UNIVERSITÀ
DEGLI STUDI
DI PADOVA

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

CORSO DI LAUREA IN INGEGNERIA INFORMATICA

"Classificazione automatica di specie ittiche nell'Adriatico mediante l'utilizzo di tecniche di Deep Learning"

Relatore:
Prof. Loris Nanni

Laureando:
Davide Fantin 1148098

ANNO ACCADEMICO 2021 – 2022

Data di laurea 14/11/2022

Sommario

I sistemi osservativi marini in grado di registrare dati sia oceanografici (temperatura, salinità, ossigeno disciolto etc.), che biologici (grazie alla presenza di telecamere, quindi sulla flora e fauna presente) rappresentano una grossa sfida per le future strategie di monitoraggio dell'ambiente marino. L'acquisizione di dati in continuo e in quasi real-time permette di avere indicazioni immediate sui cambiamenti in essere delle comunità biologiche legati a fenomeni naturali o impatto all'antropico diretto e non, come ad esempio cambiamenti climatici. L'immensa mole di dati che questi sistemi possono acquisire necessita di tecniche moderne per il loro processamento per questo si è scelto di trattare i dati con tecniche di Deep Learning che consentono di riconoscere e classificare le specie ittiche presenti in un dataset di video registrati ogni ora per 12 mesi (da Febbraio 2020 a Febbraio 2021) presso una stazione localizzata sul fondo a circa 17 m di profondità al largo della costa abruzzese (Adriatico Centrale), già analizzati in parte con metodologia tradizionale (conteggio manuale e identificazione delle specie).
Per analizzare la grande mole di dati acquisiti si è pensato di utilizzare gli strumenti offerti dall'intelligenza artificiale.

Indice

Introduzione

Nonostante i mari coprano più di due terzi della superfice della Terra gli ecosistemi marini sono molto meno studiati di quelli terrestri, proprio per la difficoltà intrinseca nello svolgere operazioni di ricerca in ambienti non adatti all'uomo. La nostra conoscenza dei primi è perciò limitata, soprattutto per quanto riguarda i danni provocati dall'inquinamento e dai cambiamenti climatici.

Per facilitare le operazioni di ricerca è stato pensato di sviluppare, in collaborazione con l'Università Politecnica delle Marche (UNIVPM), un software in grado di supportare i ricercatori e di far risparmiare loro tempo prezioso. L'Università Politecnica delle Marche ha fornito un ampio catalogo di filmati, registrati al largo della costa abruzzese da una telecamera posta su di una piattaforma marina a 17 metri di profondità. Il software sviluppato utilizzando tecniche di deep learning si occuperà di individuare i pesci e di identificarne la specie.

Per quanto riguarda la prima parte si è deciso di usare l'algoritmo You Only Look Once (YOLO) per le sue ottime performance, mentre la seconda parte utilizzando le reti neurali AlexNet e VGG16 anche esse tre le migliori soluzioni disponibili in fatto di prestazioni.

Capitolo1

Object detection

1.1 Introduzione a Computer vision e Object detection

La visione artificiale (o computer vision) è una branca dell'intelligenza artificiale il cui scopo è ricavare dati significativi da immagini o video. Mimando la vista umana la computer vision si occupa per esempio di identificare oggetti o il loro movimento utilizzando telecamere e algoritmi al posto di occhi e corteccia visiva. Tuttavia, spesso è richiesta nelle applicazioni industriali un'altissima velocità o precisione il che ci fa capire quanto poi diventa complesso gestire i problemi anche dal punto di vista della potenza computazionale richiesta. Per raggiungere tali risultati vengono solitamente utilizzate due tecnologie: il machine learning e le reti convoluzionali.

Il machine learning permette a un computer di imparare in autonomia, cioè senza l'intervento di un operatore, a distinguere le immagini, posto che abbia ricevuto una sufficiente quantità di dati da astrarre.

Mentre una rete convoluzionale detta anche CNN (convolutional neural network), aiuta il modello di deep learning, tramite operazioni matematiche di convoluzione, a concentrarsi su determinati dettagli dell'immagine scomponendole in pixel e catalogandone le varie aree.

Nella pratica una CNN è composta da un livello di input, dei livelli nascosti (hidden layers) e uno di output. Solitamente i livelli di input e hidden sono quelli convoluzionali e applicano su pozioni dell'immagine filtri che compiono operazioni di convoluzione. L'output di ogni livello viene poi passato al livello successivo in cui ogni neurone si occupa solo del suo spazio sensoriale ma ha spesso pesi condivisi con i neuroni adiacenti. Lo scopo di questi livelli è generare una features map che serve a raccogliere la descrizione caratteristica dell'immagine. Alla fine, i livelli di output, solitamente di tipo fully connected, si occupano della classificazione vera e propria dell'immagine.

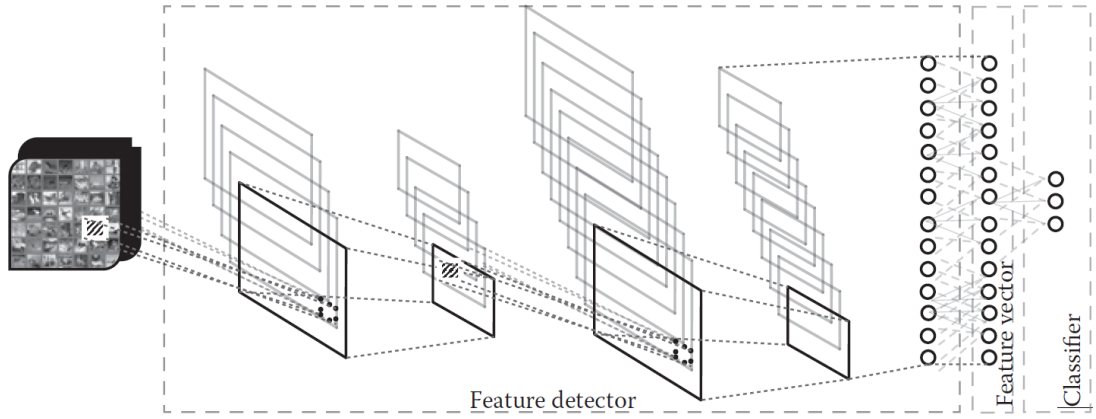

Figura 1: architettura tipica di una CNN, dove sono evidenziati i livelli convoluzionali e il classificatore. Copyright: R. Venkatesan, B. Li, *"Convolutional Neural Networks in Visual Computing. A Concise Guide"*

L'object detection è una branca della computer vision ed è la tecnologia che si occupa di individuare oggetti appartenenti a un determinato significato semantico all'interno di un'immagine, cioè assegnare un'etichetta ad un oggetto all'interno di una regione detta bounding box, ed è uno dei campi più importanti all'interno della computer vision.

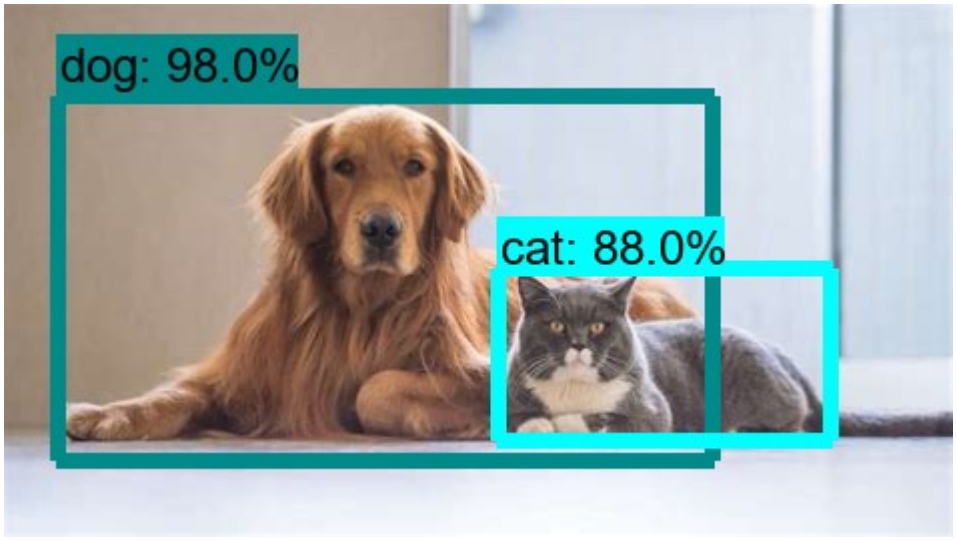

Figura 2: esempio di object detection. Copyright: ai-pool.com

Negli ultimi anni le reti neurali profonde, come le CNN, hanno dimostrato ottime performance nella risoluzione di molte difficoltà in diversi ambiti; tuttavia, il loro addestramento è molto lungo e dispendioso in termini di risorse. Perciò l'approccio più utilizzato consiste nell'usare classificatori già addestrati e riadattarli per il loro nuovo utilizzo. Questo processo prende il nome di transfer learning.

1.2 YOLO

Per risolvere i problemi dati da questo approccio ultimamente ha preso piede You Only Look Once che diminuisce la complessità computazionale riducendo il problema di rilevamento a una singola regressione. Una singola rete neurale prova a prevedere le bounding boxes e contemporaneamente assegna le probabilità su tutta l'immagine, per questo viene definito one stage detector. Il risultato è una rete molto veloce dato che non necessita di più passaggi da eseguire in serie, tanto da poter essere utilizzata anche per applicazioni real time.

Inoltre, YOLO ragiona sull'intera immagine, questo permette di imparare anche le informazioni di contesto e commettere meno errori nell'identificazione di oggetti sullo sfondo delle CNN.

YOLOv2 eYOLOv3 sono evoluzioni incrementali YOLO che ne migliorano le prestazioni principalmente grazie ad alcune modifiche negli algoritmi che predicono le anchor boxes, che non sono più bounding boxes arbitrarie, e all'utilizzo di reti neurali più ampie.

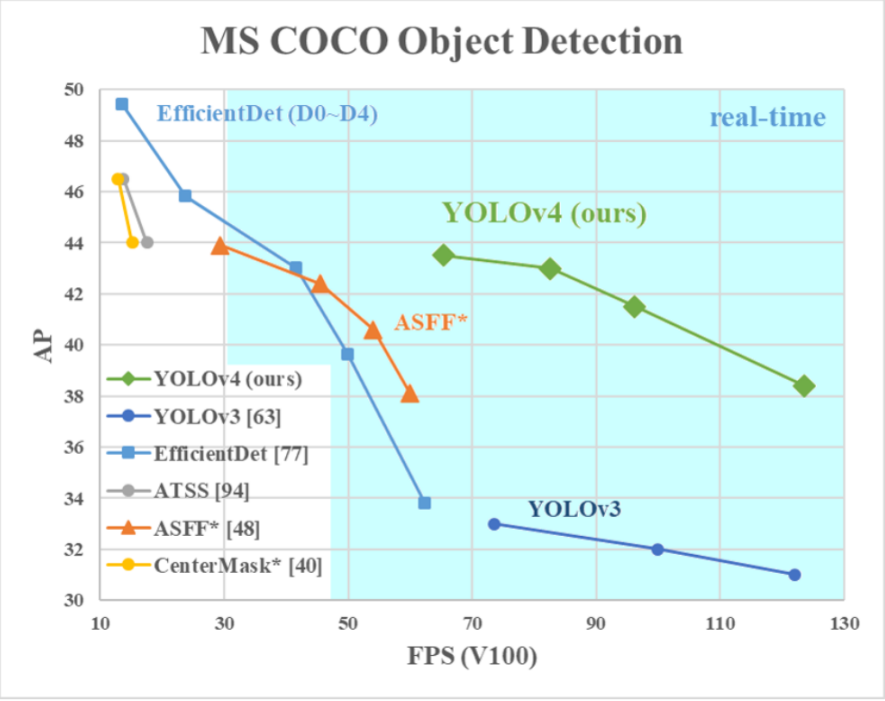

Figura 3: prestazioni di YOLOv4 confrontate con altri object detector allo stato dell'arte. Copyright: A. Bochkovskiy, C.-Y. Wang, H.-Y. M. Liao

La rete del classificatore YOLOv4 può essere divisa in tre componenti detti: backbone, neck e head.

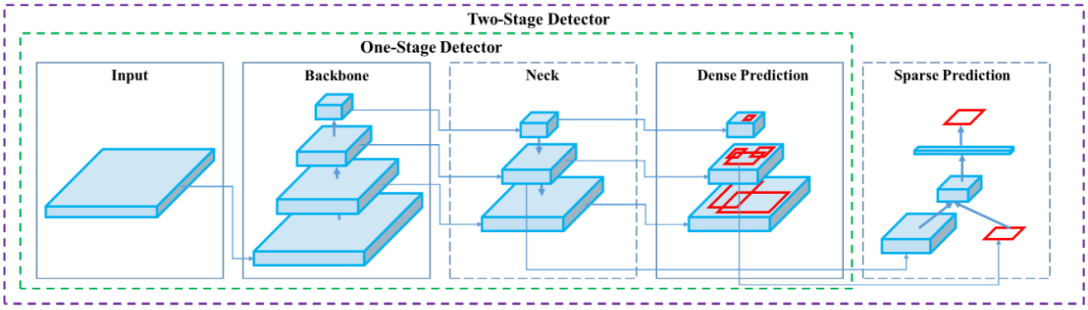

Figura 4: struttura di YOLOv4. Copyright: A. Bochkovskiy, C.-Y. Wang, H.-Y. M. Liao

La backbone si occupa di estrarre feautures dai pattern ed è affidata a CSPDarknet-53. Il neck di un object detector ha il compito di raggruppare le fetures map cioè gli output della backbone di fasi diverse, in questo caso si utilizzano Spatial pyramid pooling e Path aggregation network.

Head, che si occupa di compiere il riconoscimento e della gestione dell'output, invece è rimasta invariata da YOLOv3.

Capitolo 2

Addestramento rete detection

2.1 Datasets

Come già anticipato la rete YOLO, che è stata usata per rilevare in maniera autonoma le specie ittiche, deve essere addestrata. Per velocizzare il processo, che altrimenti sarebbe molto lungo, è stata usata una rete preaddestrata sul dataset COCO (Common Objects in Context) che contiene 2500000 istanze catalogate divise in 328000 immagini di oggetti comuni ma non di pesci. Dunque, è poi stato necessario fare un ulteriore addestramento sugli ultimi livelli della rete per adattarla a riconoscere specie ittiche. Tale processo prende il nome di transfer learning e a questo scopo sono stati utilizzati i seguenti dataset:

- Labeled Fishes in the Wild: dataset di NOAA che contiene immagini scattate mediante telecamere poste su un veicolo telecomandato sottomarino per indagini sulla pesca.

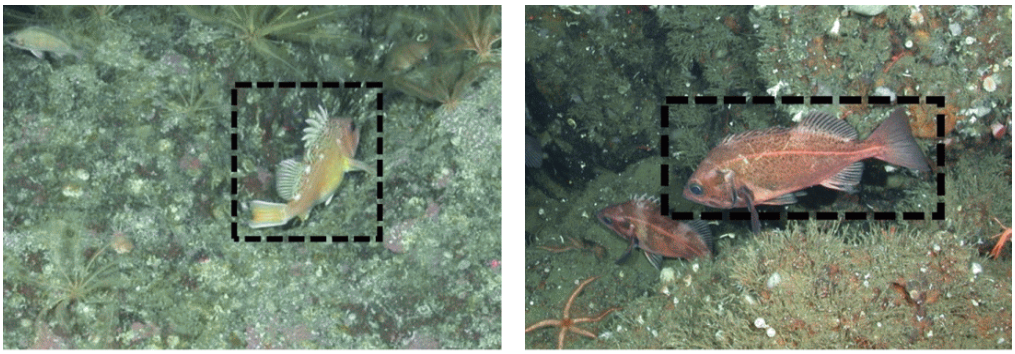

Figura 5: Esempio "Labeled Fishes in the Wild". Copyright: NOAA Fisheries (National Marine Fisheries Service).

- Aquarium Dataset: contiene immagini scattate negli acquari dell'Henry Doorly Zoo di Omaha e nel National Aquarium di Baltimora, comprende non solo pesci ma anche altri animali marini.

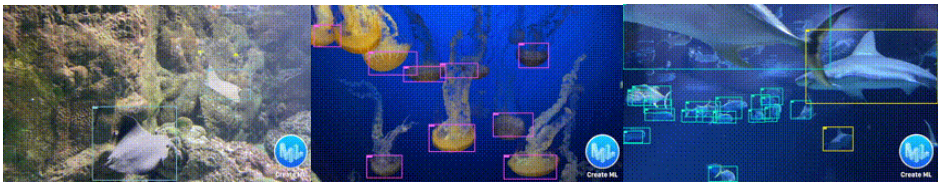

Figura 6: Esempio "Aquarium Dataset". Copyright: roboflow

- NorFisk Dataset: contiene immagini relative all'allevamento di salmonidi e merluzzo carbonaro. Il dataset è molto esteso perciò per non sbilanciare il processo di training ne è stata usata solo una parte.

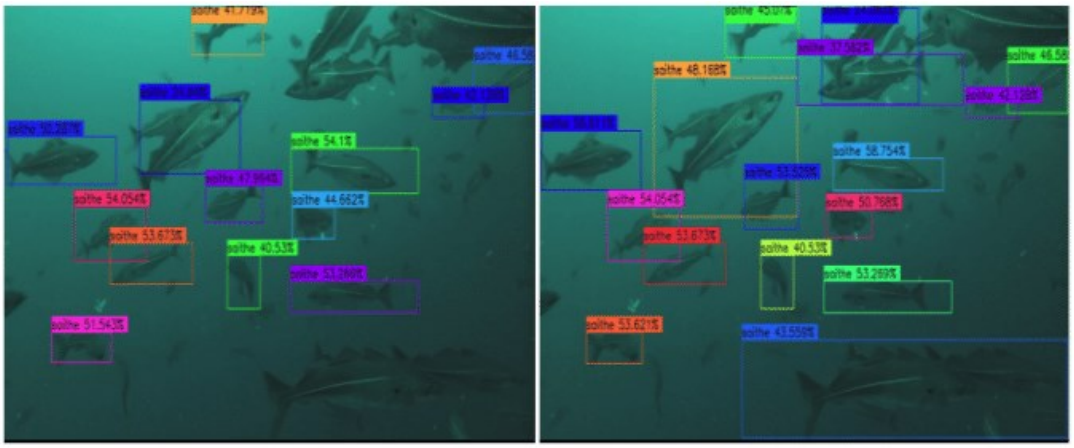

Figura 7: Esempio "NorFisk Dataset". Copyright: Lars Gansel

- LifeCLEF 2015 Fish task: contiene 93 video di 15 differenti specie di pesci, anche in questo caso ne è stata usata solo una parte.

Figura 8: Esempio "LifeCLEF 2015 Fish". Copyright:A. Ben Tamou, B. Abdesslam, K. Nasreddine

Per la parte di testing sono state utilizzate parti degli stessi dataset a cui sono stati aggiunti:

- Dataset dell'Università Politecnica delle Marche (UNIVPM): contiene video della durata di 3 minuti registrati ogni ora da febbraio 2020 a febbraio 2021 da una webcam posta su una piattaforma sottomarina a circa 17 metri di profondità

al largo della costa abruzzese nell'adriatico centrale. Data la mancanza di annotazioni i pesci sono stati localizzati dal collega Dario Mameli tramite Image Labeler di Mathworks su un numero limitato di frame che però potesse comunque offrire immagini in ogni condizione di luminosità e torbidità.

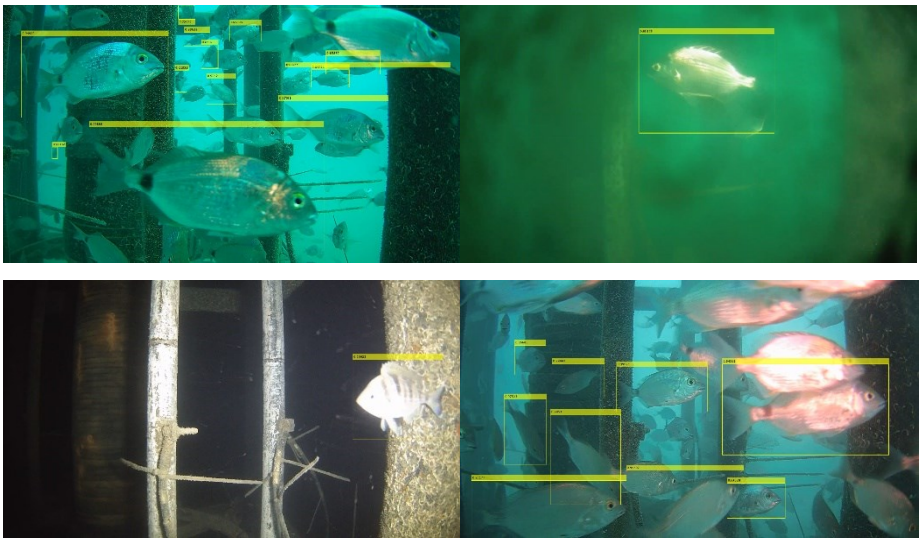

Figura 9: Esempio "Dataset Università Politecnica delle Marche".

- Dataset Liguria: contiene 18 video registrati con actioncam nel mar Ligure.

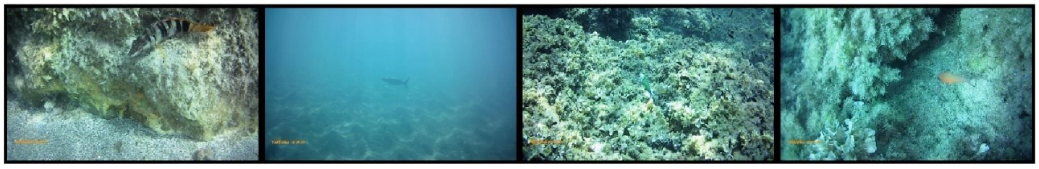

Figura 10: Esempio "Dataset Liguria". Copyright: D. Mameli

2.2 Preprocessing

Per rendere compatibile con il primo livello di YOLOv4 le immagini devono essere prima scalate ad una risoluzione minore, ciò diminuisce notevolmente i tempi di calcolo. YOLOv4 accetta immagini RGB di dimensione 608x608 pixel. Si noti che nel processo di scala non vengono mantenute le proporzioni dell'immagine originale.

2.3 Data Augmentation

Spesso la parte più critica nell'addestramento di una rete neurale è la mancanza di una quantità sufficiente di dati di qualità. Il processo di deep learning per ottenere buoni risultati necessita di un'enorme quantità di dati che spesso sono difficili da reperire.

Inoltre, se una rete viene addestrata su un campione troppo ristretto di dati si corre il rischio di incorrere nell'Overfitting, ossia l'algoritmo ha una buona performance nel training set e scarsa nel test set, si potrebbe quasi dire che smetta di astrarre e cominci a imparare a memoria i features.

In questo caso si utilizzano tecniche di data augmentation per aumentare in maniera artificiale il numero di dati disponibili e migliorare la loro qualità. Le tecniche che sono state usate nel nostro caso sono:

- Flipping e rotation, ossia ribaltamento e rotazione, l'immagine viene semplicemente ribaltata solitamente sull'asse verticale e/o ruotata.
- jitterColorHSV, altera l'immagine intervenendo nello spazio dei colori HSV. Cioè modificandone hue (gamma cromatica), saturation (saturazione) e value (luminosità).
- Bilanciamento del colore, particolarmente importante nel nostro caso in quanto l'acqua tende ad assorbire prima i colori con lunghezze d'onda maggiore. Quindi il rosso viene tagliato già a pochi metri di profondità e i colori tendono a virare verso verde e blu.

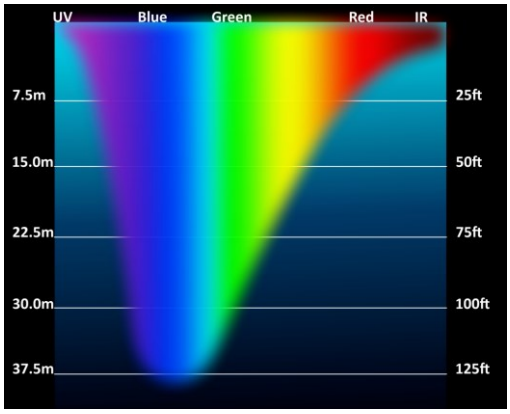

Figura 11: Light Penetration Spectrum in Water. Copyright: Tomemorris at wikimedia commons

- Aggiustamento di luminosità e contrasto, in particolare la seconda mediante l'algoritmo (CLAHE).
- Image fusion (fusione d'immagini), in cui due o più immagini vengono fuse in una che contiene le informazioni più significative di entrambe. Una delle tecniche più usate è la wavelet fusion dove, dopo aver applicato alle immagini una serie di filtri, i vari output vengono fusi insieme.

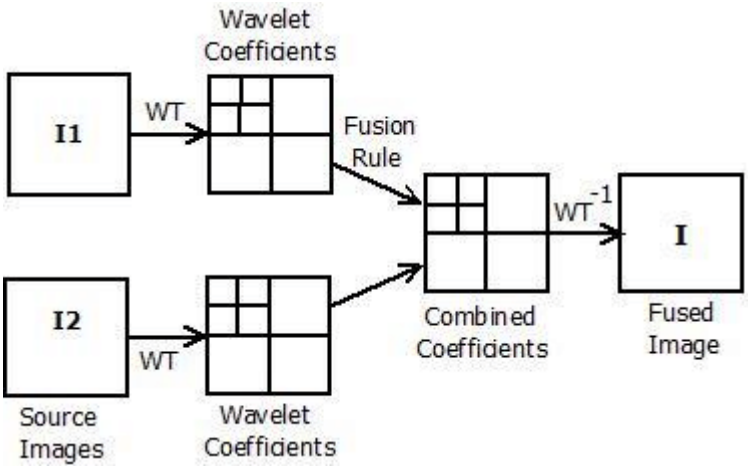

Figura 12: digramma del funzionamento di image fusion tramite wavelet transform. Copyright: R. Barani, S. Mahadevan

2.4 Training

Le informazioni all'interno di una rete YOLOv4 vengono passate dal livello precedente a quello successivo. Viene poi calcolata una loss function, una funzione che ha lo scopo di misurare la distanza tra l'output ricevuto e quello previsto. Nel nostro caso abbiamo usato SSE (sum of squared error), che calcola la somma dei quadrati residui tra la previsione e valore vero.

Sulla base della loss function vengono aggiornati i pesi dei singoli neuroni secondo il principio della backward propagation.

L'algoritmo usato per l'addestramento prende il nome di mini-batch gradient descent. Si basa sulla divisione del training set in sottoinsiemi, detti per l'appunto mini-batch, ogni volta che un'intera batch viene elaborata dall'algoritmo, viene calcolata la loss function e aggiornati i pesi e si ripete per ogni mini-batch. A questo punto viene conclusa un'epoca e si può ripetere il tutto per un certo numero di epoche predefinite o finché la loss function si stabilizza.

2.5 Testing

A questo punto, il training è finito e si deve valutare il risultato. A questo scopo vengono utilizzate due metriche.

- Recall: indica quanto è selettivo il sistema, cioè la proporzione tra i veri positivi e i positivi predetti dalla rete.

$$Recall = \frac{truePositive}{truePositive + falseNegative}$$

- Precision: indica quanto è accurato il sistema, cioè la proporzione tra i positivi predetti dalla rete e i veri positivi.

$$Precision = \frac{truePositive}{truePositive + falsePositive}$$

Dove si indicano con truePositive i veri positivi, falsePositive i falsi positivi e falseNegative i falsi negativi della matrice di confusione.

		Valori predetti		
		n'	p'	totale
Valori reali	n	Veri negativi	Falsi positivi	N
	p	Falsi negativi	Veri positivi	P
	totale	N'	P'	

Figura 13: matrice di confusione

Dato che precision e recall sono fortemente correlate, solitamente vengono rappresentate in un unico grafico dove precision è posta sull'asse delle ordinate e recall su quello delle ascisse.

Avere un modello con alta precision, infatti, porta ad avere bassa recall: il modello seleziona solo pattern pertinenti ma tende a saltare anche quelli validi.

Al contrario uno con alta recall porta ad avere bassa precision: il modello seleziona tutti i pattern validi ma tende a selezionare anche quelli che non lo sono.

- Average precision: è definita come l'area sotto la curva Precision-recall. È compresa tra 0 e 1 ed è alta quando sia precision che recall sono alte e bassa se almeno una è bassa.

Nel test effettuato sul Dataset dell'Università Politecnica delle Marche (UNIVPM) il grafico precision-recall ottenuto è il seguente.

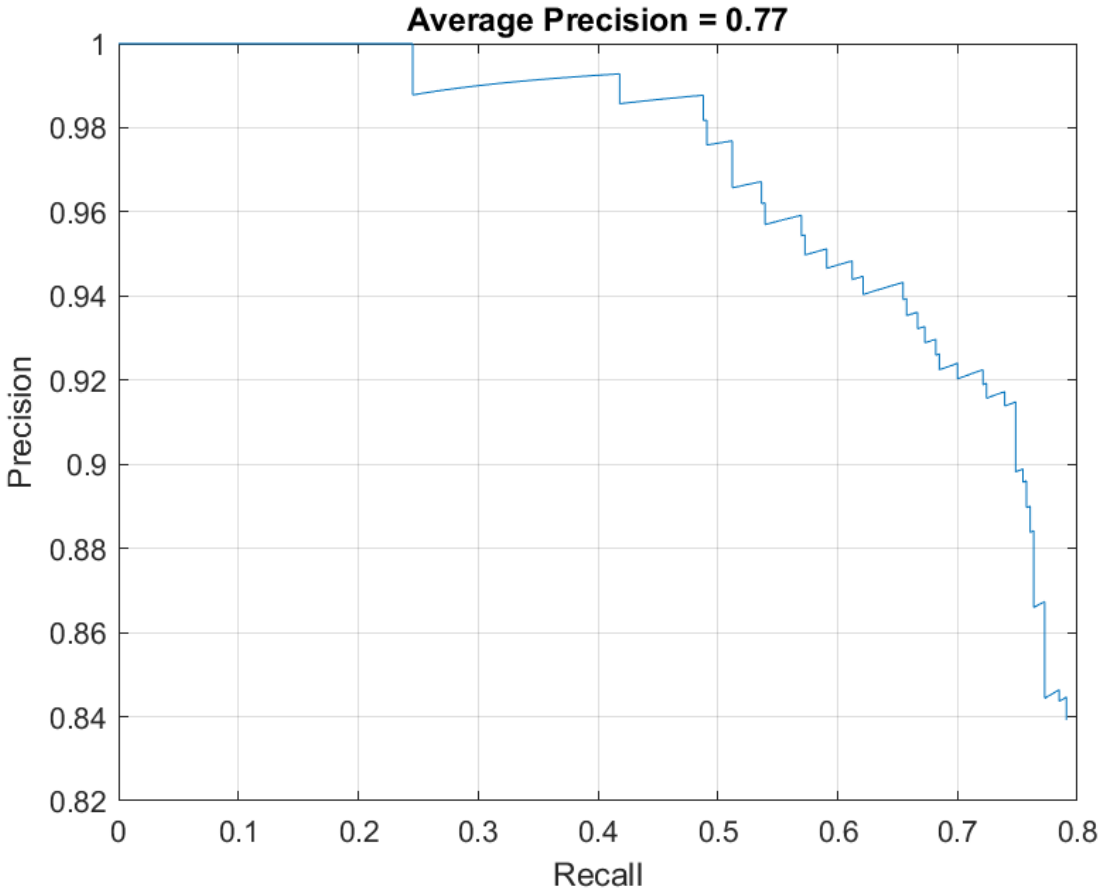

Figura 14: Grafico Precision-Recall del test su dataset UNIVPM. Copyright: D. Mameli

Capitolo 3

Addestramento rete classification

3.1 Classification

Per addestrare la rete per la classification è stato compilato dal sottoscritto un nuovo dataset a partire da quello fornito da UNIVPM. Sono state ricavate dal classificatore delle immagini, con relative bounding boxes, cercando di rispecchiare nella maniera più completa le varie condizioni dell'ambiente ed evitando le immagini completamente inutilizzabili dovute ad un guasto del tergicristallo della webcam. Le 476 istanze estratte sono poi state catalogate a mano. Si è scelto di procedere all'addestramento solo con il dataset UNIVPM perché le specie già catalogate nei dataset disponibili non rispecchiano le specie ittiche che possono essere trovate nel mediterraneo e sarebbero quindi state poco utili nella ricerca del nostro caso specifico. Ripensando il progetto con un respiro più ampio si potrebbe pensare di addestrare la rete su un campione più completo. Il dataset viene poi diviso durante l'addestramento al 75% training set e 25% test set.
Come per l'addestramento del detector siamo partiti da delle reti pre-addestrate di cui è stato fatto solo il fine tuning.

3.2 AlexNet

Partendo dal dataset ImageNet contenente 15 milioni di immagini catalogate con 22000 categorie è stata addestrata AlexNet una CNN tra le più grandi al tempo della sua uscita e vincitrice di numerose competizioni. La rete è composta da otto livelli 5 convoluzionali e 3 fully connected. Alcuni dei livelli convoluzionali sono seguiti da livelli max-pooling che eseguono un'aggregazione delle informazioni in input con lo scopo di diminuire le informazioni, in particolare quelle ininfluenti al riconoscimento del pattern mantenendo però le features significative. Utilizza poi una funzione di attivazione $RelU\ f(x) = max(0, x)$ dove x è l'input del neurone. Come per

l'addestramento del detector siamo partiti da delle reti pre-addestrate di cui è stato fatto solo il fine tuning.

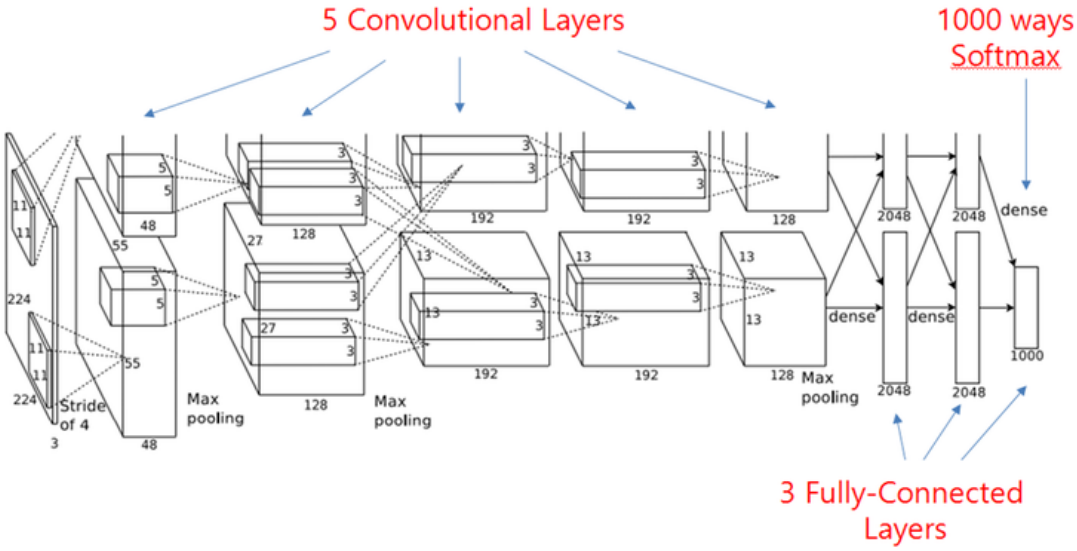

Figura 15: struttura della rete AlexNet. Copyright: narenmanoharan at github.com

3.3 VGG16

VGG16 è una CNN composta da 16 livelli e addestrata su 14 milioni di immagini con circa 1000 categorie di oggetti. È composta da a una serie di livelli convoluzionali con ReLU e max-pooling, prosegue con 3 livelli fully connected e termina con un livello finale SoftMax con funzione di attivazione $z_k = f(net_k) = \frac{e^{net_k}}{\sum_{c=1,\ldots,s} e^{net_c}}$.

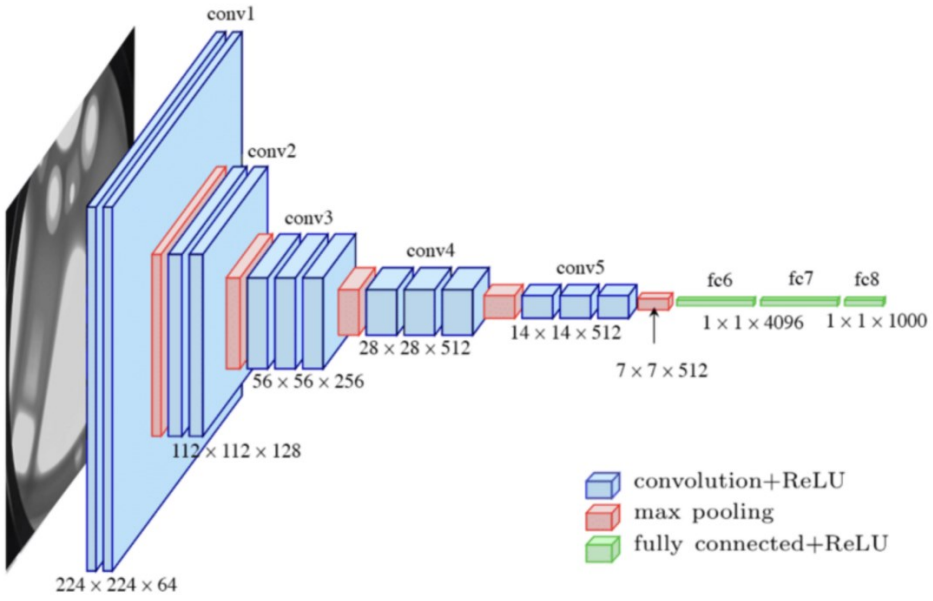

Figura 16: struttura della rete VGG16. Copyright: researchgate.net

Per l'addestramento degli ultimi tre livelli delle reti si sono usati per entrambi i casi i seguenti parametri:

- miniBatch = 30
- learningRate = 0.0001
- numEpochs = 30

3.4 Risultati

I risultati ottenuti con sono abbastanza simili sia con AlexNet che con VGG16 e sono riassunti nei seguenti grafici in cui si fa uso della metrica di accuracy, semplicemente il numero di pattern che sono stati classificati in maniera corretta fratto il numero di pattern totali.

$$Accuracy = \frac{\text{truePositive} + \text{trueNegative}}{\text{truePositive} + \text{trueNegative} + \text{falsePositive} + \text{falseNegative}}$$

3.4.1 Test 1

Il primo test è stato eseguito in maniera identica su AlexNet e VGG16 con 30 epoche.

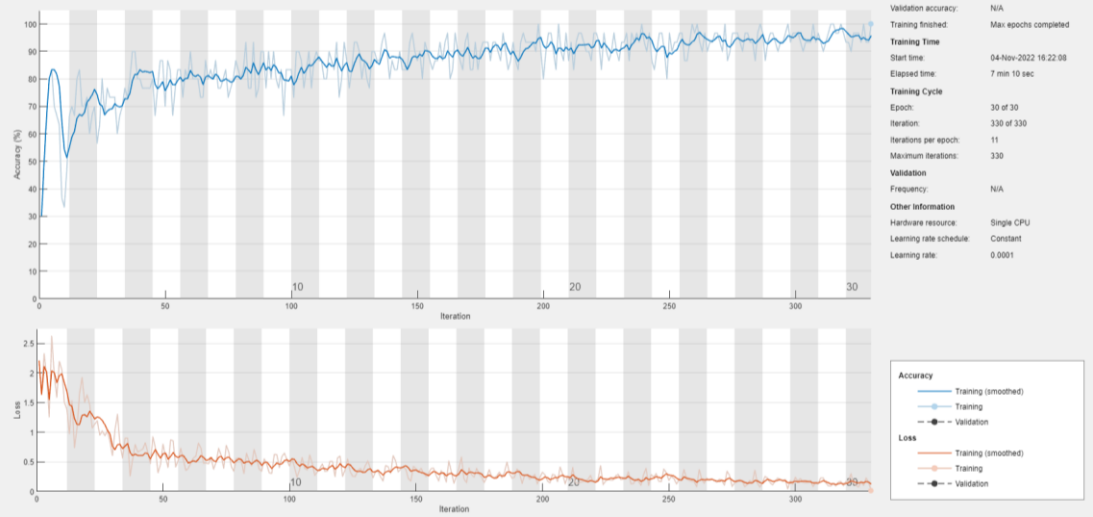

Figura 17: accuracy e loss function ottenute con Alexnet e 30 epoche.

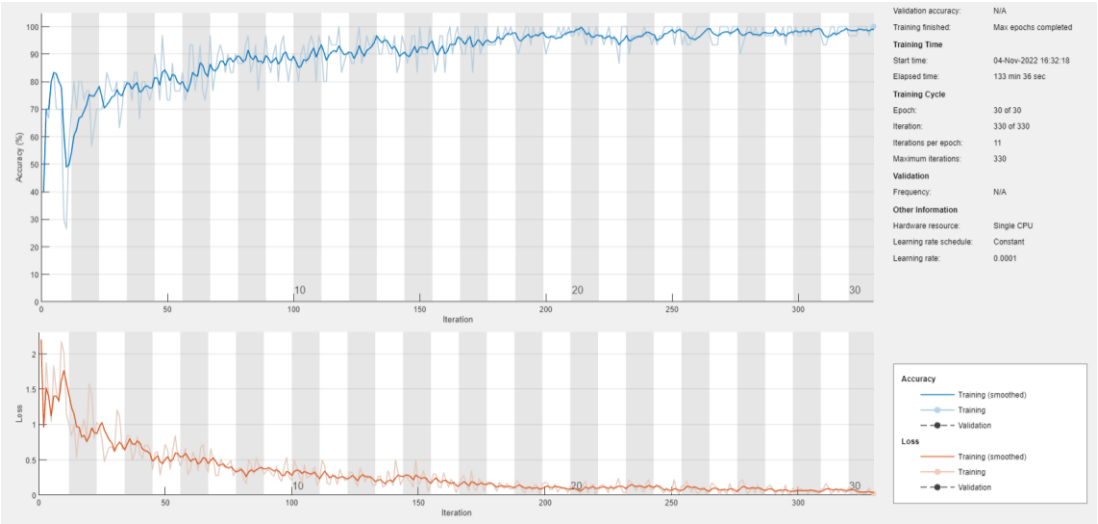

Figura 18: accuracy e loss function ottenute con VGG16 e 30 epoche.

Come si può notare l'accuracy nel training set si attesta attorno al 95% per AlexNet mentre sfiora il 100% su VGG16. Invece per il test set si ottengono i seguenti risultati:

Rete	AlexNet	VGG16
Accuracy	0.7264957264957265	0.7350427350427351

I risultati ottenuti nel test set si discostano molto da quelli ottenuti nel training set, questo può essere un chiaro segnale di overfitting; perciò, sarà opportuno eseguire degli altri test.

3.4.2 Test 2

Il secondo test è stato eseguito solo su AlexNet, molto più rapida da addestrare, e le epoche sono state abbassate a 20.

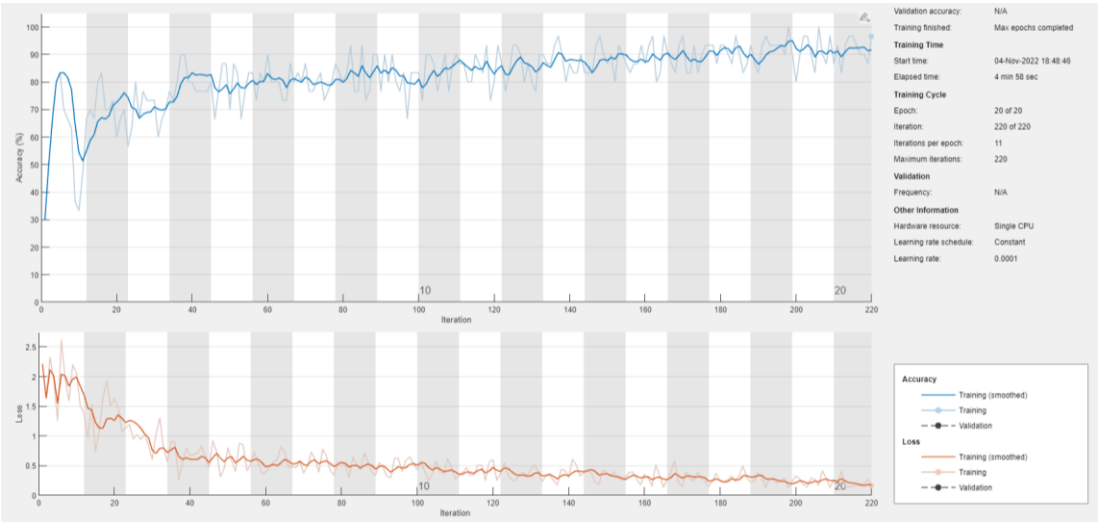

Figura 19: accuracy e loss function ottenute con AlexNet e 20 epoche.

In questo caso l'accuracy nel training si ferma attorno al 90% e nel test set si ottiene un accuracy nel test set di 0.7435897435897436.
Il risultato è un po' migliorato ma i valori sono ancora distanti, possiamo provare a diminuire ancora il numero delle epoche.

3.4.3 Test 3

Il terzo test è stato di nuovo eseguito solo su AlexNet e le epoche sono state ulteriormente abbassate a 15.

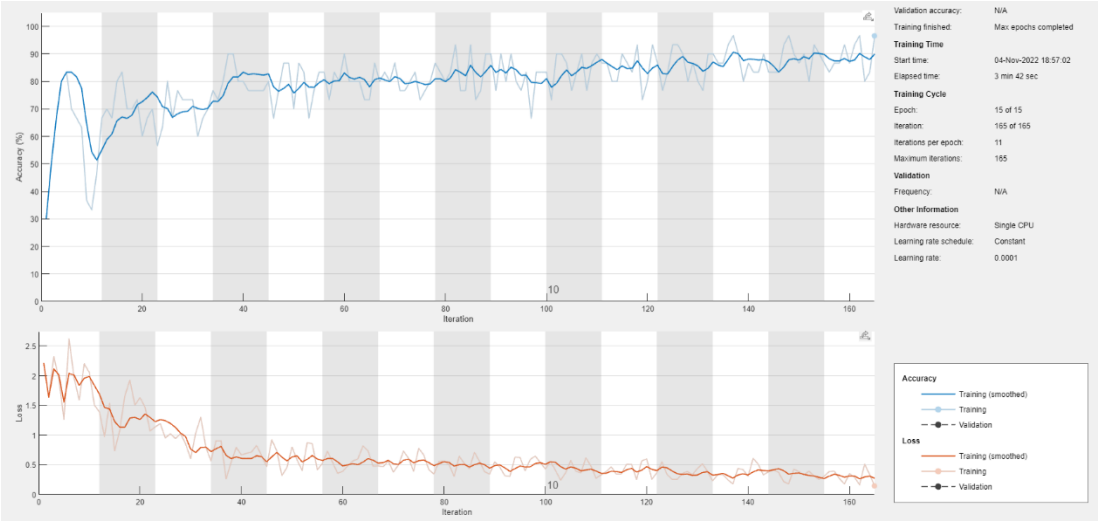

Figura 20: accuracy e loss function ottenute con AlexNet e 15 epoche.

In questo test l'accuracy si ferma appena sotto al 90% sul training set ma inizia ad abbassarsi anche nel test set fermandosi a 0.7008547008547008.
Evidentemente il training sta iniziando a diventare troppo poco e la rete perde prestazioni è quindi opportuno non diminuire ancora il numero di epoche.

3.4.4 Test 4

Dato che AlexNet ha offerto risulti migliori completando l'addestramento in 20 epoche è stato ripetuto il test con lo stesso numero di epoche anche con VGG16.

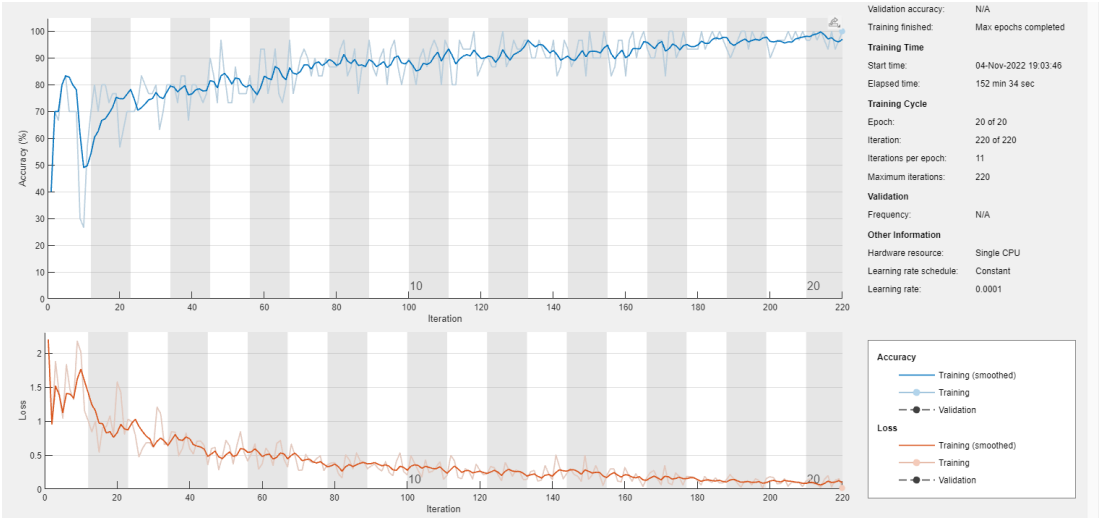

Figura 21: accuracy e loss function ottenute con VGG16 e 20 epoche.

VGG16 ottiene ancora un'accuracy molto alta nel training set mentre un'accuracy del 0.7350427350427351 nel test set. Un risultato molto simile a quello ottenuto 30, per VGG16 è probabilmente opportuno ridurre ulteriormente il numero di epoche.

3.4.4 Test 5

Per un ultimo tentativo è stato eseguito un test con VGG16 addestrata solamente 6 epoche.

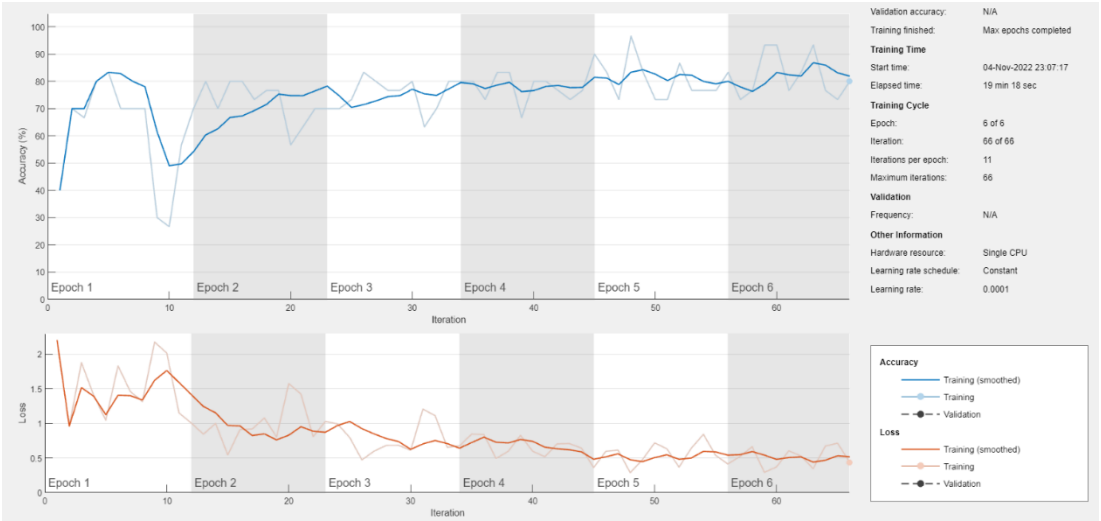

Figura 22: accuracy e loss function ottenute con VGG16 e 6 epoche.

Accuracy scende attorno all'80% nel training set ma si avvicina nel test set raggiungendo 0.7692307692307693, il miglior risultato ottenuto in totale.

Capitolo 4

Conclusioni

L'obiettivo che il lavoro di tesi mira a raggiungere è quello di addestrare tramite transfer learning una rete neurale AlexNet o VGG16 con lo scopo di catalogare automaticamente le specie marine, già rilevate in precedenza da un'altra rete neurale YOLOv4; al fine assistere i biologi dell'Università Politecnica delle Marche nei lavori di monitoraggio di aree marine.
L'esito è stato raggiunto, tramite l'addestramento delle due reti neurali, che seppur con risultati non ottimi, possono rivelarsi comunque un buon strumento per un'iniziale analisi dei dati raccolti. Un'accuracy del 75%, infatti, dopo le dovute considerazioni, denota che si sta procedendo verso la giusta strada per ottenere risultatati e che con ulteriore lavoro si potrebbe arrivare ad avere un software veramente prestante.
Per avere un quadro complessivo della situazione, bisogna poi ricordare che sono presenti alcune criticità nel dataset utilizzato per la classificazione:

- La dimensione del dataset utilizzato non è, in generale, molto ampia.

- Il dataset è per alcune specie sbilanciato, le specie più comuni hanno una grande rappresentanza mentre altre specie compaiono un numero molto limitato di volte. La specie più comune, che si muove spesso in banco, conta più di 300 istanze, al contrario la meno avvistata solo 8. Così vengono date poche possibilità alle reti di estrarre le features caratteristiche di una specie sottorappresentata e viene eseguito il test su un campione che potrebbe non essere abbastanza significativo.

- La qualità delle immagini, scattate spesso in condizioni non ottimali, è in alcuni casi scarsa. Ci sono immagini praticamente inutilizzabili per settimane a causa della rottura del tergicristallo della telecamera, in altri casi è la torbidità dell'acqua a creare problemi.

In conclusione, per migliorare la prestazione dei classificatori e per renderli uno strumento effettivamente utilizzabile senza troppe precauzioni sul campo, in futuro sarebbe opportuno:

- Espandere il dataset di training e quello di testing, specialmente per le specie che sono poco rappresentate, con l'aiuto dei biologi. In questo modo si può permettere alla rete di generalizzare meglio e verificare in maniera opportuna i risultati ottenuti.

- Migliorare la qualità delle immagini, soprattutto in condizioni di scarsa visibilità, come quando l'obiettivo della telecamera si sporca, magari utilizzando nuove tecnologie sempre più sofisticate di Image enhancement. Inoltre, potrebbe essere anche interessante aggiungere un'altra webcam vicino alla prima, utile in caso di guasti e per applicare altri algoritmi che possano, grazie a una visione stereoscopica, stimare la dimensioni dei pesci inquadrati.

Bibliografia

[1] “Cos'è la Computer Vision?”, IBM, retrieved 01 november 2022: https://www.ibm.com/it-it/topics/computer-vision

[2] *R. Venkatesan, B. Li, Convolutional Neural Networks in Visual Computing: A Concise Guide. CRC Press. ISBN 978-1-351-65032-8.*

[3] S. Dasiopoulou, V. Mezaris, I. Kompatsiaris, V. -K. Papastathis and M. G. Strintzis, "," in *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, no. 10, pp. 1210-1224, Oct. 2005, doi: 10.1109/TCSVT.2005.854238.

[4] M. J. Shafiee, B. Chywl, F. Li, A. Wong, “Fast YOLO: A Fast You Only Look Once System for Real-time Embedded Object Detection in Video”, 18 Sep 2017, https://doi.org/10.48550/arXiv.1709.05943

[5] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection”, 8 Jun 2015, https://doi.org/10.48550/arXiv.1506.02640

[6] C.-Y. Wang, H.-Y. M. Liao, Y.-H. Wu, P.-Y. Chen, J.-W. Hsieh, I-H. Yeh, “CSPNet: A new backbone that can enhance learning capability of cnn.”, 27 Nov 2019, https://doi.org/10.48550/arXiv.1911.11929

[7] K. He, X. Zhang, S. Ren, J. Sun, “Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition”, 23 Apr 2015, https://doi.org/10.48550/arXiv.1406.4729

[8] S. Liu, L. Qi, H. Qin, J. Shi, J. Jia, “Path Aggregation Network for Instance Segmentation”, 18 Sep 2018, https://doi.org/10.48550/arXiv.1803.01534

[9] J. Redmon, A. Farhadi, “YOLOv3: An Incremental Improvement”, 8 Apr 2018, https://doi.org/10.48550/arXiv.1804.02767

[10] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. Lawrence Zitnick, P. Dollár, “Microsoft COCO: Common Objects in Context”, 21 Feb 2015, https://doi.org/10.48550/arXiv.1405.0312

[11] C. Shorten, T. M. Khoshgoftaar, “A survey on Image Data Augmentation for Deep Learning.”, *J Big Data* **6**, 60 (2019). https://doi.org/10.1186/s40537-019-0197-0

[12] "Le proprietà ottiche dell'acqua", INFN - Istituto Nazionale di Fisica Nucleare, retrieved 01 november 2022, https://web.infn.it/fisicainbarca/images/stories/catania/GRiccobene_Proprieta_ottiche_acqua.pdf

[13] S. Ruder, "An overview of gradient descent optimization algorithms", 15 Jun 2017, https://doi.org/10.48550/arXiv.1609.04747

[14] D. M. W. Powers, "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation", 11 Oct 2020, https://doi.org/10.48550/arXiv.2010.16061

[15] A. Anwar, "What is Average Precision in Object Detection & Localization Algorithms and how to calculate it?", towardsdatascience, retrieved 01 november 2022, https://towardsdatascience.com/what-is-average-precision-in-object-detection-localization-algorithms-and-how-to-calculate-it-3f330efe697b

[16] A. Krizhevsky, I. Sutskever, G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks", Advances in Neural Information Processing Systems 25 (NIPS 2012), https://papers.nips.cc/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html

[17] S. Mascarenhas and M. Agarwal, "A comparison between VGG16, VGG19 and ResNet50 architecture frameworks for Image Classification," *2021 International Conference on Disruptive Technologies for Multi-Disciplinary Research and Applications (CENTCON)*, 2021, pp. 96-99, doi: 10.1109/CENTCON52345.2021.9687944.

[18] Raquel Urtasun, "Lecture 4: Neural Networks (PDF)", in "Introduction to Machine Learning", cs.toronto.edu, Università di Toronto - Dipartimento di Informatica, 2015, p. 19. retrieved 01 november 2022.

[19] "3.3. Metrics and scoring: quantifying the quality of predictions". scikit-learn, Retrieved 01 november 2022, https://scikit-learn.org/stable/modules/model_evaluation.html#accuracy-score