# Primera ejecución del modelo CNN simple

## Cambios vs. Version anterior:

* 1. Sin cambios en la arquitectura de la CNN.
  2. Se aumenta ligeramente el número de épocas, de 25 a 100.
  3. Se aumenta el tamaño del batch, de 32 a 64 ejemplos.

## Estructura:

* 1. 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=0.2,

shear\_range=0.05,

zoom\_range=[0.95, 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=64,

class\_mode='categorical',

shuffle=True,

seed=42)

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

target\_size=(64, 64),

batch\_size=64,

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(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 // 64),

epochs=100,

validation\_data=test\_set,

validation\_steps=(4742 // 64))

## Resultado:

1. Epoch 1/100
2. 2020-05-20 15:01:10.471322: I tensorflow/stream\_executor/platform/default/dso\_loader.cc:44] Successfully opened dynamic library libcublas.so.10
3. 2020-05-20 15:01:10.601890: I tensorflow/stream\_executor/platform/default/dso\_loader.cc:44] Successfully opened dynamic library libcudnn.so.7
4. 296/296 [==============================] - 36s 121ms/step - loss: 1.3469 - accuracy: 0.4502 - val\_loss: 1.0675 - val\_accuracy: 0.5412
5. Epoch 2/100
6. 296/296 [==============================] - 37s 125ms/step - loss: 1.1061 - accuracy: 0.5310 - val\_loss: 0.9898 - val\_accuracy: 0.5703
7. Epoch 3/100
8. 296/296 [==============================] - 37s 125ms/step - loss: 1.0209 - accuracy: 0.5597 - val\_loss: 0.9624 - val\_accuracy: 0.5830
9. Epoch 4/100
10. 296/296 [==============================] - 38s 127ms/step - loss: 0.9646 - accuracy: 0.5834 - val\_loss: 0.9241 - val\_accuracy: 0.5952
11. Epoch 5/100
12. 296/296 [==============================] - 38s 129ms/step - loss: 0.9262 - accuracy: 0.5918 - val\_loss: 0.8617 - val\_accuracy: 0.6189
13. Epoch 6/100
14. 296/296 [==============================] - 38s 128ms/step - loss: 0.9041 - accuracy: 0.6099 - val\_loss: 0.8520 - val\_accuracy: 0.6248
15. Epoch 7/100
16. 296/296 [==============================] - 38s 127ms/step - loss: 0.8798 - accuracy: 0.6176 - val\_loss: 0.8897 - val\_accuracy: 0.6035
17. Epoch 8/100
18. 296/296 [==============================] - 38s 129ms/step - loss: 0.8617 - accuracy: 0.6216 - val\_loss: 0.8580 - val\_accuracy: 0.6166
19. Epoch 9/100
20. 296/296 [==============================] - 40s 135ms/step - loss: 0.8387 - accuracy: 0.6334 - val\_loss: 0.8211 - val\_accuracy: 0.6478
21. Epoch 10/100
22. 296/296 [==============================] - 44s 150ms/step - loss: 0.8279 - accuracy: 0.6367 - val\_loss: 0.8259 - val\_accuracy: 0.6328
23. Epoch 11/100
24. 296/296 [==============================] - 46s 155ms/step - loss: 0.8135 - accuracy: 0.6465 - val\_loss: 0.8223 - val\_accuracy: 0.6484
25. Epoch 12/100
26. 296/296 [==============================] - 42s 141ms/step - loss: 0.8030 - accuracy: 0.6469 - val\_loss: 0.8037 - val\_accuracy: 0.6478
27. Epoch 13/100
28. 296/296 [==============================] - 40s 134ms/step - loss: 0.7966 - accuracy: 0.6539 - val\_loss: 0.8075 - val\_accuracy: 0.6505
29. Epoch 14/100
30. 296/296 [==============================] - 40s 134ms/step - loss: 0.7781 - accuracy: 0.6571 - val\_loss: 0.8068 - val\_accuracy: 0.6571
31. Epoch 15/100
32. 296/296 [==============================] - 40s 134ms/step - loss: 0.7721 - accuracy: 0.6671 - val\_loss: 0.7828 - val\_accuracy: 0.6620
33. Epoch 16/100
34. 296/296 [==============================] - 39s 133ms/step - loss: 0.7636 - accuracy: 0.6660 - val\_loss: 0.7869 - val\_accuracy: 0.6562
35. Epoch 17/100
36. 296/296 [==============================] - 41s 139ms/step - loss: 0.7524 - accuracy: 0.6696 - val\_loss: 0.7882 - val\_accuracy: 0.6620
37. Epoch 18/100
38. 296/296 [==============================] - 41s 140ms/step - loss: 0.7510 - accuracy: 0.6686 - val\_loss: 0.7976 - val\_accuracy: 0.6645
39. Epoch 19/100
40. 296/296 [==============================] - 40s 136ms/step - loss: 0.7417 - accuracy: 0.6774 - val\_loss: 0.8005 - val\_accuracy: 0.6524
41. Epoch 20/100
42. 296/296 [==============================] - 39s 132ms/step - loss: 0.7289 - accuracy: 0.6799 - val\_loss: 0.7875 - val\_accuracy: 0.6569
43. Epoch 21/100
44. 296/296 [==============================] - 42s 141ms/step - loss: 0.7380 - accuracy: 0.6746 - val\_loss: 0.7713 - val\_accuracy: 0.6750
45. Epoch 22/100
46. 296/296 [==============================] - 40s 134ms/step - loss: 0.7277 - accuracy: 0.6835 - val\_loss: 0.7681 - val\_accuracy: 0.6708
47. Epoch 23/100
48. 296/296 [==============================] - 44s 147ms/step - loss: 0.7098 - accuracy: 0.6853 - val\_loss: 0.7750 - val\_accuracy: 0.6683
49. Epoch 24/100
50. 296/296 [==============================] - 46s 155ms/step - loss: 0.7147 - accuracy: 0.6855 - val\_loss: 0.7861 - val\_accuracy: 0.6672
51. Epoch 25/100
52. 296/296 [==============================] - 46s 154ms/step - loss: 0.7072 - accuracy: 0.6923 - val\_loss: 0.7878 - val\_accuracy: 0.6601
53. Epoch 26/100
54. 296/296 [==============================] - 44s 147ms/step - loss: 0.7001 - accuracy: 0.6944 - val\_loss: 0.8070 - val\_accuracy: 0.6516
55. Epoch 27/100
56. 296/296 [==============================] - 39s 131ms/step - loss: 0.7002 - accuracy: 0.6957 - val\_loss: 0.7833 - val\_accuracy: 0.6562
57. Epoch 28/100
58. 296/296 [==============================] - 39s 131ms/step - loss: 0.6914 - accuracy: 0.7007 - val\_loss: 0.7968 - val\_accuracy: 0.6535
59. Epoch 29/100
60. 296/296 [==============================] - 39s 131ms/step - loss: 0.6818 - accuracy: 0.7000 - val\_loss: 0.7881 - val\_accuracy: 0.6691
61. Epoch 30/100
62. 296/296 [==============================] - 39s 130ms/step - loss: 0.6766 - accuracy: 0.7021 - val\_loss: 0.7802 - val\_accuracy: 0.6691
63. Epoch 31/100
64. 296/296 [==============================] - 38s 130ms/step - loss: 0.6675 - accuracy: 0.7106 - val\_loss: 0.7783 - val\_accuracy: 0.6708
65. Epoch 32/100
66. 296/296 [==============================] - 39s 131ms/step - loss: 0.6676 - accuracy: 0.7074 - val\_loss: 0.8101 - val\_accuracy: 0.6565
67. Epoch 33/100
68. 296/296 [==============================] - 38s 129ms/step - loss: 0.6592 - accuracy: 0.7103 - val\_loss: 0.7875 - val\_accuracy: 0.6681
69. Epoch 34/100
70. 296/296 [==============================] - 38s 130ms/step - loss: 0.6615 - accuracy: 0.7085 - val\_loss: 0.8021 - val\_accuracy: 0.6624
71. Epoch 35/100
72. 296/296 [==============================] - 39s 131ms/step - loss: 0.6516 - accuracy: 0.7156 - val\_loss: 0.7893 - val\_accuracy: 0.6613
73. Epoch 36/100
74. 296/296 [==============================] - 39s 131ms/step - loss: 0.6481 - accuracy: 0.7156 - val\_loss: 0.7922 - val\_accuracy: 0.6651
75. Epoch 37/100
76. 296/296 [==============================] - 39s 131ms/step - loss: 0.6487 - accuracy: 0.7179 - val\_loss: 0.8041 - val\_accuracy: 0.6645
77. Epoch 38/100
78. 296/296 [==============================] - 38s 129ms/step - loss: 0.6496 - accuracy: 0.7162 - val\_loss: 0.7764 - val\_accuracy: 0.6729
79. Epoch 39/100
80. 296/296 [==============================] - 38s 129ms/step - loss: 0.6298 - accuracy: 0.7265 - val\_loss: 0.8006 - val\_accuracy: 0.6647
81. Epoch 40/100
82. 296/296 [==============================] - 38s 129ms/step - loss: 0.6362 - accuracy: 0.7234 - val\_loss: 0.7937 - val\_accuracy: 0.6706
83. Epoch 41/100
84. 296/296 [==============================] - 38s 130ms/step - loss: 0.6350 - accuracy: 0.7236 - val\_loss: 0.7970 - val\_accuracy: 0.6677
85. Epoch 42/100
86. 296/296 [==============================] - 39s 133ms/step - loss: 0.6267 - accuracy: 0.7292 - val\_loss: 0.8030 - val\_accuracy: 0.6679
87. Epoch 43/100
88. 296/296 [==============================] - 39s 130ms/step - loss: 0.6234 - accuracy: 0.7282 - val\_loss: 0.7772 - val\_accuracy: 0.6693
89. Epoch 44/100
90. 296/296 [==============================] - 39s 131ms/step - loss: 0.6149 - accuracy: 0.7325 - val\_loss: 0.7870 - val\_accuracy: 0.6698
91. Epoch 45/100
92. 296/296 [==============================] - 38s 129ms/step - loss: 0.6149 - accuracy: 0.7284 - val\_loss: 0.8031 - val\_accuracy: 0.6710
93. Epoch 46/100
94. 296/296 [==============================] - 40s 135ms/step - loss: 0.6219 - accuracy: 0.7289 - val\_loss: 0.7895 - val\_accuracy: 0.6710
95. Epoch 47/100
96. 296/296 [==============================] - 38s 130ms/step - loss: 0.6062 - accuracy: 0.7331 - val\_loss: 0.8174 - val\_accuracy: 0.6622
97. Epoch 48/100
98. 296/296 [==============================] - 39s 131ms/step - loss: 0.6064 - accuracy: 0.7375 - val\_loss: 0.7920 - val\_accuracy: 0.6704
99. Epoch 49/100
100. 296/296 [==============================] - 38s 130ms/step - loss: 0.6001 - accuracy: 0.7371 - val\_loss: 0.7987 - val\_accuracy: 0.6742
101. Epoch 50/100
102. 296/296 [==============================] - 38s 129ms/step - loss: 0.5936 - accuracy: 0.7426 - val\_loss: 0.7918 - val\_accuracy: 0.6710
103. Epoch 51/100
104. 296/296 [==============================] - 38s 130ms/step - loss: 0.5967 - accuracy: 0.7420 - val\_loss: 0.8401 - val\_accuracy: 0.6577

## Conclusiones y Resultado vs. Version Anterior:

Se fuerza la parada del proceso se entrenamiento en la época 51 al ver que se estaba disparando el overfitting. Probablemente el generador de datos de entrenamiento ya haya pasado muchos ejemplos muy parecidos y el modelo esté sobreajustando el aprendizaje a los ejemplos vistos, en lugar de generalizar correctamente.

Próxima ejecución, valorar:

- Menos épocas, alrededor de 25-35

- Más variabilidad en los parámetros de la “augmentation” del conjunto de train.

- Añadir más capas, Dense y/o Conv2D.