David Parreño Barbuzano
Sergio del Ser Muñoz

Práctica 2 Redes Neuronales. Clasificador de peces.

Introducción.

En esta práctica se ha implementado una red neuronal convolutiva que es capaz de clasificar distintos grupos de peces. El dataset que se ha empleado está formado por cuatro clases de peces, en el que cada uno se compone de 1000 imágenes, obtenidas de un dataset de Kaggle (ver https://www.kaggle.com/crowww/a-large-scale-fish-dataset).

Hiperparámetros y parámetros.

- **Inicializador del Dataset**

Nombre	Definición	Valor
Training percentage	Porcentaje de ficheros para el entrenamiento	80%
Validation percentage	Porcentaje de fichero para la validación	20%

- **Compilación de la red neuronal**

Nombre	Definición	Valor
Loss function	Función de pérdida	Categorical Cross Entropy
Optimizer	Optimizador	Ada Delta

- **Entrenamiento y EarlyStopping**

Nombre	Definición	Valor
Number of epochs	Número máximo de iteraciones	30
Steps per epoch	Pasos totales para cada época	70
Batch size	Número de ficheros cargados por cada época	30
Patience	Número máximo de intentos para mejorar val_accuracy antes de que finalice el entrenamiento	3

- **Data augmentation**

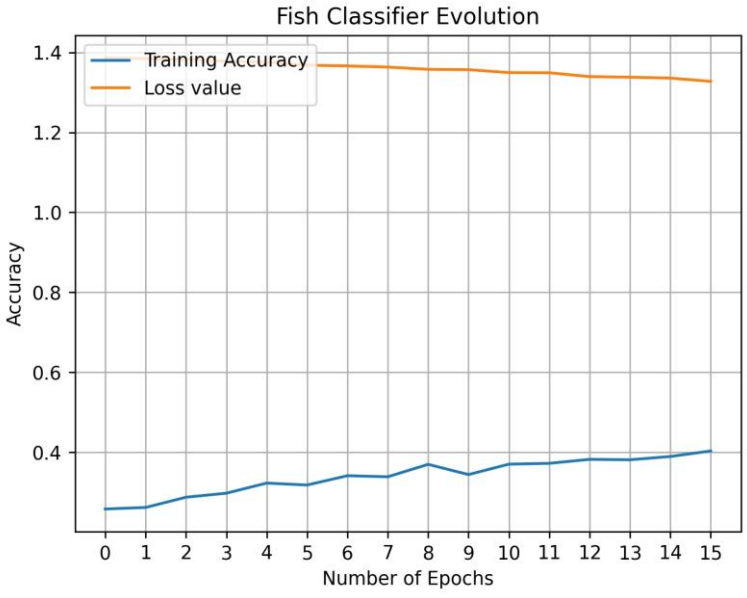
Nombre	Definición	Valor
Target size	Ancho y alto para cada imagen cargada	200w, 200h
Rescale	Reescalado usado sobre las imágenes	1./255
Rotation range	Rango de rotación para las imágenes	30
Zoom range	Rango del zoom para las imágenes	0.7
Width shift range	Rango del desplazamiento del ancho	0.1
Height shift range	Rango del desplazamiento del alto	0.1
Brightness range	Rango del brillo de las imágenes	(0.2,0.8)
Horizontal flip	Volteo horizontal de las imágenes	True
Vertical flip	Volteo vertical de las imágenes	True

David Parreño Barbuzano
Sergio del Ser Muñoz

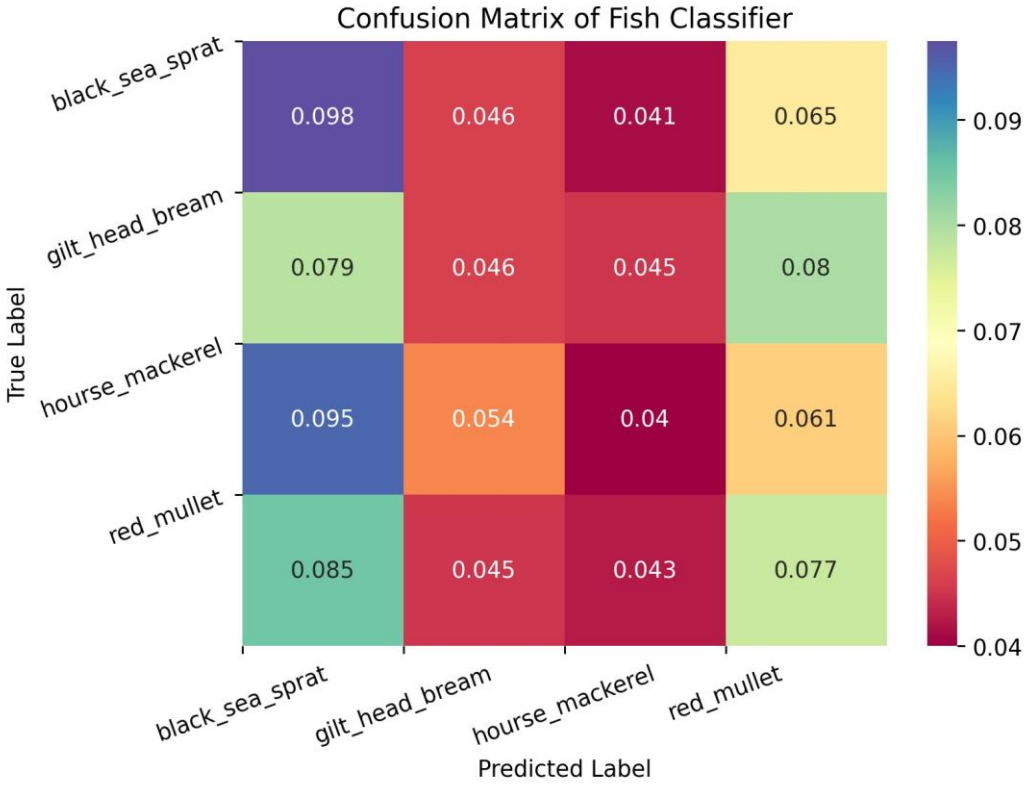
Resultados obtenidos.

De entre todas las posibles configuraciones que se han probado, la que mejor resultados ha ofrecido ha sido una que contiene una capa convolutiva de 128 filtros con función de activación ReLU, una capa LeakyReLU con α=0.1, y tres capas densas, todo ello con capas Dropout y de MaxPooling. Esta arquitectura de red llega a alcanzar alrededor del 38-40% de precisión, y un 70% en la validación. Los motivos de que la red neuronal no llegue a ser más eficiente y precisa son varios, los cuales se explicarán a continuación:

- **Número de pasos por época:** el proceso de entrenamiento se ha restringido de tal manera que en cada época se hagan sólo 70 pasos, cuando realmente este número debería corresponder a la fórmula nºmuestras / batch_size. El motivo de que se haya reducido tanto este número se debe más bien a limitaciones en el hardware, lo que significa que muy posiblemente la red llegaría a alcanzar mejores resultados si el número de pasos no se limitase.
- **Número de épocas:** el número de épocas que realiza la red neuronal se ha limitado a 30, por exactamente la misma razón que ocurría en el número de pasos por época, es decir, limitaciones en el hardware empleado durante el entrenamiento.
- **Mala configuración de la red neuronal:** aunque la arquitectura de red que se acabó escogiendo es la que mejor resultados ha ofrecido, ello no implica que no sea mejorable, por lo que puede ser que hagan falta más capas o que las existentes se localicen en la red de distinta manera. Este puede ser el motivo más evidente, pues durante el entrenamiento la red neuronal llega en muchas ocasiones a necesitar dos épocas más para superar la mejor precisión alcanzada, lo que supone que, si no se cambiase la red, 30 épocas no son suficientes.
- **Alto sobreajuste:** en muchas ocasiones la precisión de la validación llega a superar a la del entrenamiento, lo que puede significar que la red no está realmente aprendiendo, sino que trata de mejorar la precisión en la validación, pero no tanto en el entrenamiento. Una solución podría ser añadir más ficheros de test que no se empleen durante en el entrenamiento.
- **Similitud de las clases:** el dataset que se ha escogido para la realización de la práctica está compuesto por imágenes de peces que puede que la red neuronal no sea capaz de diferenciarlos por su gran similitud, siendo en muchas ocasiones el tamaño y el color lo que realmente los hace distintos. Esto se ha intentado arreglar con el data augmentation y usando un dataset amplio.



David Parreño Barbuzano
Sergio del Ser Muñoz




```
Epoch 7/30
70/70 [==============================] - 93s 1s/step - loss: 1.3667 - accuracy: 0.3536 - val_loss: 1.3019 - val_accuracy: 0.5675
Epoch 8/30
70/70 [==============================] - 89s 1s/step - loss: 1.3627 - accuracy: 0.3385 - val_loss: 1.2873 - val_accuracy: 0.6550
Epoch 9/30
70/70 [==============================] - 87s 1s/step - loss: 1.3575 - accuracy: 0.3682 - val_loss: 1.2693 - val_accuracy: 0.6525
Epoch 10/30
70/70 [==============================] - 86s 1s/step - loss: 1.3574 - accuracy: 0.3315 - val_loss: 1.2562 - val_accuracy: 0.5437
Epoch 11/30
70/70 [==============================] - 87s 1s/step - loss: 1.3460 - accuracy: 0.3703 - val_loss: 1.2487 - val_accuracy: 0.6737
Epoch 12/30
70/70 [==============================] - 87s 1s/step - loss: 1.3500 - accuracy: 0.3658 - val_loss: 1.2315 - val_accuracy: 0.6450
Epoch 13/30
70/70 [==============================] - 87s 1s/step - loss: 1.3404 - accuracy: 0.3717 - val_loss: 1.2219 - val_accuracy: 0.7013
Epoch 14/30
70/70 [==============================] - 87s 1s/step - loss: 1.3389 - accuracy: 0.3815 - val_loss: 1.2070 - val_accuracy: 0.6650
Epoch 15/30
70/70 [==============================] - 87s 1s/step - loss: 1.3308 - accuracy: 0.4033 - val_loss: 1.1943 - val_accuracy: 0.6400
Epoch 16/30
70/70 [==============================] - 87s 1s/step - loss: 1.3324 - accuracy: 0.3927 - val_loss: 1.1808 - val_accuracy: 0.6612
```