DEEP LEARNING PRÁCTICA 3 - 2025

Redes Convolucionales

1. Entrenar y evaluar la performance de una red convolucional que aprende a clasificar la base de datos MNIST. Primero utilizaremos Keras (v3) con tensorflow como backend

   - Bajar la base de datos usando keras.datasets.mnist.load_data()
   - Examinar visualmente los imágenes (x) y compararlas con las clasificacione (y)
   - Rescalear los datos entre 0 y 1
   - Definie la dimensión de entrada y el número de clases
   - Crear un modelo con la sintaxis secuencial usando $keras.Sequential$
   - Agregar capas alternando capas Conv2D y MaxPooling. terminar con GlobalAveragePooling2D y una capa dense
   - Imprimir el modelo con model.summary()
   - Compilar el modelo definiendo la pérdida (CategoricalCrossEntropy), el optimizador (Adam) y la métrica (acc)
   - Definir el número de épocas y el batchsize
   - entrenar el modelo con model.fit
   - Calcular los scores con sobre el conjunto de test con model.evaluate

2. Ahora repitamos el ejercicio usando pytorch

   - Crear la clase NN que hereda los atributos de torch.nn
   - En __init__ definir las capas combinando Conv2D y Linear
   - En forward definir la propagaci'on combinando relu con max_pool2d y finalmente xview para poner la matriz en un vector. La salida final se optiene aplicando log_softmax.
   - Comparar esta sintaxis con la de keras, especialmente el cálculo de dimensiones
   - Definir el batch size, número de classes, learning rate y número de épocas.

- Cargar los datos con torchvision.datasets.MNIST y crear el dataloader
- Instanciar el modelo, definir la funci'on de pérdida y el optimizador.
- Implementar el proceso de aprendizaje con un loop sobre épocas y un loop sobre batches anidados
- En cada paso realizar
  - La propagación para delante
  - El cáculo de la pérdida
  - Poner los gradientes a 0
  - Cálculo de los gradientes
  - Llama al optimizador
  - acumular la pérdida
- Calcular las predicciones sobre los datos de test item Opcionalmente calcular la matriz de confusión usando confusion_matrix y ConfusionMatrixDisplay de sklearn.metrics

3. Entrenar y evaluar la preformance de una red con arquitectura U-net que realza una tarea de segmantación semántica

   - Descargar el directorio oxford-iiit-pet del repositirio. Las imágenes se encuentra en el subdirectorio images/ y los targets en en subdirectorio annotations/trimaps/. Examinar un ejemplo.
   - Crear una función que carga las imágenes por batch haciendo un resizing a tamaño adecuado. Se pueden usar las funciones tf_io.read_file, f_io.decode_png, f_image.resize, f_image.convert_image_dtype.
   - Crear la arquitectura unet usando la sintaxis fincional. Creal un path de bloques convolucionales cada vez mas chicos pero con mas canales, guardando las salidas adecuiadas para ser concatenadas en ek camino de upsampling en la posición adecuada. Calcular las dimensiones en cada paso del oroceso.
   - Enternar el modelo usando sparse_categorical_crossentropy como pérdida.
   - Examinar la performance sobre los datos de valudación
   - Optimizar la arquitectura.