ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

ΠΡΟΓΡΑΜΜΑ ΠΡΟΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Σαμουρίδη Χρήστου (ΑΜ: 131117)

Ανίχνευση όγκου εγκεφάλου χρησιμοποιώντας αλγόριθμους ανίχνευσης αντικειμένων και νευρωνικά δίκτυα και απεικόνιση σε υπολογιστικό νέφος

Επιβλέπων: Βογιατζής Ιωάννης

Αθήνα, Φεβρουάριος 2022

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον επιβλέποντα κ. Ιωάννη Βογιατζή και τον υποψήφιο διδάκτορα κ. Δημήτριο Ψιλιά για την εμπιστοσύνη που μου έδειξαν με την ανάθεση της συγκεκριμένης διπλωματικής εργασίας, την επιστημονική καθοδήγηση, τις υποδείξεις τους και για τις συμβουλές τους σε όλη την πορεία της υλοποίησής της.

Ioannis Voyiatzis

Digitally signed by Ioannis Voyiatzis
Date: 2022.06.07 14:45:24 +03'00'

Kleio Sgouropoulou

Digitally signed by Kleio Sgouropoulou
Date: 2022.06.07 15:18:18 +03'00'

Panagiotis Giannakopoulos

Digitally signed by Panagiotis Giannakopoulos
Date: 2022.06.07 22:59:57 +03'00'

Αθήνα, Φεβρουάριος 2022

Contents

Λίστα Σχημάτων-Εικόνων

Λίστα Πινάκων

Συντομογραφίες – Ακρωνύμια

ANN	Artificial Neural Network
API	Application Programming Interface
CNN	Convolutional Neural Network
CPU	Central Processing Unit
F.C.	Fully Connected
GPU	Graphics Processing Unit
HGG	High Grade Glioma
IOU	Intersection Over Unit
LGG	Low Grade Glioma
MRI	Magnetic Resonance Imaging
R-CNN	Region-based Convolutional Neural Network
ReLU	Rectified Linear Unit
ROI	Region Of Interest
RPN	Region Proposal Network
SSD	Single Shot Detector
SVM	Support Vector Machine
TPU	Tensor Processing Unit
TLD	Tracking Learning Detection
YOLO	You Only Look Once

Περίληψη

Στην παρούσα εργασία έγινε έρευνα ως προς τις λειτουργίες ενός συστήματος υπολογιστικής όρασης, ή αλλιώς με ένα αυτοματοποιημένο σύστημα με σκοπό την εκπαίδευσή του ώστε να είναι σε θέση να ανιχνεύει και να αναγνωρίζει τύπους όγκου εγκεφάλου μέσω νευρωνικών δικτύων.

Ξεκινώντας θα εξηγηθούν βασικές αρχές-έννοιες, που σχετίζονται με τις προαναφερθείσες. Οι αρχές αυτές έχουν να κάνουν με τον ταξινομητή του αντικειμένου, την ταξινόμηση, την ανίχνευση αντικειμένου, την συλλογή των δεδομένων, την επεξεργασία εικόνων και τέλος τη δημιουργία ενός μοντέλου που θα μπορεί να συνδυάσει χαρακτηριστικά και να αντλήσει χρήσιμες πληροφορίες εξάγοντάς τες με κάποιους αλγορίθμους από ένα σύνολο δεδομένων.

Επίσης θα γίνει αναφορά στους παράγοντες οι οποίοι καθιστούν δυσκολότερη την αναγνώριση αντικειμένου αλλά και στις προσεγγίσεις της αναγνώρισης. Στην συνέχεια θα γίνει αναφορά στους ανιχνευτές υψηλής τεχνολογίας CNN (R-CNN, Fast R-CNN, Faster R-CNN, SSD, YOLO). Τέλος θα παρουσιαστούν τα εργαλεία που ήταν αναγκαία για την υλοποίηση της έρευνας (TensorFlow, CUDA και βιβλιοθήκη Nvidia Cudnn, γλώσσα python, OpenCV, Raspberry Pi, LabelImg).

Για το κομμάτι της υλοποίησης μέσω συνελικτικών νευρωνικών δικτύων θα γίνει εκπαίδευση ενός ταξινομητή ανίχνευσης αντικειμένων, με σκοπό την ανίχνευση όγκου στον εγκέφαλο. Η εκπαίδευση θα γίνει με την χρήση της βιβλιοθήκης που δημιουργήθηκε από την ομάδα Google Brain για αριθμητικούς υπολογισμούς και μηχανική μάθηση, το TensorFlow, την παράλληλη υπολογιστική πλατφόρμα της NVIDIA CUDA v10, τις βιβλιοθήκες cuDNN 8 και το περιβάλλον Anaconda. Κατόπιν, θα γίνει συνοπτική αναφορά όλων των βημάτων της υλοποίησης και στην συνέχεια θα περιγραφούν αναλυτικά τα βήματα που ακολουθήσαμε με ταυτόχρονη παράθεση εικόνων επεξήγησης του κάθε βήματος. Ακολούθως, θα γίνουν παρατηρήσεις μεταξύ των εκπαιδεύσεων αποκλειστικά με GPU και CPU ώστε να εξαχθούν συμπεράσματα για τους λόγους που η χρήση GPU είναι σημαντική κατά την εκπαίδευση μοντέλων βαθιάς μάθησης. Επιπλέον αφού ο ανιχνευτής είναι ολοκληρωμένος θα γίνει η ενσωμάτωσή του σε ένα Raspberry Pi και θα αναλυθεί η διαδικασία βημάτων υλοποίησής του.

Από τους ανιχνευτές υψηλής τεχνολογίας που αναλύθηκαν χρησιμοποιήθηκε ένας Faster R-CNN ανιχνευτής, που κρίθηκε αποτελεσματικός ως προς την έρευνα που υλοποιήθηκε. Τέλος παραθέτονται τα αποτελέσματα σε έκθεση μέσω των τεχνολογιών του υπολογιστικού νέφους για μελλοντική πιθανή χρήση.

Λέξεις Κλειδιά: Faster R-CNN, Computer Vision, CNN, Object Detection, Raspberry Pi

Abstract

In the following bachelor's thesis, research was done on the functions of computer vision system, or in other words with an automated system for the purpose of training it to be able to detect and recognize types of brain tumors through neural networks.

To begin with, basic principles-concepts will be explained, related to the above. These principles have to do with the object classifier, sorting, object detection, data collection, image processing and finally creating a model that can combine features and extract useful information by exporting them with some algorithms from a dataset.

Reference will also be made to the factors that make object recognition more difficult as well as to the recognition approaches. Next we will refer to the high technology CNN detectors(R-CNN, Fast R-CNN, Faster R-CNN, SSD, YOLO). Finally, reference will be made to the tools needed to complete the research (TensorFlow, CUDA, Nvidia Cudnn, python, OpenCV, Raspberry Pi, LabelImg).

For the implementation part though convolutional neural networks, an object detection classifier will be trained, with the aim of detecting tumors in the brain. Training will be done using the library created by the Google Brain team for arithmetic and machine learning, TensorFlow-GPU, the NVIDIA CUDA v10 parallel computing platform, cuDNN 8 libraries and Anaconda environment. Then, all the steps of the implementation will be briefly reported and then the steps we followed will be described in detail with a simultaneous presentation of explanatory images of each step. Next, observations will be made between GPU and CPU-only training in order to draw conclusions as to why GPU usage is important in the training of deep learning models. In addition, once the detector is complete, it will be integrated into a Raspberry Pi and the implementation process will be analyzed.

From the high-tech detectors analyzed, a Faster R-CNN was used, which considered effective in terms of the research was carried out. Finally, the results are presented in a cloud report through cloud computing technologies for possible future work.

Keywords: Faster R-CNN, Computer Vision, CNN, Object Detection, Raspberry Pi

1. Εισαγωγή

1.1 Ανίχνευση όγκου εγκεφάλου χρησιμοποιώντας αλγορίθμους ανίχνευσης αντικειμένων και νευρωνικά δίκτυα και απεικόνιση σε υπολογιστικό νέφος

Η συγκεκριμένη έρευνα αφορά ένα σύστημα ανίχνευσης βασισμένο στην υπολογιστική όραση. Στο πρώτο στάδιό της θα αναλυθούν και εξηγηθούν οι αλγόριθμοι αναγνώρισης αντικειμένου. Στην συνέχεια θα δημιουργηθεί το μοντέλο που θα έχει την δυνατότητα να κάνει εξαγωγή της απόφασης. Ολόκληρη η υλοποίηση θα αφορά ένα σύστημα ανίχνευσης όγκου εγκεφάλου, το οποίο θα βασίζεται σε κάποια δεδομένα εκπαίδευσης, που αποτελούνται από ένα σύνολο μαγνητικών τομογραφιών. Αφού ολοκληρωθεί η εκπαίδευση του μοντέλου ανίχνευσης, θα ενσωματωθεί το μοντέλο σε ένα Raspberry Pi.

Σημαντικός στόχος των υπολογιστικών συστημάτων στον τομέα της ιατρικής είναι η ανίχνευση όγκου στον εγκέφαλο. Ένας όγκος εγκεφάλου (Brain Tumor), όπως και κάθε όγκος διακρίνεται σε δύο κατηγορίες, οι οποίες είναι: ο καλοήθης (LGG - χαμηλού βαθμού γλοίωμα) και ο κακοήθης (HGG - υψηλού βαθμού γλοίωμα) [3]. Η έννοια καλοήθης αναφέρεται στον όγκο ο οποίος αναπτύσσεται αργά και χωρίς να εισχωρεί στον εγκέφαλο, ενώ η έννοια κακοήθης αναφέρεται στον όγκο ο οποίος αναπτύσσεται με γρήγορο ρυθμό και έχει την τάση να εισχωρεί στον εγκέφαλο που βρίσκεται γύρω του. Το είδος του κάθε όγκου έχει κρίσιμη σημασία για το είδος της θεραπείας. Κατά την εξέταση και προς την υποστύλωση των νευροχειρούργων, χρησιμοποιείται ένα συνελικτικό νευρωνικό δίκτυο για την ανίχνευση του όγκου μέσω ενός υπολογιστικού συστήματος. Σε αυτήν την έρευνα θα εφαρμοστεί η αρχιτεκτονική Faster R-CNN. Για τους σκοπούς της εκπαίδευσης θα χρησιμοποιηθεί σύνολο δεδομένων από διαγωνισμό BraTS 2016 (Brain Tumor Image Segmentation). Παρόλα αυτά, δεν παίζει ρόλο η κατηγοριοποίηση του αν είναι καλοήθης ή κακοήθης, διότι κάθε όγκος ακόμα και καλοήθης μπορεί να προκαλέσει εξίσου σοβαρά προβλήματα και να θέσει σε κίνδυνο την ζωή του πάσχοντος. Το ίδιο ισχύει και για τους κακοήθεις, συνεπώς είναι αναγκαία η αναγνώριση και η άμεση αντιμετώπιση.

1.2 Πληροφορίες σχετικά με όγκο στον εγκέφαλο

Τις τελευταίες δεκαετίες έχει επιτευχθεί η διάγνωση όγκων του εγκεφάλου μέσω απεικόνισης της μαγνητικής τομογραφίας (MRI), λόγω της ευκρινούς αντίθεσης των μαλακών ιστών στον εγκέφαλο. Οι όγκοι αυτοί ονομάζονται και γλοιώματα τα οποία αποτελούν το μεγαλύτερο ποσοστό των κακοήθων όγκων του εγκεφάλου. Αυτά συνήθως προέρχονται από τα νευρογλοιακά κύτταρα του κεντρικού νευρικού συστήματος. Ο Παγκόσμιος Οργανισμός Υγείας (World Health Organization-WHO) [19],

ανάλογα με την επιθετικότητα των γλοιωμάτων, τα κατέταξε σε δύο κατηγορίες, η πρώτη κατηγορία είναι τα γλοιώματα χαμηλής ποιότητας (LGG), τα οποία αποτελούνται από γλοιώματα χαμηλού βαθμού και μεσαίου βαθμού και η δεύτερη κατηγορία είναι τα γλοιώματα υψηλού βαθμού (HGG). Στην πρώτη κατηγορία (LGG) αναφερόμαστε σε διεισδυτικά εγκεφαλικά νεοπλάσματα που περιλαμβάνουν ιστολογικές τάξεις, τα νεοπλάσματα βαθμού II και III. Ασθενείς οι οποίοι έπασχαν από LGG, που συνεπώς είναι σε καλύτερη κατάσταση από αυτούς με HGG, παρατηρήθηκε πως προχωρούν σε δευτερογενή γλοιωβλάστωμα (γλοιώματα υψηλού βαθμού κακοήθειας).

Όσον αφορά στην διάγνωση ενός όγκου εγκεφάλου, για την εύρεση της ύπαρξης αυτού, οι ασθενείς ελέγχονται μέσω απεικονιστικών εξετάσεων όπως αξονικές ή μαγνητικές τομογραφίες. Σε περιπτώσεις εμφάνισης κάποιων δειγμάτων ύπαρξης, για την έγκυρη αναγνώριση της πάθησης οι ασθενείς υπόκεινται σε χειρουργική επέμβαση με σκοπό την λήψη υλικού για βιοψία. Τα συστατικά υψηλής ποιότητας, μεταξύ των παραγόμενων δειγμάτων βιοψίας, χρησιμοποιούνται για την σωστή πρόβλεψη του βαθμού του όγκου. Τα γλοιώματα τα οποία είναι ετερογενή, πολλές φορές βάσει των ιστοπαθολογικών δειγμάτων που συλλέγονται από διαφορετικά μέρη του ίδιου όγκου είναι πιθανό να εμφανίσουν διαφορετικούς βαθμούς. Είναι επίσης πιθανό, κατά την δειγματοληψία ενός όγκου, να λείπει το συστατικό υψηλής ποιότητας από το δείγμα, με αποτέλεσμα την λανθασμένη ανίχνευση και αντιμετώπιση της νόσου. Παρ' όλα αυτά υπάρχουν πιθανότητες επιπλέον κινδύνων όπως, η αιμορραγία από τον όγκο ή και τον εγκέφαλο κατά την διάρκεια της βιοψίας λόγω της βελόνας της βιοψίας, που μπορεί να προκαλέσει σοβαρό εγκεφαλικό επεισόδιο, ημικρανία, κώμα έως και θάνατο.

1.3 Μαγνητική τομογραφία

Η μαγνητική τομογραφία (MRI-Magnetic Resonance Imaging) είναι μια ακτινολογική μέθοδος για την απεικόνιση του εσωτερικού ενός οργανισμού. Με την μαγνητική τομογραφία, είναι δυνατή η ανίχνευση ολόκληρου όγκου, με δυνατότητα επίδειξης μεγάλης συσχέτισης με ιστολογική ποιότητα. Δεν υπάρχει ευαισθησία στο σφάλμα δειγματοληψίας και στην μεταβλητότητα μεταξύ των παραμέτρων και μεταξύ των παρατηρητών [9]. Γι' αυτό τον λόγο η MRI πολλαπλών ακολουθιών παίζει σημαντικό ρόλο στην ανίχνευση, την διάγνωση και την διαχείριση αυτών των όγκων του εγκεφάλου. Με την αποκλειστική χρήση μαγνητικού τομογράφου είναι δυνατή η λήψη πληροφοριών σχετικά με την βιοχημική κατάσταση των ιστών με την μορφή εικόνων και φασμάτων. Προσφέρει την δυνατότητα του έγκαιρου εντοπισμού διαφόρων βιοχημικών αλλαγών οι οποίες συμβαίνουν πριν το σχηματισμό κακοήθειας. Επίσης σε σχέση με άλλες απεικονιστικές μεθόδους, οι οποίες βασίζονται στην ανίχνευση συγκεκριμένων ιχνηθετών με εξειδικευμένη δράση, προσφέρουν μεγαλύτερη ευαισθησία, καλύτερη

διακριτική ικανότητα και μεγαλύτερη ευελιξία στην εφαρμογή.

1.4 Θεραπεία όγκων εγκεφάλου

Για την αντιμετώπιση όγκων εγκεφάλου έχουν αναπτυχθεί οι παρακάτω τρόποι οι οποίοι ανάλογα με την επικινδυνότητα τους επιλέγεται και ο αντίστοιχος:

- **Χειρουργική αφαίρεση**: Χρησιμοποιείται και για την διάγνωση αλλά και την θεραπεία των όγκων εγκεφάλου. Η αφαίρεση όγκων βοηθάει στην ελάττωση της πίεσης σε γειτονικά τμήματα του εγκεφάλου. Αφού επιτευχθεί η αφαίρεση του κακοήθη όγκου, υπάρχει περίπτωση ανάγκης ακτινοθεραπειών ή και χημειοθεραπειών για την ολική εξάλειψη καρκινικών κυττάρων.
- **Χημειοθεραπεία**: Χρησιμοποιούνται φάρμακα για την αναστολή της ανάπτυξης των καρκινικών κυττάρων. Η χορήγηση γίνεται σε μορφή χαπιών ή ενδοφλέβια, ώστε το φάρμακο να φτάσει τα καρκινικά κύτταρα όπου χρειάζεται. Επιπλέον τρόπος είναι η απευθείας χορήγησης του φαρμάκου στο εγκεφαλονωτιαίο υγρό.
- **Ακτινοθεραπεία**: Χρησιμοποιούνται υψηλής ενέργειας ακτίνες Χ ή άλλα είδη ακτινοβολιών ώστε να εξοντωθούν τα καρκινικά κύτταρα ή να ανασταλεί η ανάπτυξή τους.
- **Στοχευμένη θεραπεία**: Χορηγούνται φάρμακα που κατευθύνονται σε συγκεκριμένα καρκινικά κύτταρα. Έχει μικρότερη βλάβη σε φυσιολογικά κύτταρα σε σχέση με την χημειοθεραπεία ή την ακτινοβολία. Μια τέτοια ουσία είναι η μπεβασιζουμάμπη, που χρησιμοποιείται για την θεραπεία πρωτοπαθούς κακοήθους όγκου εγκεφάλου.

1.5 Δομή της εργασίας

Ξεκινώντας, στο πρώτο κεφάλαιο της εργασίας γίνεται μία αναφορά περιληπτικά στις διαφορές στους όγκους εγκεφάλου (LGG-χαμηλού βαθμού γλοίωμα, HGG-υψηλού βαθμού γλοίωμα) και στην απεικόνιση μαγνητικών τομογραφιών. Στην συνέχεια, στο δεύτερο κεφάλαιο γίνεται μια επεξήγηση των εννοιών της ταξινόμησης, των ειδών ταξινομητή αντικειμένου, της επεξεργασίας εικόνας, της ανίχνευσης αντικειμένου, της μαζικής συλλογής δεδομένων και της υλοποίησης ενός μοντέλου που έχει την δυνατότητα να συνδυάζει τα χαρακτηριστικά τα οποία αντλήθηκαν μέσω κάποιων αλγορίθμων από το σύνολο δεδομένων. Στο σημείο του τρίτου κεφαλαίου, θα εμβαθύνουμε στους παράγοντες οι οποίοι δυσχεραίνουν την αναγνώριση αντικειμένου και την προσέγγιση αναγνώρισης αυτού. Ύστερα, στο τέταρτο κεφάλαιο θα αναλυθούν οι ανιχνευτές CNN (RCNN, Fast RCNN, Faster R-CNN, YOLO, SSD)

οι οποίοι είναι ανιχνευτές υψηλής τεχνολογίας. Στο πέμπτο κεφάλαιο θα γίνει αναφορά στα λογισμικά τα οποία θα χρησιμοποιηθούν για την δημιουργία του ανιχνευτή (Python, OpenCV, TensorFlow, βιβλιοθήκη Nvidia Cudnn, CUDA, LabelImg). Ακολούθως, στο έκτο κεφάλαιο θα παρουσιαστεί η δημιουργία των καταλόγων που θα περιέχουν όλες τις αναγκαίες εξαρτήσεις και κάνουμε αναφορά στα βήματα της εκπαίδευσης του μοντέλου, που θα είναι σε θέση να κάνει αναγνώριση όγκου στον εγκέφαλο. Επίσης παρουσιάζουμε τις διαφορές μεταξύ δύο εκπαιδεύσεων που κάναμε, μια αποκλειστικά σε GPU και μια στην CPU του συστήματος. Έπειτα, στο έβδομο κεφάλαιο γίνεται σχολιασμός των μικροϋπολογιστών Raspberry και ακολουθώντας κάποια βήματα υλοποιείται η ανάπτυξη-ενσωμάτωση του εκπαιδευμένου μοντέλου σε ένα Raspberry Pi. Τέλος, στο όγδοο κεφάλαιο γίνεται έκθεση των αποτελεσμάτων τα οποία αντλήσαμε από την εκπαίδευση του μοντέλου μας στην cloud-based πλατφόρμα Google Data Studio που δίνει την δυνατότητα μέσω διαγραμμάτων να προβληθούν τα δεδομένα.

2. Επιστημονικοί όροι και αρχές

Ο άνθρωπος έχει την ικανότητα της αναγνώρισης αντικειμένων μέσω της όρασης τους και όχι μόνο. Μπορεί να αντιλαμβάνεται την κατηγορία κάθε αντικειμένου, τον σκοπό του, παρά τις παραμορφώσεις που μπορεί να έχουν υποστεί, δηλαδή τις εναλλαγές στον φωτισμό και την θέση. Έτσι δημιουργείται και η ανάγκη δημιουργίας συστημάτων τα οποία θα εκπαιδεύονται για να μπορούν να έχουν και αυτά αυτήν την δυνατότητα αναγνώρισης και αντίληψης αντικειμένων.

2.1 Μέθοδος ανίχνευσης αντικειμένου

Ο όρος μέθοδος ανίχνευσης αντικειμένου αναφέρεται σε νέες τεχνολογίες υπολογιστών οι οποίες έχουν να κάνουν με την υπολογιστική όραση ή τεχνητή όραση, μια διαδικασία κατά την οποία είναι δυνατή η ανίχνευση ύπαρξης αντικειμένου και επεξεργασίας αυτού μέσω εικόνας και βίντεο.

Με τον όρο ανίχνευση δηλώνεται η διαδικασία κατά την οποία πραγματοποιείται η εύρεση του αντικειμένου μέσα σε ένα πλαίσιο. Για την αναγνώριση μεγέθους ή τοποθεσίας είναι αναγκαίο να χρησιμοποιηθεί πλαίσιο οριοθέτησης, που στην πορεία η εικόνα που βρίσκεται εντός οριοθέτησης ταξινομείται μέσω ενός αλγορίθμου εκπαιδευμένου με μηχανική μάθηση. Δυνατότητα βελτίωσης ορίων είναι δυνατό να επιτευχθεί μέσω επαναληπτικών διαδικασιών.

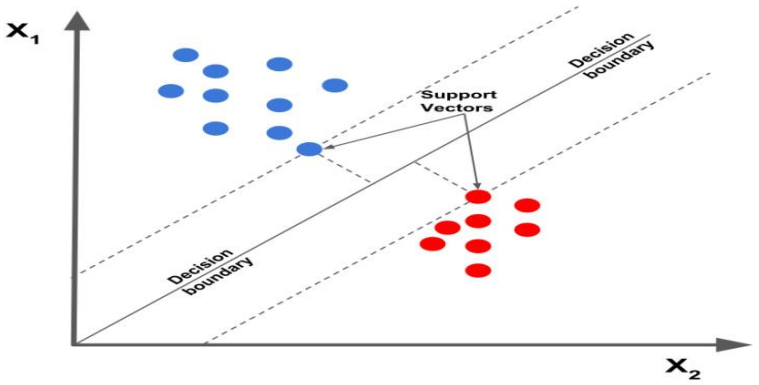
2.2 Μέθοδοι ταξινόμησης αντικειμένων

Ταξινόμηση ονομάζεται η διαδικασία κατά την οποία ένα οποιουδήποτε είδους αντικείμενο αναγνωρίζεται και κατηγοριοποιείται βάσει κάποιων συγκεκριμένων χαρακτηριστικών. Σύμφωνα με τα χαρακτηριστικά που έχουν εξαχθεί μέσα από ένα σύνολο αντικειμένων, γίνεται μια ταξινόμηση που έχει στόχο τις περισσότερες κοινές προσεγγίσεις μεταξύ τους. Οι πιο συνηθισμένοι μέθοδοι ταξινόμησης είναι ο **Support Vector Machine**, ο **Template Matching** και μέσω **Artificial Neural Network.** Θα γίνει μια περίληψη των τριών πιο συνηθισμένων τρόπων ταξινόμησης ώστε να αποτυπωθούν τα χαρακτηριστικά τους.

Support Vector Machine

Το **Support Vector Machine (SVM)** είναι μοντέλο μάθησης που χρησιμοποιούνται είτε για ταξινόμηση είτε για επιλογές αποφάσεων. Είναι από τους πιο ισχυρούς αλγόριθμους πρόβλεψης που βασίζεται σε στατιστικά πλαίσια μάθησης. Λαμβάνοντας υπόψη κάποιο σύνολο παραδειγμάτων

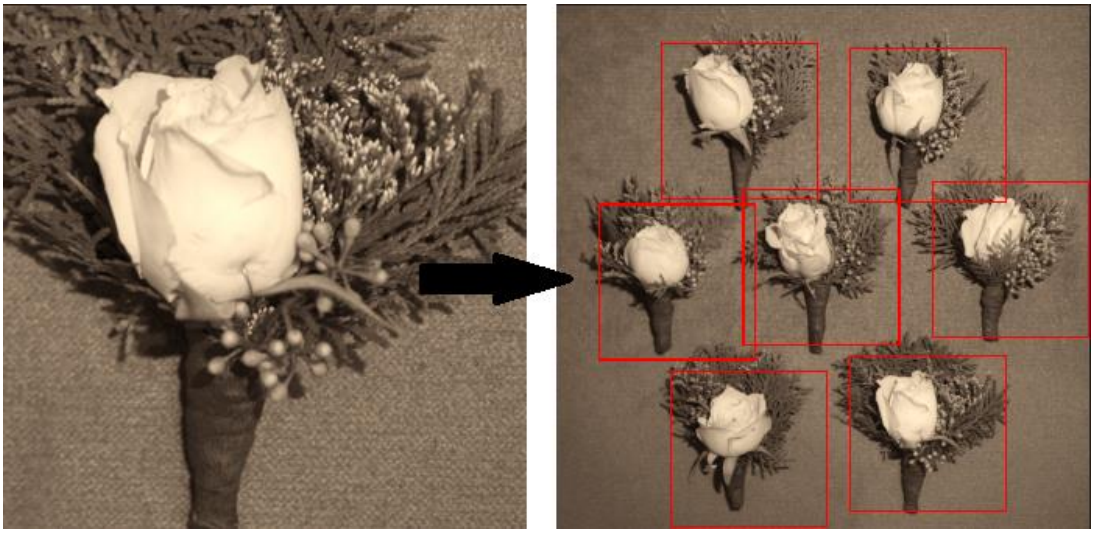
εκπαίδευσης, κάθε αντικείμενο επισημαίνεται ότι ανήκει σε μια από τις δύο κατηγορίες [6].



Εικόνα 1: Support Vector Machine

Template Matching

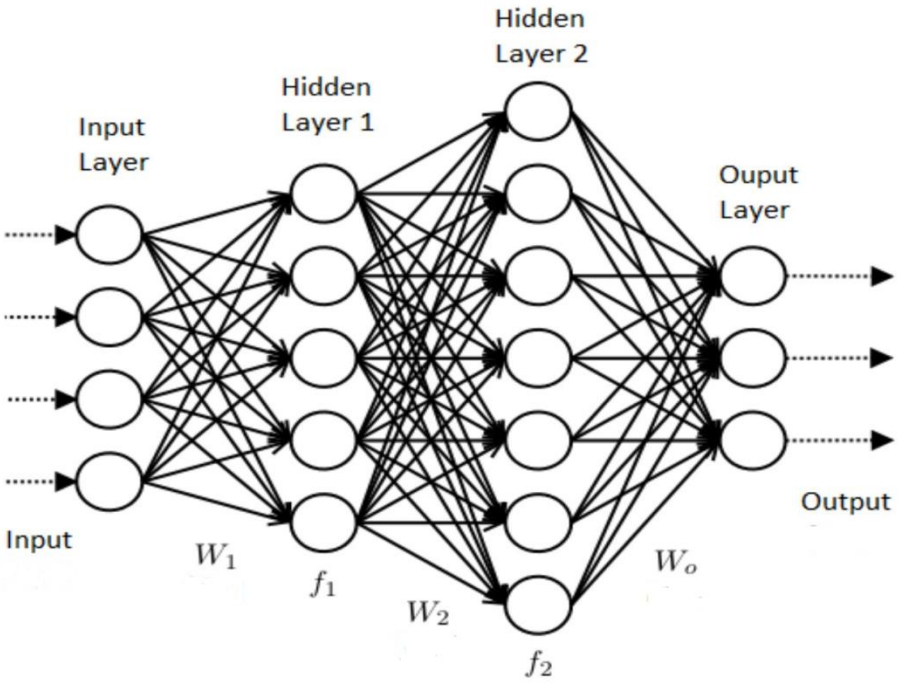
Το Template Matching κατατάσσεται στους αλγορίθμους αντιστοίχισης και χρησιμοποιείται κυρίως στην ψηφιακή επεξεργασία εικόνας. Σε αυτή την μέθοδο όλα τα χαρακτηριστικά των αντικειμένων καταγράφονται σε μια βάση δεδομένων, με σκοπό να τα επεξεργαστεί και να ταξινομήσει βάσει κοινών χαρακτηριστικών. Τέτοια χαρακτηριστικά είναι οι γωνίες θέασης, ο φωτισμός,, οι αλλαγές φόντου κ.ά. . Έτσι βάσει αυτών, κάθε αντικείμενο συγκρίνεται με κάθε πρότυπο ίδιων χαρακτηριστικών [25]. Είναι δυνατή η επεξεργασία-τροποποίηση προτύπων για αντιστοίχιση νέων κλάσεων.



Εικόνα 2: Template Matching

Artificial Neural Network

Το **Artificial Neural Network** είναι σύστημα βασισμένο σε τεχνητά νευρωνικά δίκτυα, τα οποίο εμπνεύστηκε από τα βιολογικά νευρωνικά δίκτυα, δηλαδή αυτά του ανθρώπινου εγκεφάλου με σκοπό την επεξεργασία της πληροφορίας [22]. Το βασικό χαρακτηριστικό τους είναι πως αποτελούνται από συνδεδεμένα στοιχεία επεξεργασίας ή αλλιώς νευρώνες που λειτουργούν όλοι μαζί με σκοπό την επίλυση του προβλήματος ακριβώς όπως οι νευρώνες σε ένα βιολογικό εγκέφαλο. Όπως ένας άνθρωπος, έτσι και τα ANN μπορούν να μάθουν μέσω παραδειγμάτων, μέχρι ο συντελεστής της συνάρτησης λάθους να ελαχιστοποιηθεί. Τέτοια συστήματα μπορούν να δημιουργηθούν για μεγάλο εύρος εφαρμογών, που με γνώμονα ένα σύνολο δεδομένων-πληροφοριών, εκπαιδεύονται οι νευρώνες με στόχο την κατηγοριοποίηση αντικειμένων στις αντίστοιχες συστάδες.



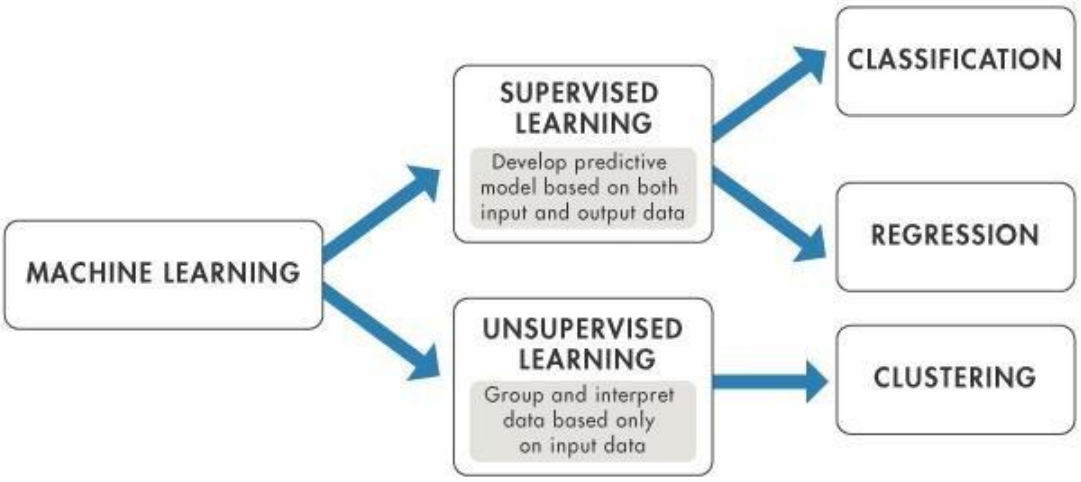
Εικόνα 3: Αρχιτεκτονική τεχνητού νευρωνικού δικτύου

2.3 Είδη ταξινομητών αντικειμένων

Ένας ταξινομητής αντικειμένου είναι ένας αλγόριθμος ο οποίος βάσει ενός συνόλου χαρακτηριστικών που περιγράφουν τα αντικείμενα, τα αντιστοιχίζει στις αντίστοιχες κλάσεις που τους αναλογούν. Ο αλγόριθμος δέχεται τα χαρακτηριστικά του κάθε αντικειμένου και ο ταξινομητής τα

επεξεργάζεται ώστε να αποφασίζει σε ποια κλάση θα το κατατάξει. Υπάρχουν δύο ειδών ταξινομητές οι οποίοι θα αναλυθούν στην επόμενη παράγραφο.

Τα δύο είδη των ταξινομητών είναι οι **unsupervised** και οι **supervised**. Στην πρώτη κατηγορία είναι οι unsupervised learning ταξινομητές στους οποίους το σύστημα ταξινόμησης λειτουργεί κατευθείαν μέσω των δεδομένων, χωρίς να υπάρχει κάποιο σύνολο εκπαίδευσης ή προκαθορισμένες κατηγορίες. Δηλαδή στην unsupervised learning υπάρχουν μόνο δεδομένα εισόδου, χωρίς συγκεκριμένη μεταβλητή εξόδου. Αντίθετα στους ταξινομητές supervised learning, έχουν οριστεί κάποιες κλάσεις στις οποίες αντιστοιχεί κάθε αντικείμενο. Υπάρχει ένα σύνολο δεδομένων στο οποίο περιέχονται τα χαρακτηριστικά όλων των αντικειμένων, που μέσω αυτών οι ταξινομητές θα μάθουν να ταξινομούν τα αντικείμενα στις αντίστοιχες κλάσεις. Όσον αφορά την δημιουργία και την ανάπτυξη των ταξινομητών, πρέπει να ακολουθηθεί μια διαδικασία κάποιων βημάτων που είναι η δημιουργία του συνόλου της εκπαίδευσης, η επιλογή των χαρακτηριστικών, η εκπαίδευση του ταξινομητή και τέλος η αξιολόγησή του.



Εικόνα 4: Supervised & Unsupervised Learning

2.4 Μέθοδος παρακολούθησης αντικειμένων

Για την παρακολούθηση ενός αντικειμένου είναι απαραίτητη μια ακολουθία βημάτων τα οποία είναι, αρχικά η εισαγωγή μιας εισόδου από μια ακολουθία πινάκων (κάθε πίνακας αντιπροσωπεύει μια εικόνα σε μια ροή βίντεο που δέχεται με την χρήση ενός μέσου καταγραφής π.χ. κάμερα, κάθε καταχώρηση αντιπροσωπεύει μεμονωμένα εικονοστοιχεία) και ενός αρχικού συνόλου επιλεγμένων

καταχωρήσεων από τον πίνακα πρότυπο που απεικονίζει το αντικείμενο ενδιαφέροντος της παρακολούθησης. Σκοπός της μεθόδου αυτής είναι να γίνει παρακολούθηση του αντικείμενο ως προς το συνόλου των εικόνων και για κάθε μια από αυτές τις εικόνες να εκπέμπονται οι καταχωρίσεις οι οποίες αντιστοιχούν στο αντικείμενο ενδιαφέροντος, πολλές φορές και με αλλαγμένη εμφάνιση ως προς την εικόνα πρότυπο. Ένα πρόβλημα σε αυτήν την περιγραφή είναι πως υπάρχει περίπτωση το αντικείμενο ενδιαφέροντος να μην έχει οριστεί σωστά ως προς την αρχική εικόνα, λόγω εξωτερικών παραγόντων. Τέτοιες συνθήκες μπορεί να είναι η περιστροφή του αντικειμένου μέσα και έξω από την εικόνα, οι αλλαγές στο περιβάλλον της εικόνας κ.λπ.. Συμπέρασμα πως ακόμα και στις ιδανικότερες συνθήκες, κατά την μέθοδο της ανίχνευσης πρέπει να αντιμετωπιστούν αρκετά εμπόδια ώστε να αποδώσει σωστά.

Για να γίνει μια ικανοποιητική παρακολούθηση ενός αντικειμένου, ένας ανιχνευτής πρέπει να ενσωματώσει στο μοντέλο πολλές όψεις που απεικονίζουν τον στόχο. Αυτή η τεχνική όμως, είναι πιθανό να καταλήξει σε λάθη, καθώς η συνεχής προσαρμογή και αναγνώριση του αντικειμένου, θα ωθήσει το μοντέλο να παραπαίει από τον στόχο λόγω της ενσωμάτωσης. Έτσι δημιουργείται η ανάγκη της υλοποίησης ενός αλγορίθμου ο οποίος θα μπορεί να μεταβάλει τον στόχο, αλλά ταυτόχρονα να εξαφανίζει την πιθανή παραποίηση στην κίνηση του στόχου. Μια αρκετά αποδοτική αρχιτεκτονική παρακολούθησης είναι ο αλγόριθμος παρακολούθησης εκμάθησης ανίχνευσης (TLD) [13]. Αυτή η τεχνική βασίζεται στην επεξεργασία εικόνας και στην δυνατότητα οριοθέτησης πλαισίων του στόχου. Όπως υποδηλώνει και το όνομά του, TLD-Tracking Learning Detection, ο αλγόριθμος προφανώς κάνει χρήση της μηχανικής μάθησης, που στοχεύει στο πρόβλημα της ενσωμάτωσης νέων προβολών του προς ανίχνευση αντικειμένου. Όσον αφορά το κομμάτι της παρακολούθησης του αλγορίθμου, επικεντρώνεται στο πρόβλημα της αναζήτησης της θέσης του στόχου σε προηγούμενα πλαίσια ώστε να προσδιορίσει τη θέση στο τρέχον πλαίσιο. Τέλος το κομμάτι της ανίχνευσης επικεντρώνεται στο πρόβλημα της εύρεσης του αντικειμένου σε ένα πλαίσιο, δεδομένου ενός μοντέλου αυτού του στόχου-αντικειμένου.

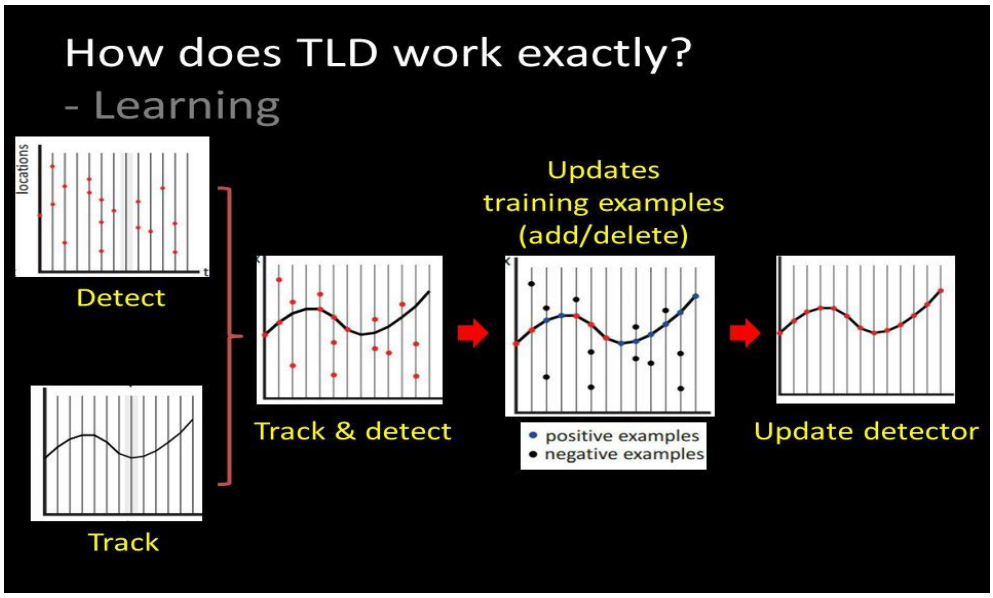
Κάνοντας αναφορά στα τρία στάδια του αλγορίθμου ανίχνευσης εκμάθησης εντοπισμού, παρατηρείται πως οι έννοιες ανίχνευση και παρακολούθηση μπορεί να φανούν κοινές. Παρ' όλα αυτά οι δύο έννοιες αφορούν εντελώς διαφορετικό σκοπό και λειτουργία. Στην περίπτωση της ανίχνευσης, η λειτουργία που εξετάζεται είναι η εύρεση ενός αντικειμένου-στόχου βασιζόμενη σε ένα μοντέλο αντικειμένου. Ενώ στην περίπτωση της παρακολούθησης γίνεται μια επεξεργασία χαρακτηριστικών ώστε να γίνει αντιληπτή η θέση του αντικειμένου-στόχου στα τρέχοντα πλαίσια.

Προφανώς και οι δύο αυτές διαδικασίες έχουν πιθανότητα σφάλματος, οι ανιχνευτές πολλές φορές παρασύρονται από τον στόχο, και πολλές φορές αποτυγχάνουν στην ανίχνευση όταν ο στόχος αφήσει το πλαίσιο και επιστρέψει. Αντίθετα στην περίπτωση της παρακολούθησης, το αντικείμενο

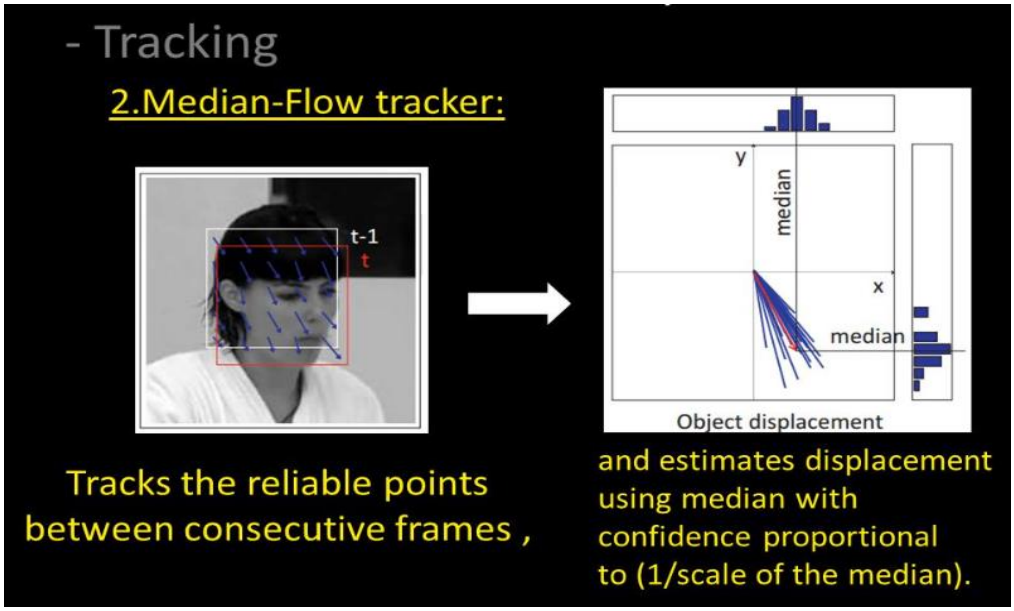
εντοπίζεται σχεδόν πάντα σωστά, υποθέτοντας ότι βρίσκει ομοιότητες ως προς το μοντέλο αντικειμένου με κάποιο κριτήριο εμπιστοσύνης.

Η ιδέα πίσω από τον αλγόριθμο TLD είναι να γίνει η εκπαίδευση επαναληπτικά, χρησιμοποιώντας ένα σύνολο δεδομένων εκπαίδευσης που ενημερώνονται από προηγούμενες εσφαλμένες εικόνες-πλαίσια. Η τεχνική αυτή έχει επιτυχή αποτελέσματα όταν τα αντικείμενα-στόχοι είναι στατικά-σταθερά. Επειδή όμως τα αντικείμενα-στόχοι παρουσιάζουν αλλαγές στην αποτύπωση τους, δηλαδή αλλαγές ως προς περιστροφή του αντικειμένου και παραμορφώσεις αυτού πολλές φορές οι ανιχνευτές αποτυγχάνουν στην αναγνώριση. Σύμφωνα με τον Z. Kalal, K. Mikolajczyk, και J. Matas, για την τεχνική αυτή χρησιμοποιείται ένα ζεύγος εμπειρογνωμόνων, στο οποίο ο ένας χρησιμοποιείται για την εύρεση false positives και ο άλλος false negatives, ώστε να προστίθενται νέες ετικέτες εικόνων σε κάθε στιγμιότυπο του χρόνου και να δημιουργούνται παραδείγματα εκπαίδευσης για την αποφυγή λαθών στο μέλλον.

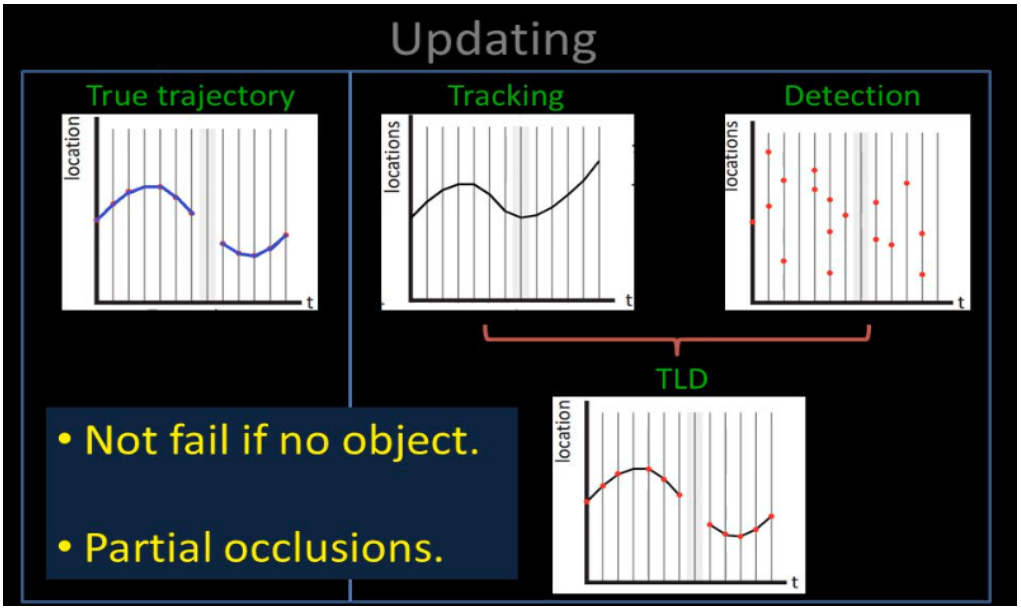
Ωστόσο για την λειτουργία της παρακολούθησης κάνει χρήση ενός ανιχνευτή μέσης ροής (Median-Flow), ο οποίος εκμεταλλεύεται την οπτική ροή των μέσων ώστε να προσδιορίσει την κίνηση των αντικειμένων εντός των πλαισίων. Λειτουργεί δηλαδή βάσει πρόβλεψης της νέας θέσης των αντικειμένων-στόχων, εξαλείφοντας λανθασμένες προβλέψεις μέσω κριτηρίων εμπιστοσύνης.



Εικόνα 5: Λειτουργία εκμάθησης αλγόριθμου TLD



Εικόνα 6: Λειτουργία παρακολούθησης αλγόριθμου TLD



Εικόνα 7: Λειτουργία παρακολούθησης και ανίχνευσης αλγόριθμου TLD

2.5 Προβλήματα που αντιμετωπίζονται κατά την αναγνώριση-ανίχνευση-ταξινόμηση αντικειμένων

Οι παράγοντες οι οποίοι ευθύνονται για λάθη που μπορεί να προκύψουν κατά την αναγνώριση-ανίχνευση-ταξινόμηση αντικειμένων αναφέρονται παρακάτω:

1. Σημείο θέασης: Αναφερόμαστε στην θέση του αντικειμένου, στην οποία ο ανιχνευτής πρέπει να έχει την δυνατότητα να το αναγνωρίζει ανεξάρτητα από τις εναλλαγές της θέσης του μέσα στην εικόνα.

2. Περιστροφή: Έχοντας την δυνατότητα περιστροφής οποιαδήποτε εικόνας σε σχέση με την αρχική μορφή, πρέπει ο ανιχνευτής να μπορεί να αναγνωρίζει το αντικείμενο ανεξάρτητα από την κατεύθυνση στην οποία βρίσκεται.

3. Κλιμάκωση: Η αναγνώριση του αντικειμένου πρέπει να γίνεται ανεξάρτητα από την αλλαγή μεγέθους της εικόνας βάσει της αρχικής.

4. Φωτισμός: Ο φωτισμός, τεχνητός ή φυσικός παίζει σημαντικό ρόλο στην ανίχνευση ενός αντικειμένου. Ο ανιχνευτής πρέπει να μπορεί να ανιχνεύσει το αντικείμενο στόχο ανεξαρτήτως του φωτισμού.

5. Κατοπτρισμός: Κάθε είδωλο από το καθρεπτισμένο αντικείμενο, ο ανιχνευτής θα πρέπει να είναι σε θέση να το ανιχνεύει.

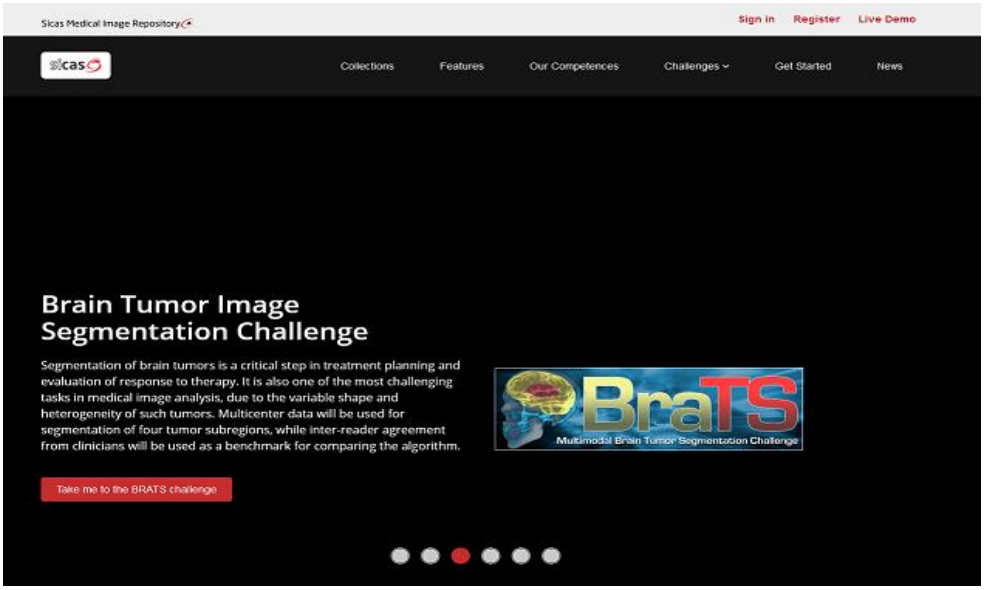
6. Ορατότητα: Στην περίπτωση που το αντικείμενο προς ανίχνευση δεν είναι ευδιάκριτο ή κρύβεται πίσω από άλλα αντικείμενα, ο ανιχνευτής θα πρέπει να είναι σε θέση να το αναγνωρίζει.

7. Δονήσεις: Παραμορφώσεις σε ροή εικόνας ή βίντεο που μπορεί να έχουν δημιουργηθεί από δονήσεις ή ασταθής λήψεις μπορεί να ωθήσουν τον ανιχνευτή σε λανθασμένα αποτελέσματα.

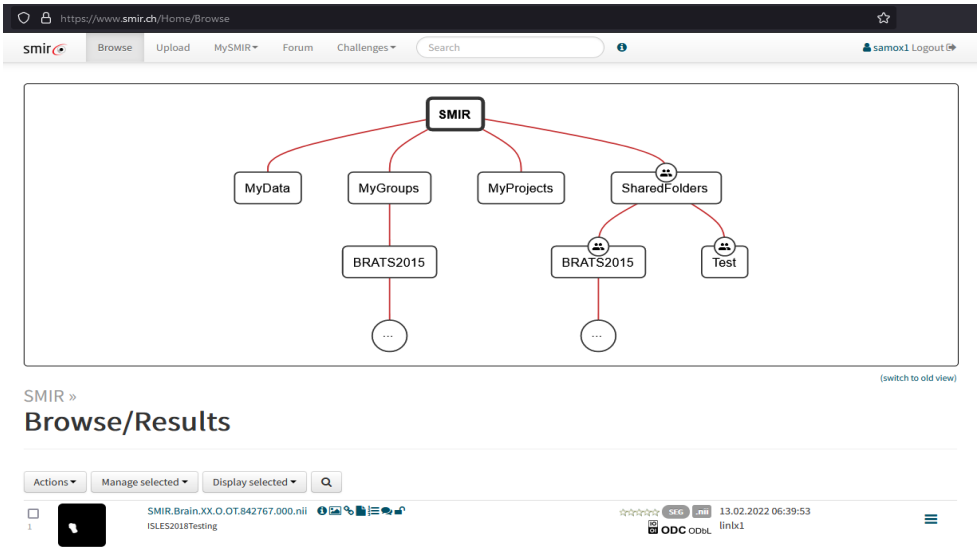
2.6 Αποθήκες δεδομένων

Ο όρος αποθήκες δεδομένων αναφέρεται σε μια βάση δεδομένων που χρησιμοποιείται για αναφορά και ανάλυση. Το σύνολο δεδομένων που είναι αποθηκευμένα αποτελούνται από διάφορα χαρακτηριστικά ανάλογα με τον σκοπό που εξυπηρετούν. Μέχρι σήμερα έχουν δημιουργηθεί πολλές αποθήκες δεδομένων, καθεμία επικεντρωμένη στο στοιχείο της. Κάποιες από αυτές είναι η ImageNet, OASIS, TCIA (The Cancer Imaging Archive) και Smir. Πιο συγκεκριμένα η Smir[1] (Sicas Medical Image Repository) είναι μια βάση ιατρικών γνωματεύσεων-δεδομένων που έχει δημιουργηθεί για ιατρικούς σκοπούς και έρευνες. Τα δεδομένα της αποθήκης αυτής εξάγουν και καταγράφουν χαρακτηριστικά τα οποία είναι αναγκαία για την διαδικασία της ανίχνευσης αντικειμένου, η οποία γίνεται όλο και πιο ακριβής ανάλογα με το πόσο ποιοτικά είναι τα χαρακτηριστικά που θα εξαχθούν από τον αλγόριθμο και συνεπώς του μοντέλου που θα δημιουργηθεί. Κάποια από τα στοιχεία-χαρακτηριστικά είναι το αντικείμενο που απεικονίζει, το μέγεθος κ.λπ.. Όσο περισσότερα δεδομένα δέχεται σαν παραμέτρους για

την εξίσωση, τόσο καλύτερες προσεγγίσεις θα προκύψουν μέσω της εκπαίδευσης καθώς αυτή είναι λογική πίσω από την υπολογιστική όραση.



Εικόνα 8: Ιστοσελίδα Smir (Sicas Medical Image Repository)



Εικόνα 9: Περιήγηση στα σύνολα δεδομένων του Smir

1. https://www.smir.ch

3. Τεχνικές προσέγγισης στην αναγνώριση στόχου-αντικειμένου

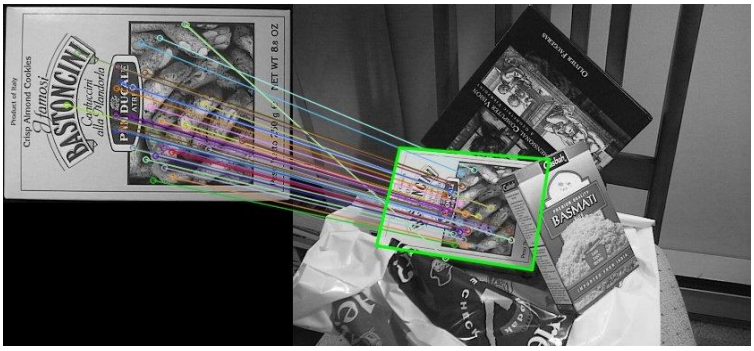
Βάσει των προαναφερθέντων "προβλημάτων" κατά την αναγνώριση (κεφάλαιο 2.5) ενός αντικειμένου ή την παρακολούθησή του, καθίσταται αναγκαία η εξειδίκευση του αλγορίθμου ώστε να είναι σε θέση να ανιχνεύει στόχους, ανεξάρτητα από τις ακολουθίες επεξεργασμένων εικόνων. Δηλαδή μια διαδικασία που δίνει την δυνατότητα στον αλγόριθμο να αναγνωρίζει σε πραγματικό χρόνο αντικείμενα ή στόχους [8]. Στην συνέχεια θα αναλυθούν κάποιες από αυτές τις προσεγγίσεις για την αναγνώριση αντικειμένου.

3.1 Αντιστοίχιση βασισμένη στο χρώμα

Όπως υποδηλώνει και το όνομά της, αναφέρεται σε μια τεχνική η οποία επιδιώκει την ανίχνευση αντικειμένου μέσω των χρωμάτων της εικόνας. Αν όμως οι εικόνες προτύπων διαφέρουν ως προς τα χαρακτηριστικά (π.χ. φωτισμός) από τις εικόνες που επιδιώκεται η ανίχνευση, τότε η αποτελεσματικότητα μειώνεται σε σημαντικό βαθμό [7].

3.2 Μέθοδος ανίχνευσης βασισμένη σε χαρακτηριστικά

Η μέθοδος αυτή συγκρίνει τα χαρακτηριστικά που εξάγονται από την εικόνα εισόδου με τα χαρακτηριστικά που έχουν εξαχθεί από την εικόνα πρότυπο του μοντέλου. Η μια εικόνα θεωρείται ως πηγή και η άλλη ως στόχος και η τεχνική αντιστοίχισης χαρακτηριστικών χρησιμοποιείται είτε για την εύρεση, είτε για την εξαγωγή και την μεταφορά χαρακτηριστικών από την εικόνα πηγής στην εικόνα προορισμού [26].



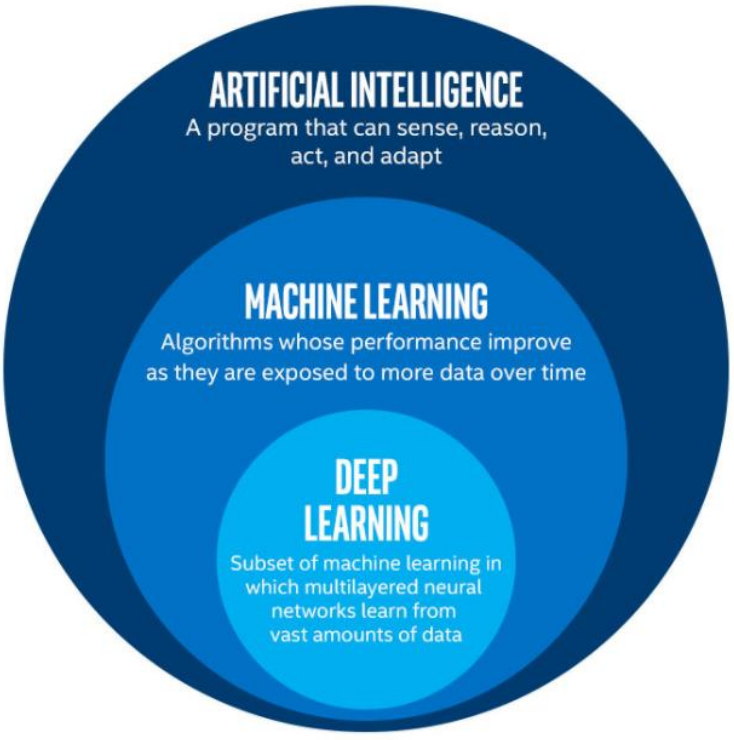
Εικόνα 10: Ανίχνευση βασισμένη σε χαρακτηριστικά

3.3 Μέθοδος ανίχνευσης βασισμένη στο ταίριασμα προτύπων

Η μέθοδος ανίχνευσης βασισμένη στο ταίριασμα προτύπου είναι μία από τις πιο απλές διαδικασίες αναγνώρισης προτύπου. Κατά την υλοποίηση αυτής της μεθόδου γίνεται μια διαδικασία ταιριάσματος αποθηκευμένων εικόνων προτύπων με μια εικόνα εισόδου. Ταξινομούνται δεδομένα ως προς τα χαρακτηριστικά και τις πληροφορίες που αποκτούνται από τα πρότυπα και την αναπαράστασή τους ώστε να πραγματοποιηθεί η κατάλληλη αντιστοίχιση [25].

3.4 Μέθοδος ανίχνευσης βασισμένη στη βαθιά μάθηση

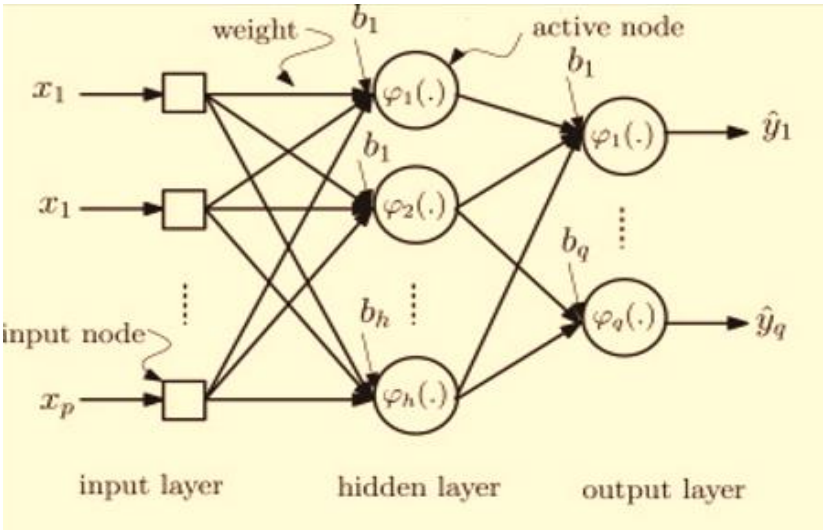
Κάνοντας σημαντικά τεχνολογικά άλματα τα τελευταία χρόνια, η ανίχνευση-αναγνώριση αντικειμένων συνδυάστηκε με την υπολογιστική όραση και την τεχνητή νοημοσύνη [11]. Αυτό διότι πιο εξελιγμένες αρχιτεκτονικές όπως δίκτυα τα οποία χρησιμοποιούν deep learning αλγορίθμους που λειτουργούν βασιζόμενοι σε μεγάλες αποθήκες δεδομένων για την εκπαίδευση γίνονται πιο αποδοτικοί από κάθε μέθοδο ανίχνευσης.



Εικόνα 11: Διαφορές εννοιών AI – Machine Learning – Deep Learning

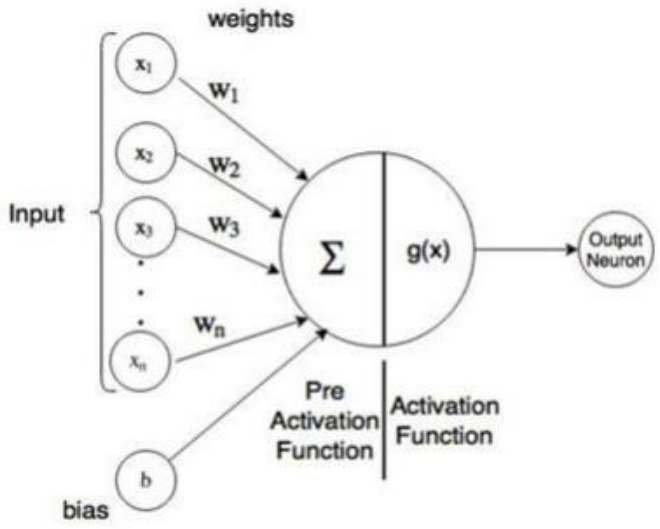
Όπως παρατηρείται στην εικόνα 11, το deep learning είναι ένα υποπεδίο του machine learning που στοχεύει σε τεχνικές οι οποίες λειτουργούν με νευρώνες προσπαθώντας να μιμηθούν την λειτουργία του ανθρώπινου εγκεφάλου. Δηλαδή αναφέρεται σε νευρώνες οι οποίοι συνδέονται μεταξύ τους και μεταφέρουν την πληροφορία. Κάθε deep learning μοντέλο συνήθως λειτουργεί σε τρία στρώματα διαδοχικά, στέλνοντας την πληροφορία το ένα στο επόμενο.

Τέτοια δίκτυα, τα οποία χρησιμοποιούνται για την αναγνώριση προτύπων-αντικειμένων είναι τα νευρωνικά δίκτυα. Ένας από τους βασικούς και απλούς τύπους νευρωνικών δικτύων είναι τα νευρωνικά δίκτυα πολλαπλών στρωμάτων (feed forward neural networks). Σε αυτό το είδος των δικτύων οι συνδέσεις των κόμβων δεν σχηματίζουν κύκλο (υπάρχουν επίσης δίκτυα τα οποία ονομάζονται recurrent neural networks, τα οποία έχουν την δυνατότητα και δομή ώστε να μπορούν οι κόμβοι να συνδέονται ακόμα και κυκλικά). Ένα feed forward neural network αποτελείται από ένα επίπεδο εισόδου (input layer) το οποίο συνδέεται με τα κρυφά επίπεδα (hidden layers) και στην συνέχεια καταλήγει σε ένα επίπεδο εξόδου (output layer). Κάθε σύνδεση που γίνεται ελέγχεται από το βάρος (weight) το οποίο καθορίζει την ευαισθησία κάθε νευρώνα ως προς τις τιμές εξόδου. Πέρα από το βάρος, κάθε νευρώνας έχει ένα bias number (προκατάληψη) που ενισχύει ή εξασθενεί τις δραστηριότητες των νευρώνων.



Εικόνα 12: Νευρωνικό δίκτυο πολλαπλών στρωμάτων

Στην συνέχεια θα γίνει ανάλυση για το πώς οι παράγοντες που αναφέρθηκαν προηγουμένως (bias number, βάρος κόμβου) επηρεάζουν μαθηματικά τις αποφάσεις του νευρωνικού δικτύου. Στην εικόνα 13 παρουσιάζονται οι μεταβλητές και οι συναρτήσεις μέσω μιας εικονικής αναπαράστασης ενός νευρώνα.



Εικόνα 13: Μονάδα νευρωνικού δικτύου

1. Pre-Activation Function (Συνάρτηση προ-ενεργοποίησης εισόδου)

Η συνάρτηση που αντιστοιχεί στην προ-ενεργοποίηση είναι,

$$a(x) = b + \mathbf{w}^T\mathbf{x}$$

α → Λειτουργία της προ-ενεργοποίησης

x → Διάνυσμα εισόδου

b → bias (προκατάληψη): Σε περίπτωση που δεν υπάρχουν είσοδοι, το b θα είναι η τιμή του νευρώνα. Η τιμή της προκατάληψης είναι μια αρνητική τιμή του threshold (κατώφλι), που σημαίνει μια οριακή τιμή ώστε να μπορέσει ο νευρώνας να ενεργοποιηθεί.

w → Βάρη συνδέσεων: Συμβολίζει την δύναμη των συνδέσεων μεταξύ των κόμβων.

2. Activation Function (Συνάρτηση ενεργοποίησης-εξόδου)

Η συνάρτηση που αντιστοιχεί στην ενεργοποίηση των νευρώνων είναι,

$$h(x) = g(a(x)) = g(b + \mathbf{w}^T\mathbf{x})$$

χρησιμοποιεί τιμές από την προ-επεξεργασία για τον υπολογισμό της ενεργοποίησης.

x → Διάνυσμα εισόδου

h → Τελική απόφαση-αποτέλεσμα του νευρώνα

g → Λειτουργία-είδος ενεργοποίησης (linear, sigmoid, hyperbolic, ReLU)

Όταν τα νευρωνικά δίκτυα χρησιμοποιούνται με σκοπό την αποτελεσματικότερη αναγνώριση, είναι αναγκαία η εκπαίδευση του μοντέλου. Με την έννοια εκπαίδευση του μοντέλου εννοούμε την ελαχιστοποίηση της συνάρτησης κόστους του δικτύου. Μια κλασική τεχνική που χρησιμοποιείται για αυτό τον λόγο είναι η τεχνική back propagation (επανα-προσανατολισμός), στην οποία τροποποιούνται τα βάρη συνδέσεων των κόμβων x και x-1 και οι τάσεις των νευρώνων x-1 βάσει των σφαλμάτων στην τιμή του νευρώνα στο στρώμα x.

Όταν όμως γίνεται αναφορά σε regular neural networks για την αναγνώριση εικόνας, η είσοδος στο δίκτυο θα είναι μια τιμή pixel για κάθε pixel της εικόνας, με σκοπό το στρώμα εισόδου να έχει έναν κόμβο για κάθε pixel που δέχεται σαν είσοδο. Αυτό θα είχε σαν αποτέλεσμα να δημιουργούνται χιλιάδες νευρώνες για κάθε εικόνα που δεν είναι αρκετά αποδοτικό, γι’ αυτό το λόγο δεν είναι η κατάλληλη επιλογή για την αναγνώριση εικόνας.

4. Συνελικτικά νευρωνικά δίκτυα και ανιχνευτές υψηλής τεχνολογίας CNN

4.1 Συνελικτικό Νευρωνικό Δίκτυο CNN

Τα Συνελικτικά Νευρωνικά Δίκτυα (CNN) είναι ένας τύπος δικτύων που σε αντίθεση με τα regular neural networks, χρειάζεται ελάχιστη προ-επεξεργασία. Ο πιο συνήθης σκοπός λειτουργίας τους είναι η αναγνώριση εικόνων μετατρέποντας-χωρίζοντας την αρχική σε στρώματα, ώστε κάθε φορά που εντοπίζει κάποιο στόχο, ενεργοποιείται μια σειρά από στρώματα νευρώνων που κάθε ένα από αυτά θα έχει ως στόχο την ανίχνευση ενός συνόλου χαρακτηριστικών [24].

Ένα Συνελικτικό Νευρωνικό Δίκτυο χωρίζεται σε δύο επιμέρους τμήματα, το ένα τμήμα είναι αυτό **της εκμάθησης χαρακτηριστικών** (Convolution, ReLU, pooling) και το δεύτερο είναι **της ταξινόμησης** (Fully Connected Layer-F.C., Softmax - Sigmoid activation function).

Βήματα επιπέδου εκμάθησης χαρακτηριστικών

Στο στρώμα εισαγωγής γίνεται η είσοδος μιας εικόνας και των ακόλουθων διαστάσεών της (πλάτος, ύφος, βάθος) ως ένας πίνακας από τις τιμές των pixels της εικόνας για να τροφοδοτηθούν τα επόμενα επίπεδα.

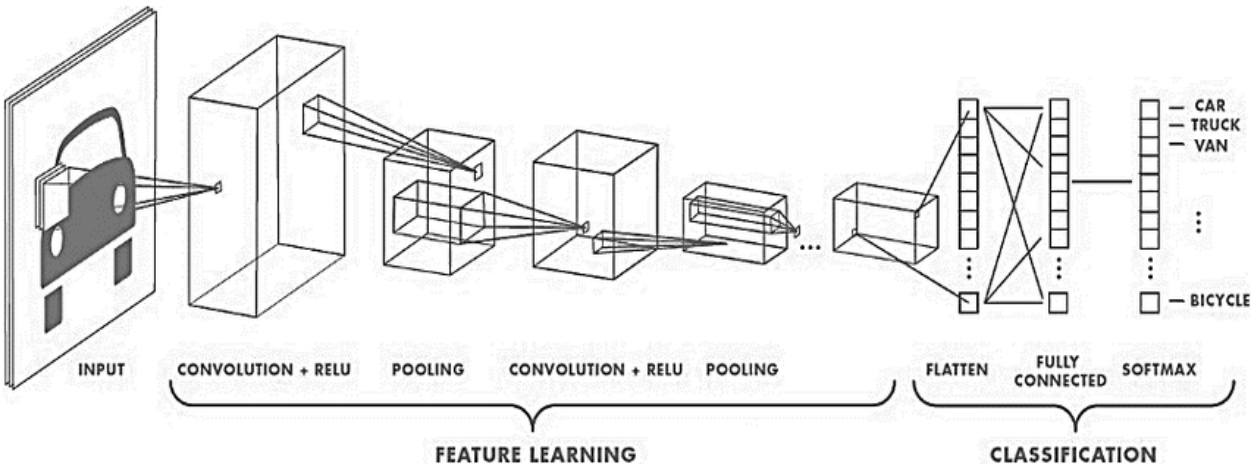
Στην πορεία ακολουθεί το επίπεδο της συνέλιξης στο οποίο γίνεται η εξαγωγή-κωδικοποίηση των χαρακτηριστικών της εικόνας που δόθηκε σαν είσοδος. Ο υπολογισμός του εσωτερικού γινομένου της εικόνας που δόθηκε σαν είσοδος και του συνελικτικού φίλτρου (Kernel-Filter: πίνακας με σκοπό την εξαγωγή χαρακτηριστικών) μέσω της συνελικτικής συνάρτησης convolution(), τροφοδοτεί ένα χάρτη χαρακτηριστικών ή αλλιώς Feature Map (το αποτέλεσμα που δημιουργείται αφού ενεργοποιήσουμε το φίλτρο στην εικόνα εισόδου υπολογίζοντας το εσωτερικό γινόμενο τους). Στην πορεία της διαδικασίας, γίνεται μεταφορά του φίλτρου στο επόμενο πεδίο υποδοχής της ίδιας εικόνας που δόθηκε σαν είσοδος με συγκεκριμένο βήμα (Stride: βήμα μετατόπισης του φίλτρου) με σκοπό να υπολογιστεί ξανά το εσωτερικό γινόμενο του νέου πεδίου υποδοχής (Receptive Field: πεδία υποδοχής τα οποία παρέχουν μονάδες εισόδου στο αντίστοιχο στρώμα και έχει μέγεθος ίδιο με το φίλτρο) και του φίλτρου. Αυτή η διαδικασία γίνεται επανειλημμένα μέχρις ότου να περάσει από το φίλτρο ολόκληρη η εικόνα. Κάθε έξοδος έχει σκοπό να είναι είσοδος για το επόμενο στρώμα.

Στο επίπεδο της μη γραμμικότητας, διαδραματίζεται μια λειτουργία υπερβολικής ή σιγμοειδούς εφαπτομένης. Μια από τις πιο συνήθεις λειτουργίες αυτού του επιπέδου είναι η ενεργοποίηση της συνάρτησης ReLU που εφαρμόζεται μέσω μιας συνάρτησης ενεργοποίησης max(0,x) με σκοπό την μετατροπή των αρνητικών τιμών σε μηδενικές. Στόχος της ενεργοποίησης συνάρτησης μη γραμμικότητας στα CNN είναι η βελτίωση του ταιριάσματος των αποτελεσμάτων, δηλαδή η βελτίωση της ακρίβειας.

Τέλος προκύπτει ένα στρώμα συγκέντρωσης χαρακτηριστικών (Pooling Layer), του οποίου ο βασικός στόχος είναι η επισήμανση του χάρτη χαρακτηριστικών, μειώνοντας τον χώρο διαστάσεων, δημιουργώντας μια καλύτερη αναπαράσταση χαρακτηριστικών. Στο συγκεκριμένο στρώμα γίνεται έλεγχος υπερφόρτωσης σε κάθε μέρος της εισόδου. Σε αυτό το επίπεδο, παράμετροι δεν είναι τα βάρη (weights) και οι προκαταλήψεις (bias numbers) του κάθε νευρώνα, δεν υπάρχει μηδενική συμπλήρωση (Zero Padding: διαδικασία συμμετρικής προσθήκης μηδενικών στον πίνακα εισόδου ώστε να γίνεται προσαρμογή του μεγέθους της με βάση τον αριθμό εξόδων για να μην χάνεται πληροφορία στις άκρες-σύνορα μετά από κάθε στρώμα) και δέχεται ως παραμέτρους το αντίστοιχο φίλτρο της συνέλιξης που χρησιμοποιείται και το βήμα (Stride).

Βήματα επιπέδου ταξινόμησης

Το τελευταίο στάδιο ενός CNN καταλήγει σε κάποια πλήρως συνδεδεμένα στρώματα (Fully Connected Layer-F.C.), των οποίων η λειτουργία είναι η σύνδεση κάθε νευρώνα σε ένα στρώμα με κάθε νευρώνα στο επόμενο. Σε αυτό το τελικό συνδεδεμένο επίπεδο χρησιμοποιείται μια λειτουργία ενεργοποίησης Softmax (είδος κατανεμητή που χρησιμοποιείται για να εκχωρεί δεκαδικές πιθανότητες σε κάθε αντίστοιχη κλάση σε προβλήματα πολλαπλών κλάσεων) ή Sigmoid (είδος κατανεμητή που χρησιμοποιείται από νευρωνικά δίκτυα με σκοπό την κατηγοριοποίηση δυαδικών αποτελεσμάτων, δηλαδή είναι ιδανική σε περιπτώσεις που στο μοντέλο η πιθανότητα ύπαρξης είναι μεταξύ 0 και 1) που στόχο έχει την ταξινόμηση των εξαχθέντων αποτελεσμάτων της εικόνας εισόδου σε ένα σύνολο από διαφορετικές κατηγορίες σύμφωνα με τα δεδομένα εκπαίδευσης.



Εικόνα 14: Αρχιτεκτονική CNN

4.2 Ανιχνευτές Υψηλής Τεχνολογίας CNN

Αφού αναλύθηκε η δομή και η λειτουργία των τυπικών Συνελικτικών Νευρωνικών Δικτύων, θα εξεταστεί και η εξέλιξη αυτών, δηλαδή των R-CNN ανιχνευτών που στην πορεία εξελίχθηκαν στα γρήγορα R-CNN και στην συνέχεια στα γρηγορότερα R-CNN.

4.2.1 R-CNN

Τα R-CNN (Region-based CNN) δίκτυα διαμορφώθηκαν βάσει αρχιτεκτονικής και λειτουργίας των CNN, και στοχεύουν στην αναγνώριση παραπάνω από ενός αντικειμένου σε μια εικόνα, χρησιμοποιώντας μια τεχνική που ονομάζεται **Selective Search** [17]. Πιο συγκεκριμένα η τεχνική αυτή αναγνωρίζει και οριοθετεί μέσα σε ένα ορθογώνιο κάθε αντικείμενο της εικόνας, ενεργοποιεί την αναγνώριση μέσω ενός βελτιωμένου AlexNet (αρχιτεκτονική για αναγνώριση αντικειμένου στην υπολογιστική όραση) αλγορίθμου, και στο τελικό του στάδιο μέσω της χρήσης ενός SVM κατηγοριοποιεί το αντικείμενο και έτσι το νευρωνικό δίκτυο δέχεται ως εισόδους τις περιοχές που ανήκουν τα αντικείμενα-στόχοι και τις επιστρέφει βελτιωμένες. Έχει την δυνατότητα real time αναγνώρισης αλλά είναι αρκετά χρονοβόρο με δέσμευση αρκετής μνήμης.

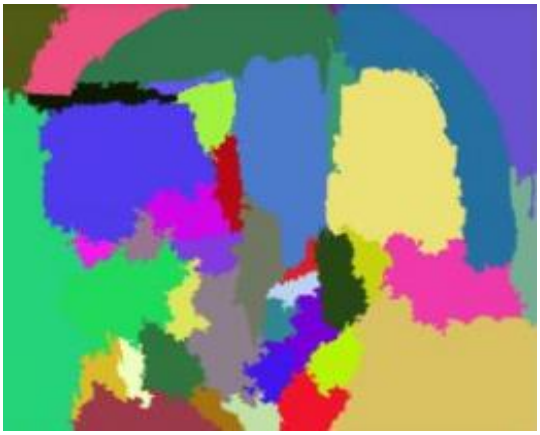
Επεξήγηση Selective Search

Η ύπαρξη κάθε αντικειμένου χαρακτηρίζεται από ποικιλίες σε κλίμακες, σε χρώματα, σε υφές και σε περιβλήματα. Σκοπός της Επιλεκτικής Αναζήτησης είναι να προσδιορίσει τέτοιου είδους μοτίβα στην εικόνα εισόδου και βάσει αυτών να προτείνει πιθανές περιοχές ύπαρξης αντικειμένων. Με μια σύντομη επισκόπηση θα απεικονίσουμε την λειτουργία του Selective Search.

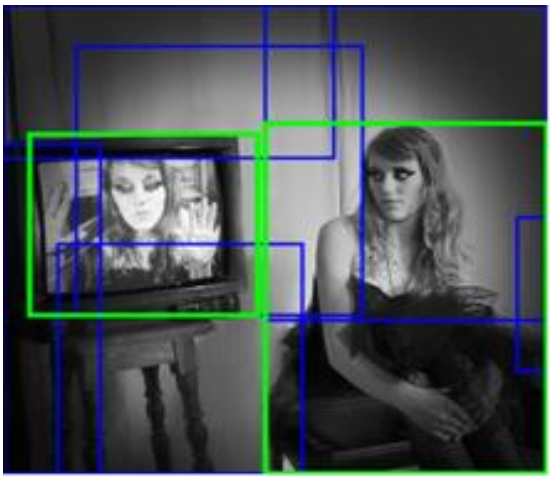
1. Δέχεται μια εικόνα εισόδου.

2. Γίνεται μια διαδικασία τμηματοποίησης της εικόνας ώστε να χωριστεί σε πολλαπλές περιοχές.

3. Αυτή η τεχνική συνδυάζει όλες τις περιοχές αυτές και σχηματίζει μεγαλύτερες περιοχές βάσει της ομοιότητας χρωμάτων, αποχρώσεων, υφής, μεγέθους και συμβατότητας του σχήματος.



4. Αφού ολοκληρωθεί η διαδικασία, έχουν παραχθεί μοτίβα που υποδεικνύουν τις τελικές τοποθεσίες των αντικειμένων που βρίσκονται στην εικόνα εισόδου.

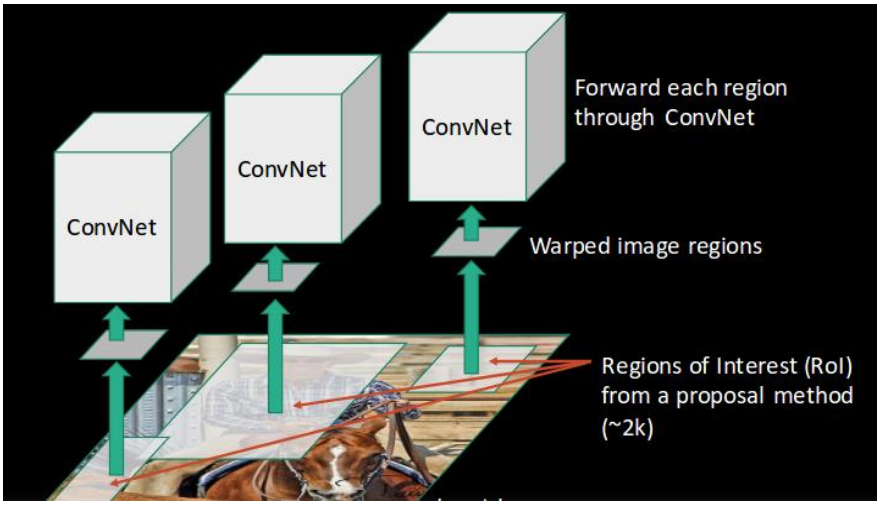


Βήματα R-CNN

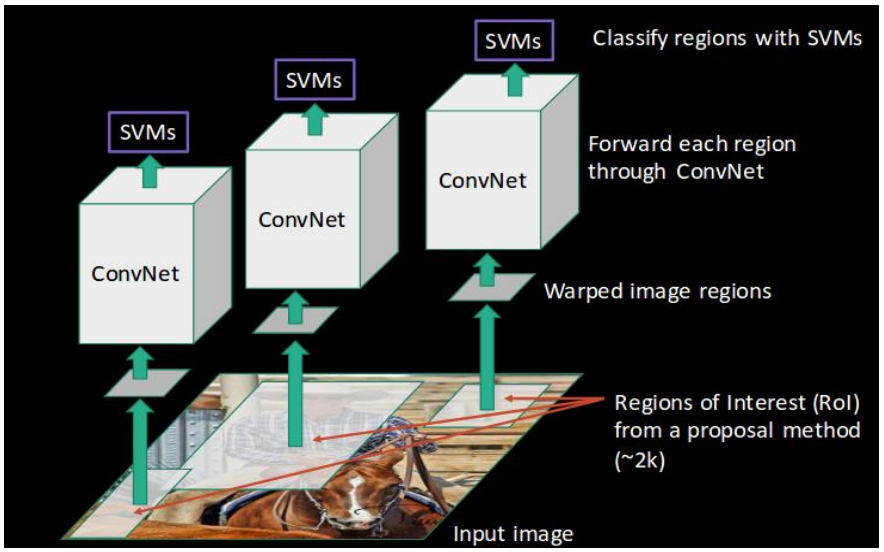
1. Αρχικά γίνεται τροποποίηση ενός προ-εκπαιδευμένου συνελικτικού νευρωνικού δικτύου ώστε το τελευταίο στρώμα να ανιχνεύει βάσει του αριθμού των κλάσεων-αντικειμένων που είναι επιθυμητό κάθε φορά. Μέτα από την τροποποίηση αυτή γίνεται η εισαγωγή της εικόνας εισόδου.

2. Στην συνέχεια λαμβάνονται οι περιοχές ενδιαφέροντος της εικόνας (Regions of Interest-RoI) (περίπου 2 χιλιάδες περιοχές) μέσω της διαδικασίας Selective Search που αναλύθηκε προηγουμένως.

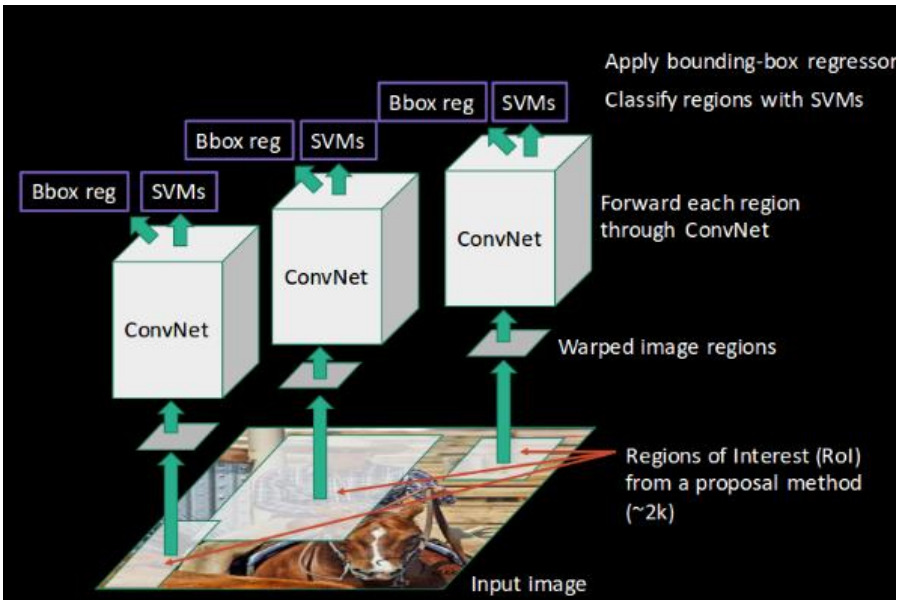
3. Έπειτα διαμορφώνονται αυτές οι περιοχές ώστε να συμβαδίζουν με το μέγεθος εισόδου του CNN που θα εφαρμοστεί.

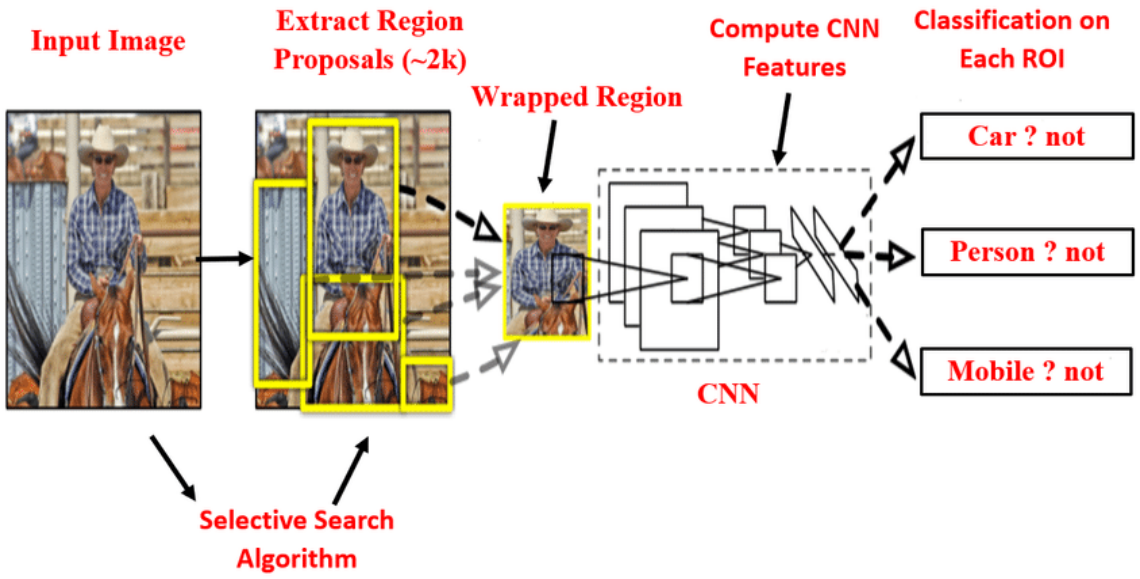


4. Αφού εξαχθούν μέσω του CNN τα χαρακτηριστικά από κάθε περιοχή ενδιαφέροντος, χρησιμοποιείται ένας ταξινομητής SVM για την ταξινόμηση των αντικειμένων.



5. Στο τελικό στάδιο της διαδικασίας χρησιμοποιείται ένα μοντέλο παλινδρόμησης οριοθέτησης πλαισίου (bounding box regressor) για την αυστηρότερη πρόβλεψη των πλαισίων που εμφανίζεται αντικείμενο.





Εικόνα 15: Αρχιτεκτονική R-CNN

Μειονεκτήματα R-CNN

Παρότι τα R-CNN δίκτυα βοήθησαν αρκετά στην ανίχνευση αντικειμένων, η τεχνική αυτή παρουσιάζει προβλήματα:

- Εξάγονται περίπου 2000 περιοχές ενδιαφέροντος κατά την λειτουργία της επιλεκτικής αναζήτησης (Selective Search).
- Γίνεται εξαγωγή χαρακτηριστικών με χρήση CNN για κάθε περιοχή ενδιαφέροντος, δηλαδή για περίπου 2000 περιοχές ενδιαφέροντος εκτελείται και ένα CNN για εξαγωγή χαρακτηριστικών.
- Η διαδικασία του R-CNN χωρίζεται σε τρία μέρη. Στο πρώτο μέρος το CNN για την εξαγωγή χαρακτηριστικών, στο δεύτερο μέρος ένας γραμμικός ταξινομητής (SVM-Support Vector Machine) για την ταξινόμηση αντικειμένων και στο τρίτο μέρος ένα μοντέλο παλινδρόμησης (regression model) για την ακρίβεια των πλαισίων οριοθέτησης.

Όλα τα παραπάνω κατατάσσουν τον R-CNN σε ένα αργό ανιχνευτή καθώς χρειάζεται περίπου 40-50 sec για τις προβλέψεις κάθε εικόνας, με αποτέλεσμα να μην μπορεί να αντιμετωπίσει μεγάλο όγκο δεδομένων.

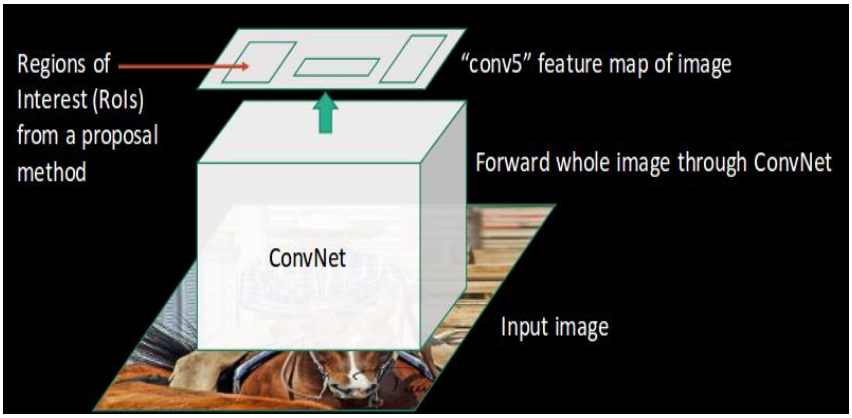
4.2.2 Fast R-CNN

Κατά την πορεία της εξέλιξης αυτών των ανιχνευτών υλοποιήθηκε και το Fast R-CNN με εμπνευστή τον Ross Girshick [15]. Ο σκοπός της δημιουργίας αυτού ήταν η αντιμετώπιση των προβλημάτων που έκαναν το R-CNN αργό. Ο συγκεκριμένος ανιχνευτής λειτουργεί ως εξής: Ξεκινώντας, η εικόνα εισόδου περνάει από ένα CNN και γίνεται Pooling σε κάθε περιοχή αντικειμένου (RoIPool-Region of Interest Pooling). Έτσι αντίθετα με τον R-CNN ανιχνευτή στον οποίο θα πραγματοποιούνταν περίπου 2000 περάσματα της εικόνας εισόδου, απαιτείται μόνο ένα με την χρήση του Fast R-CNN. Επίσης, στο στάδιο της κατηγοριοποίησης το οποίο γίνεται με την χρήση ενός Softmax Classifier που έχει είσοδο το αποτέλεσμα του RoIPool, επιτυγχάνεται και η παλινδρόμηση με σκοπό την βελτίωση των ορθογωνίων περιγραμμάτων. Πιο συγκεκριμένα, η κατηγοριοποίηση σε ένα Fast R-CNN δίκτυο γίνεται ενεργοποιώντας και τα τρία μοντέλα μέσα σε ένα κοινό δίκτυο (το SVM, το CNN για ανίχνευση χαρακτηριστικών, το μοντέλο παλινδρόμησης για την βελτίωση των ορθογωνίων περιγραμμάτων) αντί να τα εκπαιδεύσει ξεχωριστά, με σκοπό την επίτευξη καλύτερων χρόνων ανίχνευσης και καλύτερη ακρίβεια.

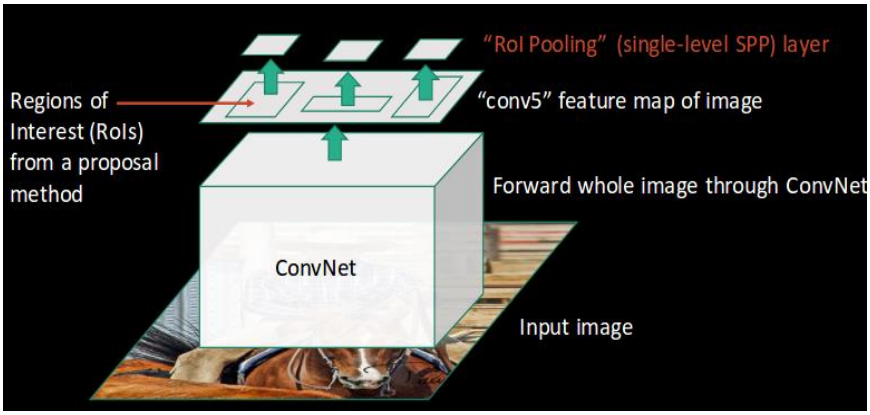
Βήματα Fast R-CNN:

1. Εισάγεται μια εικόνα εισόδου στο μοντέλο.

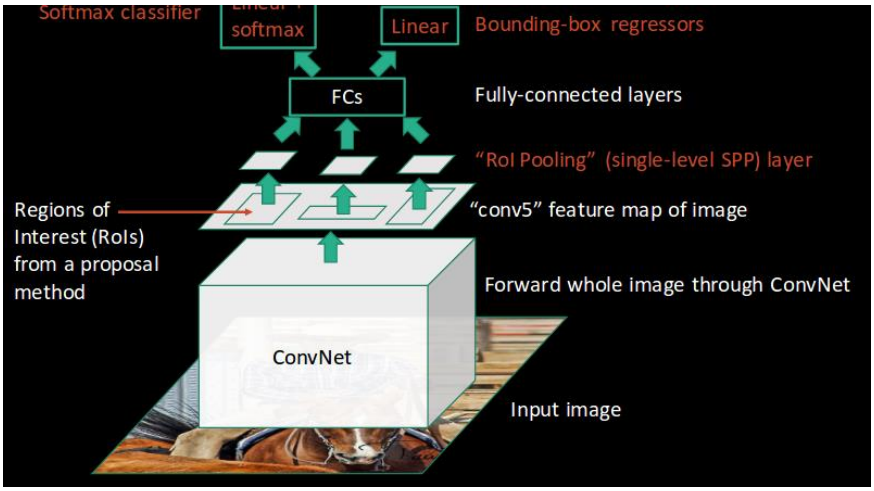
2. Μεταβιβάζεται η εικόνα σε ένα CNN, με σκοπό την δημιουργία περιοχών ενδιαφέροντος (Regions of Interests).



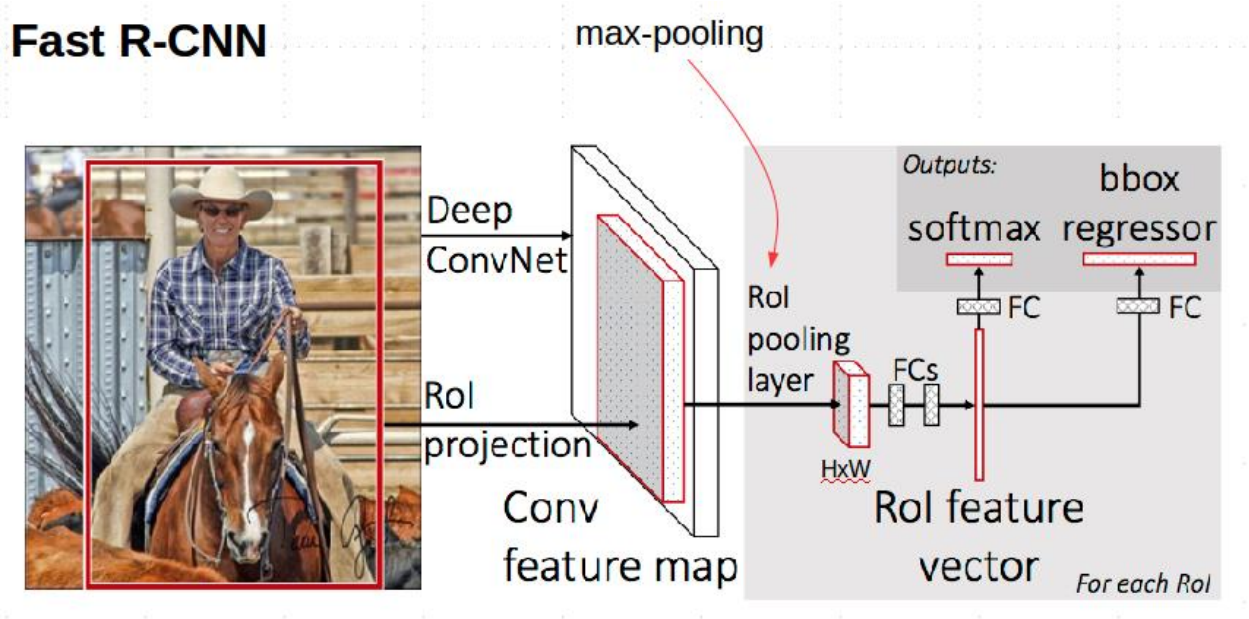
3. Στο επόμενο επίπεδο, ενεργοποιείται ένα στρώμα συγκέντρωσης RoI (Region of interest pooling layer), εφαρμόζεται σε όλες τις περιοχές ενδιαφέροντος από το προηγούμενο βήμα, που έχει σκοπό να αναδιαμορφώσει το μέγεθός τους ανάλογα με την είσοδο που παίρνει το CNN ώστε να μεταβιβαστούν στο πλήρως συνδεδεμένο δίκτυο.



4. Αφού περαστούν όλες αυτές οι περιοχές στο πλήρως συνδεδεμένο δίκτυο (Fully connected layer), χρησιμοποιείται ένας Softmax κατανεμητής, με σκοπό να αντιστοιχήσει τις κλάσεις. Μαζί με αυτόν υπάρχει και ένα επίπεδο γραμμικής παλινδρόμησης (linear regression) που χρησιμοποιείται παράλληλα για την έξοδο οριοθετημένων πλαισίων (bounding boxes) για τις προβλεπόμενες κλάσεις.



Σε αντίθεση με το R-CNN δίκτυο που χρησιμοποιεί 3 διαφορετικές φάσεις, το Fast R-CNN χρησιμοποιεί ένα μοναδικό ενιαίο μοντέλο, ώστε να κάνει εξαγωγή χαρακτηριστικών από τις περιοχές ενδιαφέροντος (Regions of Interest), να ταξινομεί σε διαφορετικές κλάσεις και να επιστρέφει τα όρια πλαισίων για τις αναγνωρισμένες κλάσεις.



Εικόνα 16: Αρχιτεκτονική Fast R-CNN

Μειονεκτήματα Fast R-CNN

Παρότι είναι μια εξέλιξη του R-CNN, παρουσιάζει και αυτό κάποια προβλήματα:

- Χρησιμοποιείται ακόμα η επιλεκτική αναζήτηση (Selective Search) για την εύρεση των περιοχών ενδιαφέροντος, η οποία είναι αρκετά χρονοβόρα.
- Η ανίχνευση γίνεται πολύ πιο γρήγορα από το R-CNN, σε χρόνο 2 sec ανά εικόνα, αλλά σε περιπτώσεις με μεγάλο όγκο δεδομένων, εξακολουθεί να είναι αργό.

4.2.3 Faster R-CNN

Πλέον ένας από τους πιο σύγχρονους ανιχνευτές που είναι διαθέσιμοι σήμερα είναι ο Faster R-CNN, ο οποίος είναι ένας αρκετά γρήγορος region based ανιχνευτής [14]. Η διαφορά με την προηγούμενη έκδοση Fast R-CNN και η αιτία που έγινε γρηγορότερος από τον προκάτοχό του είναι ο

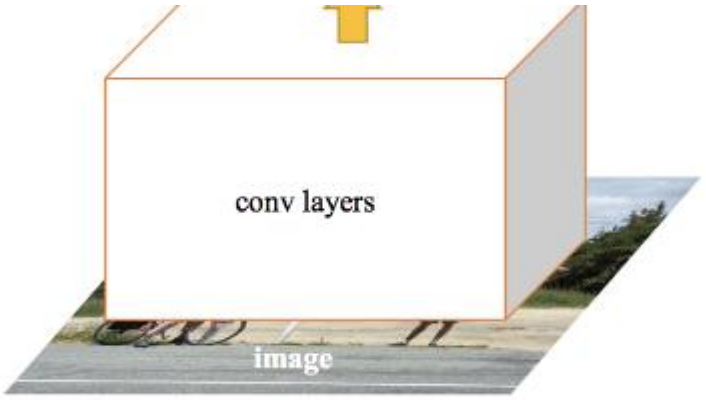
αλγόριθμος που έχει την δυνατότητα να παραθέτει υποθέσεις σχετικά με τις τοποθεσίες των αντικειμένων που ανιχνεύονται στην εικόνα. Η ιδέα βασίζεται στο συνδυασμού του Fast R-CNN με άλλα περιφερειακά προγράμματα που εφαρμόζονται σε Μονάδα Επεξεργασίας Γραφικών (Graphics Processing Unit-GPU) και μοιράζονται συνελικτικά στρώματα με τον ανιχνευτή. Η πρωτοπορία σε σχέση με το Fast R-CNN είναι η χρήση ενός νευρωνικού δικτύου που ονομάζεται RPN (Region Proposal Network), το οποίο αντικατέστησε την λειτουργία του Selective Search με αποτέλεσμα την μείωση του χρόνου παραγωγής προτάσεων περιοχής.

Επεξήγηση RPN (Region Proposal Network)

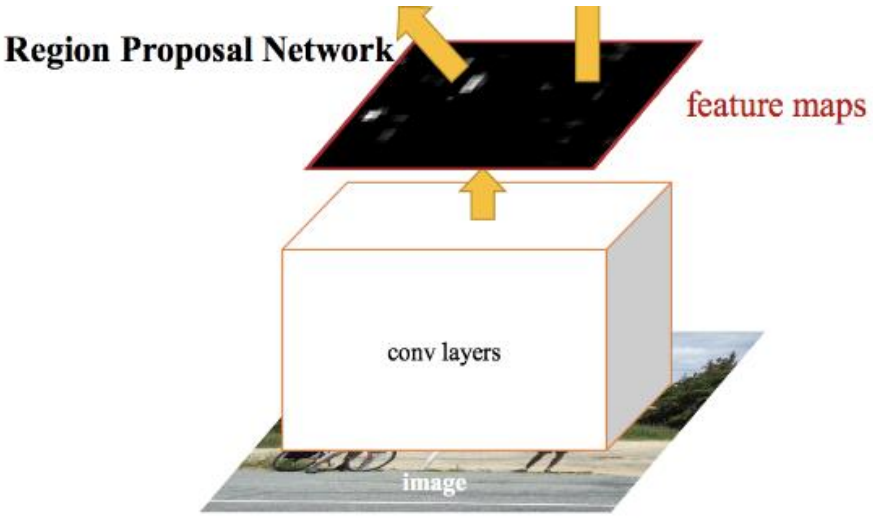
1. Κατά την εκκίνησή του, ο Faster R-CNN δέχεται τους χάρτες χαρακτηριστικών (feature maps) από το CNN στο αρχικό επίπεδο και τα μεταβιβάζει στο RPN.
2. Το RPN χρησιμοποιεί ένα κινούμενο παράθυρο πάνω στους feature maps από το προηγούμενο επίπεδο και δημιουργεί κάποια πλαίσια που ονομάζονται anchors και διαφέρουν σε μέγεθος και σχήμα μεταξύ τους.
3. Ενεργοποιείται ένας ταξινομητής οριοθέτησης (classifier bounding box) ο οποίος ελέγχει την πιθανότητα κάποιο anchor να περιέχει κάποιο αντικείμενο-στόχο, εξετάζοντας την κλάση στην οποία ανήκει και ένας παλινδρομητής οριοθέτησης (bounding box regressor) για την ρύθμιση των anchors ώστε να ταιριάζουν καλύτερα στο αντικείμενο.
4. Αφού έχουν δημιουργηθεί τα οριοθετημένα anchors διαφορετικών σχημάτων και μεγεθών που περνάνε στο RoI (Region of Interest) στρώμα μετά το τέλος του RPN, μπορεί να υπάρχουν προτάσεις περιοχών οι οποίες δεν έχουν αντιστοιχηθεί σε κάποια κλάση.

Βήματα Faster R-CNN:

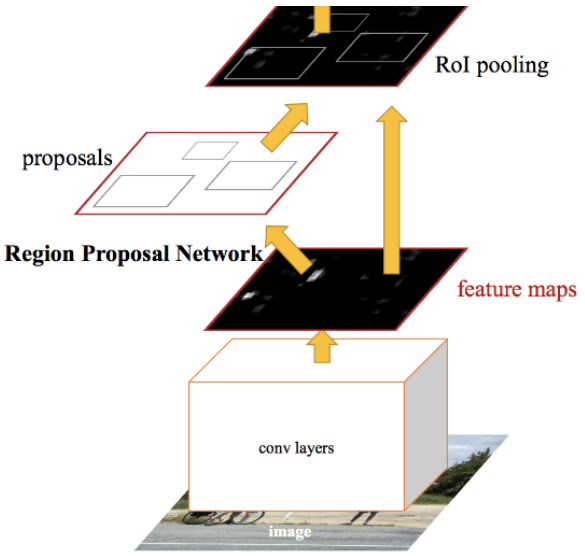
1. Αρχικά γίνεται τροφοδότηση της εικόνας εισόδου σε ένα CNN με σκοπό την δημιουργία του χάρτη χαρακτηριστικών (feature map).



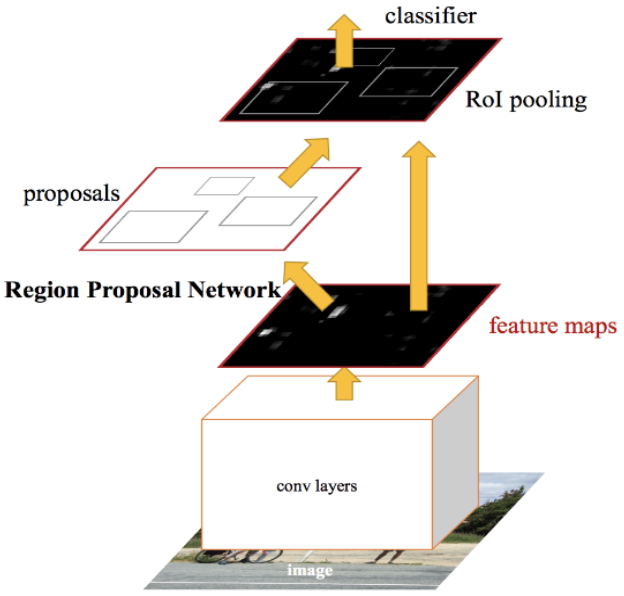
2. Στην συνέχεια το RPN εφαρμόζεται σε αυτόν τον χάρτη χαρακτηριστικών (feature map) ώστε να εξάγει βαθμούς αντικειμενικότητας.

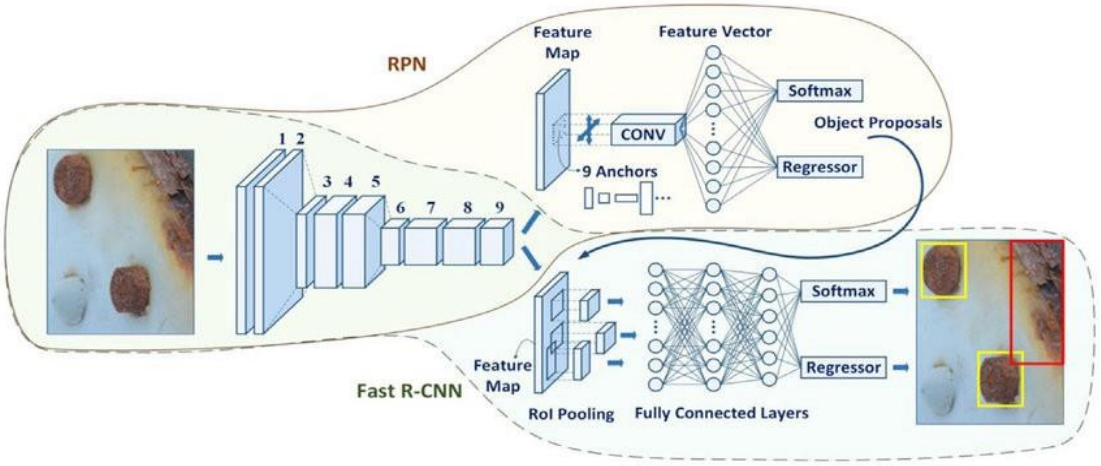


3. Στο επόμενο βήμα εφαρμόζεται ένα RoI pooling layer στις εξαχθείσες προτάσεις ώστε να μειωθούν όλες στο ίδιο μέγεθος.



4. Στο τελικό στάδιο οι προτάσεις περνάνε σε ένα πλήρως συνδεδεμένο επίπεδο (Fully Connected Layer) που παρέχει ένα Softmax layer και ένα Linear regression layer (στρώμα γραμμικής παλινδρόμησης) ώστε να ταξινομήσει και να εξάγει τα όρια των πλαισίων για τα αντικείμενα.



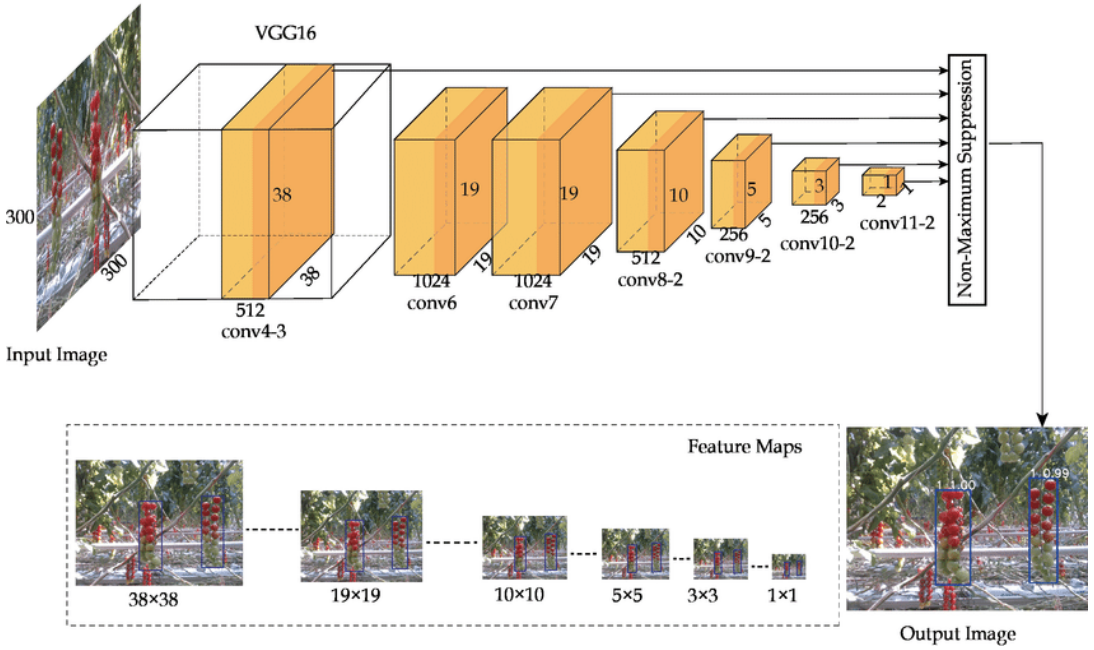


Εικόνα 17: Αρχιτεκτονική Faster R-CNN

Όπως φαίνεται και στην εικόνα 17 η οποία αποτυπώνει την αρχιτεκτονική ενός Faster R-CNN ανιχνευτή, η υλοποίηση του Faster R-CNN ανιχνευτή έγινε πράξη συνδυάζοντας ένα νευρωνικό RPN δίκτυο μαζί με ένα Fast R-CNN δίκτυο που χρησιμοποιείται για την ανίχνευση αντικειμένων.

4.2.4 SSD (Single Shot Detector)

Ο αλγόριθμος SSD (Single Shot Detector) [16] έχει μια παρόμοιας προσέγγισης λειτουργία με αυτήν του αλγορίθμου YOLO που θα παρουσιαστεί στην πορεία.. Ο σχεδιασμός του χωρίζεται σε δύο μέρη, ένα για την εξαγωγή των χαρτών χαρακτηριστικών και ένα για ταξινόμηση των αντικειμένων που εμφανίζονται. Το συγκεκριμένο μοντέλο λειτουργεί ως εξής: η εικόνα εισόδου χωρίζεται σε πλέγματα (grids) διαφόρων μεγεθών και σε κάθε τέτοιο πλέγμα εκτελείται ανίχνευση για διαφορετικές κλάσεις και σε διαφορετικούς λόγους διαστάσεων. Για κάθε ανίχνευση που εκτελείται, εκχωρείται μια βαθμολογία σε κάθε πλέγμα που δηλώνει το πόσο καλά ταιριάζει το αντικείμενο-στόχος στο αντίστοιχο πλέγμα. Τέλος ενεργοποιείται η τεχνική μη μέγιστης καταστολής (Non Maximum Suppression) που στοχεύει στην παραγωγή των τελικών αποφάσεων του ανιχνευτή. Αυτές οι τεχνικές που χρησιμοποιούνται θέτουν τον SSD ικανό για ανίχνευση υψηλής ακρίβειας με χαμηλή ανάλυση εισόδου, αυξάνοντας συνεχώς την ταχύτητά του.



Εικόνα 18: Αρχιτεκτονική SSD

4.2.5 YOLO (You Only Look Once)

Το σύστημα YOLO (You Only Look Once) [1], [2] είναι ένας αλγόριθμος που έχει την δυνατότητα να ανιχνεύσει και να αναγνωρίσει αντικείμενα σε μία εικόνα σε πραγματικό χρόνο. Η επίτευξη της ανίχνευσης αντικειμένων γίνεται μέσω ενός προβλήματος παλινδρόμησης ώστε να παραχθούν πιθανότητες κλάσεων των ανιχνευόμενων αντικειμένων.

Όπως φανερώνει και το όνομά του η λειτουργία του υλοποιείται κάνοντας μια προς τα εμπρός διάδοση-επικοινωνία μέσω ενός νευρωνικού δικτύου για την ανίχνευση αντικειμένων. Αυτό σημαίνει πως η πρόβλεψη για την ύπαρξη αντικειμένων γίνεται σε μια μόνο εκτέλεση του αλγορίθμου. Το CNN χρησιμοποιείται για την πρόβλεψη των πιθανοτήτων κλάσεων αλλά και για την οριοθέτηση πλαισίων ταυτοχρόνως.

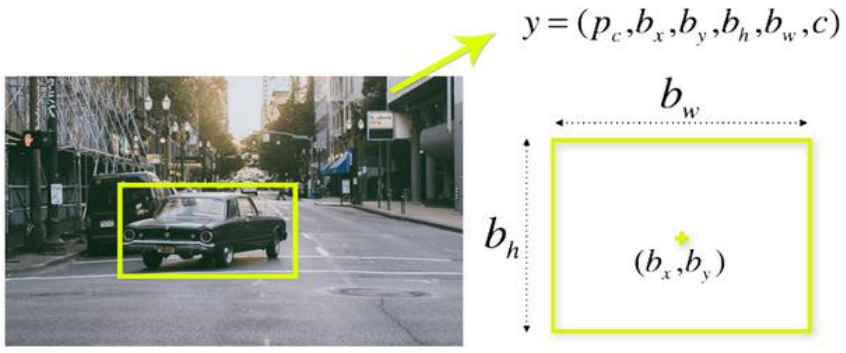
Πιο αναλυτικά, ο αλγόριθμος χρησιμοποιεί τρείς τεχνικές για την ολοκληρωμένη λειτουργία του:

1. **Residual Blocks (Υπολειπόμενα τετράγωνα):** Η τεχνική Residual Blocks έχει σκοπό να χωρίζει την εικόνα εισόδου σε πλέγματα διαστάσεων NxN, ώστε η ανίχνευση αντικειμένων να γίνεται σε κάθε πλέγμα ξεχωριστά. Έτσι σε περίπτωση που ένα αντικείμενο εμφανιστεί στο εσωτερικό ενός πλέγματος, τότε το πλέγμα αυτό είναι υπεύθυνο για την ανίχνευση του συγκεκριμένου

αντικειμένου.

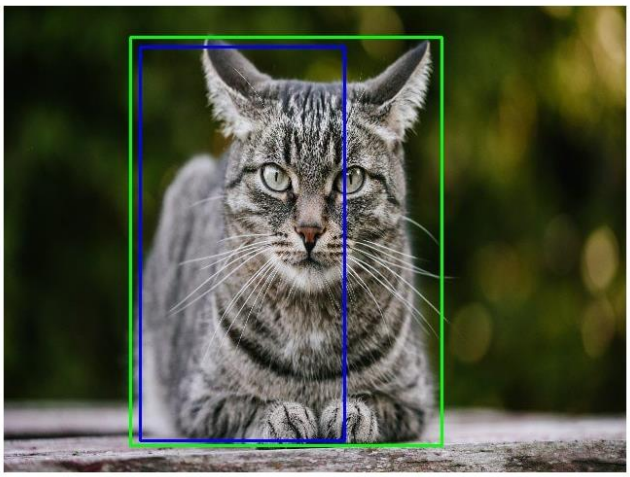
Εικόνα 19: Αναπαράσταση τεχνικής Residual Blocks

2. **<u>Bounding box regression (Παλινδρόμηση οριοθετημένου πλαισίου):</u>** Η τεχνική bounding box regression πραγματοποιεί το μαρκάρισμα-επισήμανση μέσω περιγράμματος ενός αντικειμένου που βρίσκεται στην εικόνα. Κάθε ένα τέτοιο πλαίσιο αποτελείται από το πλάτος (width - bw), το ύψος (height- bh), την κλάση (το είδος του αντικειμένου - C) και το κέντρο οριοθέτησης (bounding box center - bx,by). Η λειτουργία του στον αλγόριθμο αναφέρεται στην πρόβλεψη των παραπάνω χαρακτηριστικών.



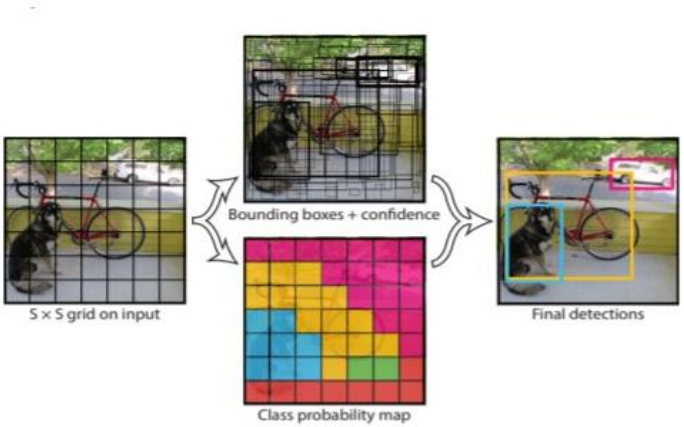
Εικόνα 20: Αναπαράσταση τεχνικής Bounding box regression

3. **Intersection Over Union-IOU (Αλληλοεπικάλυψη):** Η τεχνική Intersection Over Union-IOU περιγράφει τον τρόπο επικάλυψης των πλαισίων. Αυτή η τεχνική χρησιμοποιείται ώστε στην εικόνα εξόδου να υπάρχει ένα πλαίσιο στο οποίο θα περιέχει αψεγάδιαστα όλα τα αντικείμενα. Κάθε ένα κελί είναι υπεύθυνο για την πρόβλεψη των οριοθετημένων πλαισίων και την βαθμολογία εμπιστοσύνης τους. Η τιμή του IOU θα είναι ίση με 1 αν το προβλεπόμενο πλαίσιο είναι ίδιο με το πραγματικό. Σκοπός αυτού είναι η εξάλειψη των οριοθετημένων κουτιών που δεν είναι ίδια με το πραγματικό πλαίσιο.



Εικόνα 21: Αναπαράσταση τεχνικής Intersection Over Union-IOU

Τέλος ο συνδυασμός αυτών των τεχνικών εφαρμόζεται ώστε να παραχθούν τα τελικά αποτελέσματα ανίχνευσης του αλγορίθμου YOLO (You Only Look Once).



Εικόνα 22: Αναπαράσταση συνδυασμών τεχνικών λειτουργίας αλγορίθμου YOLO (You Only Look Once)

5. Λογισμικά που χρησιμοποιήθηκαν για την υλοποίηση του έργου

5.1 Matlab

Το Matlab[2] (matrix laboratory) είναι ένα λογισμικό αριθμητικής υπολογιστικής και γλώσσα προγραμματισμού τέταρτης γενιάς. Ένας από τους βασικούς λόγους ο οποίος χρησιμοποιείται είναι η επίλυση μαθηματικών προβλημάτων αλλά επίσης και στον στατιστικό κλάδο με την χρήση ιστογραμμάτων, τομεογραμμάτων, εμβαδογραμμάτων και άλλα που έχουν να κάνουν με την επεξεργασία εικόνας. Στην συγκεκριμένη εργασία θα χρησιμοποιηθεί για την μετατροπή εικόνων MR τύπου .mha σε αρχεία μορφής .jpg ή .png που είναι αναγκαία για να εκπαιδευτεί ο ανιχνευτής.

5.2 Python

Η Python[3] είναι γλώσσα προγραμματισμού υψηλού επιπέδου, υποστηρίζοντας διαδικαστικό προγραμματισμό (Imperative programming) αλλά και αντικειμενοστραφής (object-oriented programming). Κατασκευάστηκε από τον Guido van Rossum το 1989 και κυκλοφόρησε για πρώτη φορά το 1991. Διαθέτει μεγάλη ποικιλία βιβλιοθηκών και μπορεί να χρησιμοποιηθεί για λειτουργίες μηχανικής μάθησης και τεχνικής νοημοσύνης. Κάποιες από αυτές τις βιβλιοθήκες είναι η NumPy, TensorFlow και Keras. Μειονέκτημά της όμως, επειδή είναι διερμηνευόμενη, κατατάσσεται στις πιο αργές γλώσσες σε αντίθεση με τις μεταγλωττιζόμενες όπως C και C++, και αυτός είναι ο λόγος που δεν είναι κατάλληλη γλώσσα για δημιουργία λειτουργικών συστημάτων.

5.3 Anaconda

Το Anaconda[4] είναι ένα λογισμικό-διανομέας (distributor) ανάπτυξης εφαρμογών σε γλώσσα Python και R που χρησιμοποιείται για επιστήμες δεδομένων, μηχανικής μάθησης, επεξεργασίας δεδομένων. Περιλαμβάνει επεξεργασία δεδομένων μεγάλης κλίμακας, αναλυτικά στοιχεία κ.ά. και έχει την δυνατότητα απλοποίησης στην διαχείριση και την ανάπτυξη πακέτων. Εφόσον η Python είναι μια γλώσσα προγραμματισμού γενικής χρήσης, το Anaconda βοηθά στην ανάπτυξη εφαρμογών στην επιστήμη των δεδομένων, την μηχανική μάθηση, τα ενσωματωμένα συστήματα, την ανάπτυξη ιστού, τον προγραμματισμό δικτύων αλλά και την οπτική των υπολογιστών.

5.4 Cuda-CuDNN

Το Cuda[5] αναφέρεται σε μια παράλληλη πλατφόρμα υπολογιστών και ενός μοντέλου προγραμματισμού αναπτυγμένο από την Nvidia για υπολογιστές με δικές της GPU (Graphic processing Units-Μονάδες επεξεργασίας γραφικών) και δίνει την δυνατότητα στους προγραμματιστές να επιταχύνουν τις εφαρμογές που απαιτούν πολύπλοκες ενέργειες και υπολογισμούς αξιοποιώντας την δύναμη των GPU ως παράλληλο μέρος του υπολογισμού. Δηλαδή στις εφαρμογές οι οποίες έχουν επιταχυνθεί με την χρήση κάποιων GPU, το ένα τμήμα της εργασίας τρέχει στην μονάδα CPU (Central Processing Unit), της οποίας η απόδοσή είναι βελτιστοποιημένη σε ένα λογικό πυρήνα ενώ το τμήμα που καταναλώνει μεγάλο μέρος και ισχύ της εφαρμογής τρέχει σε χιλιάδες πυρήνες της GPU παράλληλα.

Nvidia CuDNN

Το Nvidia CuDNN[6] Deep Neural Network-CuDNN είναι μια βιβλιοθήκη επιταχυνόμενων υπολογισμών μέσω GPU για νευρωνικά δίκτυα. Μέσω του CuDNN παρέχονται εξαιρετικά συντονισμένες εφαρμογές για τυπικές ρουτίνες, όπως στρώματα εμπρόσθιας και προς τα πίσω συνέλιξης (convolution), συγκέντρωσης (pooling), κοινωνικοποίησης και ενεργοποίησης. Το deep learning (βαθιά μάθηση) των ερευνητικών και προγραμματιστικών πλαισίων παγκοσμίως είναι βασισμένο στο cuDNN για την επιτάχυνση GPU υψηλής απόδοσης. Δίνει δηλαδή την ικανότητα στους ερευνητές να επικεντρωθούν στην εκπαίδευση νευρωνικών δικτύων και ανάπτυξη εφαρμογών λογισμικού, αντί να σπαταλούν χρόνο σε συντονισμό επιδόσεων GPU χαμηλού επιπέδου. Ευρέως χρησιμοποιούμενα πλαίσια βαθιάς μάθησης που επιταχύνονται από CuDNN είναι το Keras, TensorFlow, PyTorch κ.ά .

5.5 OpenCV

Το OpenCV[7] (Open Source Computer Vision Library) είναι πλατφόρμα ανοιχτού κώδικα και βιβλιοθήκη λειτουργιών προγραμματισμού με σκοπό την υπολογιστική όραση. Διαθέτει διασυνδέσεις με C++, Java, Python και έχει σχεδιαστεί για υπολογιστικές εφαρμογές. Περιέχει πολλές χρήσιμες λειτουργίες και υλοποιήσεις αλγορίθμων όπως:

- Παρακολούθηση αντικειμένων
- Εργαλεία σχεδίασης
- Επεξεργασία εικόνας
- Είσοδος ροής βίντεο-φωτογραφίας
- Ταξινομητής Haar-cascade (Αλγόριθμος ανίχνευσης αντικειμένων για αναγνώριση

προσώπων)

5.6 TensorFlow

Το TensorFlow[8] είναι βιβλιοθήκη τεχνητής νοημοσύνης ανοιχτού κώδικα που με την λειτουργία γραφημάτων ροής δεδομένων χρησιμοποιείται για την δημιουργία μοντέλων. Η ευέλικτη αρχιτεκτονική του επιτρέπει την εύκολη ανάπτυξη υπολογισμού σε διάφορες μονάδες επεξεργασίας (GPU, CPU, TPU). Αναπτύχθηκε από ερευνητές και μηχανικούς από την ομάδα του Google Brain μέσω του AI της Google. Παρέχει ισχυρή υποστήριξη για μηχανική μάθηση (machine learning), βαθιά μάθηση (deep learning) και ευέλικτο αριθμητικό υπολογισμό. Είναι εξαιρετική επιλογή για δημιουργία Deep Neural Networks (Βαθιά Νευρωνικά Δίκτυα) με δυνατότητες προσαρμογής με ακρίβεια κάθε μεμονωμένου στοιχείου δικτύου.

5.7 LabelImg

Το LabelImg[9] είναι δωρεάν λογισμικό ανοιχτού κώδικα για την υλοποίηση γραφικής επισήμανσης εικόνων. Είναι γραμμένο σε γλώσσα Python και χρησιμοποιεί QT (λογισμικό για υλοποίηση γραφικών διεπαφών) για την γραφική του διεπαφή-interface. Χρησιμοποιείται κυρίως για την τοποθέτηση ετικετών σε εικόνες για την χρήση αυτών σε μοντέλα ανίχνευσης.

2. https://www.mathworks.com/products/matlab.html

3. https://www.python.org/

4. https://www.anaconda.com/

5. https://developer.nvidia.com/cuda-toolkit

6. https://developer.nvidia.com/cudnn

7. https://opencv.org/

8. https://www.tensorflow.org/

9. https://github.com/tzutalin/labelImg

6. Υλοποίηση πειράματος-έργου αναγνώρισης όγκου εγκεφάλου με την χρήση Faster R-CNN αλγορίθμου

6.1 Περιγραφή υλοποίησης

Στόχος αυτής της έρευνας είναι η κατάλληλη εκπαίδευση ενός ανιχνευτή-ταξινομητή αντικειμένων, ώστε να είναι σε θέση να διαχωρίζει το είδος όγκων εγκεφάλου (LGG-HGG) μέσω μαγνητικών τομογραφιών. Στο τέλος αυτής της διαδικασίας, θα έχει δημιουργηθεί ένα μοντέλο το οποίο θα μπορεί να ανιχνεύσει, εντοπίσει και κατηγοριοποιήσει όγκους και να τοποθετήσει πλαίσια γύρω από τα σημεία ανίχνευσης σε ροές εικόνας με τον αντίστοιχο βαθμό εμπιστοσύνης. Η υλοποίηση θα γίνει σε σύστημα με χαρακτηριστικά: CPU: Intel(R) Core(TM) i5-8400 CPU @ 2.80GHz-6 Core(s)-6 Logical Processor(s), GPU: NVIDIA GeForce GTX 1050Ti 4GB, RAM: 8GB.

6.2 Εγκατάσταση κατάλληλων framework, βιβλιοθηκών και πακέτων

Ξεκινώντας, θα γίνει λήψη του ανοιχτού κώδικα λογισμικού TensorFlow Object Detection API από τον σύνδεσμο https://github.com/tensorflow/models/tree/r1.13.0 , ο οποίος απαιτεί συγκεκριμένη δομή καταλόγου που παρέχεται στην αποθήκη του GitHub και θα τοποθετηθεί στον κατάλογο tensorflow1 που θα δημιουργηθεί σε κάποιο αποθηκευτικό μέσο του συστήματος. Έπειτα θα εγκατασταθεί η πλατφόρμα CUDA v10.0 και η βιβλιοθήκη cuDNN 8.

Επόμενο βήμα είναι η λήψη του μοντέλου Faster RCNN Inception V2 με χαρακτηριστικά ρυθμισμένα για αναγνώριση προσώπου από το TensorFlow model zoo, που είναι αποθήκη προ-εκπαιδευμένων μοντέλων ανίχνευσης αντικειμένων. Το αρχείο αυτό θα αποθηκευτεί στον φάκελο που έχει δημιουργηθεί στην θέση C:/tensorflow1/models/research/object_detection.

Επιπλέον θα εγκατασταθεί η βιβλιοθήκη tensorflow-GPU 1.13.1 η οποία δίνει την δυνατότητα χρησιμοποίησης της κάρτα γραφικών ως πρόσθετη ισχύ επεξεργασίας κατά την διάρκεια της εκπαίδευσης και θα γίνει διαμόρφωση του PYTHONPATH όπως θα παρουσιαστεί παρακάτω.

Για την υλοποίηση της έρευνας είναι επίσης αναγκαία η δημιουργία ενός εικονικού περιβάλλοντος (virtual environment) το οποίο θα λειτουργεί ως αυτόνομο περιβάλλον και θα περιέχει συγκεκριμένη έκδοση Python και άλλα επιπλέον πακέτα χωρίς να επηρεάζεται από άλλες εγκαταστάσεις. Αυτό το virtual environment θα δημιουργηθεί με την βοήθεια του Anaconda Prompt, το οποίο είναι

λογισμικό που λειτουργεί ως διαχειριστής πακέτων με διανομή Python, με δυνατότητα χρήσης προ-εγκατεστημένων πακέτων για επεξεργασία αριθμητικών δεδομένων, λύση μαθηματικών προβλημάτων, ακόμα και λειτουργίες μηχανικής μάθησης (NumPy, scipy κ.ά.).

Για την δημιουργία του εικονικού περιβάλλοντος, το οποίο θα είναι υπεύθυνο για την εκπαίδευσή και την εγκατάσταση των απαραίτητων βιβλιοθηκών εκτελείται η εντολή:

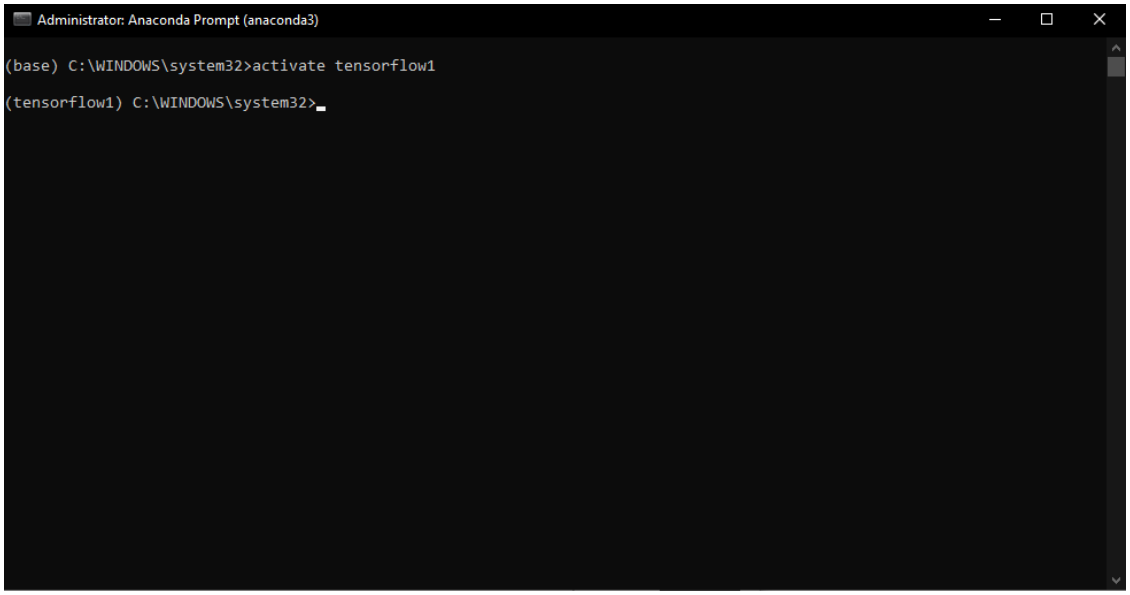
- conda create –n tensorflow1 pip python=3.7

Έπειτα ενεργοποιείται μέσω του Anaconda Prompt το εικονικό περιβάλλον (όπως παρουσιάζεται στην εικόνα 23) και γίνεται εγκατάσταση του tensforlow-gpu με την χρήση των παρακάτω εντολών:

- activate tensorflow1
- pip install tensorflow-gpu==1.13.1

Στην συνέχεια εγκαθίστανται οι υπόλοιπες απαραίτητες βιβλιοθήκες και πακέτα (όπως παρουσιάζεται στην εικόνα 24):

1. Protobuf-Protocol Buffers (ανοιχτού κώδικα μορφή δεδομένων πολλαπλών πλατφορμών για σειριοποίηση δομημένων δεδομένων)
2. Lxml (διαχείριση XML αρχείων)
3. Pillow (επεξεργασία εικόνας)
4. Cython (μεταγλωττιστής για γλώσσα python και επεκτάσεις γλώσσας C)
5. Matplotlib (βιβλιοθήκη σχεδίασης)
6. Opencv-python (υποστήριξη αλγορίθμων που σχετίζονται με Computer Vision, Machine Learning για επεξεργασία αριθμητικών δεδομένων)
7. Pandas (παροχή δομών δεδομένων σχεδιασμένων ώστε να απλοποιούν εργασίες που αφορούν σχεσιακά δεδομένα)



Εικόνα 23: Ενεργοποίηση εικονικού περιβάλλοντος

```
(tensorflow1) C:\> conda install -c anaconda protobuf
(tensorflow1) C:\> pip install pillow
(tensorflow1) C:\> pip install lxml
(tensorflow1) C:\> pip install Cython
(tensorflow1) C:\> pip install contextlib2
(tensorflow1) C:\> pip install jupyter
(tensorflow1) C:\> pip install matplotlib
(tensorflow1) C:\> pip install pandas
(tensorflow1) C:\> pip install opencv-python
```

Εικόνα 24: Εγκατάσταση απαραίτητων βιβλιοθηκών και πακέτων

Σε αυτό το σημείο πρέπει να διαμορφωθεί το PYTHONPATH, το οποίο είναι ουσιαστικά μια μεταβλητή η οποία θα προσθέτει καταλόγους που θα αναζητήσει η Python. Με αυτόν τον τρόπο θα γνωρίζει που να αναζητήσει πακέτα και βιβλιοθήκες για την υλοποίηση. Την διαμόρφωση αυτή την κάνουμε μέσω της εντολής:

- set PYTHONPATH=C:\tensorflow1\models;C:\tensorflow1\models\rese arch;C:\tensorflow1\models\research\slim

Επόμενη διεργασία είναι η υλοποίηση της λειτουργίας των Protobufs (Protocol Buffers), που όπως αναφέρθηκε και προηγουμένως είναι μέθοδος σειριοποίησης (serialization) για δομημένα δεδομένα.

Εκτελείται μεταγλώττιση των Protobufs για να χρησιμοποιηθούν από το TensorFlow για τη ρύθμιση της εκπαίδευσης και των παραμέτρων του μοντέλου ώστε να περιγράφονται τα δεδομένα στην μορφή .proto που χρειάζεται το σύστημα. Έτσι μέσω του Anaconda Command Prompt γίνεται μετάβαση στον κατάλογο C:/tensorflow1/models/research και μέσω της εντολής protoc που κάνει μεταγλώττιση τα αρχεία .proto και δίνει δυνατότητα δημιουργίας κώδικα που μπορεί να επικαλείται ένας αποστολέας ή αποδέκτης του συνόλου των δεδομένων, εκτελείται η εντολή:

- protoc --python_out=. .\object_detection\protos\anchor_generator.proto .\object_detection\protos\argmax_matcher.proto .\object_detection\protos\bipartite_matcher.proto .\object_detection\protos\box_coder.proto .\object_detection\protos\box_predictor.proto .\object_detection\protos\eval.proto .\object_detection\protos\faster_rcnn.proto .\object_detection\protos\faster_rcnn_box_coder.proto .\object_detection\protos\grid_anchor_generator.proto .\object_detection\protos\hyperparams.proto .\object_detection\protos\image_resizer.proto .\object_detection\protos\input_reader.proto .\object_detection\protos\losses.proto .\object_detection\protos\matcher.proto .\object_detection\protos\mean_stddev_box_coder.proto .\object_detection\protos\model.proto .\object_detection\protos\optimizer.proto .\object_detection\protos\pipeline.proto .\object_detection\protos\post_processing.proto .\object_detection\protos\preprocessor.proto .\object_detection\protos\region_similarity_calculator.proto .\object_detection\protos\square_box_coder.proto .\object_detection\protos\ssd.proto .\object_detection\protos\ssd_anchor_generator.proto .\object_detection\protos\string_int_label_map.proto .\object_detection\protos\train.proto .\object_detection\protos\keypoint_box_coder.proto .\object_detection\protos\multiscale_anchor_generator.proto .\object_detection\protos\graph_rewriter.proto .\object_detection\protos\calibration.proto .\object_detection\protos\flexible_grid_anchor_generator.proto

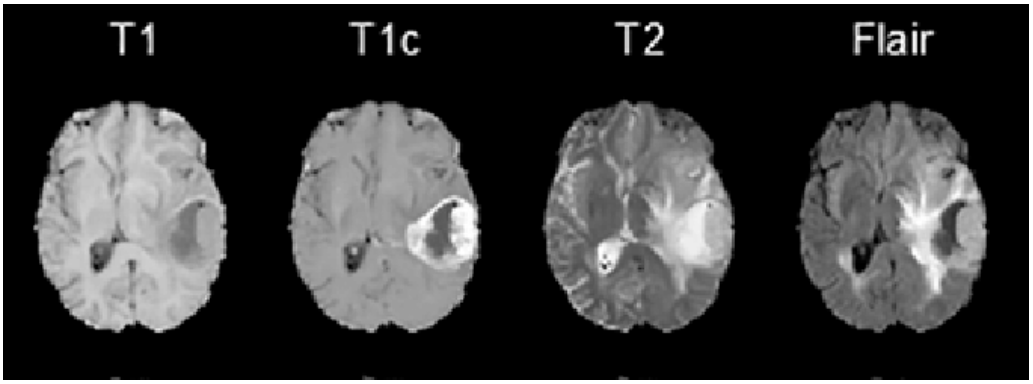
Τέλος εκτελώντας τις παρακάτω εντολές γίνεται η μεταγλώττιση, σύνδεση και εγκατάσταση των πακέτων:

```
(tensorflow1) C:\tensorflow1\models\research> python setup.py build
(tensorflow1) C:\tensorflow1\models\research> python setup.py install
```

Εικόνα 25: Σύνδεση και εγκατάσταση πακέτων

6.3 Εισαγωγή και παραμετροποίηση του συνόλου δεδομένων που θα χρησιμοποιηθεί για την εκπαίδευση του μοντέλου και προσθήκη ετικετών

Αφού ολοκληρωθεί η εγκατάσταση των κατάλληλων βιβλιοθηκών και πακέτων, ακολουθεί η εισαγωγή και παραμετροποίηση των δεδομένων που θα χορηγηθούν στο μοντέλο για την εκπαίδευση. Για την εκπαίδευση θα χρησιμοποιηθεί ένα σύνολο δεδομένων από το διαγωνισμό BraTS 2016, μέσω του SICAS Medical Image Repository που είναι βάση δεδομένων για ιατρικούς σκοπούς και έρευνες. Αποτελείται από μαγνητικές τομογραφίες τύπων T1, T1c, T2, FLAIR οι οποίες είναι τεχνικές διαφοροποίησης στους χρονισμούς των παλμών ραδιοσυχνοτήτων με σκοπό να τονίζουν η καθεμία διαφορετικούς ιστούς στο σημείο που γίνονται (π.χ. λίπος, νερό). Το σύνολο των δεδομένων αποτελείται από μαγνητικές (MRI) με όγκους χαμηλού βαθμού γλοιώματος (LGG) και με όγκους υψηλού βαθμού γλοιώματος (HGG), σε μορφή αρχείου .mha. Η συγκεκριμένη μορφή αρχείου χρησιμοποιείται για την απεικόνιση ιατρικών δεδομένων.



Εικόνα 26: Διαφοροποιήσεις στις απεικονίσεις ιστών T1, T1c, T2, FLAIR

Όπως αναφέρθηκε και στο κεφάλαιο 5.1, το Matlab θα χρησιμοποιηθεί για την μετατροπή αρχείων .mha σε .jpg. Το Matlab διαθέτει τέτοιες συναρτήσεις μέσω πακέτων, οι οποίες είναι κατάλληλες για τέτοιες μετατροπές. Οι συναρτήσεις αυτές παρουσιάζονται παρακάτω:

1. mha_read_volume(): Ανάγνωση αρχείων τύπου .mha.
2. normalize_Images(): Μετατροπή double μεταβλητών σε unit8.
3. Conversion(): Μετατροπή και αποθήκευση αρχείων από .mha σε .jpg .

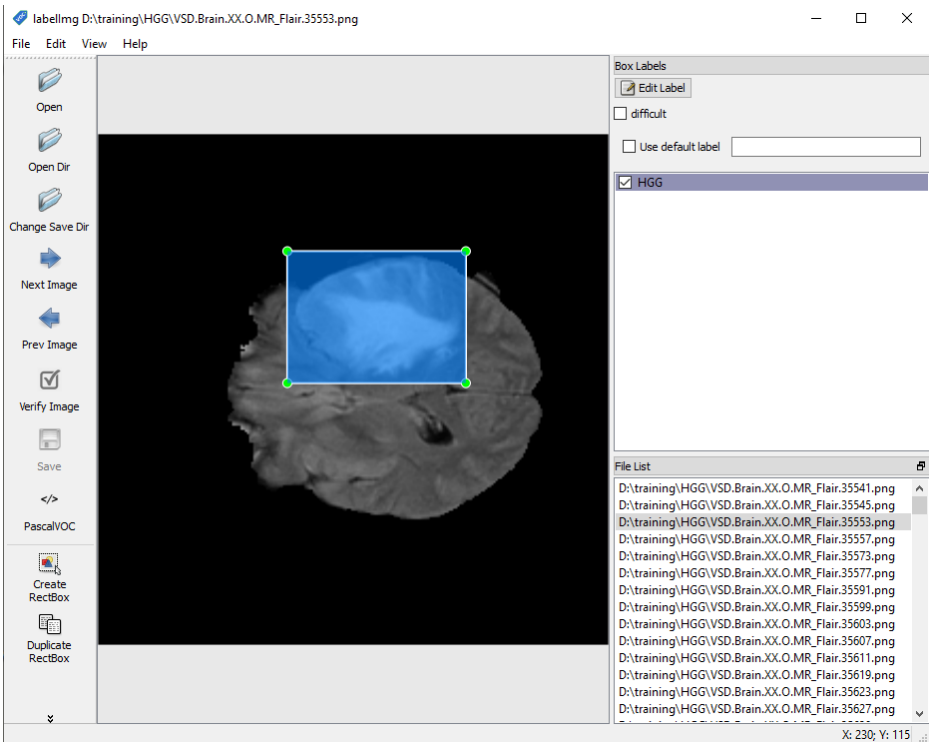
```
function Conversion()
    fileListStruct = dir('*.mha');

    for i = 1: length(fileListStruct)
        sprintf('%d out of %d', i, length(fileListStruct) )
        currentFileName = fileListStruct(i).name;
        currentData = mha_read_volume(currentFileName);
        rawFileName = convertCharsToStrings(extractBetween( currentFileName, 1, ( strlength(currentFileName) - 4 ) ));
        newCurrentData = normalize_Images( double(currentData) );
        imwrite(squeeze(newCurrentData(:,:,round(end/2))), strcat(rawFileName, '.jpg') );
    end
end
```

Εικόνα 27: Συνάρτηση μετατροπής εικόνων από .mha σε .jpg

Σκοπός αυτής της διεργασίας ομαλοποίησης των δεδομένων μέσω των συναρτήσεων που εκτελέστηκαν αλλά και της χειροκίνητης απαλοιφής κάποιων εικόνων, είναι αρχικά να μετατραπούν τα δεδομένα στην μορφή κατάλληλη για τροφοδοσία του μοντέλο και έπειτα η απόρριψη εικόνων οι οποίες δεν διευκολύνουν, διότι δεν είναι ευδιάκριτα τα σημεία στα οποία εμφανίζονται γλοιώματα δίχως γνώσεις πάνω στις επιστήμες της νευρολογίας ή ογκολογίας.

Έτσι το τελικό σύνολο δεδομένων αποτελείται από 450 εικόνες από τις οποίες οι 134 απεικονίζουν LGG γλοίωμα ενώ οι υπόλοιπες 316 απεικονίζουν HGG γλοίωμα. Από αυτές θα δημιουργηθεί ένα σύνολο δεδομένων για την εκπαίδευσή και ένα για την αξιολόγηση. Στο πρώτο θα αντιστοιχεί το 76% του συνόλου ενώ στο δεύτερο το υπόλοιπο 24%. Οι εικόνες αυτές θα αποθηκευτούν στον κατάλογο C:/tensorflow1/models/research/object_detection/images, στον φάκελο /train και /test αντίστοιχα.

Αφού έχει συλλεχτεί το τελικό σύνολο δεδομένων για την εκπαίδευση και την αξιολόγηση του ανιχνευτή, σειρά έχει η προσθήκη ετικετών σε αυτές ή αλλιώς η τιτλοφόρησή τους. Για τον σκοπό αυτό θα χρησιμοποιηθεί το λογισμικό LabelImg, που είναι ένα διαδεδομένο λογισμικό για την δημιουργία τιτλοφόρησης σε εικόνες και χρησιμοποιείται για εφαρμογή ετικετών σε μοντέλα ανίχνευσης αντικειμένων. Η τιτλοφόρηση αυτή αποθηκεύεται σε μορφή XML, όπως θα παρουσιαστεί στις εικόνες παρακάτω. Αυτή η διαδικασία θα γίνει και για τους δύο φακέλους (test, train) στον κατάλογο C:\tensorflow1\models\research\object_detection\images.



Εικόνα 28: Διαδικασία τιτλοφόρησης μέσω LabelImg

```
<annotation>
    <folder>HGG</folder>
    <filename>VSD.Brain.XX.O.MR_Flair.36185.png</filename>
    <path>D:\training\HGG\VSD.Brain.XX.O.MR_Flair.36185.png</path>
    <source>
        <database>Unknown</database>
    </source>
    <size>
        <width>240</width>
        <height>240</height>
        <depth>1</depth>
    </size>
    <segmented>0</segmented>
    <object>
        <name>HGG</name>
        <pose>Unspecified</pose>
        <truncated>0</truncated>
        <difficult>0</difficult>
        <bndbox>
            <xmin>61</xmin>
            <ymin>53</ymin>
            <xmax>162</xmax>
            <ymax>122</ymax>
        </bndbox>
    </object>
</annotation>
```

Εικόνα 29: Μορφή αρχείου XML μετά την τιτλοφόρηση

6.4 Παραγωγή δεδομένων εισόδου και διαμόρφωση εκπαίδευσης

Επόμενο βήμα είναι η δημιουργία των TensorFlow Records. Τα TFRecords είναι τύπος αρχείου του TensorFlow για την αποθήκευση ακολουθιών δυαδικού τύπου. Χρησιμεύουν ως δεδομένα εισόδου στο μοντέλο εκπαίδευσης TensorFlow. Η δομή ενός αρχείου TFRecord είναι ακολουθία δυαδικών χαρακτήρων, που σημαίνει πως πριν γραφτούν στο αρχείο, τα δεδομένα αυτά πρέπει να προσδιοριστούν. Αυτό το αναλαμβάνει ένας protocol buffer του TensorFlow, που είναι ουσιαστικά ένας τύπος "μηνύματος" που χρησιμοποιείται για την αντιστοίχιση ενός αντικειμένου με έναν αναγνωριστικό αριθμό. Έτσι λοιπόν, πρώτο βήμα είναι η χρήση των .xml αρχείων των εικόνων που εξάχθηκαν μέσω της τιτλοφόρησης στο προηγούμενο βήμα για την δημιουργία αρχείων .csv που θα περιέχουν κάθε δεδομένο για την δοκιμή αλλά και για την εκπαίδευση.

Αυτό γίνεται με την χρήση του αρχείου xml_to_csv.py, ώστε να δημιουργηθούν δύο αρχεία τύπου .csv, το test_labels.csv και το train_labels.csv στον φάκελο /images. Έτσι έχουν καταγραφεί σε 2 διαφορετικά .csv αρχεία τα χαρακτηριστικά κάθε εικόνας από το σύνολο των δεδομένων, δηλαδή το όνομα της εικόνας, η κλάση του αντικειμένου, οι διαστάσεις της εικόνας και οι διαστάσεις του ορθογωνίου στο οποίο εμφανίζεται το προς ανίχνευση αντικείμενο.

```
(tensorflow1) C:\tensorflow1\models\research\object_detection>python xml_to_csv.py
Successfully converted xml to csv.
Successfully converted xml to csv.
```

Εικόνα 30: Δημιουργία .csv αρχείων

Επιπλέον πρέπει να ενημερωθεί το αρχείο generate_tfrecord.py σχετικά με τις κλάσεις οι οποίες αφορούν τον ανιχνευτή ως προς την εκπαίδευση.

```
# TO-DO replace this with label map
def class_text_to_int(row_label):
    if row_label == 'LGG':
        return 1
    elif row_label == 'HGG':
        return 2
    else:
        0
```

Εικόνα 31: Τροποποιημένο generate_tfrecords.py για την εκπαίδευση

Αφού γίνει επιτυχώς η μετατροπή των αρχείων .xml σε .csv (εικόνα 30) και ενημερωθεί το αρχείο generate_tfrecord.py (εικόνα 31), θα δημιουργηθούν τα αρχεία train.record και test.record. Για την υλοποίηση αυτού εκτελούνται οι παρακάτω εντολές:

- python generate_tfrecord.py --csv_input=images\train_labels.csv --image_dir=images\train --output_path=train.record
- python generate_tfrecord.py --csv_input=images\test_labels.csv --image_dir=images\test --output_path=test.record

```
(tensorflow1) C:\tensorflow1\models\research\object_detection>python generate_tfrecord.py --csv_input=images\train_label
s.csv --image_dir=images\train --output_path=train.record
2022-01-20 10:37:50.201800: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic
library cudart64_100.dll
Successfully created the TFRecords: C:\tensorflow1\models\research\object_detection\train.record
```

Εικόνα 32: Επιτυχής παραγωγή train records

```
(tensorflow1) C:\tensorflow1\models\research\object_detection>python generate_tfrecord.py --csv_input=images\test_labels
.csv --image_dir=images\test --output_path=test.record
2022-01-20 10:38:42.000385: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic
library cudart64_100.dll
Successfully created the TFRecords: C:\tensorflow1\models\research\object_detection\test.record
```

Εικόνα 33: Επιτυχής παραγωγή test records

Αφού ολοκληρωθεί η δημιουργία των TFRecords, έχει σειρά η υλοποίηση της **αντιστοίχισης των ετικετών** και η **διαμόρφωση της σωλήνωσης ανίχνευσης αντικειμένου** όπου μέσω αυτού θα διασφαλιστεί η αλληλουχία των λειτουργιών της εκπαίδευσης και θα καθοριστούν οι παράμετροι αυτής.

Αντιστοίχιση ετικετών

Ο λόγος της αντιστοίχισης των ετικετών είναι για να ενημερωθεί ο εκπαιδευτής ως προς το είδος της κλάσης κάθε αντικειμένου αντιστοιχίζοντας τα ονόματα της κάθε κλάσης με έναν αναγνωριστικό αριθμό. Για αυτήν την εργασία θα χρησιμοποιηθεί κάποιο πρόγραμμα επεξεργασίας κειμένου ή κώδικα και θα δημιουργηθεί ένα αρχείο labelmap.pbtxt στον κατάλογο C:/tensorflow1/models/research/object_detection/training που θα περιέχει τον αναγνωριστικό αριθμό της αντίστοιχης κλάσης και το όνομα της κλάσης. Ο αναγνωριστικός αριθμός που αναγράφεται στο labelmap.pbtxt πρέπει να είναι ο ίδιος με τους αριθμούς αναγνωριστικών που ορίζονται και στο generate_tfrecord.py.

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

item {
    id: 2
    name: 'HGG'
}
```

Εικόνα 34: Αρχείο labelmap.pbtxt

Διαμόρφωση σωλήνωσης ανίχνευσης αντικειμένου

Τελευταίο βήμα πριν την εκπαίδευση είναι η διαμόρφωση σωλήνωσης της ανίχνευσης αντικειμένου, που όπως αναφέρθηκε και προηγουμένως είναι υπεύθυνη για την αλληλουχία των λειτουργιών της εκπαίδευσης και για τον καθορισμό των παραμέτρων. Από τον κατάλογο C:/tensorflow1/models/research/object_detection/samples/configs γίνεται αντιγραφή του αρχείου faster_rcnn_inception_v2_pets.config στον κατάλογο /training.

Μέσω λογισμικού επεξεργασίας κειμένου-κώδικα γίνονται αλλαγές στο αρχείο .config :

1. Αλλαγή του αναγνωριστικού αριθμού που υποδηλώνει το σύνολο των διαφορετικών κλάσεων που θα ανιχνεύσει το μοντέλο.

```
model {
  faster_rcnn {
    num_classes: 2
```

2. Στην γραμμή 106, γίνεται αλλαγή του path fine_tune_checkpoint, το οποίο έχει τα αντίστοιχα βάρη του μοντέλου για καλύτερη πρόβλεψη.

```
gradient_clipping_by_norm: 10.0
fine_tune_checkpoint: "C:/tensorflow1/models/research/object_detection/faster_rcnn_inception_v2_coco_2018_01_28/model.ckpt"
from_detection_checkpoint: true
```

3. Αλλαγή στην ενότητα train_input_reader και αντίστοιχα στην ενότητα eval_input_reader των paths των μεταβλητών input_path και label_map_path.

   Στην ενότητα train_input_reader:

```
train_input_reader: {
  tf_record_input_reader {
    input_path: "C:/tensorflow1/models/research/object_detection/train.record"
  }
  label_map_path: "C:/tensorflow1/models/research/object_detection/training/labelmap.pbtxt"
}
```

Αντίστοιχα στην ενότητα eval_input_reader:

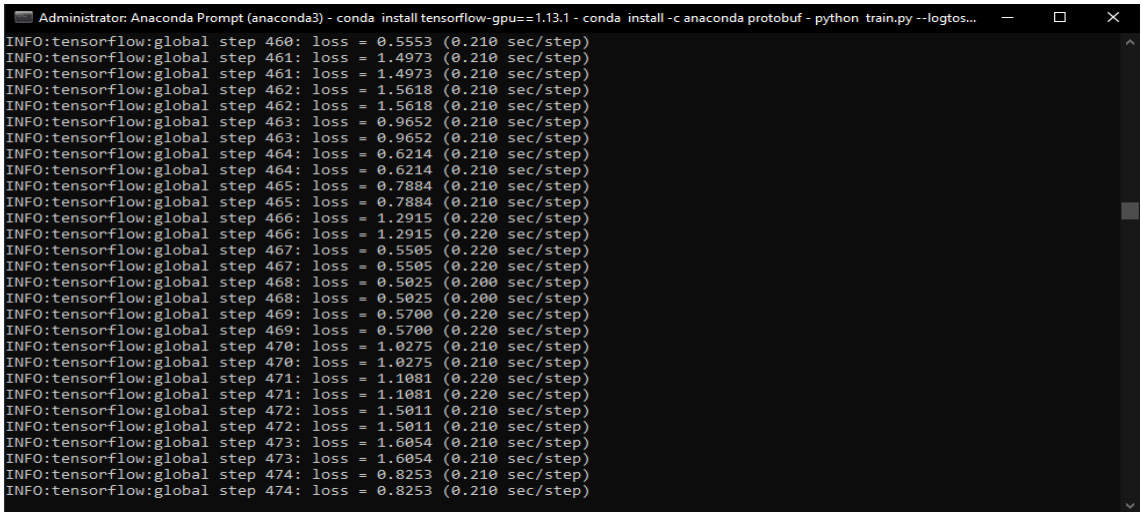
```
eval_input_reader: {
  tf_record_input_reader {
    input_path: "C:/tensorflow1/models/research/object_detection/test.record"
  }
  label_map_path: "C:/tensorflow1/models/research/object_detection/training/labelmap.pbtxt"
  shuffle: false
  num_readers: 1
}
```

6.5 Εκπαίδευση και αξιολόγηση μοντέλου

Αφού ολοκληρωθούν όλα τα βήματα και οι προϋποθέσεις για την εγκατάσταση των απαραίτητων εργαλείων, ξεκινάει η εκπαίδευση του μοντέλου, εκτελώντας την ακόλουθη εντολή:

- python train.py --logtostderr --train_dir=training/ --pipeline_config_path=training/faster_rcnn_inception_v2_pets.config

Κατά την διάρκεια της εκπαίδευσης, όπως παρατηρείται και στην εικόνα 32, καταγράφεται η απώλεια που υπάρχει σε κάθε βήμα. Για την συγκεκριμένη εκπαίδευση στο δίκτυο με την χρήση του Faster RCNN Inception V2, στόχος ήταν η μείωση της απώλειας κάτω του 0.05 σταθερά. Με αυτό το δεδομένο η εκπαίδευση διήρκησε 2 ώρες και 5 λεπτά.



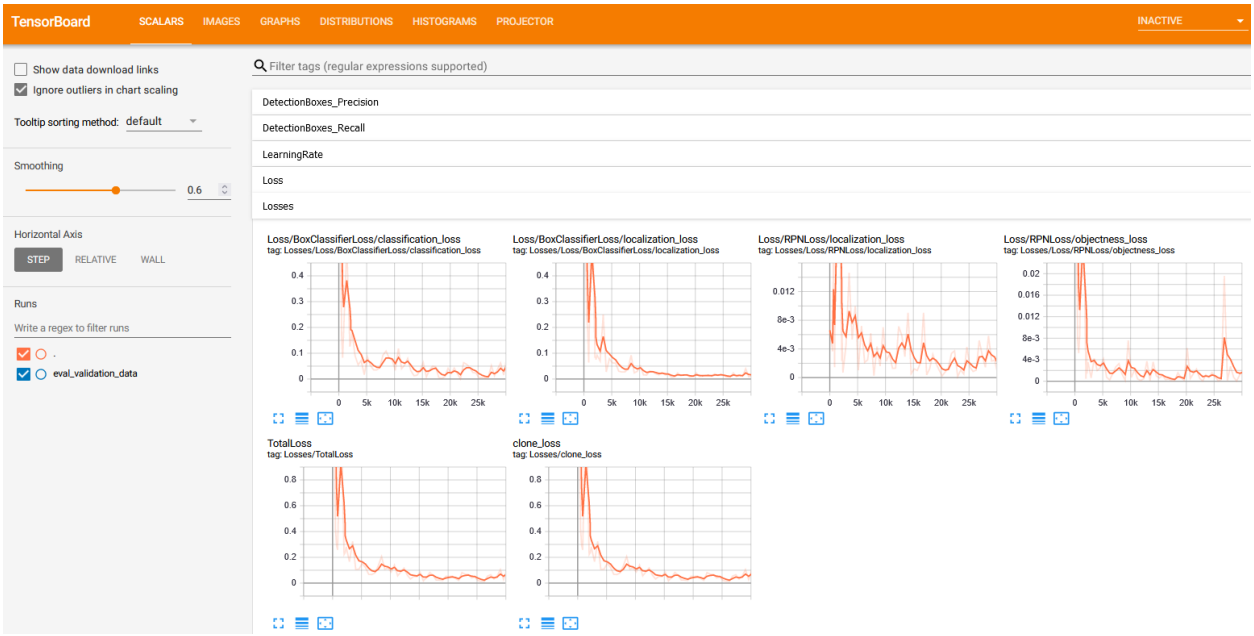
Εικόνα 32: Διαδικασία εκπαίδευσης δικτύου

Μέσω του εικονικού περιβάλλοντος του TensorFlow, υπάρχει δυνατότητα απεικόνισης της προόδου της εκπαίδευσης. Ανοίγοντας ακόμα ένα παράθυρο του Anaconda Prompt, ενεργοποιείται το εικονικό περιβάλλον του tensorflow1.

Στον κατάλογο C: /tensorflow1 /models /research /object_detection, εκτελείται η εντολή:

- tensorboard -- logdir=training

Με αυτόν τον τρόπο όπως παρατηρείται στην παρακάτω εικόνα απεικονίζονται οι απώλειες κατά την πορεία της εκπαίδευσης.



Εικόνα 33: Απεικόνιση των απωλειών στο TensorBoard

Η απώλεια (loss) όπως παρουσιάζεται με την χρήση γραφημάτων στο TensorBoard, αποτελείται από το άθροισμα της απώλειας εντοπισμού (localization loss-για την πρόβλεψη αντιστάθμισης του πλαισίου οριοθέτησης) και της απώλειας ταξινόμησης (classification loss-για την πρόβλεψη ταξινόμησης σε κλάση βάσει χαρακτηριστικών). Και στις δύο περιπτώσεις υπολογίζεται ως άθροισμα τετραγωνικών σφαλμάτων. Το RPN loss είναι και αυτός ένας συνδυασμός του regression loss και του classification loss, κατά την υλοποίηση και εκπαίδευση ενός Faster RCNN νευρωνικού δικτύου. Η έννοια global step που αναγράφεται κατά την εκπαίδευση του δικτύου αναφέρεται στην επανάληψη την οποία επεξεργάζεται το σύστημα.

Κατά την διάρκεια της εκπαίδευσης, επαναληπτικά και περιοδικά αποθηκεύονται τα σημεία

ελέγχου (checkpoints) τα οποία θα χρησιμοποιηθούν για την δημιουργία του παγωμένου γραφήματος συμπερασμάτων (frozen inference graph). Τα frozen inference graphs είναι γραφήματα που περιέχουν πληροφορίες σχετικά με το μοντέλο, την εκπαίδευσή του και τα χαρακτηριστικά του.

Για την αξιολόγηση του μοντέλου, εκτελείται το αρχείο eval.py που βρίσκεται στο φάκελο /object_detection. Η συγκεκριμένη συνάρτηση υπολογίζει τις μετρήσεις αξιολόγησης (evaluation metrics), δηλαδή τις προβλέψεις πάνω σε διαθέσιμες εικόνες που βρίσκονται στον κατάλογο C:/tensorflow1/models\research/object_detection/images/test. Κατά την διάρκεια αυτής της διαδικασίας οι μετρήσεις αξιολόγησης χωρίζονται σε δύο μέρη:

1. αξιολόγηση ύπαρξης αντικειμένου στην εκάστοτε εικόνα (classification-ταξινόμηση)
2. αξιολόγηση και καθορισμός θέσης του αντικειμένου στην εικόνα (regression-παλινδρόμηση, localization-εντοπισμός)

Εκτελείται η εντολή:

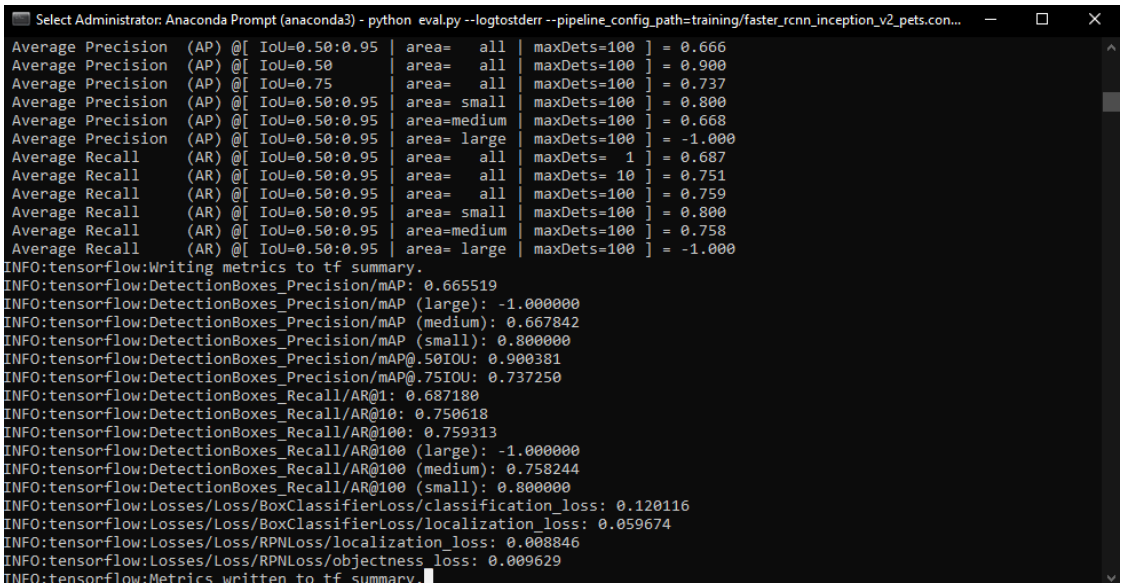
- python eval.py --logtostder --pipeline_config_path=training/faster_rcnn_inception_v2_pets.config –checkpoint_dir=training/ --eval_dir=eval/

```
(tensorflow1) C:\tensorflow1\models\research\object_detection>python eval.py --logtostder --pipeline_config_path=trainin
g/faster_rcnn_inception_v2_pets.config -checkpoint_dir=training/ --eval_dir=eval/
```

Εικόνα 34: Εκτέλεση εντολής αξιολόγησης μοντέλου

```
Administrator: Anaconda Prompt (anaconda3)
INFO:tensorflow:Running eval ops batch 100/108
INFO:tensorflow:Running eval batches done.
INFO:tensorflow:# success: 108
INFO:tensorflow:# skipped: 0
creating index...
index created!
INFO:tensorflow:Loading and preparing annotation results...
INFO:tensorflow:DONE (t=0.01s)
creating index...
index created!
Running per image evaluation...
Evaluate annotation type *bbox*
DONE (t=0.48s).
Accumulating evaluation results...
DONE (t=0.11s).
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.666
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.900
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.737
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.800
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.668
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = -1.000
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.687
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.751
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.759
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.800
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.758
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = -1.000
INFO:tensorflow:Writing metrics to tf summary.
INFO:tensorflow:DetectionBoxes_Precision/mAP: 0.665519
INFO:tensorflow:DetectionBoxes_Precision/mAP (large): -1.000000
```

Εικόνα 35: Precision - Recall no1



Εικόνα 36: Precision - Recall no2

Ο όρος **maxDets** που αναγράφεται στις εικόνες 35, 36 είναι το όριο μέγιστης ανίχνευσης σε κάθε εικόνα βάσει των ακόλουθων τιμών 1, 10, 100. Όλες οι αξιολογήσεις γίνονται επιτρέποντας έως 100 ανιχνεύσεις ανά εικόνα (για κάθε κατηγορία). Στην προκειμένη περίπτωση, στο μοντέλο που αναπτύχθηκε στην έρευνα και με την αντίστοιχη τιτλοφόρηση, το μέγιστο όριο ανίχνευσης δεν μπορεί να ξεπερνάει το 2 (καθώς έχουμε δύο κλάσεις LGG-HGG), και αυτός είναι ο λόγος οι τιμές των maxDets=10 και maxDets=100 είναι παρόμοιες.

Με τον όρο **IoU (Intersection over union-αλληλοεπικάλυψη)** αναφερόμαστε στην επικάλυψη που μπορεί να υπάρχει μεταξύ ενός ανιχνεύσιμου πλαισίου και του ground truth. Ο όρος **Ground Truth** δηλώνει τις μετρήσεις οι οποίες είναι σίγουρα έγκυρες σε σχέση με τις μετρήσεις ενός συστήματος που είναι σε δοκιμή. Έτσι με την τεχνική IoU στόχος είναι να μετρηθεί το πόσο καλή είναι η πρόβλεψη με τον εκάστοτε ανιχνευτή αντικειμένου. Όσο αυξάνεται το κατώτατο όριο-threshold, θα υπάρξει μια ποσοστιαία αύξηση των αληθινών θετικών (True Positives), δηλαδή θα συμπεριληφθούν στην ανίχνευση περισσότερα πλαίσια οριοθέτησης, συμπεριλαμβανομένων πολλών παραπάνω ψευδών θετικών (False Positives). Ταυτόχρονα με αυτόν τον τρόπο μειώνονται τα ψευδή αρνητικά (False Negatives), και έτσι θα μειωθεί η ακρίβεια και θα αυξηθεί η ανάκληση.

Όπως φαίνεται στις μετρήσεις, γίνεται υπολογισμός σε μικρές (small) και μεσαίες (medium) περιοχές για την ακρίβεια και την ανάκληση. Όμως για τις μεγάλες (large) περιοχές διατηρείται ακόμα η default τιμή -1.000. Αυτό συμβαίνει διότι οι κατηγοριοποιήσεις μικρής, μεσαίας και μεγάλης κλίμακας γίνονται βάσει των pixels-εικονοστοιχείων που ορίζει το κάθε μέγεθος. Το σύνολο δεδομένων που

χρησιμοποιήθηκε για την αξιολόγηση δεν ικανοποιεί το αντίστοιχο μέγεθος και για αυτό τον λόγο αναγράφεται η default τιμή στο συγκεκριμένο πεδίο.

Επεξήγηση Precision και Recall

Με τον όρο Precision (ακρίβεια) αποτυπώνεται το ποσοστό των προβλέψεων που είναι σωστές κατά την διάρκεια ανίχνευσης κάποιου αντικειμένου, ενώ με τον όρο Recall (ανάκληση - ευαισθησία) αποτυπώνεται ο αριθμός επιστρεφόμενων αποτελεσμάτων κατά την διάρκεια μιας ανίχνευσης.

Μαθηματικοί τύποι Precision - Recall

$$\text{Precision} = \frac{TP}{TP + FP} \qquad \text{Recall} = \frac{TP}{TP + FN}$$

Όπου TP=True Positive, FP=False Positive, FN=False Negative

Για την απεικόνιση του συνδυασμού της ακρίβειας και την ανάκλησης χρησιμοποιείται η έννοια του F1 Score, η οποία είναι ο αρμονικός μέσος μεταξύ της ακρίβειας και της ανάκλησης, καθιστώντας τον έναν καλό συγκεντρωτικό δείκτη των παραπάνω τιμών.

Μαθηματικός τύπος F1 Score

$$F1 = \frac{2 \times precision \times recall}{precision + recall}$$

Η επιλογή του F1 Score είναι καταλληλότερη για την απεικόνιση αποτελεσμάτων σε προβλήματα που βασίζονται σε ιατρικά δεδομένα και την εξαγωγή κρίσιμων αποτελεσμάτων. Ειδικά σε περιπτώσεις που το πρόβλημα είναι δυαδικό και το σύνολο δεδομένων δεν είναι ισορροπημένο, ο ανιχνευτής τείνει να στοχεύει σε μια συγκεκριμένη κατηγορία, γεγονός που μπορεί να οδηγήσει σε μοιραία σφάλματα. Με την απεικόνιση του F1 Score δίνεται έμφαση στα False Negatives και στα False Positives (FN: αριθμός προβλέψεων που έγιναν λανθασμένα, FP: αριθμός στόχων που θα έπρεπε να έχουν προβλεφθεί αλλά χάθηκαν) κατά τον εντοπισμό, σε αντίθεση με την μέτρηση Accuracy που δίνει έμφαση στα True Positives και True Negatives (TP: σωστές προβλέψεις, TN: σωστή πρόβλεψη αρνητικών ανιχνεύσεων ύπαρξης). Στον παρακάτω πίνακα γίνεται η απεικόνιση των μετρήσεων και ο υπολογισμός του F1 Score ως προς το σύνολο δεδομένων δοκιμής.

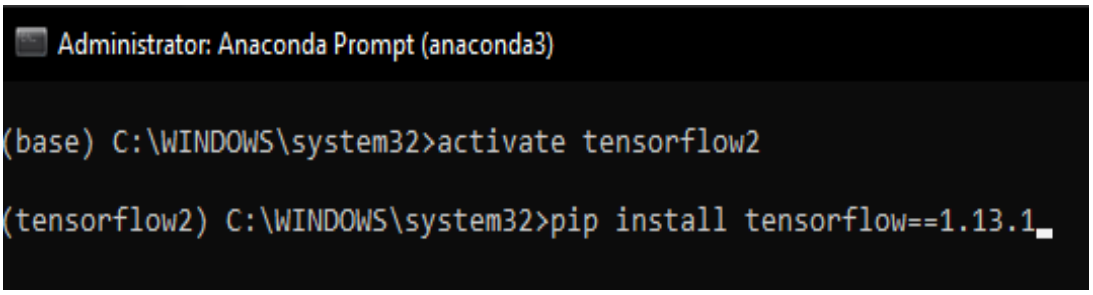
Πίνακας 1: Υπολογισμός F1 Score στο σύνολο δεδομένων δοκιμών

Classes	HGG - LGG
Images used for evaluation	108
Precision	0.67
Recall	0.69
F1 Score	69%

6.6 Σύγκριση χαρακτηριστικών μεταξύ εκπαιδεύσεων σε GPU και σε CPU

Για την παρουσίαση των διαφορών μεταξύ της εκπαίδευσης του μοντέλου με την κάρτα γραφικών του συστήματος (NVIDIA GeForce GTX 1050Ti 4GB) και της εκπαίδευσης του μοντέλου αποκλειστικά με τον επεξεργαστή (Intel(R) Core(TM) i5-8400 CPU @ 2.80GHz, 6 Core(s), 6 Logical Processor(s)), υλοποιήθηκε άλλη μια εκπαίδευση χωρίς την χρήση της παράλληλης υπολογιστικής δύναμης της κάρτας γραφικών που χρησιμοποιήθηκε στην πρώτη εκπαίδευση μέσω των δυνατοτήτων που προσφέρει η CUDA-cuDNN που αναλύεται στο κεφάλαιο 5.4. Στόχος αυτής της εκπαίδευσης είναι η παρουσίαση των διαφορών που παρατηρήθηκαν στους χρόνους εκπαίδευσης, στα global steps και στην ισχύ κατανάλωσης μεταξύ των δύο και η απόδειξη της αιτίας για την οποία τέτοιου είδους ανιχνευτές χρειάζονται αρκετή υπολογιστική ισχύ.

Για την δεύτερη εκπαίδευση τα βήματα για την εγκατάσταση των απαραίτητων βιβλιοθηκών και πακέτων είναι τα ίδια με της πρώτης (Κεφάλαιο 6.2, 6.4, 6.5) με μόνες διαφορές πως θα δημιουργηθεί ένας νέος κατάλογος με όνομα tensorflow2 σε κάποιο αποθηκευτικό μέσο που θα αποθηκευτεί η δομή καταλόγου του TensorFlow Object Detection API για την διαχείριση της δεύτερης εκπαίδευσης. Θα δημιουργηθεί ένα νέο εικονικό περιβάλλον tensorflow2 μέσω του Anaconda και αντί για την εγκατάσταση του tensorflow-gpu, θα εγκατασταθεί το πακέτο tensorflow όπως παρουσιάζεται στην εικόνα παρακάτω.



Εικόνα 37: Εγκατάσταση Tensorflow στο εικονικό περιβάλλον tensorflow2

Στον πίνακα 2 αναγράφεται ο αριθμός των global steps και οι απώλειες του ανιχνευτή που εκπαιδεύτηκε με GPU και CPU αντίστοιχα, λαμβάνοντας υπόψη σαν χρόνο εκπαίδευσης το χρόνο που χρειάστηκε η GPU. Αυτό γίνεται με σκοπό την σύγκριση της απόδοσης των δύο περιπτώσεων.

Πίνακας 2: Διαφορές απώλειας και global steps εκπαίδευσης σε GPU- CPU σε ίδιο χρόνο

Training Hardware	Training time (h/min)	Global Steps (επαναλήψεις εκπαίδευσης)	Total Loss (Starting value≈3.0)
NVIDIA GeForce GTX 1050Ti 4GB	2 h 5 min	31.9k (Full training)	< 0.05 (consistently below)
Intel(R) Core(TM) i5-8400 CPU @ 2.80GHz, 6 Core(s), 6 Logical Processor(s)	2 h 5 min	3.4k (Partial training)	> 1.6 (consistently above)

Όπως παρατηρείται στον πίνακα 2 η ταχύτητα επαναλήψεων των υπολογισμών με κάρτα γραφικών και με επεξεργαστή διαφέρει αρκετά. Η υπολογιστική ταχύτητα και ισχύς που παρέχεται από την κάρτα γραφικών είναι σχεδόν δεκαπλάσια για τον ίδιο χρόνο εκπαίδευσης και η απώλεια πολύ μικρότερη. Στην συνέχεια η εκπαίδευση για την CPU συνεχίστηκε μέχρις ότου να επιτευχθεί και σε αυτήν την περίπτωση η ζητούμενη τιμή απώλειας που είναι 0.05 σταθερά. Οι απαιτούμενοι χρόνοι εκπαίδευσης παρουσιάζονται στον πίνακα 3, καθώς και οι τιμές κατανάλωσης ισχύος (Watt) για την κάθε περίπτωση.

Πίνακας 3: Διαφορές σε χρόνους εκπαίδευσης και ισχύς κατανάλωσης GPU-CPU

Training Hardware	Full Training time (h/min)	Power Consumption (Watt)
NVIDIA GeForce GTX 1050Ti 4GB	2 h 5 min	≈ 215 W
Intel(R) Core(TM) i5-8400 CPU @ 2.80GHz, 6 Core(s), 6 Logical Processor(s)	21 h 16 min	≈ 1000 W

Συμπερασματικά η χρήση μιας GPU είναι η βέλτιστη επιλογή κατά την εκπαίδευση ενός μοντέλου, αφού είναι ικανή να επεξεργάζεται πολλαπλούς υπολογισμούς ταυτόχρονα λόγω του μεγάλου αριθμού πυρήνων. Επιπλέον η ισχύς που καταναλώνει είναι περίπου τέσσερις φορές μικρότερη από την αντίστοιχη μιας CPU, όπως παρατηρείται στον πίνακα 3.

6.7 Εξαγωγή γραφήματος inference graph και δοκιμή ανιχνευτή

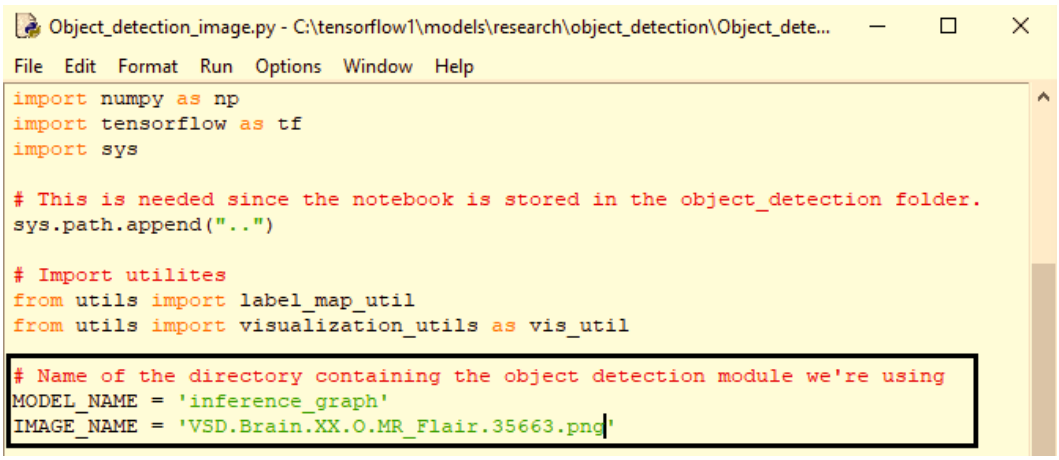
Αφού ολοκληρώθηκε η εκπαίδευση, πρέπει να δημιουργηθεί το τελικό γράφημα συμπεράσματος (frozen_inference_graph .pb) στον φάκελο /object_detection, με την εκτέλεση της παρακάτω εντολής εισάγοντας το πιο υψηλό σημείο ελέγχου (checkpoint) που βρίσκεται στον φάκελο /training:

- python export_inference_graph.py --input_type image_tensor --pipeline_config_path training/faster_rcnn_inception_v2_pets.config--trained_checkpoint_prefix training/model.ckpt-**31277** --output_directory inference_graph

Με αυτόν τον τρόπο δημιουργήθηκε το frozen_inference_graph.pb το οποίο έχει τα χαρακτηριστικά του ανιχνευτή ταξινόμησης αντικειμένων, στον φάκελο /object_detection/inference_graph.

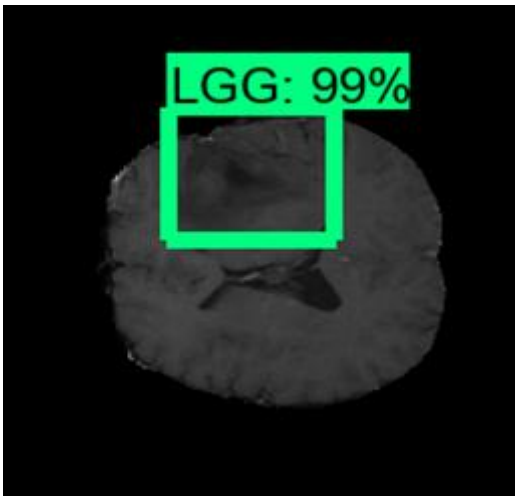
Σε αυτό το σημείο ο ανιχνευτής είναι έτοιμος να τεθεί σε λειτουργία. Με την χρήση του script Object_detection_image.py που δημιουργήθηκε θα εκτελεστεί ανίχνευση σε εικόνα. Στο συγκεκριμένο πρόγραμμα δηλώνεται το frozen inference graph του ανιχνευτή, η εικόνα που πρόκειται να υποστεί ανίχνευση, η πηγή εγγραφής σχολιασμών των κλάσεων προς ανίχνευση (labelmap.pbtxt) και ο συνολικός αριθμός κλάσεων προς ανίχνευση. Στην συνέχεια φορτώνεται το μοντέλο που εκπαιδεύτηκε στην μνήμη, γίνεται εισαγωγή εικόνας που δόθηκε σαν είσοδο και ξεκάνει η ανίχνευση.

Για την εκτέλεση του script, ενεργοποιείται το εικονικό περιβάλλον tensorflow1 και εκτελείται η εντολή idle. Μέσω της λειτουργικής μονάδας IDLE της Python που είναι ενσωματωμένο περιβάλλον ανάπτυξης για εκτέλεση python script, γίνεται επεξεργασία του Object_detection_image.py και αντικαθίσταται η τιμή της μεταβλητής IMAGE_NAME στην εικόνα που θα υλοποιηθεί ανίχνευση. Αφού εκτελεστεί, γίνεται μια προετοιμασία που διαρκεί 10 δευτερόλεπτα και στην συνέχεια εμφανίζεται το παράθυρο ανίχνευσης. Για τα παραδείγματα παρακάτω (εικόνες 39, 40, 41, 42) χρησιμοποιήθηκαν εικόνες για τις οποίες είναι γνωστό το είδος του όγκου που περιέχουν ώστε να είναι αντικειμενική η κρίση εγκυρότητάς της.

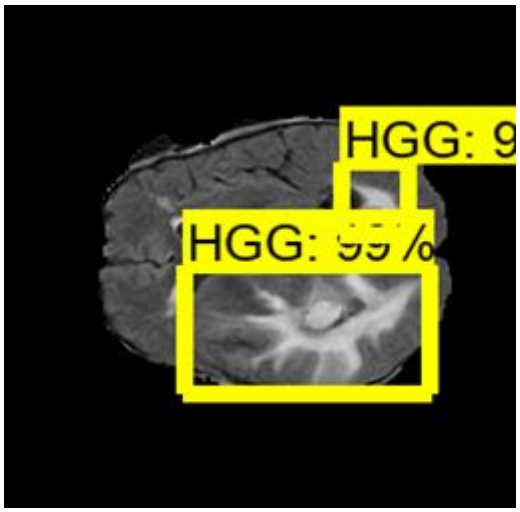


Εικόνα 38: Αλλαγή μεταβλητής IMAGE_NAME

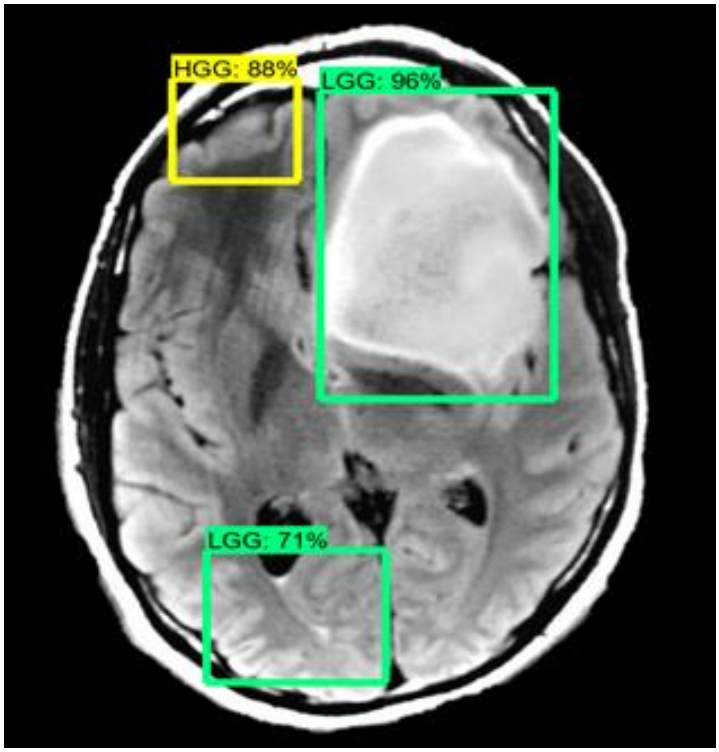
Στις παρακάτω εικόνες παρουσιάζεται η έξοδος του ανιχνευτή. Κατά την εκκίνηση της ανίχνευσης το μοντέλο ελέγχει την προς ανίχνευση εικόνα που δόθηκε ως τιμή στην μεταβλητή IMAGE_NAME, μαρκάρει το σημείο της εικόνας στο οποίο ανιχνεύεται ο όγκος με ένα ορθογώνιο και αναγράφει το είδος του όγκου με το ποσοστό σιγουριάς, το οποίο αντικατοπτρίζει κατά πόσο το πλαίσιο περιέχει ένα αντικείμενο ενδιαφέροντος και πόσο σίγουρος είναι ο ταξινομητής για το συγκεκριμένο αντικείμενο.



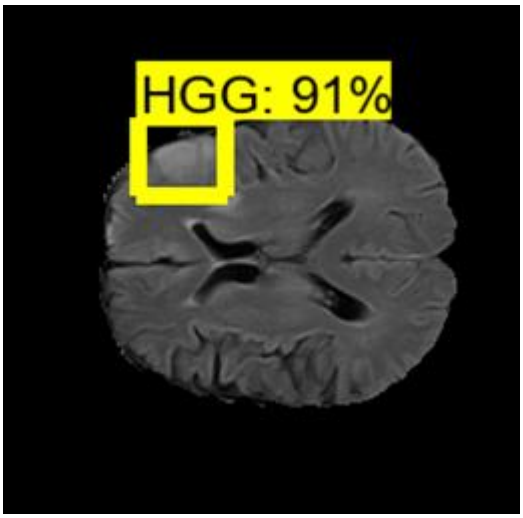
Εικόνα 39: Επιτυχής ανίχνευση LGG όγκου



Εικόνα 40: Επιτυχής ανίχνευση HGG όγκου



Εικόνα 41: Πολλαπλή ανίχνευση όγκων

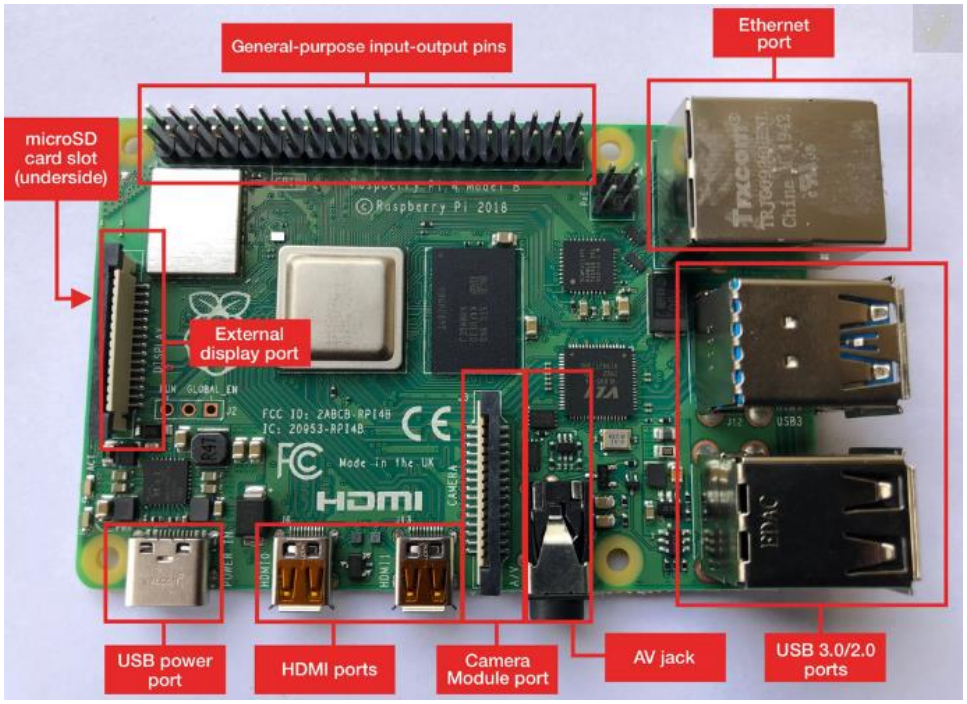


Εικόνα 42: Ανεπιτυχής ανίχνευση LGG είδους όγκου

7. Ενσωμάτωση εκπαιδευμένου μοντέλου σε Raspberry

7.1 Περιγραφή Raspberry

Το Raspberry Pi[10] είναι μια σειρά από μικρούς υπολογιστές μονής πλακέτας. Η εταιρεία δημιουργήθηκε στο Ηνωμένο Βασίλειο από το Ίδρυμα Raspberry Pi σε συνεργασία με την Broadcom. Έχουν κυκλοφορήσει τρείς σειρές Raspberry Pi και πολλαπλές γενιές για το καθένα. Τα Raspberry Pi SBC διαθέτουν σύστημα Broadcom σε τσιπ (SoC – System on a chip) με ενσωματωμένη κεντρική μονάδα επεξεργασίας (CPU) συμβατή με ARM και on-chip μονάδα επεξεργασίας γραφικών (GPU). Τα Raspberry Pi Pico διαθέτουν σύστημα on-chip RP2040 με ενσωματωμένο ARM με συμβατή κεντρική μονάδα επεξεργασίας (CPU). Βασικός λόγος δημιουργίας του Raspberry Pi ήταν η προώθηση διδασκαλίας της βασικής επιστήμης των υπολογιστών σε εκπαιδευτικούς χώρους και σε αναπτυσσόμενες χώρες με χαμηλό κόστος. Στην πορεία όμως χρησιμοποιήθηκε αρκετά σε εφαρμογές και χρήσεις ρομποτικής καθώς και σε άλλους τομείς όπως π.χ. την παρακολούθηση του καιρού, λόγω χαμηλού κόστους αλλά και του σχεδιασμού και της συνδεσιμότητάς του.



Εικόνα 43: Αρχιτεκτονική Raspberry Pi

Χαρακτηριστικά ενός Raspberry (Raspberry Pi 4)

Processor: Ο επεξεργαστής είναι βρίσκεται στο εσωτερικό μεταλλικού περιβλήματος και βασίζεται στον τετραπλό πυρήνα Broadcom BCM2711B0 64-bit ο οποίος αποτελείται από ένα Cortex A-72 core λειτουργώντας στα 1.5GHz.

RAM: Υπάρχουν τρεις εκδόσεις του Raspberry Pi που διαφέρουν στο μέγεθος της μνήμης RAM (1GB, 2GB, 4GB, 8GB).

USB Ports: Το Raspberry Pi περιλαμβάνει δύο θύρες USB 3.0 και δύο USB 2.0 καθώς επίσης και μια USB-C. Ο ρυθμός μεταφοράς δεδομένων στην USB 3.0 είναι 4.800 Mbps (megabites το δευτερόλεπτο) ενώ το USB 2.0 μπορεί να μεταφέρει μέχρι και 480Mbps, δηλαδή 10 φορές πιο αργά από ότι το USB 3.0.

Ethernet: Η θύρα του Ethernet επιτρέπει στο Raspberry Pi να συνδεθεί κατευθείαν σε μια θύρα Ethernet ενός router. Η θύρα επιτρέπει Gigabit connections (125Mbps).

HDMI: Δύο θύρες HDMI επιτρέπουν την διασύνδεση του Raspberry Pi με monitor με ανάλυση έως και 4K.

GPIO: Μια ακιδοσειρά σαράντα (40) θέσεων επιτρέπει την διασύνδεση του Raspberry Pi με άλλες πλακέτες, που ονομάζεται GPIO (General Purpose Input Output) διασύνδεση.

Audio-Video port: Με χρήση καλωδίου jack 3.5mm υπάρχει δυνατότητα για stereo ακουστική έξοδο. Ακουστικά μπορούν να συνδεθούν σε αυτή τη έξοδο. Ακουστικοί ενισχυτές δέχονται ακουστικό σήμα από αυτή την θύρα. Επίσης αυτή η θύρα υποστηρίζει compatible video, επιτρέποντας τηλεοπτικούς δέκτες, προτζέκτορες και άλλα παρόμοια συμβατά display να συνδεθούν.

CSI Port: Αυτή είναι η θύρα στην οποία μια camera (Camera Serial Interface) μπορεί να συνδεθεί με το Raspberry pi.

DSI: Αυτή είναι η θύρα οπτικής εξόδου (Display Serial Interface) η οποία επιτρέπει σε μια συμβατή μονάδα απεικόνισης να συνδεθεί με το Raspberry Pi.

Micro SD Card: Χρησιμοποιείται ως αποθηκευτικός χώρος τοποθετώντας την στην υποδοχή που

βρίσκεται στο πίσω μέρος της πλακέτας.

Για την λειτουργία του απαιτείται και ο παρακάτω εξωτερικός εξοπλισμός:

Power Supply: Για την λειτουργία του Raspberry Pi απαιτείται τροφοδοτικό τύπου adapter USB-C με παροχή τάσης 5V και έντασης 3A.

Operating System: Το λειτουργικό σύστημα βρίσκεται στην αγορά εγκατεστημένο σε μια SD Card γνωστό ως NOOBS (New Out Of the Box Software) το οποίο απαιτεί την ελάχιστη διαμόρφωση. Εναλλακτικά, δίνεται η δυνατότητα στον χρήστη να εγκαταστήσει σε ένα κενό SD Card όποιο λογισμικό επιθυμεί μέσω του Raspberry Pi Imager.

USB keyboard and mouse: Μπορεί να χρησιμοποιηθεί είτε ασύρματο είτε ενσύρματο πληκτρολόγιο και ποντίκι.

Συσκευή απεικόνισης: Απαιτείται το κατάλληλο καλώδιο από micro HDMI σε HDMI για την απεικόνιση σε μια συμβατή HDMI μονάδα. Εναλλακτικά μια VGA οθόνη με ένα καλώδιο micro HDMI σε VGA μετατροπέα ή DVI-D.

Για την ενσωμάτωση του μοντέλου που εκπαιδεύτηκε στην συγκεκριμένη έρευνα θα χρησιμοποιηθεί ένα Raspberry Pi 4 Model B 4GB.

7.2 Εγκατάσταση κατάλληλων βιβλιοθηκών και πακέτων

Για την ενσωμάτωση του εκπαιδευμένου μοντέλου στο Raspberry Pi είναι απαραίτητο να εγκατασταθούν κάποιες συγκεκριμένες βιβλιοθήκες και πακέτα όπως αντίστοιχα έγινε και για την υλοποίηση και εκπαίδευση του μοντέλου στο Desktop PC.

Αρχικά απαραίτητη εργασία είναι η ενημέρωση του Raspberry που χρησιμοποιείται για την ενσωμάτωση του μοντέλου. Μέσω του Command Prompt του Raspberry εκτελούνται οι παρακάτω εντολές:

- sudo apt-get update
- sudo apt-get dist-upgrade

Όταν ολοκληρωθεί η διαδικασία ενημέρωσης του Raspberry πρέπει να γίνει εγκατάσταση του TensorFlow v1.13.1 αφού αυτή η έκδοση χρησιμοποιήθηκε και για την εκπαίδευση του μοντέλου. Για την εγκατάσταση του TensorFlow εκτελούνται οι εντολές:

- sudo apt-get install libatlas-base-dev (πακέτο που χρειάζεται το TensorFlow)
- pip3 install tensorflow==1.13.1 (εγκατάσταση TensorFlow v1.13.1)
- sudo pip3 install pillow lxml jupyter matplotlib cython (εξαρτήσεις που χρησιμοποιούνται από το TensorFlow)
- sudo apt-get install python-tk (εξαρτήσεις που χρησιμοποιούνται από το TensorFlow)

Στην συνέχεια θα πρέπει να εγκατασταθεί το OpenCV που θα βοηθάει στην απεικόνιση των εικόνων κατά την ανίχνευση. Για να γίνει αυτό προϋπόθεση είναι να γίνει εγκατάσταση κάποιων επιπλέον εξαρτήσεων. Οι εντολές που εκτελούνται είναι:

- sudo apt-get install libjpeg-dev libtiff5-dev libjasper-dev libpng12-dev
- sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv41-dev
- sudo apt-get install libxvidcore-dev libx264-dev
- sudo apt-get install qt4-dev-tools libatlas-base-dev
- sudo pip3 install python3-opencv

Αφού εγκατασταθεί το OpenCV τελευταίο πακέτο που θα χρειάζεται είναι το Protobuf που χρησιμοποιείται από το TensorFlow για την σειριοποίηση δεδομένων. Για την εγκατάσταση του πακέτου εκτελείται η εντολή:

- sudo apt-get install protobuf-compiler

7.3 Δημιουργία δομής καταλόγου TensorFlow και μεταβλητής PYTHONPATH

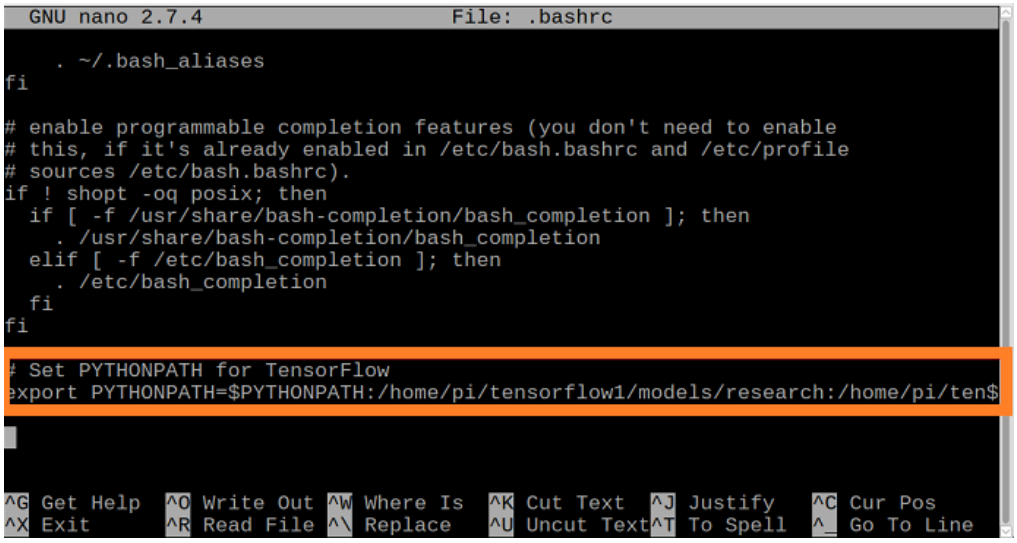
Δημιουργήθηκε ένας κατάλογος με όνομα tensorflow1 στον οποίο θα τοποθετηθεί η αποθήκη του TensorFlow από το GitHub από τον σύνδεσμο https://github.com/tensorflow/models/tree/r1.13.0. Έπειτα έγινε αποσυμπίεση του αρχείου.

Για την δημιουργία του καταλόγου και την αποσυμπίεση του αρχείου που λήφθηκε από τον σύνδεσμο, εκτελούνται οι παρακάτω εντολές:

- mkdir tensorflow1
- cd tensorflow1
- unzip models-r1.13.0.zip

Αφού αποσυμπιεστούν τα αρχεία, έχει σειρά η διαμόρφωση του PYTHONPATH με σκοπό να δείχνει στους φακέλους της αποθήκης του TensorFlow. Για να μείνουν σταθερές οι ρυθμίσεις στην διαμόρφωση του PYTHONPATH εκτελούνται οι εντολές:

- sudo nano ~/.bashrc
- export PYTHONPATH=$PYTHONPATH:/home/pi/tensorflow1/models/research:/home/pi/tensorflow1/models/research/slim



Εικόνα 44: Εντολή ανάθεσης PYTHONPATH

Σε αυτό το σημείο πρέπει γίνει μεταγλώττιση των Protocol Buffers που χρησιμοποιούνται από το TensorFlow για την ρύθμιση παραμέτρων του μοντέλου. Από τον κατάλογο /tensorflow1/models/research εκτελείται η παρακάτω εντολή:

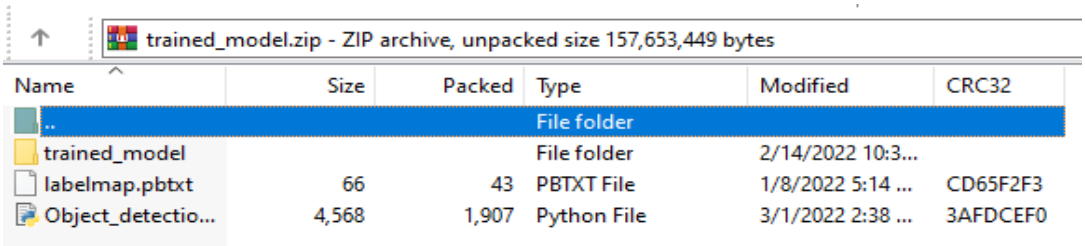
- protoc object_detection/protos/*.proto –python_out=.

Η ενέργεια αυτή μετατρέπει όλα τα proto αρχεία σε εκτελέσιμα python αρχεία στον κατάλογο protos.

7.4 Ανάπτυξη εκπαιδευμένου μοντέλου στο Raspberry Pi

Εφόσον έχει ολοκληρωθεί η ρύθμιση και η εγκατάσταση όλων των απαραίτητων πακέτων και βιβλιοθηκών που χρειάζονται για την λειτουργία της ανίχνευσης στο Raspberry Pi, έχει σειρά η ενσωμάτωση και δοκιμή του εκπαιδευμένου μοντέλου για την ανίχνευση όγκου εγκεφάλου στο Raspberry.

Για την προχωρήσει αυτό το βήμα πρέπει να προστεθεί το frozen_inference_graph στον κατάλογο object_detection, το labelmap.pbtxt με τις κλάσεις ανίχνευσης και το script Object_detection_image.py που χρησιμοποιήθηκε και στην ανίχνευση στο Desktop PC. Γίνεται συμπίεση ενός αρχείου με όνομα trained_model.zip το οποίο περιέχει όλα τα αρχεία του καταλόγου inference_graph, το αρχείο labelmap.pbtxt και το script ανίχνευσης Object_detection_image.py που δημιουργήθηκε όπως παρουσιάζεται στην εικόνα παρακάτω.



Εικόνα 45: Αρχεία εκπαιδευμένου μοντέλου, labelmap, Object_detection_image

Για την δυνατότητα αξιολόγησης του μοντέλου στο Raspberry θα αντιγραφούν και τα αρχεία test.records.

Γίνεται λήψη του παραπάνω αρχείου στο Raspberry και τοποθετείται στον κατάλογο /object_detection. Έπειτα εκτελούνται οι παρακάτω εντολές για την αποσυμπίεση και μετακίνηση των αρχείων στους κατάλληλους καταλόγους.

- cd tensorflow1/models/research/object_detection
- unzip trained.zip (αποσυμπίεση αρχείου στον φάκελο object_detection)
- mv labelmap.pbtxt data (μετακίνηση αρχείου labelmap.pbtxt στον κατάλογο data)

Εφόσον έχει ολοκληρωθεί η αποσυμπίεση και τα αρχεία βρίσκονται στους κατάλληλους καταλόγους, πρέπει να γίνουν αλλαγές σε κάποιες μεταβλητές στο αρχείο pipeline.config στον κατάλογο /trained_model και στο αρχείο Object_detection_image.py.

Στο αρχείο pipeline.config θα γίνουν οι παρακάτω αλλαγές:

- Αλλαγή στην ενότητα train_input_reader και στην ενότητα eval_input_reader τα input_path και label_map_path.

```
train_input_reader {
  label_map_path: "/home/pi/tensorflow1/models/research/object_detection/data/labelmap.pbtxt"
  tf_record_input_reader {
    input_path: "/home/pi/tensorflow1/models/research/object_detection/train.record"
  }
}
```

```
eval_input_reader {
  label_map_path: "/home/pi/tensorflow1/models/research/object_detection/data/labelmap.pbtxt"
  shuffle: false
  num_readers: 1
  tf_record_input_reader {
    input_path: "/home/pi/tensorflow1/models/research/object_detection/test.record"
  }
}
```

- Αλλαγή του checkpoint input της εκπαίδευσης.

```
fine_tune_checkpoint: "/home/pi/tensorflow1/models/research/object_detection/trained_model/model.ckpt"
```

Στο script Object_detection_image.py θα γίνουν οι παρακάτω αλλαγές όπως παρουσιάζονται στην εικόνα:

- Αλλαγή MODEL_NAME path ώστε να δείχνει στον φάκελο που βρίσκεται το frozen_inference _graph.
- Αλλαγή PATH_TO_LABELS path ώστε να δείχνει στον φάκελο που βρίσκεται το αρχείο labelmap.pbtxt.

```
# Import utilites
from utils import label_map_util
from utils import visualization_utils as vis_util

# Name of the directory containing the object detection module we're using
MODEL_NAME = 'trained_model'
IMAGE_NAME = '|.jpg'

# Grab path to current working directory
CWD_PATH = os.getcwd()

# Path to frozen detection graph .pb file, which contains the model that is used
# for object detection.
PATH_TO_CKPT = os.path.join(CWD_PATH,MODEL_NAME,'frozen_inference_graph.pb')

# Path to label map file
PATH_TO_LABELS = os.path.join(CWD_PATH,'data','labelmap.pbtxt')

# Path to image
PATH_TO_IMAGE = os.path.join(CWD_PATH,IMAGE_NAME)
```

7.5 Αξιολόγηση και δοκιμή ανιχνευτή στο Raspberry Pi

Σε αυτό το σημείο μπορεί να γίνει δοκιμή και αξιολόγηση του ανιχνευτή μετά την ενσωμάτωσή του στο Raspberry.

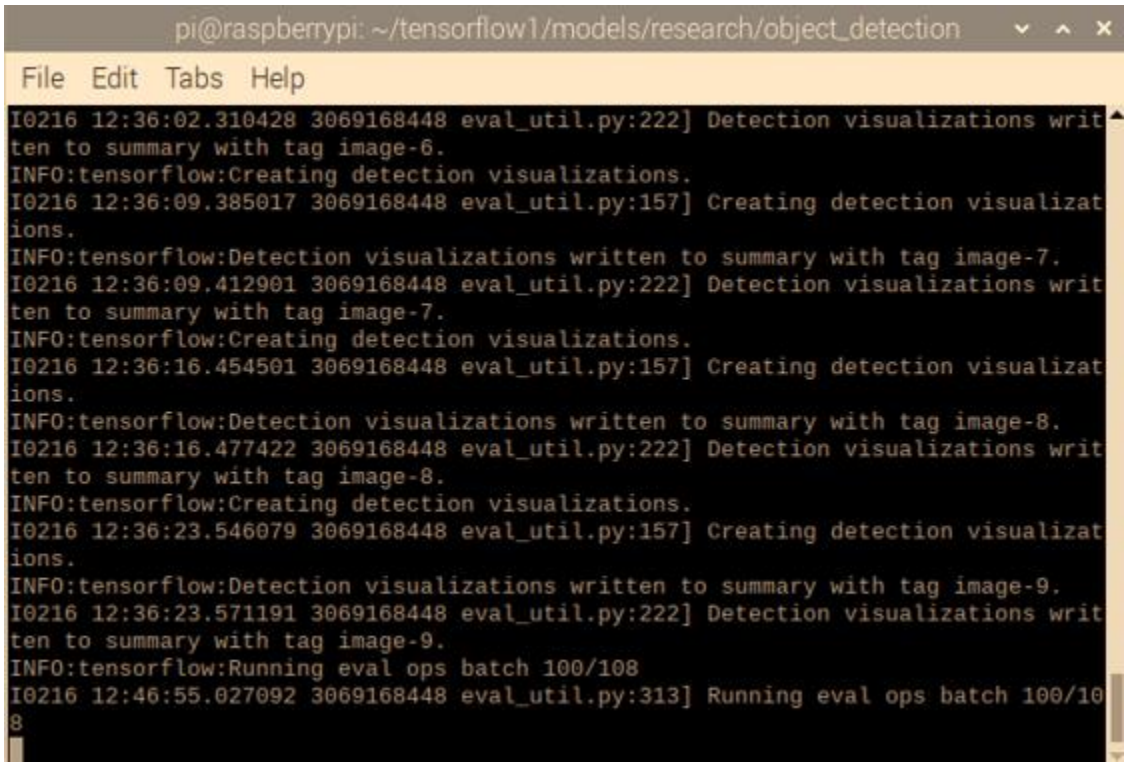
Αξιολόγηση ανιχνευτή σε Raspberry

Κατά την αξιολόγηση του μοντέλου φαίνεται πως για τις ίδιες δοκιμές τα αποτελέσματα είναι ακριβώς τα ίδια με αυτά της αξιολόγησης του μοντέλου στο Desktop PC.

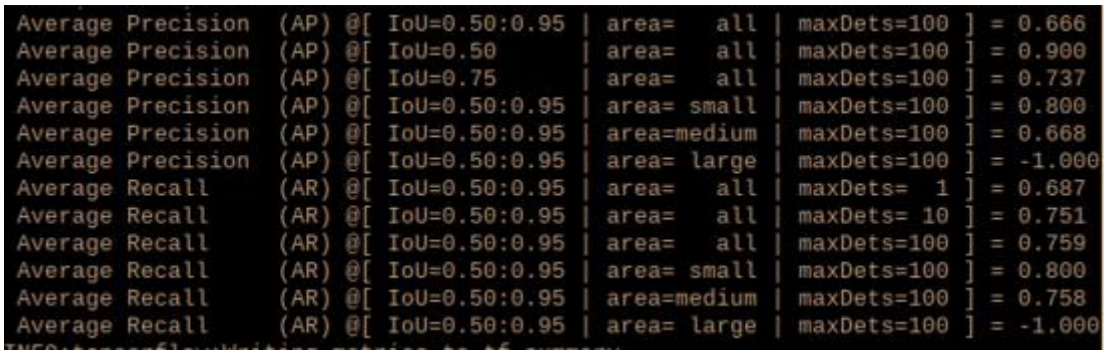
Για την υλοποίηση της αξιολόγησης εκτελείται η παρακάτω εντολή και παρουσιάζονται τα αποτελέσματα στις παρακάτω εικόνες.

```
pi@raspberrypi:~/tensorflow1/models/research/object_detection $ python3 eval.py
--logtostder --pipeline_config_path=trained_model/pipeline.config --checkpoint_d
ir=trained_model
```

Εικόνα 46: Εντολή αξιολόγησης μοντέλου στο σύνολο δεδομένων δοκιμών



Εικόνα 47: Διαδικασία αξιολόγησης



Εικόνα 48: Αποτελέσματα Precision – Recall

Όπως παρατηρείται οι μετρήσεις ακρίβειας και ανάκλησης κατά την αξιολόγηση του μοντέλου στο Raspberry Pi είναι ακριβώς ίδιες με τις μετρήσεις αξιολόγησης που έγιναν στο αρχικό σύστημα (Κεφάλαιο 6.5). Η μόνη διαφορά που παρατηρήθηκε είναι πως η αξιολόγηση διήρκησε περισσότερο χρόνο λόγω των χαρακτηριστικών του συστήματος του Raspberry.

Δοκιμή ανίχνευσης όγκου εγκεφάλου σε εικόνα

Για την δοκιμή του ανιχνευτή, με την χρήση κάποιου λογισμικού επεξεργασίας κειμένου ή κώδικα γίνεται επεξεργασία του script Object_detection_image.py που βρίσκεται στον κατάλογο /object_detection. Κατά την επεξεργασία αυτή γίνεται αλλαγή στην τιμή της μεταβλητής IMAGE_NAME ώστε να δείχνει στην εικόνα στην οποία θα υλοποιηθεί ανίχνευση όπως ακριβώς έγινε και στο κεφάλαιο 6.6. Η ανίχνευση γίνεται σε εικόνες που είναι γνωστό το είδος του όγκου ώστε να είναι αντικειμενική η κρίση εγκυρότητάς της.

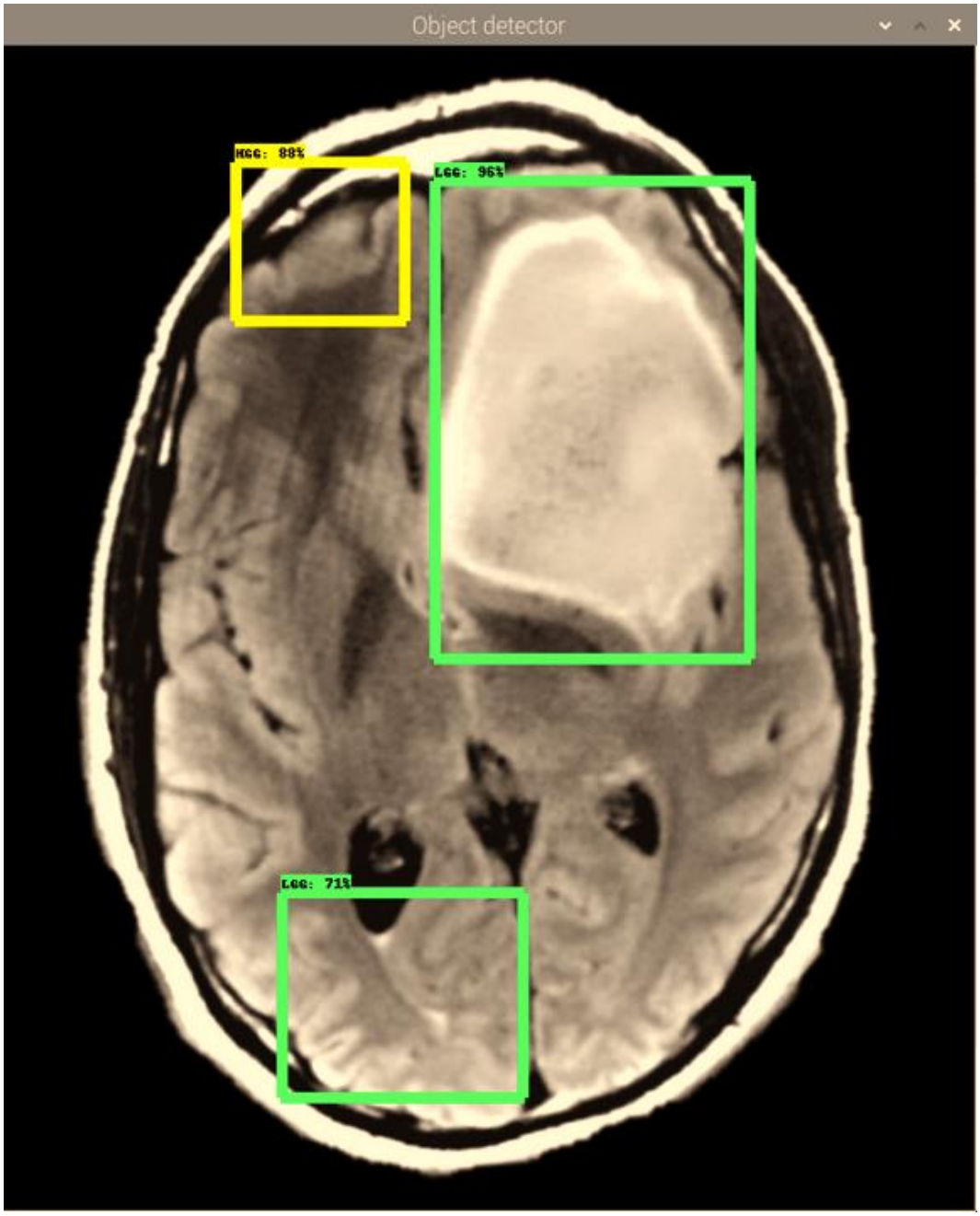
Αφού αλλάξει η τιμή του IMAGE_NAME για να δείχνει την εικόνα που υποστεί ανίχνευση εκτελείται η εντολή που παρουσιάζεται στην εικόνα παρακάτω:

- python3 Object_detection_image.py

Αφού γίνει εκτέλεση της εντολής, απαιτείται από το σύστημα ένα χρονικό διάστημα 10-15 δευτερολέπτων για την προετοιμασία και στην συνέχεια εμφανίζεται το παράθυρο ανίχνευσης. Σε κάθε ανίχνευση αναγράφεται η αντίστοιχη πρόβλεψη κλάσης και ένα confidence score. Το confidence score αντικατοπτρίζει κατά πόσο το κουτί περιέχει ένα αντικείμενο ενδιαφέροντος και πόσο σίγουρος είναι ο ταξινομητής για το συγκεκριμένο αντικείμενο.

Στην παρακάτω εικόνα παρουσιάζεται μια από τις δοκιμές που έγινε μέσω του Raspberry Pi σε εικόνα

που έχει ήδη υποστεί ανίχνευση στο κεφάλαιο 6.6 (εικόνα 41). Οι ανιχνεύσεις στις εικόνες 41 και 49 είναι ακριβώς ίδιες αφού λειτουργεί ο ίδιος ακριβώς ανιχνευτής και στις δύο περιπτώσεις.

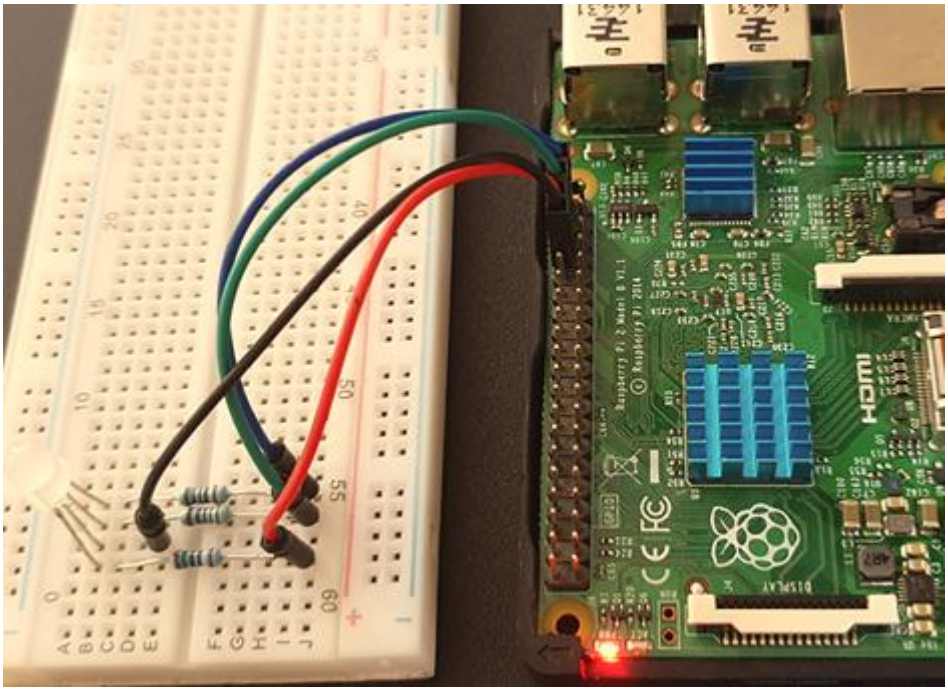


Εικόνα 49: Ανίχνευση πολλαπλών όγκων

Για την οπτική διεπαφή κατά την ανίχνευση ενός όγκου σε μια εικόνα, τοποθετήθηκε ένα RGB Led το οποίο συνδέθηκε μαζί με το Raspberry και έχει σκοπό τον αντίστοιχο φωτισμό την στιγμή της ανίχνευσης ανάλογα με το είδος του όγκου την κάθε φορά. Για την υλοποίηση αυτού χρησιμοποιήθηκαν τα παρακάτω εξαρτήματα:

- 1 x 5mm RGB Led
- 3 x Resistors (Red Resistor=160Ω, Green Resistor=100Ω, Blue resistor=100Ω)
- 1 x Breadboard
- 4 x male to female jumper wires

Αφού ολοκληρωθούν οι συνδέσεις του Breadboard με το Raspberry με την χρήση των παραπάνω εξαρτημάτων στις θύρες GPIO (όπως παρουσιάζεται παρακάτω στην εικόνα) και προγραμματιστούν στις αντίστοιχες εξόδους, κάθε φορά που ολοκληρώνεται ανίχνευση, το σύστημα είναι σε θέση να ειδοποιεί με την ενεργοποίηση του αντίστοιχου χρώματος του Led για το είδος της ανίχνευσης. Δηλαδή αν είναι LGG όγκος να ανάβει πράσινο, αν είναι HGG όγκος να ανάβει κόκκινο και αν δεν ανιχνευτεί κάποιος όγκος να ανάβει μπλε.



Εικόνα 50: Συνδεσμολογία Breadboard με Raspberry

10. https://www.raspberrypi.org/

8. Απεικόνιση αποτελεσμάτων-υπολογισμών ανιχνευτή σε Desktop PC και σε Raspberry Pi Model 4 με χρήση υπολογιστικού νέφους

Για την απεικόνιση και παρουσίαση των αποτελεσμάτων ακρίβειας, ανάκλησης και απωλειών του μοντέλου που εκπαιδεύτηκε θα χρησιμοποιηθεί η τεχνολογία υπολογιστικού νέφους ή αλλιώς Cloud.

Υπολογιστικό νέφος (Cloud-Cloud Computing)

Ο όρος Υπολογιστικό νέφος (Cloud - Cloud Computing) αναφέρεται σε υπηρεσίες client-server, αποθηκευτικών χώρων, βάσεων δεδομένων, δικτύωσης, λογισμικού και αναλυτικών στοιχείων και είναι από τις πλέον χρηστικές τεχνολογίες σε καθημερινή βάση σε πολλούς τομείς. Το Cloud Computing συνήθως προσφέρεται από εταιρείες επί πληρωμή αλλά και δωρεάν σε κάποιες περιπτώσεις και υλοποιεί την αποθήκευση των δεδομένων των χρηστών διαδικτυακά αντί τοπικά. Αυτή την δυνατότητα την προσφέρουν έχοντας κατάλληλες υποδομές, τεχνογνωσία και παρέχοντας την απαραίτητη ασφάλεια των δεδομένων των χρηστών. Με την χρήση της τεχνολογίας Cloud-Cloud Computing δίνεται η δυνατότητα πρόσβασης στα δεδομένα του εκάστοτε χρήστη από πολλαπλά σημεία και πολλαπλές συσκευές. Δηλαδή οι πλατφόρμες αυτές διαθέτουν τις κατάλληλες εφαρμογές ώστε να μπορούν να συγχρονίζονται και να επιτρέπουν την πλοήγηση στα δεδομένα του χρήστη χωρίς καμία δυσκολία. Πολλές μεγάλες εταιρίες Πληροφορικής όπως η Microsoft, Google, Apple, Dropbox κ.ά. προσφέρουν αυτήν την υπηρεσία διαθέτοντας στον χρήστη δωρεάν αποθηκευτικό χώρο για χρήση.

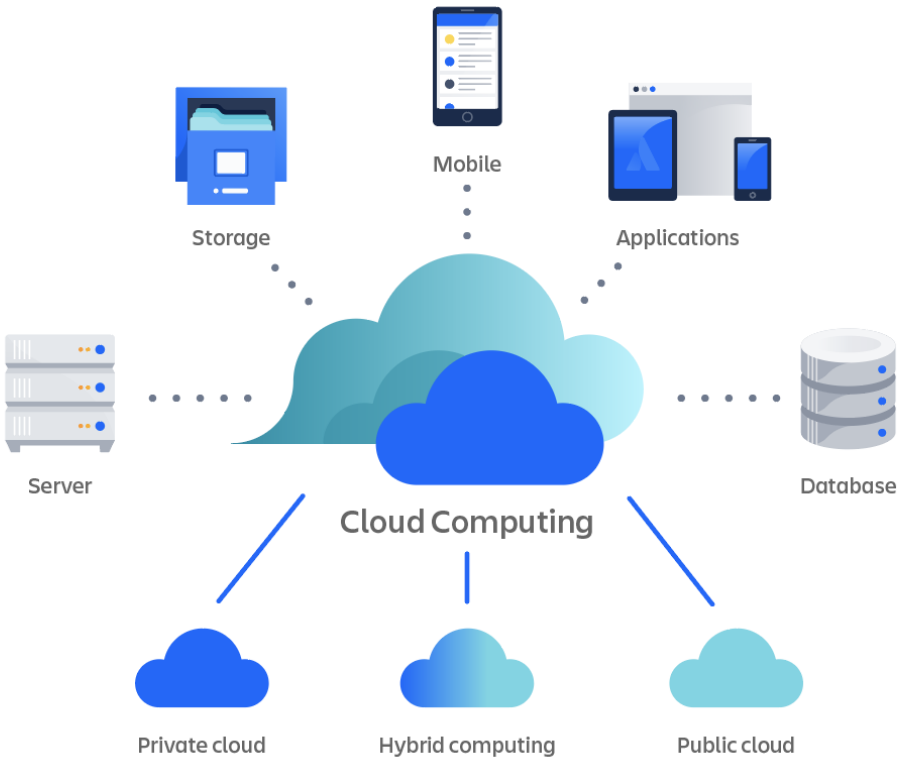
Δυνατότητες υπολογιστικού νέφους

- Δημιουργία εφαρμογών και υπηρεσιών
- Αποθήκευση, δημιουργία αντιγράφων ασφαλείας και ανάκτηση δεδομένων
- Φιλοξενία ιστοτόπων
- Μετάδοση ήχου και βίντεο
- Παραδόσεις λογισμικών κατά παραγγελία
- Ανάλυση δεδομένων για μοτίβα και δημιουργία γραφημάτων και προβλέψεων

Τύποι Cloud-Cloud Computing

- IaaS-Infracture as service (Υποδομή ως υπηρεσία): Από τις πιο βασικές υπηρεσίες cloud computing με δυνατότητα ενοικίασης υποδομών πληροφορικής, servers και virtual machines για αποθηκευτικό χώρο, δίκτυα και λειτουργικά συστήματα.

- PaaS-Platform as service (Πλατφόρμα ως υπηρεσία): Υπηρεσία για δημιουργία περιβάλλοντος συνήθως κατά παραγγελία για την ανάπτυξη, έλεγχο, παράδοση και διαχείριση εφαρμογών.
- SaaS-Software as service (Λογισμικό ως υπηρεσία): Υπηρεσία παράδοσης εφαρμογών λογισμικού μέσω διαδικτύου που συνήθως γίνεται επί πληρωμή.
- SraaS-Storage as service (Αποθηκευτικός χώρος ως υπηρεσία): Υπηρεσία που παρέχει δωρεάν ή επί πληρωμή αποθηκευτικό χώρο online.
- HaaS-Hardware as service (Εξοπλισμός ως υπηρεσία): Υπηρεσία που παρέχει στους συνδρομητές της online χρήση hardware που μπορεί να χρειάζεται μια επιχείρηση όπως web servers, μνήμες, CPU, GPU, αποθηκευτικά μέσα κ.ά..
- DaaS-Database as service (Βάση δεδομένων ως υπηρεσία): Παροχή υπηρεσιών online βάσεων δεδομένων που μπορούν επιχειρήσεις να χρησιμοποιούν μέσω άλλων διαδικτυακών εφαρμογών.



Εικόνα 51: Δομή Υπολογιστικού Νέφους (Cloud Computing)

8.1 Έκθεση των δεδομένων από τις μετρήσεις-εκτιμήσεις της ανίχνευσης μέσω Google Data Studio

Για την υλοποίηση των απεικονίσεων θα χρησιμοποιηθεί το δωρεάν εργαλείο Google Data Studio[11]. Πρόκειται για λογισμικό που βασίζεται στην τεχνολογία Cloud και είναι σχεδιασμένο για μικρές ή μεγάλες επιχειρήσεις ώστε να δημιουργούν, προσαρμόζουν, μοιράζονται και να απεικονίζουν σύνολα δεδομένων μέσω γραφημάτων (γραφήματα γραμμών-πίτας-μετοχών-φυσαλίδας-δακτυλίου) και πινάκων. Η συγκεκριμένη εφαρμογή δίνει την δυνατότητα στους χρήστες να κάνουν συλλογή δεδομένων μέσω διαφορετικών πηγών και να τα προετοιμάζουν για την απεικόνισή τους μέσω των τεχνικών που αναφέρθηκαν παραπάνω.

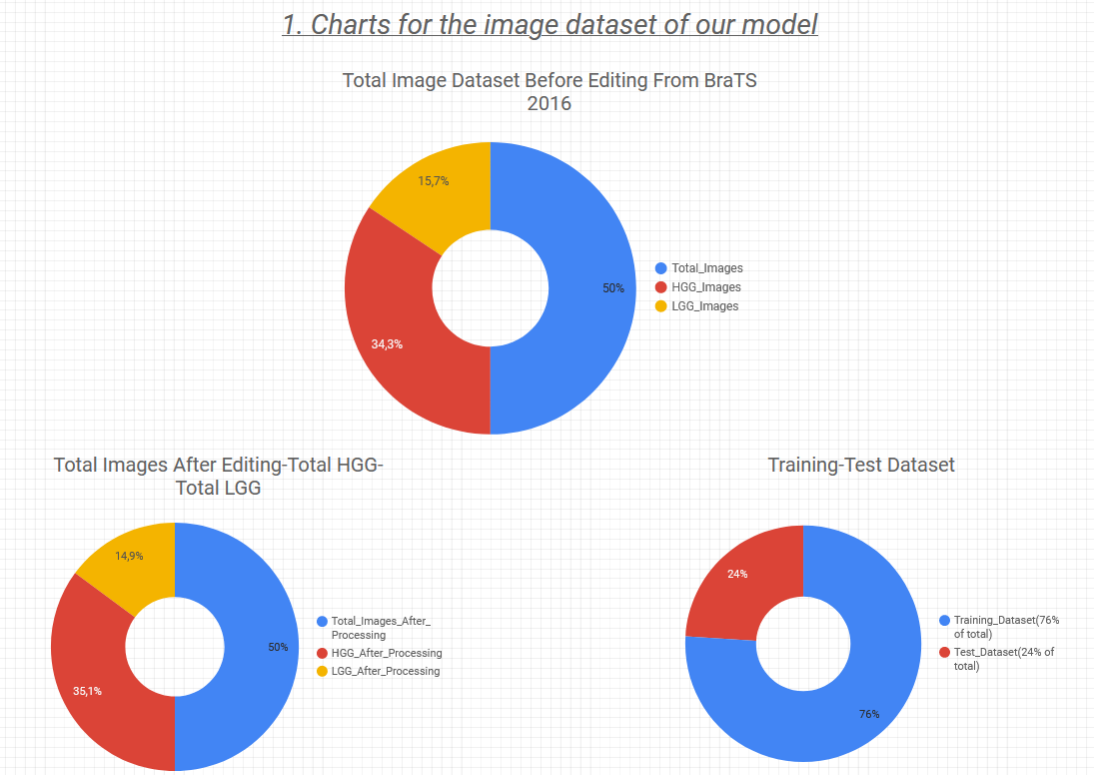
Μέσω της δυνατότητας που δίνεται από το Google Data Studio για εισαγωγή διάφορων πηγών δεδομένων, για τους σκοπούς της έρευνας που υλοποιήθηκε, χρησιμοποιήθηκαν αρχεία τύπου .csv για την απεικόνιση και δημιουργία της αντίστοιχης έκθεσης.

Τα .csv αρχεία που χρησιμοποιήθηκαν είναι ένα σύνολο δεδομένων που εξήχθησαν από το συνολικό άθροισμα των εικόνων πριν την επεξεργασία που έγινε για την εκπαίδευση του μοντέλου, αυτές που προέκυψαν μετά την επεξεργασία, συνολικό άθροισμα εικόνων με HGG γλοιώματα-LGG γλοιώματα, μετρήσεις εκτιμήσεων ακρίβειας, μετρήσεις εκτιμήσεων ανάκλησης και μετρήσεις εκτιμήσεων απωλειών.

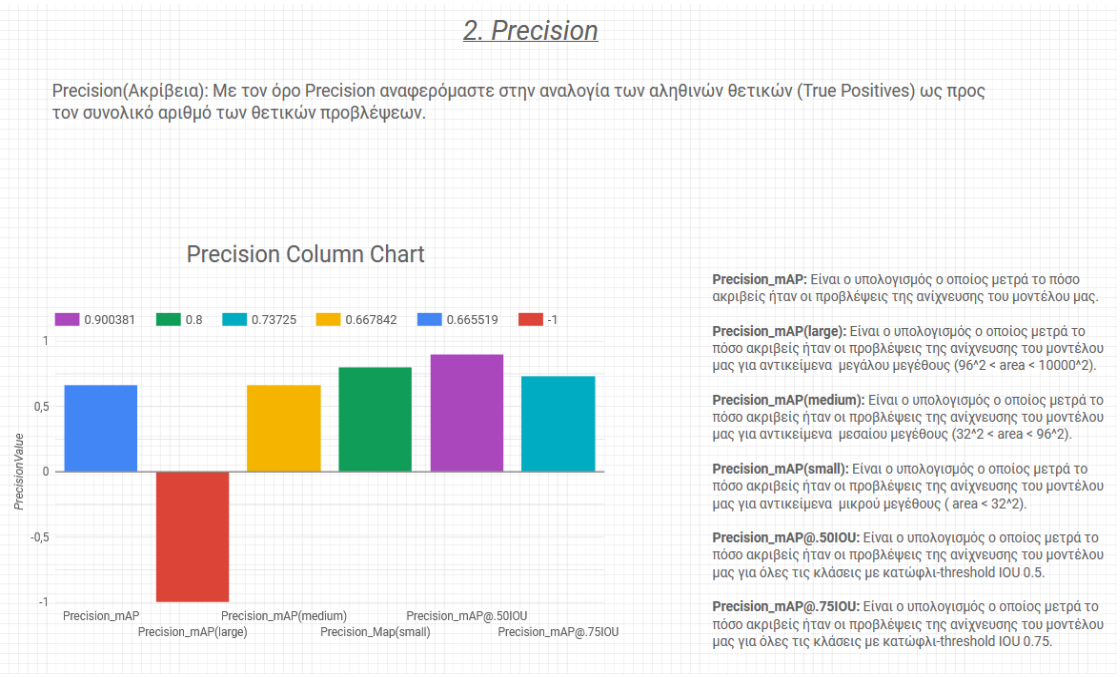
Πιο συγκεκριμένα:

1. **Pie Chart – Total Image Dataset Before Editing From BraTS 2016:** για απεικόνιση συνολικού αριθμού εικόνων.
2. **Pie Chart – Total Images After Editing-Total HGG-Total LGG:** για απεικόνιση συνολικού αριθμού επεξεργασμένων εικόνων.
3. **Pie Chart – Training-Test Dataset:** για απεικόνιση συνολικού αριθμού δεδομένων εκπαίδευσης-δεδομένων δοκιμής.
4. **Column Chart – Precision Column Chart:** για απεικόνιση τιμών ακρίβειας του μοντέλου (Precision).
5. **Column Chart – Recall Column Chart:** για απεικόνιση τιμών ανάκλησης του μοντέλου (Recall).
6. **Column Chart – Losses Column Chart:** για απεικόνιση τιμών απώλειας του μοντέλου (Box Classifier Losses, RPN Losses).

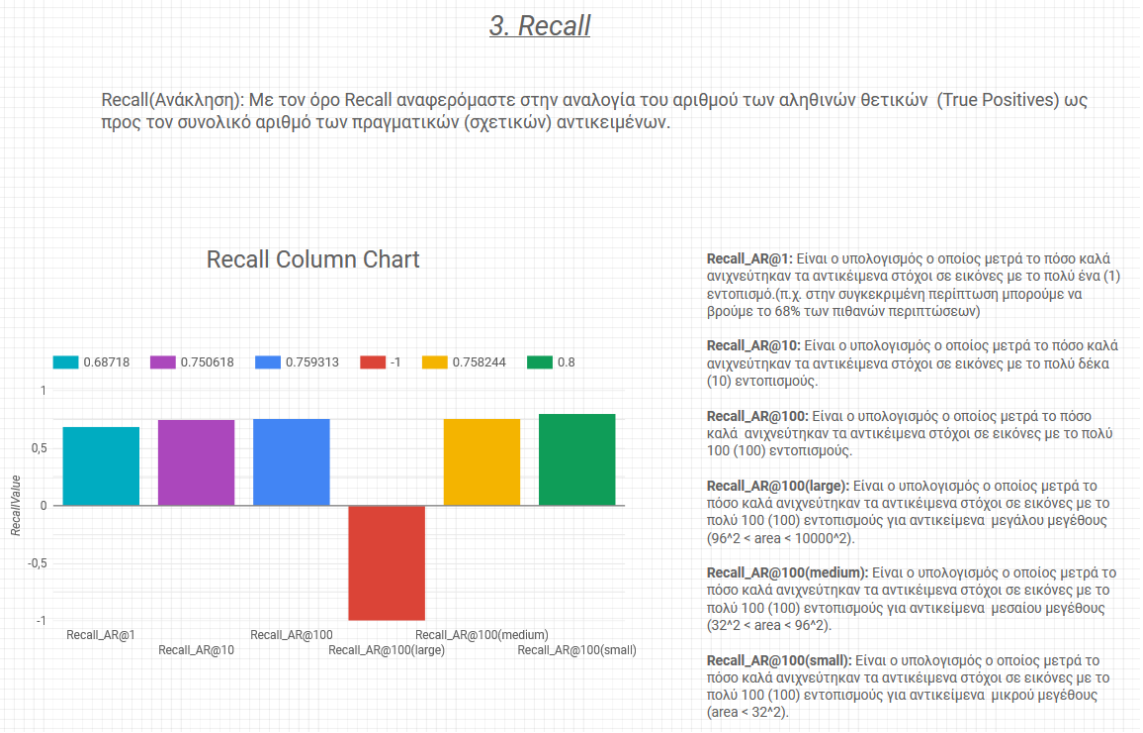
Στην συνέχεια παρουσιάζονται με screenshots τα διαγράμματα που υλοποιήθηκαν μέσω του Data Studio για το μοντέλο.



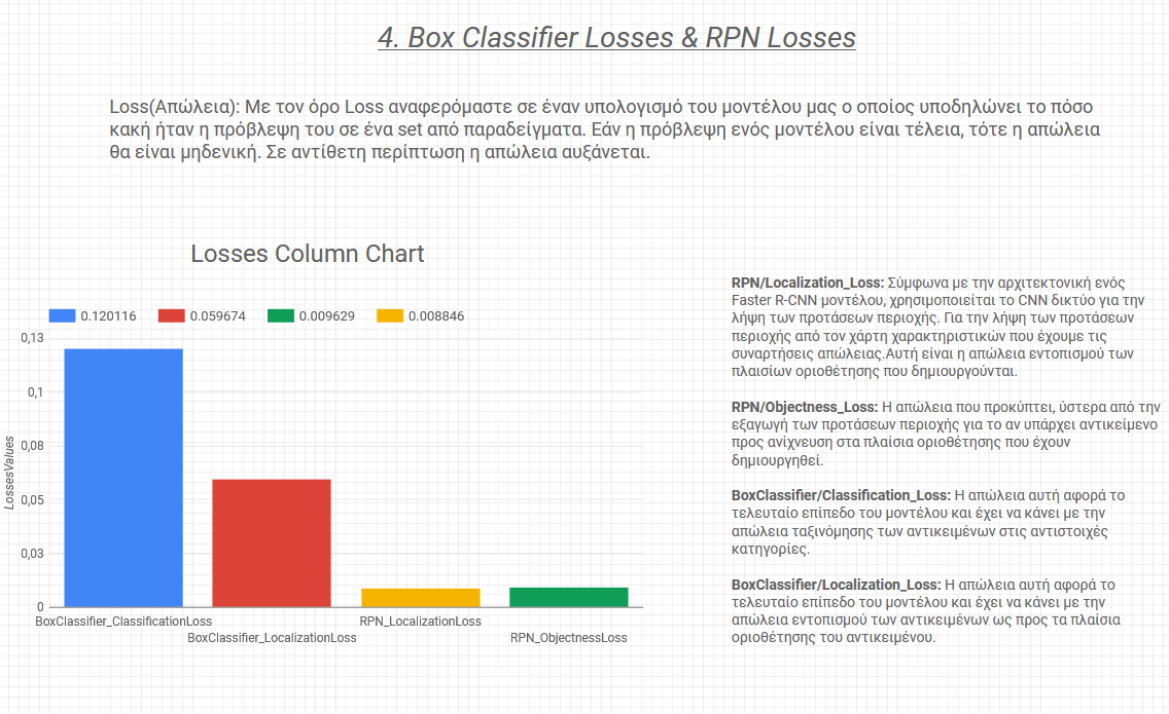
Εικόνα 52: Διάγραμμα πίτας του συνόλου των εικόνων



Εικόνα 53: Διάγραμμα ακρίβειας του μοντέλου



Εικόνα 54: Διάγραμμα ανάκλησης του μοντέλου



Εικόνα 55: Διάγραμμα απωλειών του μοντέλου

Η έκθεση είναι διαθέσιμη για ανάγνωση και σύγκριση αποτελεσμάτων για μελλοντικές έρευνες στον τομέα της ανίχνευσης όγκου στον εγκέφαλο μέσω του συνδέσμου https://datastudio.google.com/reporting/960cf71b-4677-4d6a-a841-564cde6d055e .

14. https://datastudio.google.com

9. Συμπεράσματα και μελλοντική ανάπτυξη

Κατά την υλοποίηση της συγκεκριμένης έρευνας αναλύθηκαν βασικές αρχές και αλγόριθμοι που αφορούν την υπολογιστική όραση και την ανίχνευση αντικειμένων με στόχο την αναζήτηση όγκων εγκεφάλου μέσω μαγνητικών τομογραφιών. Από το σύνολο των αλγορίθμων που αναπτύχθηκαν, χρησιμοποιήθηκε ένας Faster R-CNN αλγόριθμος, ο οποίος κρίθηκε αρκετά ικανοποιητικός και αποτελεσματικός ως προς το σύστημα και το σύνολο δεδομένων. Επίσης μέσω εργαλείων που χρησιμοποιήθηκαν αποδείχθηκε πως είναι δυνατή η δημιουργία ενός τέτοιου συστήματος για την υλοποίηση μιας ολοκληρωμένης εφαρμογής. Την εκπαίδευση του συστήματος ακολούθησε η αξιολόγηση του μοντέλου για την εξαγωγή των μετρήσεων ακρίβειας (precision) και ανάκλησης (recall), δείκτες οι οποίοι δηλώνουν κατά πόσο ο ανιχνευτής έχει την δυνατότητα να αναγνωρίζει σωστά αποτελέσματα για την ύπαρξη σχετικών όγκων στον εγκέφαλο καθώς και όσο το δυνατόν περισσότερα από αυτά.

Αφού ολοκληρώθηκε η εκπαίδευση, πραγματοποιήθηκε έλεγχος της ικανότητας αξιολόγησης του ανιχνευτή σε εικόνες στις οποίες η ύπαρξη καθώς και το είδος των όγκων ήταν από πριν γνωστά. Τα αποτελέσματα του μοντέλου εκπαίδευσης που υλοποιήθηκε είναι αρκετά ικανοποιητικά εφόσον σχεδόν πάντα ανιχνεύονται τα μέρη που υπάρχει όγκος, σε ένα σύνολο δεδομένων που αποτελείται από 450 εικόνες.

Παρόλα αυτά τα αποτελέσματα του ανιχνευτή δεν ήταν πάντα έγκυρα όσον αφορά στην κατηγοριοποίηση του είδους του όγκου (HGG-LGG) και αυτό οφείλεται στους παρακάτω παράγοντες:

- Η ανίχνευση όγκων (HGG-LGG) βάσει ορατών χαρακτηριστικών με το ανθρώπινο μάτι είναι σχεδόν απίθανη. Μέσω του RPN που λειτουργεί σε ένα Faster R-CNN δίκτυο αντλούνται αρκετά χαρακτηριστικά που μπορούν να τους διαχωρίσουν. Αφού το σύνολο δεδομένων που τροφοδότησε τον ανιχνευτή για την εκπαίδευση δέχτηκε επεξεργασία ως προς τους τύπους αρχείων και των μεγεθών τους, παρουσιάστηκαν μεταβολές στα εικονοστοιχεία που οδήγησαν στην αλλοίωση του αποτελέσματος.
- Η εκπαίδευση του μοντέλου με ένα σύνολο των 450 (342 training-108 testing) εικόνων που ήταν το τελικό σύνολο δεδομένων μετά την επεξεργασία του, ώθησε το μοντέλο σε ανιχνεύσεις που δεν ήταν πάντα έγκυρες. Επιπλέον στο σύνολο των εικόνων περιέχονταν 316 HGG και 134 LGG. Αυτός είναι άλλος ένας λόγος που δίνει στο μοντέλο μεγαλύτερη ευαισθησία στους HGG όγκους. Ιδανικά θα έπρεπε να είχαμε ίδιο ποσοστό εικόνων για τα δύο είδη όγκων.

- Σημαντικός παράγοντας που δυσκόλεψε την αναγνώριση του είδους ενός όγκου είναι η μη ικανοποιητική ύπαρξη υπολογιστικής ισχύος συστήματος, γεγονός που οδήγησε στη μείωση της ταχύτητας και του ποσοστού εγκυρότητας, στοιχεία που δεν είναι αποδεκτά για εφαρμογές που αφορούν ιατρικούς σκοπούς.

Για μελλοντική ανάπτυξη της έρευνας θα ήταν απαραίτητη η χρήση μεγαλύτερου συνόλου δεδομένων ασθενών και η χρήση εικόνων μεγαλύτερης ανάλυσης. Επίσης θα μπορούσε να οριστεί μεγαλύτερο εύρος ασθενειών εγκεφάλου για πιο συγκεκριμένη και εμπεριστατωμένη έρευνα ως προς το μέρος του εγκεφάλου στο οποίο παρατηρείται ο όγκος αλλά και για τον τύπο αυτού.

10. Βιβλιογραφία

1) Redmon, Joseph and Ali Farhadi. "YOLO9000: Better, Faster, Stronger." 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017): 6517-6525.
2) J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), σ.σ. 779–788, June 2016.
3) Robert J. Jackson, Gregory N. Fuller, Dima Abi-Said, Frederick F. Lang, Ziya L. Gokaslan, Wei Ming Shi, David M. Wildrick, Raymond Sawaya, Limitations of stereotactic biopsy in the initial management of gliomas, *Neuro-Oncology*, Volume 3, Issue 3, July 2001, Pages 193–200, https://doi.org/10.1093/neuonc/3.3.193
4) M. Y. W. Teow, "Understanding convolutional neural networks using a minimal model for handwritten digit recognition," in 2017 IEEE 2nd International Conference on Automatic Control and Intelligent Systems (I2CACIS), σ.σ. 167–172, (Oct 2017).
5) G. Nebehay and R. Pflugfelder, "Clustering of Static-Adaptive correspondences for deformable object tracking," in Computer Vision and Pattern Recognition, σ.σ. 2784– 2791, IEEE(June 2015).
6) Li, Zhifeng and Xiaoou Tang. "Bayesian face recognition using support vector machine and face clustering." Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004. 2 (2004).
7) T. Gevers and A. W. Smeulders, "Color-based object recognition," Pattern Recognition, vol. 32, no. 3, 453 – 464, (1999).
8) R. A. Khushboo Khurana, "Techniques for object recognition in images and multi-object detection," International Journal of Advanced Research in Computer Engineering & Technology, vol. 2, (apr 2013).
9) Pereira, Sérgio, et al. "Brain tumor segmentation using convolutional neural networks in MRI images." IEEE transactions on medical imaging 35.5 (2016).
10) Seetha, J., and S. Selvakumar Raja. "Brain tumor classification using convolutional neural networks." Biomedical & Pharmacology Journal 11.3 (2018).
11) Russell, Stuart, Norvig, Peter. Artificial Intelligence: A Modern Approach. 3rd edition. Prentice Hall, 2009. ISBN 0-13-604259-7.
12) Banerjee, Subhashis, et al. "Deep Radiomics for Brain Tumor Detection and Classification from Multi-Sequence MRI." arXiv preprint arXiv:1903.09240v1 (2019).

13) Z. Kalal, K. Mikolajczyk, and J. Matas, “Tracking-learning-detection,” IEEE trans-actions on pattern analysis and machine intelligence, vol. 34, no. 7, σ.σ. 1409– 1422, 2012.

14) S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks” in Advances in neural information processing systems, σ.σ. 91–99, 2015.

15) R. Girshick, “Fast r-cnn”, arXiv:1504.08083v2, 2015.

16) W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, “Ssd: Single shot multibox detector,” in European conference on computer vision, 2016.

17) Jasper RR Uijlings “Selective search for object recognition”. International journal of computer vision 104.2 (2013).

18) Abadi, M.; Agarwal, A. et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. 2015.

19) Louis, David N., et al. "The 2016 World Health Organization classification of tumors of the central nervous system: a summary." Acta Neuropathologica 131.6 (2016): 803-820.

20) H. N. T. K. Kaldera, S. R. Gunasekara and M. B. Dissanayake, "Brain tumor Classification and Segmentation using Faster R-CNN," 2019 Advances in Science and Engineering Technology International Conferences (ASET), 2019.

21) M. Ali, S. O. Gilani, A. Waris, K. Zafar and M. Jamil, "Brain Tumour Image Segmentation Using Deep Networks," in IEEE Access, vol. 8, σ.σ. 153589-153598, 2020.

22) Grosan, C., Abraham, A. (2011). Artificial Neural Networks. In: Intelligent Systems. Intelligent Systems Reference Library, vol 17. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-21004-4_12

23) Amish Jahangir Kapoor et al 2019 IOP Conf. Ser.: Earth Environ. Sci. 234 012061

24) Y. LeCun, K. Kavukcuoglu, and C. Farabet, “Convolutional networks and applications in vision,” in Proceedings of 2010 IEEE International Symposium on Circuits and Systems, 253–256, May 2010.

25) Bo Gao, M. W. Spratling, “Robust Template Matching via Hierarchical Convolutional Features from a Shape Biased CNN”, arXiv:2007.15817v3, 2021.

26) Lin Chen, Franz Rottensteiner, Christian Heipke (2021) Feature detection and description for image matching: from hand-crafted design to deep learning, Geo-spatial Information Science, 24:1, 58-74, DOI: 10.1080/10095020.2020.1843376