Bitácora – Día 1

Fecha: 2025-06-26

1. Objetivo del Día

Lograr que el prototipo detecte la letra "A" con ambas manos utilizando los *landmarks* de MediaPipe, y documentar la razón por la cual se descartó el enfoque clásico de CNN (Red Neuronal Convolucional).

2. Entorno y Hardware

- **PC:**
  - GPU: RTX 4060
  - CPU: Ryzen 7 5800X
  - RAM: 16 GB
- **Cámara:** Android Redmi 9d
- **Entorno Python:** Virtualenv con las siguientes librerías:
  - TensorFlow
  - MediaPipe
  - OpenCV
  - NumPy
  - scikit-learn (sklearn)

3. Actividades Realizadas

3.1. Revisión del Enfoque CNN

- Inicialmente, se entrenó un clasificador CNN (MobileNetV2 + capas densas) para diferenciar entre la letra "A" y otros gestos ("other").
- La precisión obtenida fue buena, sin embargo, el modelo mostró sesgo y una pobre generalización para la mano izquierda y fondos variados.

3.2. Descarte de CNN

- **Motivo:** Se descartó este enfoque debido a la alta complejidad de los datos (variabilidad de iluminación, fondos, etc.) y la necesidad de un volumen de miles de imágenes para un entrenamiento robusto.

3.3. Extracción de Landmarks

- Se utilizó MediaPipe Hands para obtener 21 puntos clave (*landmarks*) de cada mano (coordenadas x,y,z).
- Los vectores de *landmarks* se guardaron en archivos .npy en las siguientes rutas: data/landmarks/A/ y data/landmarks/other/.

3.4. Entrenamiento de MLP (Perceptrón Multicapa)

- **Arquitectura del Modelo:**
  - Capa densa (64 unidades) → ReLU → Dropout 0.3
  - Capa densa (32 unidades) → ReLU
  - Capa de salida (2 unidades) → Softmax
- **Entrenamiento:** Se realizó un entrenamiento supervisado con *EarlyStopping* para prevenir el sobreajuste.
- **Modelo Guardado:** El modelo entrenado se guardó como landmark_A_vs_other.keras.

3.5. Inferencia en Tiempo Real

- **Script:** run_inference_landmarks_with_drawing.py
- **Funcionalidad:**
  - Captura el vídeo de la cámara (Unit V2).
  - Dibuja los 21 *landmarks* detectados y sus conexiones en el *frame*.
  - Clasifica el gesto como "A" o "other" e imprime la etiqueta resultante junto con el nivel de confianza.

4. Modelo Final Utilizado

- **Tipo:** Aprendizaje supervisado (MLP sobre *landmarks*)
- **Entrada:** Vector de 63 coordenadas normalizadas (21 *landmarks* × 3 dimensiones)
- **Arquitectura:**
  - Capa de entrada (63 unidades)
  - Capa Densa (64 unidades) → ReLU → Dropout 0.3
  - Capa Densa (32 unidades) → ReLU
  - Capa Densa (2 unidades) → Softmax
- **Nombre del Archivo:** landmark_A_vs_other.keras

5. Resultados y Métricas

- **Accuracy de Validación Interna:** > 95 %
- **Inferencia en Vivo:**
  - Detección de "A" (mano derecha/izquierda): ≈ 98 %
  - Detección de gestos "other": ≈ 96 %
- **Tiempo de Inferencia:** < 5 ms por *frame*

6. Descripción del Pipeline

1. **Captura de Imagen:** Cámara → OpenCV
2. **Detección de Landmarks:** MediaPipe Hands extrae 21 puntos clave de las manos.
3. **Vectorización:** Los puntos clave son aplanados y normalizados a un array de 63

valores.
4. **Clasificación:** El MLP devuelve probabilidades para "A" y "other".
5. **Visualización:** Se dibujan los *landmarks* y se muestra la etiqueta con su confianza sobre el *stream* de vídeo.

7. Avances y Aprendizajes

- El enfoque CNN, aunque ofrecía resultados parciales prometedores, requería una cantidad excesiva de datos y un ajuste fino muy complejo.
- El método basado en *landmarks* se ha demostrado más ligero, robusto y con una menor necesidad de datos para el entrenamiento.
- La visualización en tiempo real de los puntos clave ha sido fundamental para la depuración y una mejor comprensión del comportamiento del sistema.

8. Dificultades y Próximos Pasos

8.1. Dificultades Enfrentadas

- Equilibrar la clase "other" con suficientes ejemplos variados de gestos y fondos para evitar sesgos.
- Afianzar la normalización de los *landmarks* para asegurar invarianza a la escala y la posición de la mano.

8.2. Próximos Pasos

1. Recolectar y procesar más vectores de *landmarks* para la clase "other" con el fin de mejorar su representación.
2. Implementar un preprocesamiento centrado en la muñeca y un escalado relativo para una mayor robustez.
3. Iniciar la extracción y el entrenamiento de *landmarks* para la letra "B" utilizando el mismo *pipeline* establecido.