Actividad en Clase Transfer Learning

Victor Hugo Arreola Elenes - A01635682

Fernando Ojeda Marín - A01639252

Mario Alberto Castañeda Martínez - A01640152

```
import matplotlib.pyplot as plt
import numpy as np
import os
import tensorflow as tf
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
import os

# Ruta a la carpeta en Google Drive donde se encuentran tus datos
PATH = '/content/drive/MyDrive/Tec/7mo Semestre/Inteligencia Artificial II/Deep Learning/Data-Monitor-Teclados-Mouse/'

# Rutas a las carpetas de entrenamiento y validación
train_dir = os.path.join(PATH, 'train')
validation_dir = os.path.join(PATH, 'validation')

BATCH_SIZE = 32
IMG_SIZE  =(128,128)

train_dataset = tf.keras.utils.image_dataset_from_directory(train_dir,
                                                            shuffle = True,
                                                            batch_size=BATCH_SIZE,
                                                            image_size=IMG_SIZE)

validation_dataset = tf.keras.utils.image_dataset_from_directory(validation_dir,
                                                                 shuffle = True,
                                                                 batch_size=BATCH_SIZE,
                                                                 image_size=IMG_SIZE)
```

```
Found 732 files belonging to 3 classes.
Found 93 files belonging to 3 classes.
```

```
class_names = train_dataset.class_names

plt.figure(figsize=(10,10))
for image, labels in train_dataset.take(1):
  for i in range(9):
    ax=plt.subplot(3,3,i+1)
    plt.imshow(image[i].numpy().astype('uint8'))
    plt.title(class_names[labels[i]])
    plt.axis("off")
```

mouse keyboard keyboard

```
val_batches = tf.data.experimental.cardinality(validation_dataset)
test_dataset = validation_dataset.take(val_batches // 5)
validation_dataset = validation_dataset.skip(val_batches // 5)

print("numero de batches para test_data set = %d" %tf.data.experimental.cardinality(test_dataset))
print("numero de batches para validation_data set = %d" %tf.data.experimental.cardinality(validation_dataset))
```

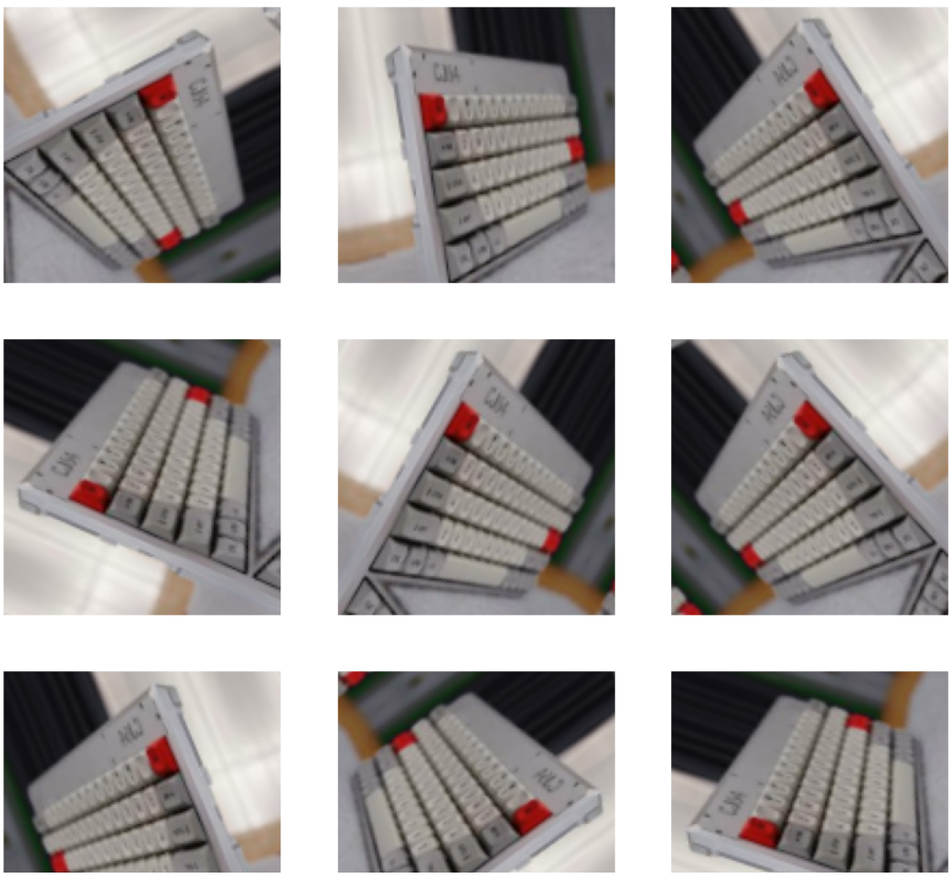
```
numero de batches para test_data set = 0
numero de batches para validation_data set = 3
```

```
AUTOTUNE = tf.data.AUTOTUNE

train_dataset = train_dataset.prefetch(buffer_size=AUTOTUNE)
validation_dataset = validation_dataset.prefetch(buffer_size=AUTOTUNE)
test_dataset = test_dataset.prefetch(buffer_size=AUTOTUNE)
```

```
data_augmentation = tf.keras.Sequential([
    tf.keras.layers.RandomFlip("horizontal"),
    tf.keras.layers.RandomRotation(0.2),
])
```

```
for image, _ in train_dataset.take(1):
  plt.figure(figsize=(10,10))
  first_image = image[0]
  for image, labels in train_dataset.take(1):
    for i in range(9):
      ax=plt.subplot(3,3,i+1)
      augmented_image = data_augmentation(tf.expand_dims(first_image,0))
      plt.imshow(augmented_image[0]/255)
      plt.axis("off")
```



```
rescale = tf.keras.layers.Rescaling(1./127.5,offset = 1)

preprocess_input = tf.keras.applications.mobilenet_v2.preprocess_input
```

```
IMG_SHAPE = IMG_SIZE + (3,)
print(IMG_SHAPE)

base_model = tf.keras.applications.InceptionV3(input_shape=IMG_SHAPE,include_top=False,weights="imagenet")
```

```
(128, 128, 3)
```

```
image_batch, label_batch = next(iter(train_dataset))
feature_batch = base_model(image_batch)

print(feature_batch.shape)
```

```
(32, 2, 2, 2048)
```

```
base_model.trainable=False
base_model.summary()
```

```
batch_normalization_454 (B  (None, 2, 2, 384)           1152      ['conv2d_454[0][0]']
atchNormalization)

batch_normalization_455 (B  (None, 2, 2, 384)           1152      ['conv2d_455[0][0]']
atchNormalization)

batch_normalization_458 (B  (None, 2, 2, 384)           1152      ['conv2d_458[0][0]']
atchNormalization)

batch_normalization_459 (B  (None, 2, 2, 384)           1152      ['conv2d_459[0][0]']
atchNormalization)

conv2d_460 (Conv2D)         (None, 2, 2, 192)           245760    ['average_pooling2d_43[0][0]']

batch_normalization_452 (B  (None, 2, 2, 320)           960       ['conv2d_452[0][0]']
atchNormalization)

activation_454 (Activation  (None, 2, 2, 384)           0         ['batch_normalization_454[0][0
)                                                                  ]']

activation_455 (Activation  (None, 2, 2, 384)           0         ['batch_normalization_455[0][0
)                                                                  ]']

activation_458 (Activation  (None, 2, 2, 384)           0         ['batch_normalization_458[0][0
)                                                                  ]']

activation_459 (Activation  (None, 2, 2, 384)           0         ['batch_normalization_459[0][0
)                                                                  ]']

batch_normalization_460 (B  (None, 2, 2, 192)           576       ['conv2d_460[0][0]']
atchNormalization)

activation_452 (Activation  (None, 2, 2, 320)           0         ['batch_normalization_452[0][0
)                                                                  ]']

mixed9_0 (Concatenate)      (None, 2, 2, 768)           0         ['activation_454[0][0]',
                                                                   'activation_455[0][0]']

concatenate_8 (Concatenate  (None, 2, 2, 768)           0         ['activation_458[0][0]',
)                                                                  'activation_459[0][0]']

activation_460 (Activation  (None, 2, 2, 192)           0         ['batch_normalization_460[0][0
)                                                                  ]']

mixed9 (Concatenate)        (None, 2, 2, 2048)          0         ['activation_452[0][0]',
                                                                   'mixed9_0[0][0]',
                                                                   'concatenate_8[0][0]',
                                                                   'activation_460[0][0]']

conv2d_465 (Conv2D)         (None, 2, 2, 448)           917504    ['mixed9[0][0]']

batch_normalization_465 (B  (None, 2, 2, 448)           1344      ['conv2d_465[0][0]']
atchNormalization)

activation_465 (Activation  (None, 2, 2, 448)           0         ['batch_normalization_465[0][0
)                                                                  ]']

conv2d_462 (Conv2D)         (None, 2, 2, 384)           786432    ['mixed9[0][0]']
```

```
global_average_layer = tf.keras.layers.GlobalAveragePooling2D()
feature_batch_average = global_average_layer(feature_batch)
print(feature_batch_average.shape)
```

```
(32, 2048)
```

```
prediction_layer = tf.keras.layers.Dense(3)
prediction_batch = prediction_layer(feature_batch_average)
print(prediction_batch.shape)
```

```
(32, 3)
```

Unir Modelo

```
from tensorflow.keras.callbacks import ModelCheckpoint
```

```
inputs = tf.keras.Input(shape=(128, 128, 3))
x = data_augmentation(inputs)
x = preprocess_input(x)
x = base_model(x, training=False)
x = global_average_layer(x)
x = tf.keras.layers.Dropout(0.2)(x)

outputs = prediction_layer(x)

model = tf.keras.Model(inputs, outputs)

```

```
base_learning_rate = 0.00001
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=base_learning_rate),
              loss=tf.keras.losses.SparseCategoricalCrossentropy(),
              metrics=['accuracy'])
model.summary()
```

```
Model: "model_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_10 (InputLayer)       [(None, 128, 128, 3)]     0
```

```
 sequential_4 (Sequential)   (None, 128, 128, 3)       0

 tf.math.truediv_4 (TFOpLam  (None, 128, 128, 3)       0
 bda)

 tf.math.subtract_4 (TFOpLa  (None, 128, 128, 3)       0
 mbda)

 inception_v3 (Functional)   (None, 2, 2, 2048)        21802784

 global_average_pooling2d_4  (None, 2048)              0
  (GlobalAveragePooling2D)

 dropout_4 (Dropout)         (None, 2048)              0

 dense_5 (Dense)             (None, 3)                 6147

=================================================================
Total params: 21808931 (83.19 MB)
Trainable params: 6147 (24.01 KB)
Non-trainable params: 21802784 (83.17 MB)
_________________________________________________________________
```

```
initial_epochs = 20
loss0, accuracy0 = model.evaluate(validation_dataset)
print(loss0)
print(accuracy0)
```

```
3/3 [==============================] - 6s 58ms/step - loss: 6.5696 - accuracy: 0.3333
6.569586753845215
0.3333333432674408
```

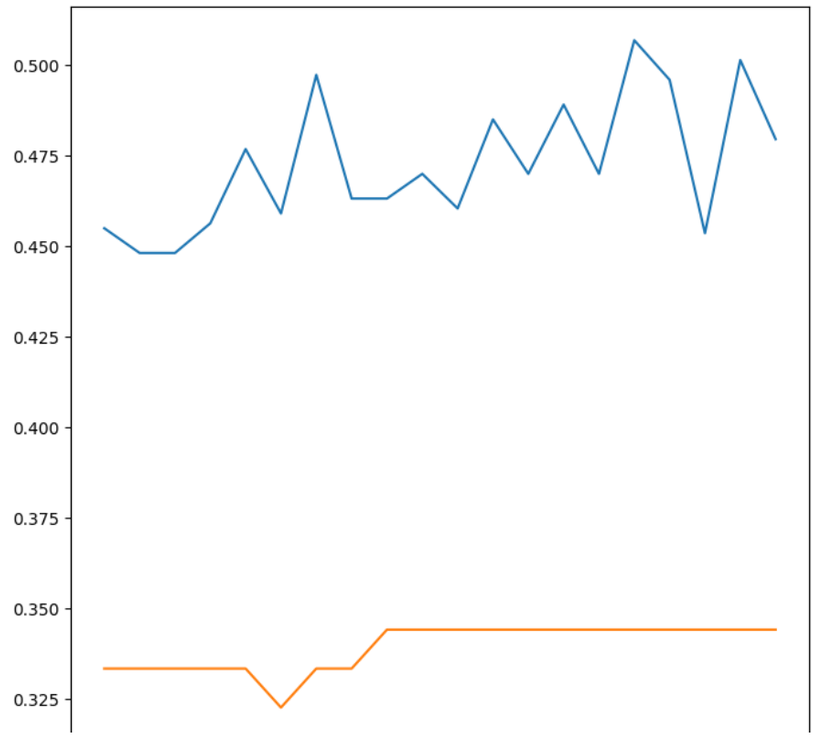
```
history = model.fit(train_dataset,
                    epochs = initial_epochs,
                    validation_data = validation_dataset)
```

```
Epoch 9/20
23/23 [==============================] - ETA: 0s - loss: 5.4831 - accuracy: 0.4399
Epoch 9: val_accuracy did not improve from 0.32258
23/23 [==============================] - 2s 73ms/step - loss: 5.4831 - accuracy: 0.4399 - val_loss: 6.6910 - val_accuracy: 0.3226
Epoch 10/20
23/23 [==============================] - ETA: 0s - loss: 5.4192 - accuracy: 0.4385
Epoch 10: val_accuracy did not improve from 0.32258
23/23 [==============================] - 2s 80ms/step - loss: 5.4192 - accuracy: 0.4385 - val_loss: 6.6904 - val_accuracy: 0.3226
Epoch 11/20
23/23 [==============================] - ETA: 0s - loss: 5.4576 - accuracy: 0.4112
Epoch 11: val_accuracy did not improve from 0.32258
23/23 [==============================] - 2s 58ms/step - loss: 5.4576 - accuracy: 0.4112 - val_loss: 6.6890 - val_accuracy: 0.3226
Epoch 12/20
22/23 [===========================>..] - ETA: 0s - loss: 5.5100 - accuracy: 0.4403
Epoch 12: val_accuracy did not improve from 0.32258
23/23 [==============================] - 2s 57ms/step - loss: 5.5302 - accuracy: 0.4440 - val_loss: 6.6991 - val_accuracy: 0.3226
Epoch 13/20
22/23 [===========================>..] - ETA: 0s - loss: 5.5067 - accuracy: 0.4190
Epoch 13: val_accuracy did not improve from 0.32258
23/23 [==============================] - 2s 62ms/step - loss: 5.5250 - accuracy: 0.4153 - val_loss: 6.6972 - val_accuracy: 0.3226
Epoch 14/20
22/23 [===========================>..] - ETA: 0s - loss: 5.0702 - accuracy: 0.4673
Epoch 14: val_accuracy did not improve from 0.32258
23/23 [==============================] - 2s 56ms/step - loss: 5.1084 - accuracy: 0.4631 - val_loss: 6.6960 - val_accuracy: 0.3226
Epoch 15/20
22/23 [===========================>..] - ETA: 0s - loss: 5.3358 - accuracy: 0.4105
Epoch 15: val_accuracy did not improve from 0.32258
23/23 [==============================] - 2s 57ms/step - loss: 5.3875 - accuracy: 0.4098 - val_loss: 6.6949 - val_accuracy: 0.3226
Epoch 16/20
22/23 [===========================>..] - ETA: 0s - loss: 5.4233 - accuracy: 0.4304
Epoch 16: val_accuracy did not improve from 0.32258
23/23 [==============================] - 2s 75ms/step - loss: 5.4437 - accuracy: 0.4331 - val_loss: 6.6944 - val_accuracy: 0.3226
Epoch 17/20
23/23 [==============================] - ETA: 0s - loss: 4.8749 - accuracy: 0.4467
Epoch 17: val_accuracy did not improve from 0.32258
23/23 [==============================] - 2s 71ms/step - loss: 4.8749 - accuracy: 0.4467 - val_loss: 6.6938 - val_accuracy: 0.3226
Epoch 18/20
23/23 [==============================] - ETA: 0s - loss: 4.9553 - accuracy: 0.4522
Epoch 18: val_accuracy did not improve from 0.32258
23/23 [==============================] - 2s 61ms/step - loss: 4.9553 - accuracy: 0.4522 - val_loss: 6.6859 - val_accuracy: 0.3226
Epoch 19/20
23/23 [==============================] - ETA: 0s - loss: 5.0189 - accuracy: 0.4194
Epoch 19: val_accuracy did not improve from 0.32258
23/23 [==============================] - 2s 58ms/step - loss: 5.0189 - accuracy: 0.4194 - val_loss: 6.6712 - val_accuracy: 0.3226
Epoch 20/20
22/23 [===========================>..] - ETA: 0s - loss: 4.8692 - accuracy: 0.4489
Epoch 20: val_accuracy did not improve from 0.32258
23/23 [==============================] - 2s 57ms/step - loss: 4.8606 - accuracy: 0.4508 - val_loss: 6.5687 - val_accuracy: 0.3226
Epoch 1/20
23/23 [==============================] - 29s 143ms/step - loss: 5.1122 - accuracy: 0.4549 - val_loss: 6.5584 - val_accuracy: 0.3333
Epoch 2/20
23/23 [==============================] - 3s 113ms/step - loss: 5.1438 - accuracy: 0.4481 - val_loss: 6.5413 - val_accuracy: 0.3333
Epoch 3/20
23/23 [==============================] - 2s 79ms/step - loss: 5.1721 - accuracy: 0.4481 - val_loss: 6.4413 - val_accuracy: 0.3333
Epoch 4/20
23/23 [==============================] - 2s 83ms/step - loss: 5.0210 - accuracy: 0.4563 - val_loss: 6.4584 - val_accuracy: 0.3333
Epoch 5/20
23/23 [==============================] - 3s 95ms/step - loss: 4.6242 - accuracy: 0.4768 - val_loss: 6.4706 - val_accuracy: 0.3333
```

```
acc = history.history['accuracy']
val_acc = history.history['val_accuracy']

plt.figure(figsize=(8,8))
plt.plot(acc, label = 'Training acc')
plt.plot(val_acc, label  ='Validation acc')
```

```
[<matplotlib.lines.Line2D at 0x783693f6dbd0>]
```



Fine Tuning

```
base_model.trainable = True

print("Numero de capas ", len(base_model.layers))
```

```
Numero de capas  311
```

```
fine_tune_at = 100
for layer in base_model.layers[:fine_tune_at]:
  layer.trainable = False
```

```
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=base_learning_rate/10),
              loss=tf.keras.losses.SparseCategoricalCrossentropy(),
              metrics=['accuracy'])

model.summary()
```

```
Model: "model_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_10 (InputLayer)       [(None, 128, 128, 3)]     0

 sequential_4 (Sequential)   (None, 128, 128, 3)       0

 tf.math.truediv_4 (TFOpLam  (None, 128, 128, 3)       0
 bda)

 tf.math.subtract_4 (TFOpLa  (None, 128, 128, 3)       0
 mbda)

 inception_v3 (Functional)   (None, 2, 2, 2048)        21802784

 global_average_pooling2d_4  (None, 2048)              0
  (GlobalAveragePooling2D)

 dropout_4 (Dropout)         (None, 2048)              0

 dense_5 (Dense)             (None, 3)                 6147

=================================================================
Total params: 21808931 (83.19 MB)
Trainable params: 19632515 (74.89 MB)
Non-trainable params: 2176416 (8.30 MB)
_________________________________________________________________
```

```
fine_tune_epochs = 10
total_epochs = initial_epochs + fine_tune_epochs

history_fine = model.fit(train_dataset,
                         epochs = total_epochs,
                         initial_epoch = history.epoch[-1],
                         validation_data = validation_dataset)
```

```
Epoch 20/30
23/23 [==============================] - 33s 153ms/step - loss: 4.5181 - accuracy: 0.4891 - val_loss: 6.8590 - val_accuracy: 0.3548
Epoch 21/30
23/23 [==============================] - 2s 78ms/step - loss: 4.5123 - accuracy: 0.4645 - val_loss: 6.9533 - val_accuracy: 0.3548
Epoch 22/30
23/23 [==============================] - 2s 84ms/step - loss: 4.4939 - accuracy: 0.4891 - val_loss: 6.8377 - val_accuracy: 0.3656
Epoch 23/30
23/23 [==============================] - 2s 80ms/step - loss: 4.8037 - accuracy: 0.4658 - val_loss: 6.8326 - val_accuracy: 0.3763
Epoch 24/30
23/23 [==============================] - 3s 105ms/step - loss: 4.5162 - accuracy: 0.4959 - val_loss: 6.8186 - val_accuracy: 0.3763
Epoch 25/30
23/23 [==============================] - 3s 88ms/step - loss: 4.3260 - accuracy: 0.4918 - val_loss: 6.8142 - val_accuracy: 0.3763
```

```
Epoch 26/30
23/23 [==============================] - 2s 79ms/step - loss: 4.4052 - accuracy: 0.5109 - val_loss: 6.8108 - val_accuracy: 0.3763
Epoch 27/30
23/23 [==============================] - 2s 78ms/step - loss: 3.7747 - accuracy: 0.4986 - val_loss: 6.7981 - val_accuracy: 0.3763
Epoch 28/30
23/23 [==============================] - 2s 78ms/step - loss: 4.1048 - accuracy: 0.5191 - val_loss: 6.7978 - val_accuracy: 0.3763
Epoch 29/30
23/23 [==============================] - 2s 85ms/step - loss: 3.9071 - accuracy: 0.5301 - val_loss: 6.7940 - val_accuracy: 0.3763
Epoch 30/30
23/23 [==============================] - 3s 104ms/step - loss: 4.1700 - accuracy: 0.5000 - val_loss: 6.7941 - val_accuracy: 0.3763
```

```
from tensorflow import keras
model = keras.models.load_model('mejor_modelo.h5')
```

```
n=10 ##Number of image
plt.figure(figsize=(2,2))
plt.imshow(image[n].numpy().astype('uint8'))
plt.title(class_names[labels[n]])
plt.axis("off")
```

```
(-0.5, 127.5, 127.5, -0.5)
```


```
import numpy as np
predictions = model.predict(image)
print(predictions[n])
print("This image most likely belongs to {} with a {:.2f} percent confidence.".format(class_names[np.argmax(predictions[n])], 10 * np.max(
5
```

```
1/1 [==============================] - 1s 1s/step
[ 1.1176975 -1.4487779 -1.0702685]
This image most likely belongs to keyboard with a 11.18 percent confidence.
```

```
n=3 ##Number of image
plt.figure(figsize=(2,2))
plt.imshow(image[n].numpy().astype('uint8'))
plt.title(class_names[labels[n]])
plt.axis("off")
```

```
(-0.5, 127.5, 127.5, -0.5)
```


```
import numpy as np
predictions = model.predict(image)
print(predictions[n])
print("This image most likely belongs to {} with a {:.2f} percent confidence.".format(class_names[np.argmax(predictions[n])], 10 * np.max(
5
```

```
1/1 [==============================] - 0s 43ms/step
[ 1.6942713  -0.07480229  2.0224931 ]
This image most likely belongs to mouse with a 20.22 percent confidence.
```

```
n=20 ##Number of image
plt.figure(figsize=(2,2))
plt.imshow(image[n].numpy().astype('uint8'))
plt.title(class_names[labels[n]])
plt.axis("off")
```

```
(-0.5, 127.5, 127.5, -0.5)
```


```
import numpy as np
predictions = model.predict(image)
print(predictions[n])
print("This image most likely belongs to {} with a {:.2f} percent confidence.".format(class_names[np.argmax(predictions[n])], 10 * np.max(
5
```

```
1/1 [==============================] - 0s 29ms/step
[ 0.00783599  0.5664257  -0.07600397]
```

```
This image most likely belongs to monitor with a 5.66 percent confidence.
```