Práctica 1

Aprendizaje Automático

Redes de Neuronas Convolucionales.

Enunciado de la práctica

El objetivo de esta práctica es diseñar, configurar y entrenar un modelo Redes de Neuronas **Convolucionales**.

- La práctica se realizará en grupos de 2 personas.

- El entregable será el fichero Jupyter Notebook con el que habréis estado trabajando en la práctica. Ahí estará tanto el código como las explicaciones en formato Markdown embebido. Tenéis un útil manual de Markdown en https://colab.research.google.com/notebooks/markdown_guide.ipynb. Os aconsejo que documentéis todo lo que hagáis, aunque sean pruebas que no os hayan salido bien. El objetivo es que contéis una historia. El Jupyter Notebook deberá comenzar con tres líneas:

    - GRUPO NºGRUPO
    - Nombre y Apellidos del primer integrante del grupo (en cuyo GitHub está el repositorio con la entrega)
    - Nombre y Apellidos del segundo integrante del grupo

- La práctica deberá subirse a un repositorio de vuestra cuenta personal de GitHub llamada AA_PRACTICA2_GRUPO_NºGRUPO. (Por ejemplo, si sois el grupo 3, vuestra entrega estará en el repositorio de GitHub de alguno de vosotros dos con el nombre AA_PRACTICA2_GRUPO_3). Tenéis que usar el mismo número de grupo que el que usasteis durante la practica 1.

- El nombre de este repositorio deberá estar en el entregable entregado en CANVAS

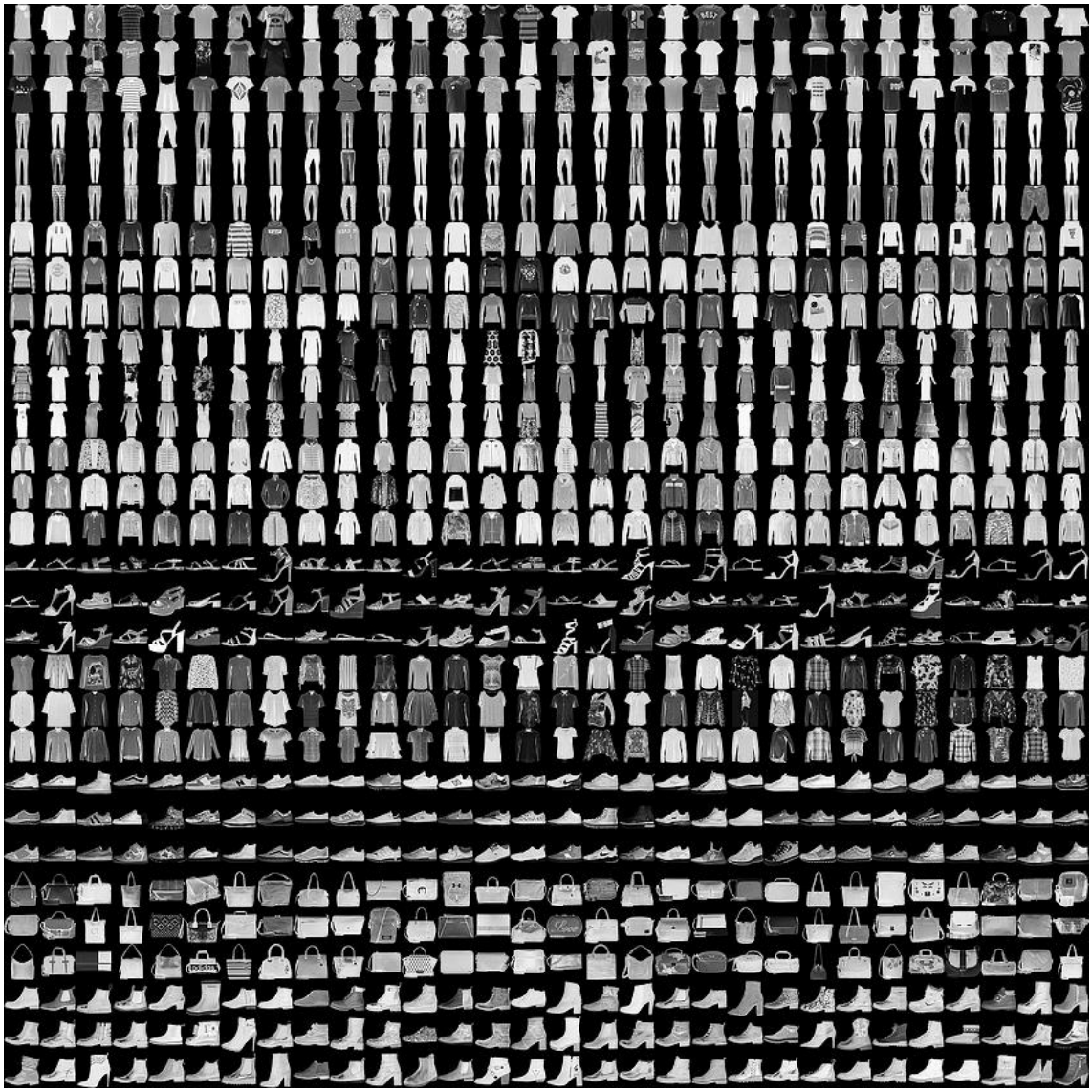
- No sólo se evaluará el resultado final sino todo el proceso hasta completarla, la documentación aportada donde se justificarán todas las decisiones de diseño y se explicaciones detalladas de los resultados y su razonamiento.

- Se evaluará positivamente todo el contenido adicional a la asignatura contenido en la práctica, siempre que guarde relación y aporte valor al objetivo de esta.

- Es importante demostrar todo el conocimiento adquirido durante las clases teóricas.

Enunciado

Crea un modelo de Red de Neuronas **Convolucionales** que sea capaz de **reconocer** y **clasificar imágenes de ropa** en sus diferentes **tipologías**. Este modelo será **definido**, **configurado**, **entrenado**, **evaluado** y **mejorado** para posteriormente usarlo para hacer **predicciones**.
Para ello tendréis que crear un modelo en **Keras** aplicando de una tirada todos los pasos al conjunto de datos **Fashion-MNIST**, precargado en Keras y que ya habéis utilizado para la práctica 1.
Fashion-MNIST es un conjunto de datos de las imágenes de los artículos de Zalando (www.zalando.com), una tienda de moda online alemana especializada en ventas de ropa y zapatos. El conjunto de datos contiene 70K imágenes en escala de grises en 10 categorías. Estas imágenes muestran prendas individuales de ropa en baja resolución (28 x 28 píxeles):



Se usan 60K imágenes para entrenar la red y 10K imágenes para evaluar la **precisión** con la que la red aprende a clasificar las imágenes

Cuestiones a tener en cuenta

- Antes de empezar a programar vuestra red neuronal deberéis importar todas las librerías que vais a requerir.
- Aseguraos que estáis ejecutando la versión 2.0.0 (o superior) de TensorFlow en vuestro Google Colab
- Cargar los datos de entrenamiento y de Test a partir de *keras.datasets.fashion_mnist*
- La clasificación corresponde, según el código numérico de clase, a:

Label	Class
0	T-shirt/top
1	Trouser
2	Pullover
3	Dress
4	Coat
5	Sandal
6	Shirt
7	Sneaker
8	Bag
9	Ankle boot

- Es una buena práctica analizar si los datos tienen la forma esperada
- Durante una posible fase de pre procesado de datos, analizar el uso de la función de keras keras.layers.Flatten()

Cuestiones para implementar y responder

1. Configurar y entrenar los siguientes modelos de red de neuronas convolucionales, analizando y reflexionando sobre los resultados:

CASO 1	
Número de filtros 1ª capa convolucional	**32**
Tamaño ventana 1ª capa convolucional	**5 x 5**
Función activación 1ª capa convolucional	**relu**
Ventana 1ª capa pooling	**2 x 2**
Número de filtros 2ª capa convolucional	**64**
Tamaño ventana 2ª capa convolucional	**5 x 5**
Función activación 2ª capa convolucional	**relu**
Ventana 2ª capa pooling	**2 x 2**
Capa Flatten	

Función activación ultima capa Densa	**softmax**
Optimizador	**sgd**
Función de Pérdida	**sparse_categorical_crosentropy**
Métrica	**accuracy**
Número de iteraciones	**5**

CASO 2	
Número de filtros 1ª capa convolucional	**64**
Tamaño ventana 1ª capa convolucional	**7 x 7**
Función activación 1ª capa convolucional	**relu**
Padding	**same**
Ventana 1ª capa pooling	**2 x 2**
Número de filtros 2ª capa convolucional	**128**
Tamaño ventana 2ª capa convolucional	**3 x 3**
Función activación 2ª capa convolucional	**relu**
Padding	**same**
Ventana 2ª capa pooling	**2 x 2**
Capa Flatten	
Función activación penult. capa densa	**65 neuronas ReLU**
Función activación ultima capa densa	**softmax**
Optimizador	**sgd**
Función de Pérdida	**sparse_categorical_crosentropy**
Métrica	**accuracy**
Número de iteraciones	**5**

CASO 3	
Número de filtros 1ª capa convolucional	**64**
Tamaño ventana 1ª capa convolucional	**7 x 7**
Función activación 1ª capa convolucional	**relu**
Padding	**same**
Ventana 1ª capa pooling	**2 x 2**
Número de filtros 2ª capa convolucional	**128**
Tamaño ventana 2ª capa convolucional	**3 x 3**
Función activación 2ª capa convolucional	**relu**
Padding	**same**
Ventana 2ª capa pooling	**2 x 2**
Capa Flatten	
Función activación penult. capa densa	**65 neuronas ReLU**
Función activación ultima capa densa	**softmax**

Optimizador	**adam**
Función de Pérdida	**sparse_categorical_crosentropy**
Métrica	**accuracy**
Número de iteraciones	**5**

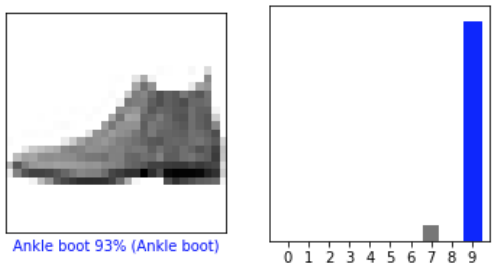
2. Explicar la salida de la llamada `model.summary()` de cada uno de los 3 casos
3. Analizar e interpretar los resultados del caso 1 frente a su original si se multiplica por 5 las épocas de entrenamiento (25)
4. Analiza el resultado del caso 1 si en lugar de ReLU usas tanh en la función de activación de las dos capas convolucionales.
5. Evaluar cada uno de los 3 modelos comparando el rendimiento del modelo en el conjunto de datos de prueba
6. Usar cada uno de los 3 modelos para hacer predicciones sobre la 6ª imagen de test (`test_images[5]`)
7. Utilice el siguiente código para graficar cómo de bien o de mal se comporta el modelo para cada uno de los 3 casos con las 14 primeras imágenes del conjunto de test. Reflexione y comente las diferencias que observa.

```
def plot_image(i, predictions_array, true_label, img):
  predictions_array, true_label, img = predictions_array, true_label[i], img[i]
  plt.grid(False)
  plt.xticks([])
  plt.yticks([])

  plt.imshow(img, cmap=plt.cm.binary)

  predicted_label = np.argmax(predictions_array)
  if predicted_label == true_label:
    color = 'blue'
  else:
    color = 'red'

  plt.xlabel("{} {:2.0f}% ({})".format(class_names[predicted_label],
                                       100*np.max(predictions_array),
                                       class_names[true_label]),
                                       color=color)
```

```
def plot_value_array(i, predictions_array, true_label):
  predictions_array, true_label = predictions_array, true_label[i]
  plt.grid(False)
  plt.xticks(range(10))
  plt.yticks([])
  thisplot = plt.bar(range(10), predictions_array, color="#00FF00")
  plt.ylim([0, 1])
  predicted_label = np.argmax(predictions_array)

  thisplot[predicted_label].set_color('red')
  thisplot[true_label].set_color('black')
```

Ejemplo de uso para ver la predicción de la 6ª figura de las imágenes de test.

```
i = 0
plt.figure(figsize=(6,3))
plt.subplot(1,2,1)
plot_image(i, predictions[i], test_labels, test_images)
plt.subplot(1,2,2)
plot_value_array(i, predictions[i],  test_labels)
plt.show()
```



8. Hacer comparativa con los resultados que obtuvisteis en la práctica 1, con las capas Dense.
9. Ver los conceptos de batch_normalization y dropout y ver si se podría mejorar el modelo con ello.
   https://keras.io/api/layers/normalization_layers/batch_normalization/
   https://keras.io/api/layers/regularization_layers/dropout/
10. Ver los conceptos de callbacks (Decaimiento del ratio de aprendizaje) y ver si se podría mejorar el modelo con ello.
    https://www.tensorflow.org/api_docs/python/tf/keras/callbacks/LearningRateScheduler y https://keras.io/api/callbacks/

Memoria

- Es necesario generar una memoria en la que recogeréis los distintos pasos que habéis realizado para resolver la práctica.

- La memoria debe estar correctamente redactada, sin faltas de ortografía y contener puntos como la introducción y las conclusiones, entre otros.

- Debéis explicar cómo habéis afrontado la solución de la práctica, cómo se ha dividido el trabajo y cómo se ha gestionado el equipo para trabajar en ella.

- Podéis adjuntar imágenes en la que deseáis mostrar algún comportamiento particular o relevante que os haya surgido.

IMPORTANTE: Todo esto, como hemos comentado anteriormente, estará integrado en el Jupyter Notebook dentro de Google Colab

Entregables

El único entregable subir a la tarea de Canvas es el siguientes:

- Vuestro usuario de GitHub (el de la persona del grupo de 2 personas a través del cual queráis tener el código de la práctica) diciendo a que grupo pertenecéis

La fecha tope de entrega de esta práctica es el día 26 de noviembre a las 23:59.

Rúbrica de evaluación

La siguiente rúbrica será para evaluar la práctica anteriormente descrita. El **desarrollo/código es el 40%** y la **memoria con las reflexiones y conclusiones son el 60%** del total de la práctica.

Cuestiones para implementar (desarrollo/código) 40%

	VALORACIÓN			
DIMENSIÓN	**0 puntos**			**10 puntos**
Diseña, configura y optimiza los 3 modelos de redes neuronales planteados	Le falta por diseñar, configurar u optimizar al menos 1 modelo.	Le falta alguna fase por completar a no más de 2 modelos de Redes de Neuronas	Utiliza las funcionalidades justas de forma correcta para todos los modelos	Utiliza todas las funcionalidades estudiadas de forma correcta para todos los modelos
Hace las predicciones solicitadas para cada uno de los modelos	Le falta predecir en al menos 2 modelos.	Le falta predecir en al menos 1 modelos.		Hay predicciones para los 3 modelos
Grafica el comportamiento de los modelos	Le falta graficar en al menos 2 modelos.	Le falta graficar en al menos 1 modelos.		Hay gráficos de comportamiento para los 3 modelos

Memoria 60%

	VALORACIÓN			
DIMENSIÓN	**0 puntos**			**10 puntos**
Redacta adecuadamente la memoria del proyecto	La memoria contiene numerosas faltas de ortografía o gramaticales, con una expresión poco formal, que dificulta su entendimiento	La memoria no expresa con un lenguaje propio del ámbito de conocimiento los conceptos clave	La memoria expresa con lenguaje propio del ámbito de conocimiento los conceptos clave, pero contiene algunos errores ortográficos y gramaticales	La memoria está libre de errores ortográficos o gramaticales y expresa con lenguaje propio del ámbito de conocimiento los conceptos
Detalle del contenido de la memoria	El contenido de la memoria es el visto en clase	La memoria contiene el contenido visto en clase,	La memoria contiene toda la información anterior además de	La memoria contiene todo lo anterior, además de conclusiones por

		complementado con información, y otras secciones investigadas por el alumno.	ventajas/desventajas de los elementos estudiados y/o otras secciones relevantes	cada sección estudiadas y probadas.
Presenta una memoria cuyos contenidos son correctos y coherentes con el proyecto desarrollado	Los diagramas de flujo y las explicaciones no son coherentes con el código.	Existen incoherencias graves entre el código y el contenido de la memoria.	Existen incoherencias entre el código y el contenido de la memoria.	El código y la memoria están perfectamente alineados.
Introduce, explica y concluye adecuadamente y con rigor académico	La introducción no permite contextualizar adecuadamente el trabajo, no está bien explicado el diseño y desarrollo del proyecto y las conclusiones no son relevantes	La introducción no permite contextualizar adecuadamente el trabajo y las conclusiones no son relevantes, pero está bien explicado el diseño y desarrollo del proyecto	La introducción y las explicaciones son relevantes, y están bien argumentadas y ajustadas al proyecto y al ámbito de conocimiento, pero no hay conclusiones o no son relevantes	La introducción, las explicaciones y las conclusiones son relevantes, y están bien argumentadas y ajustadas al proyecto y al ámbito de conocimiento