INSTITUTO TECNOLÓGICO NACIONAL DE MÉXICO CAMPUS CULIACÁN

TOPICOS DE IA

Alumno:

Portillo Zuñiga Steve Javier

Franco Flores Luis Fernando

Unidad 4 Tarea 2

Hora: 10:00 – 11:00

Maestro: Zuriel Dathan Mora Felix

Indice

Construcción del Dataset

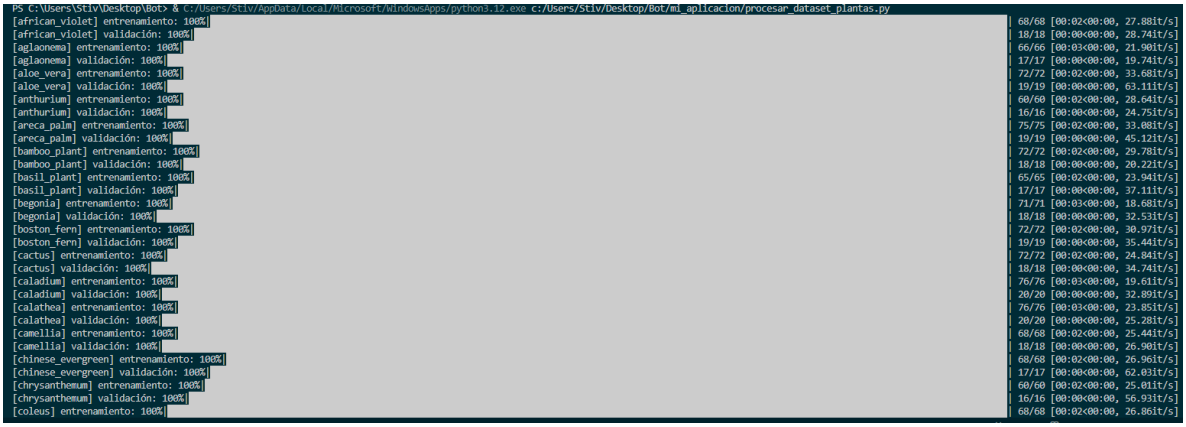
Se construyó un dataset personalizado con 55 clases distintas de plantas, asegurando variedad en forma, color y estructura de hojas o flores. La fuente de datos consistió en imágenes recopiladas de internet

Limpieza:

- Se validaron imágenes con un script para verificar formato, resolución mínima (224x224) y canales RGB.
- Se eliminaron imágenes corruptas o mal etiquetadas.

Transformación:

- Todas las imágenes fueron redimensionadas a 224x224 píxeles.
- Se normalizaron los valores de píxeles al rango [0, 1] para facilitar el aprendizaje del modelo..



Arquitectura del Modelo CNN

Se utilizó una red neuronal convolucional clásica (CNN) definida de la siguiente forma:

- Entrada: Imagen RGB (224x224x3)
- Capas:
  - Conv2D (32 filtros, 3x3, ReLU) + MaxPooling2D
  - Conv2D (64 filtros, 3x3, ReLU) + MaxPooling2D
  - Conv2D (128 filtros, 3x3, ReLU) + MaxPooling2D
  - Flatten + Dense(256, ReLU) + Dropout(0.5)

- Dense de salida con softmax y 55 clases

Esta arquitectura balancea precisión con velocidad de entrenamiento.

Implementación del Modelo

Tecnologías usadas:

- TensorFlow 2.18
- Keras
- OpenCV
- scikit-learn, NumPy

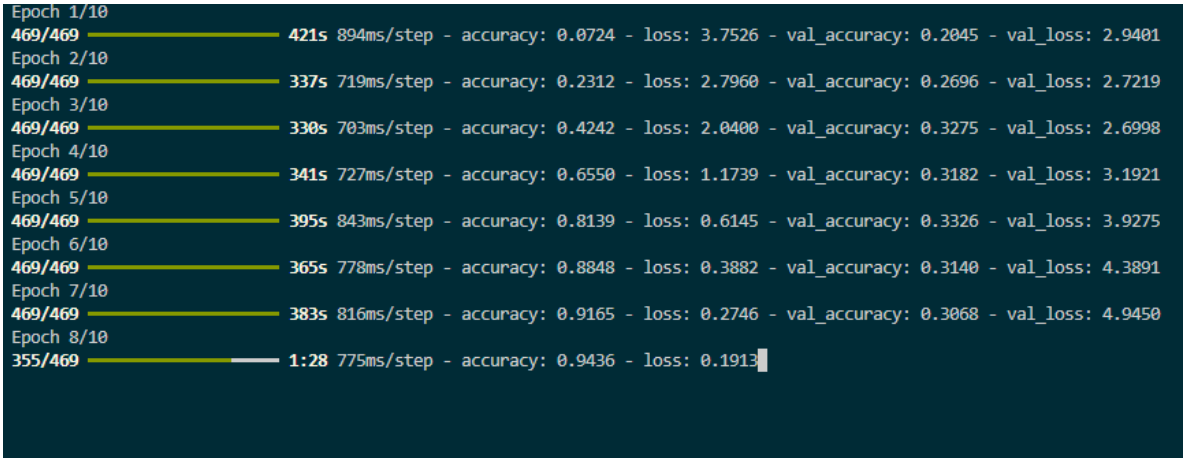
Entrenamiento y Validación:

- Dataset: 5482 imágenes de entrenamiento, 469 imágenes de validación
- Épocas: 10
- Optimización: Adam
- Pérdida: Categorical Crossentropy
- Métrica: Accuracy

Código Base:

```
model = Sequential([
    Conv2D(32, (3,3), activation='relu', input_shape=(224,224,3)),
    MaxPooling2D(2,2),
    Conv2D(64, (3,3), activation='relu'),
    MaxPooling2D(2,2),
    Conv2D(128, (3,3), activation='relu'),
    MaxPooling2D(2,2),
    Flatten(),
    Dense(256, activation='relu'),
    Dropout(0.5),
    Dense(55, activation='softmax')
])
```

```
model.compile(optimizer='adam', loss='categorical_crossentropy',
metrics=['accuracy'])

model.fit(train_gen, validation_data=val_gen, epochs=10)

model.save("modelo_plantas.h5")
```



Evaluación y Pruebas

Métricas utilizadas:

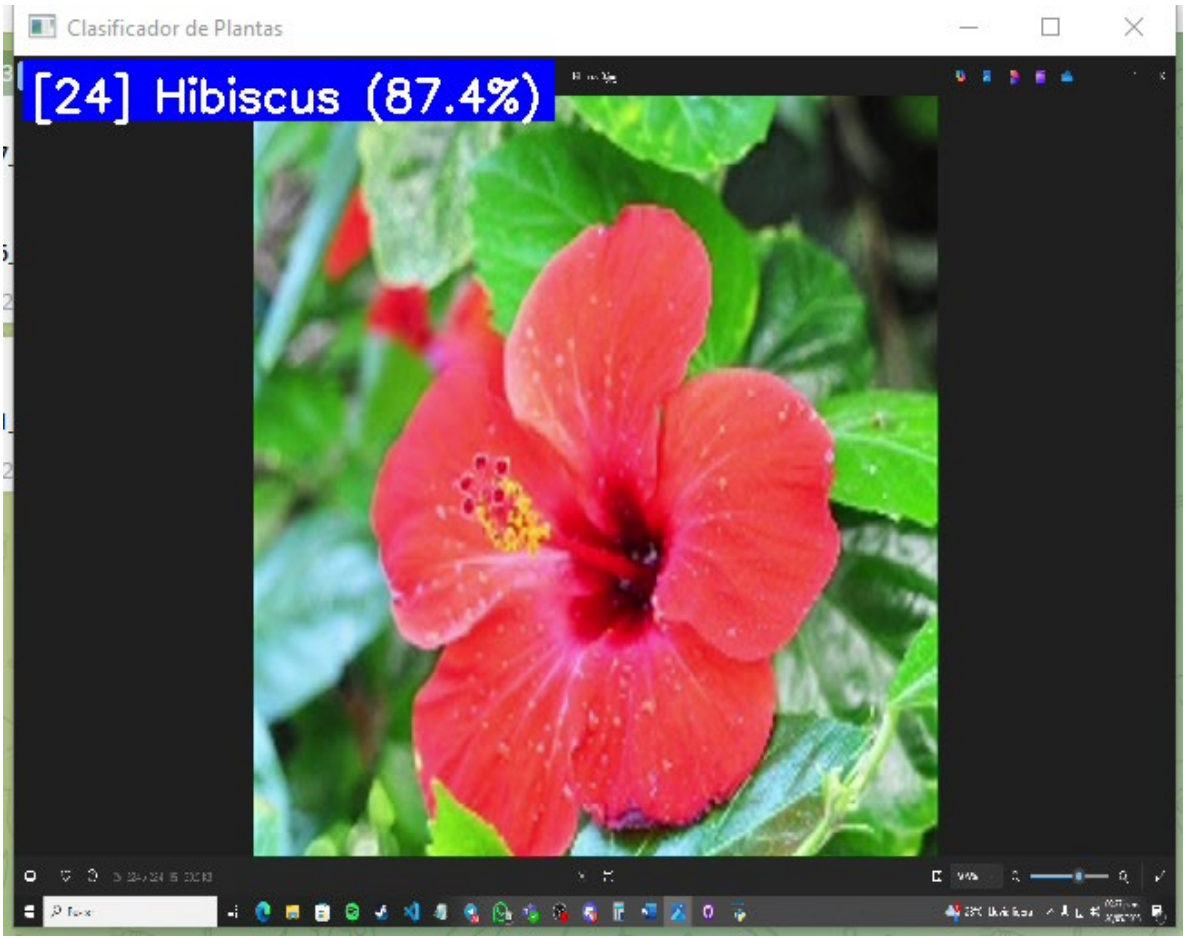
- Accuracy (exactitud global)
- Precision y Recall por clase

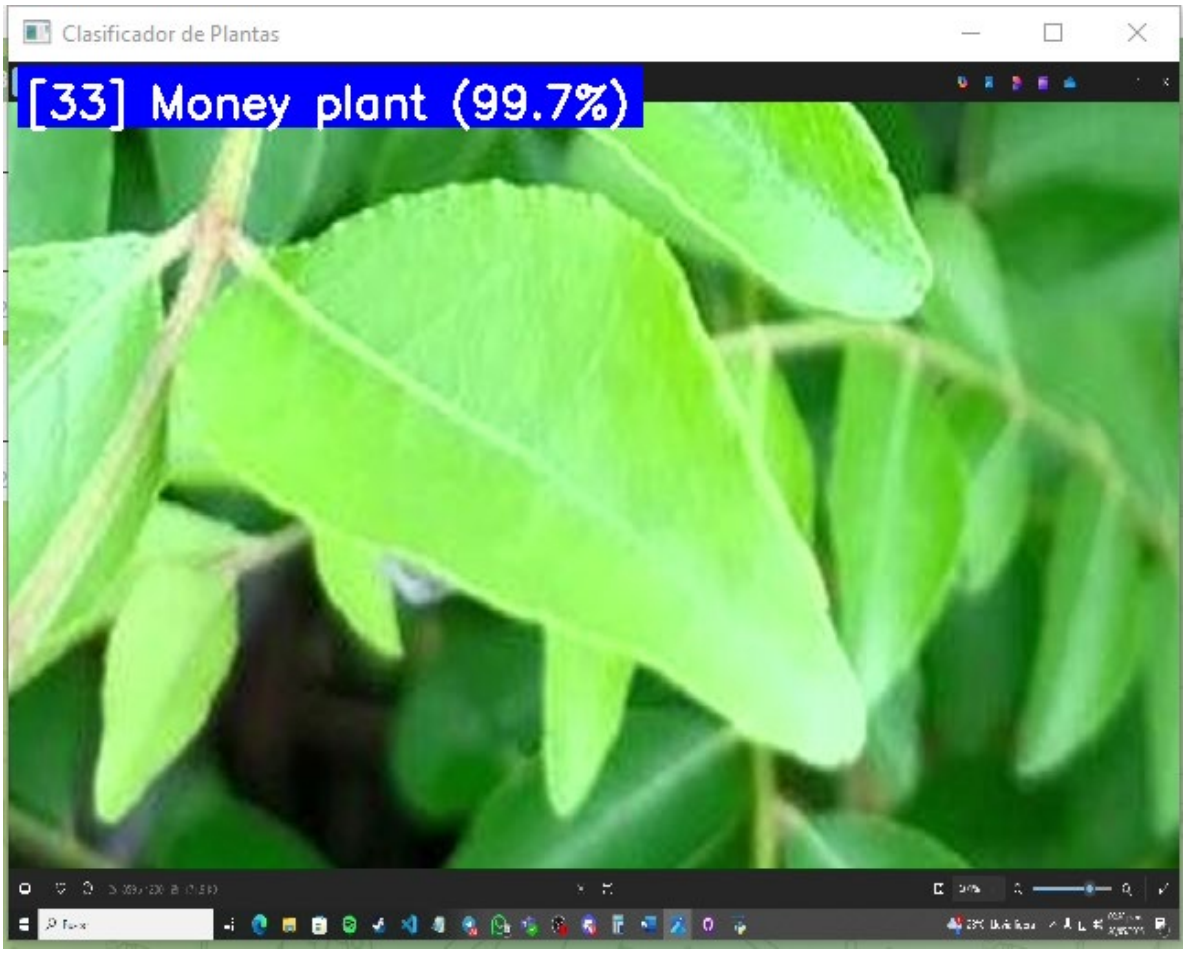
Pruebas con Cámara Web:

Se implementó un sistema en tiempo real que:

- Captura imagen con cv2.VideoCapture
- Preprocesa el frame
- Realiza inferencia con el modelo entrenado
- Muestra la etiqueta predicha sobre el video

```
cap = cv2.VideoCapture(0)

while True:

    ret, frame = cap.read()

    img = cv2.resize(frame, (224, 224))

    img_array = np.expand_dims(img / 255.0, axis=0)

    prediction = model.predict(img_array)

    label = labels[np.argmax(prediction)]

    cv2.putText(frame, f"Prediccion: {label}", (20, 40),
cv2.FONT_HERSHEY_SIMPLEX, 1, (0,255,0), 2)

    cv2.imshow("Clasificador de Plantas", frame)

    if cv2.waitKey(1) & 0xFF == ord('q'):

        break
```


Clasificador de Plantas
[24] Hibiscus (87.4%)


Clasificador de Plantas
[33] Money plant (99.7%)