

Deep Learning

Sesión 1 - Resumen

De un vistazo

- Diferencias entre IA, ML y DL
- Contexto histórico
- Google Colab
- Nuestro héroe: el Descenso del Gradiente
- Tensores y usos
- Tensorflow
 - Tensores, operaciones y grafos
 - Sesiones
 - Variables y placeholders
 - Tensorboard
- Carga de un dataset externo

Diferencias entre IA, ML y DL

- **Inteligencia Artificial:** la disciplina que estudia cómo conseguir que las máquinas realicen tareas propias de humanos. El gran paraguas en el que se engloba la IA tradicional, el Machine Learning y el Deep Learning.
 - Narrow: tareas específicas, gran auge en los últimos 10 años
 - General: dotar a las máquinas de conocimiento general, como Terminator. Aún no ha llegado.
- **Machine Learning:** cambio de paradigma de “reglas” a “datos”. Aprovechamos que tenemos muchos datos para crear algoritmos que los usen para crear las reglas de clasificación/regresión.
 - Necesita nuestra implicación para elegir qué datos vamos a utilizar en nuestros algoritmos
- **Deep Learning:** añadimos complejidad a los algoritmos para que sean capaces de encontrar las características a utilizar por sí solos. Necesitamos aún más datos y más potencia de cálculo: GPUs.

Contexto histórico

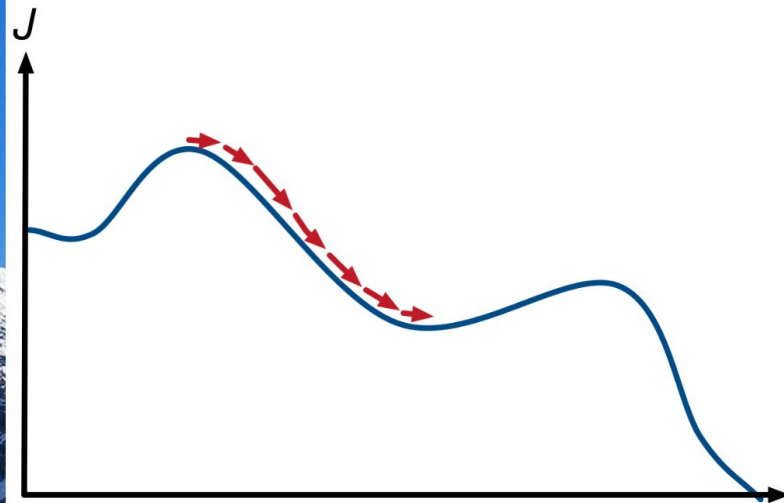
- En la IA se han producido varios “veranos” e “inviernos” debido al optimismo/pesimismo. Ahora mismo, **GRAN VERANO**, y en mi opinión aún queda MUCHO para volver a ver un invierno, pero ya sabéis... winter is coming.
- Grandes hitos de la historia de la IA:
 - 1950: Test de Turing
 - 1952: primer programa que **APRENDE**
 - 1956: se acuña el término Artificial Intelligence
 - 1958: se diseña el **PERCEPTRÓN**
 - 1967: aparece el algoritmo de Nearest Neighbor
 - 1981: aparece el precursor del ML, conocido como Explanation Based Learning
 - 1990s: cambia el enfoque de orientado al conocimiento a orientado al dato
 - **2006: aparecen las arquitecturas profundas (Deep Learning)**
 - ...

Google Colab

- Proporciona un Jupyter Notebook con GPU/TPU GRATIS
- Es necesario indicar que queremos utilizar la GPU/TPU
- Dependiendo de la carga en ese momento, obtendremos mejores o peores recursos
- Tiene límites en cuanto a memoria y tiempo de ejecución
- Podemos cargar los datos que queramos de cualquier forma
 - Tenemos línea de comandos si anteponemos “!” a la orden en la celda
 - Comandos mágicos de Jupyter Notebook
 - Conexión con GDrive
- No indicado para desarrollo profesional (considerar AWS o Google Cloud)

El Descenso del Gradiente

- Método de optimización de funciones



Tensores y usos

- Contenedores de números
- Son el alimento de nuestros algoritmos
- Podemos almacenar desde datos tabulares hasta videos
- Es necesario que tengáis soltura con el manejo de tensores: aquí tenéis un buen recurso para adquirirla

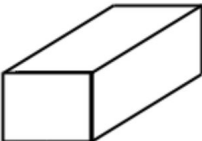
Vector
(Tensor 1D)

-1
2
7
19
-5
0.5
1.9

Matriz
(Tensor 2D)

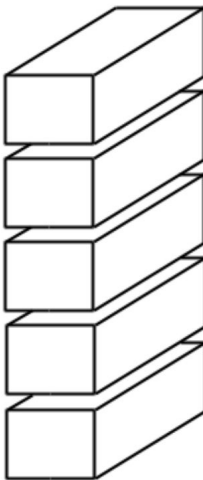
-1	-5	84	5
2	0.5	56	7
7	1.9	1	8.4
19	6	8	0.3

Matriz 3D
(Tensor 3D)

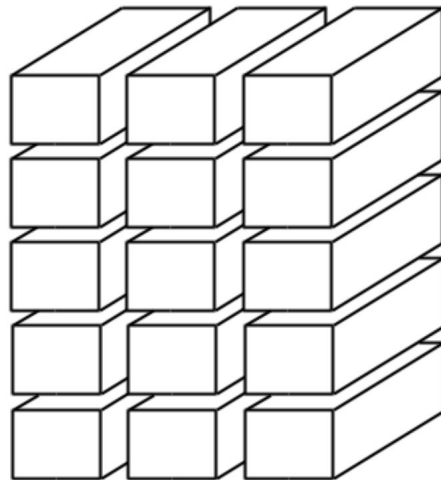


-1	-5	84	5
2	0.5	56	7
7	1.9	1	8.4
19	6	8	0.3

Vector de matrices 3D
(Tensor 4D)



Matriz de matrices 3D
(Tensor 5D)



Tensorflow

Los componentes básicos de TensorFlow son:

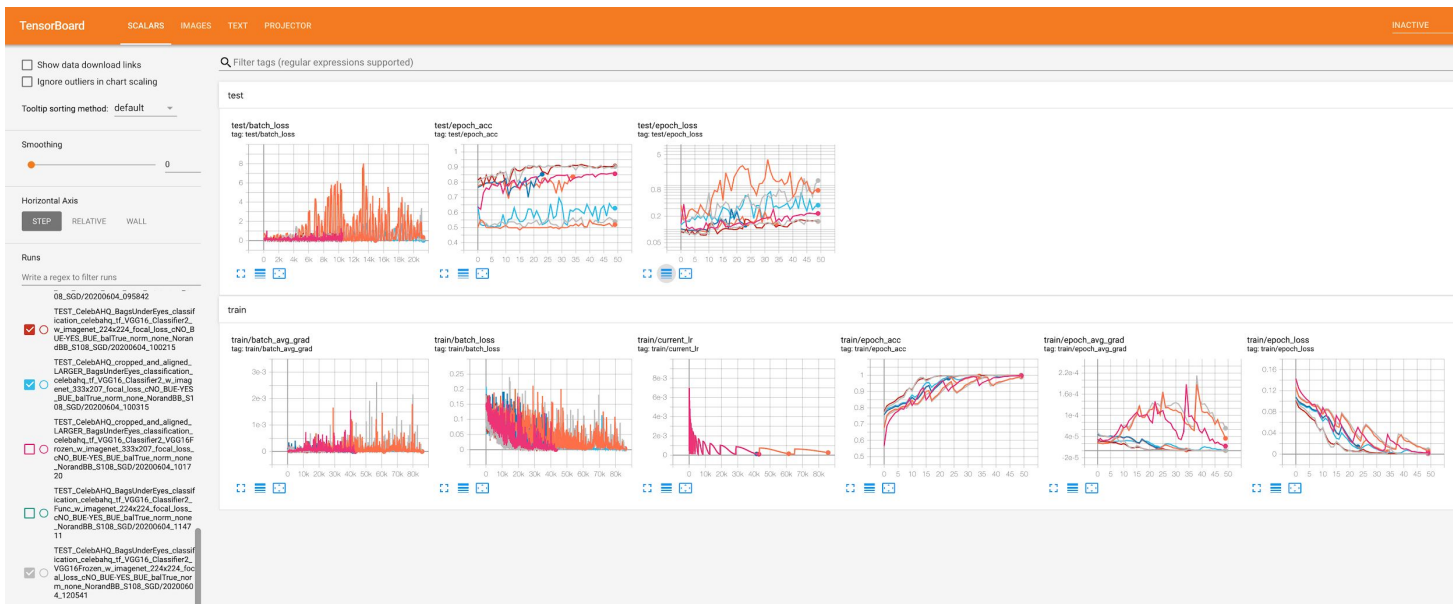
- Los **tensores**: donde almacenaremos la información. Con ellos realizaremos...
- Las **operaciones**: como su propio nombre indica, las operaciones matemáticas. En TF 2.x se comportan como en numpy, en TF 1.x son definiciones que se añaden a...
- Los **grafos**: donde se añaden las operaciones para ser ejecutadas en...
- Las **sesiones**: en TF 1.x o TF 2.x (sin eager mode) son necesarias para poder ejecutar operaciones

Tensorflow

- Las **variables** son similares a los tensores, pero permiten su diferenciación para ser optimizadas, mientras que los tensores no. Así, TF será capaz de calcular el gradiente de una variable y optimizarla (los pesos de nuestra red), pero no de un tensor.
- Los **placeholders** se usan sobretodo en TF 1.x y sirven para indicar que en tiempo de ejecución, ahí habrá una estructura de datos de las dimensiones y tipo definidas

Tensorflow

- Es una herramienta muy útil de monitoreo del proceso de entrenamiento. Permite visualizar curvas de pérdidas/accuracy, histogramas de distribución de variables o tensores, imágenes, grafos, etc.



Carga de un dataset externo

- Descargar el dataset con `!wget -O nombre_archivo.ext URL`
- Descomprimirlo con `!unzip nombre_archivo.ext`
- Cargarlo y hacer el preprocesamiento necesario:

```
# método lento
images = []
for img in list_imgs:
    images.append(load_image(img))
images = np.stack(images) # convertimos la lista en un array de tensores

# método rápido
n_images = len(list_imgs)
images = np.zeros((n_images, height, width, channels), dtype=np.uint8)
for i, img in enumerate(list_imgs):
    images[i] = load_image(img)
images = np.stack(images) # convertimos la lista en un array de tensores
```

Carga de un dataset externo

- Con preprocesamiento:

```
from tqdm import tqdm
images = []
for i, img_path in enumerate(tqdm(list_imgs)):
    img = cv2.imread(img_path)    # cargamos la imagen
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)    # la convertimos a RGB (OpenCV carga como
BGR)
    img = cv2.resize(img, dsize=( 96, 96))    # la redimensionamos
    # aquí incluiríamos los pasos que quisiésemos, por ejemplo:
    # img = img / 255.    # normalizamos la imagen entre 0 y 1
    # img = mi_funcion_random_crop(img)    # cortamos un trozo aleatoriamente
    images.append(img)
images = np.stack(images)    # convertimos la lista en un array de tensores

print('Loading completed!')
```

Carga de un dataset externo

- Recordad que este paso es el más importante de todos (regla del 80% del tiempo en los datos, 20% del tiempo en el modelo)
- Tenemos que ir con mucho cuidado de cargar las imágenes y las etiquetas correctamente
- También de normalizar las imágenes entre unos valores (normalmente 0 y 1)
- Y en definitiva, debemos asegurarnos de que los datos quedan exactamente como nosotros queremos

**!!!MUCHAS VECES LOS MODELOS NO FUNCIONAN PORQUE COMETEMOS
ALGÚN ERROR EN EL PASO DE PRE-PROCESAMIENTO Y PROCESAMIENTO DE
DATOS!!! ¡¡¡PRESTAD MUCHA ATENCIÓN A ESTE PASO!!!**