Actividad en Clase Transfer Learning

- Roberto Angel Rillo Calva A01642022
- Jacob Valdenegro Monzón A01640992
- Carlos Dhali Tejeda Tapia A00344820
- Enrique Mora Navarro A01635459

Descargar las imágenes con Icrawler para usarlas en el modelo

```
!pip install icrawler

from icrawler.builtin import GoogleImageCrawler

# Crear función para descargar imágenes
def download_images(keyword, num, output_dir):
    crawler = GoogleImageCrawler(storage={'root_dir': output_dir})
    crawler.crawl(keyword=keyword, max_num=num)

download_images('keyboard', 150, 'data/teclado')  # Pide 150 por si algunas fallan
download_images('mouse', 150, 'data/mouse')
download_images('computer monitor', 150, 'data/monitor')  # Usar un término más específico
```

```
Requirement already satisfied: icrawler in /usr/local/lib/python3.10/dist-packages (0.6.9)
Requirement already satisfied: beautifulsoup4 in /usr/local/lib/python3.10/dist-packages (from icrawler) (4.12.3)
Requirement already satisfied: bs4 in /usr/local/lib/python3.10/dist-packages (from icrawler) (0.0.2)
Requirement already satisfied: lxml in /usr/local/lib/python3.10/dist-packages (from icrawler) (4.9.4)
Requirement already satisfied: pillow in /usr/local/lib/python3.10/dist-packages (from icrawler) (10.4.0)
Requirement already satisfied: pyyaml in /usr/local/lib/python3.10/dist-packages (from icrawler) (6.0.2)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from icrawler) (2.32.3)
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from icrawler) (1.16.0)
Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.10/dist-packages (from beautifulsoup4->icrawler) (2.6)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->icrawler) (3.4.0)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->icrawler) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->icrawler) (2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->icrawler) (2024.8.30)
ERROR:downloader:Response status code 444, file https://www.walmart.com/ip/onn-Mini-Compact-Wireless-Office-Keyboard-USB-Receiver-78-Key
ERROR:downloader:Response status code 444, file https://www.walmart.com/ip/onn-Mini-Compact-Wireless-Office-Keyboard-USB-Receiver-78-Key
Exception in thread parser-001:
Traceback (most recent call last):
  File "/usr/lib/python3.10/threading.py", line 1016, in _bootstrap_inner
    self.run()
  File "/usr/lib/python3.10/threading.py", line 953, in run
    self._target(*self._args, **self._kwargs)
  File "/usr/local/lib/python3.10/dist-packages/icrawler/parser.py", line 94, in worker_exec
    for task in self.parse(response, **kwargs):
TypeError: 'NoneType' object is not iterable
ERROR:downloader:Response status code 403, file https://media.steelseriescdn.com/thumbs/catalog/items/62598/b5de2516e6524269bb4b1c0218c9
Exception in thread parser-001:
Traceback (most recent call last):
  File "/usr/lib/python3.10/threading.py", line 1016, in _bootstrap_inner
    self.run()
  File "/usr/lib/python3.10/threading.py", line 953, in run
    self._target(*self._args, **self._kwargs)
  File "/usr/local/lib/python3.10/dist-packages/icrawler/parser.py", line 94, in worker_exec
    for task in self.parse(response, **kwargs):
TypeError: 'NoneType' object is not iterable
ERROR:downloader:Response status code 403, file https://i.dell.com/is/image/DellContent//content/dam/images/products/electronics-and-acc
ERROR:downloader:Response status code 403, file https://images.philips.com/is/image/philipsconsumer/0bf2c45f5b1944ecba8bb012006654ef?$pr
ERROR:downloader:Response status code 403, file https://i.dell.com/is/image/DellContent/content/dam/ss2/product-images/dell-client-produ
ERROR:downloader:Response status code 404, file https://www.cnet.com/a/img/resize/cab157f21a6bd96061ffd7badd62786715d74e9f/hub/2024/07/3
ERROR:downloader:Response status code 403, file https://i.dell.com/is/image/DellContent/content/dam/ss2/product-images/dell-client-produ
ERROR:downloader:Response status code 403, file https://www.newegg.com/insider/wp-content/uploads/2019/02/intel-hades-canyon-multiple-mo
ERROR:downloader:Response status code 403, file https://www.pbtech.co.nz/fileslib/20231222151832_off-lease-monitors.png
ERROR:downloader:Exception caught when downloading file https://pisces.bbystatic.com/image2/BestBuy_US/dam/pol_Portable-Monitors-EVN-191
ERROR:downloader:Response status code 403, file https://www.intel.com/content/dam/www/central-libraries/us/en/images/s2-a8-1-screen-resc
ERROR:downloader:Response status code 403, file https://external-preview.redd.it/gigabyte-aorus-fo32u2p-4k-240hz-qd-oled-gaming-monitor-
ERROR:downloader:Response status code 403, file https://images.philips.com/is/image/philipsconsumer/fe3067b878ec4bdaa6c5b02000c1248d?$pr
ERROR:downloader:Response status code 403, file https://i.dell.com/is/image/DellContent//content/dam/images/products/electronics-and-acc
ERROR:downloader:Response status code 403, file http://www.w3.org/2000/svg" viewBox="0 0 24 24"><path d="M19 6.41L17.59 5 12 10.59 6.41
ERROR:downloader:Response status code 444, file https://www.walmart.com/ip/Acer-23-8-Full-HD-1920-x-1080-Ultra-Thin-IPS-Monitor-75Hz-1ms
ERROR:downloader:Response status code 444, file https://www.walmart.com/ip/Acer-23-8-Full-HD-1920-x-1080-Ultra-Thin-IPS-Monitor-75Hz-1ms
ERROR:downloader:Exception caught when downloading file https://www.ebay.com/itm/335029522240?chn=ps&amp;mkevt=1&amp;mkcid=28" data-agdh
ERROR:downloader:Exception caught when downloading file https://www.ebay.com/itm/335029522240?chn=ps&amp;mkevt=1&amp;mkcid=28" data-agdh
```

```
ERROR:downloader:Exception caught when downloading file https://www.ebay.com/itm/335029522240?chn=ps&amp;mkevt=1&amp;mkcid=28" data-agdh
ERROR:downloader:Response status code 444, file https://www.walmart.com/ip/VDSXT-24-inch-FHD-1080P-1920x1080-75hz-16-9-Flat-Panel-Ultraw
ERROR:downloader:Response status code 403, file https://www.dell.com/en-us/shop/dell-32-curved-fhd-monitor-s3222hn/apd/210-bbqy/monitors
ERROR:downloader:Response status code 403, file https://www.hp.com/us-en/shop/pdp/hp-m22f-fhd-monitor" data-agdh="arwt" id="vplaurlg_6a0
```

Cargar y procesar las imágenes

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator

# Crear el generador de imágenes
datagen = ImageDataGenerator(
    rescale=1.0/255,        # Escalar los valores de píxeles entre 0 y 1
    validation_split=0.2    # Dividir el dataset en 80% entrenamiento y 20% validación
)

# Cargar imágenes para entrenamiento y validación
train_data = datagen.flow_from_directory(
    'data/',               # Ruta a la carpeta con imágenes
    target_size=(224, 224), # Tamaño esperado por ResNet50
    batch_size=32,          # Tamaño del lote
    class_mode='categorical', # Clasificación multiclase
    subset='training'       # Subconjunto de entrenamiento
)

val_data = datagen.flow_from_directory(
    'data/',
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical',
    subset='validation'     # Subconjunto de validación
)
```

```
Found 322 images belonging to 3 classes.
Found 79 images belonging to 3 classes.
```

Se importa el modelo preentrenado

```
from tensorflow.keras.applications import ResNet50
from tensorflow.keras.layers import Dense, GlobalAveragePooling2D
from tensorflow.keras.models import Model

base_model = ResNet50(weights='imagenet', include_top=False, input_shape=(224, 224, 3))
```

```
import os
print("Imágenes de teclado:", len(os.listdir('data/teclado')))
print("Imágenes de mouse:", len(os.listdir('data/mouse')))
print("Imágenes de monitor:", len(os.listdir('data/monitor')))
```

```
Imágenes de teclado: 209
Imágenes de mouse: 115
Imágenes de monitor: 77
```

Agregar la capa de clasificación

```
# Congelar las capas de la base para usar las características preentrenadas
base_model.trainable = False

# Agregar capas de clasificación
x = base_model.output
x = GlobalAveragePooling2D()(x)  # Agregar capa de pooling para reducir dimensiones
x = Dense(1024, activation='relu')(x)  # Capa dense intermedia
predictions = Dense(3, activation='softmax')(x)  # Capa de salida con 3 clases

# Crear el modelo final
model = Model(inputs=base_model.input, outputs=predictions)
```

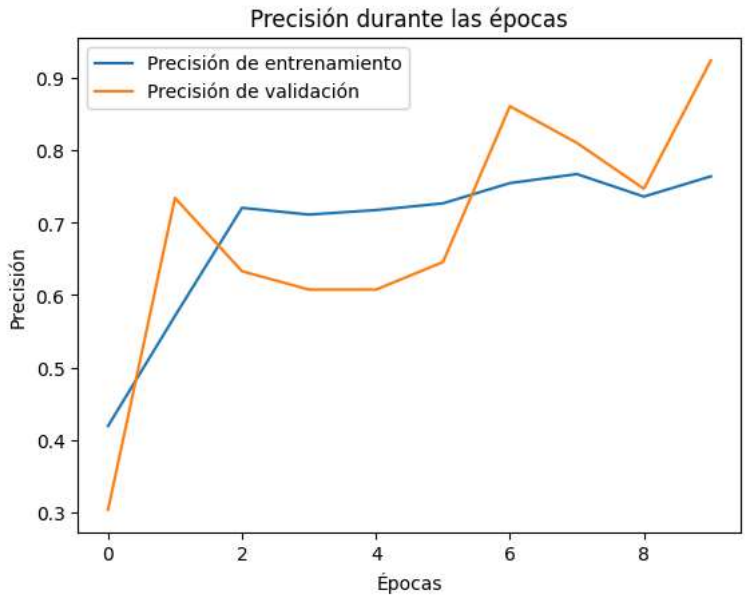
Compilar y entrenar el modelo

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

# Entrenar el modelo
history = model.fit(train_data, validation_data=val_data, epochs=10)
```

```
/usr/local/lib/python3.10/dist-packages/PIL/Image.py:1056: UserWarning: Palette images with Transparency expressed in bytes should be co
  warnings.warn(
Epoch 1/10
/usr/local/lib/python3.10/dist-packages/keras/src/trainers/data_adapters/py_dataset_adapter.py:121: UserWarning: Your `PyDataset` class
  self._warn_if_super_not_called()
11/11 ━━━━━━━━━━━━━━━━━━━━ 86s 7s/step - accuracy: 0.3715 - loss: 1.6342 - val_accuracy: 0.3038 - val_loss: 1.1551
Epoch 2/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 83s 7s/step - accuracy: 0.5106 - loss: 1.0077 - val_accuracy: 0.7342 - val_loss: 0.7125
Epoch 3/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 73s 6s/step - accuracy: 0.7061 - loss: 0.7845 - val_accuracy: 0.6329 - val_loss: 0.9681
Epoch 4/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 89s 7s/step - accuracy: 0.6830 - loss: 0.7647 - val_accuracy: 0.6076 - val_loss: 0.6965
Epoch 5/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 77s 6s/step - accuracy: 0.7139 - loss: 0.5904 - val_accuracy: 0.6076 - val_loss: 0.7121
Epoch 6/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 86s 7s/step - accuracy: 0.7298 - loss: 0.6208 - val_accuracy: 0.6456 - val_loss: 0.6110
Epoch 7/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 83s 7s/step - accuracy: 0.7208 - loss: 0.5837 - val_accuracy: 0.8608 - val_loss: 0.4482
Epoch 8/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 73s 7s/step - accuracy: 0.7604 - loss: 0.6048 - val_accuracy: 0.8101 - val_loss: 0.5312
Epoch 9/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 74s 6s/step - accuracy: 0.7346 - loss: 0.6721 - val_accuracy: 0.7468 - val_loss: 0.6137
Epoch 10/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 73s 6s/step - accuracy: 0.7414 - loss: 0.5491 - val_accuracy: 0.9241 - val_loss: 0.3929
```

Gráfica de las épocas

```
import matplotlib.pyplot as plt

# Graficar la precisión
plt.plot(history.history['accuracy'], label='Precisión de entrenamiento')
plt.plot(history.history['val_accuracy'], label='Precisión de validación')
plt.title('Precisión durante las épocas')
plt.xlabel('Épocas')
plt.ylabel('Precisión')
plt.legend()
plt.show()
```



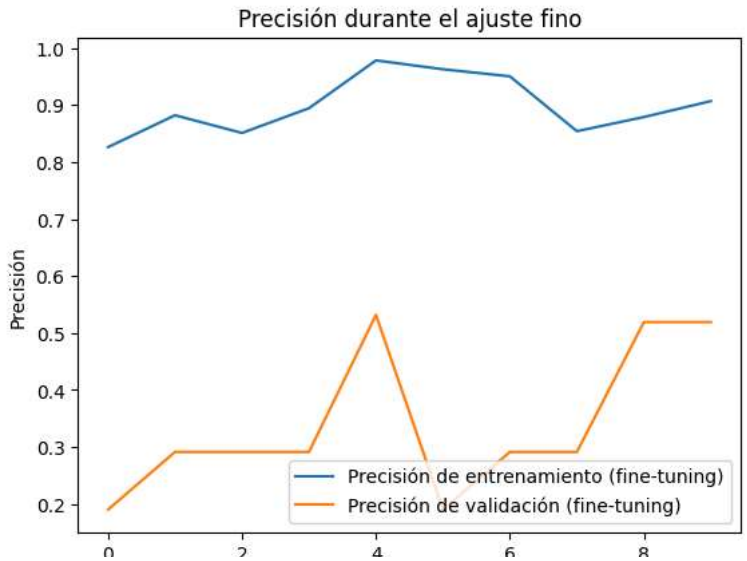
Fine-tuning

```
# Descongelar algunas capas de la base
base_model.trainable = True

# Recompilar el modelo con un menor learning rate
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

# Reentrenar el modelo
fine_tune_history = model.fit(train_data, validation_data=val_data, epochs=10)
```

```
Epoch 1/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 327s 24s/step - accuracy: 0.7494 - loss: 1.6075 - val_accuracy: 0.1899 - val_loss: 4837.6987
Epoch 2/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 260s 23s/step - accuracy: 0.8677 - loss: 1.2288 - val_accuracy: 0.2911 - val_loss: 63.1578
Epoch 3/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 272s 24s/step - accuracy: 0.8865 - loss: 0.3143 - val_accuracy: 0.2911 - val_loss: 49.4175
Epoch 4/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 262s 23s/step - accuracy: 0.8925 - loss: 0.3186 - val_accuracy: 0.2911 - val_loss: 28.3604
Epoch 5/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 268s 24s/step - accuracy: 0.9798 - loss: 0.1236 - val_accuracy: 0.5316 - val_loss: 1.0815
Epoch 6/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 267s 24s/step - accuracy: 0.9734 - loss: 0.1496 - val_accuracy: 0.1899 - val_loss: 43.9299
Epoch 7/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 322s 24s/step - accuracy: 0.9382 - loss: 0.2110 - val_accuracy: 0.2911 - val_loss: 12.1318
Epoch 8/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 318s 23s/step - accuracy: 0.8855 - loss: 0.5553 - val_accuracy: 0.2911 - val_loss: 196.0603
Epoch 9/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 318s 23s/step - accuracy: 0.8718 - loss: 0.4696 - val_accuracy: 0.5190 - val_loss: 229.5517
Epoch 10/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 258s 23s/step - accuracy: 0.9142 - loss: 0.3482 - val_accuracy: 0.5190 - val_loss: 16.0816
```

Gráfica de las épocas

```
# Graficar precisión del ajuste fino
plt.plot(fine_tune_history.history['accuracy'], label='Precisión de entrenamiento (fine-tuning)')
plt.plot(fine_tune_history.history['val_accuracy'], label='Precisión de validación (fine-tuning)')
plt.title('Precisión durante el ajuste fino')
plt.xlabel('Épocas')
plt.ylabel('Precisión')
plt.legend()
plt.show()
```



Predicciones

```
import numpy as np
from tensorflow.keras.preprocessing import image

# Cargar y preprocesar una imagen de prueba
img_path = '/content/data/mouse/000002.jpg'  # Cambia la ruta a una imagen de prueba
img = image.load_img(img_path, target_size=(224, 224))
img_array = image.img_to_array(img)
img_array = np.expand_dims(img_array, axis=0) / 255.0  # Normalización

# Realizar la predicción
pred = model.predict(img_array)
print(f"Predicción: {np.argmax(pred)}")
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 337ms/step
Predicción: 2
```

```
# Mostrar las clases que el generador ha identificado
print("Clases:", train_data.class_indices)
```

```
Clases: {'monitor': 0, 'mouse': 1, 'teclado': 2}
```

Realizamos varias predicciones pero vimos que siempre se tiene a ir por la clase 2 que es la de teclado, lo que pensamos que pudo haber pasado es que se dió un caso de overfitting hacia la clase de teclado porque es la clase que más imágenes tenía de las 3. Esto fue debido a que la herramienta que utilizamos para conseguir las imágenes puede eliminar algunas y dejar las clases irregulares, creemos que esta diferencia de cantidades de imágenes pudo provocar estos resultados.