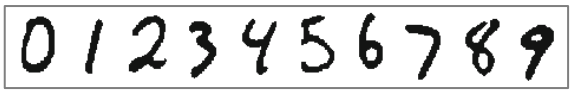
Redes Neuronales

Practica 5 – Redes Neuronales Convolucionales

Material de Lectura: Capítulo 6 del libro Neural Networks and Deep Learning.

1) La base de datos MNIST contiene imágenes de 28×28, en escala de grises, de números escritos a mano. Está conformada por 60.000 ejemplos de entrenamiento y 10.000 ejemplos de prueba.



Para cargar las imágenes utilice:

```
from tensorflow.keras.datasets import mnist
(X_train, Y_train), (X_test, Y_test) = mnist.load_data()
```

Puede visualizar una imagen utilizando:

```
nImg = 0    # nro. de imagen a visualizar
plt.imshow(X_train[0, :,:], cmap='gray')
```

Luego, con el conjunto de 60000 imágenes entrene una red neuronal para predecir el dígito presente en la imagen utilizando un modelo formado sólo por la capa de salida *softmax()* como se indica a continuación. La capa *Flatten()* convierte las imágenes de entrada que originalmente están representada como matrices en vectores para que puedan ser ingresadas a la red que, en este caso, estaría formada sólo por la capa *softmax()*. Verifique que su tasa de acierto es superior al 90%. Recuerde normalizar los valores de cada imagen restándole la media y dividiendo por el desvío.

```
from tensorflow.keras.layers import Dense, Flatten, Input
model = Sequential() model.add(Input(shape=(28,28)))
model.add(Flatten())
model.add(Dense(10, activation='softmax'))
```

2) Pruebe resolver este problema utilizando distintas arquitecturas haciendo variar los siguientes hiperparámetros:
   - Cantidad de capas convolucionales.
   - Cantidad de filtros o kernels de cada capa convolucional.
   - Tamaño de los filtros y maneras en las que serán aplicados (kernel_size, padding y stride).
   - Uso de Max_pooling
   - Cantidad y tamaño de las capas ocultas de la red feedforward (luego de “aplanar”).
   - Funciones de activación de las distintas capas

3) Se buscará resolver la clasificación de los dígitos de MNIST usando la siguiente configuración:

```
model = Sequential()
model.add(Input(shape=(28, 28, 1)))
model.add(Conv2D(F, kernel_size=K, strides=(S,S), activation=FUN))
model.add(MaxPooling2D(pool_size=(2,2)))    #-- opcional --
model.add(Flatten())
model.add(Dense(10,activation='softmax'))
model.summary()
```

donde F es la cantidad de filtros o de mapas de características, K es el tamaño del kernel o máscara, S es el valor del stride y FUN es la función de activación de la capa de convolución.

La tabla que aparece a continuación sugiere los valores a utilizar. Se recomienda emplear Parada Temprana para reducir el tiempo de entrenamiento. Para ello utilice

```
from tensorflow.keras.callbacks import EarlyStopping es
EarlyStopping(monitor='val_accuracy', patience=3, min_delta=0.001)
```

Esto indica que, si el valor del accuracy sobre los datos de validación no mejora después de 3 épocas, el entrenamiento finaliza. Puede usarse el parámetro min_delta para indicar cuando la diferencia entre dos accuracy se considerará significativa. Luego agregue este objeto en el momento del entrenamiento por medio del pa´rametros callbacks

```
H = model.fit(x = X_train, y = Y_train, batch_size = LOTES,
        validation_data = (X_test, Y_test), epochs=4000, callbacks=[es])
```

Capa Convolucional Conv2D				Max Pooling con filtro de tamaño 2x2 y stride=2	Total de parámetros	Épocas	Accuracy en Train	Accuracy en Test
Cant. De filtros	Tamaño del kernel o filtro	Stride	Función de activación					
4	3x3	1	ReLU	Si				
16	3x3	1	ReLU	Si				
64	3x3	1	ReLU	Si				
4	7x7	1	ReLU	Si				
16	7x7	1	ReLU	Si				
64	7x7	1	ReLU	Si				
64	3x3	2	ReLU	No				
64	3x3	3	ReLU	No				
64	3x3	1	TanH	Si				
64	3x3	1	Sigmoide	Si				

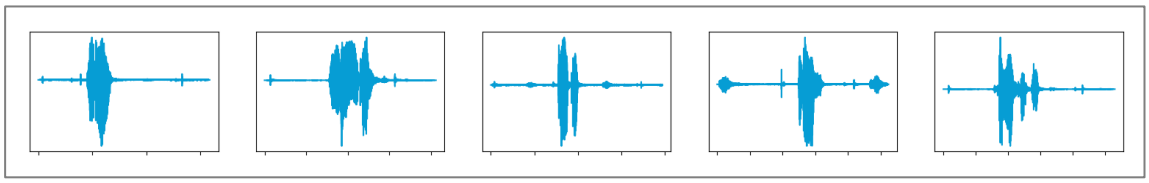
4) Repita un proceso similar al realizado sobre las imágenes MNIST del ejercicio 3, pero ahora el objetivo es reconocer la cantidad de dedos extendidos en cada mano de las imágenes que conforman la base de datos ***Fingers***.

La versión original de este conjunto de imágenes se encuentra en
https://www.kaggle.com/koryakinp/fingers

Puede hallar una versión reducida de estas imágenes en el Moodle del curso, en la misma sección donde se encuentra este enunciado de práctica. También encontrará allí ejemplos sobre cómo cargar estas imágenes y cómo procesarlas con una red neuronal convolucional.

5) Utilizando el juego de datos ***Fingers*** implemente un modelo de red neuronal convolucional que sea robusto a imágenes rotadas en unos 30 grados y tolerantes a un pequeño zoom in/out y translación horizontal y vertical. Utilice para esto el objeto ***ImageDataGenerator*** del módulo ***tensorflow.keras.preprocessing.image***

6) Mozilla Common Voice es un proyecto (https://commonvoice.mozilla.org/es/datasets) que desarrolla una base de datos de voces, abierta y multi idioma, que cualquiera puede usar para entrenar aplicaciones que utilicen la voz como interfaz.
En particular cuenta con un pequeño corpus denominado **Single Word** que contiene las palabras habladas SI, NO, Hey, Firefox y dígitos del cero al nueve.



a) Utilizando los ejemplos de una versión seleccionada de estos audios en el Moodle del curso, transformarlos en imágenes para que puedan ser procesados por la red siguiendo los siguientes criterios:
   - Eliminar los silencios iniciales y finales. La comparación de los silencios en 2 audios produce una falsa coincidencia.
   - Elegir un tamaño fijo para los audios. Todas las imágenes deben tener el mismo tamaño. Como las palabras del corpus son breves un tamaño de 0.75 segundos es razonable.
   - Achicar/agrandar los audios manteniendo el tono de la voz
   - Pasar el audio del dominio del tiempo al domino de la frecuencia. Dividir el audio en pequeños intervalos y obtener un conjunto de valores que determinen las frecuencias presentes. Para esto puede utilizarse la transformada de Fourier (FFT) o los coeficientes cepstrales en frecuencia mel (MFCC) que aproximan la sensibilidad del oído humano.
   - Finalmente convertir los coeficientes en una imagen sin compersión con pérdida como PNG.

b) Utilizando las imágenes generadas en a), entrenar un modelo de red neuronal convolucional que permita reconocer las palabras del corpus Single Word.