# Primera ejecución del modelo CNN simple

## Cambios vs. Version anterior:

* 1. Se añade una nueva capa convolucional, con su respectiva capa de MaxPooling. Esta nueva capa Conv2D tiene 64 filtros en lugar de los 32 de las dos campas anteriores. El tamaño del kernel se mantiene en 3x3.
  2. Se modifican los rangos del generador de imagenes de trainig, tales como el rango de rotación (de 0.2 a 0.3), el rango de shear/corte y el zoom.
  3. Se establecen 30 épocas.
  4. Se cambia el tamaño de batch: 32 ejemplos.

## Estructura:

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from tensorflow.keras import Sequential

from tensorflow.keras.layers import Conv2D, MaxPool2D, Dropout, Flatten, \

Dense

import tensorflow as tf

physical\_devices = tf.config.experimental.list\_physical\_devices('GPU')

tf.config.experimental.set\_memory\_growth(physical\_devices[0], True)

training\_datagen = ImageDataGenerator(rescale=1. / 255,

rotation\_range=0.3,

shear\_range=0.1,

zoom\_range=[0.90, 1.2],

horizontal\_flip=True)

test\_datagen = ImageDataGenerator(rescale=1. / 255)

training\_path = '/home/ruben/workspace/tfg/deep-learning-facial-recognition/data/age/training'

test\_path = '/home/ruben/workspace/tfg/deep-learning-facial-recognition/data/age/test'

training\_set = training\_datagen.flow\_from\_directory(training\_path,

target\_size=(64, 64),

batch\_size=32,

class\_mode='categorical',

shuffle=True,

seed=42)

test\_set = test\_datagen.flow\_from\_directory(test\_path,

target\_size=(64, 64),

batch\_size=32,

class\_mode='categorical',

shuffle=True,

seed=42)

age\_classifier = Sequential()

age\_classifier.add(

Conv2D(filters=32, kernel\_size=(3, 3), input\_shape=(64, 64, 3),

activation='relu'))

age\_classifier.add(MaxPool2D(pool\_size=(2, 2)))

age\_classifier.add(

Conv2D(filters=32, kernel\_size=(3, 3), activation='relu'))

age\_classifier.add(MaxPool2D(pool\_size=(2, 2)))

age\_classifier.add(

Conv2D(filters=64, kernel\_size=(3, 3), activation='relu'))

age\_classifier.add(MaxPool2D(pool\_size=(2, 2)))

age\_classifier.add(Flatten())

age\_classifier.add(Dense(units=128, activation='relu'))

age\_classifier.add(Dropout(0.3))

age\_classifier.add(Dense(units=64, activation='relu'))

age\_classifier.add(Dropout(0.3))

age\_classifier.add(Dense(units=6, activation='softmax'))

age\_classifier.compile(optimizer='adam', loss='categorical\_crossentropy',

metrics=['accuracy'])

age\_classifier.fit(training\_set,

steps\_per\_epoch=(18966 // 32),

epochs=30,

validation\_data=test\_set,

validation\_steps=(4742 // 32))

## Resultado:

1. Epoch 1/30
2. 2020-05-20 16:02:21.565992: I tensorflow/stream\_executor/platform/default/dso\_loader.cc:44] Successfully opened dynamic library libcublas.so.10
3. 2020-05-20 16:02:21.710407: I tensorflow/stream\_executor/platform/default/dso\_loader.cc:44] Successfully opened dynamic library libcudnn.so.7
4. 592/592 [==============================] - 37s 62ms/step - loss: 1.3655 - accuracy: 0.4470 - val\_loss: 1.1408 - val\_accuracy: 0.5177
5. Epoch 2/30
6. 592/592 [==============================] - 38s 64ms/step - loss: 1.1103 - accuracy: 0.5284 - val\_loss: 1.0079 - val\_accuracy: 0.5515
7. Epoch 3/30
8. 592/592 [==============================] - 38s 65ms/step - loss: 1.0241 - accuracy: 0.5581 - val\_loss: 0.9437 - val\_accuracy: 0.5910
9. Epoch 4/30
10. 592/592 [==============================] - 38s 64ms/step - loss: 0.9725 - accuracy: 0.5766 - val\_loss: 0.9262 - val\_accuracy: 0.5842
11. Epoch 5/30
12. 592/592 [==============================] - 39s 66ms/step - loss: 0.9412 - accuracy: 0.5912 - val\_loss: 0.8713 - val\_accuracy: 0.6244
13. Epoch 6/30
14. 592/592 [==============================] - 38s 65ms/step - loss: 0.9108 - accuracy: 0.6018 - val\_loss: 0.8692 - val\_accuracy: 0.6210
15. Epoch 7/30
16. 592/592 [==============================] - 39s 65ms/step - loss: 0.8972 - accuracy: 0.6062 - val\_loss: 0.8624 - val\_accuracy: 0.6178
17. Epoch 8/30
18. 592/592 [==============================] - 39s 65ms/step - loss: 0.8768 - accuracy: 0.6171 - val\_loss: 0.8712 - val\_accuracy: 0.6161
19. Epoch 9/30
20. 592/592 [==============================] - 38s 65ms/step - loss: 0.8632 - accuracy: 0.6200 - val\_loss: 0.8353 - val\_accuracy: 0.6356
21. Epoch 10/30
22. 592/592 [==============================] - 39s 65ms/step - loss: 0.8459 - accuracy: 0.6349 - val\_loss: 0.8350 - val\_accuracy: 0.6423
23. Epoch 11/30
24. 592/592 [==============================] - 39s 66ms/step - loss: 0.8376 - accuracy: 0.6354 - val\_loss: 0.8134 - val\_accuracy: 0.6467
25. Epoch 12/30
26. 592/592 [==============================] - 39s 65ms/step - loss: 0.8289 - accuracy: 0.6365 - val\_loss: 0.8139 - val\_accuracy: 0.6444
27. Epoch 13/30
28. 592/592 [==============================] - 39s 66ms/step - loss: 0.8267 - accuracy: 0.6403 - val\_loss: 0.8216 - val\_accuracy: 0.6423
29. Epoch 14/30
30. 592/592 [==============================] - 39s 65ms/step - loss: 0.8107 - accuracy: 0.6442 - val\_loss: 0.8110 - val\_accuracy: 0.6421
31. Epoch 15/30
32. 592/592 [==============================] - 39s 66ms/step - loss: 0.7994 - accuracy: 0.6495 - val\_loss: 0.8089 - val\_accuracy: 0.6480
33. Epoch 16/30
34. 592/592 [==============================] - 39s 65ms/step - loss: 0.7999 - accuracy: 0.6498 - val\_loss: 0.7926 - val\_accuracy: 0.6499
35. Epoch 17/30
36. 592/592 [==============================] - 39s 66ms/step - loss: 0.7915 - accuracy: 0.6521 - val\_loss: 0.7928 - val\_accuracy: 0.6516
37. Epoch 18/30
38. 592/592 [==============================] - 39s 66ms/step - loss: 0.7806 - accuracy: 0.6533 - val\_loss: 0.7985 - val\_accuracy: 0.6539
39. Epoch 19/30
40. 592/592 [==============================] - 39s 67ms/step - loss: 0.7823 - accuracy: 0.6562 - val\_loss: 0.7933 - val\_accuracy: 0.6527
41. Epoch 20/30
42. 592/592 [==============================] - 39s 66ms/step - loss: 0.7714 - accuracy: 0.6607 - val\_loss: 0.7862 - val\_accuracy: 0.6571
43. Epoch 21/30
44. 592/592 [==============================] - 39s 66ms/step - loss: 0.7669 - accuracy: 0.6607 - val\_loss: 0.7916 - val\_accuracy: 0.6592
45. Epoch 22/30
46. 592/592 [==============================] - 40s 68ms/step - loss: 0.7538 - accuracy: 0.6685 - val\_loss: 0.7781 - val\_accuracy: 0.6592
47. Epoch 23/30
48. 592/592 [==============================] - 39s 66ms/step - loss: 0.7578 - accuracy: 0.6682 - val\_loss: 0.7864 - val\_accuracy: 0.6548
49. Epoch 24/30
50. 592/592 [==============================] - 39s 65ms/step - loss: 0.7586 - accuracy: 0.6675 - val\_loss: 0.7890 - val\_accuracy: 0.6562
51. Epoch 25/30
52. 592/592 [==============================] - 39s 66ms/step - loss: 0.7479 - accuracy: 0.6687 - val\_loss: 0.8106 - val\_accuracy: 0.6510
53. Epoch 26/30
54. 592/592 [==============================] - 39s 66ms/step - loss: 0.7498 - accuracy: 0.6704 - val\_loss: 0.7725 - val\_accuracy: 0.6670
55. Epoch 27/30
56. 592/592 [==============================] - 39s 66ms/step - loss: 0.7421 - accuracy: 0.6729 - val\_loss: 0.7929 - val\_accuracy: 0.6605
57. Epoch 28/30
58. 592/592 [==============================] - 40s 67ms/step - loss: 0.7340 - accuracy: 0.6727 - val\_loss: 0.7914 - val\_accuracy: 0.6516
59. Epoch 29/30
60. 592/592 [==============================] - 39s 66ms/step - loss: 0.7305 - accuracy: 0.6787 - val\_loss: 0.7761 - val\_accuracy: 0.6603
61. Epoch 30/30
62. 592/592 [==============================] - 40s 67ms/step - loss: 0.7316 - accuracy: 0.6782 - val\_loss: 0.7769 - val\_accuracy: 0.6643
63. Process finished with exit code 0

## Conclusiones y Resultado vs. Version Anterior:

Parece que se ha reducido el overfitting, teniendo tasas de acierto muy similares, tanto en fase de entreno como en fase de test.

Aún así, esta precisión podrías ser más alta: No es mala, pues acierta 2 de cada 3 imágenes, pero sí mejorable.

Valores actuales:

Acc. Train: 67.82

Acc. Test: 66.43%