ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΙΓΑΙΟΥ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΑΚΩΝ ΚΑΙ ΕΠΙΚΟΙΝΩΝΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

Ανίχνευση Ελαττωμάτων σε Δίχτυα Ιχθυοκαλλιέργειας μέσω Υπολογιστικής Όρασης

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Παρασκευά Κωνσταντίνου

Επιβλέπουσα Καθηγήτρια: Δρ. Καβαλλιεράτου Εργίνα

Μέλη εξεταστικής επιτροπής: Δρ. Σταματάτος Ευστάθιος, Δρ. Καπόρης Αλέξιος

Σάμος, Ιανουάριος 2022

Ευχαριστίες

Με την ολοκλήρωση της Διπλωματικής μου εργασίας, θα ήθελα να ευχαριστήσω αρχικά την επιβλέπουσα καθηγήτρια κα. Εργίνα Καβαλλιεράτου για την πολύτιμη καθοδήγησή της, καθώς και την άμεση ανταπόκρισή της στα προβλήματα που προέκυψαν. Επιπλέον, ευχαριστώ την οικογένειά μου, για την οικονομική, ηθική και ψυχολογική υποστήριξη κατά τη διάρκεια των σπουδών μου. Τέλος, θα ήθελα να ευχαριστήσω τους φίλους και κοντινούς μου ανθρώπους για τις συμβουλές και την υποστήριξή τους κατά τη διάρκεια εκπόνησης της Διπλωματικής μου εργασίας.

ΠΑΡΑΣΚΕΥΑ ΚΩΝΣΤΑΝΤΙΝΟΥ
Τμήμα Μηχανικών Πληροφοριακών και Επικοινωνιακών Συστημάτων
ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΙΓΑΙΟΥ

Πίνακας περιεχομένων

Εισαγωγή **17**
1.1 Ελαττώματα σε Δίχτυα Ιχθυοκαλλιέργειας 17
1.2 Αντικείμενο διπλωματικής 18
1.3 Δομή της διπλωματικής 19

Ανίχνευση Αντικειμένων με Υπολογιστική Όραση **20**
2.1 Εισαγωγή 20
2.2 Convolutional Neural Networks (CNNs) 22
2.2.1 Είσοδος του CNN 23
2.2.2 Convolution Layer 25
2.2.3 Pooling Layer 27
2.2.4 Classification — Fully Connected Layer (FC Layer) 28
2.2.5 Error Function 29
2.2.6 Gradient Descent 30
2.2.7 Back-propagation 30
2.3 Σημαντικοί αλγόριθμοι που έχουν αναπτυχθεί 31
2.4 Faster R-CNN 32
2.5 You Only Look Once (YOLO) 35
2.6 Single Shot multibox Detector (SSD) 38
2.7 Σχετικές έρευνες ανίχνευσης ελαττωμάτων θαλάσσιας Βιορρύπανσης 40
2.7.1 Λογισμικό ανίχνευσης ελαττωμάτων σε Ιχθυοκαλλιέργειες 40
2.7.2 Ανίχνευση ελαττωμάτων Βιορρύπανσης σε σκάφη με CNN 41
2.7.3 Ανίχνευση ελαττωμάτων Βιορρύπανσης σε σκάφη με Haar Cascades 44

Εφαρμογή Αλγορίθμων Ανίχνευσης αντικειμένων σε δίχτυα Ιχθυοκαλλιέργειας **46**
3.1 Εισαγωγή 46
3.2 Εργαλεία και περιβάλλον εκπαίδευσης 47
3.2.1 TensorFlow 47
3.2.2 Anaconda 47
3.2.3 Tensorboard για την εξαγωγή αποτελεσμάτων 48
3.3 Δημιουργία του Dataset 50
3.3.1 Δημιουργία των εικόνων 50
3.3.2 Δημιουργία των labels 51
3.3.3 Μετατροπή .xml σε .csv 54

3.3.4 Δημιουργία του labelmap 55
3.3.5 Δημιουργία των tfrecords 55
3.4 Βήματα για τη δημιουργία των μοντέλων 56
3.4.1 Επιλογή μοντέλου 56
3.4.2 Λήψη προ-εκπαιδευμένων μοντέλων 56
3.4.3 Configuration των αλγορίθμων 57
3.4.4 Εκτέλεση της εκπαίδευσης και του evaluation των μοντέλων 57
3.5 Εκπαίδευση του Faster R-CNN 57
3.6 Εκπαίδευση του SSD 63
3.7 Εκπαίδευση του YOLOv3 67
3.8 Σύγκριση των αποτελεσμάτων 75
Συμπεράσματα - Προτάσεις μελλοντικής έρευνας 85
4.1 Συμπεράσματα 85
4.2 Προτάσεις μελλοντικής έρευνας 86
Βιβλιογραφία 88

Λίστα Εικόνων

- Εικόνα 1. Παράδειγμα ανίχνευσης αντικειμένων
- Εικόνα 2. Ανίχνευση γεωμετρικών σχημάτων
- Εικόνα 3. Απλουστευμένη απεικόνιση Νευρωνικού Δικτύου
- Εικόνα 4. Μετατροπή μιας εικόνας από πίνακα 33 σε διάνυσμα 91 για την είσοδο στο CNN.
- Εικόνα 5. Εικόνα μεγέθους 443 κανάλια χρωμάτων RGB
- Εικόνα 6. Συνέλιξη μιας εικόνας 551 με ένα φίλτρο 331
- Εικόνα 8. Εφαρμογή Same Padding σε εικόνα 551 με 0 δημιουργώντας μία εικόνα 771 στην οποία εφαρμόζεται φίλτρο 331 για να προκύψει convoluted feature ίδιου μεγέθους με την αρχική εικόνα. Πηγή: https://github.com/vdumoulin/conv_arithmetic
- Εικόνα 9. Τα είδη του Pooling (Max pooling και average pooling)
- Εικόνα 10. Απεικόνιση των βασικών λειτουργιών του Faster R-CNN Πηγή: https://arxiv.org/pdf/1506.01497.pdf
- Εικόνα 11. Λειτουργία sliding window του Faster R-CNN. Πηγή: [19]
- Εικόνα 12. Οπτικοποίηση βασικών λειτουργιών του YOLO. Πηγή: [16]
- Εικόνα 13. Αρχιτεκτονική του CNN του YOLO. Πηγή: [16]
- Εικόνα 14. Απεικόνιση των διαφορών των CNN του SSD και του YOLO. Πηγή: [17]
- Εικόνα 15. Manta net Cleaner
- Εικόνα 16. Αποτελέσματα του λογισμικού ανίχνευσης ελαττωμάτων. Πηγή:[47]
- Εικόνα 17. Σύστημα ανίχνευσης οργανισμών Βιορρύπανσης. Πηγή:[48]
- Εικόνα 18. Είδη οργανισμών Βιορρύπανσης. Πηγή:[48]
- Εικόνα 19. Γραφική διεπαφή ανίχνευσης οργανισμών. Πηγή:[48]
- Εικόνα 20. Dataset οργανισμών Barnacles. Πηγή:[50]
- Εικόνα 21. Αποτελέσματα ανίχνευσης οργανισμών μέσω Haar Cascade. Πηγή:[50]
- Εικόνα 22. Δημιουργία εικονικού περιβάλλοντος στο Anaconda
- Εικόνα 23. Το περιβάλλον του Tensorboard. Πηγή: https://www.tensorflow.org/tensorboard
- Εικόνα 24. Εκκίνηση του Tensorboard και εξαγωγή των αποτελεσμάτων σε τοπική σελίδα.
- Εικόνα 25. Παράδειγμα εικόνας του Dataset
- Εικόνα 26. Το λογισμικό LabelImg. Πηγή: https://github.com/tzutalin/labelImg
- Εικόνα 27. Χρήση του LabelImg στη δημιουργία του Dataset της παρούσας εργασίας
- Εικόνα 28. Απόκομμα του csv αρχείου της εκπαίδευσης
- Εικόνα 29. Λίστα μοντέλων υποστηριζόμενων από το Tensorflow. Source: https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md

- Εικόνα 30. Classification loss του Box Classifier σε σχέση με τον αριθμό των βημάτων εκπαίδευσης
- Εικόνα 31. Localization loss του Box Classifier σε σχέση με τον αριθμό των βημάτων εκπαίδευσης
- Εικόνα 32. Localization loss του RPN σε σχέση με τον αριθμό των βημάτων εκπαίδευσης
- Εικόνα 33. Objectness loss του RPN σε σχέση με τον αριθμό των βημάτων εκπαίδευσης
- Εικόνα 34. Total loss σε σχέση με τον αριθμό των βημάτων εκπαίδευσης
- Εικόνα 35.
- Εικόνα 36.
- Εικόνα 37.
- Εικόνα 38.
- Εικόνα 39.
- Εικόνα 40. Classification loss σε σχέση με τον αριθμό των βημάτων εκπαίδευσης
- Εικόνα 41. Localization loss σε σχέση με τον αριθμό των βημάτων εκπαίδευσης
- Εικόνα 42. Reguralization loss σε σχέση με τον αριθμό των βημάτων εκπαίδευσης
- Εικόνα 43. Total loss σε σχέση με τον αριθμό των βημάτων εκπαίδευσης
- Εικόνα 44.
- Εικόνα 45.
- Εικόνα 46.
- Εικόνα 47.
- Εικόνα 48.
- Εικόνα 49.
- Εικόνα 50. Confidence loss σε σχέση με τον αριθμό των βημάτων εκπαίδευσης
- Εικόνα 51. Confidence loss εστιασμένο σε σχέση με τον αριθμό των βημάτων εκπαίδευσης
- Εικόνα 52. IoU loss σε σχέση με τον αριθμό των βημάτων εκπαίδευσης
- Εικόνα 53. Prob loss σε σχέση με τον αριθμό των βημάτων εκπαίδευσης
- Εικόνα 54. Total loss σε σχέση με τον αριθμό των βημάτων εκπαίδευσης
- Εικόνα 55.
- Εικόνα 56. Συγκριτικά αποτελέσματα ακρίβειας των αλγορίθμων
- Εικόνα 57. Συγκριτικά αποτελέσματα Average Recall των αλγορίθμων
- Εικόνα 58. mean Average Precision σε σχέση με το Recall για τον YOLO
- Εικόνα 59. Στιγμιότυπο 1 δοκιμής του YOLO στα βίντεο Ιχθυοκαλλιέργειας
- Εικόνα 60. Στιγμιότυπο 2 δοκιμής του YOLO στα βίντεο Ιχθυοκαλλιέργειας
- Εικόνα 61. Στιγμιότυπο 1 δοκιμής του SSD στα βίντεο Ιχθυοκαλλιέργειας
- Εικόνα 62. Στιγμιότυπο 2 δοκιμής του SSD στα βίντεο Ιχθυοκαλλιέργειας
- Εικόνα 63. Στιγμιότυπο 3 δοκιμής του SSD στα βίντεο Ιχθυοκαλλιέργειας
- Εικόνα 64. Στιγμιότυπο 1 δοκιμής του Faster R-CNN στα βίντεο Ιχθυοκαλλιέργειας

- Εικόνα 65. Στιγμιότυπο 2 δοκιμής του Faster R-CNN στα βίντεο Ιχθυοκαλλιέργειας
- Εικόνα 66. Στιγμιότυπο 3 δοκιμής του Faster R-CNN στα βίντεο Ιχθυοκαλλιέργειας

Λίστα Πινάκων

- Πίνακας 1. Το Φίλτρο K που εφαρμόστηκε στην Εικόνα 6
- Πίνακας 2. Εγκατάσταση των απαραίτητων εργαλείων για την εκπαίδευση σε Tensorflow
- Πίνακας 3. Παράδειγμα annotation αρχείου μιας εικόνας του Dataset
- Πίνακας 4. Αρχείο labelmap
- Πίνακας 5. Απαιτούμενα πακέτα εκπαίδευσης του YOLO μέσω Tensorflow

Λίστα Τύπων

- Τύπος 1. Συνάρτηση σφάλματος Mean Sqared Error
- Τύπος 2. Συνάρτηση σφάλματος Cross Entropy
- Τύπος 3. Συνάρτηση Gradient Descent
- Τύπος 4. Υπολογισμός των απαιτούμενων Epochs
- Τύπος 5. Υπολογισμός του mean Average Precision
- Τύπος 6. Υπολογισμός του Average Recall

Ακρωνύμια

CNN	Convolutional Neural Network
AUV	Autonomous Underwater Vehicle
ROV	Remotely Operated Vehicle
YOLO	You Only Look Once
SSD	Single Shot Detector
COCO	Common Objects in Context
VOC	Visual Object Classes
mAP	mean Average Precision
RPN	Region Proposal Network
ANN	Artificial Neural Network
SIFT	Scale-Invariant Feature Transform
SURF	Speeded-Up Robust Features
MSE	Mean Squared Error
RoI	Region of Interest
IoU	Intersection over Union

Περίληψη

Στόχος της παρούσας διπλωματικής εργασίας είναι να αναπτυχθεί ένας αλγόριθμος Μηχανικής Μάθησης έτσι ώστε να ανιχνεύει επιτυχώς ελαττώματα σε δίχτυα Ιχθυοκαλλιέργειας. Η ανάπτυξη του συγκεκριμένου συστήματος θα έχει τη δυνατότητα να βοηθήσει στην αυτοματοποίηση του ελέγχου των διχτυών στις Ιχθυοκαλλιέργειες, γεγονός που ενδεχομένως να μειώσει τον χρόνο ελέγχου και να βελτιώσει την υγεία των εκτρεφόμενων ψαριών.

Στο πρώτο κεφάλαιο αναλύεται το θεωρητικό υπόβαθρο της Υπολογιστικής Όρασης. Αναλύονται μέθοδοι που έχουν εφαρμοστεί για απλές αλλά και περίπλοκες ανιχνεύσεις αντικειμένων από εικόνες. Στη συνέχεια, η εργασία στο δεύτερο κεφάλαιο επικεντρώνεται στις σύγχρονες μεθόδους Υπολογιστικής Όρασης με χρήση Μηχανικής Μάθησης. Αναλύεται η θεωρία και οι εφαρμογές των Νευρωνικών Δικτύων και γίνονται αναφορές στις πιο γνωστές έρευνες που έχουν διεξαχθεί στον τομέα. Συγκεκριμένα, αναλύονται οι αλγόριθμοι YOLO, SSD και Faster R-CNN. Στο τρίτο κεφάλαιο, γίνεται η αναφορά και περιγραφή των εργαλείων που χρησιμοποιήθηκαν καθώς και η πειραματική διαδικασία που ακολουθήθηκε για την εκπαίδευση και τη δοκιμή και των τριών αλγορίθμων. Οι αλγόριθμοι στη συνέχεια δοκιμάστηκαν σε βίντεο από προβληματικά δίχτυα για την οπτικοποίηση της ανίχνευσης των προβλημάτων. Στο τελευταίο κεφάλαιο, αναλύονται τα αποτελέσματα και επισημαίνονται οι προτάσεις για μελλοντικές έρευνες προς βελτίωση των αποτελεσμάτων.

Με την ολοκλήρωση της πειραματικής διαδικασίας, η ανίχνευση των ελαττωμάτων στα βίντεο των διχτυών Ιχθυοκαλλιέργειας που χρησιμοποιήθηκαν, αποδείχθηκε επιτυχής.

Λέξεις Κλειδιά: *Υπολογιστική Όραση, Αλγόριθμος, Νευρωνικά Δίκτυα, Μηχανική Μάθηση*

Abstract

The aim of this dissertation is to develop a Machine Learning algorithm to successfully detect defects in fish farming nets. The development of this system will be able to help automate the control of nets in fish farms, which may reduce inspection time and improve the health of farmed fish.

The first chapter analyzes the theoretical background of Computer Vision and specifically the methods of detecting simple and sophisticated objects in images. The second chapter focuses on modern methods of Computer Vision using Machine Learning. The theory and applications of Neural Networks are analyzed and references are made to the best known research that has been conducted in the field. Specifically, the algorithms YOLO, SSD and Faster R-CNN are analyzed. The third chapter, describes the tools used, as well as the experimental process followed for the training and testing of the selected algorithms. The algorithms were then tested on videos from faulty fish farming nets, to visualize the defect detection. Finally, the fourth chapter analyzes and compares the results from the different algorithms and future research for improvements of the results is proposed.

Upon completion of the experimental procedure, the detection of defects in the videos of the fish farming nets, proved to be successful.

Keywords: *Computer Vision, Algorithm, Neural Networks, Machine Learning*

1 *Εισαγωγή*

1.1 Ελαττώματα σε Δίχτυα Ιχθυοκαλλιέργειας

Στις Ιχθυοκαλλιέργειες χρησιμοποιούνται δίχτυα για τον εγκλωβισμό των εκτρεφόμενων ψαριών και για την προστασία τους από θαλάσσια θηλαστικά. Το υλικό κατασκευής, το σχήμα των διχτυών, καθώς και το μέγεθος των οπών του πλέγματός τους, διαφέρει ανάλογα με το είδος και το μέγεθος των εκτρεφόμενων ψαριών. Είναι σημαντικό τα δίχτυα να διατηρούνται σε καλή κατάσταση, τόσο για να προληφθεί η απώλεια των ψαριών από τυχόν μεγάλες οπές, όσο και για να εξασφαλιστεί η υγεία τους κατά την εκτροφή.

Ένα από τα μεγαλύτερα προβλήματα που καλείται να αντιμετωπίσει ο τομέας των Ιχθυοκαλλιεργειών είναι η Βιορρύπανση (Biofouling) [1]. Πρόκειται για ένα φαινόμενο κατά το οποίο αναπτύσσονται θαλάσσιοι μικροοργανισμοί, είτε φυτικοί είτε ζωικοί, στην επιφάνεια του διχτυού, με αποτέλεσμα να αυξάνεται το βάρος της κατασκευής και να προκαλούνται υλικές βλάβες, ή ακόμη και να εμποδίζεται η κυκλοφορία του νερού μέσα από το δίχτυ επηρεάζοντας τα επίπεδα οξυγόνου στο νερό. Η πτώση του οξυγόνου στο νερό εντός των κλωβών, μπορεί να οδηγήσει σε αύξηση της θνησιμότητας των εκτρεφόμενων ψαριών [2], [3]. Στα αρχικά στάδια της Βιορρύπανσης αναπτύσσεται μία λεπτή στρώση μικροοργανισμών, που στη συνέχεια μπορεί να εξελιχθεί στην ανάπτυξη μεγαλύτερων οργανισμών όπως άλγη, φύκια και οστρακοειδή.

Το πρόβλημα συνήθως προλαμβάνεται με την χρήση αντιμικροβιακών χημικών που αποτρέπουν την ανάπτυξη οργανισμών στα δίχτυα για ένα καθορισμένο χρονικό διάστημα

[4]. Καθώς οι συγκεκριμένες ουσίες καθυστερούν την εμφάνιση του προβλήματος και δεν την προλαμβάνουν πλήρως, το φαινόμενο της Βιορρύπανσης εντοπίζεται και αντιμετωπίζεται από καταδύσεις ρουτίνας που γίνονται από δύτες με σκοπό τον χειρονακτικό καθαρισμό των διχτυών, εργασία αρκετά επίπονη και δαπανηρή. Επιπλέον, εφαρμόζονται σύγχρονα συστήματα καθαρισμού με χρήση τηλεχειριζόμενων υποβρύχιων οχημάτων (ROVs), τα οποία διαθέτουν κάμερες για την πλοήγηση τους στην επιφάνεια των διχτυών και με χρήση πιδάκων νερού, απομακρύνουν τους προσκολλημένους οργανισμούς [5], [6].

Η χρήση ενός αυτοματοποιημένου συστήματος εντοπισμού των ελαττωμάτων των διχτυών, θα μπορούσε να μειώσει τον χρόνο και το κόστος συντήρησης και καθαρισμού τους, καθώς θα μπορούσε να οδηγήσει στον καθαρισμό στοχευμένων περιοχών των διχτυών που παρουσιάζουν πιο ανεπτυγμένα προβλήματα Βιορρύπανσης. Επιπλέον, η συνεχής ανάπτυξη των Αυτόνομων Υποβρυχίων Οχημάτων (AUVs) θα μπορούσε να επωφεληθεί από ένα σύστημα αυτόματου εντοπισμού των ελαττωμάτων με σκοπό την πλήρη αυτοματοποίηση του καθαρισμού των διχτυών Ιχθυοκαλλιέργειας.

1.2 Αντικείμενο διπλωματικής

Η συγκεκριμένη διπλωματική εργασία εκπονείται με σκοπό την ανάπτυξη ενός συστήματος αυτόματης ανίχνευσης ελαττωμάτων σε δίχτυα Ιχθυοκαλλιέργειας.

Στόχος του συγκεκριμένου συστήματος είναι να εφαρμοστεί σε Ιχθυοκαλλιέργειες προκειμένου να αυτοματοποιηθεί η διαδικασία εντοπισμού των ελαττωμάτων στα δίχτυα. Αυτό μπορεί να οδηγήσει στο συχνότερο έλεγχο των διχτυών, καθώς θα απαιτεί λιγότερο χρόνο και χαμηλότερο κόστος, και κατ’ επέκταση σε αποδοτικότερη συντήρηση των διχτυών. Γενικά, η κατάσταση των αλιευτικών μέσων επηρεάζει σημαντικά την υγεία και την ευζωία των εκτρεφόμενων ψαριών, επομένως το συγκεκριμένο σύστημα θα μπορούσε να βελτιώσει σημαντικά τη λειτουργία μιας Ιχθυοκαλλιεργητικής μονάδας.

Η μέθοδος που αναπτύσσεται στη συγκεκριμένη έρευνα, χρησιμοποιεί Μηχανική Μάθηση με Νευρωνικά Δίκτυα για την ανίχνευση των ελαττωμάτων των διχτυών μέσα από εικόνες. Αυτό απαιτεί τη δημιουργία μιας συλλογής εικόνων από παρόμοιες συνθήκες με αυτές στους κλωβούς Ιχθυοκαλλιέργειας, καθώς και την εκπαίδευση ορισμένων Νευρωνικών Δικτύων με αυτές τις εικόνες, προκειμένου να επιλεχθεί ο καταλληλότερος αλγόριθμος για τη συγκεκριμένη περίπτωση.

Συγκεκριμένα, για τη δημιουργία της συλλογής των εικόνων χρησιμοποιήθηκαν βίντεο από δίχτυα Ιχθυοκαλλιέργειας των εγκαταστάσεων των Ιχθυοτροφείων Κεφαλονιάς, από τα οποία έπρεπε να γίνει εξαγωγή και διαχωρισμός των ευδιάκριτων χρήσιμων εικόνων από τις δυσδιάκριτες, λόγω κίνησης και φωτισμού. Στη συνέχεια χρησιμοποιήθηκε η

συγκεκριμένη συλλογή εικόνων για την εκπαίδευση τριών αλγορίθμων ανίχνευσης αντικειμένων. Στο τέλος συγκρίθηκαν τα αποτελέσματα τους και εφαρμόστηκαν οι αλγόριθμοι στα αρχικά βίντεο προκειμένου να οπτικοποιηθεί το αποτέλεσμα της ανίχνευσης ελαττωμάτων, αποδεικνύοντας τη δυνατότητα χρήσης τους μελλοντικά σε παρόμοιες περιπτώσεις.

Το αποτέλεσμα της συγκεκριμένης εργασίας είναι η ανάπτυξη ενός λογισμικού ανίχνευσης ελαττωμάτων, το οποίο μπορεί να εκτελεστεί από φορητό υλισμικό ενός Αυτόνομου Υποβρύχιου Οχήματος.

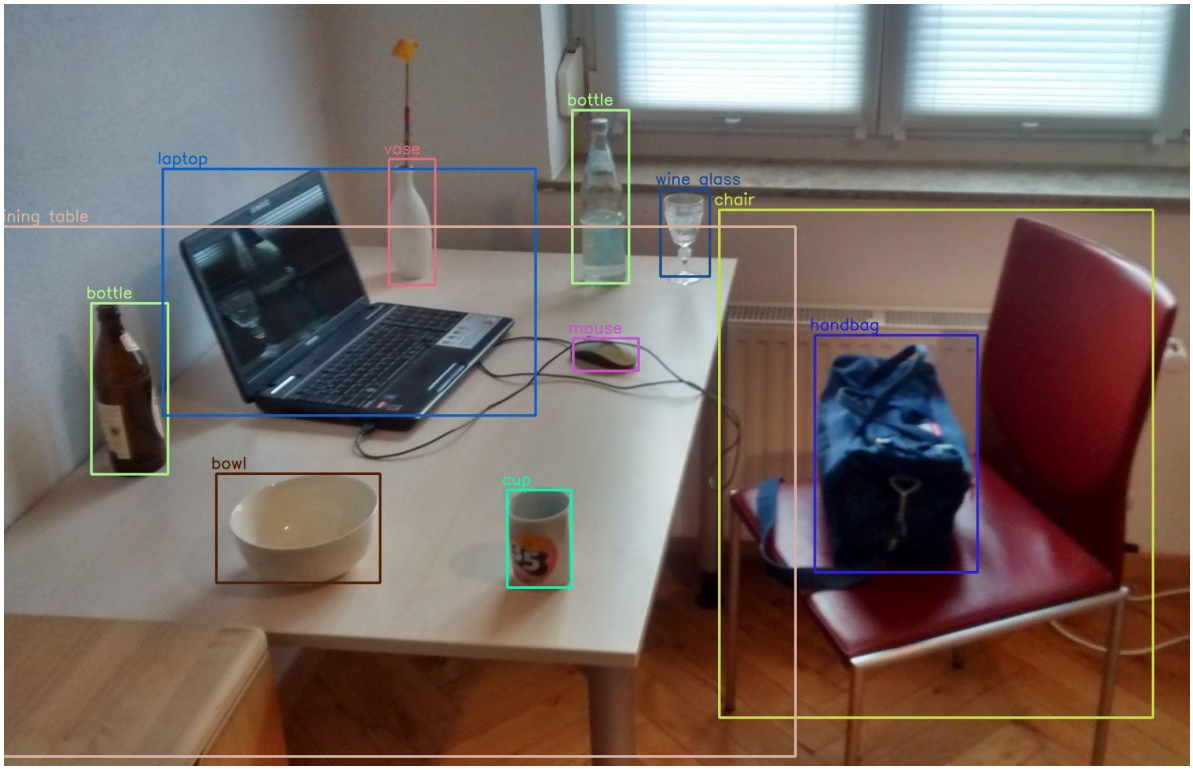
1.3 Δομή της διπλωματικής

Το θεωρητικό υπόβαθρο της Ανίχνευσης αντικειμένων, των Νευρωνικών Δικτύων, των σχετικών αλγορίθμων που έχουν αναπτυχθεί, καθώς και παρόμοιες έρευνες για την εφαρμογή αλγορίθμων ανίχνευσης ελαττωμάτων Βιορρύπανσης αναλύονται στο Κεφάλαιο 2. Στο Κεφάλαιο 3 αναπτύσσεται η μεθοδολογία που ακολουθήθηκε στη συγκεκριμένη εργασία, περιγράφεται η προετοιμασία των δεδομένων, η εκπαίδευση των νευρωνικών δικτύων για τους αλγορίθμους Faster R-CNN, SSD και YOLO, η εφαρμογή τους στον εντοπισμό ελαττωμάτων στα δίχτυα και γίνεται η σύγκριση των αποτελεσμάτων. Στο Κεφάλαιο 4 αναφέρονται τα συμπεράσματα της εργασίας και προτάσεις για μελλοντικές έρευνες.

2 *Ανίχνευση Αντικειμένων με Υπολογιστική Όραση*

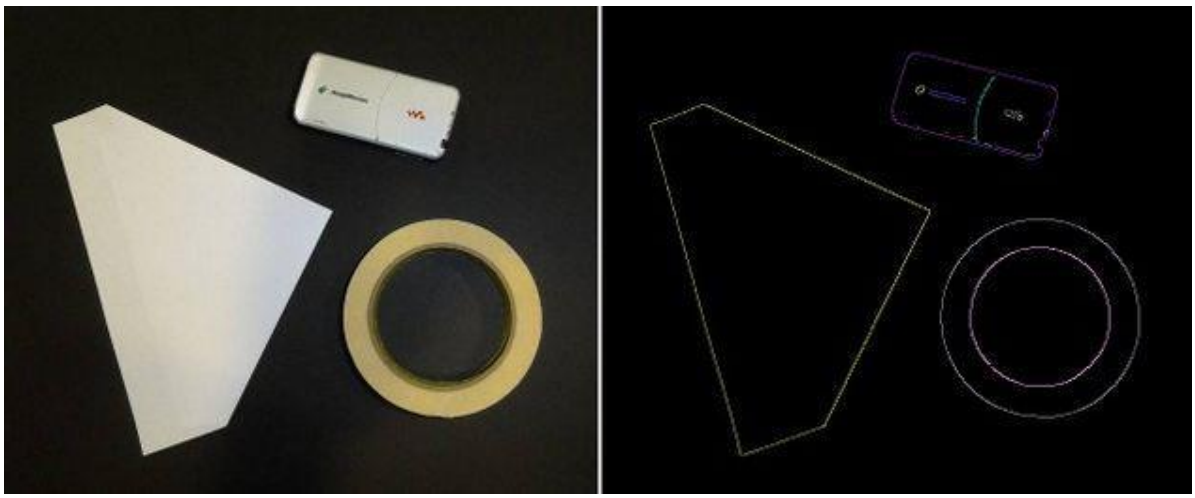
2.1 Εισαγωγή

Ως Ανίχνευση Αντικειμένων (Object Detection) ορίζεται η τεχνολογία που μέσω της Υπολογιστικής Όρασης (Computer Vision) και της Επεξεργασίας Εικόνας (Image Processing), αποσκοπεί στον εντοπισμό και την κατηγοριοποίηση των αντικειμένων μιας εικόνας [7]. Η κατηγοριοποίηση των αντικειμένων (Classification) αφορά τον εντοπισμό της κλάσης στην οποία ανήκει το κάθε αντικείμενο, για παράδειγμα σε κάμερες ασφαλείας για δημόσιους χώρους μπορεί να χρησιμοποιούνται οι κλάσεις “pedestrian” και “car” για την κατηγοριοποίηση πεζών και αυτοκινήτων αντίστοιχα [8]. Συνήθως η ανίχνευση των αντικειμένων οπτικοποιείται με τη χρήση ενός περιγράμματος γύρω από το αντικείμενο (το οποίο ονομάζεται Bounding Box) ενώ η κατηγοριοποίηση του αντικειμένου οπτικοποιείται με την απεικόνιση του ονόματος της κλάσης που ανήκει όπως φαίνεται στην Εικόνα 1.



Εικόνα 1. Παράδειγμα ανίχνευσης αντικειμένων. Πηγή: https://commons.wikimedia.org/wiki/File:Detected-with-YOLO--Schreibtisch-mit-Objekten.jpg

Στην πιο απλή της μορφή, η Ανίχνευση Αντικειμένων σε εικόνες μπορεί να γίνει με την επεξεργασία ορισμένων παραμέτρων όπως φωτεινότητα, αντίθεση, χρώμα, κορεσμός, κ.ά ανιχνεύοντας έτσι απλά γεωμετρικά σχήματα (Εικόνα 2). Με αυτό τον τρόπο τροποποιείται η εικόνα έτσι ώστε να απομονωθούν τα σχήματα προς αναγνώριση και έπειτα με την χρήση μαθηματικών εξισώσεων προσδιορίζεται το σχήμα και η θέση τους στην εικόνα [9]. Ορισμένες από αυτές τις μεθόδους περιλαμβάνουν την Ανίχνευση Ακμών (Edge Detection)[10], Ανίχνευση Κύκλων (Circle Detection) [11] και Ανίχνευση Παρασκηνίου (Background Detection) [12]. Η χρήση ορισμένων από τα προαναφερθέντα εργαλεία ανίχνευσης χαρακτηριστικών χρησιμοποιείται σε αλγορίθμους όπως οι Scale-Invariant Feature Transform (SIFT) [13] και Speeded-Up Robust Features (SURF) [14].



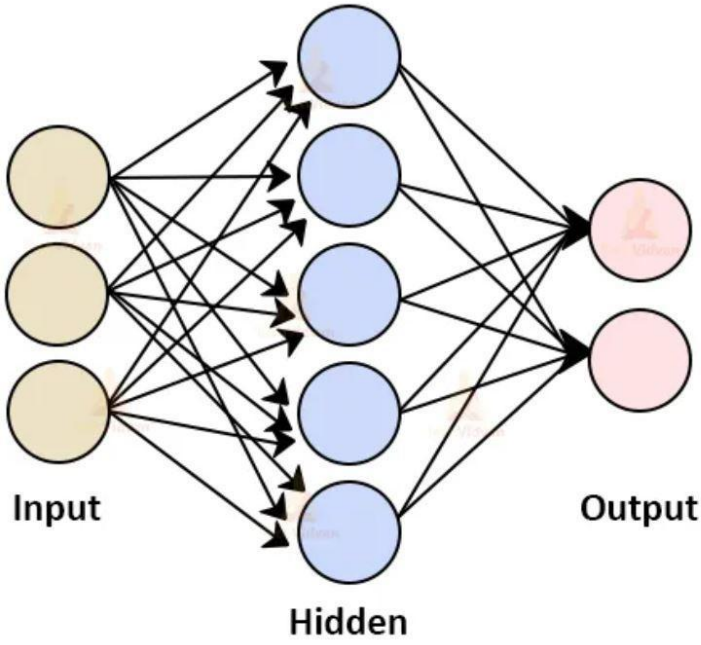
Εικόνα 2. Ανίχνευση γεωμετρικών σχημάτων

Η ανίχνευση πολύπλοκων και πιο αφηρημένων αντικειμένων συχνά επιτυγχάνεται με τρόπους Μηχανικής Μάθησης (Machine Learning) χρησιμοποιώντας Νευρωνικά Δίκτυα (Neural Networks). Αυτά τα Νευρωνικά Δίκτυα χρησιμοποιούνται σε αλγορίθμους ανίχνευσης των αντικειμένων για τα οποία εκπαιδεύτηκαν. Για την εκπαίδευση τους, χρησιμοποιείται ένα Σύνολο Δεδομένων (Dataset) το οποίο περιλαμβάνει μεγάλο αριθμό εικόνων που περιέχουν τα αντικείμενα προς αναγνώριση. Οι σύγχρονες μέθοδοι Ανίχνευσης Αντικειμένων μπορούν να χωριστούν σε δύο μεγάλες κατηγορίες, One-stage και Two-stage [15]. Οι μέθοδοι One-stage προτεραιοποιούν την ταχύτητα ανίχνευσης ενός αντικειμένου θυσιάζοντας την ακρίβεια ανίχνευσης, ενώ οι Two-stage μέθοδοι στοχεύουν στην ακριβή ανίχνευση των αντικειμένων αλλά τείνουν να απαιτούν περισσότερο χρόνο για την ανίχνευση. Παραδείγματα One-stage αλγορίθμων είναι οι You Only Look Once (YOLO) [16] και Single-Shot multibox Detector (SSD) [17], ενώ οι πιο γνωστοί Two-stage αλγόριθμοι είναι οι R-CNN [18], Faster R-CNN [19] και Libra R-CNN [20].

2.2 Convolutional Neural Networks (CNNs)

Τα Τεχνητά Νευρωνικά Δίκτυα (ANN) αποτελούν δίκτυα κόμβων (neurons) συνδεδεμένων μεταξύ τους οι οποίοι προσομοιώνουν τη λειτουργία των νευρώνων ενός βιολογικού εγκεφάλου. Κάθε νευρώνας μπορεί να επεξεργαστεί και μεταδώσει σήματα σε άλλους νευρώνες [21]. Κάθε σήμα αποτελεί έναν πραγματικό αριθμό, ο οποίος εισέρχεται στο δίκτυο μέσω των αρχικών νευρώνων, επεξεργάζεται από τον κάθε νευρώνα που διέρχεται και καταλήγει σε έναν τερματικό νευρώνα του δικτύου. Οι συνδέσεις μεταξύ των νευρώνων μπορούν να έχουν μία "προτίμηση" σε συγκεκριμένους νευρώνες με αποτέλεσμα να δημιουργείται μία "πιο επιθυμητή" διαδρομή για ένα σήμα που εισέρχεται στο δίκτυο αναλόγως την τιμή του σήματος κατά την είσοδο του. Οι συνδέσεις μεταξύ των νευρώνων

ονομάζονται άκρα (edges) και συνήθως διαθέτουν μία τιμή βάρους (weight) η οποία καθορίζει την "προτίμηση" του άκρου από τους νευρώνες του δικτύου που συνδέονται σε αυτό. Αυτά τα βάρη των άκρων ρυθμίζονται κατά την εκπαίδευση του Νευρωνικού Δικτύου προκειμένου τα σήματα εισόδου να διέρχονται από τους κατάλληλους νευρώνες χρησιμοποιώντας αυτά τα άκρα έτσι ώστε να καταλήξουν στους κατάλληλους τερματικούς νευρώνες του δικτύου, δίνοντας έτσι το επιθυμητό αποτέλεσμα. Η Εικόνα 3 απεικονίζει απλουστευμένα τη δομή ενός ANN.



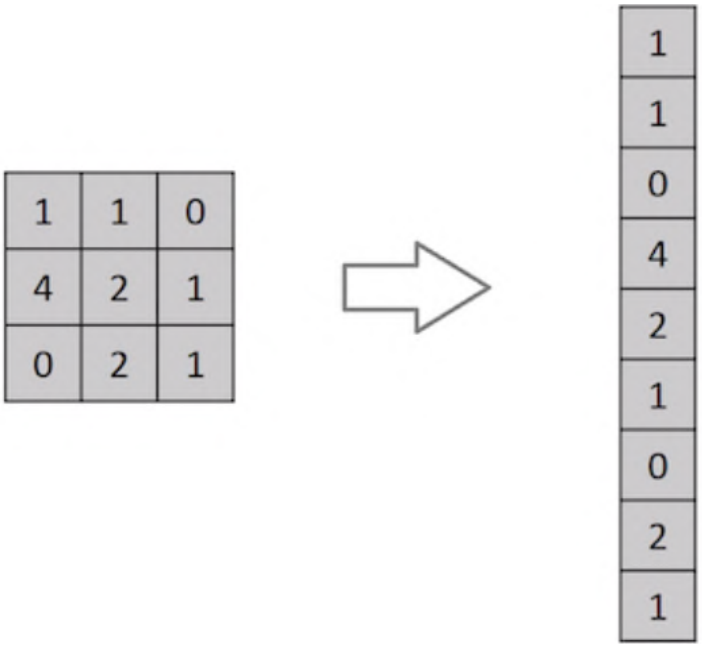
Εικόνα 3. Απλουστευμένη απεικόνιση Νευρωνικού Δικτύου

Convolutional Neural Networks ή CNNs αποτελούν ένα είδος Τεχνητών Νευρωνικών Δικτύων που χρησιμοποιούνται κυρίως στην ανάλυση οπτικών μέσων όπως εικόνες και βίντεο [22]. Διαθέτουν πολλά κοινά με άλλα είδη ANN όπως η δομή Feed-Forward με την οποία οι νευρώνες του δικτύου είναι δομημένοι σε στρώματα (layers). Το σήμα εισέρχεται στο αρχικό layer και επεξεργάζεται από τους νευρώνες του και το επεξεργασμένο σήμα μεταφέρεται στους κατάλληλους νευρώνες του επόμενου layer ανάλογα με τα βάρη των άκρων που τους συνδέουν. Η διαδικασία αυτή επαναλαμβάνεται μέχρι το επεξεργασμένο σήμα να καταλήξει σε ορισμένους από τους νευρώνες του τελικού layer.

2.2.1 Είσοδος του CNN

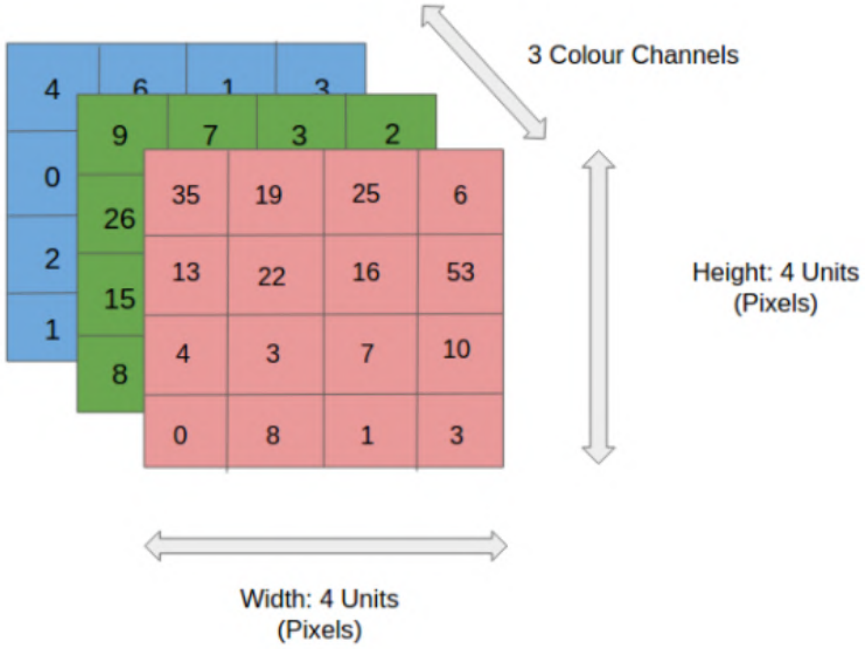
Σε απλές περιπτώσεις ανίχνευσης αντικειμένων όπως ανίχνευση απλών σχημάτων μέσα από grayscale εικόνες οι οποίες αντιστοιχούν σε δισδιάστατους πίνακες τιμών

φωτεινότητας, η είσοδος του CNN μπορεί να είναι η πεπλατυσμένη μορφή του πίνακα σε ένα διάνυσμα. Κάθε τιμή αυτού του διανύσματος θα αποτελεί την τιμή ενός νευρώνα στο αρχικό layer του δικτύου όπως φαίνεται στην Εικόνα 4.



Εικόνα 4. Μετατροπή μιας εικόνας από πίνακα 33 σε διάνυσμα 91 για την είσοδο στο CNN.

Σε πολλές περιπτώσεις, η εικόνα εισόδου αποτελείται από *3* ή περισσότερα κανάλια χρωμάτων που αντιστοιχούν σε ισάριθμους πίνακες τιμών φωτεινότητας για το κάθε χρώμα. Κατά την είσοδο της στο CNN, η εικόνα χωρίζεται στα επιμέρους κανάλια χρωμάτων της και η επεξεργασία που απαιτείται για την ανίχνευση αντικειμένων θα πρέπει να συμβεί σε καθένα από αυτά (Εικόνα 5).

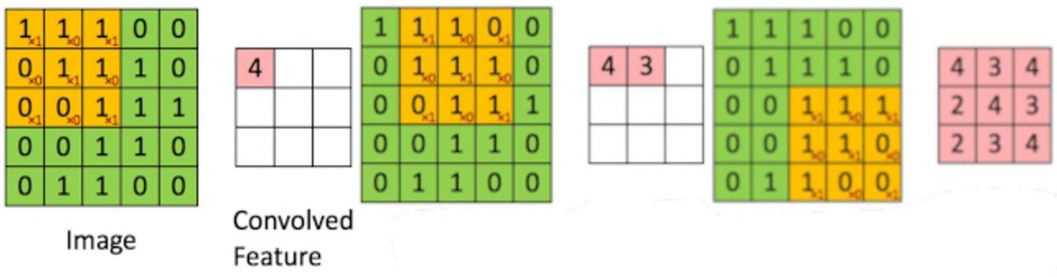


Εικόνα 5. Εικόνα μεγέθους 443 κανάλια χρωμάτων RGB

Καθώς αυτό απαιτεί μεγάλη υπολογιστική ισχύ για εικόνες μεγάλων διαστάσεων, οι εικόνες πρέπει να μειωθούν σε μέγεθος χωρίς να χαθεί πληροφορία για τα χαρακτηριστικά (features) που απεικονίζονται καθώς είναι κρίσιμα για την επιτυχή Ανίχνευση Αντικειμένων. Αυτή η διαδικασία είναι ιδιαίτερα σημαντική στη δημιουργία ενός Νευρωνικού Δικτύου με την ικανότητα να διαχειριστεί μεγάλο όγκο δεδομένων κατά την εκπαίδευση.

2.2.2 Convolution Layer

Πολλά από τα layers ενός CNN αποτελούν Συνελικτικά (Convolution) layers και έχουν σκοπό την εξαγωγή των χαμηλού και υψηλού επιπέδου features που βρίσκονται στην εικόνα εισόδου. Η διαδικασία αυτή γίνεται με την χρήση πινάκων που ονομάζονται πυρήνες (kernels) ή φίλτρα τα οποία έχοντας μικρότερες διαστάσεις από την εικόνα, εφαρμόζονται διαδοχικά σε αυτήν κάνοντας συνέλιξη των τιμών τους με τις τιμές της εικόνας στις οποίες εφαρμόζονται [23].



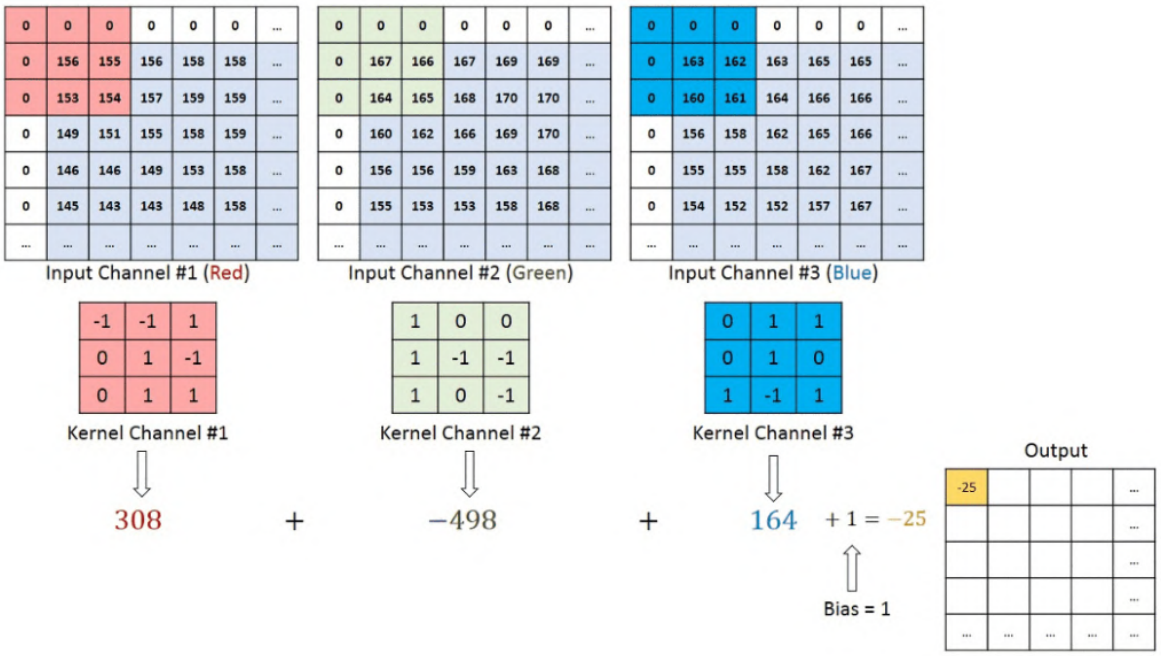
Εικόνα 6. Εφαρμογή φίλτρου βάθους 1

Στην Εικόνα 6 φαίνεται μία εικόνα μεγέθους $5 \times 5 \times 1$ και ένα φίλτρο (κίτρινο χρώμα) μεγέθους $3 \times 3 \times 1$ να εφαρμόζεται σταδιακά στην εικόνα.

```
Kernel/Filter, K =
1  0  1
0  1  0
1  0  1
```

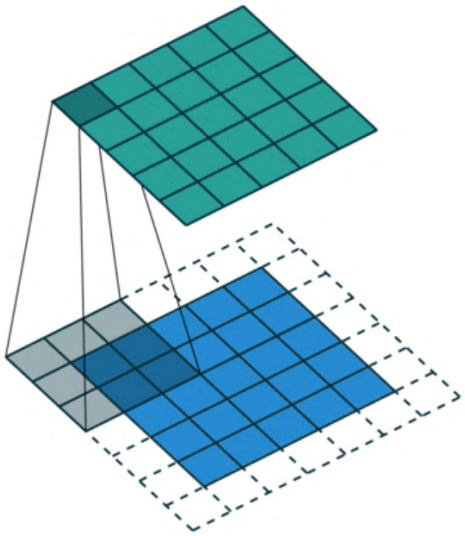
Πίνακας 1. Το Φίλτρο K που εφαρμόστηκε στην Εικόνα 6

Το φίλτρο K που παρουσιάζεται στον Πίνακα 1, έχοντας Μήκος Διασκελισμού (Stride Length) ίσο με *1*, θα κάνει συνολικά *9* προσπελάσεις πάνω από την εικόνα εφαρμόζοντας πολλαπλασιασμό πινάκων για το πλαίσιο της εικόνας πάνω από το οποίο βρίσκεται κάθε φορά, παράγοντας έτσι το convoluted feature μεγέθους $3 \times 3 \times 1$ που φαίνεται με χρώμα ροζ. Το φίλτρο κινείται οριζόντια κατά Stride Length μέχρι να γίνει προσπέλαση σε όλο το μήκος της εικόνας. Στη συνέχεια, μετακινείται προς τα κάτω κατά Stride Length και επαναλαμβάνει τη διαδικασία μέχρι να γίνει η προσπέλαση όλης της εικόνας. Σε περιπτώσεις με εικόνες πολλών χρωματικών καναλιών, το φίλτρο έχει το ίδιο βάθος με την εικόνα εισόδου. Ο πολλαπλασιασμός πινάκων εφαρμόζεται μεταξύ των *Kn* και *In* στοιβών ($[K1, I1]$; $[K2, I2]$; $[K3, I3]$), όπου *Kn* είναι ένα κανάλι του φίλτρου και *In* είναι ένα κανάλι της εικόνας. Τα αποτελέσματα προστίθενται μαζί με μία τιμή προτίμησης (bias) για να προκύψει ένα συμπιεσμένο Convoluted feature βάθους *1* (Εικόνα 7).



Εικόνα 7: Εφαρμογή φίλτρου με βάθος 3

Καθώς η διαδικασία της συνέλιξης έχει σκοπό να εξάγει όλα τα features μιας εικόνας, τα CNN χρησιμοποιούν πολλαπλά Convolutional layers για να εξάγουν σταδιακά ολοένα και πιο πολύπλοκα features. Ξεκινώντας από χαμηλού επιπέδου χαρακτηριστικά (low-level features) όπως ακμές, χρώμα, gradient και προσανατολισμό, μέχρι να επιτευχθεί σε μεγάλο βαθμό η ανίχνευση του περιεχομένου της εικόνας. Υπάρχουν δύο τύποι αποτελεσμάτων για αυτή τη λειτουργία, ένας στον οποίο το convoluted feature μειώνεται σε διαστάσεις σε σύγκριση με την είσοδο και ο άλλος στον οποίο η διάσταση είτε αυξάνεται είτε παραμένει η ίδια. Αυτό γίνεται με την εφαρμογή Valid Padding στην περίπτωση του πρώτου, ή Same Padding στην περίπτωση του δεύτερου. Στην περίπτωση του Same Padding, αυξάνεται η εικόνα (για παράδειγμα από $5 \times 5 \times 1$) σε μια εικόνα $7 \times 7 \times 1$ και, στη συνέχεια, εφαρμόζεται το φίλτρο $3 \times 3 \times 1$ πάνω της, δίνοντας έναν convoluted πίνακα με διαστάσεις $5 \times 5 \times 1$ (Εικόνα 8). Από την άλλη, η εκτέλεση της ίδιας λειτουργίας χωρίς padding (Valid Padding), θα εμφανίσει έναν πίνακα που έχει διαστάσεις του ίδιου του φίλτρου ($3 \times 3 \times 1$).

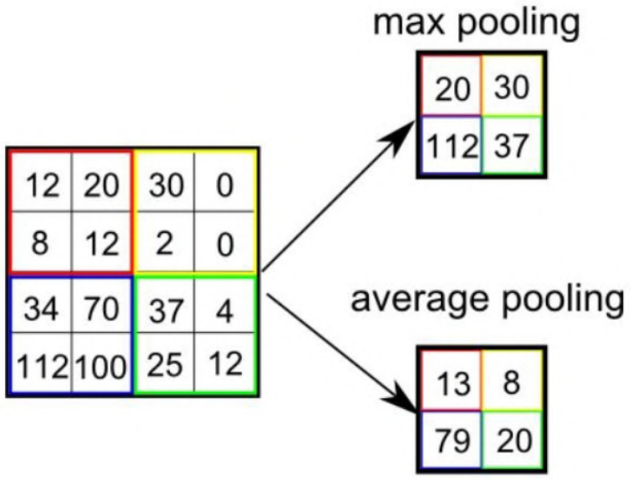


Εικόνα 8. Εφαρμογή Same Padding σε εικόνα $5 \times 5 \times 1$ με 0 δημιουργώντας μία εικόνα $7 \times 7 \times 1$ στην οποία εφαρμόζεται φίλτρο $3 \times 3 \times 1$ για να προκύψει convoluted feature ίδιου μεγέθους με την αρχική εικόνα. Πηγή: https://github.com/vdumoulin/conv_arithmetic

2.2.3 Pooling Layer

Παρομοίως με τα Convolutional layers, το Pooling layer είναι υπεύθυνο για τη μείωση των διαστάσεων της εικόνας του Convoluted feature. Αυτό χρειάζεται για την μείωση της υπολογιστικής ισχύος που απαιτείται για την επεξεργασία των δεδομένων. Επιπλέον, τα Pooling layers είναι χρήσιμα στην εξαγωγή των κυρίαρχων χαρακτηριστικών που είναι ανεξάρτητα περιστροφής και θέσης [24].

Υπάρχουν δύο τύποι Pooling layers, τα Max Pooling και τα Average Pooling. Τα Max Pooling επιστρέφουν τη μέγιστη τιμή από το πλαίσιο της εικόνας που καλύπτει το φίλτρο ενώ τα Average Pooling επιστρέφουν το μέσο όρο όλων των τιμών της περιοχής που καλύπτει το φίλτρο (Εικόνα 9).



Εικόνα 9. Τα είδη του Pooling (Max pooling και average pooling). Πηγή: https://poojamahajan5131.medium.com/max-pooling-210fc94c4f11

Το Max Pooling επιπλέον χρησιμεύει και σαν καταστολέας θορύβου καθώς απορρίπτει τις θορυβώδεις τιμές μειώνοντας έτσι και τον θόρυβο παράλληλα με το μέγεθος της εικόνας. Από την άλλη, το Average Pooling, απλώς μειώνει το μέγεθος της εικόνας χωρίς να προκύπτει κάποια επιπλέον μείωση θορύβου, επομένως το Max Pooling θεωρείται πιο αποτελεσματικό. Τα Convolutional layers και τα Pooling layers, μαζί σχηματίζουν τα στρώματα που αφορούν την εικόνα εισόδου ενός CNN. Ανάλογα με την πολυπλοκότητα των εικόνων, ο αριθμός των Pooling layers μπορεί να αυξηθεί για την περαιτέρω καταγραφή χαρακτηριστικών χαμηλού επιπέδου, αλλά με κόστος μεγαλύτερης υπολογιστικής ισχύος. Μετά την επιτυχή εφαρμογή των προαναφερθέντων διαδικασιών, έχει επιτευχθεί η ικανότητα του μοντέλου να ανιχνεύσει τα χαρακτηριστικά μιας εικόνας. Στη συνέχεια, γίνεται ο μετασχηματισμός της τελικής εξόδου σε μία διάσταση και η είσοδος της σε ένα κανονικό Νευρωνικό Δίκτυο για να γίνει το Classification.

2.2.4 Classification — Fully Connected Layer (FC Layer)

Η προσθήκη ενός πλήρως συνδεδεμένου επιπέδου είναι ένας εύκολος τρόπος εκμάθησης μη γραμμικών συνδυασμών των χαρακτηριστικών υψηλού επιπέδου (high-level features) όπως φαίνονται από την έξοδο του Convolutional layer. Το Fully-Connected layer μαθαίνει μια πιθανώς μη γραμμική συνάρτηση σε αυτόν τον χώρο [25].

Αφού έχει γίνει η μετατροπή της εικόνας εισόδου στην κατάλληλη μορφή από τα Convolutional και τα Pooling layers, γίνεται η μετατροπή της εικόνας σε μονοδιάστατο διάνυσμα. Αυτό το διάνυσμα εισάγεται σε ένα ANN και εφαρμόζεται η μεθοδολογία Back Propagation [26] για την εκπαίδευση του. Μετά από πολλές προσπελάσεις κατά την εκπαίδευση, το δίκτυο είναι σε θέση να ξεχωρίσει τα κυρίαρχα low-level features σε εικόνες και να τις κατηγοριοποιήσει με τεχνική Softmax Classification [27]. Υπάρχουν διάφορες αρχιτεκτονικές CNN που έχουν παίξει καθοριστικό ρόλο στην κατασκευή αλγορίθμων τεχνητής νοημοσύνης όπως οι LeNet [28], AlexNet [29], VGGNet [30], GoogLeNet [31], ResNet [32] και ZFNet [33].

2.2.5 Error Function

Error Function ή Συνάρτηση Σφάλματος είναι η συνάρτηση που χρησιμοποιείται στην εκπαίδευση ενός νευρωνικού δικτύου, η οποία δείχνει την απόκλιση που έχουν τα αποτελέσματα του δικτύου από τις επιθυμητές τιμές. Αναλόγως το είδος του νευρωνικού δικτύου και το είδος του προβλήματος που προσπαθεί να βελτιστοποιήσει, χρησιμοποιούνται διαφορετικές Συναρτήσεις Σφάλματος. Για παράδειγμα, συχνή επιλογή σε προβλήματα Regression είναι η συνάρτηση Μέσου Τετραγώνου του Σφάλματος ή Mean Squared Error (MSE) [34], ενώ σε προβλήματα Classification χρησιμοποιείται συνήθως η συνάρτηση Cross Entropy ή Log Loss [35].

$$MSE = \frac{\Sigma(y_i - \hat{y}_i)^2}{n}$$

Τύπος 1. Mean Squared Error

Ο Τύπος 1 αφορά την Mean Squared Error συνάρτηση όπου y_i είναι η i-οστή παρατηρήσιμη τιμή, $\widehat{y_i}$ είναι η αντίστοιχη προβλεπόμενη τιμή και n είναι ο αριθμός των παρατηρήσεων.

$$-\sum_{c=1}^{M} y_{o,c} log(p_{o,c})$$

Τύπος 2. Cross Entropy

Ο Τύπος 2 αφορά την Cross Entropy συνάρτηση όπου y είναι το επιθυμητό αποτέλεσμα (0 ή 1 επειδή εκφράζει πιθανότητα), p είναι η προβλεπόμενη πιθανότητα της παρατήρησης o να ανήκει στην κλάση c και M είναι ο αριθμός των κλάσεων.

Επομένως, δεδομένης μιας Συνάρτησης Σφάλματος E για τα ζευγάρια διανυσμάτων $X = \{(\vec{x}_i, \vec{y}_i),..., (\vec{x}_N, \vec{y}_N)\}$ *θα πρέπει η* $E(X, \theta)$ να είναι μικρή όταν $f_\theta(\vec{x}_i) \approx \vec{y}_i$ για όλα τα i, όπου $f_\theta(\vec{x})$ η συνάρτηση που υπολογίζει το ANN με παραμέτρους θ (weights, biases και τιμές κόμβων). Οπότε, η εκπαίδευση του ANN έχει ως σκοπό την ελαχιστοποίηση του σφάλματος $E(X, \theta)$ τροποποιώντας τις παραμέτρους θ (αφού το X είναι σταθερό).

2.2.6 Gradient Descent

Από τη στιγμή που η Συνάρτηση Σφάλματος $E(X, \theta)$ είναι συνάρτηση που περιλαμβάνει το αποτέλεσμα ενός ANN που αποτελείται από πολλές μη γραμμικές εξισώσεις, η εύρεση του ελάχιστου σε αυτή τη συνάρτηση με αναλυτικό τρόπο είναι σχεδόν αδύνατη. Για τον ρόλο αυτό, υπάρχει μία γενικευμένη μέθοδος εύρεσης ελάχιστου σε διαφοροποιήσιμες συναρτήσεις που ονομάζεται Gradient Descent. Ουσιαστικά, το Gradient Descent, βρίσκει την κλίση μιας συνάρτησης f σε μία συγκεκριμένη τιμή x και στη συνέχεια ανανεώνει την τιμή της κλίσης κάνοντας "βήματα" προς την αρνητική κατεύθυνση της κλίσης [36]. Σε γενικές γραμμές, θα προκύψει μία τιμή x' κοντά στην x όπου $x' = x - \eta\nabla f(x)$ (Τύπος 3) για την οποία $f(x') < f(x)$. Αυτή η διαδικασία επαναλαμβάνεται μέχρι να βρεθεί ένα τοπικό ελάχιστο ή η κλίση να γίνει μικρότερη από μία επιθυμητή ελάχιστη (threshold) τιμή. Η εκπαίδευση του ANN συχνά ξεκινάει με την τυχαία αρχικοποίηση των παραμέτρων θ οι οποίες ανανεώνονται συνεχώς με βάση το Gradient Descent που εφαρμόζεται, μέχρι η Συνάρτηση Σφάλματος να ελαχιστοποιηθεί.

Από τη στιγμή που οι τιμές δεν υπολογίζονται αλλά βελτιώνονται σταδιακά, η μέθοδος Gradient Descent μπορεί να εφαρμοστεί σε διαδοχικά ζευγάρια εισόδων-επιθυμητών εξόδων για ορισμένες επαναλήψεις, δημιουργώντας το λεγόμενο Batch Learning, το οποίο σε ορισμένες περιπτώσεις μπορεί να οδηγήσει σε αποτελέσματα συγκρίσιμα με αποτελέσματα εφαρμογής Gradient Descent σε ολόκληρο Dataset [37].

Το αρνητικό της μεθόδου Gradient Descent είναι ότι μπορεί να βρει μόνο τοπικό ελάχιστο και όχι το πραγματικό ελάχιστο της συνάρτησης σφάλματος.

2.2.7 Back-propagation

Ο υπολογισμός του Gradient για Νευρωνικά Δίκτυα με πολλές παραμέτρους και πολλά layers είναι δύσκολος, καθώς η Συνάρτηση Σφάλματος απέχει πολύ από τη γραμμικότητα. Επομένως, η μέθοδος Backpropagation χρησιμοποιείται για την εκπαίδευση των Feed-Forward Νευρωνικών Δικτύων [34]. Ο αλγόριθμος Backpropagation για να λειτουργήσει χρειάζεται ένα Dataset αποτελούμενο από N ζευγάρια εισόδων-επιθυμητών εξόδων $(\vec{xi}, \vec{yi})$, όπου $\vec{xi}$ η είσοδος και $\vec{yi}$ η επιθυμητή έξοδος του Νευρωνικού Δικτύου.

Επιπλέον, απαιτείται ένα Νευρωνικό Δίκτυο με παραμέτρους όπως weights, biases και τιμές εξόδου των νευρώνων, οι οποίες συνολικά συμβολίζονται με θ. Οι πιο σημαντικές παράμετροι είναι τα weights w, όπου w_{ij}^{k} η τιμή weight μεταξύ του κόμβου j στο layer l_k και του κόμβου i στο layer l_{k-1}, και τα biases b, όπου b_i^k η τιμή bias για τον κόμβο i στο layer l_k. Επίσης, ο αλγόριθμος Backpropagation χρειάζεται μία Συνάρτηση Σφάλματος $E(X, \theta)$ για τον υπολογισμό του σφάλματος της τιμής που έχει προβλεφθεί σε σχέση με την επιθυμητή τιμή.

Έχοντας ένα Νευρωνικό Δίκτυο και μια Συνάρτηση Σφάλματος, ο αλγόριθμος Backpropagation υπολογίζει το Gradient της Συνάρτησης Σφάλματος με βάση τις παραμέτρους του δικτύου (weights και biases). Ο υπολογισμός του Gradient ξεκινάει από τα τελευταία layers του δικτύου και προχωράει προς τα αρχικά. Ορισμένοι από τους υπολογισμούς του Gradient ενός layer χρησιμοποιούνται και στον επόμενο υπολογισμό του Gradient του προηγούμενου layer. Αυτή η ροή της πληροφορίας του σφάλματος επιτρέπει τον αποδοτικότερο υπολογισμό του Gradient σε κάθε layer σε σχέση με τον υπολογισμό του Gradient μεμονωμένα για κάθε layer.

2.3 Σημαντικοί αλγόριθμοι που έχουν αναπτυχθεί

Οι καθιερωμένες μέθοδοι Ανίχνευσης Αντικειμένων βασίζονται είτε σε τεχνικές Ολίσθησης Παραθύρων (Sliding Window), κατά τις οποίες εφαρμόζεται ένα φίλτρο σταδιακά σε όλη την εικόνα (Deformable Part Model [38]), είτε σε Προτάσεις Περιοχών (Region Proposals) για τη θέση ενός αντικειμένου σε μία εικόνα (Selective Search [39]). Πριν την ευρεία χρήση των CNN οι προαναφερθείσες μέθοδοι είχαν συγκρίσιμη απόδοση, μέχρι που η επιτυχία του αλγορίθμου R-CNN [18], ο οποίος συνδυάζει Selective Search με κατηγοριοποίηση βασισμένη σε CNN δίκτυο, οδήγησε στην επικράτηση των Selective Search προσεγγίσεων.

Ο αλγόριθμος R-CNN έχει βελτιωθεί με διάφορους τρόπους. Οι πρώτες προσεγγίσεις βελτιώνουν την ποιότητα και την ταχύτητα του Post-Classification, καθώς αυτό απαιτεί την ταξινόμηση χιλιάδων τμημάτων μιας εικόνας, το οποίο είναι χρονοβόρο. Ο αλγόριθμος SPPnet [40] επιταχύνει σημαντικά τον R-CNN. Εισάγει ένα νέο Pooling layer, το οποίο είναι πιο σταθερό στις διαφορετικές θέσεις και στην κλίμακα των περιοχών που εξετάζονται, και επιτρέπει στα Classification layers να επαναχρησιμοποιούν χαρακτηριστικά (features), τα οποία έχουν υπολογιστεί χρησιμοποιώντας Feature Maps από εικόνες με διαφορετική ανάλυση. Ο αλγόριθμος Fast R-CNN επεκτείνει τον SPPnet έτσι

ώστε να ρυθμίζονται κατάλληλα όλα τα layers, ελαχιστοποιώντας τις αποκλίσεις των Confidence Values και των θέσεων των Bounding Boxes.

Ακόμη, άλλες προσεγγίσεις έχουν βελτιώσει τον R-CNN ως προς την ποιότητα των Προτάσεων Περιοχών των αντικειμένων με χρήση Deep Neural Networks. Στην προσέγγιση Multibox [41], οι Προτάσεις Περιοχών για Selective Search αντικαθίστανται από προτάσεις που δημιουργούνται από ένα ξεχωριστό Deep Neural Network, πράγμα που βελτιώνει την ακρίβεια ανίχνευσης, αλλά αποτελεί πολύπλοκη προσέγγιση στη χρήση, καθώς απαιτεί την εκπαίδευση δύο ξεχωριστών Νευρωνικών Δικτύων τα οποία αλληλοεξαρτώνται. Ο αλγόριθμος Faster R-CNN [19] αντικαθιστά τις Προτάσεις Περιοχών για Selective Search με προτάσεις που προέρχονται από ένα Region Proposal Network (RPN) το οποίο εφαρμόζεται με τον Fast R-CNN εναλλάσοντας Convolutional layers που χρησιμοποιούνται για τελειοποίηση (fine-tuning) με layers που παράγουν προβλέψεις περιοχών (Prediction layers). Με αυτό τον τρόπο, οι Προτάσεις Περιοχών χρησιμοποιούνται για να συγκεντρώσουν μεσαίου επιπέδου χαρακτηριστικά εικόνων (mid-level features), κάνοντας το τελικό Classification λιγότερο χρονοβόρο.

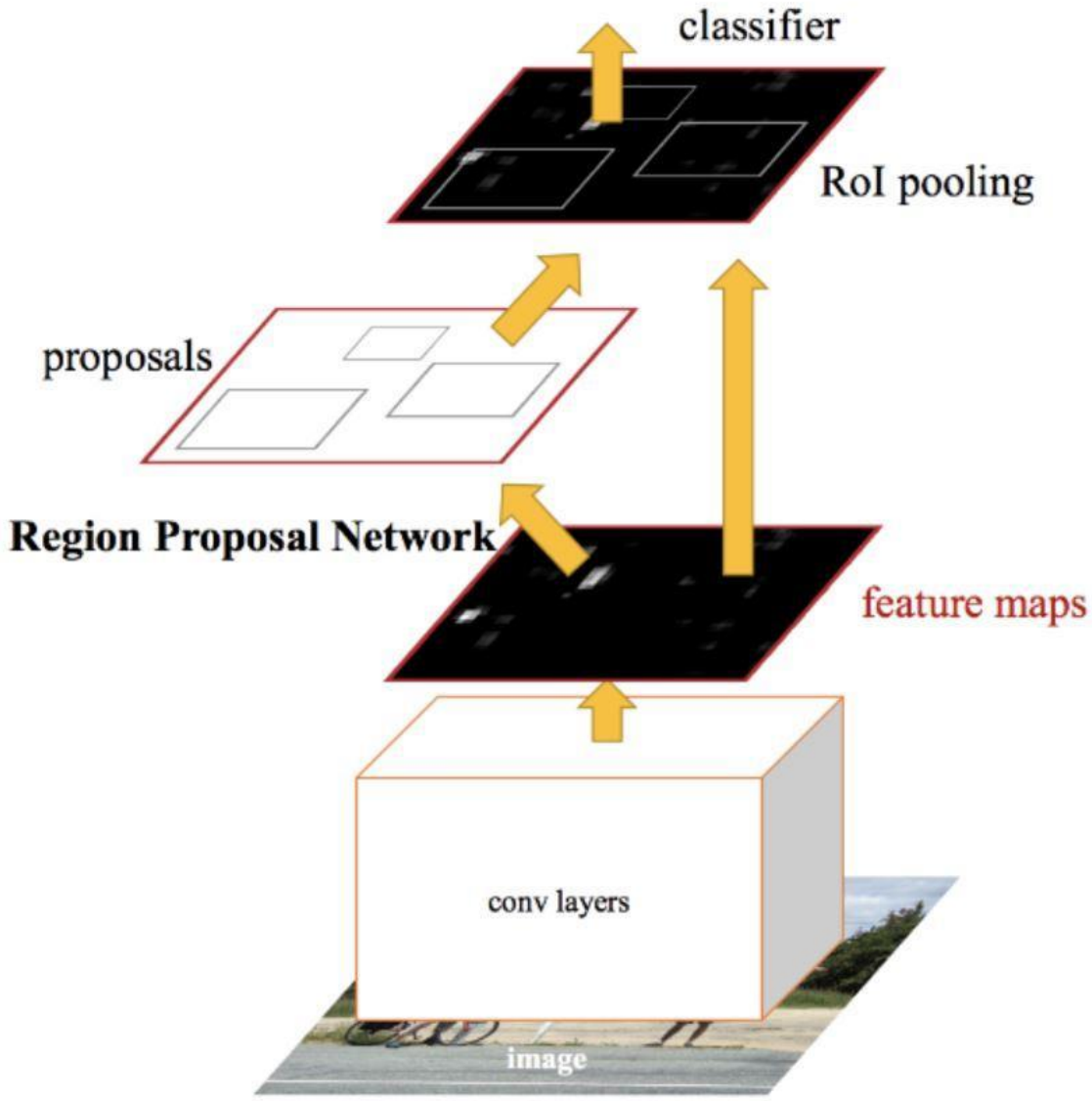
Επιπλέον, υπάρχουν άλλες μέθοδοι που παρακάμπτουν το βήμα των Προτάσεων Περιοχών και προβλέπουν απευθείας Bounding Boxes και Confidence Values. Ο OverFeat αλγόριθμος [42] αποτελεί μία έκδοση της μεθόδου Sliding Window με χρήση Deep Neural Networks, ο οποίος προβλέπει απευθείας ένα Bounding Box από κάθε τοποθεσία του κορυφαίου Feature Map, γνωρίζοντας τα Confidence Values των κατηγοριών των αντικειμένων προς ανίχνευση. Ο αλγόριθμος YOLO [16] χρησιμοποιεί ολόκληρο το κορυφαίο Feature Map για να προβλέψει τα Confidence Values για πολλαπλές κατηγορίες αντικειμένων και τα Bounding Boxes τους. Ο αλγόριθμος SSD [17] λειτουργεί με παρόμοιο τρόπο, με την έννοια ότι παραβλέπει τις προτάσεις περιοχών και χρησιμοποιεί τα προκαθορισμένα Bounding Boxes. Επιπλέον, ο SSD επιτρέπει τη χρήση Bounding Boxes με διαφορετικές αναλογίες σε κάθε τοποθεσία ενός feature από πολλά Feature Maps και σε διαφορετική κλίμακα.

2.4 Faster R-CNN

Ο Faster R-CNN αλγόριθμος αναπτύχθηκε το 2015 από τους Shaoqing Ren, Kaiming He, Ross Girshick και Jian Sun με σκοπό τη βελτίωση των υπαρχόντων αλγορίθμων ανίχνευσης αντικειμένων, οι οποίοι βασίζονταν σε Προτάσεις Περιοχών (Region Proposals) στους αλγορίθμους τους για τον εντοπισμό των αντικειμένων [19]. Οι αλγόριθμοι SPPnet και Fast R-CNN βελτίωσαν αρκετά το χρονικό κόστος της Ανίχνευσης Αντικειμένων αποκαλύπτοντας τον πραγματικό περιοριστικό παράγοντα στην ταχύτητα ανίχνευσης, τον υπολογισμό των θέσεων των αντικειμένων. Ο Faster R-CNN εισάγει ένα Δίκτυο Πρότασης Περιοχών (Region Proposal Network ή RPN) το οποίο μοιράζεται την πλήρη εικόνα

εισόδου με το δίκτυο ανίχνευσης των αντικειμένων μειώνοντας αρκετά το κόστος πρόβλεψης των περιοχών.

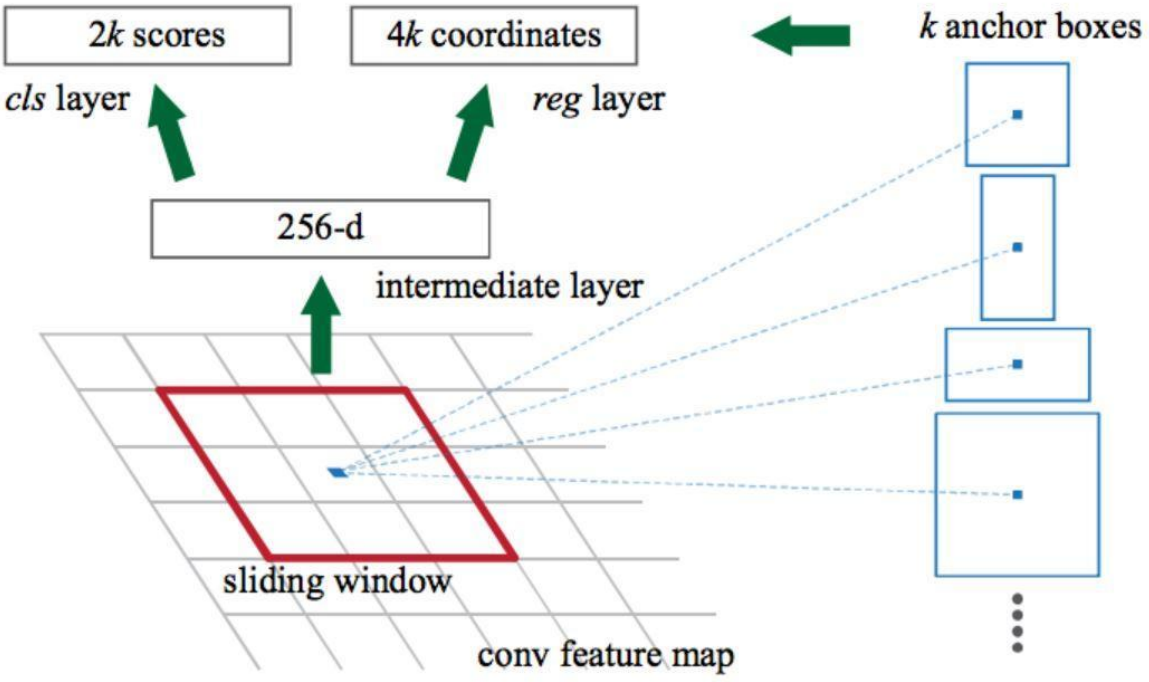
Ο Faster R-CNN αποτελεί μία τροποποιημένη έκδοση του Fast R-CNN, με την κύρια διαφορά να είναι ο τρόπος εντοπισμού των θέσεων των αντικειμένων. Ενώ ο Fast R-CNN χρησιμοποιεί Επιλεκτική Αναζήτηση (Selective Search), ο Faster R-CNN χρησιμοποιεί το προαναφερθέν RPN το οποίο δέχεται έναν Χάρτη Χαρακτηριστικών (Feature Map) σαν είσοδο και παράγει έναν αριθμό προβλέψεων αντικειμένων με συγκεκριμένη τιμή σιγουριάς (Confidence Value) για το καθένα. Στη προσέγγιση του Faster R-CNN, αρχικά η εικόνα εισόδου διέρχεται από ένα CNN το οποίο παράγει το Feature Map που απαιτείται για το Region Proposal. Στη συνέχεια, το RPN δέχεται αυτό το Feature Map και παράγει τις προτάσεις των αντικειμένων μαζί με τα Confidence Values τους. Έπειτα, εφαρμόζεται ένα Region of Interest Pooling layer προκειμένου να μετατραπούν όλες οι προτάσεις σε συγκεκριμένο μέγεθος. Τέλος, οι προτάσεις περνάνε από ένα πλήρως συνδεδεμένο layer το οποίο αναλαμβάνει την κατηγοριοποίηση (classification) των αντικειμένων και την έξοδο των Bounding Boxes. Η γραφική απεικόνιση της διαδικασίας φαίνεται στην Εικόνα 10.



Εικόνα 10. Απεικόνιση των βασικών λειτουργιών του Faster R-CNN Πηγή: https://arxiv.org/pdf/1506.01497.pdf

Πιο συγκεκριμένα, ο Faster R-CNN λειτουργεί περνώντας την εικόνα εισόδου από το CNN και παράγει τα Feature Maps τα οποία δέχεται σαν είσοδο το RPN. Το RPN εφαρμόζει τεχνική Ολίσθησης Παραθύρου (Sliding Window) πάνω από τα Feature Maps και σε κάθε παράθυρο δημιουργεί ένα συγκεκριμένο αριθμό Bounding Boxes διαφορετικών μεγεθών και αναλογιών (Εικόνα 11). Για κάθε Bounding Box, το RPN προβλέπει την πιθανότητα να υπάρχει ένα αντικείμενο μέσα σε αυτό, ανεξάρτητα από την κλάση στην οποία ενδεχομένως να ανείκει, καθώς και ένα Bounding Box Regressor για τη σωστή ρύθμιση των Bounding Boxes, έτσι ώστε να προσδιορίζουν καλύτερα το αντικείμενο. Το αποτέλεσμα αυτής της διαδικασίας είναι να έχουν δημιουργηθεί Bounding Boxes

διαφορετικών μεγεθών, τα οποία εισέρχονται στο RoI Pooling layer. Είναι πιθανό μετά το βήμα εφαρμογής του RPN, να υπάρχουν προτάσεις αντικειμένων χωρίς συγκεκριμένη κλάση. Οπότε, κάθε πρόταση αντικειμένου τροποποιείται σε μέγεθος έτσι ώστε να περιέχει ακριβώς το αντικείμενο. Αυτός είναι ο ρόλος του RoI Pooling layer, το οποίο εξαγάγει σταθερού μεγέθους Feature Maps για κάθε Bounding Box. Αυτά τα Feature Maps, στη συνέχεια περνούν από ένα πλήρως συνδεδεμένο layer, το οποίο περιλαμβάνει Softmax και Linear Regression layers για την τελική κατηγοριοποίηση και πρόβλεψη των Bounding Boxes των αντικειμένων.



Εικόνα 11. Λειτουργία sliding window του Faster R-CNN. Πηγή: [19]

Τα ελαττώματα του Faster R-CNN είναι ότι δεν "κοιτάει" ολόκληρη την εικόνα, αλλά επικεντρώνεται σε συγκεκριμένα μέρη της εικόνας διαδοχικά. Αυτό δημιουργεί την ανάγκη πολλών επαναλήψεων της διαδικασίας ανίχνευσης σε μία εικόνα για να ανιχνευθούν όλα τα αντικείμενα. Επιπλέον, δεδομένου ότι εφαρμόζονται διαδοχικά διαφορετικά συστήματα κατά την ανίχνευση, η απόδοση το ενός συστήματος περιορίζεται από την απόδοση του προηγούμενου του.

2.5 You Only Look Once (YOLO)

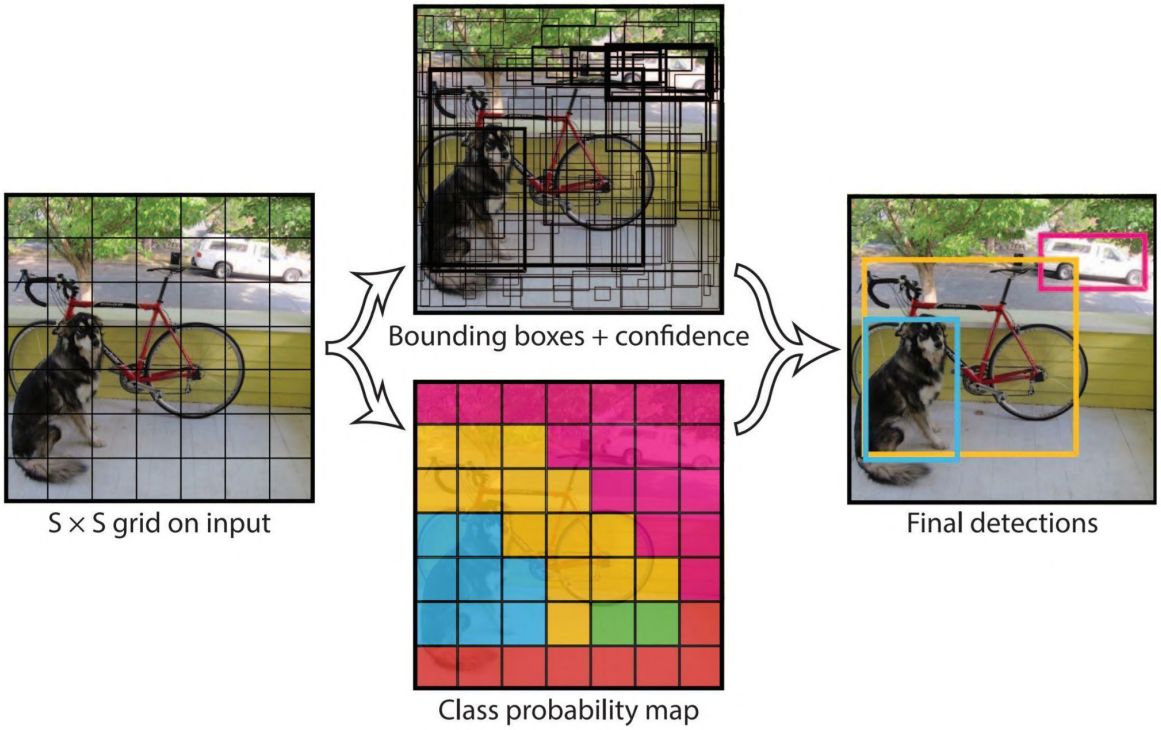
Ο συγκεκριμένος αλγόριθμος αναπτύχθηκε το 2016 [16] με σκοπό την επίτευξη ανίχνευσης αντικειμένων σε πραγματικό χρόνο. Ο YOLO αποτελεί προσέγγιση ανίχνευσης

αντικειμένων με χρήση ενός CNN δικτύου, στο οποίο με μία προσπέλαση επιτυγχάνει ανίχνευση και κατηγοριοποίηση του αντικειμένου για το οποίο το CNN έχει εκπαιδευτεί.

Ο YOLO, σε αντίθεση με άλλες προσεγγίσεις όπως ο R-CNN, δεν εφαρμόζει ταξινομητές (classifiers) σταδιακά στην εικόνα για να εντοπίσει τα χαρακτηριστικά που μπορούν να αναγνωριστούν, ούτε εφαρμόζει τοπικές προβλέψεις για την τοποθεσία των αντικειμένων, αλλά κάνει απευθείας προβλέψεις σε όλη την εικόνα για την τοποθεσία και την κλάση των αντικειμένων προς αναγνώριση. Η θέση ενός αντικειμένου ορίζεται από τις διαστάσεις και την θέση ενός κουτιού οριοθέτησης (Bounding Box), και η κλάση ενός αντικειμένου ορίζεται από το όνομα που έχει δοθεί για τα συγκεκριμένα αντικείμενα.

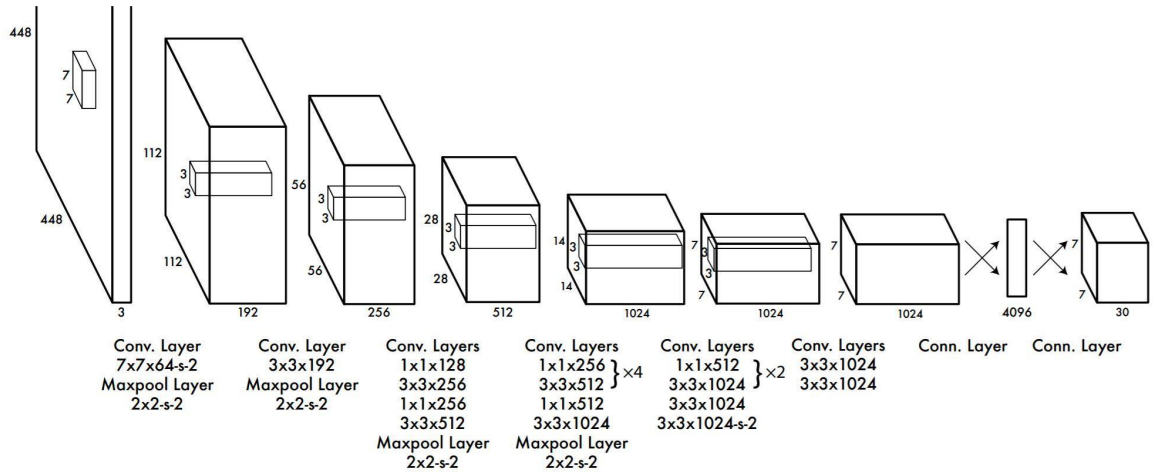
Ο αλγόριθμος χωρίζει την εικόνα σε ένα πλέγμα. Εφόσον το κέντρο ενός αντικειμένου προς αναγνώριση βρίσκεται μέσα σε ένα από τα κελιά του πλέγματος, αυτό το κελί είναι υπεύθυνο για την αναγνώριση του αντικειμένου. Κάθε κελί του πλέγματος προβλέπει έναν αριθμό από Bounding Boxes συνοδευόμενα από έναν δείκτη σιγουριάς (Confidence Value) για το κάθε ένα. Το Confidence Value εκφράζει το πόσο πιθανό είναι να υπάρχει ένα αντικείμενο μέσα στο συγκεκριμένο κελί του πλέγματος, καθώς και το πόσο ακριβής είναι η θέση του Bounding Box σε σχέση με τη θέση του αντικειμένου. Στην περίπτωση που δεν υπάρχει κάποιο αντικείμενο μέσα σε ένα κελί, το Confidence Value για αυτό το Bounding Box πρέπει να είναι μηδέν, διαφορετικά θα πρέπει να ισούται με την πιθανότητα να υπάρχει αντικείμενο στο κελί επί την ακρίβεια με την οποία έχει υπολογιστεί η θέση του Bounding Box σε σχέση με τη θέση του αντικειμένου.

Κάθε Bounding Box αποτελείται από 5 προβλέψεις: x, y, w, h, και το Confidence Value. Οι τιμές x και y αντιστοιχούν στις συντεταγμένες του κέντρου του αντικειμένου που προβλέπεται σε σχέση με τα όρια του κελιού, ενώ οι τιμές w και h αντιστοιχούν στο πλάτος και το ύψος του Bounding Box σε σχέση με όλη την εικόνα. Επιπλέον, κάθε κελί προβλέπει και έναν αριθμό κλάσεων για τα αντικείμενα που εντοπίζονται σε αυτό, με τον αριθμό των προβλεπόμενων κλάσεων να είναι συγκεκριμένος για κάθε κελί του πλέγματος, ανεξάρτητα από τον αριθμό των Bounding Boxes που έχει προβλέψει. Η πρόβλεψη των κλάσεων γίνεται με τον υπολογισμό της πιθανότητας να έχει το αντικείμενο μία συγκεκριμένη κλάση, δεδομένου ότι υπάρχει το αντικείμενο στο συγκεκριμένο Bounding Box. Αυτή η δεσμευμένη πιθανότητα, πολλαπλασιάζεται με το Confidence Value του κάθε Bounding Box και προκύπτει ένα confidence score για μία συγκεκριμένη κλάση για το κάθε κελί του πλέγματος. Αυτά τα confidence scores αντιστοιχούν στην πιθανότητα να υπάρχει η κάθε κλάση στο κάθε Bounding Box, καθώς και στο πόσο καλά έχει προβλεφθεί η θέση του αντικειμένου αυτής της κλάσης. Τα αποτελέσματα αυτών των βασικών λειτουργιών του YOLO φαίνονται στην Εικόνα 12.



Εικόνα 12. Οπτικοποίηση βασικών λειτουργιών του YOLO. Πηγή: [16]

Η εφαρμογή του προαναφερθέντος μοντέλου γίνεται με την χρήση ενός CNN, το οποίο αποτελείται από 24 convolutional layers τα οποία ακολουθούνται από 2 πλήρως συνδεδεμένα layers. Τα convolutional layers εντοπίζουν τα χαρακτηριστικά (features) μίας εικόνας, ενώ τα πλήρως συνδεδεμένα layers είναι υπεύθυνα για τις προβλέψεις των Confidence Values και των συντεταγμένων των Bounding Boxes της εξόδου. Η Εικόνα 13 απεικονίζει την αρχιτεκτονική του CNN του YOLO.



Εικόνα 13. Αρχιτεκτονική του CNN του YOLO. Πηγή: [16]

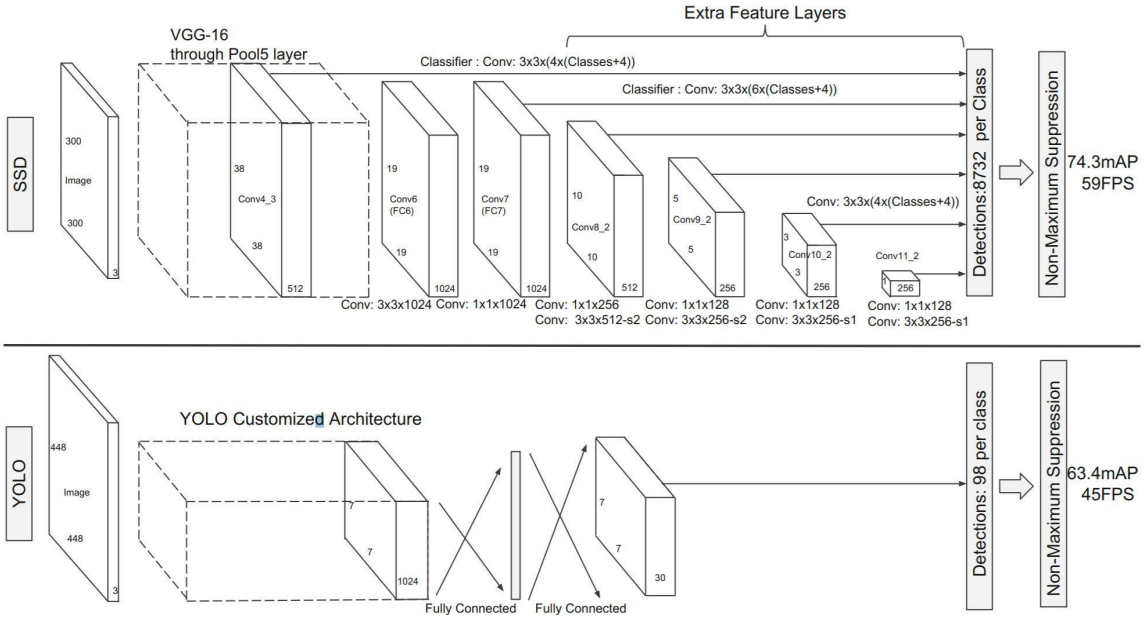
Ο YOLO περιορίζει την πρόβλεψη των Bounding Boxes, καθώς κάθε κελί του πλέγματος της εικόνας προβλέπει μόνο 2 Bounding Boxes και μπορεί να περιγράφεται από μία μόνο κλάση αντικειμένων. Αυτό περιορίζει τον αριθμό των κοντινών αντικειμένων που μπορούν να ανιχνευτούν, έτσι ο αλγόριθμος δυσκολεύεται να ανιχνεύσει πολλά ίδια αντικείμενα που βρίσκονται πολύ κοντά μεταξύ τους, όπως για παράδειγμα σμήνη πτηνών. Επιπλέον, από τη στιγμή που το συγκεκριμένο μοντέλο εκπαιδεύεται να προβλέπει Bounding Boxes από συγκεκριμένα δεδομένα, δυσκολεύεται να ανιχνεύσει παρόμοια αντικείμενα σε πολύ διαφορετική κλίμακα και αναλογία απεικόνισης. Επίσης, κατά την εκπαίδευση του ο YOLO αντιμετωπίζει τα σφάλματα στα Bounding Boxes με την ίδια βαρύτητα παρόλο που το ίδιο σφάλμα είναι πιο σημαντικό σε μικρό Bounding Box σε σχέση με ένα μεγαλύτερο Bounding Box.

Η μέθοδος YOLO βελτιώθηκε αρκετές φορές στα επόμενα έτη από τη δημιουργία του. Συγκεκριμένα, η πρώτη βελτίωση έγινε το 2017 με τον YOLO9000 [43], ο οποίος μπορεί να ανιχνεύσει 9000 διαφορετικές κατηγορίες αντικειμένων και σε ταχύτητα 40 καρέ ανά δευτερόλεπτο, η απόδοση του είναι 78.6 mAP στο Dataset Pascal Visual Object Classes (VOC 2007), ξεπερνώντας τους Faster R-CNN και SSD σε απόδοση και ταχύτητα. Το 2018, δημοσιεύτηκε ο YOLOv3 [44] ο οποίος εμφανίζει μικρές αλλαγές σε σχέση με τον YOLO9000 κυρίως όσον αφορά την ακρίβεια ανίχνευσης. Τέλος, το 2020 δημοσιεύτηκε η πιο πρόσφατη έκδοση, ο YOLOv4 [45], ο οποίος επιλύοντας τα προβλήματα των προηγούμενων εκδόσεων όπως το πρόβλημα ευαισθησίας του πλέγματος, πέτυχε τις καλύτερες επιδόσεις σε ταχύτητα και ακρίβεια από όλους τους εναλλακτικούς αλγορίθμους.

2.6 Single Shot multibox Detector (SSD)

Η μέθοδος SSD αναπτύχθηκε το 2016 με σκοπό την επίτευξη ανίχνευσης αντικειμένων με μονή προσπέλαση ενός Deep Neural Network σε πραγματικό χρόνο, δίνοντας έμφαση στην ταχύτητα ανίχνευσης [17]. Η συγκεκριμένη προσέγγιση έχει αρκετές ομοιότητες με τον YOLO καθώς ακολουθούν παρόμοια μεθοδολογία με τη διαφορά ότι ο SSD προβλέπει Bounding Boxes με διαφορετικές διαστάσεις και aspect ratios για κάθε κελί του πλέγματος και παρουσιάζει καλύτερες επιδόσεις σε ταχύτητα και ακρίβεια από την πρώτη έκδοση του YOLO. Οι βελτιώσεις του SSD ως προς τον YOLOv1 περιλαμβάνουν τη χρήση ενός Convolutional φίλτρου για την πρόβλεψη των κατηγοριών των αντικειμένων και αποκλίσεων στις θέσεις των Bounding Boxes. Το δεύτερο επιτυγχάνεται χρησιμοποιώντας ξεχωριστές προβλέψεις για διαφορετικού aspect ratio Bounding Boxes και εφαρμόζοντας αυτές τις προβλέψεις σε πολλαπλά Feature Maps από τα μετέπειτα στάδια του δικτύου για να επιτευχθεί ανίχνευση σε πολλές κλίμακες. Ειδικότερα η χρήση πολλαπλών layers για πρόβλεψη σε διαφορετικές κλίμακες οδηγεί σε ανίχνευση καλύτερης ακρίβειας χρησιμοποιώντας είσοδο σχετικά χαμηλής ανάλυσης.

Η προσέγγιση SSD βασίζεται σε ένα CNN το οποίο παράγει συγκεκριμένου μεγέθους συλλογή από Bounding Boxes, καθώς και Confidence Values για την ύπαρξη ενός αντικειμένου μιας συγκεκριμένης κατηγορίας ακολουθούμενα από ένα βήμα non-Maximum Suppression [46] για την παραγωγή των τελικών ανιχνεύσεων. Τα αρχικά layers του δικτύου βασίζονται σε τυπικές μεθόδους για την κατηγοριοποίηση εικόνων με καλή ποιότητα. Στη συνέχεια προστίθεται μία βοηθητική δομή από convolutional layers στο δίκτυο για να παραχθούν οι επιθυμητές ανιχνεύσεις. Η δομή του CNN του SSD οπτικοποιείται σε σχέση με το CNN του YOLO στην Εικόνα 14.



Εικόνα 14. Απεικόνιση των διαφορών των CNN του SSD και του YOLO. Πηγή: [17]

Ένα χαρακτηριστικό αυτής της προσέγγισης είναι η χρήση πολλών Feature Maps για την ανίχνευση. Στον SSD έχουν προστεθεί επιπλέον convolutional layers με προοδευτικά μικρότερο μέγεθος, τα οποία επιτρέπουν την ανίχνευση σε πολλές κλίμακες. Το convolutional model για την πρόβλεψη των ανιχνεύσεων είναι διαφορετικό για κάθε feature layer σε αντίθεση με τον YOLOv1. Επιπλέον, αντιστοιχίζεται μία ομάδα από Bounding Boxes με κάθε κελί του Feature Map για πολλαπλά Feature Maps στην αρχή του δικτύου. Αυτά τα Bounding Boxes καλύπτουν το Feature Map συνελικτικά έτσι ώστε η θέση του κάθε Bounding Box σε σχέση με το κελί που αντιστοιχεί είναι σταθερή. Σε κάθε κελί του Feature Map προβλέπονται αποκλίσεις των Bounding Boxes καθώς και Confidence Values ανά κλάση αντικειμένου. Επιτρέποντας διαφορετικά σχήματα για τα Bounding Boxes ο πιθανός χώρος που καταλαμβάνει το αντικείμενο εξόδου γίνεται πιο διακριτός.

Σχετικά με την εκπαίδευση του SSD, η κύρια διαφορά του με την εκπαίδευση πιο συμβατικών αλγορίθμων είναι ότι πρέπει να οριστεί συγκεκριμένη αληθής πληροφορία σε συγκεκριμένες εξόδους του δικτύου. Μόλις οριστεί αυτή η πληροφορία, εφαρμόζεται το

Backpropagation σε όλο το δίκτυο. Επιπλέον, η εκπαίδευση περιλαμβάνει την επιλογή συγκεκριμένων κλιμάκων των Bounding Boxes για την ανίχνευση.

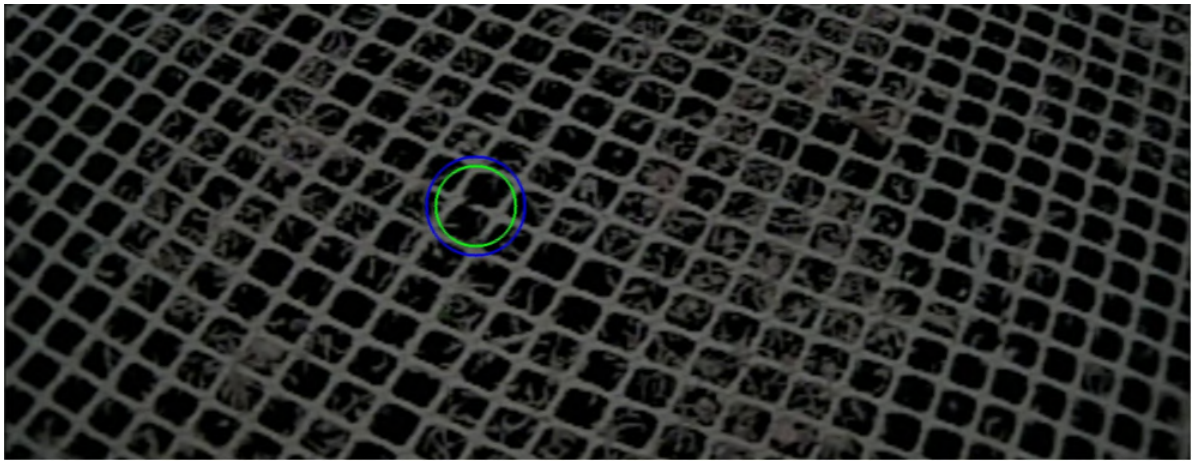
2.7 Σχετικές έρευνες ανίχνευσης ελαττωμάτων θαλάσσιας Βιορρύπανσης

2.7.1 Λογισμικό ανίχνευσης ελαττωμάτων σε Ιχθυοκαλλιέργειες

Το 2019, στο πλαίσιο πτυχιακής εργασίας, φοιτητές του Norwegian University of Science and Technology δημιούργησαν ένα λογισμικό για την ανίχνευση ελαττωμάτων με τη μορφή οπών σε δίχτυα Ιχθυοκαλλιέργειας μέσα από ροή βίντεο μιας συσκευής καθαρισμού διχτυών [47]. Στη συγκεκριμένη εργασία χρησιμοποιήθηκε το Remotely Operated Vehicle (ROV) της εταιρείας Stranda με όνομα Manta Net Cleaner (Εικόνα 15), το οποίο προσκολλημένο στα δίχτυα Ιχθυοκαλλιέργειας, τα καθαρίζει με χρήση πίδακα νερού, ενώ ο χειρισμός του γίνεται με τη βοήθεια ροής βίντεο από την κάμερα που έχει προσαρμοσμένη πάνω του. Τα βίντεο του συγκεκριμένου ROV χρησιμοποιήθηκαν για τη δημιουργία του λογισμικού ανίχνευσης ελαττωμάτων από την ερευνητική ομάδα.

Εικόνα 15. Manta net Cleaner

Για την ανίχνευση των ελαττωμάτων, χρησιμοποιήθηκε η πλατφόρμα Υπολογιστικής Όρασης OpenCV με σκοπό την επεξεργασία των καρέ του βίντεο και την ανίχνευση των ματιών του διχτυού. Με την κατάλληλη ρύθμιση των παραμέτρων επεξεργασίας του βίντεο, το λογισμικό κατάφερε να ανιχνεύσει επιτυχώς βλάβες στα δίχτυα. Συγκεκριμένα, η διαδικασία ανίχνευσης ξεκινάει με την προ-επεξεργασία των εικόνων του ROV έτσι ώστε να απομονωθούν οι καθαρές εικόνες που δείχνουν ξεκάθαρα το δίχτυ από τις θολές και δύσκολες προς ανίχνευση εικόνες. Στη συνέχεια, γίνεται η επεξεργασία των επιλεγμένων εικόνων σε διάφορα στάδια, όπως η μετατροπή σε grayscale, η εφαρμογή φίλτρου Gauss και έπειτα η μετατροπή σε binary. Με αυτό τον τρόπο γίνεται η απομόνωση του μοτίβου του διχτυού το οποίο με περαιτέρω επεξεργασία γίνεται εντονότερο και πιο λεπτό για τον ευκολότερο προσδιορισμό των ελαττωμάτων του διχτυού. Το αποτέλεσμα χρήσης του λογισμικού φαίνεται ενδεικτικά στην Εικόνα 16.



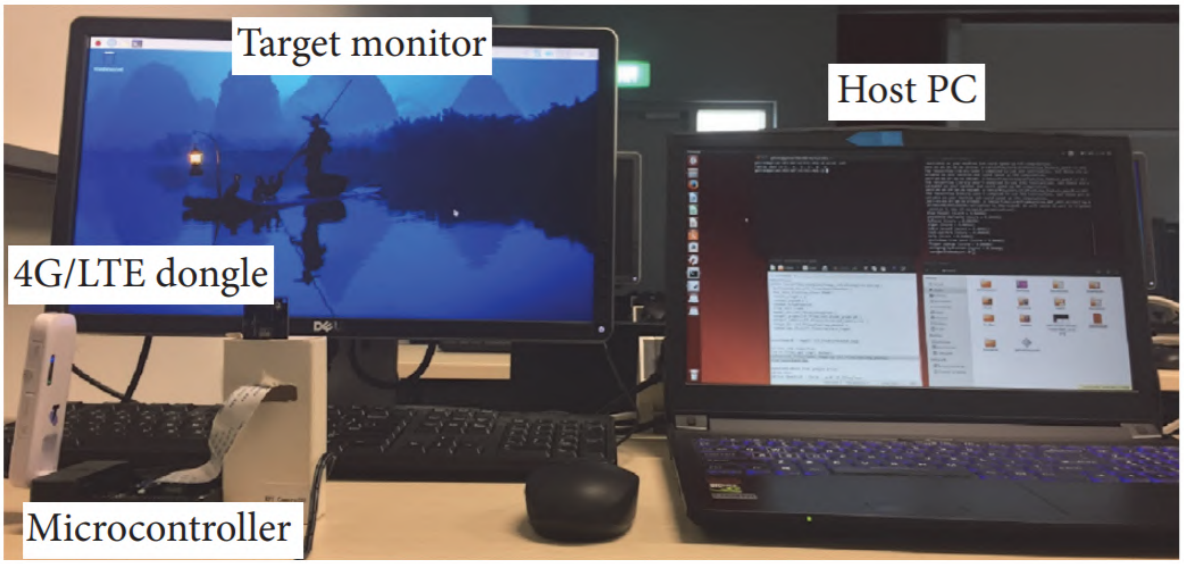
Εικόνα 16. Αποτελέσματα του λογισμικού ανίχνευσης ελαττωμάτων. Πηγή: [47]

2.7.2 Ανίχνευση ελαττωμάτων Βιορρύπανσης σε σκάφη με CNN

Το 2017 ομάδα ερευνητών πραγματοποίησε μία μελέτη με σκοπό την ανίχνευση οργανισμών Βιορρύπανσης σε επιφάνειες πλοίων μέσω Υπολογιστικής Όρασης και Νευρωνικών Δικτύων [48]. Συγκεκριμένα, χρησιμοποιήθηκε το μοντέλο Inception V3 [49] με την τεχνική transfer learning, κατά την οποία ένα δίκτυο προ-εκπαιδευμένο με ένα μεγάλο Dataset (στη συγκεκριμένη περίπτωση το ImageNet) χρησιμοποιείται για περαιτέρω εκπαίδευση με μικρό αριθμό δειγμάτων προκειμένου να εκπαιδευτεί για το συγκεκριμένο σκοπό χωρίς να απαιτείται μεγάλος αριθμός εικόνων στο Dataset.

Το σύστημα της συγκεκριμένης έρευνας αποτελούνταν από έναν Single Board Computer και συγκεκριμένα το Raspberry Pi το οποίο φωτογράφιζε τα σκάφη. Έπειτα, οι παραγόμενες φωτογραφίες αποστέλλονταν σε έναν κεντρικό υπολογιστή ο οποίος

χρησιμοποιούσε το εκπαιδευμένο CNN για να διαπιστώσει αν και ποιοι οργανισμοί βρίσκονται προσκολλημένοι στην επιφάνεια του πλοίου (Εικόνα 17).



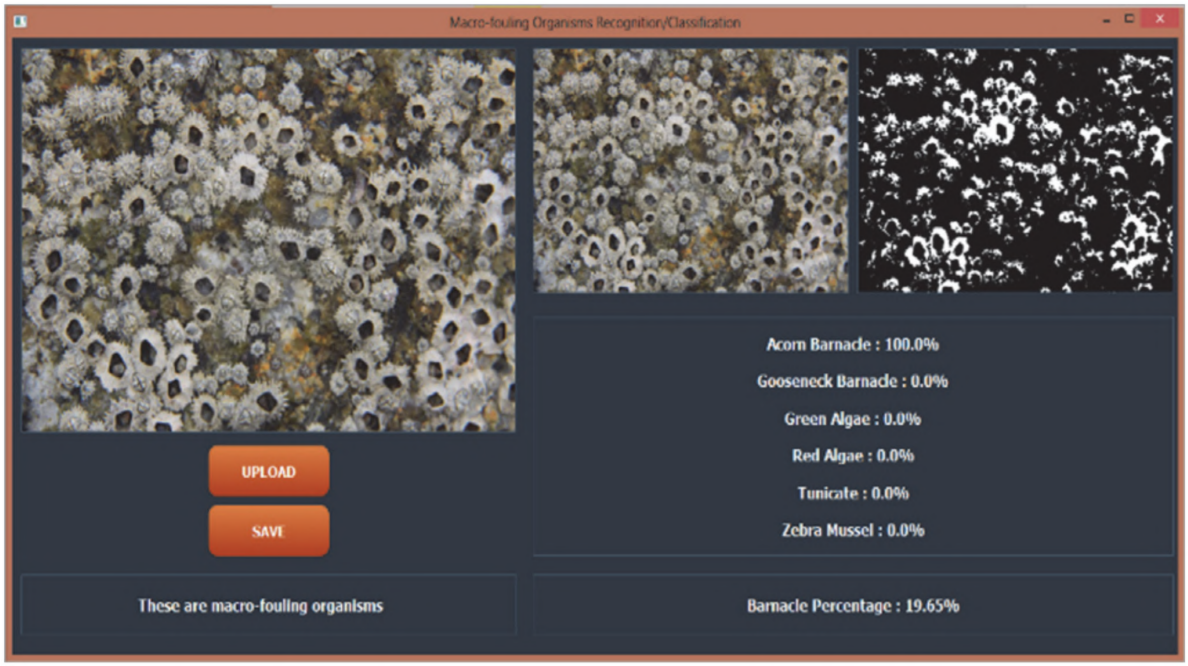
Εικόνα 17. Σύστημα ανίχνευσης οργανισμών Βιορρύπανσης. Πηγή: [48]

Συγκεκριμένα, Το CNN εκπαιδεύτηκε για να αναγνωρίζει 10 από τα είδη οργανισμών που αναπτύσσονται κατά το φαινόμενο της Βιορρύπανσης. Για την εκπαίδευση χρησιμοποιήθηκε το εργαλείο Tensorflow και ένα Dataset από 1825 εικόνες αυτών των οργανισμών από τις οποίες ένα δείγμα φαίνεται στην Εικόνα 18.

Εικόνα 18. Είδη οργανισμών Βιορρύπανσης. Πηγή: [48]

Η διαδικασία ανίχνευσης των οργανισμών Βιορρύπανσης στη συγκεκριμένη έρευνα γίνεται σε ξεχωριστά στάδια. Αρχικά γίνεται ο έλεγχος για την ύπαρξη οργανισμών σε μία φωτογραφία. Αν δεν βρεθούν οργανισμοί, η διαδικασία σταματάει και υπολογίζεται το ποσοστό Βιορρύπανσης της επιφάνειας που εξετάζεται. Διαφορετικά, η διαδικασία συνεχίζεται με την κατηγοριοποίηση των επιμέρους οργανισμών που εντοπίστηκαν στη φωτογραφία. Επιπλέον, στα αρχικά στάδια γίνεται η επεξεργασία των εικόνων έτσι ώστε οι οργανισμοί που απεικονίζονται να έχουν πιο αντιπροσωπευτική μορφή. Στο τελικό στάδιο της ανίχνευσης, χρησιμοποιείται η πλατφόρμα OpenCV για τον υπολογισμό του ποσοστού Βιορρύπανσης στη συνολική επιφάνεια που εξετάζεται.

Επιπλέον, τα στάδια της διαδικασίας ανίχνευσης των οργανισμών Βιορρύπανσης οπτικοποιήθηκαν μέσω ενός λογισμικού με γραφική διεπαφή που αναπτύχθηκε για τη συγκεκριμένη λειτουργία (Εικόνα 19).

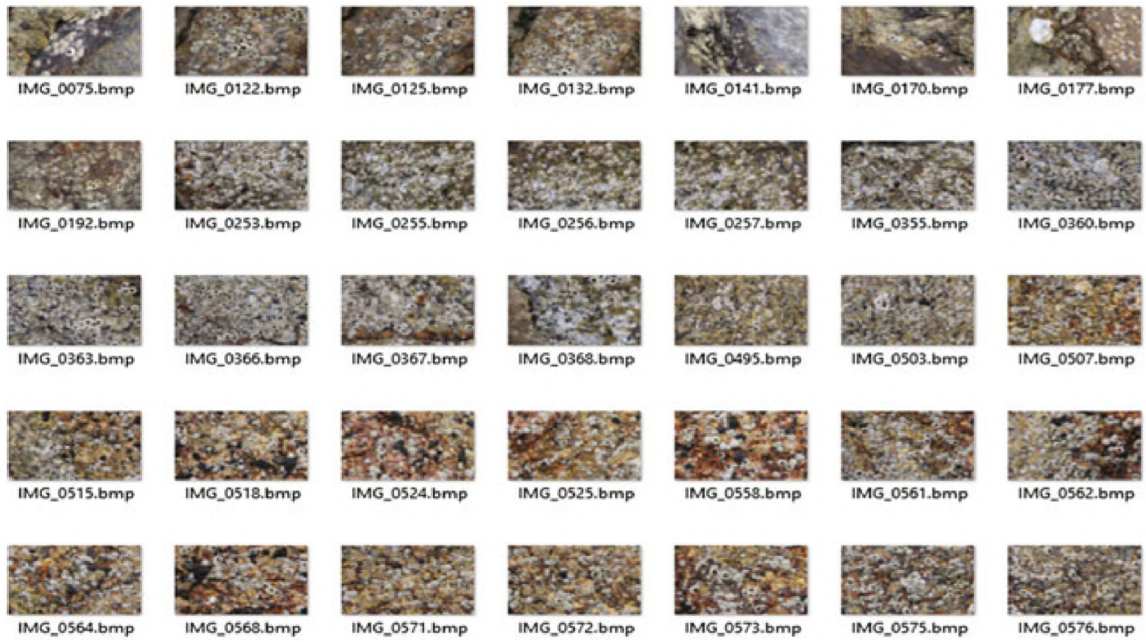


Εικόνα 19. Γραφική διεπαφή ανίχνευσης οργανισμών. Πηγή: [48]

2.7.3 Ανίχνευση ελαττωμάτων Βιορρύπανσης σε σκάφη με Haar Cascades

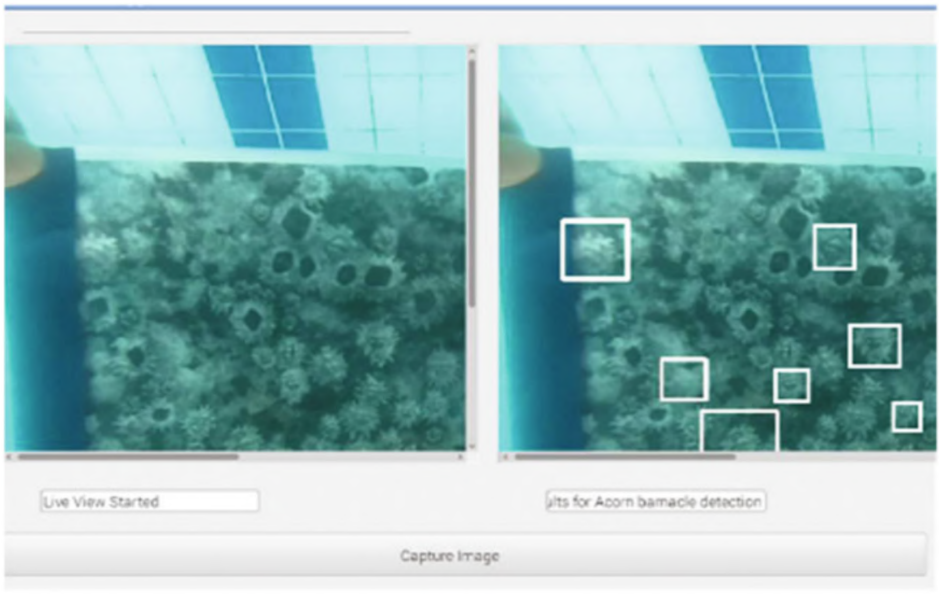
Στην έκδοση του βιβλίου *Advances in Computer Communication and Computational Sciences Proceedings of IC4S 2017, Volume 2* που δημοσιεύτηκε το 2018 περιλαμβάνεται ένα κεφάλαιο στο οποίο παρουσιάζεται μία έρευνα ανίχνευσης οργανισμών Βιορρύπανσης σε πλοία με χρήση Υπολογιστικής Όρασης [50].

Η κύρια διαφορά της συγκεκριμένης έρευνας με αυτή που παρουσιάστηκε στο υποκεφάλαιο *2.7.2* είναι ότι χρησιμοποιήθηκε η τεχνική των Haar Cascades μέσω της πλατφόρμας OpenCV χωρίς να γίνει η χρήση κάποιου εξειδικευμένου CNN για την ανίχνευση των οργανισμών. Η εκπαίδευση του cascade γίνεται με την χρήση "θετικών" εικόνων οι οποίες περιέχουν το αντικείμενο προς αναγνώριση και "αρνητικών" εικόνων οι οποίες δεν περιέχουν αντικείμενα προς αναγνώριση. Συνήθως οι αρνητικές εικόνες είναι το background της εικόνας στο πραγματικό σενάριο χρήσης του cascade. Με αυτό τον τρόπο, το cascade μαθαίνει να αναγνωρίζει τα μοτίβα των θετικών εικόνων και εφαρμόζει αυτή τη γνώση σε νέες εικόνες προκειμένου να αναγνωριστούν τα συγκεκριμένα μοτίβα. Σε αντίθεση με την μέθοδο του υποκεφαλαίου *2.7.2*, η συγκεκριμένη τεχνική δεν μπορεί να αναγνωρίσει διαφορετικές κλάσεις αντικειμένων, επομένως η αναζήτηση έγινε στους οργανισμούς του είδους Barnacles (Εικόνα 20).



Εικόνα 20. Dataset οργανισμών Barnacles. Πηγή: [50]

Το σύστημα της συγκεκριμένης εργασίας αποτελούνταν από ένα Raspberry Pi το οποίο, μέσα από ένα υδατοστεγές κουτί, έκανε τη λήψη και επεξεργασία φωτογραφιών του σκάφους καθώς και την εφαρμογή του Haar Cascade μέσω του OpenCV. Στη συνέχεια, οι τελικές εικόνες αποστέλλονταν σε μία Web διεπαφή που αναπτύχθηκε για την παρατήρηση των αποτελεσμάτων (Εικόνα 21). Τα αποτελέσματα της συγκεκριμένης έρευνας ήταν να ανιχνευθούν οι οργανισμοί Βιορρύπανσης με μέσο ποσοστό επιτυχίας 50.51%.



Εικόνα 21. Αποτελέσματα ανίχνευσης οργανισμών μέσω Haar Cascade. Πηγή: [50]

3 *Εφαρμογή Αλγορίθμων Ανίχνευσης αντικειμένων σε δίχτυα Ιχθυοκαλλιέργειας*

3.1 Εισαγωγή

Στη συγκεκριμένη διπλωματική εργασία επιλέχθηκαν οι αλγόριθμοι Faster R-CNN, SSD και YOLOv3 για τη δοκιμή τους στην ανίχνευση ελαττωμάτων σε δίχτυα Ιχθυοκαλλιέργειας. Τα κριτήρια επιλογής τους ήταν η ταχύτητα αναγνώρισης σε συνδυασμό με τη μέση ακρίβειά τους, καθώς και η μεθοδολογία που εφαρμόζουν. Συγκεκριμένα, οι YOLO και SSD είναι δύο από τους πιο γρήγορους αλγόριθμους ανίχνευσης και χρησιμοποιούν one-stage μεθοδολογία, ενώ ο Faster R-CNN αποτελεί τον πιο γρήγορο two-stage αλγόριθμο με σχετικά μεγάλο mAP 73.2% στο Pascal VOC 2007. Οι αλγόριθμοι YOLOv3 και SSD επιτυγχάνουν mAP 80% και 74.3% αντίστοιχα στο Pascal VOC 2007.

Αρχικά, δημιουργήθηκε ένα Dataset με εικόνες από δίχτυα Ιχθυοκαλλιέργειας που παρουσιάζουν ελαττώματα λόγω της Βιορρύπανσης. Οι εικόνες αυτές έπρεπε να κατηγοριοποιηθούν με κατάλληλο τρόπο προκειμένου να χρησιμοποιηθούν στην εκπαίδευση των Νευρωνικών Δικτύων. Στη συνέχεια χρησιμοποιήθηκε το Dataset για την εκπαίδευση του κάθε Δικτύου ξεχωριστά. Κατά τη διάρκεια της εκπαίδευσης, κάποιες από τις εικόνες του Dataset χρησιμοποιήθηκαν για testing προκειμένου να γίνει η καταγραφή της προόδου της εκπαίδευσης και ο σωστός υπολογισμός των επιδόσεων του κάθε αλγορίθμου. Τέλος, τα εκπαιδευμένα δίκτυα χρησιμοποιήθηκαν σε βίντεο από τα δίχτυα Ιχθυοκαλλιέργειας για να φανεί ενδεικτικά ποια θα είναι η συμπεριφορά τους σε πραγματικές συνθήκες σε περίπτωση που εφαρμοστούν σε ένα AUV.

3.2 Εργαλεία και περιβάλλον εκπαίδευσης

Στο συγκεκριμένο κεφάλαιο περιγράφονται τα εργαλεία που χρησιμοποιήθηκαν για την εκπαίδευση των αλγορίθμων, τη δημιουργία των μοντέλων, την εφαρμογή τους στην πειραματική διαδικασία, καθώς και την αποτίμηση και σύγκριση των αποτελεσμάτων τους.

3.2.1 TensorFlow

Οι αλγόριθμοι SSD και Faster R-CNN υποστηρίζονται από το TensorFlow, που αποτελεί μία end-to-end πλατφόρμα ανάπτυξης μοντέλων Μηχανικής Μάθησης . Παρέχει πληθώρα, εργαλείων, βιβλιοθηκών καθώς και community support κάνοντας πιο προσιτή την ανάπτυξη των μοντέλων.

Η έκδοση που χρησιμοποιήθηκε για την εκπαίδευση των μοντέλων είναι η 1.15 η οποία συνδυάζεται με την έκδοση Cuda 10.1 για την αξιοποίηση της κάρτας γραφικών του τοπικού υπολογιστή. Τα βήματα της εγκατάστασης του TensorFlow και των απαραίτητων για τη λειτουργία του εργαλείων ακολουθήθηκαν σε εικονικό περιβάλλον python και είχαν τη μορφή των εντολών που απεικονίζονται στον Πίνακα 2.

```
pip install tensorflow=1.15
pip install tensorflow-gpu=1.15
pip install lxml
pip install pillow
pip install matplotlib
pip install jupyter
pip install contextlib2
pip install cython
pip install tf_slim
```

Πίνακας 2. Εγκατάσταση των απαραίτητων εργαλείων για την εκπαίδευση σε Tensorflow

3.2.2 Anaconda

Για να αποφευχθούν τυχόν λάθη και ασυμβατότητες μεταξύ πακέτων εγκατάστασης, βιβλιοθηκών και dependencies, η εγκατάσταση και η παραμετροποίηση τους έγινε σε εικονικά περιβάλλοντα που δημιουργήθηκαν με το εργαλείο Anaconda, το οποίο αποτελεί βασικό εργαλείο για Data Science.

Το Anaconda είναι μία πλατφόρμα η οποία υποστηρίζει τη δημιουργία πολλών εικονικών περιβαλλόντων για Data Science και Machine Learning. Μέσω του Anaconda μπορεί κανείς να δημιουργήσει και να εκπαιδεύσει μοντέλα μηχανικής μάθησης χρησιμοποιώντας

πληθώρα πακέτων Python που δημιουργήθηκαν από το Open source community, συμπεριλαμβανομένων των TensorFlow και PyTorch. Η δωρεάν έκδοση του Anaconda παρέχει όλα τα εργαλεία που χρειάστηκαν στη συγκεκριμένη εργασία.

Η δημιουργία εικονικού περιβάλλοντος γίνεται με την εντολή που φαίνεται στην Εικόνα 22. Στη συγκεκριμένη περίπτωση φαίνεται η δημιουργία του περιβάλλοντος με τίτλο "ssd-env" το οποίο θα χρησιμοποιεί την έκδοση python 3.6. Με αυτό τον τρόπο, δημιουργήθηκε ένα εικονικό περιβάλλον για κάθε μοντέλο που αναπτύχθηκε και μέσα σε αυτά τα περιβάλλοντα εγκαταστάθηκαν οι σωστές εκδόσεις των πακέτων που απαιτούνταν για την εκπαίδευση του κάθε μοντέλου.

```
(base) C:\Users\Frapais>conda create -n ssd-env python=3.6 anaconda
Collecting package metadata (current_repodata.json): done
Solving environment: failed with repodata from current_repodata.json, will retry with next repodata source.
Collecting package metadata (repodata.json): done
Solving environment: done
```

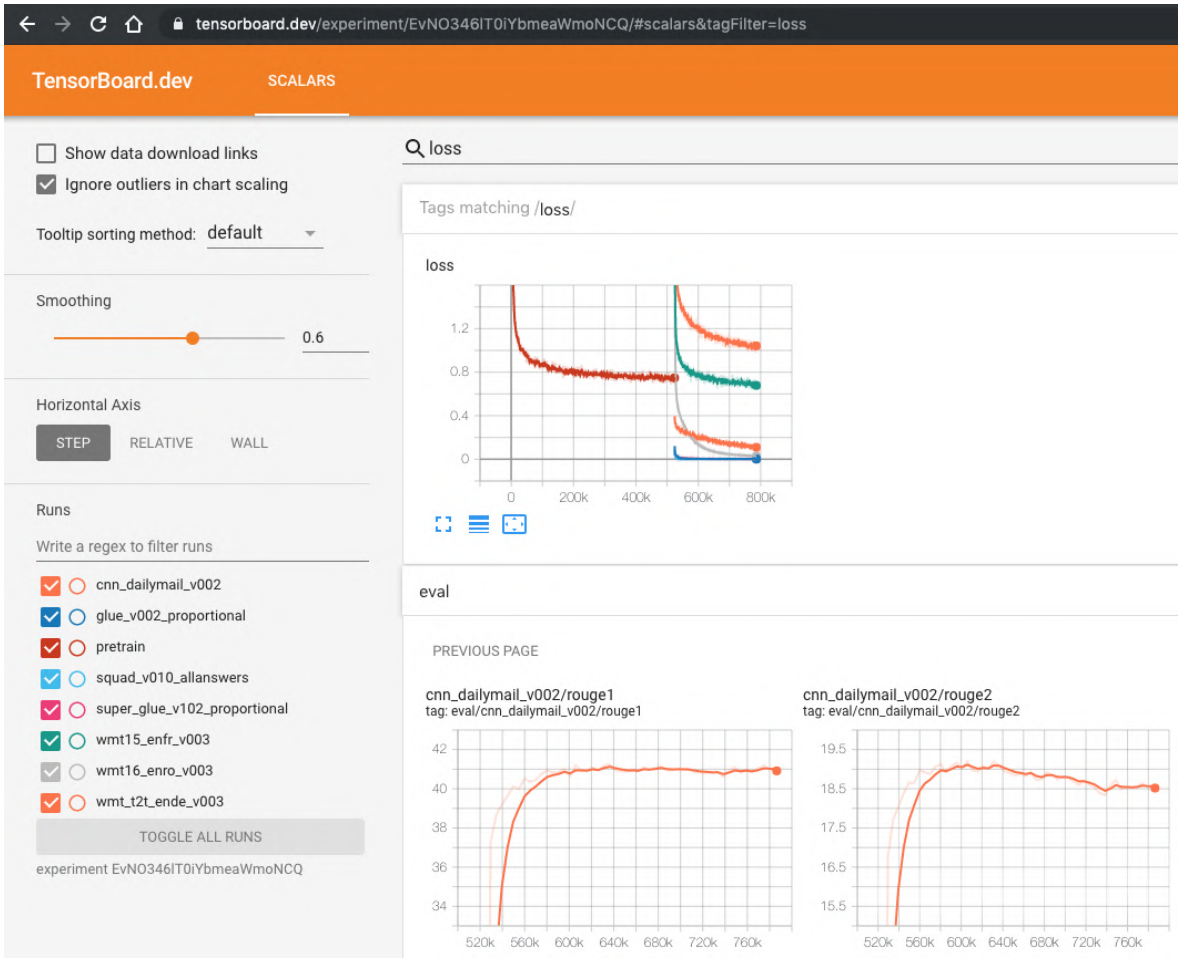
Εικόνα 22. Δημιουργία εικονικού περιβάλλοντος στο Anaconda

3.2.3 Tensorboard για την εξαγωγή αποτελεσμάτων

Όπως αναφέρθηκε στην εισαγωγή (υποκεφάλαιο 3.1), αρχικά τα Νευρωνικά Δίκτυα των αλγορίθμων έπρεπε να εκπαιδευτούν χρησιμοποιώντας το Dataset που δημιουργήθηκε και στη συνέχεια να δοκιμαστούν για να συγκριθούν τα αποτελέσματα τους. Η διαδικασία εκπαίδευσης των Νευρωνικών Δικτύων επιτρέπει τον έλεγχο της προόδου της εκπαίδευσης και της απόδοσης των δικτύων ενώ αυτά εκπαιδεύονται. Κατά την εκπαίδευση των δικτύων, δημιουργούνται περιοδικά ορισμένα αρχεία τα οποία περιέχουν τα δεδομένα της εκπαίδευσης τη χρονική στιγμή που δημιουργούνται.

Επιπλέον, μετά την εκπαίδευση τους, χρησιμοποιούνται ορισμένες εικόνες του Dataset που δεν χρησιμοποιήθηκαν στην εκπαίδευση για το λεγόμενο evaluation του εκπαιδευμένου δικτύου. Αυτή η διαδικασία είναι παρόμοια με τη διαδικασία της εκπαίδευσης, με τη διαφορά ότι αντί να βελτιώνει το μοντέλο σε κάθε προσπέλαση, συγκρίνει τα αποτελέσματα με τα επιθυμητά και παράγει όλες τις μετρήσεις απόδοσης του δικτύου.

Ένα πολύ δημοφιλές εργαλείο για την εξαγωγή και την επεξεργασία των δεδομένων της εκπαίδευσης είναι το TensorBoard (Εικόνα 23). Το Tensorboard αποτελεί toolkit του Tensorflow το οποίο στη συγκεκριμένη εργασία χρησιμοποιήθηκε για την εκπαίδευση των SSD Faster R-CNN και YOLOv3. Εξειδικεύεται στην παροχή των μετρήσεων και των οπτικοποιήσεων που απαιτούνται σε πειράματα Μηχανικής Μάθησης. Επιτρέπει την παρακολούθηση μετρήσεων πειράματος όπως απώλεια και ακρίβεια, οπτικοποίηση του γραφήματος μοντέλου και πολλά άλλα.



Εικόνα 23. Το περιβάλλον του Tensorboard. Πηγή: https://www.tensorflow.org/tensorboard

Η εγκατάσταση του γίνεται αυτόματα με την εγκατάσταση οποιασδήποτε έκδοσης του Tensorflow και με τη χρήση της εντολής "tensorboard --logdir=.", γίνεται εξαγωγή των δεδομένων εκπαίδευσης και οπτικοποίηση τους μέσω μιας τοπικής σελίδας στο πρόγραμμα περιήγησης.

Αντιγράφοντας τη διεύθυνση που εμφανίζεται στο αποτέλεσμα της εντολής σε ένα πρόγραμμα περιήγησης, παρέχεται πρόσβαση στα δεδομένα που εξήγαγε το Tensorboard (Εικόνα 24).

```
(fasterRcnn-env) C:\Users\Frapais\Object Detection\models\research\object_detection\training\logs>tensorboard --logdir=.

2021-10-21 22:56:25.101288: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic
library cudart64_100.dll
TensorBoard 1.15.0 at http://DESKTOP-5NNJF2L:6006/ (Press CTRL+C to quit)
```

Εικόνα 24. Εκκίνηση του Tensorboard και εξαγωγή των αποτελεσμάτων σε τοπική σελίδα.

Το Tensorboard μπορεί να διαχειριστεί αρχεία τύπου .tfevents τα οποία δημιουργούνται αυτόματα ανά τακτά χρονικά διαστήματα κατά την εκπαίδευση και το

evaluation των δικτύων. Οπτικοποιώντας τα αρχεία εκπαίδευσης προκύπτουν διάφορα γραφήματα όπως αυτά των σφαλμάτων ως προς τον χρόνο εκπαίδευσης τα οποία πρέπει να δείχνουν μία ξεκάθαρη πτώση αν το δίκτυο εκπαιδεύτηκε σωστά. Οπτικοποιώντας τα αρχεία που προκύπτουν από το evaluation του δικτύου, προκύπτουν οι τιμές mAP, mAR καθώς και οι εικόνες με τα εντοπισμένα αντικείμενα που χρησιμοποιήθηκαν στο evaluation.

3.3 Δημιουργία του Dataset

Dataset είναι ένα σύνολο εικόνων που χρησιμοποιείται για την εκπαίδευση Νευρωνικών Δικτύων. Πιο συγκεκριμένα, κάθε εικόνα πρέπει να συνοδεύεται από ένα αρχείο με πληροφορίες σχετικά με το περιεχόμενο της κάθε εικόνας. Στη συγκεκριμένη εργασία στόχος είναι η ανίχνευση ελαττωμάτων σε εικόνες διχτυών, επομένως κάθε εικόνα του Dataset πρέπει να προέρχεται από ένα περιβάλλον με συνθήκες παρόμοιες με αυτές στα δίχτυα Ιχθυοκαλλιέργειας και να περιέχει τουλάχιστον 1 εμφανές ελάττωμα. Το αρχείο που συνοδεύει κάθε εικόνα θα πρέπει να περιέχει τη θέση, τον αριθμό καθώς και τις κλάσεις των αντικειμένων προς αναγνώριση που απεικονίζονται. Η δομή και ο τύπος του αρχείου διαφέρει μεταξύ των αλγορίθμων ανίχνευσης αντικειμένων. Συγκεκριμένα, ο YOLO φτιάχτηκε για το Darknet Framework το οποίο χρησιμοποιεί αρχεία σε μορφή .txt και τα αντικείμενα χαρακτηρίζονται από τις συντεταγμένες του κέντρου του κάθε Bounding Box καθώς και το μήκος και το πλάτος του ενώ οι κλάσεις των αντικειμένων καταγράφονται με έναν ακέραιο αριθμό. Οι SSD και Faster R-CNN υποστηρίζονται από το Tensorflow το οποίο χρησιμοποιεί αρχεία της μορφής .xml που περιλαμβάνουν τη θέση των Bounding Boxes με τις συντεταγμένες των ακμών τους ενώ οι κλάσεις των αντικειμένων καταγράφονται με τον τίτλο τους. Στη συγκεκριμένη εργασία χρησιμοποιήθηκε η πλατφόρμα Tensorflow για την εκπαίδευση και το evaluation των αλγορίθμων, επομένως το Labeling των εικόνων έγινε στη μορφή .xml ενώ για την εκπαίδευση του YOLO απαιτήθηκε μία διαφορετική διαδικασία προκειμένου να μπορέσει να εκπαιδευτεί μέσω του Tensorflow.

3.3.1 Δημιουργία των εικόνων

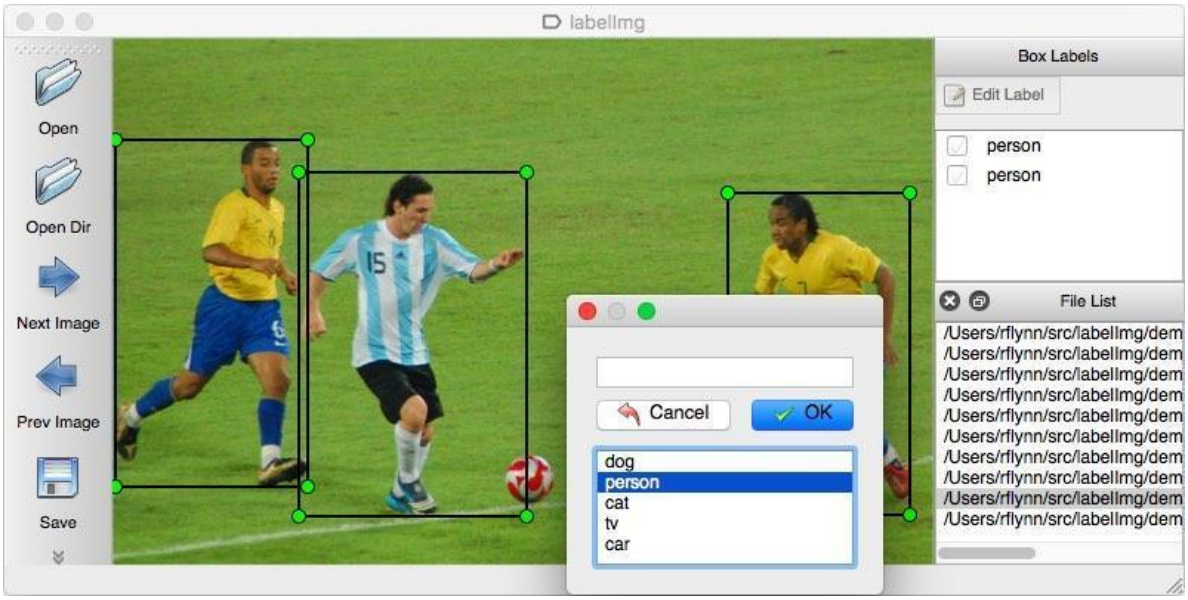
Για τη δημιουργία του Dataset χρησιμοποιήθηκαν βίντεο από δίχτυα Ιχθυοκαλλιέργειας η λήψη των οποίων έγινε από δύτη με υποβρύχια κάμερα χειρός. Για να παραχθούν εικόνες από τα βίντεο, χρησιμοποιήθηκε το λογισμικό ανοιχτού κώδικα VLC media player. Με το συγκεκριμένο πρόγραμμα, κάθε καρέ του κάθε βίντεο χωρίστηκε σε ξεχωριστή jpeg εικόνα με αποτέλεσμα να δημιουργηθούν χιλιάδες εικόνες συνολικά από όλα τα καρέ των βίντεο (Εικόνα 25). Πολλές από τις εικόνες που δημιουργήθηκαν είτε δεν απεικόνιζαν τα δίχτυα, είτε ήταν θολές λόγω των απότομων κινήσεων του βίντεο, είτε δεν απεικόνιζαν κάποιο εμφανές ελάττωμα στα δίχτυα το οποίο μπορούσε να προσδιοριστεί. Επομένως, οι εικόνες χωρίστηκαν χειροκίνητα έτσι ώστε να παραμείνουν στο Dataset μόνο

οι εικόνες που απεικονίζουν ξεκάθαρα ελαττώματα στα δίχτυα. Από τις 378 εικόνες συνολικά, οι 341 χρησιμοποιήθηκαν στην εκπαίδευση των δικτύων ενώ οι 37 χρησιμοποιήθηκαν μόνο στο evaluation των δικτύων.

Εικόνα 25. Παράδειγμα εικόνας του Dataset

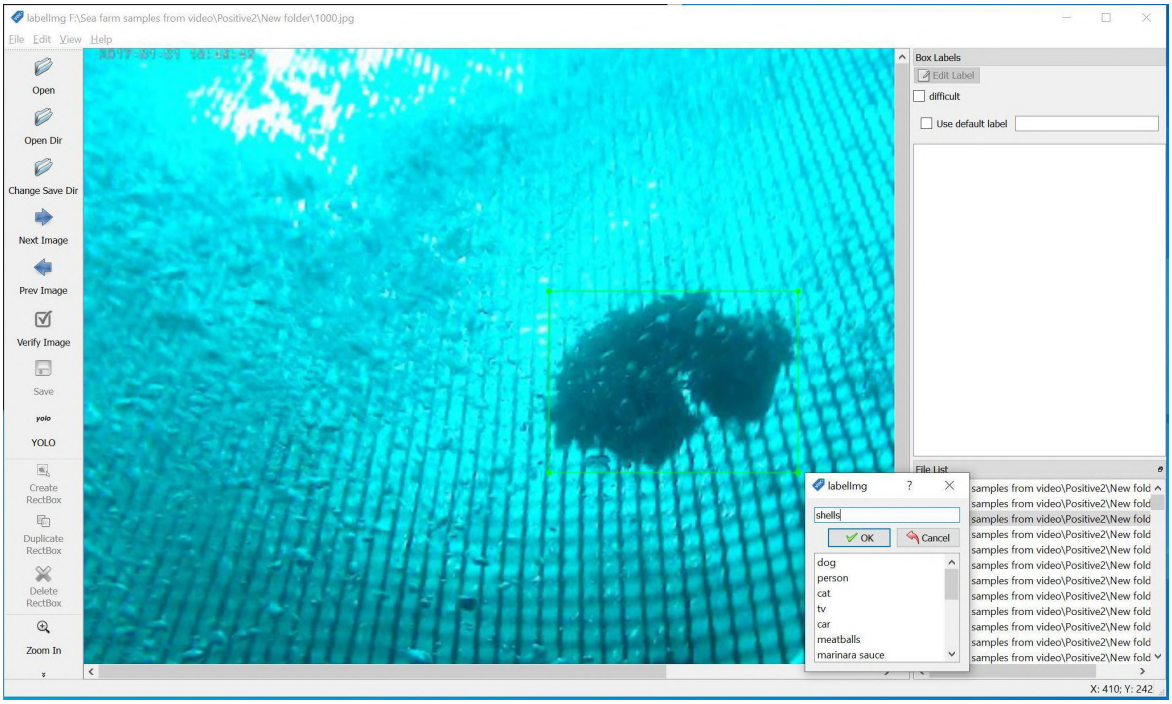
3.3.2 Δημιουργία των labels

Το επόμενο βήμα στη δημιουργία του Dataset είναι η κατηγοριοποίηση των εικόνων (Labeling). Όπως αναφέρθηκε στην εισαγωγή, η εκπαίδευση των Νευρωνικών Δικτύων γίνεται με τη μέθοδο Back Propagation όπου γίνεται προσπέλαση του δικτύου με είσοδο μία εικόνας γνωστού επιθυμητού αποτελέσματος, έπειτα γίνεται η σύγκριση του αποτελέσματος με του επιθυμητού και η αντίστροφη προσπέλαση του δικτύου ρυθμίζοντας τα βάρη των άκρων που συνδέουν του νευρώνες έτσι ώστε να μειωθεί το σφάλμα της τελικής ανίχνευσης. Τα αρχεία που συνοδεύουν τις εικόνες του Dataset παρέχουν ακριβώς αυτές τις πληροφορίες που χρειάζονται για τη σωστή σύγκριση του αποτελέσματος της προσπέλασης του δικτύου κατά την εκπαίδευση του Νευρωνικού Δικτύου. Για τη δημιουργία αυτών των αρχείων χρησιμοποιήθηκε το λογισμικό LabelImg (Εικόνα 26). Το LabelImg είναι ένα εργαλείο για την κατηγοριοποίηση και το annotation σε εικόνες. Είναι λογισμικό ανοιχτού κώδικα το οποίο δημοσιεύτηκε στο GitHub από τον χρήστη TzuTan και είναι γραμμένο σε γλώσσα Python. Το LabelImg επιτρέπει τον χειροκίνητο προσδιορισμό Bounding Boxes για κάθε αντικείμενο που απεικονίζεται και την αυτόματη δημιουργία του αρχείου σε μορφή .txt για το Darknet Framework του YOLO ή σε μορφή .xml με τη δομή που χρησιμοποιείται στο Pascal VOC.



Εικόνα 26. Το λογισμικό LabelImg. Πηγή: https://github.com/tzutalin/labelImg

Με αυτό τον τρόπο έγινε η κατηγοριοποίηση και των 378 εικόνων θέτοντας χειροκίνητα ένα περίγραμμα γύρω από κάθε εμφανές ελάττωμα σε κάθε εικόνα και δίνοντας του την κλάση με όνομα "shells". Η επιλογή του ονόματος έγινε λόγω της φύσης των ελαττωμάτων καθώς σχεδόν αποκλειστικά αποτελούνταν από οστρακοειδή που είχαν προσκολληθεί στα δίχτυα. Στην Εικόνα 27 φαίνεται η οριοθέτηση ενός ελαττώματος σε μία εικόνα μέσω του LabelImg.



Εικόνα 27. Χρήση του LabelImg στη δημιουργία του Dataset της παρούσας εργασίας

Το αποτέλεσμα ήταν να δημιουργηθεί ένα Dataset 378 εικόνων με Labeling σε μορφή .xml. Μετά την εκπαίδευση των Νευρωνικών Δικτύων, χρησιμοποιούνται ορισμένες εικόνες του Dataset για τη μέτρηση των επιδόσεων του δικτύου. Στη συγκεκριμένη περίπτωση, το 10% των εικόνων των Datasets δεσμεύτηκαν για να χρησιμοποιηθούν στο evaluation, επομένως δεν χρησιμοποιήθηκαν για την εκπαίδευση των Δικτύων. Έτσι η εκπαίδευση των Νευρωνικών δικτύων έγινε με 341 εικόνες και το evaluation τους έγινε με 37 τυχαία επιλεγμένες εικόνες. Για λόγους συνέπειας των αποτελεσμάτων, οι 37 εικόνες που επιλέχθηκαν για το evaluation, ήταν οι ίδιες για όλους τους αλγορίθμους.

Το απόκομμα κώδικα στον Πίνακα 3 είναι ένα αρχείο .xml που αντιστοιχεί στην εικόνα 246.jpg του Dataset. Διακρίνονται δύο περιοχές <object> που σημαίνει ότι στη συγκεκριμένη εικόνα υπάρχουν 2 ελαττώματα στα δίχτυα. Σε κάθε <object> αναγράφεται η κλάση <name> στην οποία ανήκει το αντικείμενο καθώς και το Bounding Box <bndbox> στο οποίο βρίσκεται.

```
<annotation>
        <folder>New folder (2)</folder>
        <filename>246.jpg</filename>
        <path>F:\Sea farm samples from video\Positive2\New folder (2)\246.jpg</path>
        <source>
                <database>Unknown</database>
        </source>
        <size>
                <width>640</width>
                <height>360</height>
                <depth>3</depth>
        </size>
        <segmented>0</segmented>
        <object>
                <name>shells</name>
                <pose>Unspecified</pose>
                <truncated>1</truncated>
                <difficult>0</difficult>
                <bndbox>
                        <xmin>432</xmin>
                        <ymin>242</ymin>
                        <xmax>542</xmax>
                        <ymax>360</ymax>
                </bndbox>
        </object>
        <object>
                <name>shells</name>
                <pose>Unspecified</pose>
                <truncated>1</truncated>
                <difficult>0</difficult>
                <bndbox>
                        <xmin>445</xmin>
                        <ymin>1</ymin>
                        <xmax>551</xmax>
                        <ymax>133</ymax>
                </bndbox>
        </object>
</annotation>
```

Πίνακας 3. Παράδειγμα annotation αρχείου μιας εικόνας του Dataset

3.3.3 Μετατροπή .xml σε .csv

Για την πιο εύκολη διαχείριση των annotations στην εκπαίδευση και των τριών αλγορίθμων, τα δεδομένα όλων των αρχείων .xml συγκεντρώθηκαν σε ένα .csv αρχείο όπου κάθε στήλη αντιστοιχεί σε μία συγκεκριμένη παράμετρο του κάθε xml αρχείου.

Στην Εικόνα 28 φαίνεται ένα μέρος του csv αρχείου που χρησιμοποιήθηκε στην εκπαίδευση. Κάθε γραμμή είναι ένα ξεχωριστό αντικείμενο προς αναγνώριση, ενώ κάθε στήλη περιέχει ένα συγκεκριμένο χαρακτηριστικό που έχουν όλα τα αντικείμενα όπως διαστάσεις και συντεταγμένες του Bounding Box. Στις περιπτώσεις όπου μια εικόνα έχει

πάνω από ένα αντικείμενο προς αναγνώριση, κάθε αντικείμενο καταγράφεται σε ξεχωριστή γραμμή του csv αρχείου όπως φαίνεται για την εικόνα 1003.jpg.

	A	B	C	D	E	F	G	H
1	ImageID	width	height	LabelNam	XMin	YMin	XMax	YMax
2	1000.jpg	640	360	shells	266	137	410	241
3	1003.jpg	640	360	shells	142	7	230	106
4	1003.jpg	640	360	shells	185	127	238	193
5	1005.jpg	640	360	shells	46	33	331	299

Εικόνα 28. Απόκομμα του csv αρχείου της εκπαίδευσης

3.3.4 Δημιουργία του labelmap

Κατά την εκπαίδευση των δικτύων, χρειάζεται ένα αρχείο για την καταγραφή όλων των κλάσεων για τις οποίες πρέπει να εκπαιδευτούν. Αυτό το αρχείο, για την χρήση του στο Tensorflow, πρέπει να έχει την μορφή .pbtxt. Η δομή των labelmap αρχείων είναι συγκεκριμένη και περιλαμβάνει την εμφάνιση των κλάσεων σαν αντικείμενα τύπου item με 2 παραμέτρους, "id" και "name" όπου id είναι ένας αριθμός που εκφράζει την αρίθμηση των κλάσεων και name είναι το όνομα της κάθε κλάσης. Στην περίπτωση της συγκεκριμένης εργασίας υπάρχει μόνο μία κλάση αντικειμένων για εκπαίδευση, επομένως το labelmap αρχείο που δημιουργήθηκε είχε την μορφή του Πίνακα 4.

```
item {
      id: 1
      name: 'shells'
}
```

Πίνακας 4. Αρχείο labelmap

3.3.5 Δημιουργία των tfrecords

Το Tensorflow χρησιμοποιεί μία συγκεκριμένη δομή binary αρχείων που ονομάζεται tfrecords με σκοπό την αποδοτικότερη αποθήκευση και επεξεργασία των δεδομένων της εκπαίδευσης. Τα αρχεία τύπου .tfrecord αποτελούν τροποποιημένη μορφή των δεδομένων του Dataset. Με τη χρήση των συγκεκριμένων αρχείων, τα δεδομένα της εκπαίδευσης χρησιμοποιούν λιγότερο αποθηκευτικό χώρο και το Tensorflow μπορεί να τα επεξεργαστεί πιο γρήγορα κατά την εκπαίδευση. Στη συγκεκριμένη εργασία, με τη χρήση ενός python script, έγινε η μετατροπή των .csv αρχείων εκπαίδευσης και evaluation, σε δύο αρχεία .tfrecord για την εκπαίδευση και το evaluation των δικτύων αντίστοιχα.

3.4 Βήματα για τη δημιουργία των μοντέλων

3.4.1 Επιλογή μοντέλου

Η δημιουργία των ανιχνευτών αντικειμένων με τη χρήση του Tensorflow απαιτεί μία συγκεκριμένη διαδικασία. Αρχικά πρέπει να γίνει η επιλογή της αρχιτεκτονικής που θα χρησιμοποιηθεί ανάλογα με την περίπτωση εφαρμογής Υπολογιστικής Όρασης μέσα από τους διαθέσιμους αλγορίθμους του Tensorflow Object Detection API (Εικόνα 29). Η επιλογή του κατάλληλου αλγορίθμου είναι σημαντική γιατί θα καθορίσει την ταχύτητα και την ακρίβεια του τελικού αποτελέσματος. Επομένως, σε περιπτώσεις όπου η ακρίβεια ανίχνευσης των αντικειμένων είναι πιο κρίσιμη από την ταχύτητα, προτιμάται ένας αλγόριθμος με υψηλό mAP από τη λίστα υποστηριζόμενων αλγορίθμων του Tensorflow Object Detection API. Αντίστοιχα, στις περιπτώσεις όπου είναι πιο σημαντικό να ανιχνευθούν αντικείμενα σε πραγματικό χρόνο απ' ό,τι να προσδιοριστούν με ακρίβεια οι θέσεις τους, είναι προτιμότεροι οι πιο γρήγοροι αλγόριθμοι.

Model name	Speed (ms)	COCO mAP	Outputs
CenterNet HourGlass104 512x512	70	41.9	Boxes
CenterNet HourGlass104 Keypoints 512x512	76	40.0/61.4	Boxes/Keypoints
CenterNet HourGlass104 1024x1024	197	44.5	Boxes

Εικόνα 29. Λίστα μοντέλων υποστηριζόμενων από το Tensorflow. Source: https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md

Στη συγκεκριμένη εργασία, από τη λίστα προ-εκπαιδευμένων μοντέλων χρησιμοποιήθηκαν τα Faster R-CNN ResNet50 V1 640x640 και SSD ResNet50 V1 FPN 640x640 (RetinaNet50). Ο Faster R-CNN σύμφωνα με τη λίστα των μοντέλων έχει ακρίβεια 29.3mAP στο COCO Dataset, ενώ ο SSD έχει ακρίβεια 34.3mAP. Ο YOLOv3 καθώς δεν υποστηρίζεται εγγενώς από το Tensorflow δεν υπήρχε στη λίστα διαθέσιμων μοντέλων του Object Detection API.

3.4.2 Λήψη προ-εκπαιδευμένων μοντέλων

Τα μοντέλα που περιλαμβάνονται στη λίστα υποστηριζόμενων μοντέλων του Object Detection API του Tensorflow, αποτελούν προ-εκπαιδευμένα μοντέλα για συγκεκριμένα μεγάλα Datasets. Αυτό γίνεται για να μπορέσει να εκπαιδευτεί ένα μοντέλο με μικρά Datasets όπως αυτό της συγκεκριμένης εργασίας. Η μέθοδος αυτή λέγεται *Transfer*

Learning και με αυτήν εκπαιδεύονται οι αλγόριθμοι με χρήση μεγάλων Datasets για να αρχικοποιηθούν τα weights and biases των δικτύων. Με αυτό τον τρόπο, μπορεί να επιτευχθεί ανίχνευση αντικειμένων χρησιμοποιώντας Datasets μερικών εκατοντάδων εικόνων όπως το Dataset της συγκεκριμένης εργασίας.

Επομένως, το επόμενο βήμα της δημιουργίας των ανιχνευτών είναι η λήψη των προ-εκπαιδευμένων μοντέλων που επιλέχθηκαν για την ανίχνευση αντικειμένων σε δίχτυα Ιχθυοκαλλιέργειας, από την επίσημη πηγή του Tensorflow Object Detection API.

3.4.3 Configuration των αλγορίθμων

Καθένας από τους αλγορίθμους ανίχνευσης αντικειμένων χρησιμοποιεί ένα αρχείο για την αρχικοποίηση των παραμέτρων της εκπαίδευσης και του evaluation. Αυτό το αρχείο είναι συνήθως της μορφής .config ή .cfg και περιέχει μεταξύ άλλων, τις δηλώσεις των διαστάσεων των εικόνων, τον αριθμό των βημάτων εκπαίδευσης, το batch size, το learning rate και τα paths στα αρχεία του Dataset.

Αρχικά έγινε η λήψη των συγκεκριμένων αρχείων από τη σελίδα του Tensorflow Object Detection API και έπειτα έγινε η κατάλληλη ρύθμιση ορισμένων παραμέτρων για τη σωστή εκπαίδευση τους. Αυτές οι παράμετροι είναι οι εξής:

- Διαστάσεις εικόνας
- Αριθμός βημάτων εκπαίδευσης
- Batch size
- Μονοπάτι προς τις εικόνες εκπαίδευσης και evaluation
- Μονοπάτι προς το Labelmap
- Μονοπάτι προς το προ-εκπαιδευμένο μοντέλο

3.4.4 Εκτέλεση της εκπαίδευσης και του evaluation των μοντέλων

Τα τελευταία στάδια της δημιουργίας των ανιχνευτών είναι η εκπαίδευση και του evaluation τους. Με το Tensorflow, η εκπαίδευση και το evaluation επιτυγχάνονται με το ίδιο python script θέτοντας διαφορετικές παραμέτρους. Οι παράμετροι που χρησιμοποιούνται ως είσοδοι στην εκτέλεση του script είναι το αρχείο .tfrecords, το configuration αρχείο και το μονοπάτι για τα αρχεία εξόδου.

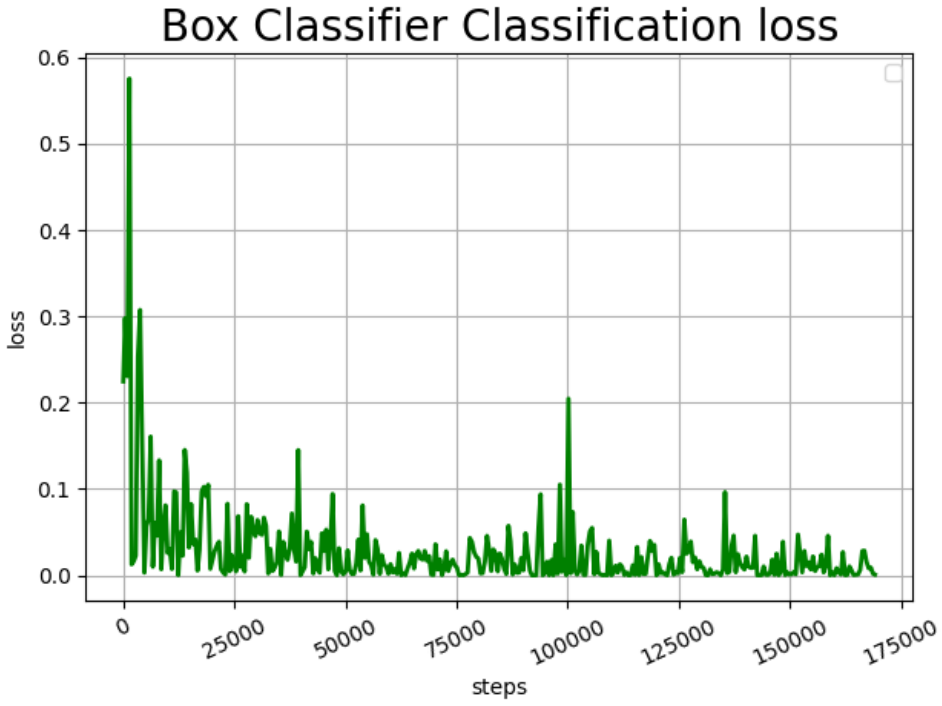
3.5 Εκπαίδευση του Faster R-CNN

Με χρήση της μεθόδου που αναφέρθηκε στο κεφάλαιο 3.4, έγινε η εκπαίδευση του Faster R-CNN. Αρχικά χρησιμοποιήθηκε το Anaconda για τη δημιουργία εικονικού περιβάλλοντος για την εγκατάσταση των εργαλείων και των dependencies που απαιτούνταν για την

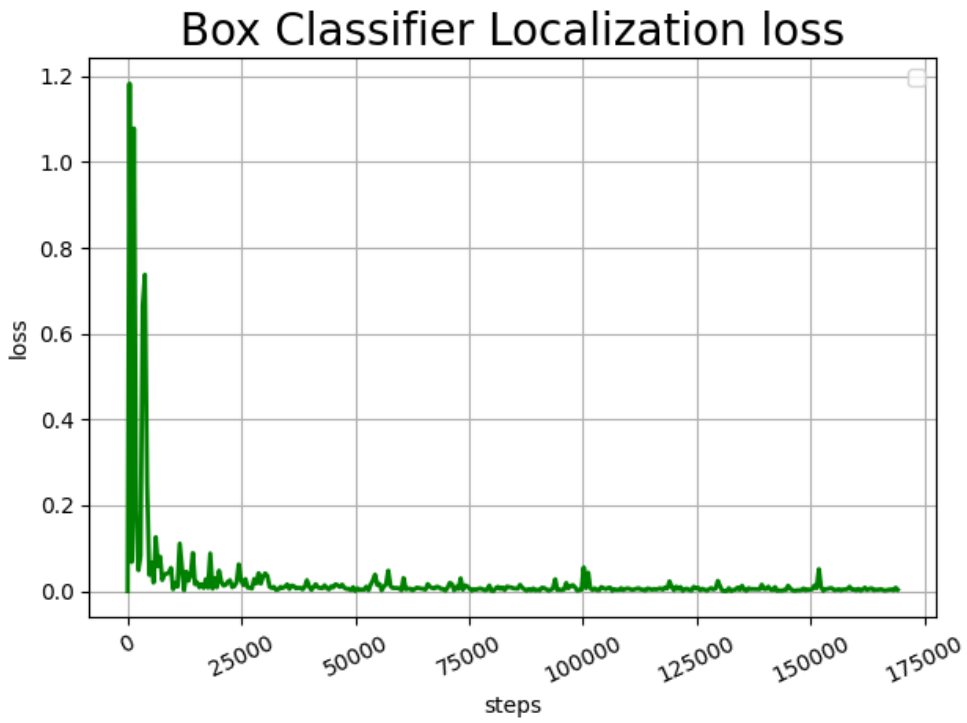
εκπαίδευση. Έπειτα, στο εικονικό περιβάλλον που δημιουργήθηκε, εγκαταστάθηκε το Tensorflow version 1.15 καθώς και το Tensorflow-gpu version 1.15. Το δεύτερο, κάνει χρήση της κάρτας γραφικών του συστήματος επιταχύνοντας την εκπαίδευση του μοντέλου από 6 δευτερόλεπτα ανά βήμα με χρήση cpu, στα 0.3 δευτερόλεπτα ανά βήμα. Για την ενεργοποίηση της χρήσης της κάρτας γραφικών έπρεπε να εγκατασταθούν ορισμένα εργαλεία της Nvidia όπως το Cuda v10.1 και το CuDnn 10.2.

Έπειτα, έγινε ο ορισμός των κατάλληλων παραμέτρων του configuration αρχείου για την εκπαίδευση και το evaluation. Συγκεκριμένα, ορίστηκε η εκπαίδευση να γίνει για 200000 βήματα με batch size ίσο με 1. Batch size είναι ο αριθμός των εικόνων που χρησιμοποιούνται για κάθε βήμα της εκπαίδευσης. Μεγαλύτερος αριθμός batch συνήθως οδηγεί σε καλύτερα αποτελέσματα στο evaluation καθώς γίνεται πιο ομαλή η μείωση του σφάλματος ανίχνευσης στην πορεία της εκπαίδευσης. Το αρνητικό είναι ότι όσο αυξάνεται το batch size, τόσο αυξάνεται η απαιτούμενη μνήμη της κάρτας γραφικών που θα εκτελέσει την εκπαίδευση. Επομένως, για την εκπαίδευση του Faster R-CNN στον διαθέσιμο τοπικό υπολογιστή, χρησιμοποιήθηκε batch size ίσο με 1 λόγω περιορισμού μνήμης.

Τα checkpoint αρχεία μαζί με το configuration αρχείο χρησιμοποιήθηκαν για τη δημιουργία ενός tfevents αρχείου. Αυτό το .tfevents αρχείο χρησιμοποιήθηκε από το Tensorboard για να παραχθούν τα γραφήματα των αποτελεσμάτων του Faster R-CNN μοντέλου.

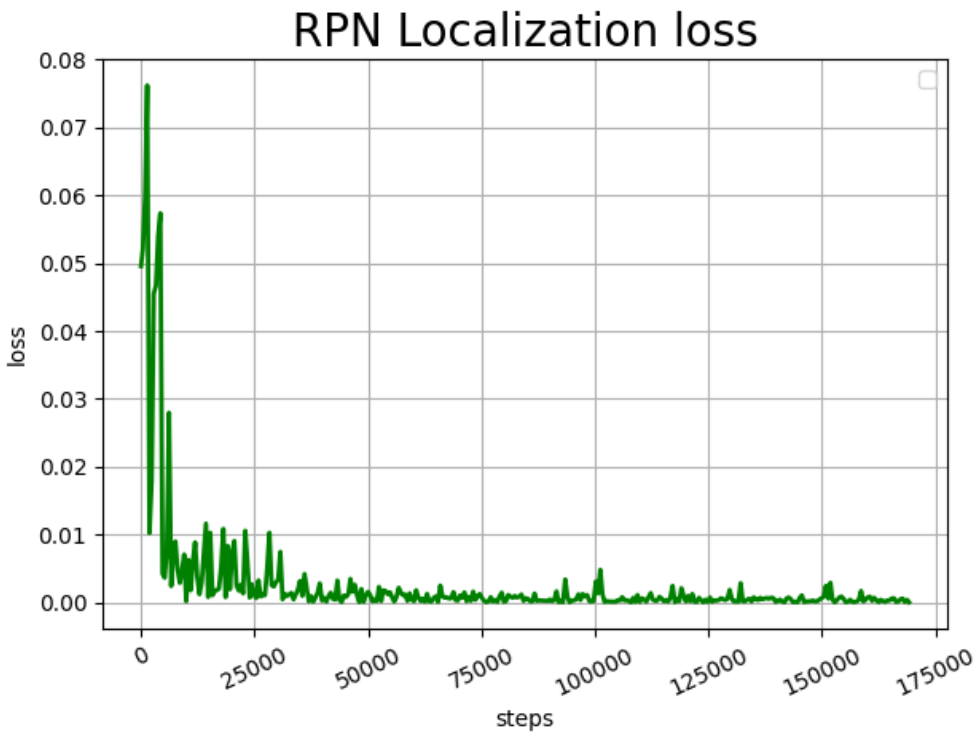


Εικόνα 30. Classification loss του Box Classifier σε σχέση με τον αριθμό των βημάτων εκπαίδευσης

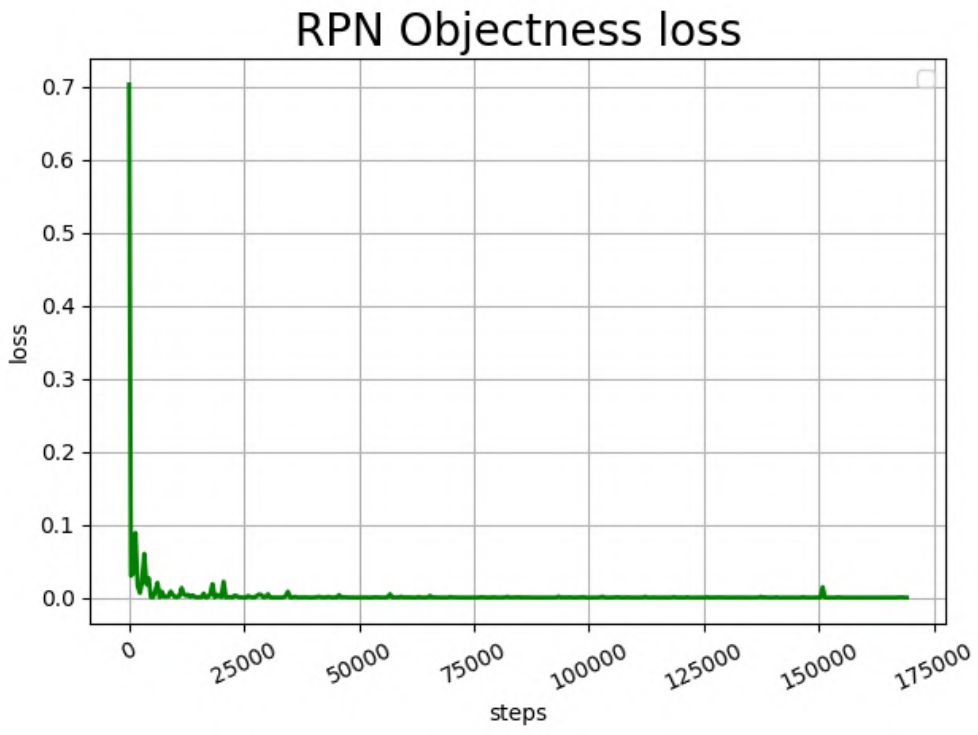


Εικόνα 31. Localization loss του Box Classifier σε σχέση με τον αριθμό των βημάτων εκπαίδευσης

Στις εικόνες 30 και 31, φαίνονται τα classification και localization σφάλματα του Box Classifier. Localization loss στον Box classifier είναι μία τιμή απόκλισης της θέσης των Bounding Boxes που δημιουργήθηκαν σε σχέση με τις επιθυμητές τους θέσεις, δηλαδή όσο μικρότερο το localization loss τόσο μεγαλύτερη η ακρίβεια τοποθέτησης των Bounding Boxes γύρω από τα αντικείμενα. Αντίστοιχα, το Classification loss του Box Classifier είναι μία τιμή που εκφράζει πόσο σωστά εφαρμόστηκε η κατάλληλη κλάση σε κάθε Bounding Box. Το Localization loss βλέπουμε ότι μειώνεται απότομα μέχρι τα 40000 βήματα και μετά ακολουθεί πιο ήπια καθοδική πορεία μέχρι το τέλος της εκπαίδευσης στα 180000 βήματα. Για το Classification loss είναι πιο δύσκολο να διακρίνουμε την πτωτική καμπύλη καθώς παρουσιάζει έντονες διακυμάνσεις. Αυτό οφείλεται στη χρήση batch number ίσου με 1. Στην περίπτωση που η εκπαίδευση γινόταν με μεγαλύτερο batch number, θα εισάγονταν περισσότερες εικόνες σε κάθε βήμα της εκπαίδευσης και θα εξομαλυνόταν η καμπύλη.

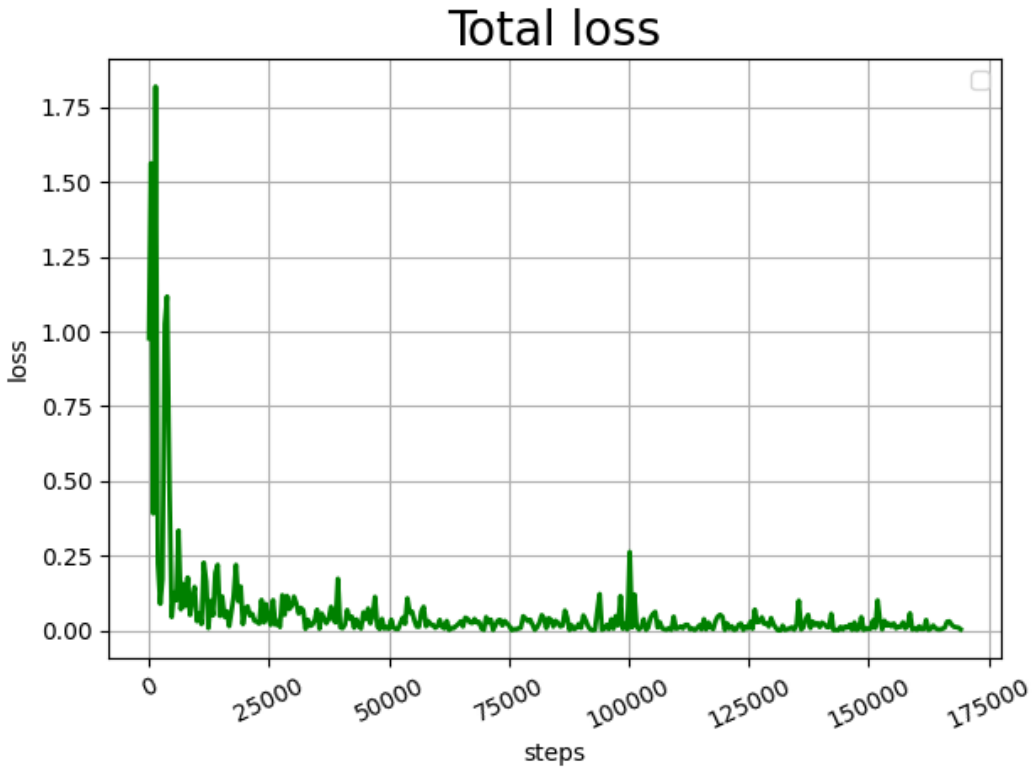


Εικόνα 32. Localization loss του RPN σε σχέση με τον αριθμό των βημάτων εκπαίδευσης



Εικόνα 33. Objectness loss του RPN σε σχέση με τον αριθμό των βημάτων εκπαίδευσης

Στον Faster R-CNN χρησιμοποιείται επιπλέον ένα δίκτυο RPN για το οποίο προκύπτουν κάποια γραφήματα μέσω του Tensorboard. Τα γραφήματα αυτά δείχνουν τη μείωση των localization loss (Εικόνα 32) και objectness loss (Εικόνα 33) του RPN δικτύου στην πορεία της εκπαίδευσης.

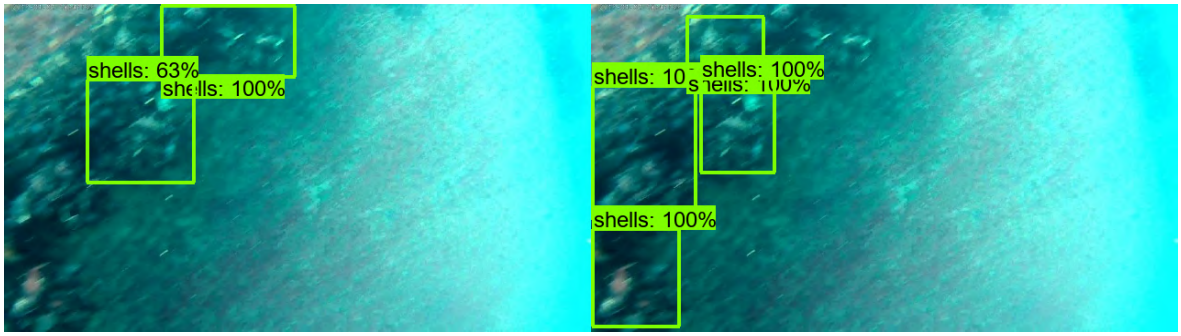


Εικόνα 34. Total loss σε σχέση με τον αριθμό των βημάτων εκπαίδευσης

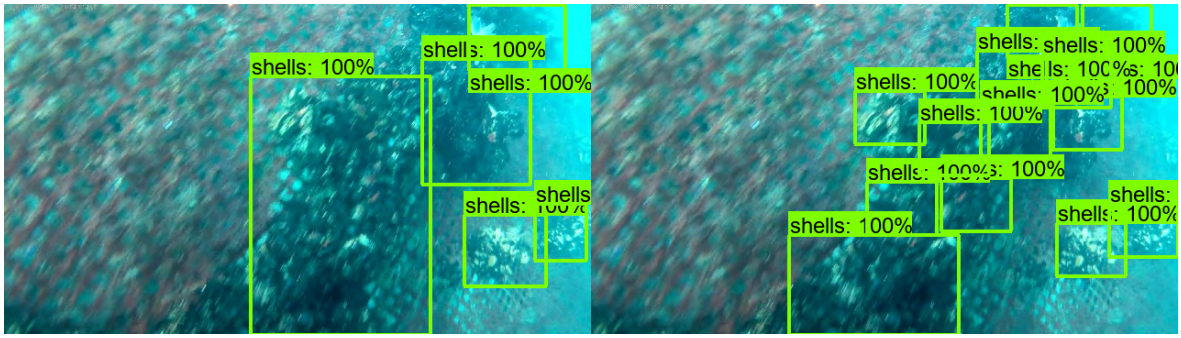
Στο γράφημα της Εικόνας 34 φαίνεται το συνολικό loss του αλγορίθμου κατά την εκπαίδευση. Παρατηρείται ότι αρχικά μειώνεται γρήγορα και στη συνέχεια φαίνεται να μειώνεται με μικρότερο ρυθμό, παρά τις μεγάλες διακυμάνσεις λόγω του μικρού batch number. Το γράφημα αυτό δείχνει ότι το μοντέλο βελτιώνεται όσο προχωράει η εκπαίδευση.

Οι εικόνες 35 - 39 είναι το αποτέλεσμα του evaluation του εκπαιδευμένου μοντέλου. Συγκεκριμένα, οι εικόνες στα δεξιά αποτελούν εικόνες αναφοράς (ground truths) δηλαδή τις εικόνες του Dataset που επιλέχθηκαν τυχαία για το evaluation των δικτύων όπως προέκυψαν από το χειροκίνητο Labeling κατά τη δημιουργία του Dataset. Αριστερά, είναι τα αποτελέσματα της χρήσης του Faster R-CNN στις συγκεκριμένες εικόνες κατά το evaluation του δικτύου. Παρατηρείται ότι στα περισσότερα Bounding Boxes, ο Faster R-CNN έχει αναγνωρίσει τα αντικείμενα με ποσοστό 100% ενώ μόνο σε μία περίπτωση φαίνεται ποσοστό αναγνώρισης 63%. Επιπλέον, φαίνεται να αναγνωρίστηκαν ως ένα μεγάλο αντικείμενο, αντικείμενα που στο Dataset είχαν προσδιοριστεί ως πολλά μικρότερα.

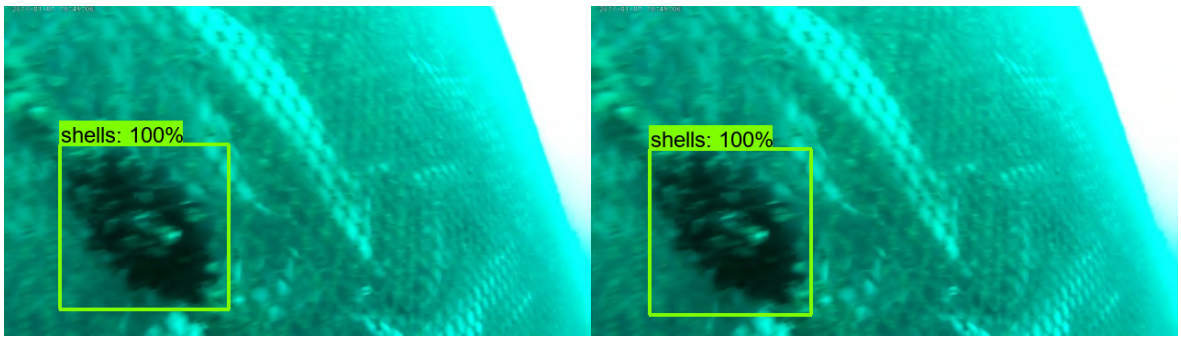
Επίσης, δεν φαίνεται ξεκάθαρα κάποια περιοχή η οποία έχει προσδιοριστεί από το Dataset ως αντικείμενο προς ανίχνευση, που να μην έχει ανιχνευθεί από τον Faster R-CNN.



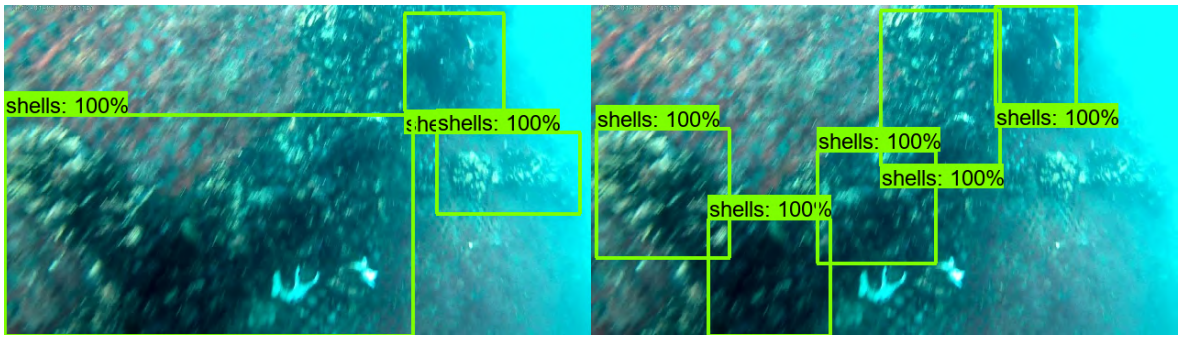
Εικόνα 35.



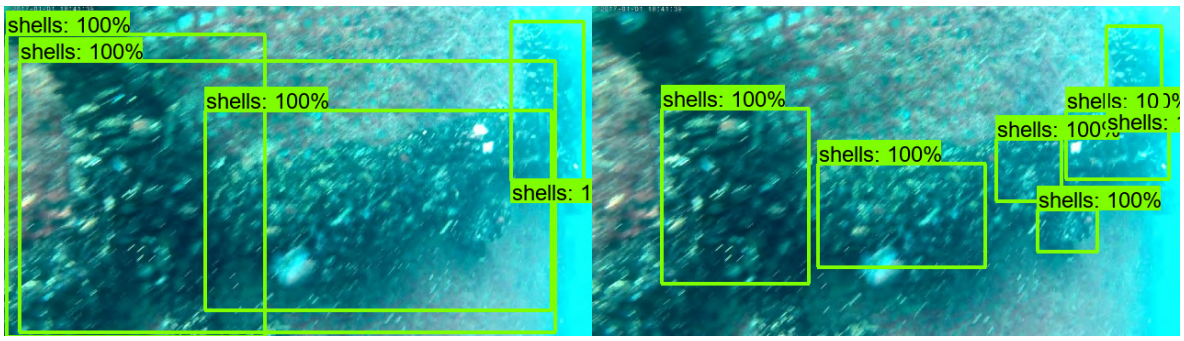
Εικόνα 36.



Εικόνα 37.



Εικόνα 38.



Εικόνα 39.

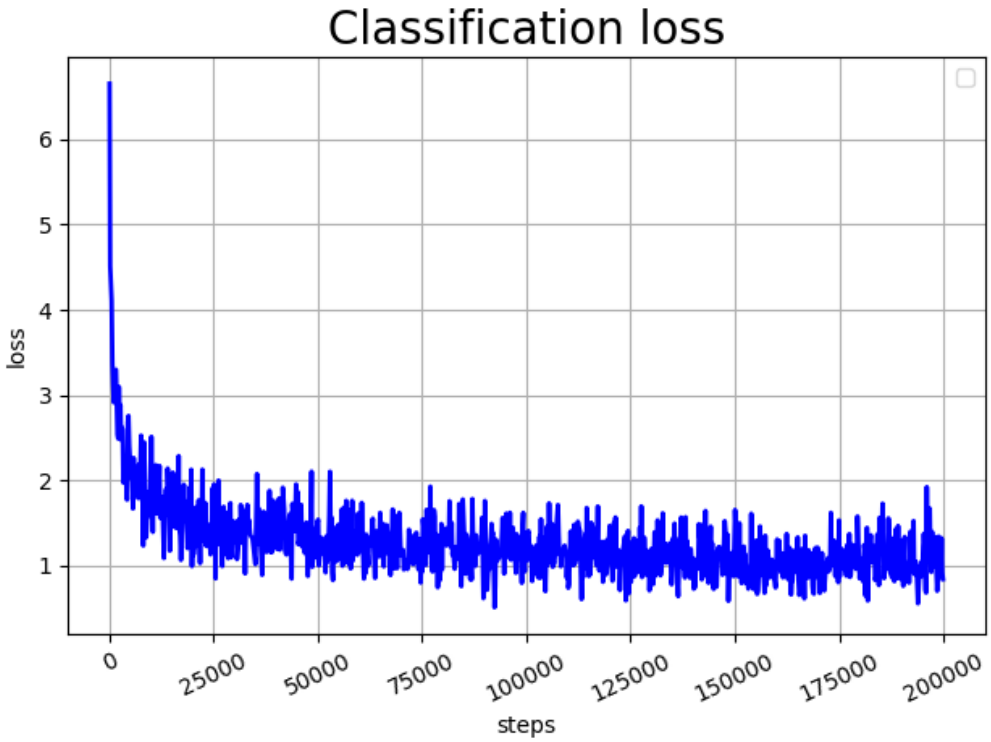
3.6 Εκπαίδευση του SSD

Η εκπαίδευση του SSD έγινε με την εφαρμογή των αντίστοιχων βημάτων με την εκπαίδευση του Faster R-CNN.

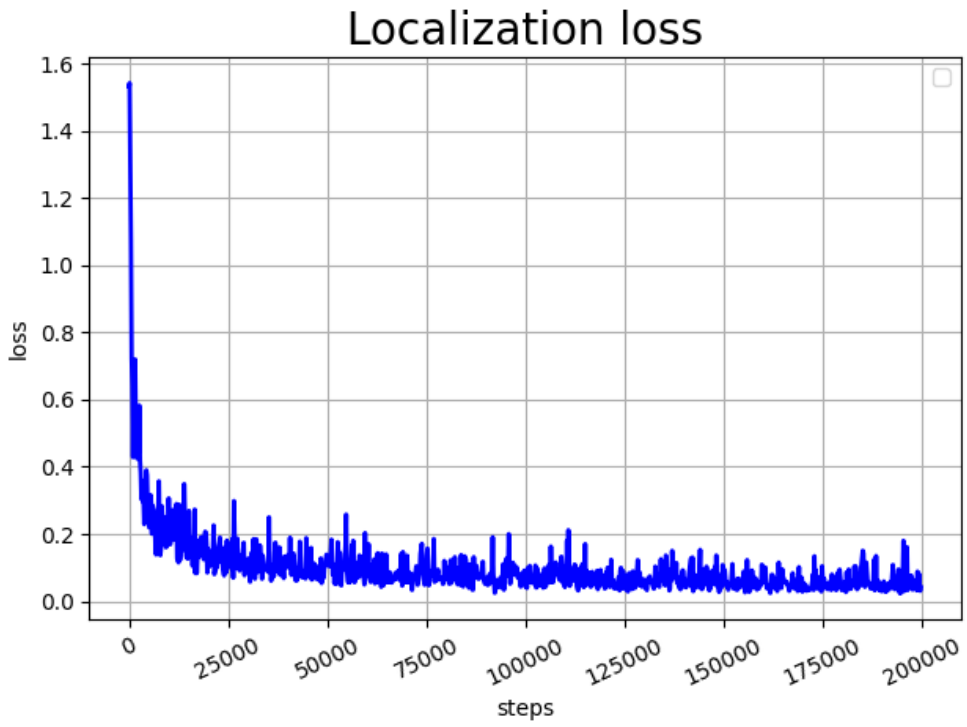
Αρχικά χρησιμοποιήθηκε το Anaconda για τη δημιουργία εικονικού περιβάλλοντος και την εγκατάσταση των εργαλείων και των dependencies που απαιτούνταν για την εκπαίδευση. Έπειτα έγινε η παραμετροποίηση του configuration αρχείου για τον SSD. Συγκεκριμένα, όπως και για το configuration του Faster R-CNN, ορίστηκαν οι θέσεις των tfrecords αρχείων της εκπαίδευσης και του evaluation, του labelmap αρχείου, καθώς και του προεκπαιδευμένου μοντέλου του SSD. Επιπλέον, τα βήματα εκπαίδευσης τέθηκαν στα 200000, οι διαστάσεις εικόνων ορίστηκαν ως 640 x 360 και το batch number πήρε τη μέγιστη τιμή για τη διαθέσιμη μνήμη, που ήταν 24.

Στη συνέχεια εκτελέστηκε η εντολή για εκπαίδευση του δικτύου με παραμέτρους το configuration αρχείο, το tfrecords αρχείο και την θέση των αρχείων εξόδου. Μετά το τέλος της εκπαίδευσης, κλήθηκε η ίδια εντολή με διαφορετικό tfrecords αρχείο, με σκοπό το evaluation του δικτύου που εκπαιδεύτηκε. Στη συνέχεια, με τη χρήση του Tensorboard, έγινε η οπτικοποίηση των αποτελεσμάτων του training και του evaluation.

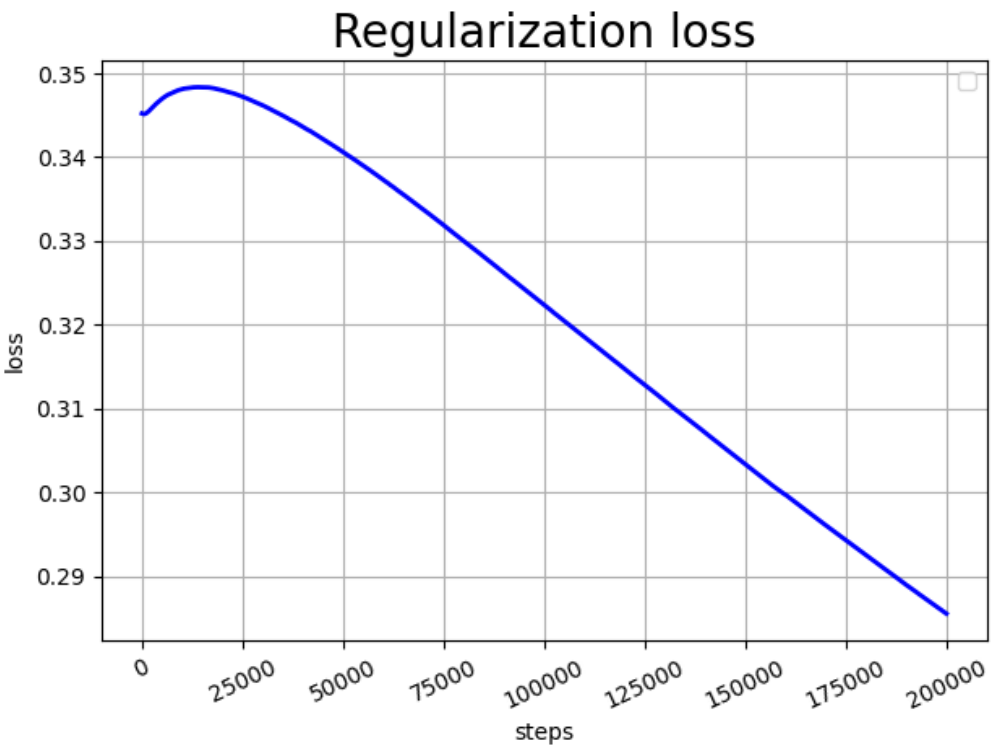
Τα γραφήματα των εικόνων 40 - 43 απεικονίζουν τις επιμέρους απώλειες του μοντέλου κατά την εκπαίδευση. Παρατηρείται ότι όπως και στον Faster R-CNN, οι απώλειες μειώνονται κατά τη διάρκεια της εκπαίδευσης. Πιο συγκεκριμένα, τα Classification και Localization losses μειώνονται με μεγάλο ρυθμό κατά τα πρώτα 25000 στάδια εκπαίδευσης ενώ συνεχίζουν την μείωση με μικρότερο ρυθμό μέχρι το τέλος της εκπαίδευσης. Αυτό σημαίνει πρακτικά ότι το συγκεκριμένο μοντέλο δεν επωφελείται σημαντικά από εκπαίδευση πάνω από 25000 βήματα.



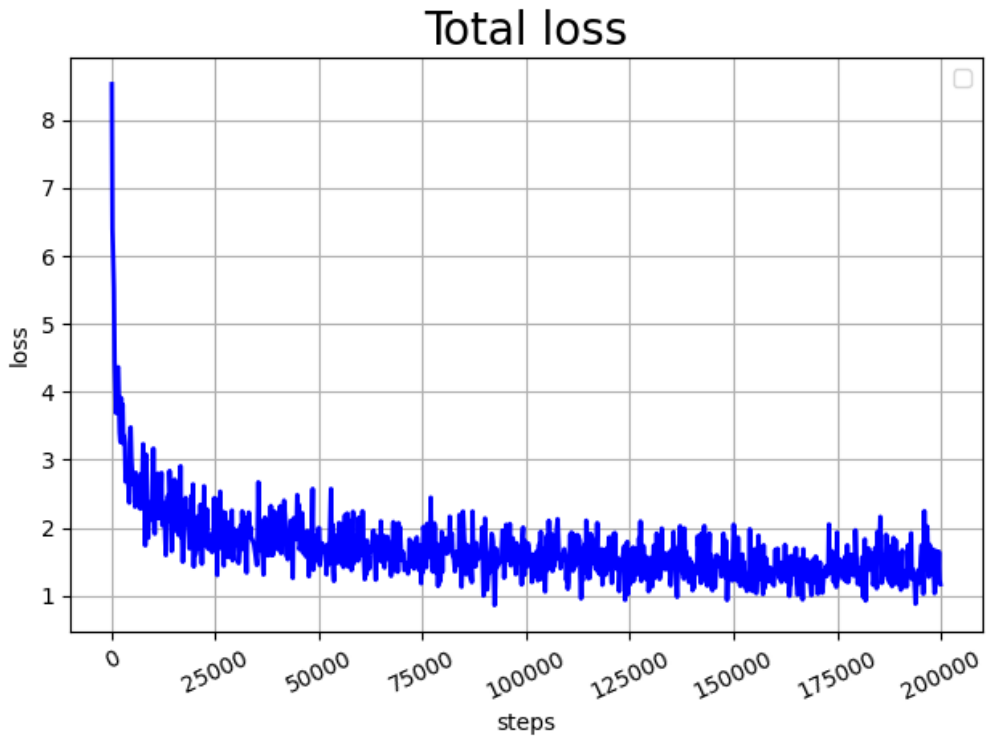
Εικόνα 40. Classification loss σε σχέση με τον αριθμό των βημάτων εκπαίδευσης



Εικόνα 41. Localization loss σε σχέση με τον αριθμό των βημάτων εκπαίδευσης

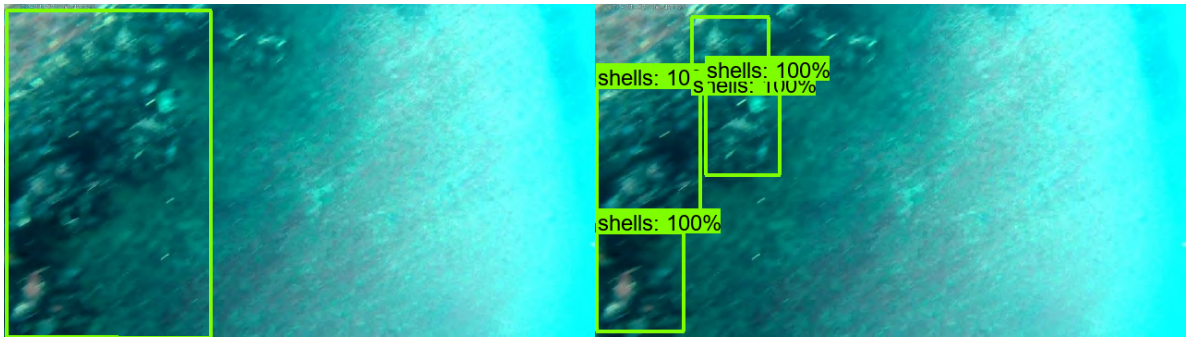


Εικόνα 42. Reguralization loss σε σχέση με τον αριθμό των βημάτων εκπαίδευσης



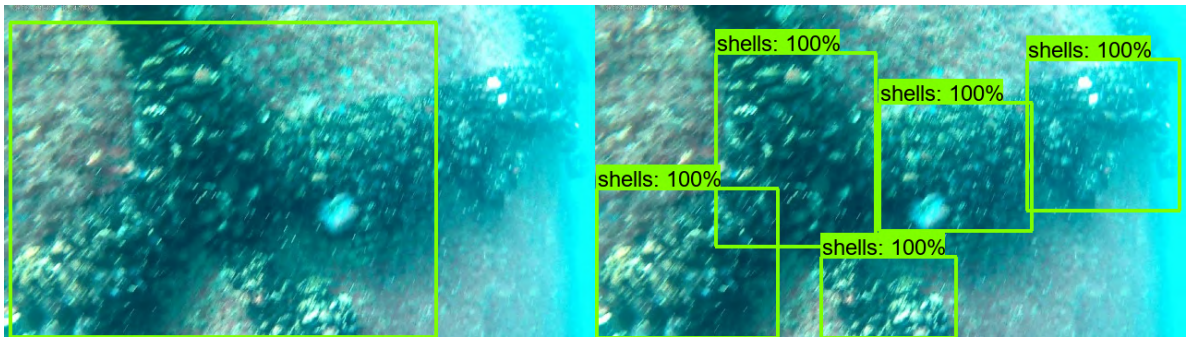
Εικόνα 43. Total loss σε σχέση με τον αριθμό των βημάτων εκπαίδευσης

Στις εικόνες 44 - 49 φαίνονται τα αποτελέσματα του evaluation. Συγκεκριμένα, στα δεξιά απεικονίζονται οι εικόνες όπως ορίστηκαν στο Dataset, ενώ αριστερά φαίνονται οι αντίστοιχες εικόνες με τα Bounding Boxes και τα confidences των αντικειμένων που ανιχνεύτηκαν από τον SSD. Παρατηρείται ότι ο SSD τείνει να ανιχνεύει πολλά μικρά αντικείμενα ως ένα μεγάλο, σε πιο έντονο βαθμό από τον Faster R-CNN. Αυτό έχει ως αποτέλεσμα σε ορισμένες περιοχές να μην έχει τοποθετηθεί Bounding Box ενώ υπάρχει Bounding Box στην εικόνα αναφοράς.

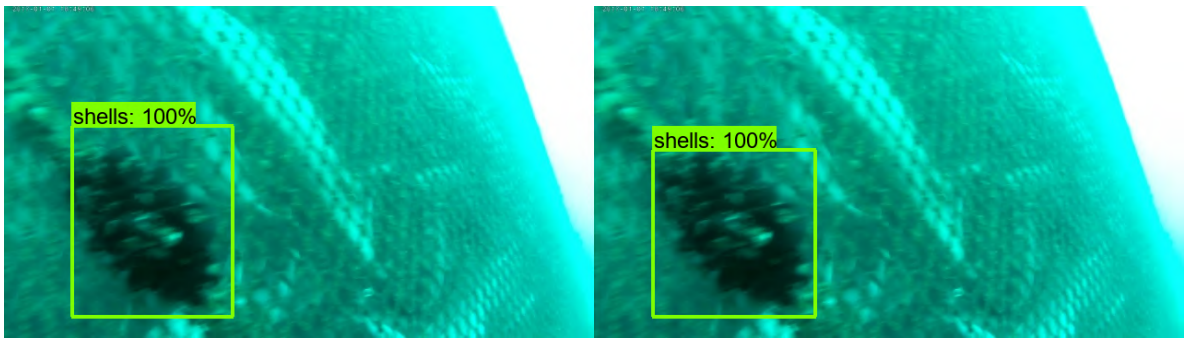


Εικόνα 44.

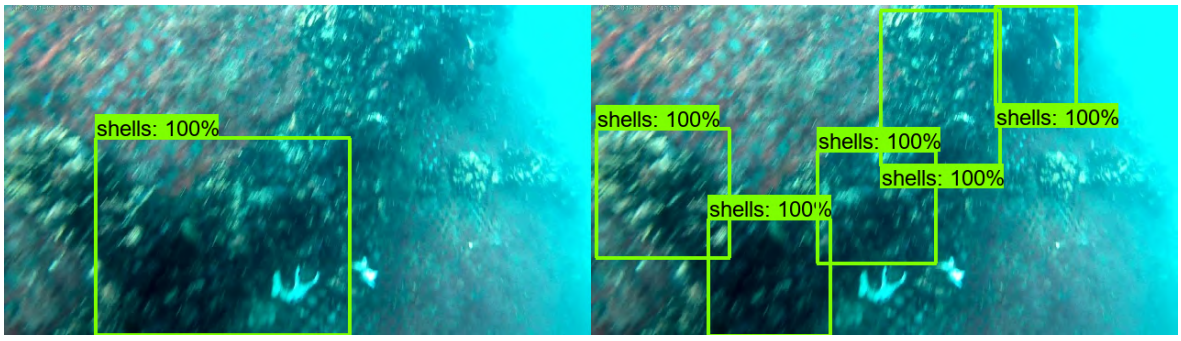
Εικόνα 45.



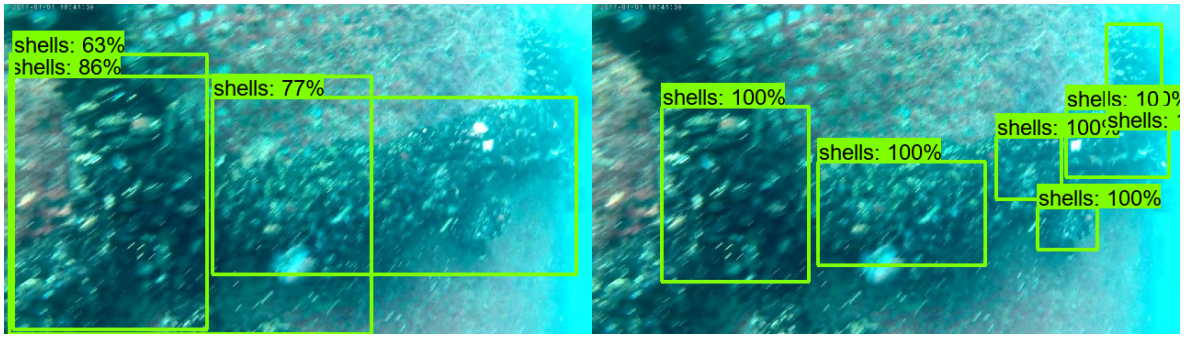
Εικόνα 46.



Εικόνα 47.



Εικόνα 48.



Εικόνα 49.

3.7 Εκπαίδευση του YOLOv3

Ο αλγόριθμος YOLO αναπτύχθηκε με βάση το Darknet framework και κατά συνέπεια χρησιμοποιεί διαφορετικού είδους αρχεία και διαφορετική μεθοδολογία από αυτά που υποστηρίζει το Tensorflow. Επομένως, θα μπορούσε να γίνει η εκπαίδευση του YOLOv3 στο Darknet και να γίνει η σύγκριση των αποτελεσμάτων με τα αποτελέσματα των Faster R-CNN και SSD που εκπαιδεύτηκαν μέσω Tensorflow. Ο YOLO χρησιμοποιεί επίσης διαφορετικού τύπου αρχεία για το Labeling του Dataset. Για λόγους συνέπειας των πειραμάτων, επιλέχθηκε να χρησιμοποιηθεί το ίδιο Dataset και τα ίδια εργαλεία για την εκπαίδευση και το evaluation όλων των αλγορίθμων, επομένως και ο YOLO για τη

συγκεκριμένη εργασία έπρεπε να εκπαιδευτεί και να δοκιμαστεί με τα ίδια εργαλεία (Tensorflow, Tensorboard).

Για την επίλυση του προβλήματος συμβατότητας του YOLO με το Tensorflow, υπάρχουν διάφορα repositories στην πλατφόρμα GitHub από τις προσπάθειες άλλων ατόμων να επιτύχουν το αντίστοιχο. Ένα από αυτά, το οποίο χρησιμοποιήθηκε στη συγκεκριμένη εργασία, περιλάμβανε αρχεία για την εκπαίδευση και το evaluation του YOLOv3 μέσω του Tensorflow 2.0.

Η διαδικασία έγινε σε ένα ξεχωριστό εικονικό περιβάλλον που δημιουργήθηκε μέσω του Anaconda όπως και στις προηγούμενες περιπτώσεις. Η προετοιμασία του μοντέλου για εκπαίδευση ξεκίνησε με την εγκατάσταση ορισμένων απαιτούμενων πακέτων για την εκπαίδευση (Πίνακας 5).

numpy
pillow
scipy
wget
seaborn
easydict
grpcio

Πίνακας 5. Απαιτούμενα πακέτα εκπαίδευσης του YOLO μέσω Tensorflow

Στη συνέχεια έγινε η λήψη του προεκπαιδευμένου μοντέλου του YOLO το οποίο είναι ένα αρχείο της μορφής .weights σε αντίθεση με τα μοντέλα του Tensorflow που έχουν τη μορφή .pb.

Επόμενο βήμα ήταν η αλλαγή της δομής των .csv annotations αρχείων του Dataset. Η μορφή που υποστηρίζει το Tensorflow περιλαμβάνει την εμφάνιση κάθε αντικειμένου μιας εικόνας σε μία ξεχωριστή γραμμή του αρχείου μαζί με το label σε μορφή string και τις διαστάσεις του Bounding Box. Για την εκπαίδευση του YOLO, η δομή των .csv αρχείων έπρεπε να αλλάξει σε .txt έτσι ώστε κάθε γραμμή να αντιστοιχεί σε μία εικόνα με όλα τα αντικείμενα που περιέχει και τις διαστάσεις των αντίστοιχων Bounding Boxes, ενώ τα αντικείμενα μεταξύ τους χωρίζονται από τον κενό χαρακτήρα. Επιπλέον, το label του κάθε αντικειμένου περιγράφεται με έναν ακέραιο δείκτη αντί για το string στο οποίο αντιστοιχεί, με το πρώτο αντικείμενο στο αρχείο names (το αντίστοιχο labelmap) να παίρνει τον δείκτη 0, το δεύτερο τον δείκτη 1, κλπ.

Στη συνέχεια, δημιουργήθηκε το αρχείο .names το οποίο είναι το αντίστοιχο labelmap (.pbtxt) που χρησιμοποιείται στο Tensorflow για να καταγράφει τα ονόματα των αντικειμένων προς αναγνώριση. Για την εκπαίδευση του YOLO, αυτό το αρχείο είναι της μορφής .names και δεν περιλαμβάνει κάποια περίπλοκη δομή, μόνο τα ονόματα των αντικειμένων προς αναγνώριση γράφονται σε ξεχωριστές γραμμές του αντικειμένου. Στο .csv αρχείο του Dataset, χρησιμοποιείται ο αριθμός γραμμής στο αρχείο .names του ονόματος του κάθε αντικειμένου.

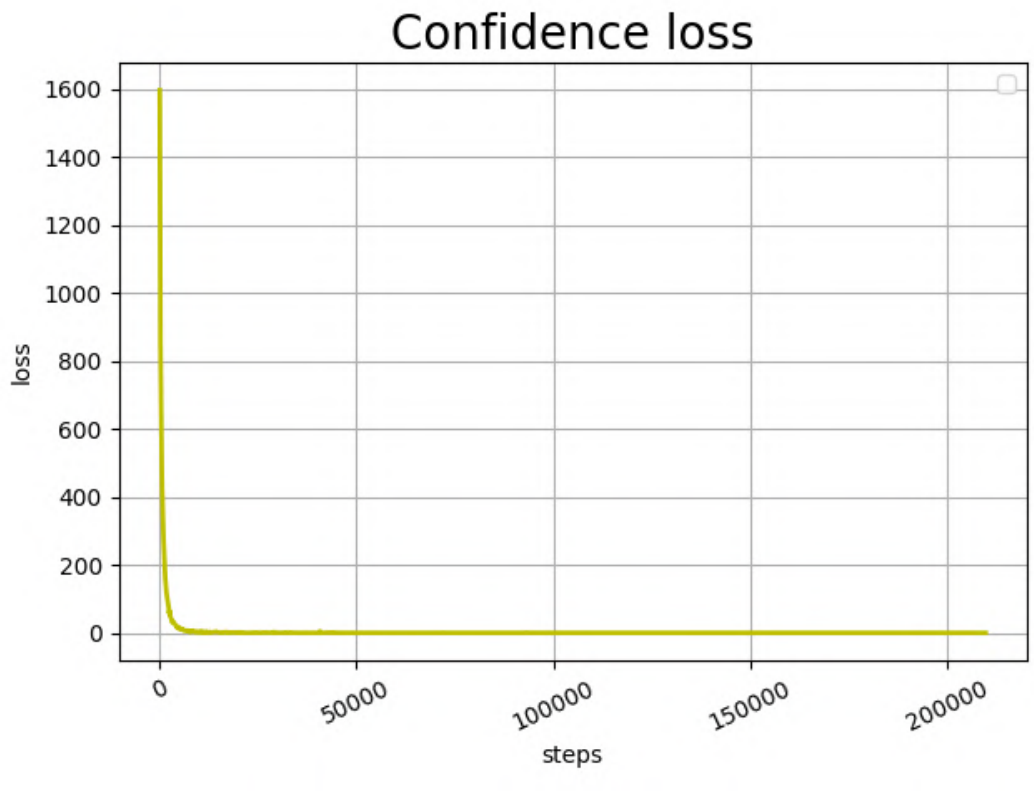
Το τελευταίο στάδιο της προετοιμασίας για την εκπαίδευση του YOLO ήταν η παραμετροποίηση του configuration αρχείου. Συγκεκριμένα, ορίστηκε η θέση του αρχείο .names και του τροποποιημένου αρχείου .txt του Dataset για την εκπαίδευση και για του αντίστοιχου αρχείου για το evaluation. Σε αυτό το σημείο δεν χρειάζεται να οριστεί η θέση των εικόνων του Dataset καθώς η θέση τους ορίζεται στα.txt annotations αρχεία της εκπαίδευσης και του evaluation. Επιπλέον, ορίστηκε η θέση για την έξοδο των αποτελεσμάτων του evaluation. Όπως και στα προηγούμενα μοντέλα, ορίστηκε το μεγαλύτερο δυνατό batch size για τον YOLO με βάση τη διαθέσιμη μνήμη, το οποίο ήταν 4. Για να προκύψει η μέγιστη τιμή του batch size, εκτελέστηκε η εκπαίδευση των δικτύων πολλές φορές με μεγαλύτερα batch sizes μέχρι να μην προκύπτει σφάλμα λόγω έλλειψης μνήμης. Επιπλέον, ορίστηκε ο αριθμός train.epoch ο οποίος είναι το αντίστοιχο του training steps που υπάρχει στα configuration αρχεία των προηγούμενων αλγορίθμων. Η διαφορά του train.epoch με τα βήματα εκπαίδευσης είναι ότι ο train.epoch μετράει τον αριθμό προσπελάσεων του Dataset, ο οποίος εξαρτάται από το batch size. Στους προηγούμενους αλγορίθμους χρησιμοποιήθηκαν 200000 βήματα, επομένως για να χρησιμοποιηθεί ο ίδιος αριθμός βημάτων στον YOLO, θα πρέπει να υπολογιστεί το train.epoch μέσω του τύπου 4.

$$200000\ required\ steps\ /\ (341\ Dataset\ images\ /\ 4\ batch\ size)\ =\ 2439\ epochs$$

Τύπος 4.

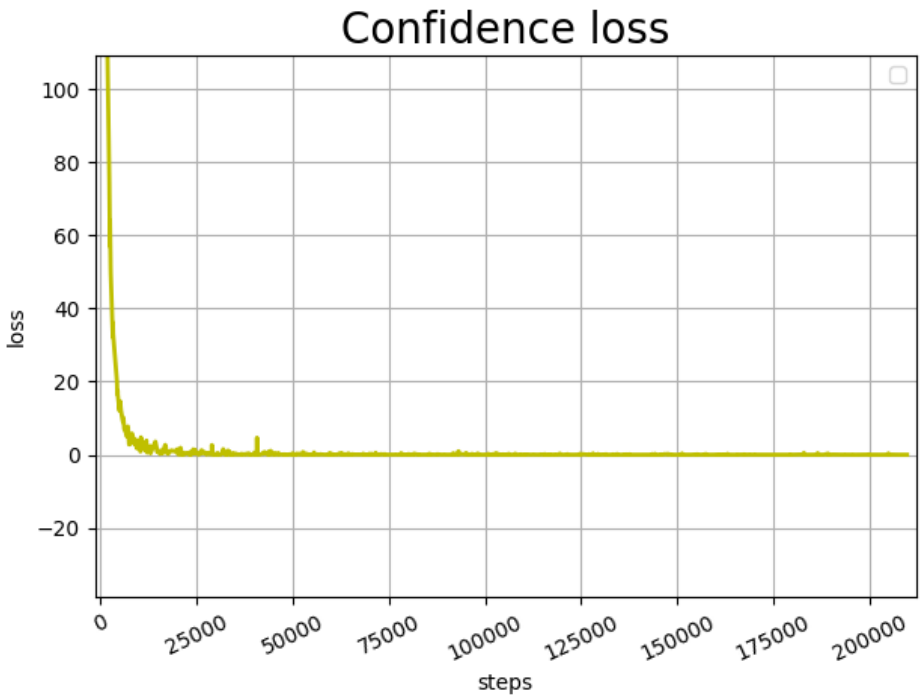
Επομένως, ορίστηκε το 2439 ως αριθμός train.epoch στο configuration αρχείο.

Στη συνέχεια με χρήση του κατάλληλου αρχείου, ξεκίνησε η εκπαίδευση του YOLO μέσω του Tensorflow. Επειδή χρησιμοποιήθηκε το Tensorflow, δημιουργήθηκαν τα αντίστοιχα .tfevents αρχεία τα οποία υποστηρίζονται από το Tensorboard για την εξαγωγή γραφημάτων της εκπαίδευσης του YOLO. Επομένως, μετά την εκπαίδευση, χρησιμοποιήθηκε το Tensorboard για την εξαγωγή και οπτικοποίηση των αποτελεσμάτων.



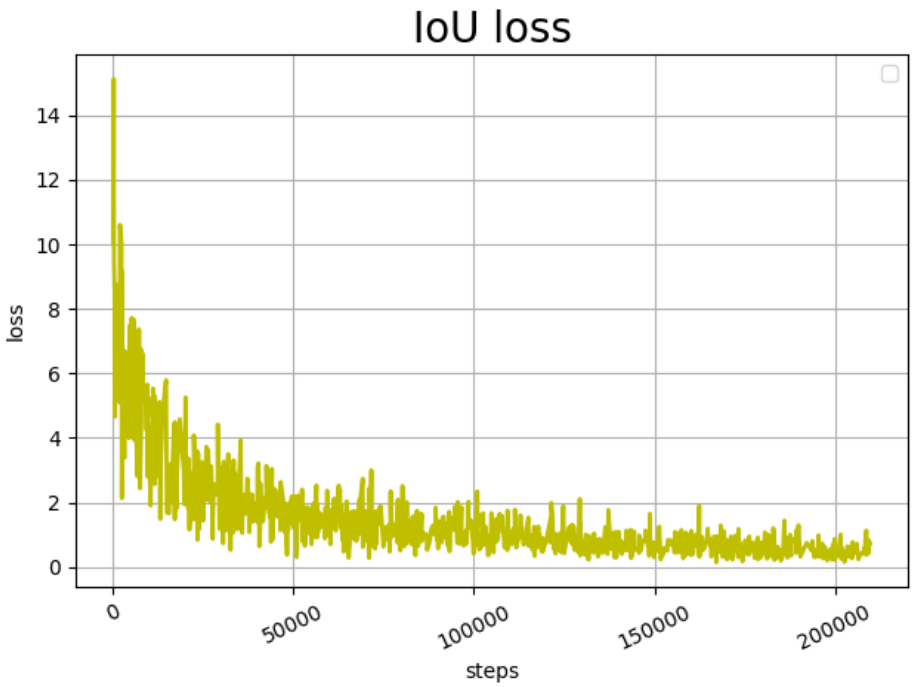
Εικόνα 50. Confidence loss σε σχέση με τον αριθμό των βημάτων εκπαίδευσης

Στην Εικόνα 50 φαίνεται το confidence loss κατά τη διάρκεια της εκπαίδευσης. Αρχικά φαίνεται ότι το σφάλμα είναι πολύ μεγάλο και μειώνεται πολύ γρήγορα, με αποτέλεσμα να μην διακρίνονται οι τυχόν διακυμάνσεις στα μετέπειτα βήματα εκπαίδευσης. Γι αυτό τον λόγο, στο γράφημα της Εικόνας 50 έγινε εστίαση στις πιο χαμηλές τιμές σφάλματος όπως φαίνεται στο γράφημα της Εικόνας 51.

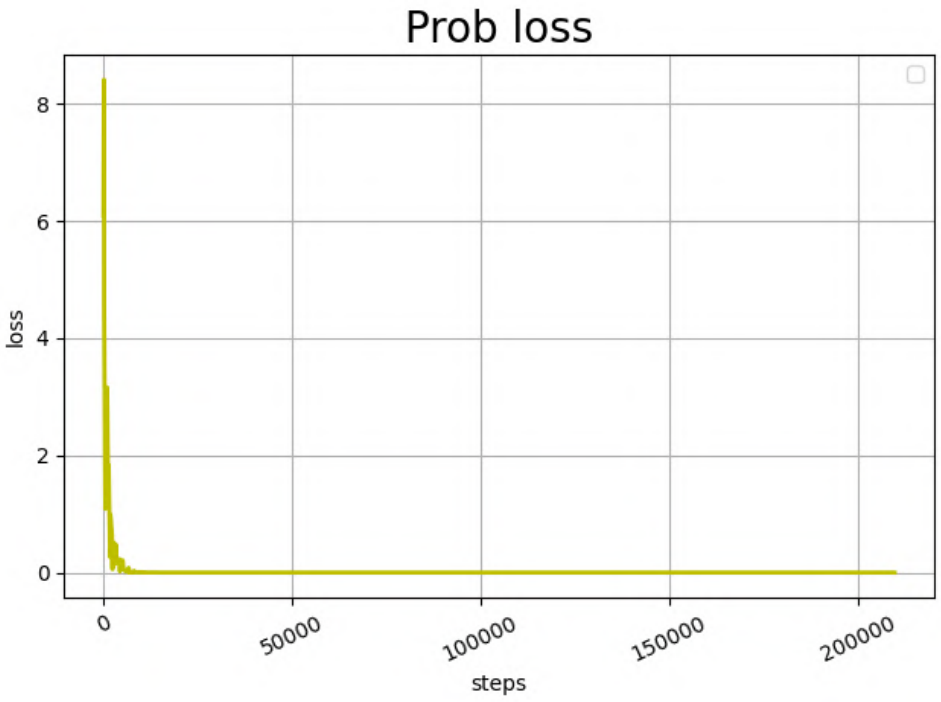


Εικόνα 51. Confidence loss εστιασμένο σε σχέση με τον αριθμό των βημάτων εκπαίδευσης

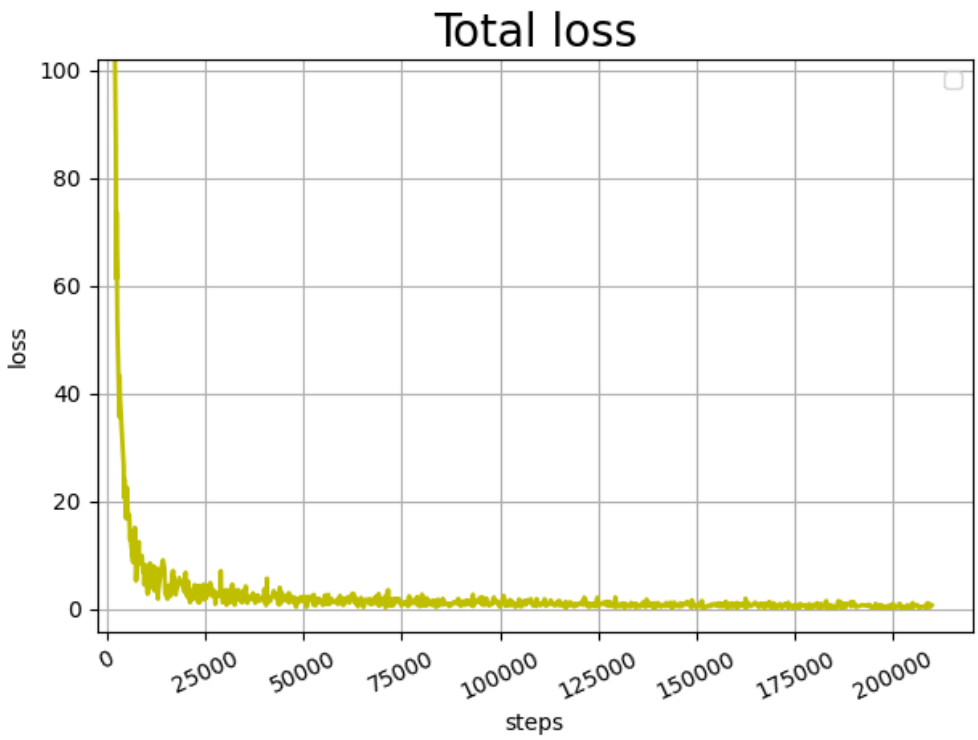
Παρατηρείται ότι ενώ υπάρχουν ορισμένες διακυμάνσεις στις τιμές των Confidence loss (Εικόνα 51), Prob loss (Εικόνα 53) και Total loss (Εικόνα 54), οι τιμές τους τείνουν στο μηδέν από τα αρχικά στάδια της εκπαίδευσης, ενώ το IoU loss (Εικόνα 52) παρουσιάζει έντονες διακυμάνσεις μέχρι και το τέλος της εκπαίδευσης.



Εικόνα 52. IoU loss σε σχέση με τον αριθμό των βημάτων εκπαίδευσης



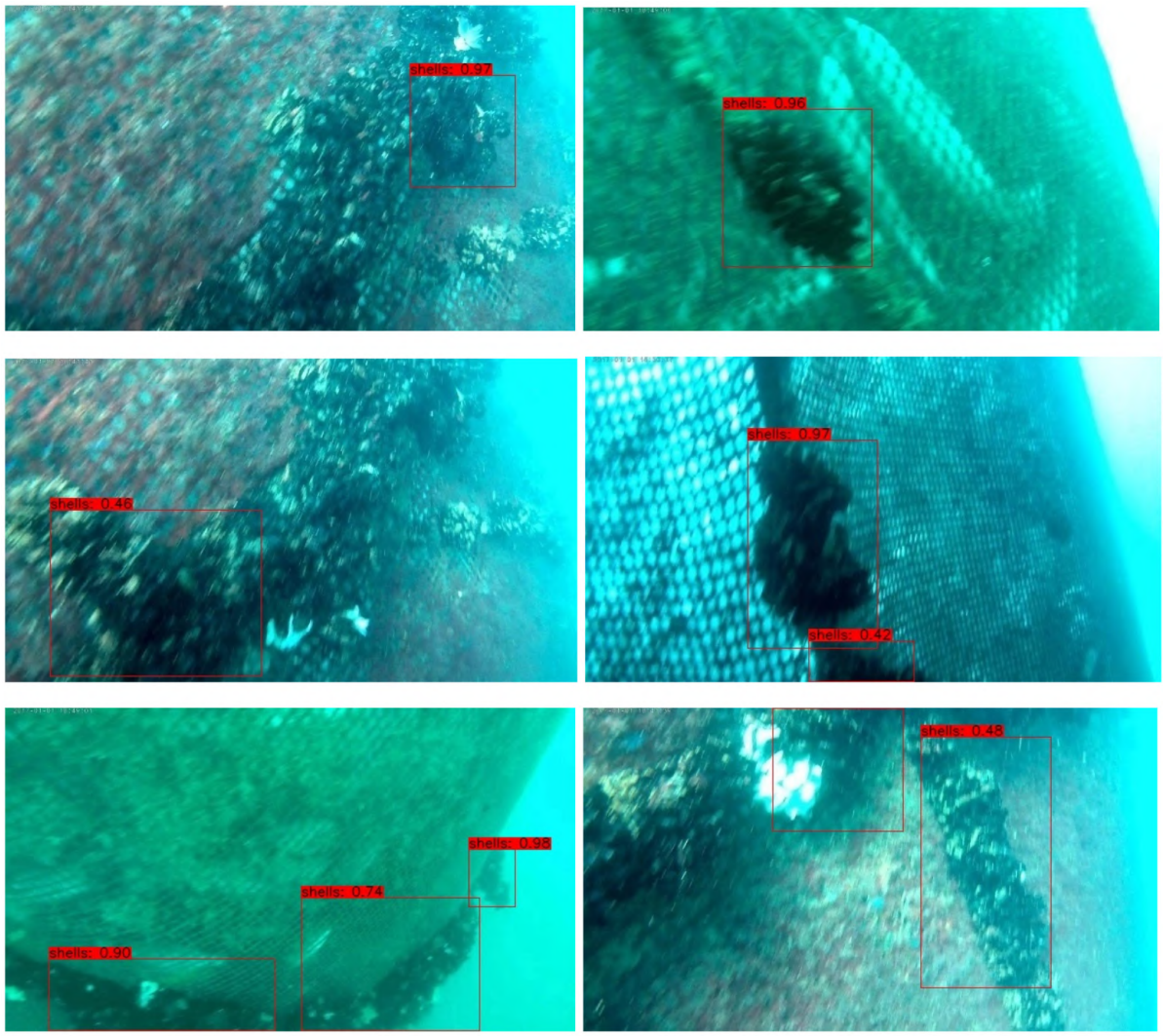
Εικόνα 53. Prob loss σε σχέση με τον αριθμό των βημάτων εκπαίδευσης



Εικόνα 54. Total loss σε σχέση με τον αριθμό των βημάτων εκπαίδευσης

Όπως και στο Confidence loss, το Total loss εμφάνισε ασυνήθιστα υψηλές τιμές στα αρχικά στάδια εκπαίδευσης, επομένως στο γράφημα του Total loss (Εικόνα 54) έγινε περικοπή έτσι ώστε να απεικονίζεται μέγιστη τιμή σφάλματος 100. Με αυτό τον τρόπο μπορούν και διακρίνονται οι διακυμάνσεις που υπάρχουν στο σφάλμα κατά την εκπαίδευση.

Στην Εικόνα 55 φαίνονται τα αποτελέσματα του evaluation του YOLO. Στη συγκεκριμένη περίπτωση δεν υπάρχουν εικόνες αναφοράς για σύγκριση λόγω της διαφορετικής διαδικασίας εκπαίδευσης και evaluation από τους προηγούμενους αλγορίθμους. Είναι εμφανές ότι ο YOLO αναγνωρίζει επιτυχώς αντικείμενα με μεγάλη αντίθεση, ενώ υπάρχουν πολλές περιοχές με χαμηλή αντίθεση στις οποίες δεν έχει ανιχνευθεί επιτυχώς αντικείμενο.



Εικόνα 55. Αποτελέσματα evaluation του YOLO

3.8 Σύγκριση των αποτελεσμάτων

Με την εξαγωγή των δεδομένων evaluation από το Tensorboard σε συνδυασμό με την χρήση των κατάλληλων python scripts από τα εικονικά περιβάλλοντα, έγινε η εξαγωγή των δεδομένων απόδοσης των δικτύων που εκπαιδεύτηκαν. Οι σημαντικότερες μετρήσεις επιδόσεων είναι το mean Average Precision (mAP) και Average Recall (AR). mAP είναι η ακρίβεια με την οποία ανιχνεύθηκαν τα αντικείμενα και ορίζεται ως εξής:

$$mAP = \frac{True\ positives}{True\ positives + False\ positives}$$

Τύπος 5. Υπολογισμός του mean Average Precision

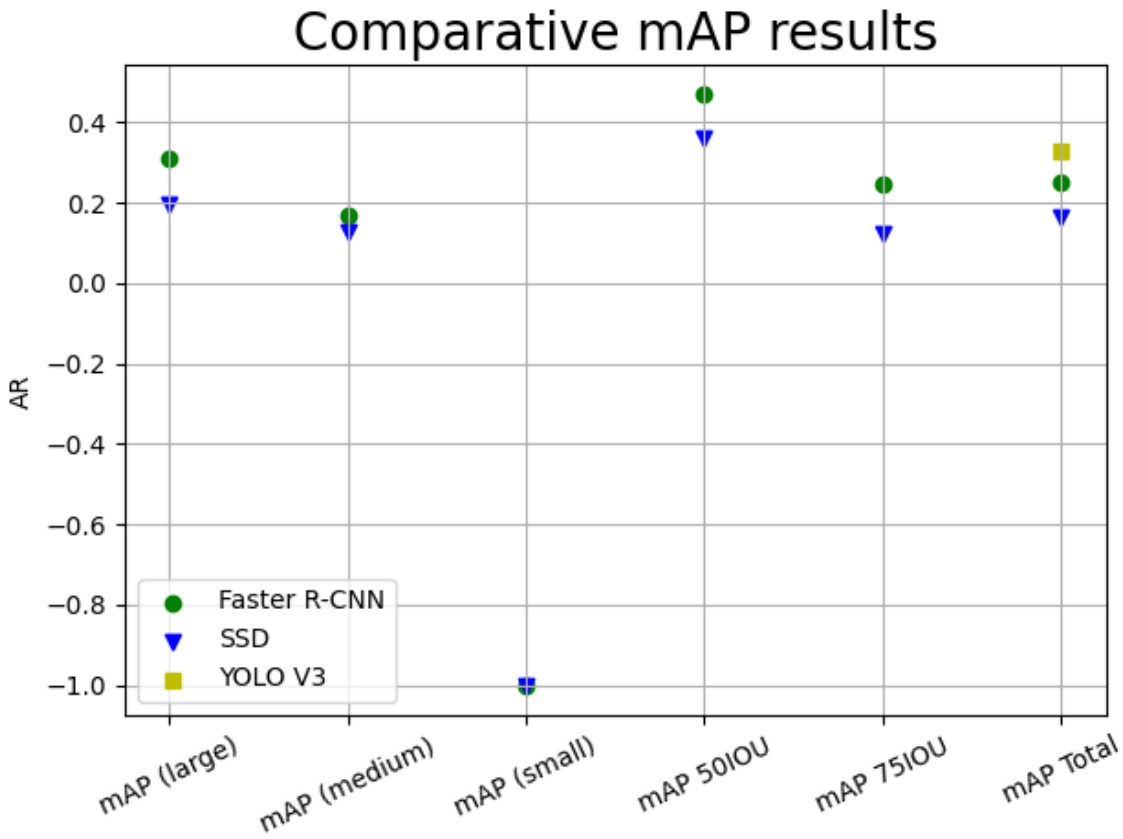
Ουσιαστικά, το mAP μετράει πόσο στοχευμένα το μοντέλο ανίχνευσε τα αντικείμενα που πρέπει. Το AR μετράει το ποσοστό των ανιχνευμένων αντικειμένων σε σχέση με όσα έπρεπε να ανιχνευθούν και υπολογίζεται ως εξής:

$$AR = \frac{True\ positives}{True\ positives + False\ negatives}$$

Τύπος 6. Υπολογισμός του Average Recall

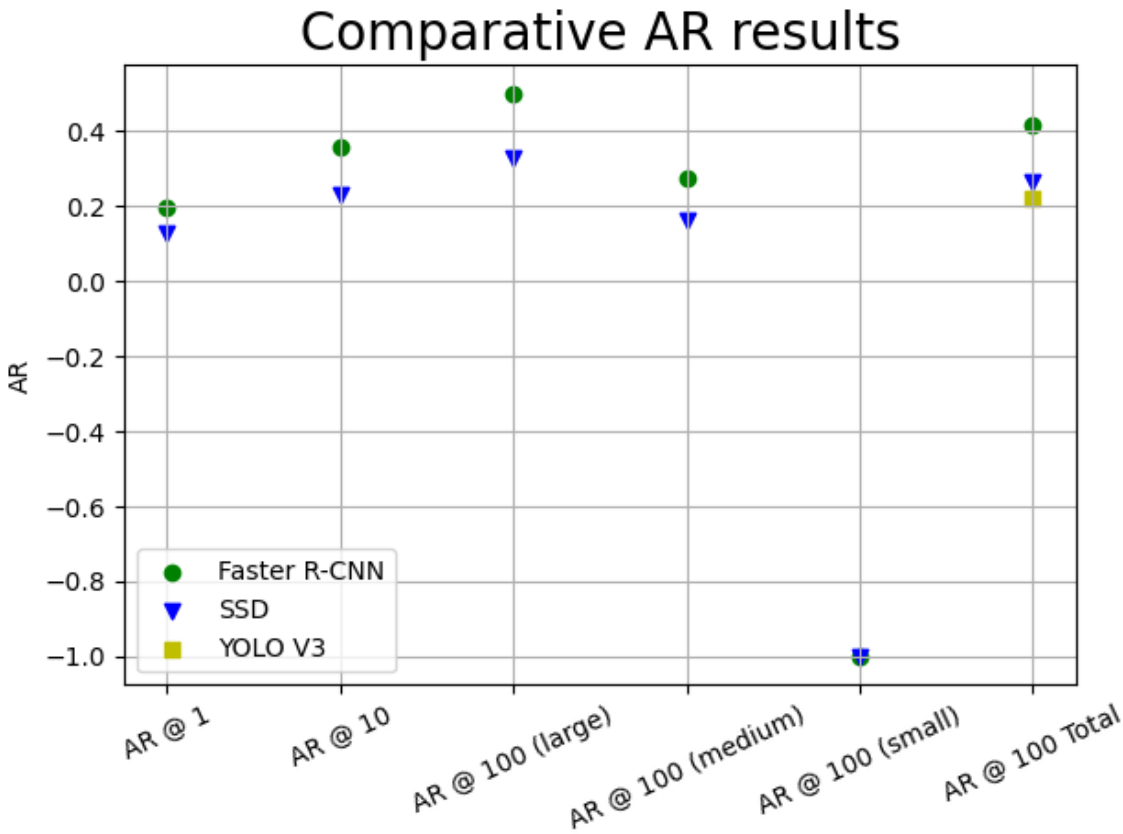
Οι τιμές mAP και AR υπολογίζονται με τη βοήθεια των εικόνων αναφοράς. Εφαρμόζεται η διαδικασία Intersection over Union μεταξύ των Bounding Boxes που εντοπίστηκαν και των Bounding Boxes του Dataset και όπου το IoU είναι μεγαλύτερο του 50%, υπολογίζονται τα mAP και AR.

Το evaluation των Faster R-CNN και SSD επέστρεψε τιμές mAP και AR για IoU 50% και 75% όπως επίσης και σε διαφορετικές κλίμακες (small, medium και large). Ο YOLO λόγω της διαφορετικής διαδικασίας evaluation, δεν έχει εξάγει λεπτομερή δεδομένα για το mAP και το AR αλλά μόνο τις τελικές τιμές οι οποίες φαίνονται με κίτρινο χρώμα στα γραφήματα των εικόνων 56 και 57. Επιπλέον, για μικρές κλίμακες οι Faster R-CNN και SSD έδιναν σταθερά τιμή -1 τόσο για mAP όσο και για AR, πράγμα που δηλώνει ότι δεν υπήρχαν μικρές κλίμακες για να παραχθούν τα συγκεκριμένα δεδομένα.



Εικόνα 56. Συγκριτικά αποτελέσματα ακρίβειας των αλγορίθμων

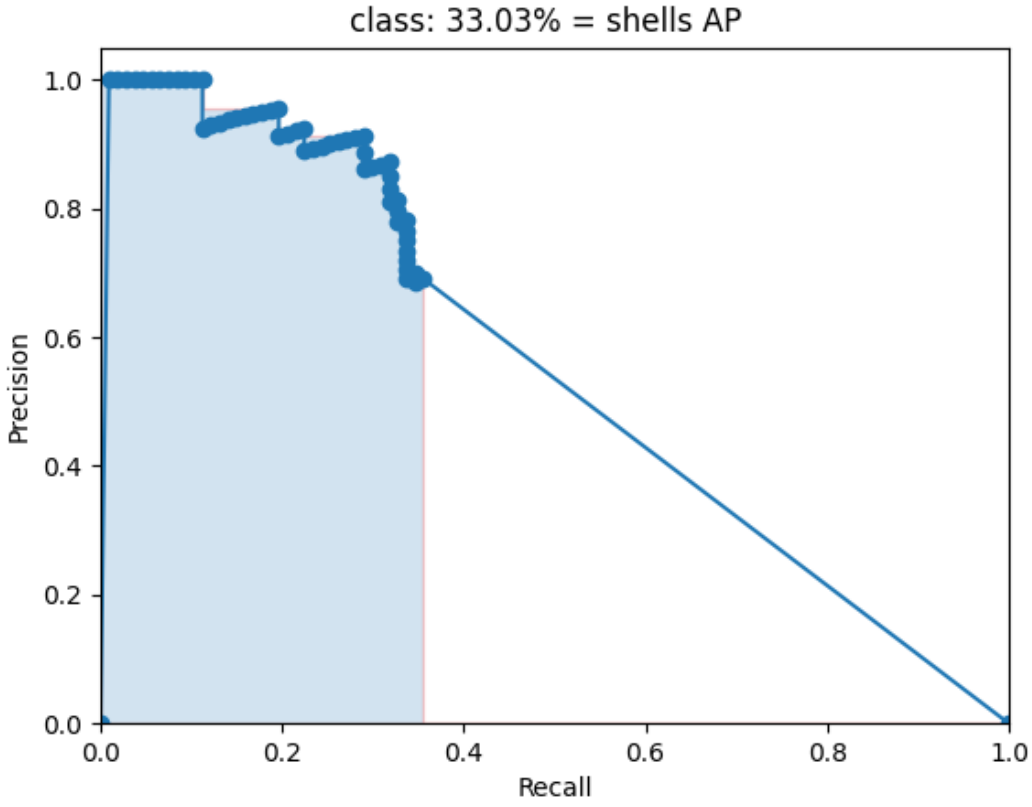
Παρατηρείται ότι ο Faster R-CNN παρουσιάζει σταθερά καλύτερη ακρίβεια σε σχέση με τον SSD, αλλά η υψηλότερη τελική τιμή ακρίβειας, είναι από τον αλγόριθμο YOLOv3 με 33.03%. Οι τελικές τιμές ακρίβειας των Faster R-CNN και SSD ήταν 24.9% και 16.5% αντίστοιχα.



Εικόνα 57. Συγκριτικά αποτελέσματα Average Recall των αλγορίθμων

Στο γράφημα του Average Recall παρατηρείται ότι ο Faster R-CNN είναι σημαντικά καλύτερος από τον SSD και στην τελική τιμή με *minimum detections* = 100, να ξεπερνά και τον YOLOv3. Συγκεκριμένα οι τελικές τιμές του AR τους ήταν 26.6% για τον SSD, 41.7% για τον Faster R-CNN και 22% για τον YOLOv3.

Ενδεικτικά, από το evaluation του YOLO προέκυψαν οι τιμές precision και recall για κάθε αντικείμενο του evaluation Dataset επομένως δημιουργήθηκε το γράφημα της Εικόνας 58, που δείχνει το mAP σε σχέση με το AR του YOLO.

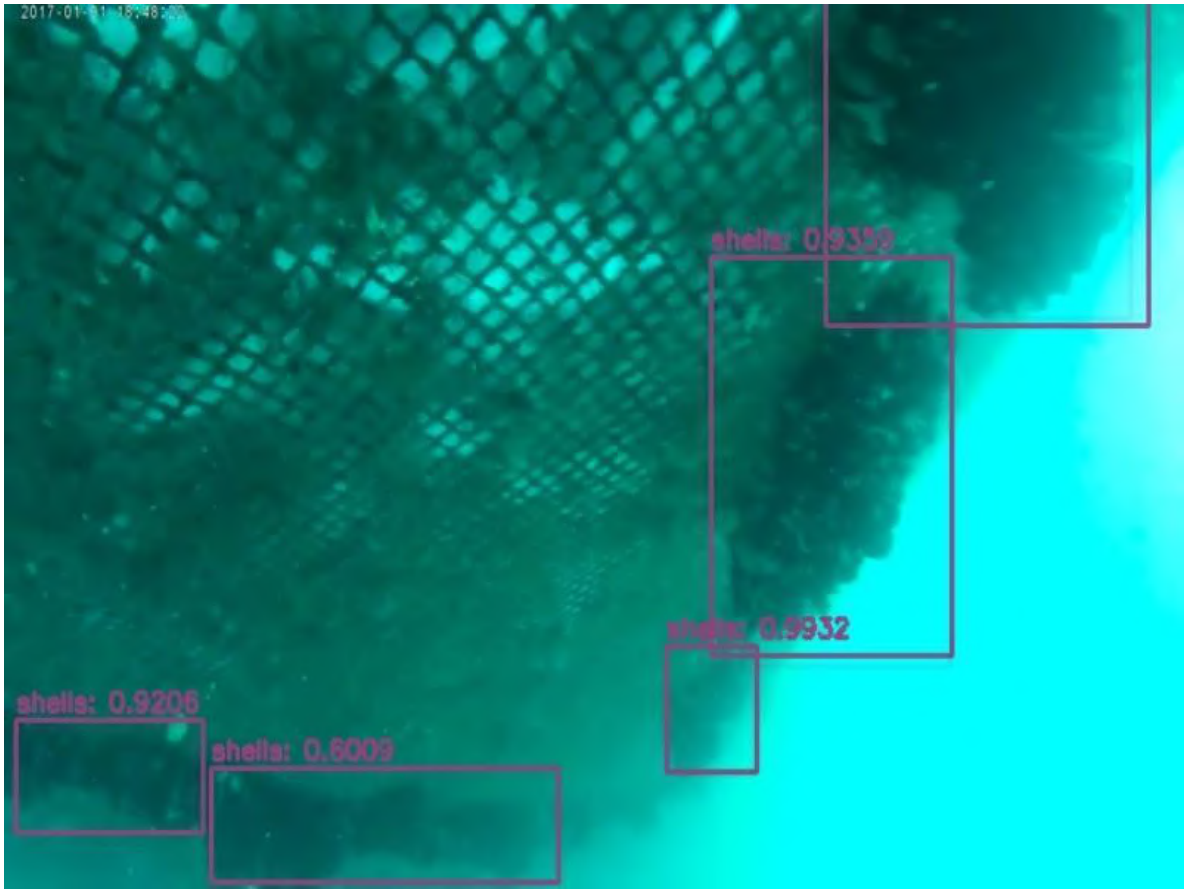


Εικόνα 58. mean Average Precision σε σχέση με το Recall για τον YOLO

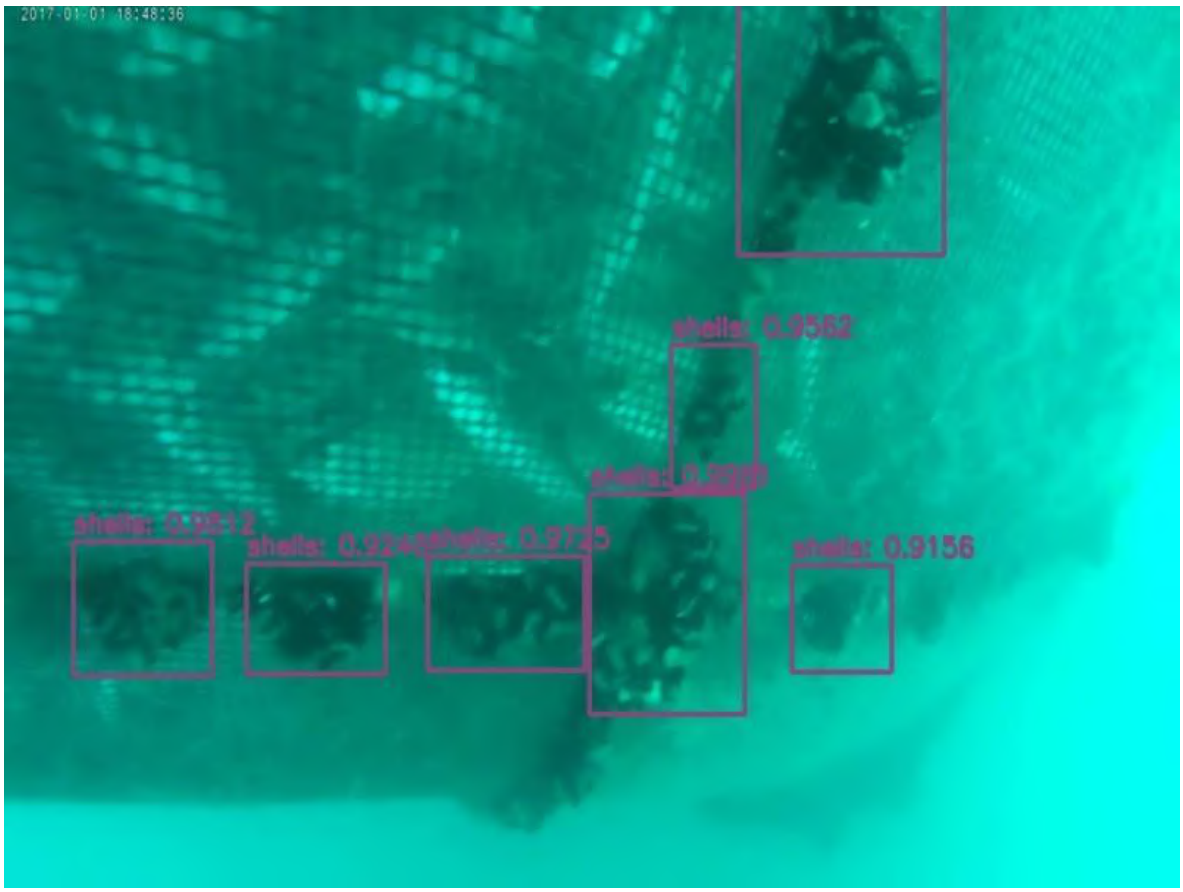
Για την εφαρμογή του κάθε αλγορίθμου στα αρχικά βίντεο των διχτυών Ιχθυοκαλλιέργειας, χρησιμοποιήθηκε ένα python script το οποίο χρησιμοποιεί το εκπαιδευμένο μοντέλο και εφαρμόζει τα βήματα που απαιτούνται για την ανίχνευση αντικειμένων.

Αρχικά για τον YOLO, τα πρώτα βήματα περιλαμβάνουν την αρχικοποίηση και το διάβασμα των layers του δικτύου με χρήση του .weights αρχείου που δημιουργήθηκε στην εκπαίδευση. Στη συνέχεια, λαμβάνεται ένα καρέ του βίντεο το οποίο χρησιμοποιείται ως είσοδος στο πρώτο layer του δικτύου. Έπειτα, η έξοδος του κάθε layer χρησιμοποιείται σαν είσοδος στο επόμενο layer μέχρι να καταλήξει το σήμα στο τελικό layer. Η έξοδος του τελικού layer αποτελείται, όπως προβλέπει η θεωρία, από τις προβλέψεις των Bounding Boxes με τις συντεταγμένες του κέντρου και τις διαστάσεις τους, καθώς και από τις προβλέψεις κλάσεων για κάθε Bounding Box. Στη συγκεκριμένη περίπτωση, η ακρίβεια των προβλέψεων των κλάσεων δεν μας απασχολεί, καθώς υπάρχει μόνο μία κλάση για την οποία εκπαιδεύτηκε το μοντέλο. Στο τέλος, γίνεται ένα thresholding στις τελικές προβλέψεις με βάση το Confidence Value το οποίο επίσης εμφανίζεται στην έξοδο του δικτύου και εμφανίζονται τα κατάλληλα Bounding Boxes. Η διαδικασία αυτή

επαναλαμβάνεται για κάθε καρέ του βίντεο και εμφανίζεται στην οθόνη το κάθε καρέ μαζί με τα Bounding Boxes, και τις κλάσεις με το Confidence Value που ανιχνεύθηκαν. Οι Εικόνες 59 και 60 αποτελούν στιγμιότυπα του βίντεο που παράχθηκε μετά την ανίχνευση. Φαίνεται ότι όλα τα ευδιάκριτα ελαττώματα ανιχνεύθηκαν επιτυχώς με μεγάλο Confidence Value.

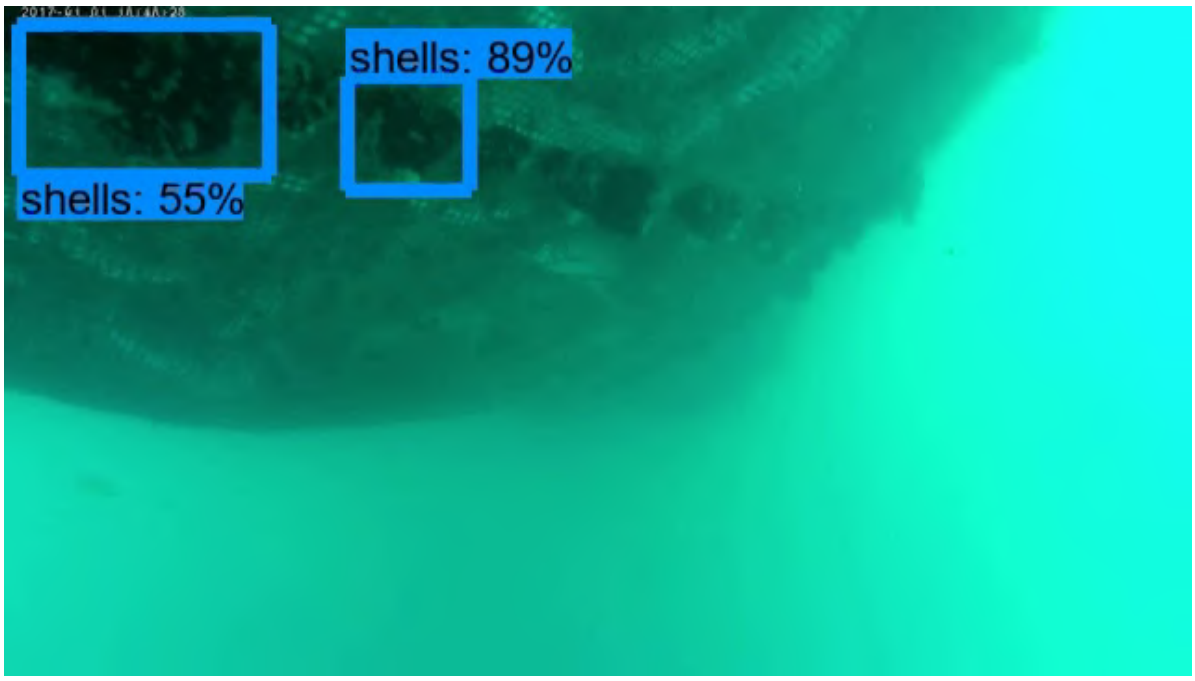


Εικόνα 59. Στιγμιότυπο 1 δοκιμής του YOLO στα βίντεο Ιχθυοκαλλιέργειας

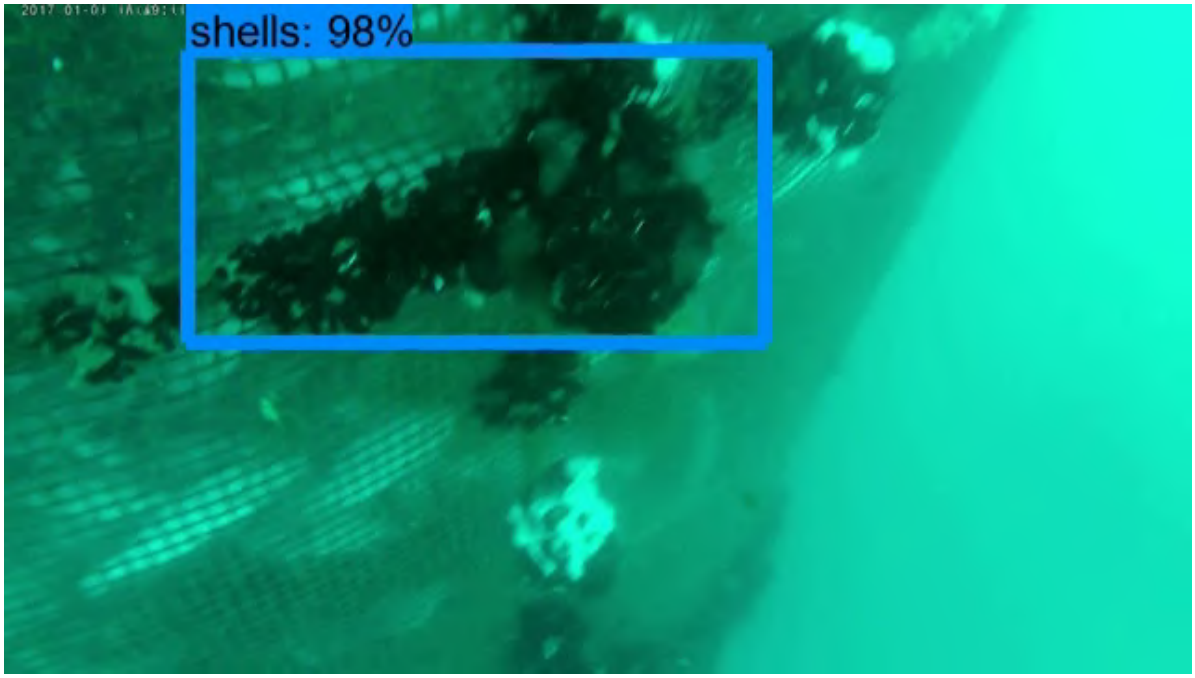


Εικόνα 60. Στιγμιότυπο 2 δοκιμής του YOLO στα βίντεο Ιχθυοκαλλιέργειας

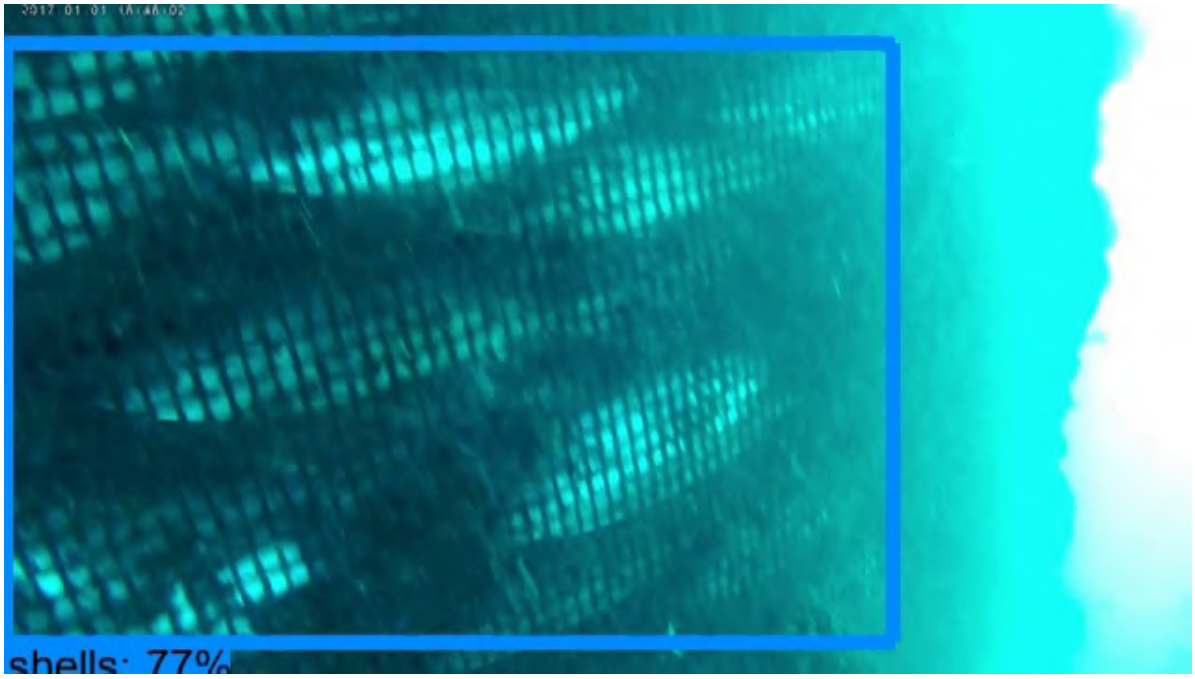
Για την εφαρμογή του SSD, χρησιμοποιήθηκε ένα αντίστοιχο python script το οποίο αρχικά εισάγει το εκπαιδευμένο μοντέλο με μορφή αρχείου .pb καθώς και το αρχείο που ορίζει τα labels για το κάθε Bounding Box. Το .pb αρχείο δημιουργήθηκε με τη χρήση του τελευταίου checkpoint της εκπαίδευσης σε συνδυασμό με το configuration αρχείο του SSD. Στη συνέχεια, εισάγεται το βίντεο από το οποίο γίνεται λήψη ενός καρέ. Έπειτα το καρέ προσαρμόζεται σε μέγεθος σύμφωνα με τις διαστάσεις που υποστηρίζονται από το μοντέλο, και γίνεται η ανίχνευση των αντικειμένων μέσα από αυτό. Γίνεται προσπέλαση του δικτύου, και αποθηκεύεται ο αριθμός των Bounding Boxes, τα scores (Confidence Values) για το κάθε Bounding Box, καθώς και οι κλάσεις που ανιχνεύθηκαν. Έπειτα εμφανίζεται το καρέ στην οθόνη μαζί με τα Bounding Boxes και τα scores για την κάθε ανίχνευση. Στις Εικόνες 61 και 62 φαίνονται οι ανιχνεύσεις που έγιναν με τον SSD. Διακρίνονται χαμηλότερα Confidence Values και αποτυχία ανίχνευσης όλων των ελαττωμάτων όπως φάνηκε και προηγουμένως από τα αποτελέσματα του evaluation στις τιμές mAP και AR.



Εικόνα 61. Στιγμιότυπο 1 δοκιμής του SSD στα βίντεο Ιχθυοκαλλιέργειας



Εικόνα 62. Στιγμιότυπο 2 δοκιμής του SSD στα βίντεο Ιχθυοκαλλιέργειας

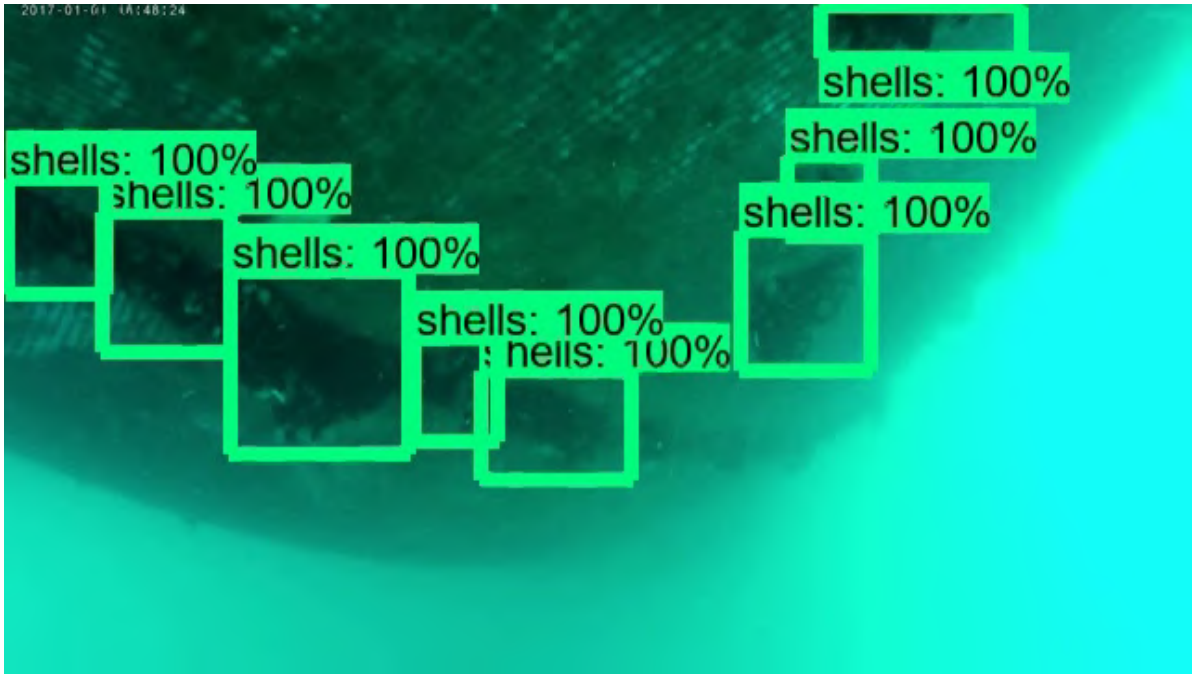


Εικόνα 63. Στιγμιότυπο 3 δοκιμής του SSD στα βίντεο Ιχθυοκαλλιέργειας

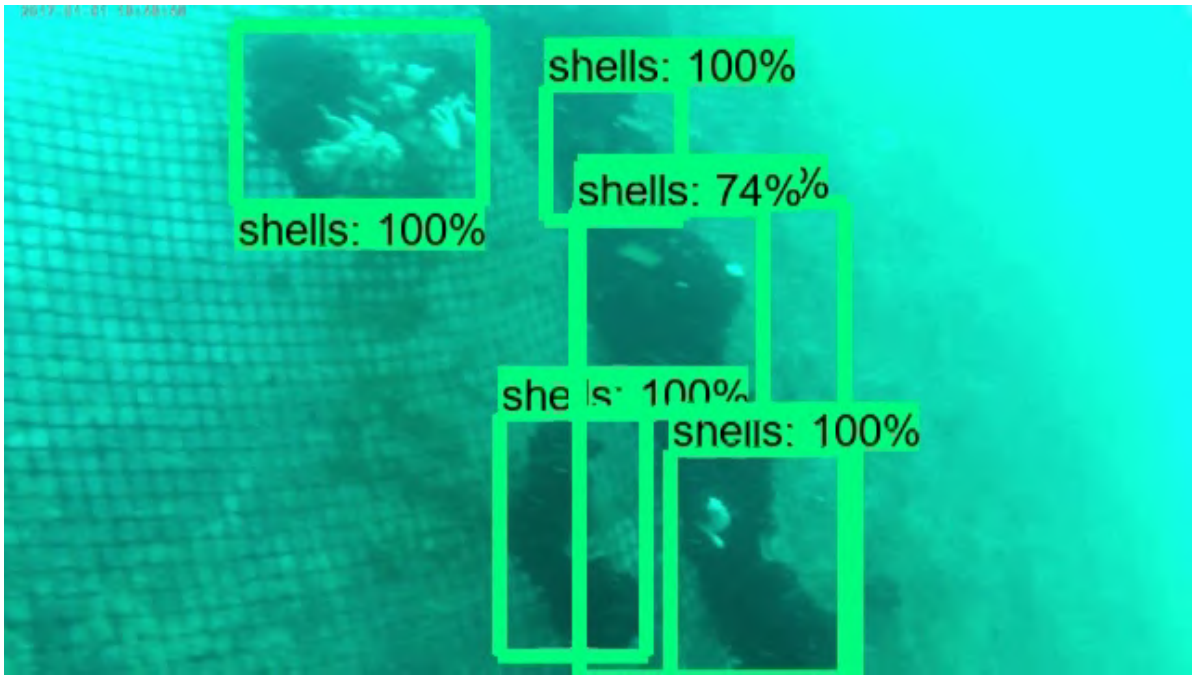
Συγκρίνοντας ποιοτικά τα αποτελέσματα της εφαρμογής των δύο αλγορίθμων στα συγκεκριμένα βίντεο, παρατηρήθηκαν πολλές false positive και false negative ανιχνεύσεις στον SSD σε σχέση με τον YOLO. Επιπλέον, ο SSD ανιχνεύει αισθητά λιγότερα ελαττώματα στα δίχτυα, ενώ τα bounding boxes που σχηματίζονται, συχνά περιλαμβάνουν όλη την εικόνα όπως φαίνεται στην Εικόνα 63.

Το ίδιο python script που χρησιμοποιήθηκε για τον SSD, χρησιμοποιήθηκε και για την ανίχνευση ελαττωμάτων μέσω του Faster R-CNN με τη διαφορά ότι το μοντέλο, που εισάγεται με τη μορφή *frozen_model.pb,* ήταν αυτό που προέκυψε από την εκπαίδευση του Faster R-CNN. Στις Εικόνες 64 - 66 φαίνονται στιγμιότυπα του βίντεο που προέκυψε από τη χρήση του Faster R-CNN. Παρατηρείται ότι ο συγκεκριμένος αλγόριθμος αναγνωρίζει με μεγάλη επιτυχία μικρές ομάδες ελαττωμάτων, σε αντίθεση με τον SSD όπου συνήθως εντοπίζει ως ελάττωμα ένα μεγάλο μέρος της εικόνας και δεν εστιάζει στα συγκεκριμένα σημεία της εικόνας που εμφανίζουν ελαττώματα.

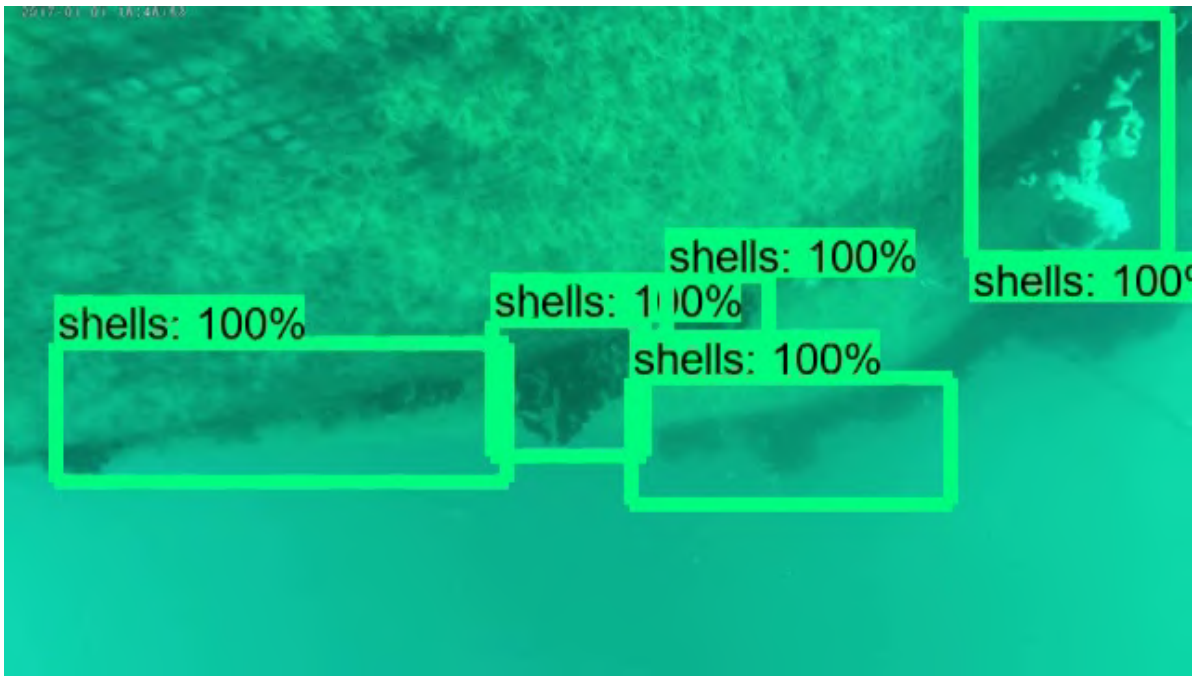
Αξίζει να σημειωθεί ότι κατά τη διάρκεια της ανίχνευσης μέσω του python script που προαναφέρθηκε, ο ρυθμός εμφάνισης των καρέ του βίντεο ήταν διαφορετικός σε κάθε αλγόριθμο. Συγκεκριμένα, με τον Faster R-CNN, τα καρέ του βίντεο κατά την ανίχνευση εμφανίζονταν με αισθητά πιο αργό ρυθμό σε σχέση με το ρυθμό εμφάνισης των SSD και YOLO. Το αποτέλεσμα αυτό είναι αναμενόμενο σύμφωνα με τη θεωρία, που αναφέρει ότι ο Faster R-CNN είναι πιο αργός από τους SSD και YOLO.



Εικόνα 64. Στιγμιότυπο 1 δοκιμής του Faster R-CNN στα βίντεο Ιχθυοκαλλιέργειας



Εικόνα 65. Στιγμιότυπο 2 δοκιμής του Faster R-CNN στα βίντεο Ιχθυοκαλλιέργειας



Εικόνα 66. Στιγμιότυπο 3 δοκιμής του Faster R-CNN στα βίντεο Ιχθυοκαλλιέργειας

Τα αποτελέσματα χρήσης των εκπαιδευμένων μοντέλων στα βίντεο των διχτυών Ιχθυοκαλλιέργειας συμφωνούν με τα αποτελέσματα του evaluation που έγινε στις εκπαιδεύσεις τους. Συγκεκριμένα, οι Faster R-CNN και YOLO φαίνεται να αναγνωρίζουν με μεγαλύτερη ακρίβεια τα ελαττώματα που εμφανίζονται στα βίντεο σε σχέση με τον SSD, γεγονός που επιβεβαιώνεται με τις διαφορές τους στις τιμές των mAP και AR. Επιπλέον, ο Faster R-CNN επιτυγχάνει την ανίχνευση με πιο αργό ρυθμό από τους SSD και YOLO όπως φαίνεται από τον ρυθμό εμφάνισης των καρέ των βίντεο, γεγονός που επιβεβαιώνεται από τη θεωρία.

4 *Συμπεράσματα - Προτάσεις μελλοντικής έρευνας*

4.1 Συμπεράσματα

Η χρήση Μηχανικής Μάθησης μέσω Νευρωνικών Δικτύων αποδεικνύεται επιτυχημένη στην ανίχνευση ελαττωμάτων σε δίχτυα Ιχθυοκαλλιέργειας. Τα προβλήματα που παρουσιάζονται στη συγκεκριμένη περίπτωση χρήσης Νευρωνικών Δικτύων όπως η αλλοίωση των χρωμάτων, το ακανόνιστο σχήμα και το σκούρο χρώμα των ελαττωμάτων καθώς και η μειωμένη ορατότητα, δεν είναι αρκετά για να εμποδίσουν την επιτυχή ανίχνευση των ελαττωμάτων.

Το κύριο πρόβλημα που παρουσιάστηκε κατά την εκπόνηση της συγκεκριμένης εργασίας ήταν η έλλειψη εφαρμογής των συγκεκριμένων αλγορίθμων σε παρόμοιες περιπτώσεις χρήσης με τα δίχτυα Ιχθυοκαλλιέργειας. Το γεγονός αυτό έκανε δύσκολη τη προσαρμογή των παραδειγμάτων και των τεχνικών υλοποίησης των αλγορίθμων στη συγκεκριμένη περίπτωση χρήσης. Επιπλέον, η δημιουργία του Dataset αποδείχθηκε χρονοβόρα και προβληματική καθώς βασίζεται στον ανθρώπινο παράγοντα για την εκτίμηση της θέσης και του αριθμού των ελαττωμάτων κατά τη διαδικασία του Labeling. Επιπροσθέτως, αλγόριθμοι που επιλέχθηκαν έχουν σχεδιαστεί για να εφαρμόζονται με χρήση ξεχωριστών εργαλείων, πράγμα που δυσκολεύει την εκπόνηση των πειραμάτων με πανομοιότυπο τρόπο για κάθε αλγόριθμο ξεχωριστά.

Από τα δεδομένα απόδοσης των αλγορίθμων που εκπαιδεύτηκαν και δοκιμάστηκαν, την υψηλότερη ακρίβεια είχε ο YOLOv3 με 33.03% ενώ το υψηλότερο Recall το είχε ο Faster R-CNN με 41.7%. Επομένως, για την περίπτωση ανίχνευσης ελαττωμάτων σε δίχτυα Ιχθυοκαλλιέργειας με το συγκεκριμένο Dataset, προτείνεται η χρήση του Faster R-CNN, αν

ο αριθμός των ανιχνεύσεων είναι η σημαντικότερη παράμετρος, ενώ αν η ακρίβεια και η ταχύτητα είναι σημαντικότερα, προτείνεται ο YOLOv3. Ο SSD δεν εμφάνισε κάποια σημαντικά προτερήματα σε σχέση με τους άλλους δύο αλγορίθμους επομένως δεν προτείνεται για τη συγκεκριμένη περίπτωση χρήσης.

Με τη χρήση των αλγορίθμων στην ανίχνευση ελαττωμάτων στα βίντεο Ιχθυοκαλλιέργειας αποδεικνύεται ότι μπορούν μελλοντικά να χρησιμοποιηθούν από ένα AUV προκειμένου να χαρτογραφήσουν τις προβληματικές περιοχές των διχτυών σε μία Ιχθυοκαλλιέργεια. Ο περιοριστικός παράγοντας για τη χρήση τους από ένα AUV, είναι η επεξεργαστική ισχύς του, καθώς οι συγκεκριμένοι αλγόριθμοι χρησιμοποιούσαν μεγάλο ποσοστό της μνήμης μιας ισχυρής κάρτας γραφικών.

4.2 Προτάσεις μελλοντικής έρευνας

Για την εκπαίδευση των νευρωνικών δικτύων είναι πολύ σημαντική η χρήση ενός σωστά διαμορφωμένου Dataset με μεγάλο αριθμό εικόνων και με ακριβές Labeling. Συγκεκριμένα, μπορούν να παραχθούν πιο ακριβή μοντέλα και στους τρεις αλγορίθμους, χρησιμοποιώντας ένα Dataset με χιλιάδες εικόνες αντί για εκατοντάδες. Επιπλέον, είναι σημαντικό να εξαλειφθεί ο ανθρώπινος παράγοντας στην κατηγοριοποίηση των εικόνων. Ειδικότερα στη συγκεκριμένη περίπτωση των ελαττωμάτων σε Ιχθυοκαλλιέργειες, τα ελαττώματα έχουν ακανόνιστο σχήμα και δεν υφίστανται αντικειμενικά κριτήρια για τη σωστή οριοθέτηση τους σε μία εικόνα. Αποδείχθηκε επίσης ιδιαίτερα δύσκολο να χαρακτηριστεί ένα αντικείμενο ως ελάττωμα προς αναγνώριση σε μία εικόνα λόγω της φύσεως των εικόνων που παρήχθησαν από βίντεο με έντονη κίνηση με αποτέλεσμα να εμφανίζεται θολό το περιεχόμενο τους. Επομένως, μελλοντικά χρειάζεται να αναπτυχθεί ένας πιο αντικειμενικός τρόπος καθορισμού των ελαττωμάτων στα δίχτυα Ιχθυοκαλλιέργειας για τη δημιουργία ακριβέστερων Datasets.

Η χρήση διαφορετικών εργαλείων και παραμέτρων για την εκπαίδευση και την αξιολόγηση των Νευρωνικών Δικτύων δυσκολεύει τον εντοπισμό των συγκεκριμένων παραμέτρων που επηρεάζουν περισσότερο τα αποτελέσματα για τον κάθε αλγόριθμο. Επομένως, προτείνεται η περαιτέρω έρευνα για εφαρμογές των συγκεκριμένων αλγορίθμων με παρόμοια εργαλεία προκειμένου να γίνουν πιο ξεκάθαρες συγκρίσεις μεταξύ τους και να ρυθμιστούν οι παράμετροι εκπαίδευσης τους έτσι ώστε να παραχθούν τα καλύτερα δυνατά αποτελέσματα.

Τέλος, προτείνεται η χρήση ισχυρότερου υλισμικού για την εκπαίδευση και τη δοκιμή των Νευρωνικών Δικτύων, καθώς η χρήση ενός ισχυρού υπολογιστικού συστήματος μπορεί να βοηθήσει στη γρηγορότερη εκπαίδευση και βελτιστοποίηση των Νευρωνικών Δικτύων.

Βιβλιογραφία

1. Flemming, H. C. (2002). Biofouling in water systems–cases, causes and countermeasures. *Applied microbiology and biotechnology*, *59*(6), 629-640.
2. Fitridge, I., Dempster, T., Guenther, J., & De Nys, R. (2012). The impact and control of biofouling in marine aquaculture: a review. *Biofouling*, *28*(7), 649-669.
3. Mahalik, N., & Kim, K. (2014). Aquaculture monitoring and control systems for seaweed and fish farming.
4. Bannister, J., Sievers, M., Bush, F., & Bloecher, N. (2019). Biofouling in marine aquaculture: a review of recent research and developments. *Biofouling*, *35*(6), 631-648.
5. Hodson, S. L., Lewis, T. E., & Burkea, C. M. (1997). Biofouling of fish-cage netting: efficacy and problems of in situ cleaning. *Aquaculture*, *152*(1-4), 77-90.
6. Borović, B., Vasilijević, A., & Kuljača, O. (2011). Potentials of using underwater robotics for fishing and fish farming. *on the Theory of Fishing Gears and Related Marine Systems*, 95.
7. Zou, Z., Shi, Z., Guo, Y., & Ye, J. (2019). Object detection in 20 years: A survey. *arXiv preprint arXiv:1905.05055*.
8. Papageorgiou, C., & Poggio, T. (1999, October). Trainable pedestrian detection. In *Proceedings 1999 International Conference on Image Processing (Cat. 99CH36348)* (Vol. 4, pp. 35-39). IEEE.
9. Lindeberg, T. (1998). Feature detection with automatic scale selection. *International journal of computer vision*, *30*(2), 79-116.
10. Ding, L., & Goshtasby, A. (2001). On the Canny edge detector. *Pattern Recognition*, *34*(3), 721-725.
11. Atherton, T. J., & Kerbyson, D. J. (1999). Size invariant circle detection. *Image and Vision computing*, *17*(11), 795-803.
12. Zhu, W., Liang, S., Wei, Y., & Sun, J. (2014). Saliency optimization from robust background detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2814-2821).
13. Lindeberg, T. (2012). Scale invariant feature transform.
14. Bay, H., Tuytelaars, T., & Van Gool, L. (2006, May). Surf: Speeded up robust features. In *European conference on computer vision* (pp. 404-417). Springer, Berlin, Heidelberg.
15. Lu, X., Li, Q., Li, B., & Yan, J. (2020). MimicDet: Bridging the Gap Between One-Stage and Two-Stage Object Detection. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16* (pp. 541-557). Springer International Publishing.
16. Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 779-788).
17. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016, October). Ssd: Single shot multibox detector. In *European conference on computer vision* (pp. 21-37). Springer, Cham.

18. Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 580-587).
19. Ren, S., He, K., Girshick, R., & Sun, J. (2016). Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, *39*(6), 1137-1149.
20. Pang, J., Chen, K., Shi, J., Feng, H., Ouyang, W., & Lin, D. (2019). Libra r-cnn: Towards balanced learning for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 821-830).
21. Kleene, S. C. (2016). *Representation of events in nerve nets and finite automata* (pp. 3-42). Princeton University Press.
22. Albawi, S., Mohammed, T. A., & Al-Zawi, S. (2017, August). Understanding of a convolutional neural network. In *2017 International Conference on Engineering and Technology (ICET)* (pp. 1-6). Ieee.
23. Liu, L., Shen, C., & Van Den Hengel, A. (2015). The treasure beneath convolutional layers: Cross-convolutional-layer pooling for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 4749-4757).
24. Yu, D., Wang, H., Chen, P., & Wei, Z. (2014, October). Mixed pooling for convolutional neural networks. In *International conference on rough sets and knowledge technology* (pp. 364-375). Springer, Cham.
25. Basha, S. S., Dubey, S. R., Pulabaigari, V., & Mukherjee, S. (2020). Impact of fully connected layers on performance of convolutional neural networks for image classification. *Neurocomputing*, *378*, 112-119.
26. Hecht-Nielsen, R. (1992). Theory of the backpropagation neural network. In *Neural networks for perception* (pp. 65-93). Academic Press.
27. Qin, Z., Kim, D., & Gedeon, T. (2019). Rethinking softmax with cross-entropy: Neural network classifier as mutual information estimator. *arXiv preprint arXiv:1911.10688*.
28. LeCun, Y. (2015). LeNet-5, convolutional neural networks. *URL: http://yann. lecun. com/exdb/lenet*, *20*(5), 14.
29. Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, *25*, 1097-1105.
30. Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.
31. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1-9).
32. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).
33. Zeiler, M. D., & Fergus, R. (2014, September). Visualizing and understanding convolutional networks. In *European conference on computer vision* (pp. 818-833). Springer, Cham.
34. Hecht-Nielsen, R. (1992). Theory of the backpropagation neural network. In *Neural networks for perception* (pp. 65-93). Academic Press.
35. Zhang, Z., & Sabuncu, M. R. (2018, January). Generalized cross entropy loss for training deep neural networks with noisy labels. In *32nd Conference on Neural Information Processing Systems (NeurIPS)*.
36. Ruder, S. (2016). An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*.
37. Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., & Tang, P. T. P. (2016). On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*.

38. Felzenszwalb, P., McAllester, D., & Ramanan, D. (2008, June). A discriminatively trained, multiscale, deformable part model. In *2008 IEEE conference on computer vision and pattern recognition* (pp. 1-8). Ieee.
39. Uijlings, J. R., Van De Sande, K. E., Gevers, T., & Smeulders, A. W. (2013). Selective search for object recognition. *International journal of computer vision*, *104*(2), 154-171.
40. He, K., Zhang, X., Ren, S., & Sun, J. (2015). Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, *37*(9), 1904-1916.
41. Erhan, D., Szegedy, C., Toshev, A., & Anguelov, D. (2014). Scalable object detection using deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2147-2154).
42. Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., & LeCun, Y. (2013). Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*.
43. Redmon, J., & Farhadi, A. (2017). YOLO9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 7263-7271).
44. Redmon, J., & Farhadi, A. (2018). Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*.
45. Bochkovskiy, A., Wang, C. Y., & Liao, H. Y. M. (2020). Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*.
46. Hosang, J., Benenson, R., & Schiele, B. (2017). Learning non-maximum suppression. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4507-4515).
47. Bjørlo-Larsen, V., & Skeide, E. S. (2019). *Damage detection in fish farming nets using video analysis* (Bachelor's thesis, NTNU).
48. Chin, C. S., Si, J., Clare, A. S., & Ma, M. (2017). Intelligent image recognition system for marine fouling using Softmax transfer learning and deep convolutional neural networks. *Complexity*, *2017*.
49. Dong, N., Zhao, L., Wu, C. H., & Chang, J. F. (2020). Inception v3 based cervical cell classification combined with artificially extracted features. *Applied Soft Computing*, *93*, 106311.
50. Chin, C. S., Si, J., Clare, A. S., & Ma, M. (2019). Intelligent Fouling Detection System Using Haar-Like Cascade Classifier with Neural Networks. In *Advances in Computer Communication and Computational Sciences* (pp. 393-406). Springer, Singapore.