Intento 3

Este intento es similar al anterior con la única diferencia de que el *frame size* empleado es (96,96) en lugar de (64,64). Además emos usado una tasa de *dropout* algo mayor (del 50% en lugar del 30%)

0. Descarga del dataset

In [1]:
```
# from google.colab import drive
# drive.mount('/content/drive')

train_ds_path  ='C:/Users/Usuario/Documents/Master/Aprendizaje Profundo/Udemy/
test_ds_path   ='C:/Users/Usuario/Documents/Master/Aprendizaje Profundo/Udemy/
cat_or_dog_path='C:/Users/Usuario/Documents/Master/Aprendizaje Profundo/Udemy/


#train_ds_path  ='.\\data\\training_set'
#test_ds_path   ='.\\data\\test_set'
#cat_or_dog_path='.\\data\\single_prediction\\cat_or_dog_1.jpg'
```

Primero, importar las librerías y paquetes

In [2]:
```
from keras.models import Sequential
from keras.layers import Conv2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
from keras.layers import Dense
from keras.layers import BatchNormalization
from keras.layers import GlobalAveragePooling2D
from keras.layers import Dropout
# Nota, algunas capas no están importadas aquí y se importan directamente en e

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

Fijamos seeds para poder reproducir resultados (aunque aun así a veces no lo conseguimos, probablementa haya inicializaciones que no dependan de estas seeds)

In [3]:
```
os.environ['PYTHONHASHSEED'] = '0'
np.random.seed(42)
rn.seed(12345)
tf.random.set_seed(1234)
```

1. Construcción del modelo CNN añadiendo un tamaño de imagen mayor

El tamaño de imagen que emplearemos será de 96x96, y el dropout rate es del 30%

In [4]:
```
frame_size = (96, 96)
"""
esta dupla nos permitirá parametrizar la resolución
de entrada de las imágenes
"""

def crear_clasificador_intento2():
    classifier = Sequential()
    classifier.add(Conv2D(filters = 32,kernel_size = (3, 3),
                          input_shape = (*frame_size, 3), activation = "relu"))
    classifier.add(MaxPooling2D(pool_size = (2,2)))
    classifier.add(Conv2D(filters = 32,kernel_size = (3, 3), activation = "rel
    classifier.add(MaxPooling2D(pool_size = (2,2)))
    classifier.add(Flatten())
    classifier.add(Dense(units = 128, activation = "relu"))
    classifier.add(Dropout(0.3))
    classifier.add(Dense(units = 1, activation = "sigmoid"))
    return classifier
```

2. Entrenamiento del intento 2

En primer lugar instanciamos nuestro modelo y compilamos usando:

- Un optimizador Adam. La learning rate que emplea por defecto es 0.001
- Binary cross entropy como función de coste a minimizar.

In [5]:
```
classifier = crear_clasificador_intento2()
classifier.compile(optimizer = "adam",
                   loss = "binary_crossentropy",
                   metrics = ["accuracy"])
classifier.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 94, 94, 32)        896
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 47, 47, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 45, 45, 32)        9248
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 22, 22, 32)        0
_________________________________________________________________
flatten (Flatten)            (None, 15488)             0
_________________________________________________________________
dense (Dense)                (None, 128)               1982592
_________________________________________________________________
dropout (Dropout)            (None, 128)               0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 129
=================================================================
Total params: 1,992,865
Trainable params: 1,992,865
```

```
Non-trainable params: 0
```

En segundo lugar, generamos los datasets de entrenamiento y test. Emplearemos un tamaño de batch de 32

In [6]:
```
from keras.preprocessing.image import ImageDataGenerator

batch_size=32

train_datagen = ImageDataGenerator(
        rescale=1./255,
        shear_range=0.2,
        zoom_range=0.2,
        horizontal_flip=True)

test_datagen = ImageDataGenerator(rescale=1./255)

training_dataset = train_datagen.flow_from_directory(train_ds_path,
                                                     target_size=frame_size,
                                                     batch_size=batch_size,
                                                     class_mode='binary')

testing_dataset = test_datagen.flow_from_directory(test_ds_path,
                                                   target_size=frame_size,
                                                   batch_size=batch_size,
                                                   class_mode='binary')
```

```
Found 8000 images belonging to 2 classes.
Found 2000 images belonging to 2 classes.
```

Definimos el callback y realizamos el entrenamiento con las condiciones descritas en la sección de introducción.

In [7]:
```
callbacks = [
    tf.keras.callbacks.EarlyStopping(monitor='val_loss', min_delta = 0, patie
]

history1 = classifier.fit(x=training_dataset,
                          steps_per_epoch=8000/batch_size,
                          epochs=100,
                          validation_data=testing_dataset,
                          validation_steps=2000/batch_size,
                          workers=4) # "Si pedimos más de un proceso el rendim
```

```
Epoch 1/100
250/250 [==============================] - 31s 123ms/step - loss: 0.6808 - acc
uracy: 0.5705 - val_loss: 0.6015 - val_accuracy: 0.6755
Epoch 2/100
250/250 [==============================] - 31s 124ms/step - loss: 0.5921 - acc
uracy: 0.6837 - val_loss: 0.5527 - val_accuracy: 0.7220
Epoch 3/100
250/250 [==============================] - 30s 120ms/step - loss: 0.5401 - acc
uracy: 0.7212 - val_loss: 0.5366 - val_accuracy: 0.7390
Epoch 4/100
250/250 [==============================] - 36s 144ms/step - loss: 0.5216 - acc
uracy: 0.7476 - val_loss: 0.4993 - val_accuracy: 0.7560
Epoch 5/100
250/250 [==============================] - 48s 191ms/step - loss: 0.4813 - acc
uracy: 0.7595 - val_loss: 0.4875 - val_accuracy: 0.7605
```

```
Epoch 6/100
250/250 [==============================] - 51s 202ms/step - loss: 0.4782 - acc
uracy: 0.7798 - val_loss: 0.5000 - val_accuracy: 0.7655
Epoch 7/100
250/250 [==============================] - 52s 206ms/step - loss: 0.4559 - acc
uracy: 0.7773 - val_loss: 0.4705 - val_accuracy: 0.7790
Epoch 8/100
250/250 [==============================] - 49s 196ms/step - loss: 0.4415 - acc
uracy: 0.7942 - val_loss: 0.4802 - val_accuracy: 0.7800
Epoch 9/100
250/250 [==============================] - 51s 200ms/step - loss: 0.4200 - acc
uracy: 0.8071 - val_loss: 0.4457 - val_accuracy: 0.7955
Epoch 10/100
250/250 [==============================] - 51s 204ms/step - loss: 0.4033 - acc
uracy: 0.8203 - val_loss: 0.4808 - val_accuracy: 0.7970
Epoch 11/100
250/250 [==============================] - 51s 203ms/step - loss: 0.3832 - acc
uracy: 0.8235 - val_loss: 0.4549 - val_accuracy: 0.7925
Epoch 12/100
250/250 [==============================] - 51s 204ms/step - loss: 0.3679 - acc
uracy: 0.8400 - val_loss: 0.4687 - val_accuracy: 0.7995
Epoch 13/100
250/250 [==============================] - 52s 205ms/step - loss: 0.3771 - acc
uracy: 0.8321 - val_loss: 0.4950 - val_accuracy: 0.7895
Epoch 14/100
250/250 [==============================] - 51s 204ms/step - loss: 0.3499 - acc
uracy: 0.8455 - val_loss: 0.4840 - val_accuracy: 0.7935
Epoch 15/100
250/250 [==============================] - 51s 203ms/step - loss: 0.3513 - acc
uracy: 0.8403 - val_loss: 0.4694 - val_accuracy: 0.7990
Epoch 16/100
250/250 [==============================] - 51s 202ms/step - loss: 0.3377 - acc
uracy: 0.8500 - val_loss: 0.4739 - val_accuracy: 0.8055
Epoch 17/100
250/250 [==============================] - 52s 206ms/step - loss: 0.3351 - acc
uracy: 0.8514 - val_loss: 0.4828 - val_accuracy: 0.8080
Epoch 18/100
250/250 [==============================] - 50s 200ms/step - loss: 0.3082 - acc
uracy: 0.8712 - val_loss: 0.4865 - val_accuracy: 0.8070
Epoch 19/100
250/250 [==============================] - 51s 204ms/step - loss: 0.3013 - acc
uracy: 0.8748 - val_loss: 0.5454 - val_accuracy: 0.7910
Epoch 20/100
250/250 [==============================] - 52s 206ms/step - loss: 0.2903 - acc
uracy: 0.8783 - val_loss: 0.4950 - val_accuracy: 0.8035
Epoch 21/100
250/250 [==============================] - 51s 202ms/step - loss: 0.2914 - acc
uracy: 0.8778 - val_loss: 0.5043 - val_accuracy: 0.8130
Epoch 22/100
250/250 [==============================] - 52s 205ms/step - loss: 0.2827 - acc
uracy: 0.8810 - val_loss: 0.4590 - val_accuracy: 0.8210 2 - ETA: 1s - loss: 0.
282
Epoch 23/100
250/250 [==============================] - 52s 206ms/step - loss: 0.2637 - acc
uracy: 0.8913 - val_loss: 0.5329 - val_accuracy: 0.7905
Epoch 24/100
250/250 [==============================] - 52s 206ms/step - loss: 0.2646 - acc
uracy: 0.8918 - val_loss: 0.5540 - val_accuracy: 0.8045
Epoch 25/100
250/250 [==============================] - 51s 204ms/step - loss: 0.2652 - acc
uracy: 0.8937 - val_loss: 0.5182 - val_accuracy: 0.8100
Epoch 26/100
250/250 [==============================] - 52s 206ms/step - loss: 0.2231 - acc
uracy: 0.9157 - val_loss: 0.5641 - val_accuracy: 0.8080: 0.2179 -  - ETA: 26s
- loss: 0.2183 - accuracy:
```

```
Epoch 27/100
250/250 [==============================] - 51s 203ms/step - loss: 0.2270 - acc
uracy: 0.9084 - val_loss: 0.5729 - val_accuracy: 0.8025
Epoch 28/100
250/250 [==============================] - 51s 204ms/step - loss: 0.2161 - acc
uracy: 0.9080 - val_loss: 0.5320 - val_accuracy: 0.8165
Epoch 29/100
250/250 [==============================] - 47s 188ms/step - loss: 0.1940 - acc
uracy: 0.9197 - val_loss: 0.5607 - val_accuracy: 0.8180
Epoch 30/100
250/250 [==============================] - 30s 119ms/step - loss: 0.2031 - acc
uracy: 0.9195 - val_loss: 0.5985 - val_accuracy: 0.7970
Epoch 31/100
250/250 [==============================] - 30s 119ms/step - loss: 0.1970 - acc
uracy: 0.9207 - val_loss: 0.5450 - val_accuracy: 0.8160
Epoch 32/100
250/250 [==============================] - 30s 119ms/step - loss: 0.1962 - acc
uracy: 0.9181 - val_loss: 0.5906 - val_accuracy: 0.8135
Epoch 33/100
250/250 [==============================] - 30s 120ms/step - loss: 0.1892 - acc
uracy: 0.9225 - val_loss: 0.5804 - val_accuracy: 0.8195
Epoch 34/100
250/250 [==============================] - 30s 120ms/step - loss: 0.1845 - acc
uracy: 0.9238 - val_loss: 0.5593 - val_accuracy: 0.8230
Epoch 35/100
250/250 [==============================] - 30s 121ms/step - loss: 0.1653 - acc
uracy: 0.9316 - val_loss: 0.6065 - val_accuracy: 0.8220
Epoch 36/100
250/250 [==============================] - 30s 121ms/step - loss: 0.1657 - acc
uracy: 0.9317 - val_loss: 0.6520 - val_accuracy: 0.8150
Epoch 37/100
250/250 [==============================] - 30s 121ms/step - loss: 0.1611 - acc
uracy: 0.9365 - val_loss: 0.6151 - val_accuracy: 0.8140
Epoch 38/100
250/250 [==============================] - 31s 123ms/step - loss: 0.1597 - acc
uracy: 0.9357 - val_loss: 0.6640 - val_accuracy: 0.8150
Epoch 39/100
250/250 [==============================] - 31s 122ms/step - loss: 0.1565 - acc
uracy: 0.9342 - val_loss: 0.6872 - val_accuracy: 0.8040
Epoch 40/100
250/250 [==============================] - 30s 120ms/step - loss: 0.1593 - acc
uracy: 0.9425 - val_loss: 0.6456 - val_accuracy: 0.8155
Epoch 41/100
250/250 [==============================] - 31s 122ms/step - loss: 0.1539 - acc
uracy: 0.9433 - val_loss: 0.6664 - val_accuracy: 0.8180
Epoch 42/100
250/250 [==============================] - 30s 119ms/step - loss: 0.1421 - acc
uracy: 0.9470 - val_loss: 0.6420 - val_accuracy: 0.8060
Epoch 43/100
250/250 [==============================] - 30s 120ms/step - loss: 0.1532 - acc
uracy: 0.9403 - val_loss: 0.7020 - val_accuracy: 0.8305
Epoch 44/100
250/250 [==============================] - 30s 121ms/step - loss: 0.1328 - acc
uracy: 0.9510 - val_loss: 0.7258 - val_accuracy: 0.8090
Epoch 45/100
250/250 [==============================] - 30s 120ms/step - loss: 0.1349 - acc
uracy: 0.9466 - val_loss: 0.7336 - val_accuracy: 0.8020
Epoch 46/100
250/250 [==============================] - 30s 119ms/step - loss: 0.1359 - acc
uracy: 0.9474 - val_loss: 0.6755 - val_accuracy: 0.8160
Epoch 47/100
250/250 [==============================] - 30s 120ms/step - loss: 0.1291 - acc
uracy: 0.9494 - val_loss: 0.6929 - val_accuracy: 0.8190
Epoch 48/100
250/250 [==============================] - 30s 119ms/step - loss: 0.1347 - acc
```

```
uracy: 0.9446 - val_loss: 0.8148 - val_accuracy: 0.8080
Epoch 49/100
250/250 [==============================] - 30s 120ms/step - loss: 0.1311 - acc
uracy: 0.9527 - val_loss: 0.7355 - val_accuracy: 0.8160
Epoch 50/100
250/250 [==============================] - 30s 121ms/step - loss: 0.1205 - acc
uracy: 0.9564 - val_loss: 0.7563 - val_accuracy: 0.8085
Epoch 51/100
250/250 [==============================] - 30s 121ms/step - loss: 0.1175 - acc
uracy: 0.9570 - val_loss: 0.6976 - val_accuracy: 0.8150
Epoch 52/100
250/250 [==============================] - 30s 120ms/step - loss: 0.1385 - acc
uracy: 0.9458 - val_loss: 0.7039 - val_accuracy: 0.8205
Epoch 53/100
250/250 [==============================] - 30s 121ms/step - loss: 0.1217 - acc
uracy: 0.9521 - val_loss: 0.6861 - val_accuracy: 0.8195
Epoch 54/100
250/250 [==============================] - 30s 120ms/step - loss: 0.1074 - acc
uracy: 0.9563 - val_loss: 0.7655 - val_accuracy: 0.8030
Epoch 55/100
250/250 [==============================] - 30s 120ms/step - loss: 0.1110 - acc
uracy: 0.9606 - val_loss: 0.8339 - val_accuracy: 0.8015
Epoch 56/100
250/250 [==============================] - 31s 122ms/step - loss: 0.1006 - acc
uracy: 0.9606 - val_loss: 0.7995 - val_accuracy: 0.8090
Epoch 57/100
250/250 [==============================] - 30s 121ms/step - loss: 0.1011 - acc
uracy: 0.9592 - val_loss: 0.7548 - val_accuracy: 0.8260
Epoch 58/100
250/250 [==============================] - 30s 119ms/step - loss: 0.1106 - acc
uracy: 0.9574 - val_loss: 0.8070 - val_accuracy: 0.8180
Epoch 59/100
250/250 [==============================] - 30s 120ms/step - loss: 0.1029 - acc
uracy: 0.9596 - val_loss: 0.7569 - val_accuracy: 0.8225
Epoch 60/100
250/250 [==============================] - 30s 120ms/step - loss: 0.0976 - acc
uracy: 0.9592 - val_loss: 0.7856 - val_accuracy: 0.8130
Epoch 61/100
250/250 [==============================] - 30s 119ms/step - loss: 0.1067 - acc
uracy: 0.9613 - val_loss: 0.7978 - val_accuracy: 0.8155
Epoch 62/100
250/250 [==============================] - 30s 120ms/step - loss: 0.1011 - acc
uracy: 0.9649 - val_loss: 0.8525 - val_accuracy: 0.8250
Epoch 63/100
250/250 [==============================] - 30s 119ms/step - loss: 0.0950 - acc
uracy: 0.9646 - val_loss: 0.8882 - val_accuracy: 0.8065
Epoch 64/100
250/250 [==============================] - 30s 119ms/step - loss: 0.0997 - acc
uracy: 0.9637 - val_loss: 0.7904 - val_accuracy: 0.8185
Epoch 65/100
250/250 [==============================] - 30s 121ms/step - loss: 0.1111 - acc
uracy: 0.9585 - val_loss: 0.8236 - val_accuracy: 0.8145
Epoch 66/100
250/250 [==============================] - 31s 124ms/step - loss: 0.0889 - acc
uracy: 0.9684 - val_loss: 0.8733 - val_accuracy: 0.8200
Epoch 67/100
250/250 [==============================] - 31s 123ms/step - loss: 0.0927 - acc
uracy: 0.9666 - val_loss: 0.9019 - val_accuracy: 0.8110
Epoch 68/100
250/250 [==============================] - 30s 120ms/step - loss: 0.0903 - acc
uracy: 0.9667 - val_loss: 0.8887 - val_accuracy: 0.8165
Epoch 69/100
250/250 [==============================] - 30s 119ms/step - loss: 0.0943 - acc
uracy: 0.9666 - val_loss: 0.8939 - val_accuracy: 0.8015
Epoch 70/100
```

```
250/250 [==============================] - 30s 120ms/step - loss: 0.0844 - acc
uracy: 0.9669 - val_loss: 0.8512 - val_accuracy: 0.8215
Epoch 71/100
250/250 [==============================] - 31s 123ms/step - loss: 0.0839 - acc
uracy: 0.9692 - val_loss: 0.8960 - val_accuracy: 0.8235
Epoch 72/100
250/250 [==============================] - 31s 121ms/step - loss: 0.0812 - acc
uracy: 0.9709 - val_loss: 0.9073 - val_accuracy: 0.8240
Epoch 73/100
250/250 [==============================] - 31s 124ms/step - loss: 0.0922 - acc
uracy: 0.9651 - val_loss: 0.8398 - val_accuracy: 0.8200
Epoch 74/100
250/250 [==============================] - 31s 122ms/step - loss: 0.0847 - acc
uracy: 0.9671 - val_loss: 0.9049 - val_accuracy: 0.7940
Epoch 75/100
250/250 [==============================] - 31s 122ms/step - loss: 0.0947 - acc
uracy: 0.9645 - val_loss: 0.8411 - val_accuracy: 0.8140
Epoch 76/100
250/250 [==============================] - 31s 125ms/step - loss: 0.0913 - acc
uracy: 0.9679 - val_loss: 0.9235 - val_accuracy: 0.8090
Epoch 77/100
250/250 [==============================] - 30s 121ms/step - loss: 0.0865 - acc
uracy: 0.9651 - val_loss: 0.9955 - val_accuracy: 0.8150
Epoch 78/100
250/250 [==============================] - 30s 119ms/step - loss: 0.0945 - acc
uracy: 0.9626 - val_loss: 0.8622 - val_accuracy: 0.8120
Epoch 79/100
250/250 [==============================] - 30s 120ms/step - loss: 0.0893 - acc
uracy: 0.9653 - val_loss: 0.9019 - val_accuracy: 0.8105
Epoch 80/100
250/250 [==============================] - 30s 121ms/step - loss: 0.0725 - acc
uracy: 0.9726 - val_loss: 0.9189 - val_accuracy: 0.8130
Epoch 81/100
250/250 [==============================] - 30s 120ms/step - loss: 0.0906 - acc
uracy: 0.9687 - val_loss: 1.0249 - val_accuracy: 0.7915
Epoch 82/100
250/250 [==============================] - 30s 119ms/step - loss: 0.0854 - acc
uracy: 0.9687 - val_loss: 0.8745 - val_accuracy: 0.8220
Epoch 83/100
250/250 [==============================] - 30s 120ms/step - loss: 0.0860 - acc
uracy: 0.9700 - val_loss: 0.8928 - val_accuracy: 0.8075
Epoch 84/100
250/250 [==============================] - 30s 121ms/step - loss: 0.0865 - acc
uracy: 0.9706 - val_loss: 0.8927 - val_accuracy: 0.8125
Epoch 85/100
250/250 [==============================] - 31s 122ms/step - loss: 0.0680 - acc
uracy: 0.9740 - val_loss: 0.8738 - val_accuracy: 0.8125
Epoch 86/100
250/250 [==============================] - 31s 122ms/step - loss: 0.0706 - acc
uracy: 0.9758 - val_loss: 0.9148 - val_accuracy: 0.8150
Epoch 87/100
250/250 [==============================] - 30s 121ms/step - loss: 0.0596 - acc
uracy: 0.9787 - val_loss: 0.9912 - val_accuracy: 0.8170
Epoch 88/100
250/250 [==============================] - 30s 119ms/step - loss: 0.0663 - acc
uracy: 0.9757 - val_loss: 0.9598 - val_accuracy: 0.8110
Epoch 89/100
250/250 [==============================] - 30s 120ms/step - loss: 0.0930 - acc
uracy: 0.9657 - val_loss: 0.9194 - val_accuracy: 0.8210
Epoch 90/100
250/250 [==============================] - 31s 122ms/step - loss: 0.0823 - acc
uracy: 0.9687 - val_loss: 0.9145 - val_accuracy: 0.8015
Epoch 91/100
250/250 [==============================] - 30s 120ms/step - loss: 0.0931 - acc
uracy: 0.9674 - val_loss: 0.9383 - val_accuracy: 0.8255
```

```
Epoch 92/100
250/250 [==============================] - 30s 119ms/step - loss: 0.0688 - acc
uracy: 0.9746 - val_loss: 1.0724 - val_accuracy: 0.8125
Epoch 93/100
250/250 [==============================] - 30s 119ms/step - loss: 0.0673 - acc
uracy: 0.9736 - val_loss: 0.9429 - val_accuracy: 0.8235
Epoch 94/100
250/250 [==============================] - 31s 123ms/step - loss: 0.0885 - acc
uracy: 0.9682 - val_loss: 0.9551 - val_accuracy: 0.8175
Epoch 95/100
250/250 [==============================] - 30s 119ms/step - loss: 0.0729 - acc
uracy: 0.9745 - val_loss: 0.9136 - val_accuracy: 0.8145
Epoch 96/100
250/250 [==============================] - 31s 122ms/step - loss: 0.0694 - acc
uracy: 0.9735 - val_loss: 1.0607 - val_accuracy: 0.7930
Epoch 97/100
250/250 [==============================] - 30s 119ms/step - loss: 0.0754 - acc
uracy: 0.9748 - val_loss: 1.0211 - val_accuracy: 0.8180
Epoch 98/100
250/250 [==============================] - 30s 119ms/step - loss: 0.0773 - acc
uracy: 0.9726 - val_loss: 1.1429 - val_accuracy: 0.8020
Epoch 99/100
250/250 [==============================] - 32s 126ms/step - loss: 0.0687 - acc
uracy: 0.9747 - val_loss: 0.9836 - val_accuracy: 0.8160
Epoch 100/100
```

Ploteamos el resultado

In [8]:

```
def plot_resultados_training(history):
    fig, axes = plt.subplots(1,2, figsize=(18,6))
    axes[0].plot(history.history['accuracy'], label='Train')
    axes[0].plot(history.history['val_accuracy'], label='Validation')
    axes[0].legend()
    axes[0].set_title('Accuracy')
    axes[1].plot(history.history['loss'], label='Train')
    axes[1].plot(history.history['val_loss'], label='Validation')
    axes[1].set_title('Cross entropy')
plot_resultados_training(history)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-8-a72c62acc41b> in <module>
      8     axes[1].plot(history.history['val_loss'], label='Validation')
      9     axes[1].set_title('Cross entropy')
---> 10 plot_resultados_training(history)

NameError: name 'history' is not defined
```

In [ ]:

```
classifier.save('.\\models\\clasificador3')
```

Comentario

La divergencia entre las precisiones de entrenamiento y validación ha empeorado. Aunque cabe destacar que la red ha mejorado su tasa de precisión en entrenamiento, seguramente debido a que las imágenes aportan más información útil.

Propuesta de mejora