especializados** para acelerar cálculo tensorial (álgebra lineal).

Uso desde **TensorFlow y PyTorch**

En fase **beta**:

- De Gigas a Tera Bytes de memoria.
- De cientos a miles de núcleos.

Disponible en Cloud: 8\$/hora





# Índice

---

1. Introducción a la regularización.
2. Redes neuronales convolucionales.
3. Ejercicio 1: nuestra primera red neuronal convolucional con Keras.
4. Hardware para Deep Learning.
5. Algunas redes neuronales profundas y transferencia de aprendizaje.
6. Ejercicio 2: transfer learning con Keras.

# Clasificación de objetos

Tareas en **visión por computador**

**Clasificación**



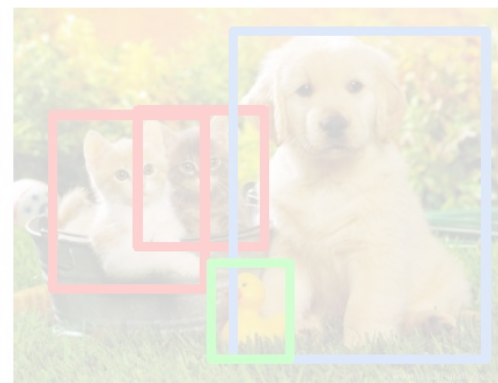
CAT

Clasificación  
y localización



CAT

Detección de  
objetos



CAT, DOG, DUCK

Segmentación  
de instancias



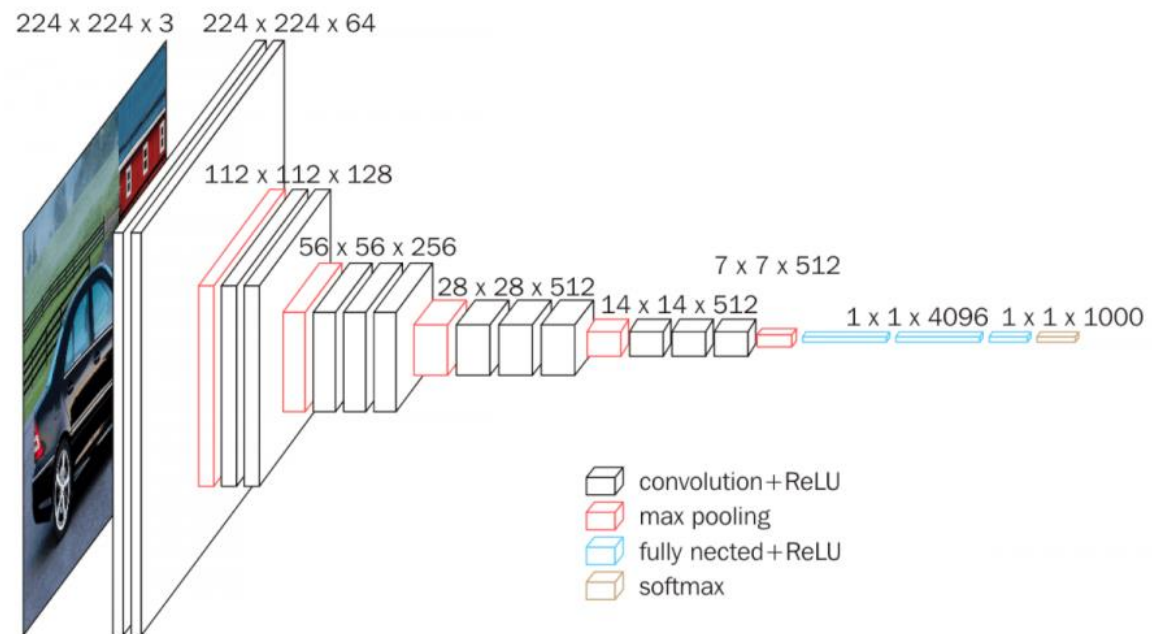
CAT, DOG, DUCK

Single object

Multiple objects

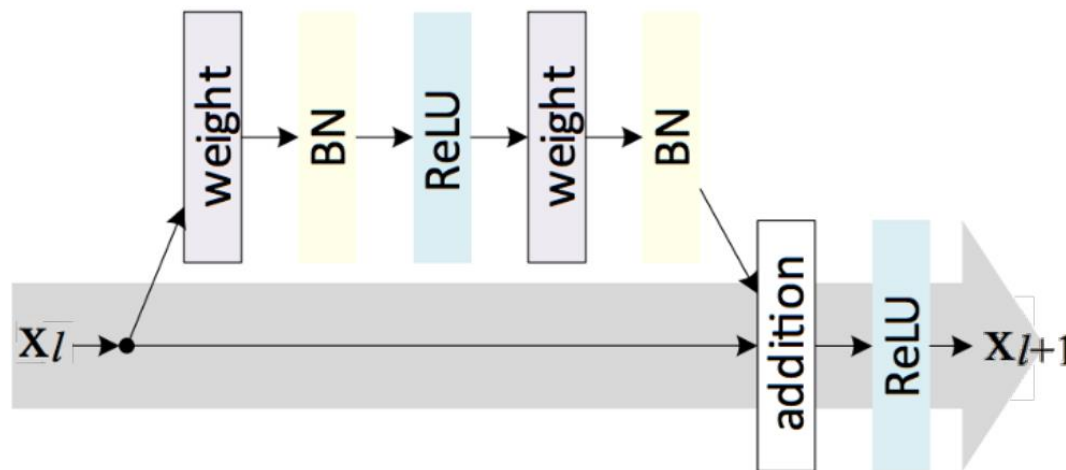
# VGG

- Versión **VGG16** y **VGG19** (con 16 y 19 capas)
- Convolución 3x3 (con efectividad de 5x5)
- Pooling 2x2



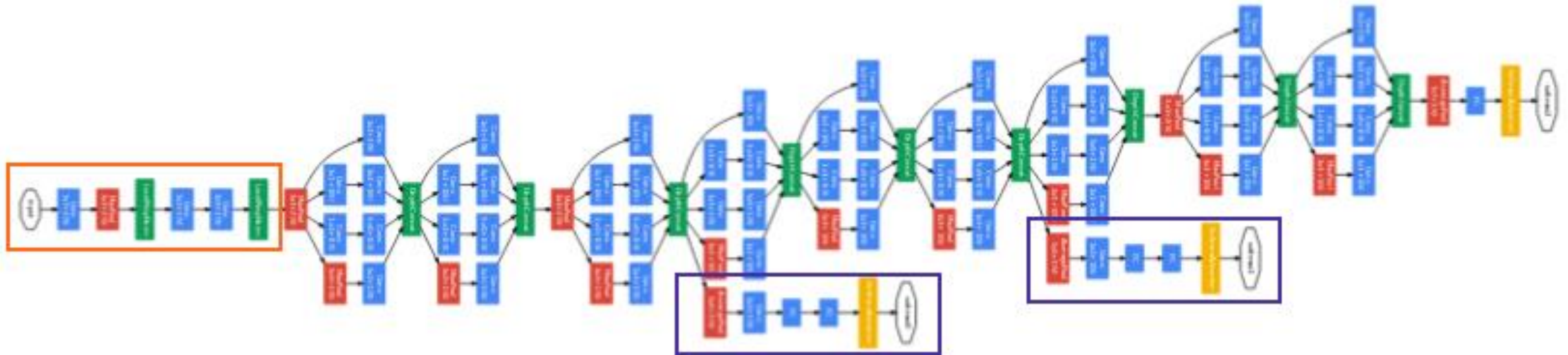
# ResNet

- Desarrollado en Microsoft Research en 2015
- Variantes: **ResNet20**, **ResNet32**, **ResNet56**...



# Inception

- Desarrollado en Google
- Versiones: GoogLeNet (Inception v1), v2, v3, v4...



# Comparativa

| Year | CNN                | Developed by                                     | Place | Top-5 error rate | No. of parameters |
|------|--------------------|--|-------|------------------|-------------------|
| 1998 | LeNet(8)           | Yann LeCun et al                                 |       |                  | 60 thousand       |
| 2012 | AlexNet(7)         | Alex Krizhevsky, Geoffrey Hinton, Ilya Sutskever | 1st   | 15.3%            | 60 million        |
| 2013 | ZFNet()            | Matthew Zeiler and Rob Fergus                    | 1st   | 14.8%            |                   |
| 2014 | GoogLeNet(19)      | Google   | 1st   | 6.67%            | 4 million         |
| 2014 | VGG Net(16)        | Simonyan, Zisserman                              | 2nd   | 7.3%             | 138 million       |
| 2015 | <u>ResNet(152)</u> | Kaiming He                                       | 1st   | 3.6%             |                   |

# Detección de objetos

Tareas en **visión por computador**

Clasificación



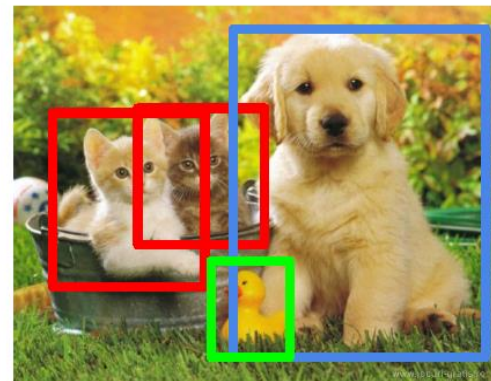
CAT

Clasificación  
y localización



CAT

**Detección de  
objetos**



CAT, DOG, DUCK

Segmentación  
de instancias



CAT, DOG, DUCK

Single object

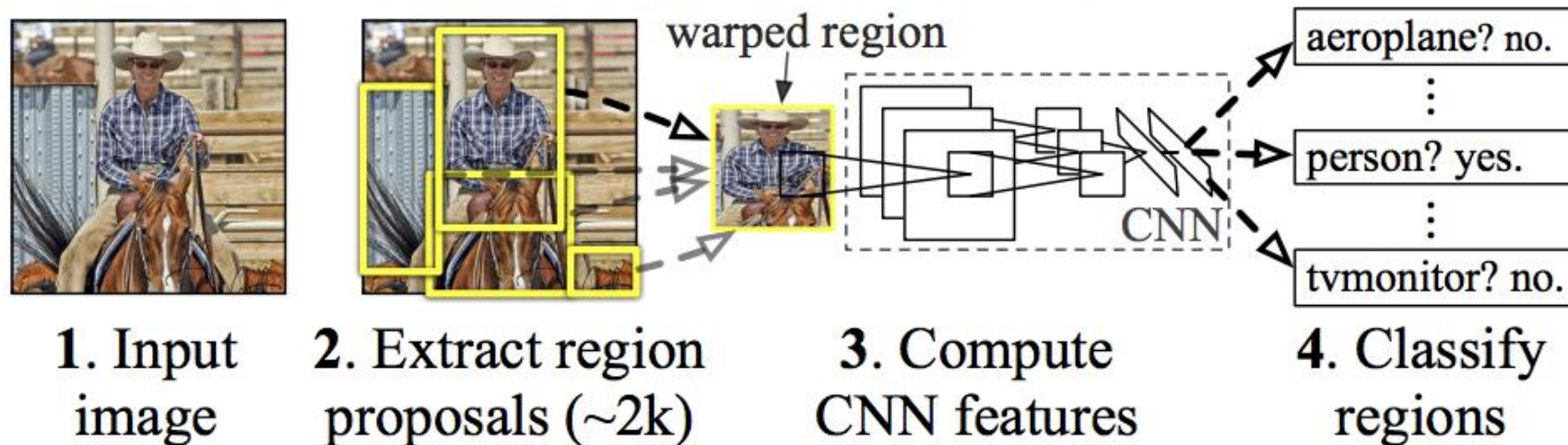
Multiple objects



# R-CNN

- **R-CNN** (2014): familia de modelos creados en Microsoft Research
- **Fast R-CNN** (2015)
- **Faster R-CNN** (2016)
- **MASK R-CNN** (segmentación de regiones)

## **R-CNN: *Regions with CNN features***

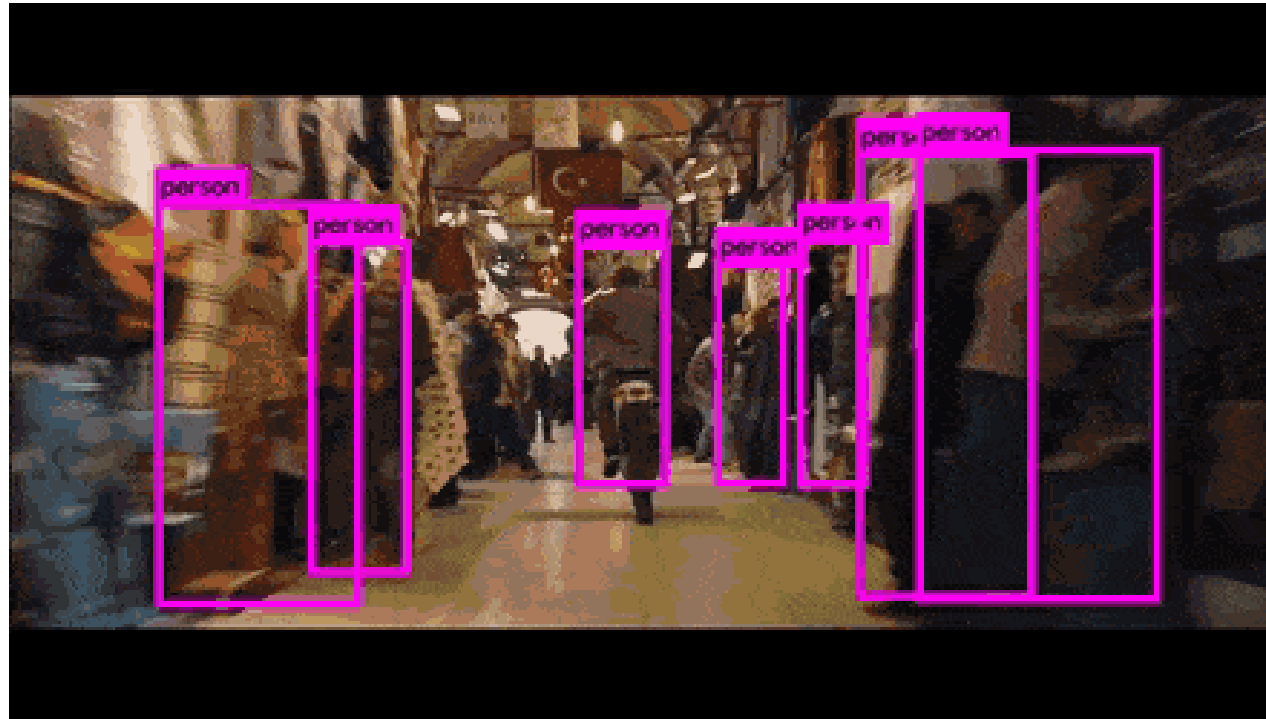




# YOLO (*You Only Look Once*, 2015)

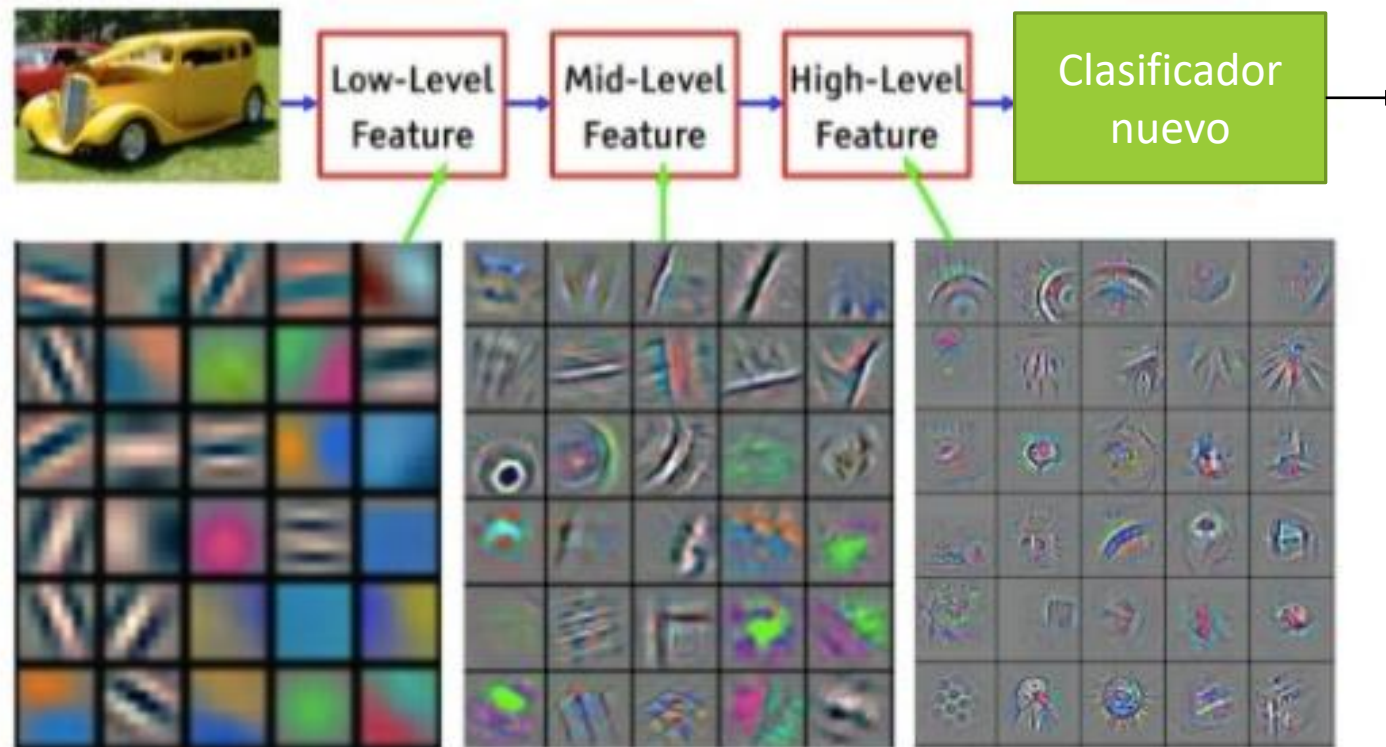
---

- Más rápido que los R-CNN (hasta tiempo real), pero menos preciso



# Transfer Learning en Deep Learning

## Convolutional Neural Network



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

# Ejemplo: Tag extraction

(extracción de etiquetas)

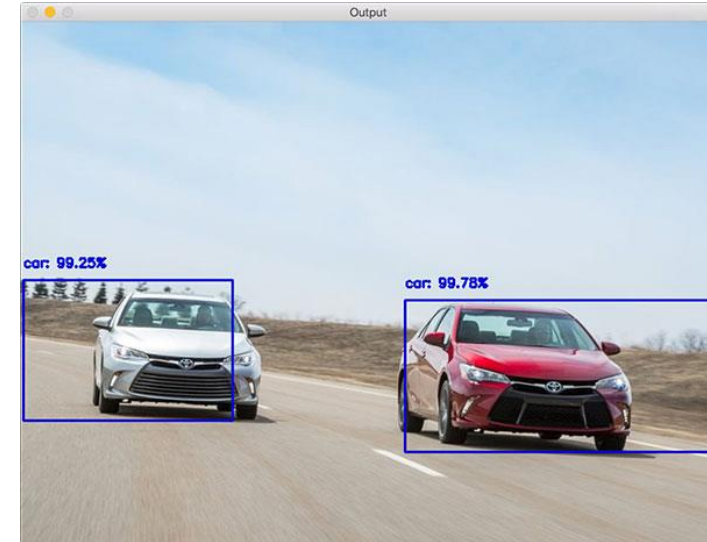
**Extracción** (automática) de etiquetas de películas:

- *Metadatos imprecisos* generado por humanos
- Extracción de *información saliente* usando machine learning
- *Semántica* de “alto nivel”

**Idea:** selección de etiquetas clave, representando el tema general

**Diferente** a la mayoría de tareas de reconocimiento de objetos o escenas:

- No nos interesa si en un vídeo aparece una escopeta
- Nos interesa saber si el video es de violencia, acción, etc.



**Object detection:** 2 cars

**vs**

**Movie tag:** car chase

# Ejemplo: Tag extraction

## Diseño conceptual

---

### **Problemática:**

- No existe un dataset para etiquetas
- Confección de uno desde cero y de forma manual
- No disponíamos de recursos computacionales ni de mucho tiempo (beca posdoctoral).

### **Solución:** Transfer Learning para Fixed Feature Extractor

- Necesitamos un dataset “mediano”
- Inception-v3: Eliminada última capa, y añadida una de Dropout, ReLu y Softmax.

# Ejemplo: Tag extraction

## Conjunto de datos

- Vocabulario 50 etiquetas (con solapamiento)
- 700 imágenes/etiqueta

|             |                |                |
|-------------|----------------|----------------|
| Action      | Bomb explosion | Car chase      |
| Destruction | Sword fight    | Vehicle crash  |
| Violence    | Abduction      | Heist          |
| Adventure   | Animal         | Beach/Sea      |
| Climbing    | Desert         | Hiking         |
| Forest      | Valleys/Hills  | Children       |
| Family      | Club/Bar       | Dance          |
| Music       | Wedding        | College/Univ.  |
| Hospital    | Drinking       | Food           |
| Smoking     | Exercise       | Sports         |
| Swimming    | Glamor/Fashion | Nudity         |
| Romance     | Sex            | Horror         |
| Monster     | Murder         | Lab Experiment |
| Sci-fi      | Super hero     | Technology     |
| Robot       | Military       | Police         |
| Prison      | War            | Weapon         |
| Animation   | Drama          |                |



# Ejemplo: Tag extraction

## Resultados en fotogramas individuales



Military, action, weapon, war



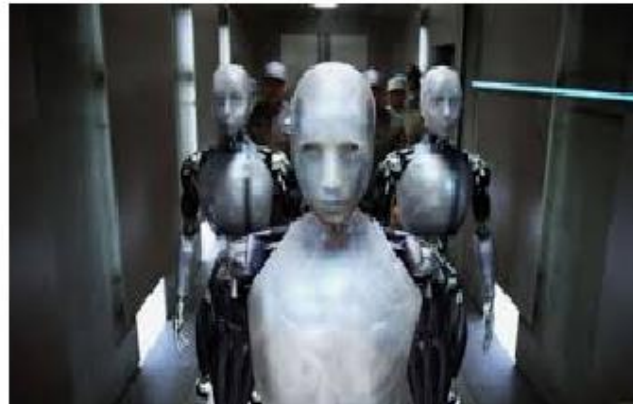
Violence, destruction, bomb explosion, action, car crash



Sex, nudity, romance, modeling



Hiking, adventure, nature, forest, valleys, hills, climbing



Sci-fi, super hero, robot, action



Violence, sci-fi, action, horror

# Ejemplo: Tag extraction

## Experimentación

### Problemática:

- **No hay un ground truth**, o marco de referencia

### Realización de **3 experimentos subjetivos**:

- Llevados a cabo en el cine del Fraunhofer IIS
- Muestra de 10 tráilers de películas
- 10 voluntarios distintos en cada uno

### Experiments:

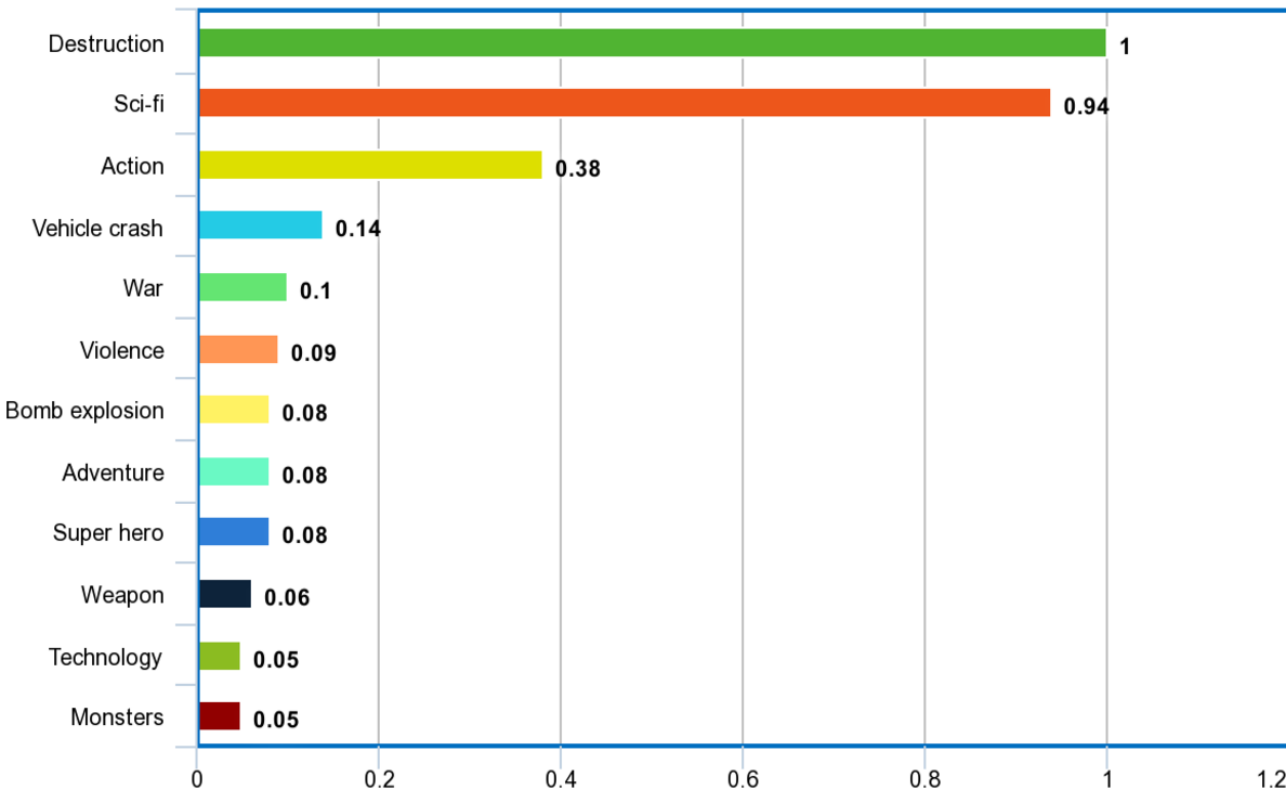
- Tags rating: Mean Opinion Score of **84.3%**
- Tags rating w.r.t. relevancy and strength: Mean Opinion Score of **77.8%**
- Tags matching (used as ground truth): F1 - Score = **0.75%**



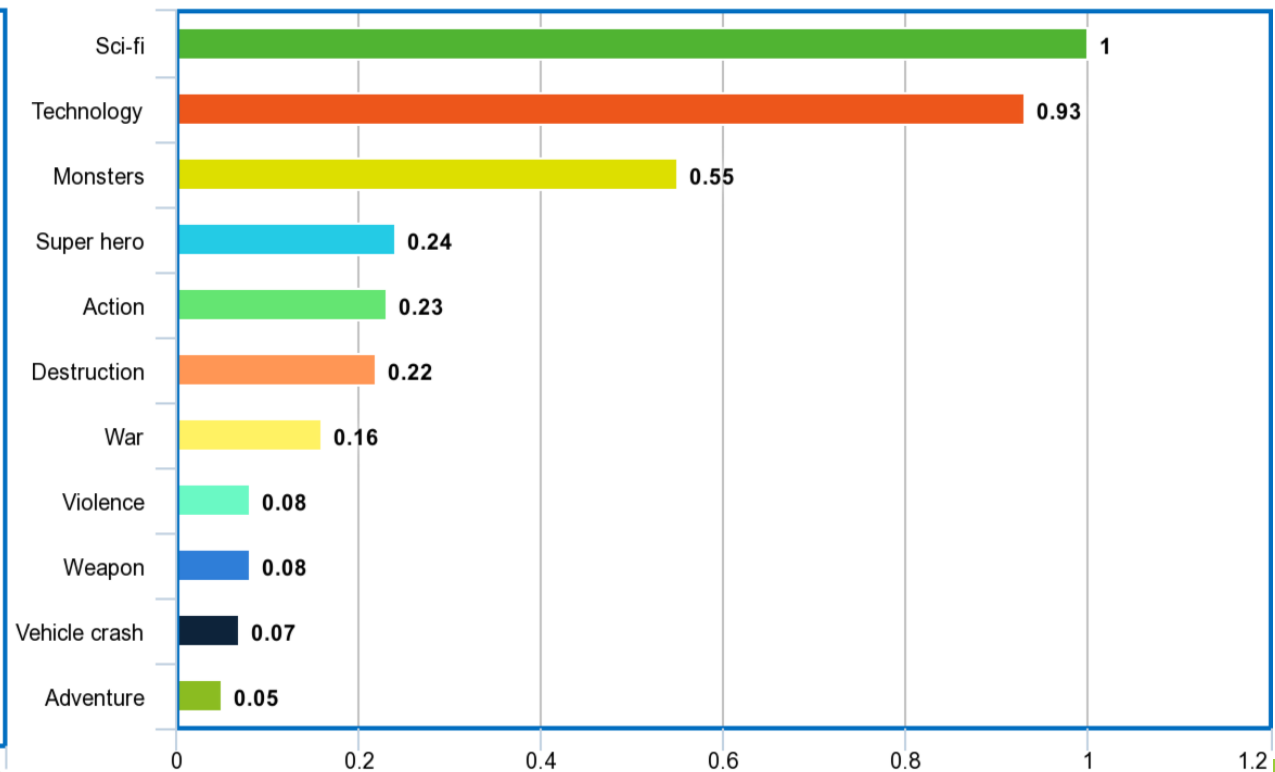
# Ejemplo: Tag extraction

Experimentación (tráiler vs película completa)

- **Trailer duration:** 2 min, 26 sec, **Processing time:** 17sec
- **Full length movie duration:** 1 hr, 24 min, **Processing time:** 10 min



The Guardians (2017) Trailer



The Guardians Full Length Movie



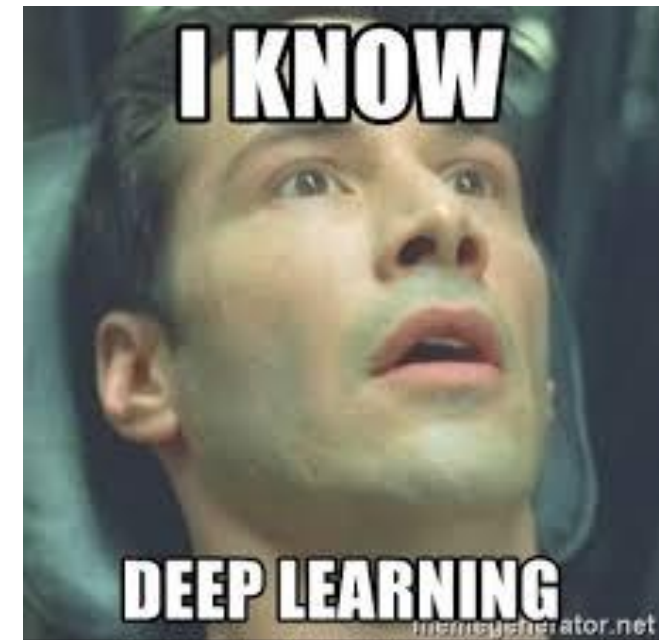
# Índice

---

1. Introducción a la regularización.
2. Redes neuronales convolucionales.
3. Ejercicio 1: nuestra primera red neuronal convolucional con Keras.
4. Hardware para Deep Learning.
5. Algunas redes neuronales profundas y transferencia de aprendizaje.
6. Ejercicio 2: transfer learning con Keras.

# Fin... por ahora...

---



# Stay tuned

---

- Ponerte en contacto con IEEE SBUS: [ieeesbus@gmail.com](mailto:ieeesbus@gmail.com)
- Listas de distribución interesantes:
  - GPU computing en la US: <https://listasvol.us.es/mailman/listinfo/gpucomputing>
  - IA+ML en la US: <https://listas.us.es/mailman/listinfo/iaml>
- Mismos recursos que en la sesión 1