Advanced Machine Learning: Assignment 4

Davide Brinati

Matricola 771458

1 Introduzione

L'obiettivo di questo assignment è di effettuare finetuning di una rete neurale convoluzionale pre allenata su *Imagenet*, per poi implementarla per la classificazione di immagini. Il modello scelto per questa task è il **VGG16**.

Il dataset scelto riguarda immagini a colori di dieci diverse razze di scimmia e risulta essere composto da 1097 immagini di training e 272 di test. (Link al Dataset).

Di seguito è possibile osservare l'immagine di un esemplare per 3 delle 10 diverse specie che compongono il training.

Figure 1: Pygmy Marmoset

Figure 2: Patas Monkey

Figure 3: Japanese Macaque

2 Preprocessing Immagini

Le immagini del dataset sono a colori, risultano quindi essere composte da 3 canali, ma hanno diverse dimensione di *width* ed *height*. Dato che il modello accetta in input solo immagini di 224x224 pixel, si effettuerà un reshape. Inoltre sul train sono state effettuate operazioni di data agumentation; in particolare è stato inserito un parametro che effettua delle rotazioni random sulle immagini (rotation range=40), un altro che effettua zoom randomici (zoom range = 0.2) e un parametro che effettua un horizontal flip randomico (effetto specchio).

Il train infine è stato diviso in train e validation, con uno split pari 0.2; il primo risulta essere composto da 880 immagini e il secondo da 217.

3 Modello

Come accennato nell'introduzione, il modello convoluzionale preallenato su *imagenet*, che è stato utilizzato in questo assignment è il VGG16. Di questo modello sono stati importati tutti gli strati di convoluzione e maxpooling, ma non la parte *fully-connected*, che è stata troncata, in quanto dovrà essere

rimodellata secondo le esigenze della mia task, ovvero individuare la razza della scimmia nell'immagine, fra 10 possibili classi.

Di seguito è possibile osservare la struttura della rete VGG16, le dimensioni dell'output di ogni strato e il relativo numero di parametri.

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 224, 224, 3)	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0

Total params: 14,714,688

Il primo strato della rete riceve in input tensori di dimensione 224x224 su 3 canali. L'ultimo strato della rete è un Maxpooling, che dopo aver condensato ulteriormente le informazioni, dà in output tensori di dimensione 7x7 su 512 canali.

Questa parte di rete è stata "freezzata", ovvero è stata resa non allenabile, mantenendo quindi i pesi pre allenati. Ciò risulta ragionevole in quanto questi layer catturano delle features universali, come i contorni e le forme degli oggetti, le quali risultano essere ovviamente rilevanti anche per il nostro problema di classificazione.

Prima della parte fully connected è stato posizionato un layer flatten, con il compito di appiattire i tensori di dimensione 7x7x512 in array di dimensione 25088.

Successivamente è stato posizionato un layer con 512 neuroni, e funzione di attivazione relu, seguito da uno strato di dropout, che elimina il 25% delle connessioni ad ogni epoca di apprendimento. Lo strato successivo è quello di output, con 10 neuroni, tanti quante le possibili razze, e con funzione di attivazione softmax. Di seguito è possibile osservare la struttura della rete che addestreremo sul train, per poi generalizzare sul test.

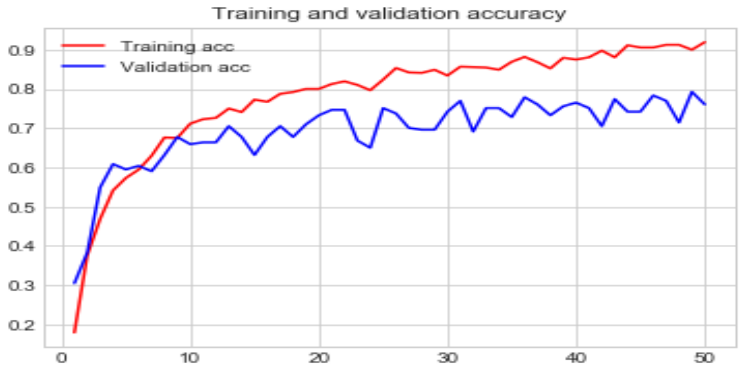
Layer (type)	Output Shape	Param #
vgg16 (Model)	(None, 7, 7, 512)	14714688
flatten_1 (Flatten)	(None, 25088)	0
dense_1 (Dense)	(None, 512)	12845568
dropout_1 (Dropout)	(None, 512)	0
dense_2 (Dense)	(None, 10)	5130

Total params: 27,565,386
Trainable params: 12,850,698
Non-trainable params: 14,714,688

Il primo layer contiene il modello pre allenato che è stato mostrato nella figura della pagina precedente. Si noti che i parametri allenabili sono 12850698, ovvero sollo quelli della parte fully connected.

La funzione di loss utilizzata è la categorical crossentropy e l'ottimizzatore è adam, con un learning rate pari a 0.0001. Il modello è stato allenato per 50 epoche e durante l'apprendimento è stata utilizzata la callback *Model Checkpoint*, che monitora l'andamento dell'accuracy sul validation in ogni epoca e permette di salvare in un file i pesi del modello che raggiunge la miglior performance.

Nelle immagini sottostanti è possibile osservare l'andamento delle performance del modello sul training e sul validation, in termini di accuracy e loss, nelle 50 epoche.



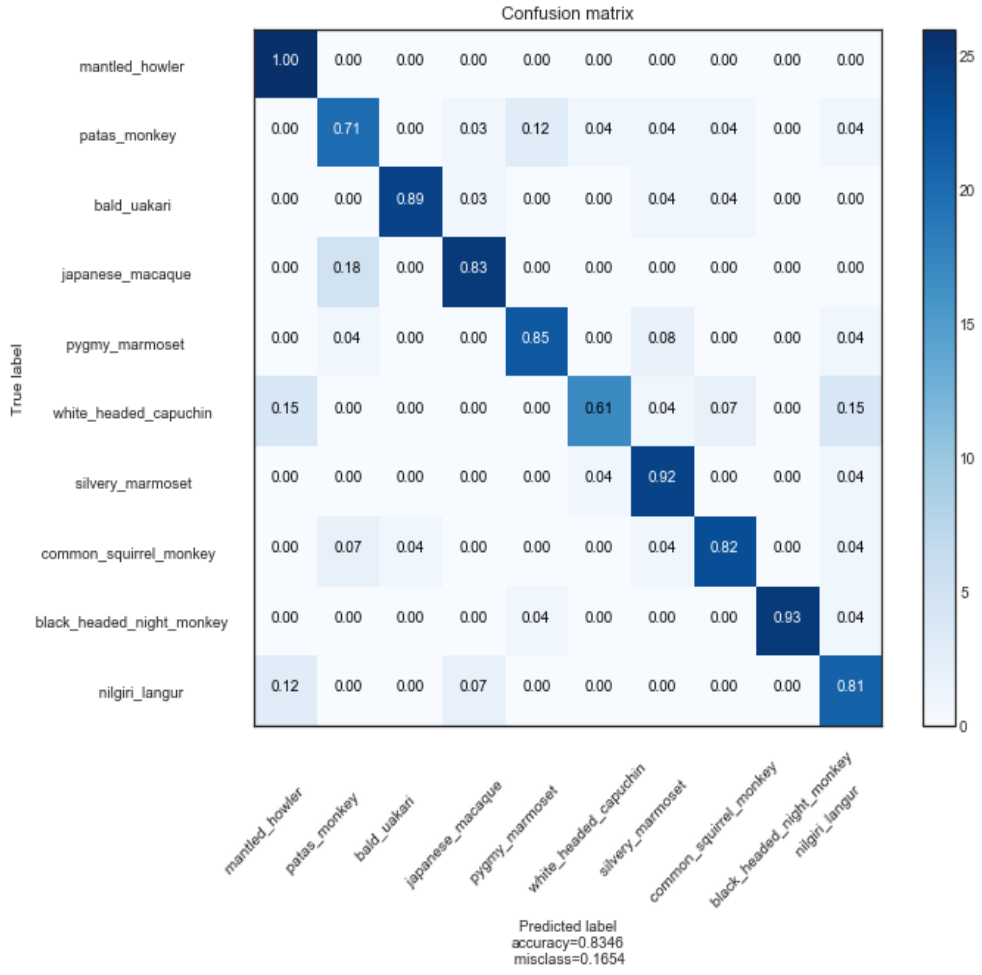
I pesi che hanno permesso al modello di raggiungere il valore più alto di accuracy **sul validation** sono salvati nel file *bestmodel.h5*.

Tali pesi verranno caricati ed utilizzati per classificare la razza delle scimmie nelle immagini di test.

4 Conclusioni

Il modello, nella classificazione delle immagini di test, ha raggiunto i seguenti risultati: **l'accuracy** è pari a **0.8346**, con un valore di **loss** pari a **0.5279**. Di seguito è possibile osservare una matrice riportante i risultati di precision, recall e F1 score per ogni classe, seguita dalla confusion matrix, che riassume i risultati della classificazione.

	precision	recall	f1-score
mantled_howler	0.79	1.00	0.88
patas_monkey	0.71	0.71	0.71
bald_uakari	0.96	0.89	0.92
japanese_macaque	0.86	0.83	0.85
pygmy_marmoset	0.85	0.85	0.85
white_headed_capuchin	0.89	0.61	0.72
silvery_marmoset	0.80	0.92	0.86
common_squirrel_monkey	0.85	0.82	0.84
black_headed_night_monkey	1.00	0.93	0.96
nilgiri_langur	0.70	0.81	0.75
avg / total	0.84	0.83	0.83



Osservando la confusion matrix, possiamo notare che la razza *White Headed Capuchin* è quella che dà maggiori problemi al modello, il quale non supera lo 0.61 di accuratezza su questa classe. Il valore di recall di questa classe risulta basso, ciò sta ad indicare che il classificatore tende a sbagliare sui *False Negative*, ovvero tende a confondere questa classe con altre. In particolar modo il modello confonde il

White Headed Capuchin con il Mantled Howler e con il Nilgiri Langun. Di seguito è possibile osservare un confronto fra le tre specie.

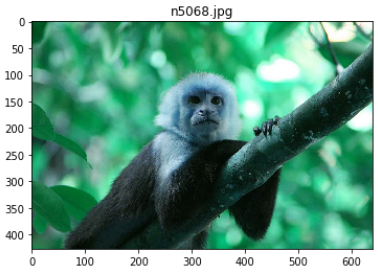


Figure 4: White Headed Capuchin

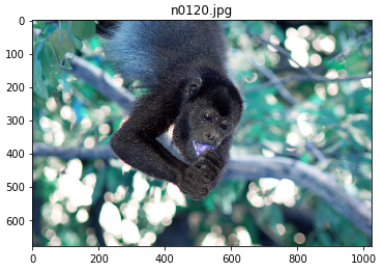


Figure 5: Mantled Howler

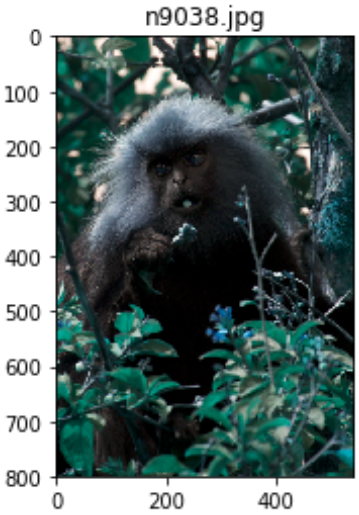


Figure 6: Nilgiri Langun

Possiamo osservare che il White Headed Capuchin (Figura 4), ha una faccia molto somigliante a quella del Mantled Howler (Figura 5), e probabilmente è per questo motivo che il modello tende a confondere il primo con il secondo. In maniera analoga, confrontando il White Headed Capuchin con il Nilgiri Langun (Figura 6) possiamo osservare che entrambi possiedono una chioma bianca sopra la testa; è presubilmente per questo motivo che il modello tende a sbagliare, classificando il White Headed Capuchin erroneamente e confondendolo con il Nilgiri Langun.