Universidad Nacional de Rosario

Facultad de Ciencias Exactas, Ingeniería y Agrimensura

T.U.I.A

Aprendizaje Automático II

Trabajo Práctico 1

Redes Densas y Convolucionales

2024

Autor/es:

Grupo Nº	
Nombre y Apellido	Nº de Legajo

Corrigió	Calificación

Problema 1

Descripción:

En este problema, se presenta un conjunto de datos que contiene información sobre el rendimiento académico de estudiantes universitarios, así como diversos factores que podrían influir en él. El objetivo es construir un modelo de regresión utilizando redes neuronales para predecir el índice de rendimiento académico de los estudiantes basado en las características proporcionadas.

Dataset:

https://drive.google.com/file/d/1mfpXVLqDJah-sO0CF29LjKUz5NtKjZqc/view?usp=drive_link

El dataset proporcionado incluye las siguientes variables para cada estudiante:

- **Hours Studied**: El número total de horas dedicadas al estudio por cada estudiante.
- **Previous Scores**: Las puntuaciones obtenidas por los estudiantes en exámenes previos.
- **Extracurricular Activities**: Si el estudiante participa en actividades extracurriculares (Sí o No).
- **Sleep Hours**: El número promedio de horas de sueño que el estudiante tuvo por día.
- **Sample Question Papers Practiced**: El número de cuestionarios de muestra que el estudiante practicó.

Además, el dataset incluye la variable objetivo:

- **Performance Index**: Un índice que representa el rendimiento académico general de cada estudiante, redondeado al entero más cercano. Este índice varía de 10 a 100, donde valores más altos indican un mejor rendimiento.

Objetivo:

Utilizando el dataset proporcionado, el objetivo es construir un modelo de regresión utilizando redes neuronales que pueda predecir con precisión el índice de rendimiento académico de los estudiantes. Se debe entrenar y evaluar el modelo utilizando técnicas adecuadas de validación y métricas de evaluación de regresión.

Entrega:

La entrega debe incluir:

Código fuente de la solución implementada en Google Colab, que incluya:

- Análisis previo y preprocesamiento del set de datos.
- Definición y entrenamiento del modelo.
- Resultados de la evaluación del modelo, incluyendo métricas de desempeño y visualizaciones relevantes.

Nota: el código debe estar debidamente documentado con comentarios explicativos para que el trabajo sea fácilmente comprensible para otros revisores.

Problema 2

Descripción:

El objetivo de este ejercicio es implementar un sistema de clasificación de gestos de "piedra", "papel" o "tijeras" utilizando **MediaPipe** para la detección de las manos y una red neuronal densa para realizar la clasificación. El ejercicio se dividirá en tres partes, cada una implementada en un script de Python.

Doc de MediaPipe para detección de landmarks:

https://ai.google.dev/edge/mediapipe/solutions/vision/hand_landmarker

Objetivo:

1. Grabación del dataset de gestos: En esta primera parte, se implementará un script llamado record-dataset.py, que permitirá grabar un dataset de gestos utilizando la cámara web y MediaPipe para detectar los landmarks (puntos clave) de la mano. Cada gesto se etiquetará como "piedra" (0), "papel" (1) o "tijeras" (2) y se almacenará junto con sus coordenadas en archivos .npy.
2. Entrenamiento del clasificador de gestos: En esta segunda parte, se implementará un script llamado train-gesture-classifier.py, donde se entrenará una red neuronal densa utilizando los datos de los landmarks obtenidos en la primera parte. El modelo resultante será capaz de clasificar los gestos basándose en las posiciones de los puntos clave de la mano.
3. Prueba del sistema completo: En la tercera parte, se implementará un script llamado rock-paper-scissors.py, que tomará como entrada la imagen de la cámara web, utilizará MediaPipe para detectar los landmarks de la mano, y clasificará el gesto en "piedra", "papel" o "tijeras" utilizando el modelo entrenado.

Tareas:

1. **Script 1: Grabación del dataset (record-dataset.py)**
    - Usar la cámara web para capturar imágenes de la mano.
    - Utilizar **MediaPipe** para detectar los landmarks de la mano (21 puntos clave con coordenadas x y y).
    - Almacenar las coordenadas de los landmarks junto con la etiqueta correspondiente (0 para "piedra", 1 para "papel", 2 para "tijeras") en archivos **.npy** (por ejemplo, **rps_dataset.npy** y **rps_labels.npy**).
2. **Script 2: Entrenamiento del clasificador (train-gesture-classifier.py)**
    - Cargar los datos del dataset grabado en la primera parte.
    - Implementar una red neuronal densa que tome como entrada las coordenadas x y y de los 21 puntos clave (un total de 42 entradas).
    - Entrenar la red para clasificar los gestos en "piedra", "papel" o "tijeras" usando el dataset.
    - Guardar el modelo entrenado en un archivo (por ejemplo, **rps_model.h5**).
3. **Script 3: Prueba del sistema completo (rock-paper-scissors.py)**
    - Cargar el modelo entrenado.

- Capturar imágenes de la cámara web en tiempo real.
- Usar **MediaPipe** para detectar los landmarks y alimentar estos datos al modelo para clasificar el gesto.
- Mostrar el gesto reconocido en pantalla.

Entrega:

La entrega debe incluir:

1. Los 3 scripts de python:
    a. **record-dataset.py**.
    b. **train-gesture-classifier.py**.
    c. **rock-paper-scissors.py.**
2. Una carpeta con imágenes en donde se muestre el funcionamiento del sistema.

Nota: el código debe estar debidamente documentado con comentarios explicativos para que el trabajo sea fácilmente comprensible para otros revisores.

Ejemplos de código:

En repositorio de github contiene ejemplos de scripts aplicación en python con Google MediaPipe.

Problema 3

Descripción:

En este problema, se proporciona un conjunto de datos que contiene imágenes de escenas naturales de todo el mundo. El objetivo es construir un modelo de clasificación utilizando redes neuronales convolucionales (CNN) para clasificar estas imágenes en una de las seis categorías predefinidas.

Dataset:

https://drive.google.com/file/d/1Pqs5Y6dZr4R66Dby5hIUIjPZtBI28rmJ/view?usp=drive_link

El dataset proporcionado contiene alrededor de 25,000 imágenes de tamaño 150x150, distribuidas en seis categorías:

- **buildings**
- **forest**
- **glacier**
- **mountain**
- **sea**
- **street**

Las imágenes están divididas en tres conjuntos:

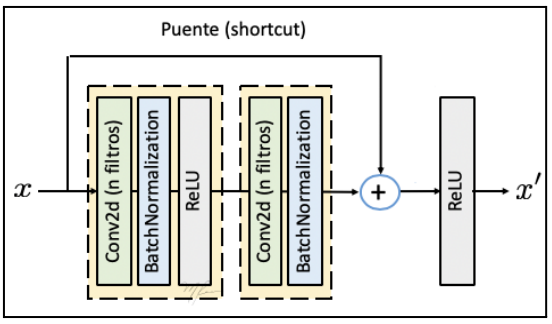
- **Train**: Alrededor de 14,000 imágenes para entrenamiento.
- **Test**: Alrededor de 3,000 imágenes para evaluación del modelo.
- **Prediction**: Alrededor de 7,000 imágenes para predicción final.

Objetivo:

Utilizando el dataset proporcionado, el objetivo es construir y comparar el rendimiento de distintos modelos de clasificación de imágenes utilizando redes neuronales convolucionales y densas que puedan clasificar con precisión las imágenes de escenas naturales en una de las seis categorías mencionadas anteriormente.

Los modelos a diseñar son:

- Modelo con capas densas.
- Modelo con capas convolucionales y densas.
- Modelo que incluya bloques residuales identidad:



Bloque residual identidad

- Modelo que utilice como backbone alguna de las arquitecturas disponibles en TensorFlow (transfer learning): https://www.tensorflow.org/api_docs/python/tf/keras/applications

Se debe entrenar y evaluar cada modelo utilizando técnicas adecuadas de validación y métricas de evaluación de clasificación.

Entrega:

La entrega debe incluir:

Código fuente de la solución implementada en Google Colab, que incluya:

- Análisis previo y preprocesamiento del set de datos.
- Definición y entrenamiento del modelo.
- Resultados de la evaluación del modelo, incluyendo métricas de desempeño y visualizaciones relevantes.

Nota: el código debe estar debidamente documentado con comentarios explicativos para que el trabajo sea fácilmente comprensible para otros revisores.