ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ & ΗΛΕΚΤΡΟΝΙΚΩΝ ΜΗΧΑΝΙΚΩΝ

Διπλωματική Εργασία

Αυτόματη αναγνώριση θέσης στάθμευσης με την χρήση Μηχανικής μάθησης

Φοιτητής: Βασίλειος Καρβέλας

ΑΜ: 262017035

Επιβλέπων Καθηγητής

Κάχρης Χριστόφορος

Επίκουρος Καθηγητής

ΑΘΗΝΑ-ΑΙΓΑΛΕΩ, ΙΟΥΛΙΟΣ 2024

UNIVERSITY OF WEST ATTICA

FACULTY OF ENGINEERING

DEPARTMENT OF ELECTRICAL & ELECTRONICS ENGINEERING

Diploma Thesis

Free Parking spot detection using Computer Vision

Student: Vasileios Karvelas

Registration Number: 262017035

Supervisor

Kachris Christoforos

Assistant Professor

ATHENS-EGALEO, JULY 2024

Η Διπλωματική Εργασία έγινε αποδεκτή και βαθμολογήθηκε από την εξής τριμελή επιτροπή:

Κάχρης Χριστόφορος, Επίκουρος Καθηγητής	Πυρομάλλης Δημήτριος, Αναπληρωτής Καθηγητής	Μετάφας Δημήτριος, Επίκουρος Καθηγητής
ΧΡΙΣΤΟΦΟΡΟΣ ΚΑΧΡΗΣ Digitally signed by ΧΡΙΣΤΟΦΟΡΟΣ ΚΑΧΡΗΣ Date: 2024.08.02 17:47:55 +03'00' (Υπογραφή)	DIMITRIOS PYROMALIS Digitally signed by DIMITRIOS PYROMALIS Date: 2024.07.30 10:11:07 +03'00' (Υπογραφή)	ΔΗΜΗΤΡΙΟΣ ΜΕΤΑΦΑΣ Digitally signed by ΔΗΜΗΤΡΙΟΣ ΜΕΤΑΦΑΣ Date: 2024.07.31 09:09:21 +03'00' (Υπογραφή)

ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ Βασίλειος Καρβέλας,
Μήνας, 2024

ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Ο κάτωθι υπογεγραμμένος Βασίλειος Καρβέλας του Γεωργίου, με αριθμό μητρώου 262017035 φοιτητής του Πανεπιστημίου Δυτικής Αττικής της Σχολής ΜΗΧΑΝΙΚΩΝ του Τμήματος ΗΛΕΚΤΡΟΛΟΓΩΝ ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΜΗΧΑΝΙΚΩΝ,

δηλώνω υπεύθυνα ότι:

«Είμαι συγγραφέας αυτής της διπλωματικής εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, οι όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε ακριβώς είτε παραφρασμένες, αναφέρονται στο σύνολό τους, με πλήρη αναφορά στους συγγραφείς, τον εκδοτικό οίκο ή το περιοδικό, συμπεριλαμβανομένων και των πηγών που ενδεχομένως χρησιμοποιήθηκαν από το διαδίκτυο. Επίσης, βεβαιώνω ότι αυτή η εργασία έχει συγγραφεί από μένα αποκλειστικά και αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο δικής μου, όσο και του Ιδρύματος.

Παράβαση της ανωτέρω ακαδημαϊκής μου ευθύνης αποτελεί ουσιώδη λόγο για την ανάκληση του διπλώματός μου.

Επιθυμώ την απαγόρευση πρόσβασης στο πλήρες κείμενο της εργασίας μου μέχρι …… και έπειτα από αίτησή μου στη Βιβλιοθήκη και έγκριση του επιβλέποντος καθηγητή.»

Ο Δηλών: Βασίλειος Καρβέλας

(ΚΕΝΟ ΦΥΛΛΟ)

ΕΥΧΑΡΙΣΤΙΕΣ

Αρχικά θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου, κύριο Χριστόφορο Κάχρη, για την ανάθεση του ενδιαφέροντος αυτού θέματος, καθώς επίσης και για τη βοήθεια που μου παρείχε καθόλη τη διάρκεια της εκπόνησης της διπλωματικής εργασίας.

Επίσης, οφείλω να ευχαριστήσω θερμά την οικογένειά μου, για την αστείρευτη αγάπη και την υποστήριξη που μου επέδειξαν σε όλη την ακαδημαϊκή μου πορεία.

Τέλος, θα ήθελα να μοιραστώ την ευγνωμοσύνη μου με τους αγαπημένους φίλους, τον Αλέξανδρο, τον Βασίλη, τον Δημήτρη και τον Φίλιππο για την φιλία και τη συμπαράσταση που μου επέδειξαν, καθόλη τη διάρκεια των σπουδών μου.

Περίληψη

Το επιστημονικό πεδίο της μηχανικής όρασης, είναι ένας από τους πιο δημοφιλείς και ταχέως αναπτυσσόμενους υπό-κλάδους της Τεχνητής νοημοσύνης. Χάρη σε αυτό τα υπολογιστικά συστήματα αποκτούν την ικανότητα όραση, επιτρέποντας τους να αντιλαμβάνονται και να ερμηνεύουν διάφορα γεγονότα και καταστάσεις στο πραγματικό κόσμο. Η παρούσα διπλωματική εργασία εστιάζει στην ανάπτυξη ενός συστήματος, αυτόματης αναγνώρισης ελεύθερων θέσεων στάθμευσης.

Αρχικά, πραγματοποιείται μία εισαγωγή στο πεδίο της Τεχνητής νοημοσύνης και σε σχετικούς, με το αντικείμενο της εργασίας, υπό-κλάδους της. Ακολουθεί σχολιασμός σχετικών δημοσιεύσεων, ενώ στη συνέχεια παρουσιάζονται οι απαιτήσεις του συστήματος, καθώς και το επιλεγμένο υλικό και λογισμικό. Στο επόμενο στάδιο, πραγματοποιείται η εκπαίδευση μοντέλων βασισμένων σε τρεις από τις εκδόσεις του αλγορίθμου YOLO. Στόχος της εκπαίδευσης είναι η επιλογή του μοντέλου που παρουσιάζει τις καλύτερες επιδόσεις στην αναγνώριση των θέσεων στάθμευσης.

Τέλος, το επιλεγμένο μοντέλο εφαρμόζεται στην τελική συσκευή, η οποία αποτελείται από έναν μικροϋπολογιστή Raspberry Pi 4 και μία κάμερα. Η συσκευή πραγματοποιεί αναγνώριση και καταμέτρηση των ελεύθερων θέσεων στάθμευσης από ζωντανή ροή και οι πληροφορίες σχετικά με την διαθεσιμότητα των χώρων στάθμευσης παρουσιάζονται σε ένα Dashboard Node-Red.

Λέξεις – κλειδιά

Μηχανική Όραση, Raspberry Pi, Python, OpenCV, YOLOv5, συσκευή ανίχνευσης κενών θέσεων στάθμευσης, Node-Red.

Abstract

The scientific field of machine vision is one of the most popular and rapidly growing sub-fields of Artificial Intelligence. Thanks to this, computing systems acquire the ability to see, allowing them to perceive and interpret various events and situations in the real world. This thesis focuses on the development of an automatic parking space recognition system.

Initially, an introduction is made to the field of Artificial Intelligence and its subfields that are relevant to the subject of this thesis. Followed by the review of related publications and then the requirements of the system, as well as the selected hardware and software. In the next step, models based on three versions of the YOLO algorithm are trained. The goal of the training is to compare the trained models and select the one that performs best in recognizing parking spaces.

Finally, the selected model is applied to the final device, which consists of a Raspberry Pi 4 and a camera. The device recognizes and counts both the occupied and unoccupied parking spaces from a live stream. Using a Node-Red Dashboard the information about the availability of the parking spaces is displayed.

Keywords

Machine Vision, Raspberry Pi, Python, OpenCV, YOLOv5, parking spot detection, Node-Red.

ΠΕΡΙΕΧΟΜΕΝΑ

Κατάλογος Πινάκων

Κατάλογος Εικόνων

Αλφαβητικό Ευρετήριο

ANN	Artificial Neural Network	I2C	Inter-Integrated Circuit
ARM	Advanced RISC Machines	SaaS	Software as a Service
CNN	Convolutional Neural Network	IoU	Intersection over Union
CPU	Central Processing Unit	Onnx	Open Neural Network Exchange
CV	Computer Vision	NumPy	Numerical Python
DNN	Deep Neural Network	LPDDR	Low-Power Double Data Rate
GPIO	General-Purpose Input-Output	SSD	Single Shot Multi-Box Detector
GUI	Graphical User Interface	UI	User Interface
IDE	Integrated Development Environment	AI	Artificial Intelligence
IoT	Internet Of Things	PWM	Pulse Width Modulation
mAP	Mean Average Precision	SPI	Serial Peripheral Interface
ML	Machine Learning	UART	Universal Asynchronous RX/TX
MQTT	Message Queuing Telemetry System		
OpenCV	Open Computer Vision Library		
OS	Operating System		
PGI	Programmable Gradient Information		
R-CNN	Region Based Convolutional Neural Network		
RPi	Raspberry Pi		
SBC	Single Board Computer		
SoC	System On A Chip		
YOLO	You Only Look Once		

(ΚΕΝΟ ΦΥΛΛΟ)

ΕΙΣΑΓΩΓΗ

Η αυξανόμενη χρήση των αυτοκινήτων, ιδίως στα αστικά κέντρα, έχει επιφέρει μια σειρά από προβλήματα, με κυριότερο την έλλειψη επαρκών θέσεων στάθμευσης. Η κατασκευή νέων υποδομών για τη στέγαση των οχημάτων, καθίσταται πλέον ανέφικτη, καθώς επιφέρει αρνητικές επιπτώσεις τόσο στην ποιότητα ζωής των ανθρώπων όσο και στο περιβάλλον.

Συνεπώς η αναζήτηση διαθέσιμης θέσης για τη στάθμευση των οχημάτων, εξελίσσεται σε μία χρονοβόρα διαδικασία, με τους οδηγούς να σπαταλούν πολύτιμο χρόνο, ψάχνοντας για μία κενή θέση. Συχνά οι διαθέσιμοι χώροι στάθμευσης είναι πλήρεις, καθιστώντας αναγκαία την περαιτέρω αναζήτηση.

Αυτό το πρόβλημα μπορεί να περιοριστεί με την υιοθέτηση ενός συστήματος πληροφόρησης για τη διαθεσιμότητα των θέσεων στάθμευσης. Τέτοια, αυτοματοποιημένα συστήματα παρακολούθησης, συναντώνται κυρίως σε ιδιωτικούς χώρους στάθμευσης, ενώ στους δημόσιους χώρους η υιοθέτηση τους υστερεί. Τα σύγχρονα συστήματα, βασίζονται σε λύσεις τεχνητής νοημοσύνης για την αποτελεσματική διαχείριση της στάθμευσης.

Αντικείμενο της Διπλωματικής Εργασίας

Αντικείμενο της παρούσας διπλωματικής εργασίας αποτελεί η ανάπτυξη ενός συστήματος αναγνώρισης κενών θέσεων στάθμευσης. Το σύστημα αυτό βασίζεται στην μηχανική όραση, η οποία αποτελεί, όπως έχει αναφερθεί, έναν κλάδο της τεχνητής νοημοσύνης.

Μεθοδολογία

Αρχικά παρουσιάστηκε το απαραίτητο θεωρητικό υπόβαθρο και έγιναν αναφορές σε σχετικές, με το αντικείμενο της εργασίας, δημοσιεύσεις. Έπειτα ακολούθησε η ανάλυση των απαιτήσεων και των απαραίτητων στοιχείων για την υλοποίηση του συστήματος. Μετά από την ανάλυση των απαραίτητων βημάτων, υλοποιήθηκε η διαδικασία εκπαίδευσης μοντέλων βασισμένων σε ορισμένες από τις εκδόσεις και παραλλαγές των μοντέλων του αλγόριθμου YOLO. Στη συνέχεια, διατυπώθηκαν τα συμπεράσματα της εργασίας, αφού προηγήθηκε η πιλοτική λειτουργία του συστήματος.

Σκοπός και στόχοι

Σκοπός της εργασίας, είναι η ανάπτυξη και ο σχεδιασμός μίας έξυπνης συσκευής, με την δυνατότητα της αυτόματης αναγνώρισης και καταμέτρησης κενών θέσεων στάθμευσης. Η συσκευή αυτή θα μπορεί να λειτουργήσει σε οποιαδήποτε χώρο στάθμευσης.

Βασικός στόχος της εργασίας είναι η επιλογή του μοντέλου με την υψηλότερη ακρίβεια για την αναγνώριση κενών θέσεων στάθμευσης. Για την επίτευξη του στόχου αυτού, θα εκπαιδευτούν πολλαπλά μοντέλα, βασισμένα σε διαφορετικές εκδόσεις του αλγορίθμου YOLO.

Δευτερεύων, αλλά εξίσου σημαντικός στόχος, είναι η οπτικοποίηση των δεδομένων σε πραγματικό χρόνο. Η οπτικοποίηση θα πραγματοποιηθεί μέσω ενός απλού και εύχρηστου πίνακα (Dashboard), ο οποίος θα παρέχει τις απαραίτητες πληροφορίες στους επισκέπτες.

Καινοτομία

Ένα από τα στοιχεία της εργασία που θα μπορούσε να θεωρηθεί ως καινοτόμο, είναι η διαφορετική προσέγγιση, που πάρθηκε για την αντιμετώπιση του προβλήματος της έλλειψης πληροφοριών σχετικά με τη διαθεσιμότητα θέσεων στάθμευσης σε μη ελεγχόμενους χώρους στάθμευσης. Εκμεταλλευόμενοι τις δυνατότητες του συστήματος, το οποίο αποτελείται από το εκπαιδευμένο μοντέλο του αλγορίθμου YOLO, σε συνδυασμό με το Raspberry Pi και οπτικό αισθητήρα, κατασκευάστηκε μία οικονομικά προσιτή συσκευή που αναγνωρίζει, με υψηλή ακρίβεια, τον αριθμό και την πληρότητα των διαθέσιμων θέσεων στάθμευσης.

Η τοποθέτηση και χρήση τη συσκευής, σε χώρους με αυξημένη κυκλοφορία οχημάτων και περιορισμένες θέσεις στάθμευσης, όπως η πανεπιστημιούπολη, μπορεί να οδηγήσει στην μείωση του θορύβου αλλά και των ρύπων που εκπέμπονται από τα οχήματα.

Δομή

Η δομή της διπλωματικής εργασίας χωρίζεται σε έξι κεφάλαια.

- Κεφάλαιο 1ο: Το πρώτο κεφάλαιο της εργασίας εστιάζει στο απαραίτητο θεωρητικό υπόβαθρο, καθώς επίσης και στα σχετικά με την εργασία υποπεδία της Τεχνητής Νοημοσύνης.

- Κεφάλαιο 2ο: Αναφορά σε δημοσιεύσεις και άρθρα που ασχολούνται με θέμα την ανίχνευση θέσεων στάθμευσης. Επίσης παρουσιάζονται τα κριτήρια επιλογής και οι δημοσιεύσεις

- Κεφάλαιο 3ο: Το τρίτο κεφάλαιο χωρίζεται σε τρία μέρη. Στο πρώτο μέρος σχολιάζονται οι απαιτήσεις του συστήματος. Έπειτα στο δεύτερο μέρος παρουσιάζεται το απαραίτητο υλικό και λογισμικό που επιλέχθηκε. Το τρίτο μέρος αποτελεί οδηγό εγκατάστασης, όπου παρουσιάζονται αναλυτικά τα βήματα για την εγκατάσταση του απαραίτητου λογισμικού.

- Κεφάλαιο 4ο: Πρώτα παρουσιάζονται τα απαραίτητα βήματα για την εκπαίδευση των μοντέλων. Έπειτα εκπαιδεύονται έξι μοντέλα YOLO (εκδόσεις 5,8 και 10), για την ανίχνευση θέσεων στάθμευσης. Στη συνέχεια παρουσιάζονται και σχολιάζονται τα αποτελέσματα εκπαίδευσης.

- Κεφαλαίο 5ο: Η συσκευή δοκιμάζεται στην εξαγωγή αποτελεσμάτων από εικόνες, βίντεο και σε πραγματικό χρόνο. Επιπλέον υλοποιείται ένα Dashboard και οπτικοποιούνται τα αποτελέσματα.

- Κεφάλαιο 6ο: Αναφέρονται τα συμπεράσματα που προέκυψαν από την εκπόνηση της διπλωματικής εργασίας. Τέλος αναφέρονται οι μελλοντικές βελτιώσεις της εργασίας.

Κεφάλαιο 1ο : Θεωρητικό Υπόβαθρο

1.1 Τεχνητή Νοημοσύνη

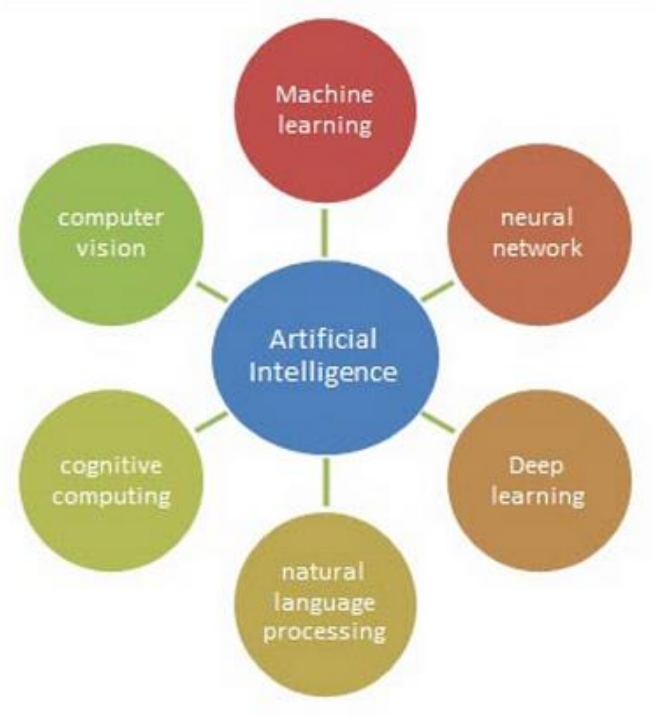
Η Τεχνητή Νοημοσύνη (Artificial Intelligence) αποτελεί τον κλάδο της πληροφορικής που ασχολείται με την ανάπτυξη και σχεδίαση υπολογιστικών συστημάτων, τα οποία μιμούνται χαρακτηριστικά από νοήμονες οντότητες [1]. Συγκεκριμένα, η Τεχνητή Νοημοσύνη περιλαμβάνει τη δημιουργία αλγορίθμων και μοντέλων που προσδίδουν την ικανότητα να εκτελούν διεργασίες που απαιτούν στοιχεία ανθρώπινης νοημοσύνης, όπως η αντίληψη, η αναγνώριση και η λήψη αποφάσεων. Η Τεχνητή Νοημοσύνη αποτελεί ένα διεπιστημονικό πεδίο, καθώς αντλεί και συνδυάζει στοιχεία από διάφορους κλάδους, όπως η επιστήμη των υπολογιστών, η φιλοσοφία, τα μαθηματικά, η στατιστική και η ψυχολογία.

Η ιδέα μηχανών που επιδεικνύουν στοιχεία ανθρώπινης νοημοσύνης και συμπεριφοράς υπήρχε από την αρχαιότητα, με αναφορές σε ευφυή μηχανήματα σε μύθους και ιστορίες. Το πεδίο της τεχνητής νοημοσύνης πρωτοεμφανίστηκε στα μέσα του 20ου αιώνα, μετά την ανάπτυξη των πρώτων υπολογιστικών συστημάτων [2].

Ο Alan Turing, πρωτοπόρος της τεχνητής νοημοσύνης, δημοσίευσε το άρθρο «Computer Machinery and Intelligence», προτείνοντας το "Test Turing" για την αξιολόγηση νοημοσύνης σε μηχανές. Το 1943, οι Warren McCulloch και Walter Pitts παρουσίασαν το πρώτο μοντέλο τεχνητών νευρώνων. Λίγα χρόνια αργότερα, το 1956, ο Jason McCarthy διατύπωσε τον πρώτο ορισμό της τεχνητής νοημοσύνης στη διάσκεψη του "Darthmouth".

Από τότε έως και σήμερα, ό τομέας της τεχνητής νοημοσύνης έχει διανύσει μία ταραχώδη πορεία. Περίοδοι άνθησης, όπως η "AI Boom" (1980 - 1987), σημαδεύτηκαν από ραγδαία ανάπτυξη και αυξημένο ενδιαφέρον. Αντιθέτως, σε περιόδους ύφεσης, όπως η "AI Winter" (1987 – 1993), η πρόοδος και το ενδιαφέρον έπεσαν αισθητά.

Η Τεχνητή Νοημοσύνη διαιρείται σε υποσύνολα [3], όπως παρουσιάζονται στην Εικόνα 1. Περιλαμβάνει την Μηχανική Μάθηση, την Υπολογιστική Όραση, την Βαθιά Μάθηση, τα Τεχνητά Νευρωνικά Δίκτυα, την Γνωστική Υπολογιστική και την Επεξεργασία Φυσικής Γλώσσας.



Εικόνα 1: Υποσύνολα της Τεχνητής Νοημοσύνης [πηγή]

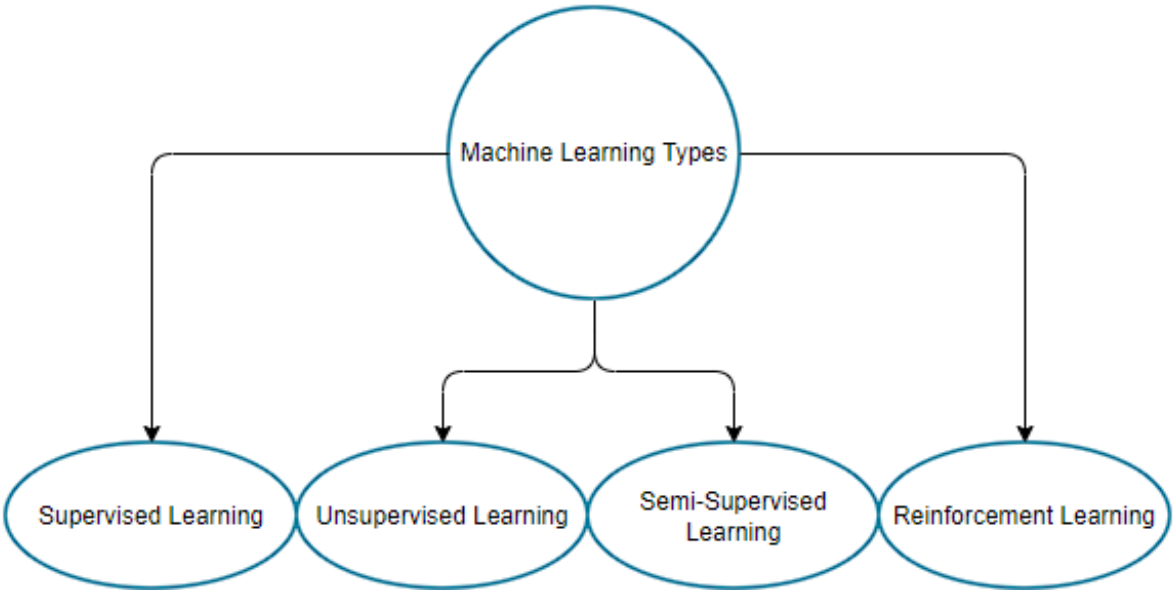
Στις υποενότητες που ακολουθούν, θα αναλύσουμε ορισμένους από τους κλάδους της τεχνητής νοημοσύνης, καθώς και τα στοιχεία τους που συσχετίζονται με το απαραίτητο γνωστικό υπόβαθρο που αφορά το αντικείμενο της εργασίας.

1.2 Μηχανική Μάθηση

Η Μηχανική Μάθηση (Machine Learning), σύμφωνα με τον Arthur Samuel (1959), αποτελεί ένα υπό-πεδίο της Τεχνητής Νοημοσύνης που εστιάζει στην ανάπτυξη αλγορίθμων. Αυτοί οι αλγόριθμο προσδίδουν στα υπολογιστικά συστήματα την ικανότητα να μαθαίνουν αυτόνομα, εκτελώντας διεργασίες για τι οποίες δεν έχουν ρητά προγραμματιστεί [4]. Στην πράξη, οι αλγόριθμοι μηχανικής μάθησης αποσκοπούν στην μείωση ή ακόμα και στην εξάλειψη της ανθρώπινης παρέμβασης στη διαδικασία της μάθησης.

Οι αλγόριθμοι Μηχανικής Μάθησης ταξινομούνται σε τέσσερεις βασικές κατηγορίες:

1. **Επιβλεπόμενη Μάθηση (Supervised Learning):** Στην επιβλεπόμενη μάθηση, οι αλγόριθμοι εκπαιδεύονται με δεδομένα εκπαίδευσης που έχουν ετικέτες που υποδεικνύουν την επιθυμητή έξοδο.
2. **Μη επιβλεπόμενης Μάθηση (Unsupervised Learning):** Στη μη επιβλεπόμενη μάθηση, οι αλγόριθμοι εκπαιδεύονται με δεδομένα εκπαίδευσης που δεν έχουν ετικέτες. Ο στόχος είναι να ανακαλύψουν μοτίβα ή δομές στα δεδομένα.
3. **Μερικώς Επιβλεπόμενη Μάθηση (Semi-Supervised Learning):** Στη μερικώς επιβλεπόμενη μάθηση, οι αλγόριθμοι εκπαιδεύονται με ένα μείγμα δεδομένων με ή χωρίς ετικέτες.
4. **Ενισχυμένη Μάθηση (Reinforcement Learning):** Στην ενισχυμένη μάθηση, οι αλγόριθμοι μαθαίνουν αλληλοεπιδρώντας με ένα περιβάλλον και λαμβάνοντας ανταμοιβές ή ποινές για τις ενέργειες τους.



Εικόνα 2: Κατηγορίες αλγορίθμων Μηχανικής Μάθησης

1.3 Κατηγορίες αλγορίθμων Μηχανικής μάθησης

1.3.1 Επιβλεπόμενη Μάθηση (Supervised Learning)

Στην επιβλεπόμενη μάθηση, ένας αλγόριθμος εκπαιδεύεται για να παράγει ένα μοντέλο, χρησιμοποιώντας ένα γνωστό σύνολο δεδομένων που αποτελείται από παραδείγματα δεδομένων με ετικέτες (labeled examples) $\{(x_i, y_i)\}_{i=1}^{N}$. Κάθε δείγμα στο σύνολο δεδομένων ονομάζεται διάνυσμα χαρακτηριστικών (feature vector) και περιέχει μία τιμή x_j για κάθε στοιχείο.

Η ετικέτα εξόδου y μπορεί να είναι απλή, για παράδειγμα, ένα στοιχείο από ένα πεπερασμένο σύνολο κλάσεων, ένας πραγματικός αριθμός ή μια πιο σύνθετη δομή, όπως ένα διάνυσμα, ένας πίνακας ή ένας γράφος. Το μοντέλο επιβλεπόμενης μάθησης λαμβάνει ένα νέο διάνυσμα εισόδου το επεξεργάζεται, εξάγει τις σχετικές πληροφορίες και καταλήγει σε μία πρόβλεψη για την ετικέτα του.

Τα προβλήματα επιβλεπόμενης μάθησης κατηγοριοποιούνται σε δύο κύριους τύπους:

1.3.1.1 Ταξινόμηση (Classification)

Η ταξινόμηση αποτελεί ένα πρόβλημα επιβλεπόμενης μάθησης, όπου ετικέτες εκχωρούνται σε μη επισημασμένα δείγματα, με στόχο την αντιμετώπιση κατηγοριών μεταβλητών εξόδου. Στο πλαίσιο της ταξινόμησης, γίνεται η κατάταξη ενός δείγματος σε μία από τις γνωστές κλάσεις. Ο ταξινομητής, ή αλλιώς αλγόριθμος κατηγοριοποίησης, εκπαιδεύεται με βάση τα επισημασμένα δεδομένα του συνόλου εκπαίδευσης. Λαμβάνοντας στην είσοδο του τα χαρακτηριστικά των νέων δειγμάτων, καθώς και τα δεδομένα εισόδου, ο ταξινομητής προβλέπει την κλάση στην οποία ανήκουν τα βήματα.

Οι αλγόριθμοι ταξινόμησης διακρίνονται σε δύο βασικές κατηγορίες: γραμμικούς και μη-γραμμικούς. Στους γραμμικούς αλγορίθμους ταξινόμησης συγκαταλέγονται η Λογιστική Παλινδρόμηση (Logistic Regression) και οι Μηχανές Διανυσμάτων Υποστήριξης (Support Vector Machines). Από την άλλη, οι μη-γραμμικοί αλγόριθμοι περιλαμβάνουν τις Μηχανές Διανυσμάτων Πυρήνα (Kerner Support Vector Machines), τον αλγόριθμο Naive Bayes, τα Δέντρα Αποφάσεων (Decision Trees) και τον αλγόριθμο k-Nearest Neighbors.

1.3.1.2 Παλινδρόμηση (Regression)

Η παλινδρόμηση αποτελεί ένα είδος επιβλεπόμενης μάθησης που ερευνά πιθανές συσχετίσεις μεταξύ εξαρτημένων και ανεξάρτητων μεταβλητών. Εφαρμόζεται στην πρόβλεψη ετικετών με συνεχή μορφή. Στόχος των μοντέλων παλινδρόμησης είναι η εύρεση της συνάρτησης *f*, η οποία αντιστοιχεί μία ανεξάρτητη τιμή x (δεδομένα εισόδου) σε μία εξαρτημένη μεταβλητή εξόδου.

Υπάρχουν διάφοροι αλγόριθμοι, με τους δημοφιλείς να περιλαμβάνουν τη Γραμμική Παλινδρόμηση (Linear Regression), την Πολυωνυμική Παλινδρόμηση (Polynomial Regression), τα Παλινδρομικά Δέντρα Αποφάσεων (Decision Tree Regression) και τα Παλινδρομικά Διανύσματα Στήριξης (Support Vector Regression).

1.3.2 Μη Επιβλεπόμενη Μάθηση (Unsupervised Learning)

Στη μη επιβλεπόμενη μάθηση, το σύνολο δεδομένων αποτελείται από μη επισημασμένα δείγματα $\{x_i\}_{i=1}^{N}$. Το σύνολο δεδομένων περιλαμβάνει μόνο δεδομένα εισόδου και το μοντέλο καλείται να ανακαλύψει μοτίβα ή σχέσεις στα δεδομένα. Τα προβλήματα μη επιβλεπόμενης μάθησης μπορούν να κατηγοριοποιηθούν περαιτέρω σε προβλήματα συσταδοποίησης (Clustering) και συσχέτισης (Association).

1.3.2.1 Συσταδοποίηση (Clustering)

Η συσταδοποίηση, είναι ένα πρόβλημα της μη επιβλεπόμενης μάθησης, που αξιοποιεί μη επισημασμένα δεδομένα για να ομαδοποιήσει δείγματα με βάση τα κοινά χαρακτηριστικά τους. Στόχος της συσταδοποίησης είναι η αναγνώριση ομάδων που αποτελούνται από δείγματα με παρόμοια χαρακτηριστικα.

Υπάρχουν διάφοροι αλγόριθμοι ομαδοποίησης, όπως η Ιεραρχική ομαδοποίηση (Hierarchical Clustering), οι k-μέσοι (k-means), η Φασματική ομαδοποίηση (Spectral Clustering) και η Ομαδοποίηση βάσει μοντέλου (Model Based Clustering).

1.3.2.2 Συσχέτιση (Association)

Οι κανόνες συσχέτισης, αποτελούνται από μεθόδους, μη επιβλεπόμενης μάθησης και χρησιμοποιούνται στην εύρεση σχέσεων, μεταξύ των μεταβλητών που ανήκουν σε ένα μεγάλο σύνολο δεδομένων.

1.3.3 Μερικώς Επιβλεπόμενη μάθηση (Semi-Supervised Learning)

Η μερικώς επιβλεπόμενη μάθηση αποτελεί μία τεχνική μάθησης που εντάσσεται μεταξύ της επιβλεπόμενης και της μη επιβλεπόμενης μάθησης. Βασίζεται στην αξιοποίηση τόσο δεδομένων με επισημάνσεις όσο και δεδομένων χωρίς. Στην περίπτωση της μερικώς επιβλεπόμενης μάθησης, το μεγαλύτερο μέρος του συνόλου δεδομένων παραμένει μη επισημασμένο, ενώ ένα μικρότερο κομμάτι φέρει επισημάνσεις.

1.3.4 Ενισχυτική μάθηση (Reinforcement Learning)

Στην ενισχυτική μάθηση, το σύνολο δεδομένων διαφοροποιείται από τις προηγούμενες κατηγορίες, μιας και είναι δυναμικό. Το μοντέλο καλείται να αλληλοεπιδράσει με ένα περιβάλλον που εξελίσσεται συνεχώς, έχοντας τη δυνατότητα να εκτελεί ενέργειες σε οποιαδήποτε κατάσταση. Κάθε ενέργεια που υλοποιείται επιφέρει μία ανταμοιβή και μπορεί να επηρεάσει τη τρέχουσα κατάσταση του περιβάλλοντος.

Απώτερος στόχος της ενισχυτικής μάθησης είναι η εκμάθηση μιας πολιτικής, η οποία επιτρέπει στο μοντέλο να λαμβάνει τη βέλτιστη ανταμοιβή όταν εκπληρώνει έναν συγκεκριμένο στόχο. Η πολιτική, γνωστή και ως στρατηγική ενεργειών, απεικονίζεται με τη συνάρτηση f, η οποία δέχεται ως είσοδο ένα διάνυσμα χαρακτηριστικών που περιγράφει την τρέχουσα κατάσταση.

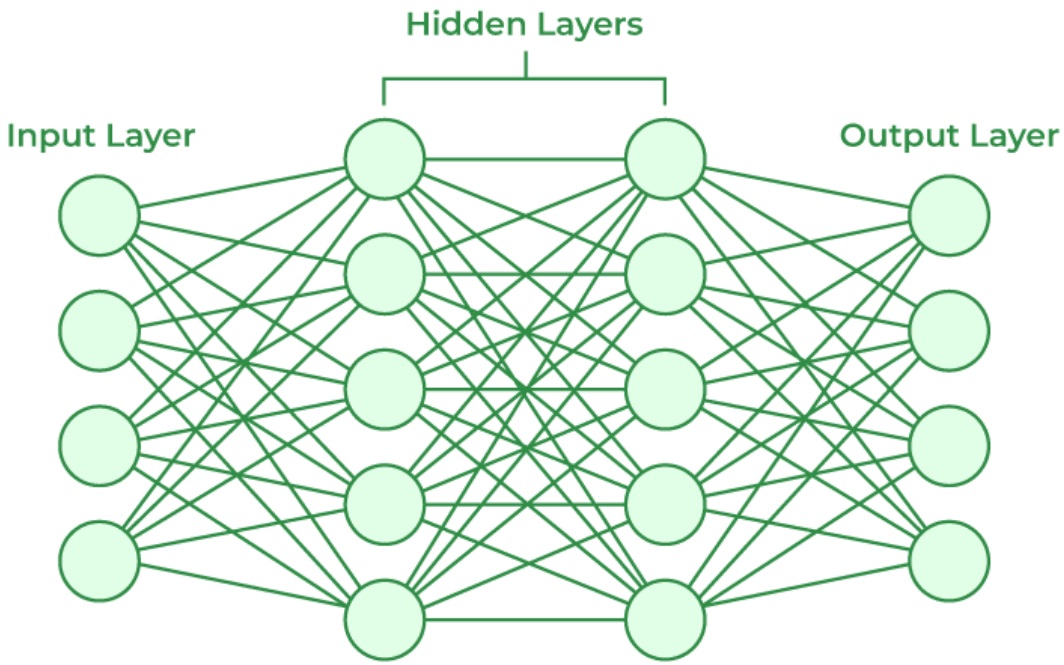
1.4 Νευρωνικά Δίκτυα

Τα Νευρωνικά δίκτυα αποτελούν έναν τύπο αλγορίθμου μηχανικής και υπό-πεδίο της Βαθιάς Μάθησης. Η λειτουργία και η δομή τους βασίζονται στη δομή και τη λειτουργία του ανθρωπίνου εγκεφάλου [5]. Εφαρμογές νευρωνικών δικτύων εντοπίζονται σε διάφορα πεδία της Τεχνητής Νοημοσύνης, όπως η αναγνώρισης εικόνων, η αναγνώριση φωνής και η μηχανική μετάφραση.

Αποτελούνται από πολλαπλά διασυνδεδεμένα επίπεδα νευρώνων, τα οποία με τη σειρά τους συνδέονται με τα γειτονικά επίπεδα. Κάθε νευρώνα στο δίκτυο έχεις ως σκοπό την επεξεργασία και μετάδοση πληροφοριών στο αμέσως επόμενο επίπεδο.

1.4.1 Δομή Νευρωνικού Δικτύου

Ένα νευρωνικό δίκτυο αποτελείται από τρία βασικά επίπεδα: τα επίπεδα εισόδου (Input Layers), τα κρυφά επίπεδα (Hidden Layers) και τα επίπεδα εξόδου (Output Layer). Στην εικόνα 3 απεικονίζεται ένα νευρωνικό δίκτυο:



Εικόνα 3: Αρχιτεκτονική Νευρωνικού Δικτύου [πηγή]

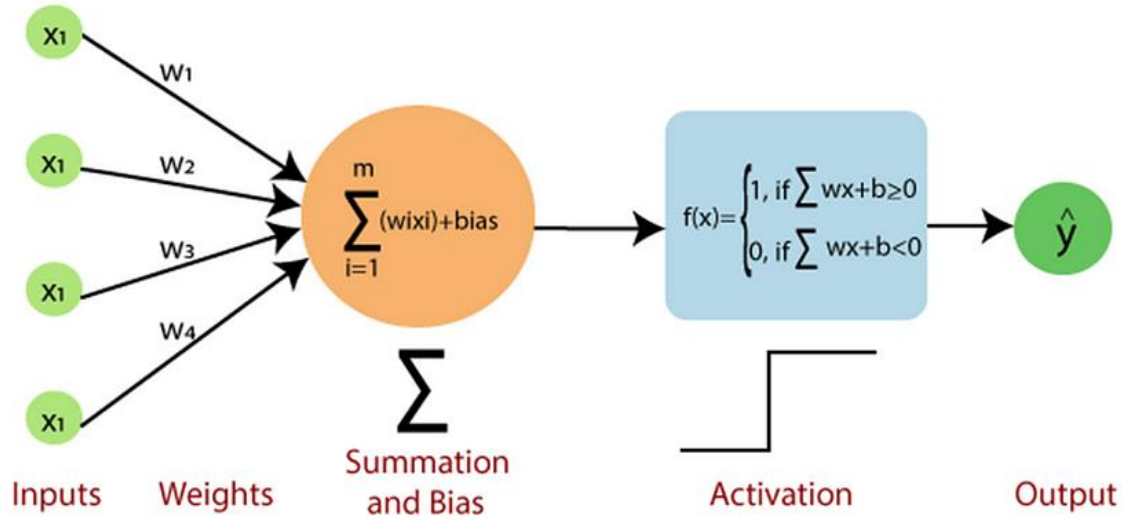
- **Επίπεδο εισόδου:** Αποτελεί το πρώτο επίπεδο του δικτύου και λαμβάνει τα δεδομένα εισόδου.
- **Κρυφά επίπεδα:** Βρίσκονται ανάμεσα στο επίπεδο εισόδου και στο επίπεδο εξόδου. Χρησιμοποιούνται για την εκμάθηση και τον μετασχηματισμό των δεδομένων.
- **Επίπεδο εξόδου:** Αποτελεί το τελευταίο επίπεδο του δικτύου και παράγει το τελικό αποτέλεσμα, την πρόβλεψη.

1.5 Τύποι Νευρωνικών Δικτύων

Υπάρχουν πολλοί τύποι νευρωνικών δικτύων, κάθε ένα από τα οποία χρησιμοποιείται για την επίλυση διαφορετικών προβλημάτων. Στα παρακάτω υπό-κεφάλαια ακολουθούν μερικοί από τους πιο συνηθισμένους

1.5.1 Τεχνητός Νευρώνας – Perceptron

Το Perceptron, το οποίο δημιουργήθηκε από τον ψυχολόγο Frank Rosenblatt, το 1957, αποτελεί το παλαιότερο μοντέλο τεχνητού νευρώνα και το πρώτο νευρωνικό δίκτυο με αλγοριθμική διατύπωση. Ως η απλούστερη μορφή εμπρόσθιου νευρωνικού δικτύου, βρίσκει εφαρμογή κυρίως σε εργασίες εποπτευόμενης μάθησης, με σκοπό την ταξινόμηση γραμμικά διαχωρίσιμών δεδομένων [6]. Το Perceptron λειτουργεί ως μονάδα νευρωνικού δικτύου, εκτελώντας υπολογισμούς για την ταξινόμηση δεδομένων στις αντίστοιχες κλάσεις τους.



Εικόνα 4: Ο Τεχνητός Νευρώνας Perceptron [πηγή]

1.5.1.1 Βασικά Στοιχεία του Perceptron

Στην εικόνα 4, φαίνεται το Perceptron, το οποίο αποτελείται από τρία βασικά στοιχεία:

1. **Σύνολο Συνάψεων (Input Features):** Κάθε είσοδος x_i πολλαπλασιάζεται με ένα συναπτικό βάρος w_i, παράγοντας το γινόμενο $x_i w_i$.

2. **Αθροιστής (Adder):** Στον αθροιστή, τα γινόμενα $x_i w_i$ αθροίζονται, μαζί με μία εξωτερική πόλωση (bias) b_k. Η πόλωση επηρεάζει τη διέγερση της συνάρτησης ενεργοποίησης.

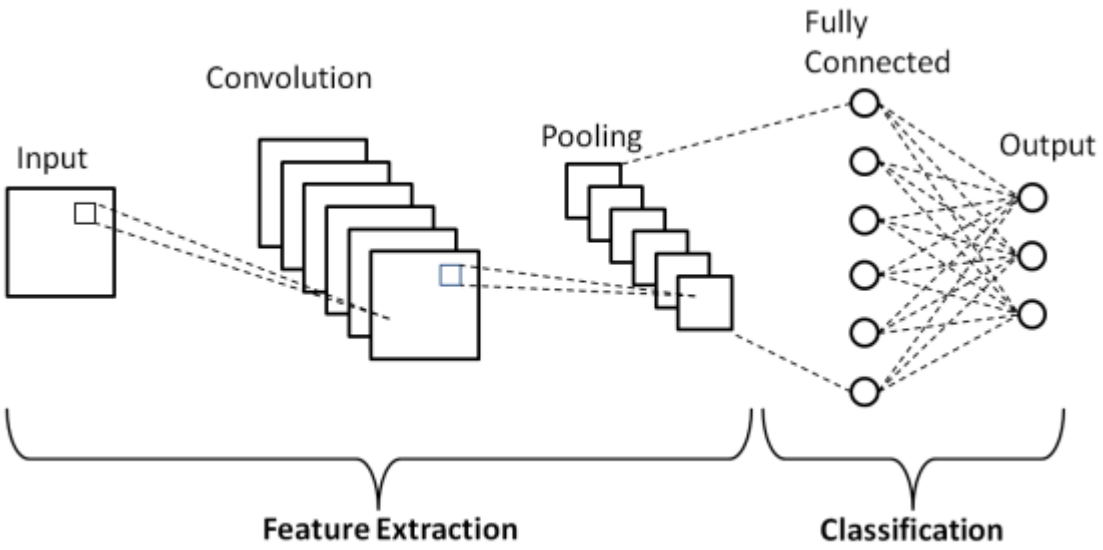
3. **Συνάρτηση Ενεργοποίησης (Activation Function):** Η συνάρτηση ενεργοποίησης ή συνάρτηση περιορισμού, περιορίζει το εύρος εξόδου του νευρώνα [-1,1].

1.5.2 Συνελικτικά Νευρωνικά Δίκτυα (Convolutional Neural Networks)

Τα συνελικτικά νευρωνικά δίκτυα αποτελούν ένα εξελιγμένο είδος τεχνητών νευρωνικών δικτύων, τα οποία χρησιμοποιούνται κυρίως στον τομέα της αναγνώρισης προτύπων για την εξαγωγή χαρακτηριστικών από εικόνες [7]. Πρόκειται για δίκτυα πολλαπλών επιπέδων, που αποτελούνται από τον συνδυασμό και τη διασύνδεση διαφόρων επιπέδων.

Αυτά τα επίπεδα, αναφορικά διακρίνονται σε τέσσερις βασικούς τύπους: Επίπεδα εισόδου (Input Layers), Επίπεδα συνέλιξης (Convolution Layers), Επίπεδα ενεργοποίησης (Activation Layers), Επίπεδα συγκέντρωσης (Pooling Layers), Πλήρως συνδεδεμένα επίπεδα (Fully Connected Layers).

Στην Εικόνα 5 απεικονίζεται η αρχιτεκτονική ενός συνελικτικού νευρωνικού δικτύου με τα επίπεδα που το απαρτίζουν.



Εικόνα 5: Αρχιτεκτονική Συνελικτικού Νευρωνικού Δικτύου

1.5.2.1 Αρχιτεκτονική Συνελικτικού Νευρωνικού Δικτύου

Παρακάτω αναφέρονται, συνοπτικά, τα επίπεδα του Συνελικτικού Νευρωνικού Δικτύου:

- **Επίπεδο Εισόδου (Input Layer):** Το επίπεδο εισόδου αποτελεί το αρχικό στάδιο του δικτύου και δέχεται ως είσοδο μία ή περισσότερες εικόνες.

- **Επίπεδο Συνέλιξης (Convolution Layer):** Το επίπεδο συνέλιξης αποτελεί βασικό δομικό στοιχείο ενός συνελικτικού νευρωνικού δικτύου. Κατά τη διάρκεια της συνέλιξης, εφαρμόζονται φίλτρα (ή πυρήνες) στην εικόνα που εισήχθη στο προηγούμενο επίπεδο. Τα φίλτρα, όντας πίνακες μικρών διαστάσεων, φέρουν αντιστοιχούμενα βάρη. Η ολίσθηση των φίλτρων κατά μήκος των δεδομένων του πίνακα εισόδου επιτρέπει τον υπολογισμό του εσωτερικού γινομένου της τιμής του πυρήνα με την αντίστοιχη τιμή της εισόδου. Η ολίσθηση αυτή έχει ως αποτέλεσμα τη δημιουργία ενός δυσδιάστατου χάρτη ενεργοποίησης (2D Activation Map) στην έξοδο του επιπέδου συνέλιξης. Ο χάρτης ενεργοποίησης καταγράφει τα χαρακτηριστικά που ανιχνεύει το φίλτρο σε κάθε θέση της εικόνας εισόδου.

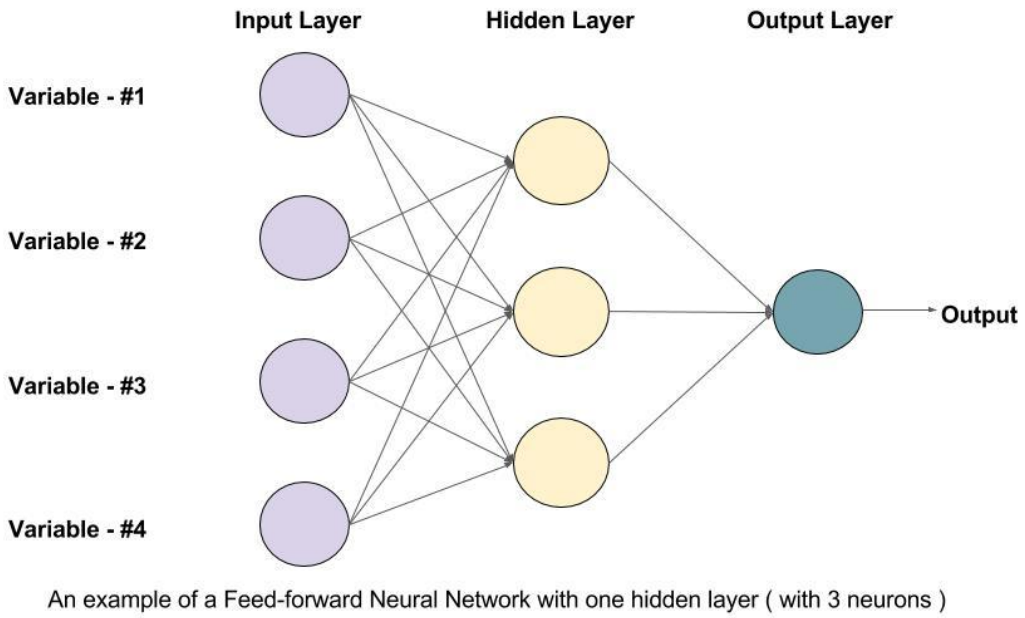
- **Επίπεδο Ενεργοποίησης (Activation Layer):** Τα επίπεδα ενεργοποίησης εισάγουν μία μη γραμμική συνάρτηση ενεργοποίησης μετά από κάθε επίπεδο συνέλιξης. Η συνάρτηση “ReLu” αποτελεί την πιο συνήθως χρησιμοποιούμενη επιλογή.

- **Επίπεδο Συγκέντρωσης (Pooling Layer):** Το επίπεδο συγκέντρωσης λαμβάνει ως είσοδο την έξοδο του προηγούμενου επιπέδου συνέλιξης και συρρικνώνει την εικόνα. Μέσω αυτής της διαδικασίας, μειώνεται ο αριθμός των παραμέτρων και η απαραίτητη υπολογιστική ισχύς, διατηρώντας παράλληλα τις χρήσιμες πληροφορίες. Η συρρίκνωση επιτυγχάνεται με τη χρήση μαθηματικών συναρτήσεων, όπως η μέγιστη συγκέντρωση (Max Pooling Operation) και η μέση συγκέντρωση (Average Pooling Operation). Η μέγιστη συγκέντρωση αποτελεί την πιο συχνά χρησιμοποιούμενη μέθοδο συγκριτικά με τη μέση.

- **Πλήρως Συνδεδεμένο Επίπεδο (Fully Connected Layer):** Το πλήρως συνδεδεμένο επίπεδο αποτελεί το τελικό στάδιο του νευρωνικού δικτύου. Σε αυτό το επίπεδο, οι νευρώνες συνδέονται με όλες τις εξόδους του προηγούμενου επιπέδου. Ο σκοπός του πλήρως συνδεδεμένου επιπέδου είναι η αξιοποίηση των χαρακτηριστικών που εξήχθησαν από τη διαδικασία συνέλιξης, με στόχο την ταξινόμηση της εικόνας εισόδου σε μία από τις διαθέσιμες κατηγορίες.

1.5.3 Νευρωνικά δίκτυα Εμπρόσθιας Τροφοδότησης (Feed-Forward Neural Networks)

Τα νευρωνικά δίκτυα εμπρόσθιας τροφοδότησης αποτελούν την απλούστερη μορφή νευρωνικού δικτύου [8]. Σε αυτά τα δίκτυα, δεν υπάρχουν βρόγχοι ανάδρασης και η πληροφορία ρέει αποκλειστικά προς μία μόνο κατεύθυνση. Κάθε νευρώνας ενός επιπέδου συνδέεται με όλους τους νευρώνες του επόμενου. Κάθε σύνδεση μεταξύ νευρώνων φέρει συναπτικά βάρη, τα οποία προσαρμόζονται βάσει του σφάλματος εξόδου. Στην εικόνα 6 απεικονίζεται ένα δίκτυο εμπρόσθιας τροφοδότησης.



Εικόνα 6: Δίκτυο Εμπρόσθιας Τροφοδότησης [πηγή]

1.5.3.1 Αρχιτεκτονική Δικτύου Εμπρόσθιας Τροφοδότησης

Παρακάτω αναφέρονται, συνοπτικά τα επίπεδα του Δικτύου Εμπρόσθιας Τροφοδότησης:

- **Επίπεδο Εισόδου (Input Layer):** Το επίπεδο εισόδου αποτελείται από νευρώνες οι οποίοι λαμβάνουν και μεταβιβάζουν δεδομένα στα επόμενα επίπεδα. Ο αριθμός των νευρώνων στο επίπεδο αυτό καθορίζεται από τις διαστάσεις των δεδομένων εισόδου.

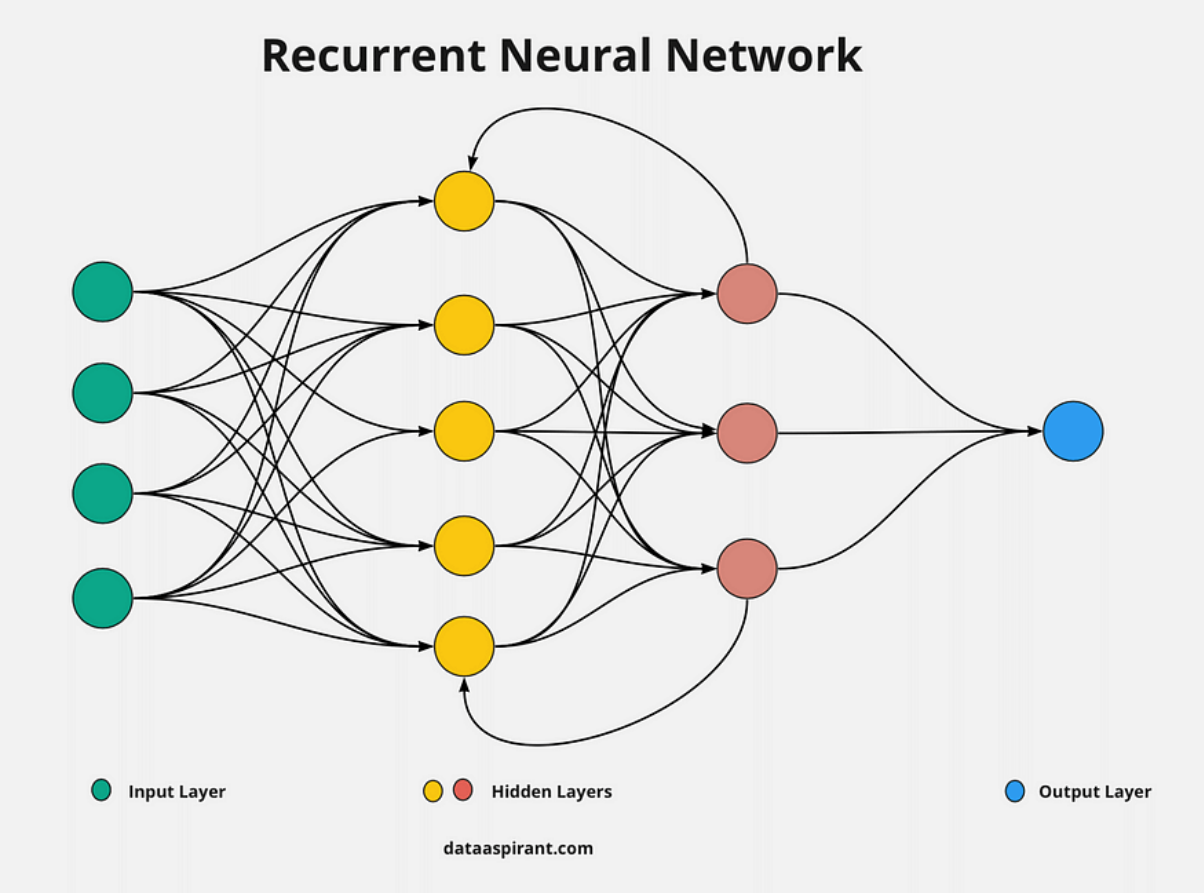
- **Κρυφά Επίπεδα (Hidden Layers):** Τα κρυφά επίπεδα είναι επίπεδα που παρεμβάλλονται μεταξύ των επιπέδων εισόδου και εξόδου. Σε αυτά τα επίπεδα λαμβάνουν χώρα οι υπολογισμοί. Κάθε κρυφό επίπεδο λαμβάνει το άθροισμα των συναπτικών βαρών από το προηγούμενο επίπεδο, το επεξεργάζεται με μία συνάρτηση ενεργοποίησης και το στέλνει στο επόμενο επίπεδο. Ένα απλό εμπρόσθιο νευρωνικό δίκτυο, μπορεί να μην έχει κρυφά επίπεδα.

- **Επίπεδο Εξόδου (Output Layer):** Στο επίπεδο εξόδου παράγεται η τελική έξοδος για τα δεδομένα εισόδου. Εάν υπάρχουν και άλλα επίπεδα, η πληροφορία μεταβιβάζεται στο επόμενο επίπεδο.

1.5.4 Επαναλαμβανόμενα Νευρωνικά Δίκτυα (Recurrent Neural Networks)

Τα επαναλαμβανόμενα ή αναδρομικά νευρωνικά δίκτυα, αποτελούν έναν τύπο νευρωνικού δικτύου που ενσωματώνει τουλάχιστον έναν βρόχο ανάδρασης [9]. Η ενσωμάτωση αυτή των βρόχων παρέχει στο δίκτυο την ικανότητα της αποθήκευσης πληροφοριών.

Ωστόσο, επηρεάζει σημαντικά τόσο την απόδοση όσο και τη δυνατότητα μάθησης του δικτύου. Τα επαναλαμβανόμενα νευρωνικά δίκτυα βρίσκουν εφαρμογή σε πεδία όπως η αναγνώριση ομιλίας και η επεξεργασία φυσικής γλώσσας.



Εικόνα 7: Επαναλαμβανόμενο Νευρωνικό Δίκτυο [πηγή]

1.5.4.1 Αρχιτεκτονική Επαναλαμβανόμενου Νευρωνικού Δικτύου

Στην Εικόνα 7 απεικονίζονται ένα επαναλαμβανόμενο νευρωνικό δίκτυο. Η δομή του παρουσιάζει ομοιότητες με τα νευρωνικά δίκτυα που εξετάστηκαν προηγουμένως, αποτελούμενο από επίπεδα εισόδου, κρυφά επίπεδα και επίπεδα εξόδου.

Το βασικό χαρακτηριστικό που διαφοροποιεί τα επαναλαμβανόμενα δίκτυα από τα υπόλοιπα εντοπίζεται στα κρυφά επίπεδα. Οι βρόγχοι ανάδρασης επιτρέπουν στο δίκτυο να λαμβάνει υπόψιν τις διαδοχικές εξαρτήσεις και να αποθηκεύει πληροφορίες από προηγούμενες εισόδους. Αυτές οι πληροφορίες μετατρέπονται σε στοιχεία μνήμης, τα οποία διατηρούνται και ενημερώνονται μετά από μία επανάληψη.

1.6 Συναρτήσεις Ενεργοποίησης (Activation Functions)

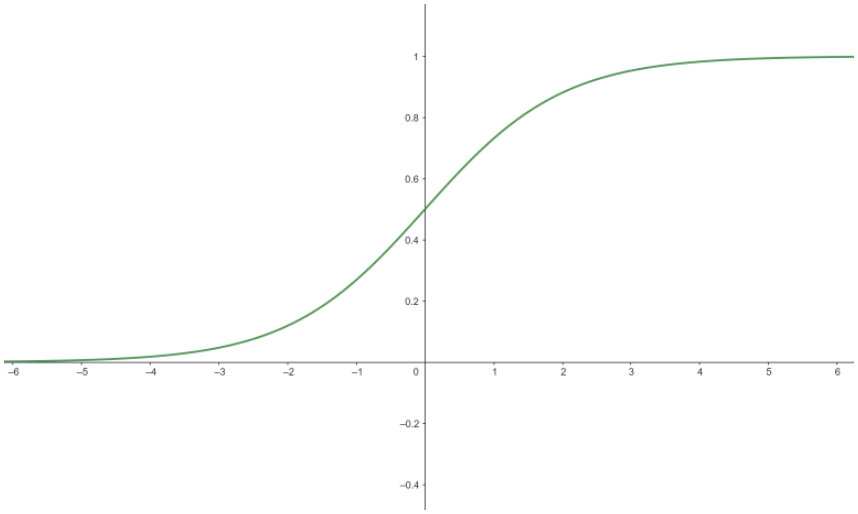
Σκοπός της συνάρτησης ενεργοποίησης είναι να καθορίζει την κατάσταση ενός νευρών, επιτρέποντας ή αποτρέποντας τη ροή πληροφοριών μεταξύ της εισόδου και της εξόδου του.

Οι συναρτήσεις ενεργοποίησης διακρίνονται σε γραμμικές και μη γραμμικές. Σήμερα, χρησιμοποιούνται κυρίως μη γραμμικές συναρτήσεις με την συνάρτηση “ReLu” να κατέχει την πρώτη θέση.

1.6.1 Σιγμοειδής συνάρτηση (Sigmoid Function)

Η σιγμοειδής συνάρτηση, αποτελεί μία μη γραμμική συνάρτηση που μετατρέπει μία τιμή εισόδου σε μία τιμή μεταξύ 0 και 1. Ανεξάρτητα από το μέγεθος της αρχικής τιμής εισόδου, η σιγμοειδής συνάρτηση θα περιορίσει την τιμή εισόδου σε αυτό το εύρος.

$$f(x) = \frac{1}{1 + e^{-x}}$$

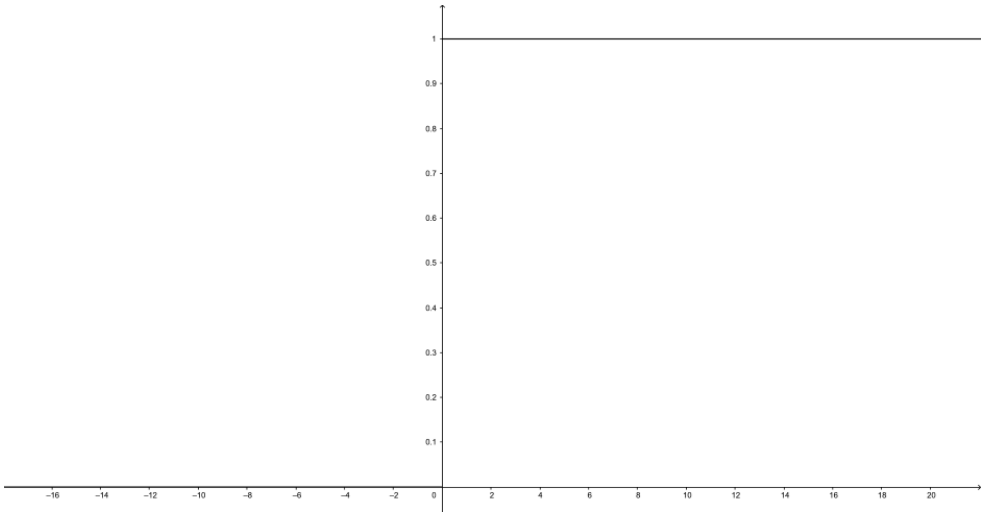


Εικόνα 8: Γραφική Παράσταση Σιγμοειδής Συνάρτησης

1.6.2 Συνάρτηση Κατωφλίου (Threshold Function)

Η λειτουργία του κατωφλίου εξαρτάται από μία τιμή, η οποία καθορίζει αν θα ενεργοποιηθεί ο νευρώνας.

$$f(x) = \begin{cases} 0 \ για \ x<0 \\ 1 \ για \ x\geq 0 \end{cases}$$

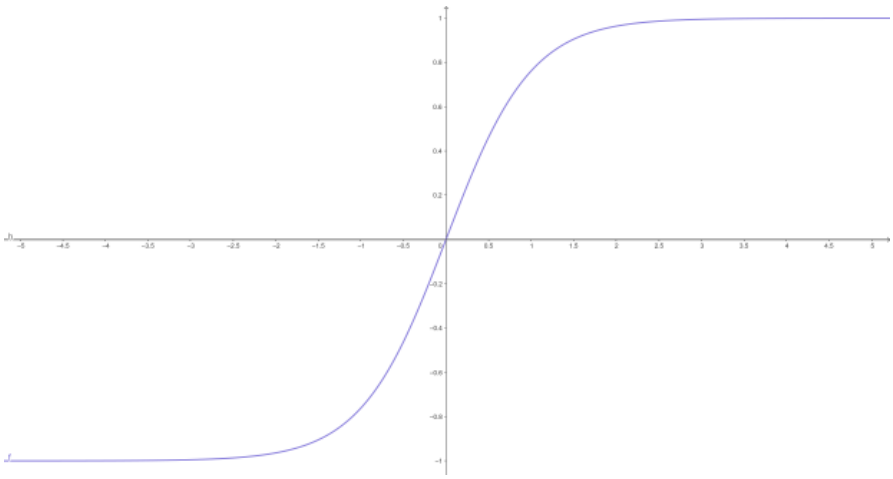


Εικόνα 9: Γραφική Παράσταση Συνάρτησης Κατωφλίου

1.6.3 Συνάρτηση Υπερβολικής Εφαπτομένης (Tanh Function)

Στην υπερβολική συνάρτηση εφαπτομένης, η τιμή εξόδου πλησιάζει τη μονάδα όσο η τιμή εισόδου αυξάνεται.

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$



Εικόνα 10: Γραφική Παράσταση Συνάρτησης Υπερβολικής Εφαπτομένης

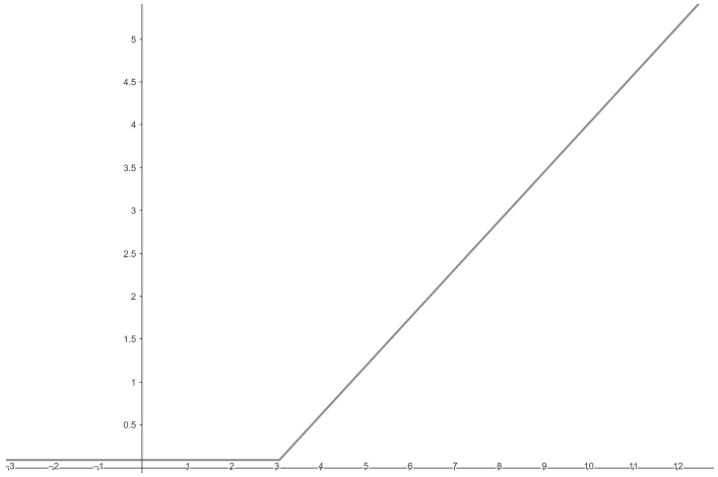
1.6.4 ReLu (Rectified Linear Unit)

Η ReLu είναι μία από τις πιο δημοφιλείς συναρτήσεις ενεργοποίησης στα νευρωνικά δίκτυα. Πρόκειται για μία μη γραμμική συνάρτηση, η οποία λειτουργεί ως εξής:

- Για θετική είσοδο x: Η ReLU επιστρέφει την αρχική τιμή x.
- Για αρνητική είσοδο x: Η ReLU επιστρέφει τον μηδέν

Ουσιαστικά η ReLU μηδενίζει τις αρνητικές εισόδους, επιτρέποντας στον νευρώνα να ενεργοποιείται μόνο για θετικές τιμές.

$$\boldsymbol{f(x)} = \max(0, x)$$



Εικόνα 11: Γραφική Παράσταση της ReLU

1.7 Εκπαίδευση Νευρωνικών Δικτύων

Η εκπαίδευση ενός νευρωνικού δικτύου αποτελεί μία διαδικασία κατά την οποία προσαρμόζονται τα βάρη και οι πολώσεις του δικτύου, αξιοποιώντας αλγορίθμους με στόχο τη μείωση του σφάλματος. Η μείωση αυτή επιτυγχάνεται μέσω της ελαχιστοποίησης μίας συνάρτησης απώλειας ή σφάλματος.

1.7.1 Τύποι μάθησης Νευρωνικών Δικτύων

Στα νευρωνικά δίκτυα, υπάρχουν δύο βασικοί τύποι μάθησης: την επιβλεπόμενη και την μη επιβλεπόμενη, όπως αυτές αναφέρονται στα υπό-κεφάλαια 1.3.1 και 1.3.2.

Σε αυτό το σημείο θα αναφερθούν συνοπτικά, οι δύο αυτοί τύποι.

Στην επιβλεπόμενη μάθηση, το δίκτυο εκπαιδεύεται σε ένα σύνολο δεδομένων με γνωστά ζεύγη εισόδου – εξόδου. Ο στόχος είναι το δίκτυο να μάθε να αντιστοιχεί σωστά κάθε είσοδο στην αντίστοιχη έξοδο. Για να το πετύχει αυτό, προσαρμόζει τα βάρη του κατά τη διάρκεια της εκπαίδευσης, ώστε να βελτιώσει την ικανότητα του να κάνει ακριβείς προβλέψεις για άγνωστα δεδομένα που θα λάβει στο μέλλον.

Στην μη επιβλεπόμενη μάθηση, το δίκτυο λαμβάνει δεδομένα εισόδου χωρίς καμία προκαθορισμένη ετικέτα ή κατηγορία. Ο στόχος εδώ είναι το δίκτυο να ανακαλύψει μόνο του τα μοτίβα και τις δομές που κρύβονται στα δεδομένα. Αυτό μπορεί να περιλαμβάνει τον εντοπισμό ομάδων δεδομένων με παρόμοια χαρακτηριστικά ή την εύρεση συσχετίσεων.

1.7.1.1 Κλίση Καθόδου (Gradient Descent)

Η κλίση καθόδου αποτελεί έναν αλγόριθμο βελτιστοποίησης που εφαρμόζεται στη Μηχανική Μάθηση και στη Βαθιά Μάθηση κατά τη διάρκεια της εκπαίδευσης ενός νευρωνικού δικτύου [11]. Σκοπός του αλγορίθμου είναι η ελαχιστοποίηση της συνάρτησης κόστους, φτάνοντας στο ελάχιστο σημείο της. Αυτό επιτυγχάνεται με την προσαρμογή των βαρών του δικτύου καθ' όλη τη διάρκεια της εκπαίδευσης και για κάθε επανάληψη.

Για την εύρεση αυτού του ελάχιστου σημείου, χρησιμοποιείται η κλίση που υπολογίζεται μέσω της οπισθοδιάδοσης. Η κλίση αυτή ορίζει την κατεύθυνση προς την οποία θα γίνει η ενημέρωση των παραμέτρων του μοντέλου, οδηγώντας σε ένα σημείο μικρότερο από το προηγούμενο.

Ένας παράγοντας που επηρεάζει την εύρεση του ολικού ελαχίστου, είναι ο ρυθμός μάθησης (learning rate), μια παράμετρος της εκπαίδευσης του δικτύου που τροποποιεί την ταχύτητα τροποποίησης της κλίσης. Η ταχύτητα, δηλαδή το βήμα κλίσης, προσαρμόζεται ανάλογα με τη συμπεριφορά της συνάρτησης κόστους. Σε περίπτωση που δεν καθοριστεί σωστά, μειώνεται η ακρίβεια και αυξάνεται ο χρόνος βελτιστοποίησης.

1.7.1.2 Οπισθοδιάδοση (Backpropagation)

Η οπισθοδιάδοση αποτελεί έναν από τους παλαιότερους και πιο χρήσιμους αλγόριθμους εκπαίδευσης νευρωνικών δικτύων με πολλά επίπεδα [12]. Η αξιοποίηση της κλίσης καθόδου στα δίκτυα εμπρόσθιας τροφοδότησης από τον αλγόριθμο οπισθοδιάδοσης άνοιξε τον δρόμο για συνεχή βελτίωση της εξόδου του δικτύου. Κατά τη διάρκεια της εκπαίδευσης, η οπισθοδιάδοση διαδίδει το σφάλμα από την έξοδο προς την είσοδο του δικτύου, λαμβάνοντας υπόψη το σφάλμα σε κάθε στάδιο για βελτιστοποίηση των συνδέσεων.

Σε κάθε επανάληψη, υπολογίζεται η κλίση της συνάρτησης απώλειας ως προς κάθε βάρος σε κάθε στρώμα του δικτύου, με κατεύθυνση από την έξοδο προς την είσοδο. Αυτό επιτρέπει την τροποποίηση των βαρών με τρόπο που μειώνει το σφάλμα και βελτιώνει την ακρίβεια των προβλέψεων.

1.8 Βαθιά μάθηση

Η Βαθιά Μάθηση, όντας ένα υποσύνολο της Μηχανικής Μάθησης, που ασχολείται με την ανάπτυξη τεχνικών προσδίδει τις ικανότητες, της μάθησης και της αυτοβελτίωσης σε υπολογιστικά συστήματα [13]. Τα μοντέλα βαθιάς μάθησης μπορούν να αναγνωρίζουν σύνθετα μοτίβα σε εικόνες, κείμενα ήχους και άλλα είδη δεδομένων, επιτυγχάνοντας ακριβείς προβλέψεις.

Ένα ιδιαίτερο χαρακτηριστικό που διαφοροποιεί τη βαθιά μάθηση από την απλή μηχανική μάθηση είναι η δυνατότητα εκπαίδευσης απευθείας από μεγάλο όγκο ανεπεξέργαστων δεδομένων. Χάρη στις σημαντικές βελτιώσεις που έχει υποστεί, η βαθιά μάθηση καθίσταται ιδανική για ένα ευρύ φάσμα εφαρμογών, όπως η υπολογιστική όραση, η επεξεργασία φυσικής γλώσσας και η ενισχυτική μάθηση.

Οι αλγόριθμοι της βαθιάς μάθησης βασίζονται σε τεχνητά νευρωνικά δίκτυα πολλαπλών στρωμάτων (Artificial Neural Networks), γνωστά και ως βαθιά νευρωνικά δίκτυα (Deep Neural Networks), τα οποία μιμούνται, έως ένα βαθμό, τη λειτουργία του ανθρώπινου εγκεφάλου. Υπάρχουν διάφοροι τύποι αρχιτεκτονικών βαθιάς μάθησης, με τα πιο διαδεδομένα νευρωνικά δίκτυα να περιλαμβάνουν Συνελικτικά Νευρωνικά Δίκτυα (Convolutional Neural Networks), Νευρωνικά Δίκτυα Πρόσθιας Τροφοδότησης (Feedforward Neural Network) και τα Επαναλαμβανόμενα Νευρωνικά Δίκτυα (Recurrent Neural Networks).

1.9 Υπολογιστική όραση

Η υπολογιστική όραση ή μηχανική όραση, αποτελεί ένα διεπιστημονικό πεδίο της τεχνητής νοημοσύνης, που εστιάζει στην εξαγωγή πληροφοριών από ψηφιακές εικόνες. Στόχος της είναι η ανάπτυξη συστημάτων ικανών να «βλέπουν» με τρόπο παρόμοιο με την ανθρώπινη όραση [14].

Το πεδίο της μηχανικής όρασης έκανε την εμφάνιση του στις αρχές της δεκαετίας του 1970, ωστόσο, το πρόβλημα παραμένει ακόμα άλυτο.

Η μηχανική όραση στον πραγματικό κόσμο έχει βρει εφαρμογή σε πλήθος τομέων όπως:

- **Επιθεώρηση μηχανών (Machine Inspection):** Ανίχνευση σφαλμάτων ή ελαττωμάτων σε εξαρτήματα.
- **Ιατρική απεικόνιση (Medical Imaging) :** Ανάλυση ιατρικών εικόνων.
- **Αναγνώριση βιομετρικών στοιχείων (Biometrics):** Γρήγορη αυθεντικοποίηση χρηστών
- **Συστήματα παρακολούθησης (Surveillance) :** Όπως της ανίχνευσης πληρότητας σε χώρους στάθμευσης (αντικείμενο μελέτης στην παρούσα εργασία).

Οι εργασίες μηχανικής όραση μπορούν να κατηγοριοποιηθούν σε διάφορες υποκατηγορίες, όπως:

- **Ταξινόμηση Αντικειμένων (Object Detection):** Ταξινόμηση εικόνων σε προκαθορισμένες κατηγορίες.
- **Σημασιολογική Τμηματοποίηση (Segmatic Segmentation):** Διαχωρισμός εικόνων σε ξεχωριστά αντικείμενα ή περιοχές ενδιαφέροντος
- **Ταξινόμηση και Εντοπισμός (Classification & Localization):** Ταυτόχρονη ταξινόμηση και εντοπισμός αντικειμένων σε εικόνα
- **Αναγνώριση Αντικειμένων (Object Detection):** Ανίχνευση και αναγνώριση αντικειμένων.

1.9.1 Αναγνώριση Αντικειμένων (Object Detection)

Η αναγνώριση αντικείμενων είναι μία εργασία της υπολογιστής όρασης, που περιλαμβάνει την αναγνώριση και τον εντοπισμό αντικείμενων σε ένα μέσα όπως εικόνες ή βίντεο [15].

Η αναγνώριση αντικειμένων αποτελεί απαραίτητο εργαλείο σε πλήθος εφαρμογών υπολογιστικής όρασης, όπως:

- **Ανίχνευση και αναγνώριση προσώπων:**

  Χρησιμοποιείται για τον εντοπισμό και την ταυτοποίηση ατόμων σε εικόνες ή βίντεο.

- **Αναγνώριση κειμένου:**

  Επιτρέπει την εξαγωγή και επεξεργασία κειμένου από εικόνες ή βίντεο, με χρήση σε οπτική αναγνώριση χαρακτήρων (OCR), ψηφιακή βιβλιοθήκη και αυτοματοποίηση εγγράφων.

- **Ανίχνευση κινήσεων:**

  Χρησιμοποιείται για τον εντοπισμό και την παρακολούθηση αντικειμένων ή ανθρώπων σε κίνηση, με εφαρμογές στην επιτήρηση και την ανάλυση συμπεριφοράς.

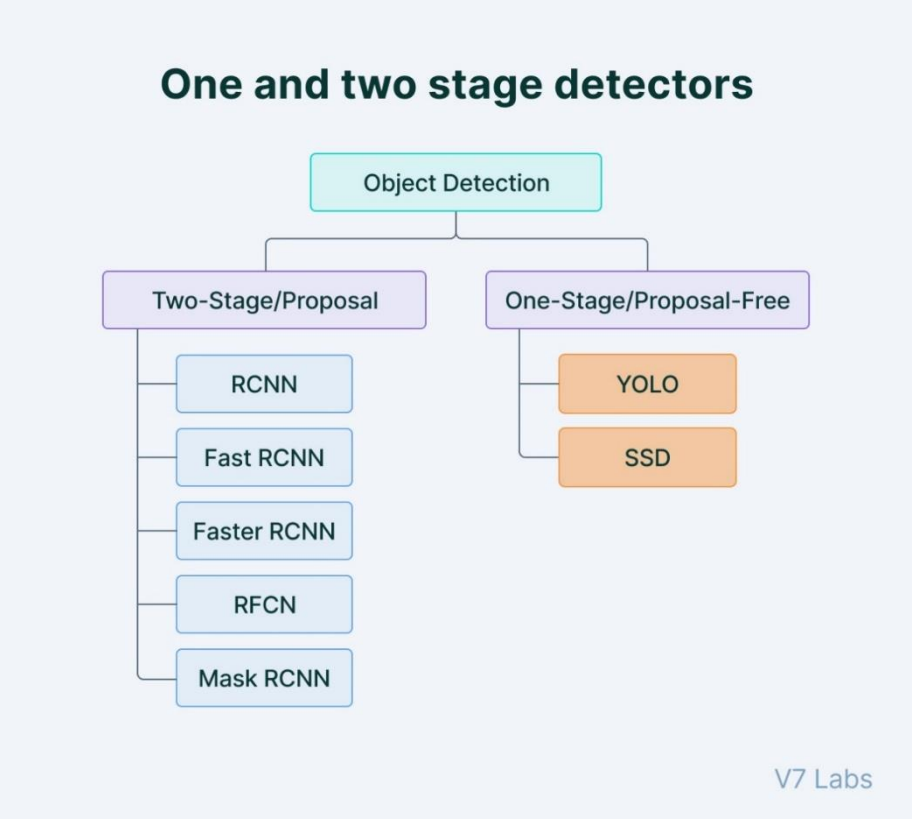
- **Επιτήρηση και καταμέτρηση αντικειμένων:**

  Εντοπίζεται σε εφαρμογές παρακολούθησης της κυκλοφορίας και στην παρακολούθηση αποθέματος.

Βασικό στοιχείο της αναγνώρισης αντικειμένων είναι η κατηγοριοποίηση (Classification) και η οριοθέτηση (Bounding Boxes).

Οι μέθοδοι αναγνώρισης αντικειμένων διακρίνονται σε δύο βασικές κατηγορίες:

- Προσεγγίσεις χωρίς νευρωνικά δίκτυα:
  - **Viola-Jones πλαίσιο αναγνώρισης αντικειμένων**
  - **Μετασχηματισμός χαρακτηριστικών με αμετάβλητη κλίμακα (SIFT)**
  - **Ιστόγραμμα με χαρακτηριστικά προσανατολισμένων κλίσεων (HOG)**
- Προσεγγίσεις με νευρωνικά δίκτυα:
  - **Ανιχνευτές ενός σταδίου (One-Stage Detectors)**
  - **Ανιχνευτές δύο σταδίων (Two-Stage Detectors)**



Εικόνα 12: One & Two stage Detectors [πηγή]

1.9.1.1 Ανιχνευτές ενός σταδίου (One-Stage Detectors)

Οι ανιχνευτές ενός σταδίου, με μία μόνο επεξεργασία της εικόνας, εντοπίζουν με ακρίβεια την κατηγορία στην οποία ανήκει το αντικείμενο, καθώς και τις συντεταγμένες του. Η υπό εξέταση εικόνα τροφοδοτείται σε έναν εξαγωγέα χαρακτηριστικών, ένα συνελικτικό νευρωνικό δίκτυο, το οποίο εξάγει τα χαρακτηριστικά της εικόνας [16].

Στη συνέχεια, αυτά τα χαρακτηριστικά χρησιμοποιούνται για ταξινόμηση και παλινδρόμηση, με σκοπό τον εντοπισμό των πλαισίων οριοθέτησης γύρω από τα αντικείμενα. Σημαντικό πλεονέκτημα των ανιχνευτών ενός σταδίου είναι η υψηλή ταχύτητα ανίχνευσης, καθιστώντας τα ιδανικά για εφαρμογές πραγματικού χρόνου. Ορισμένοι από τους πιο γνωστούς αλγόριθμους που ανήκουν σε αυτή την κατηγορία είναι οι “YOLO” και “SSD”.

Σχόλιο: Η παρούσα εργασία επικεντρώνεται στην μελέτη του αλγορίθμου YOLO

1.9.1.2 Ανιχνευτές δύο σταδίων (Two-Stage Detectors)

Οι ανιχνευτές δύο σταδίων, διαιρούν την διαδικασία ανίχνευσης σε δύο βασικά στάδια. Στο πρώτο στάδιο, χρησιμοποιώντας ένα συνελικτικό νευρωνικό δίκτυο, εξάγουν τα χαρακτηριστικά της εικόνας και προτείνουν μία σειρά από περιοχές ενδιαφέροντος (Regions of Interest), με πιθανά αντικείμενα. Στο δεύτερο στάδιο, πραγματοποιείται η διαδικασία ταξινόμησης.

Βασικό πλεονέκτημα των Two-Stage είναι η υψηλότερη ακρίβεια που διαθέτουν σε σύγκριση με τους One-Stage, θυσιάζοντας όμως την ταχύτητα της ανίχνευσης. Οι πιο δημοφιλείς αλγόριθμοι που ανήκουν στην κατηγορία Two-Stage είναι οι RCNN, Fast R-CNN, Faster R-CNN, Mask RCNN και RFC.

1.9.2 Αλγόριθμος YOLO

Ο αλγόριθμος YOLO αποτελεί έναν από τους πιο δημοφιλείς και ευρέως χρησιμοποιούμενους για εφαρμογές αναγνώρισης αντικειμένων. Αναπτύχθηκε από τους Joseph Redmon και Ali Fahradi το 2015, στα πλαίσια μίας επιστημονικής εργασίας. Ο YOLO είναι ένας αλγόριθμος ενός σταδίου (One-Stage Detector), ο οποίος αξιοποιεί συνελικτικά νευρωνικά δίκτυα για την υλοποίηση προβλέψεων σε πραγματικό χρόνο. Οι προβλέψεις αυτές αφορούν τα πλαίσια οριοθέτησης (Bounding Boxes) και τις κλάσεις των αντικειμένων που εντοπίζονται σε εικόνες [17].

Η χρήση του αλγορίθμου εκτείνεται σε ένα ευρύ φάσμα εφαρμογών, όπως συστήματα παρακολούθησης και αυτόνομα οχήματα. Από την κυκλοφορία της πρώτης έκδοσης του YOLO, έχουν ακολουθήσει πολλαπλές βελτιωμένες εκδόσεις, με την πλέον πρόσφατη κατά τη συγγραφή να είναι η YOLOv10. Παρακάτω παρουσιάζονται συνοπτικά τα χαρακτηριστικά για ορισμένες από τις εκδόσεις του αλγορίθμου [18].

1.9.2.1 YOLOv1

Αποτελεί την πρώτη έκδοση του YOLO, θεωρούμενη ως πρωτοποριακή προσέγγισης για την αναγνώριση αντικειμένων. Σε αυτή την έκδοση η αναγνώριση αντικειμένων επιτυγχάνεται σε μόνο ένα πέρασμα μέσω ενός συνελικτικού νευρωνικού δικτύου, το οποίο πραγματοποιεί προβλέψεις σχετικά με τα πλαίσια οριοθέτησης και τις κλάσεις των αντικειμένων στην εικόνα εισόδου.

Η αρχιτεκτονική του συνελικτικού δικτύου του YOLO αποτελείται από συνελικτικά επίπεδα. Τα πρώτα 20, από αυτά είναι προ-εκπαιδευμένα στο σύνολο δεδομένων “ImageNet”. Ακολουθούν δύο πλήρως συνδεδεμένα επίπεδα. Σε αυτά τα επίπεδα πραγματοποιούνται οι προβλέψεις για την κλάση και τις συντεταγμένες των πλαισίων οριοθέτησης.

Η εικόνα εισόδου διαιρείται σε πλέγμα SxS, το οποίο αποτελείται από κελιά. Σε περίπτωση που ανιχνευθεί κάποιο αντικείμενο σε ένα κελί του πλέγματος, πραγματοποιείται η αναγνώριση του. Κάθε κελί του πλέγματος παράγει προβλέψεις σχετικά με τα πλαίσια οριοθέτησης και τον βαθμό εμπιστοσύνης για την ύπαρξη του αντικειμένου.

Στη συνέχεια, ο αλγόριθμος Non-Maximum Suppression αφαιρεί τα επικαλυπτόμενα πλαίσια, διατηρώντας μόνο το πλαίσιο με την υψηλότερη πιθανότητα. Όσον αφορά την απόδοση, το YOLOv1 πέτυχε mAP 63.4% με ταχύτητα 45 καρέ ανά δευτερόλεπτο.

1.9.2.2 YOLOv2

Η YOLOv2, ή αλλιώς YOLOv9000, κυκλοφόρησε το 2016 ως βελτιωμένη έκδοση του αρχικού αλγορίθμου, με στόχο την αντιμετώπιση των περιορισμών και αδυναμιών του. Μέσω της προσθήκης νέων χαρακτηριστικών και αλλαγών στην αρχιτεκτονική του δικτύου, η απόδοση του αλγορίθμου βελτιώθηκε σημαντικά, με δυνατότητα αναγνώρισης 9000 κατηγοριών.

Αρχικά, η YOLOv2 τροποποίησε την αρχιτεκτονική του δικτύου υιοθετώντας το Darknet-19 ως κορμός (Backbone) συνελικτικού δικτύου. Η προσθήκη του επιπέδου Κανονικοποίησης παρτίδας (Batch Normalization) αύξησε το mAP κατά 2%, μειώνοντας την υπέρ-εκπαίδευση (Overfitting) και βελτιώνοντας τη συνολική απόδοση του δικτύου.

Επιπλέον, ενσωματώθηκε ένα νέο συνελικτικό δίκτυο με σταθερά πλαίσια (Anchor Boxes), βελτιώνοντας και την Ανάκληση (Recall). Η συνάρτηση απωλειών αντικαταστάθηκε με μία νέα που βασίζεται στο τετραγωνικό σφάλμα (Means Square Roots) μεταξύ των προβλέψεων και των πραγματικών κλάσεων και τα πλαίσια οριοθέτησης. Τέλος, η ανάλυση των εικόνων εισόδου για την εκπαίδευση αυξήθηκε από τα 222x222 στα 448x448 pixels.

1.9.2.3 YOLOv3

Η Τρίτη έκδοση του αλγορίθμου, YOLOv3, κυκλοφόρησε το 2018, φέρνοντας νέες βελτιώσεις με στόχο την ακόμα μεγαλύτερη ακρίβεια. Ο κορμός του δικτύου, σε σύγκριση με την YOLOv2, αντικαταστάθηκε από το Darknet-53, μία νέα παραλλαγή συνελικτικού νευρωνικού δικτύου με 53 επίπεδα συνέλιξης. Το Darknet-53 βασίζεται στην αρχιτεκτονική ResNet, η οποία έχει σχεδιαστεί ειδικά για εφαρμογές ανίχνευσης αντικειμένων.

Σε αντίθεση με την προηγούμενη έκδοση, τα πλαίσια αναφοράς (Anchor Boxes) πλεόν μπορούν να κλιμακώνονται (Scaling) και να αλλάζουν τον λόγο διαστάσεων (Aspect Ratio). Αυτό επιτρέπει στα πλαίσια αναφοράς να ταιριάζουν καλύτερα με τα αντικείμενα που ανιχνεύονται στην εικόνα, βελτιώνοντας την ακρίβεια.

1.9.2.4 YOLOv4

Η Τέταρτη έκδοση του YOLO, κυκλοφόρησε το 2020 από τους Alexey Bochkovskiy, Chien-Yao Wang και Hong-Yuan Mark Liao. Αποτελεί μία σημαντικά βελτιωμένη έκδοση σε σχέση με τον προκάτοχο της, YOLOv3, φέρνοντας αύξηση mAP κατά 10% και στα FPS κατά 12%.

Η πιο σημαντική αλλαγή στην τέταρτη έκδοση εντοπίζεται στην αρχιτεκτονική του δικτύου. Πλέον υιοθετείται η νέα αρχιτεκτονική συνελικτικού δικτύου "Cross Stage Partial Network" (CSPNet), μία παραλλαγή του "ResNet", η οποία αποτελείται από 54 συνελικτικά επίπεδα. Η δημιουργία των σταθερών πλαισίων πραγματοποιείται πλέον μέσω του αλγόριθμου ομαδοποίησης K-means.

Επιπλέον, η συνάρτηση απωλειών τροποποιήθηκε με την προσθήκη του όρου απώλειας "GHM", βελτιώνοντας την απόδοση του εκπαιδευμένου μοντέλου σε μη ισορροπημένα σύνολα δεδομένων. Τέλος, η αξιοποίηση τεχνικών όπως οι "Bag of Freebies" και "Bag of Specials" καθιστά εφικτή την εκπαίδευση του νευρωνικού δικτύου με χρήση συμβατικών καρτών γραφικών.

1.9.2.5 YOLOv5

Η Πέμπτη έκδοση του YOLO, κυκλοφόρησε τον Ιούνιο του 2023, λίγους μήνες μετά την κυκλοφορία της τέταρτης έκδοσης, από τον Glenn Jocher. Αποτελεί μία βελτιωμένη έκδοση βασισμένη στο YOLOv3, με κύριο χαρακτηριστικό την υιοθέτηση του "PyTorch" αντί για το Darknet ως βασικό πλαίσιο (Framework).

Η αρχιτεκτονική που βασίζεται η έκδοση YOLOv5 είναι η "EfficientDet". Μία βελτιωμένη παραλλαγή της αρχιτεκτονικής EfficientNet, που προσφέρει υψηλότερη ακρίβεια σε ένα ευρύτερο φάσμα κατηγοριών αντικειμένων. Η εκπαίδευση του βασικού μοντέλου πραγματοποιήθηκε με χρήση του συνόλου δεδομένων D5, το οποίο περιλαμβάνει 600 κατηγορίες αντικειμένων, σε αντίθεση με το σύνολο δεδομένων "PASCAL VOC" της YOLOv1, που περιείχε μόνο 20 κατηγορίες. Η "EfficientDet" υιοθετεί επίσης ένα νέο επίπεδο συγκέντρωσης, τη χωρική συγκέντρωση πυραμίδων (Spatial Pyramid Pooling), για την ανίχνευσης αντικειμένων μικρού μεγέθους. Τέλος η συνάρτηση απώλειας "CIoU Loss", μία παραλλαγή της "IoU Loss", εφαρμόζεται για βελτιωμένη έκδοση.

1.9.2.6 YOLOv8

Η Όγδοη έκδοση του YOLO, κυκλοφορεί τον Ιανουάριο του 2023 από τους δημιουργούς της πέμπτης έκδοσης. Το YOLOv8, γραμμένος σε PyTorch, βασίζεται στο πλαίσιο του YOLOv5, προσφέροντας όμως σημαντικές βελτιώσεις και αλλαγές. Διατίθενται πέντε παραλλαγές, κατάλληλες για εφαρμογές ανίχνευσης αντικειμένων (Object Detection) και κατάτμησης (Segmentation).

Λόγω των αλλαγών που υιοθετήθηκαν στην αρχιτεκτονική της όγδοης έκδοσης, αναπτύχθηκε ένα νέο μοντέλο με βελτιωμένη ακρίβεια και μειωμένο μέγεθος. Στον κορμό (Backbone) του δικτύου, πλέον χρησιμοποιείται το "C2f" αντί του "C3", ενώ το μέγεθος του συνελικτικού πυρήνα αυξήθηκε από 1x1 σε 3x3.

Η πρώτη σημαντική αλλαγή αφορά την κατάργηση των πλαισίων αναφοράς (Anchor Boxes), τα οποία αποτελούσαν βασικό στοιχείο των προηγουμένων εκδόσεων. Η νέα προσέγγιση, η οποία λειτουργεί χωρίς πλαίσια αναφοράς, έχει ως αποτέλεσμα τη μείωση του αριθμού των πλαισίων πρόβλεψης και την επιτάχυνση της εκτέλεσης του αλγορίθμου "Non-Maximum Suppression".

Στο σύνολο δεδομένων "MS COCO", η μέση ακρίβεια (mAP) του μοντέλου παρουσίασε αυξήσεις κατά 3.2% σε σύγκριση με το YOLOv5. Η βελτίωση αυτή οφείλεται κυρίως στο ανανεωμένο τμήμα λαιμού (Neck Segment), το οποίο συνδυάζει αποτελεσματικά χαρακτηριστικά από τα διάφορα επίπεδα του κορμού (Backbone). Η νέα αυτή προσέγγιση επιτρέπει την ανίχνευση αντικειμένων με διαφορετικές κλίμακες με υψηλότερη ακρίβεια.

1.9.2.7 YOLOv9

Η Ένατη έκδοση του αλγορίθμου YOLO, η οποία κυκλοφόρησε τον Φεβρουάριο του 2024 από τους Chien-Yao Wang, I-Hau Yeh και Hong-Yuan Mark Liao. Διατίθενται τέσσερις παραλλαγές του μοντέλου: YOLOv9-S, YOLOv9-M, YOLOv9-C και YOLOv9-E.

Χάρη στην ενσωμάτωση της τεχνολογίας "Προγραμματιζόμενες πληροφορίες βαθμίδας" (PGI) και στη χρήση ενός νέου δικτύου αρχιτεκτονικής, του Gelan, παρουσιάζονται σημαντικές βελτιώσεις στην απόδοση του αλγορίθμου.

Το PGI βελτιώνει την ικανότητα εκμάθησης και την ακρίβεια του μοντέλου, αντιμετωπίζοντας παράλληλα το πρόβλημα της απώλειας σημαντικών πληροφοριών. Η αρχιτεκτονική Gelan, όντας βελτιωμένη έκδοση της έβδομης έκδοσης, έχει σχεδιαστεί για την βελτιστοποίηση παραμέτρων, την αύξηση της ακρίβειας και την επιτάχυνση της διαδικασίας Inference.

Σε σύγκριση με το YOLOv8, στο σύνολο δεδομένων "MS COCO", παρατηρείται αύξηση της μέσης ακρίβειας (mAP) κατά 0.6%.

1.9.2.8 YOLOv10

Η δέκατη έκδοση του αλγορίθμου YOLO, που κυκλοφόρησε τον Μάϊο του 2024 από τους Ao Wang, Hui Chen, Lihao Liu, Kai Chen, Zijia Lin, Jungong Han και Guiguang Ding, αποτελεί την πιο πρόσφατη έως και την ημερομηνία συγγραφής της παρούσας εργασίας. Διατίθενται έξι παραλλαγές του μοντέλου: YOLOv10-N, YOLOv10-S, YOLOv10-M, YOLOv10-B, YOLOv10-L και YOLOv10-X.

Συγκριτικά με προηγούμενες εκδόσεις του YOLO, η νέα έκδοση παρουσιάζει σημαντικές βελτιώσεις στην ακρίβεια και την ταχύτητα ανίχνευσης αντικειμένων, ενώ παράλληλα μειώνει το μέγεθος του μοντέλου. Αυτό επιτυγχάνεται μέσω αλλαγών και τροποποιήσεων στην αρχιτεκτονική του δικτύου και στο στάδιο της μετεπεξεργασίας.

Πιο συγκεκριμένα, στον κορμό (Backbone) του δικτύου χρησιμοποιείται μία βελτιωμένη έκδοση του CSPNet, η οποία αυξάνει την αποδοτικότητα του μοντέλου. Επιπλέον, στο τμήμα λαιμού (Neck Segment) ενσωματώνονται επίπεδα PAB, βελτιώνοντας την αναγνώριση αντικειμένων. Μία ακόμα σημαντική αλλαγή είναι η εξάλειψη του αλγορίθμου Non-Maximum Suppression (NMS) κατά τη μετεπεξεργασία, με αποτέλεσμα την ενίσχυση της ακρίβειας και της ταχύτητας του Inference.

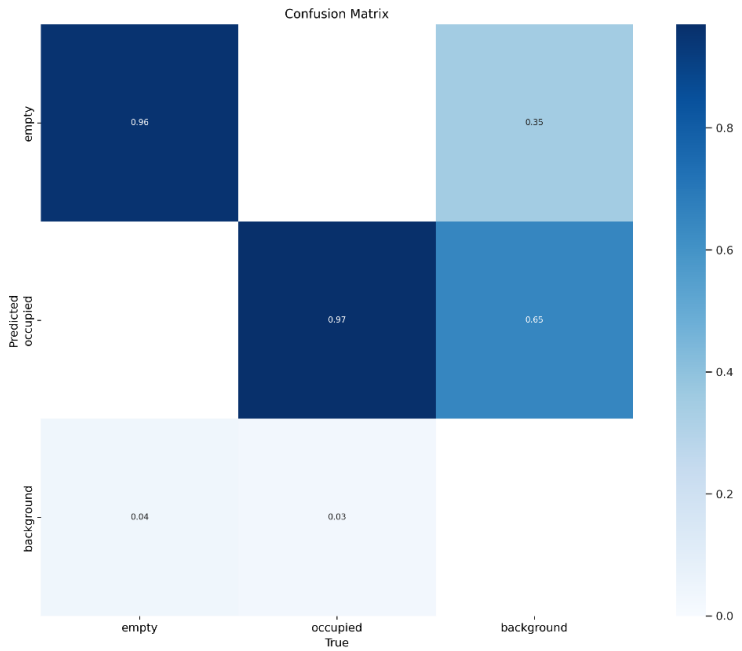
Στην εκπαίδευσης υιοθετείται μία προσέγγισης που βασίζεται σε αναθέσεις διπλής ετικέτας χωρίς την χρήση NMS. Η μέθοδος αυτή διατηρεί την υπολογιστική αποδοτικότητα, ενώ παράλληλα αυξάνει την ακρίβεια και την ταχύτητα.

1.9.3 Βασικές έννοιες αξιολόγησης

Στο Κεφάλαιο 4, θα γίνει αναφορά στην εκπαίδευση μοντέλων ανίχνευσης αντικειμένων βασισμένων στις εκδόσεις του YOLOv5, YOLOv8 και YOLOv10. Κατά τη διαδικασία εκπαίδευσης, για την αξιολόγηση της απόδοσης των μοντέλων θα αναφερθούν διάφοροι βασικοί όροι, όπως:

- **Πίνακας Σύγχυσης (Confusion Matrix)**

  - **Σωστά Θετικά (True Positive - TP) :** Κατηγοριοποιήθηκε σωστά δείγμα που ανήκει στην θετική κλάση
  - **Λάθος Θετικά (False Positive – FP) :** Κατηγοριοποιήθηκε λανθασμένα δείγμα που ανήκει στην αρνητική κλάση ως θετικό
  - **Σωστά Αρνητικά (True Negative – TN) :** Κατηγοριοποιήθηκε σωστά δείγμα που ανήκει στην αρνητική κλάση
  - **Λάθος Αρνητικά (False Negative – FN) :** Κατηγοριοποιήθηκε λανθασμένα δείγμα που ανήκει στην θετική κλάση ως αρνητικό



Εικόνα 13: Ενδεικτικό παράδειγμα Πίνακα Σύγχυσης [πηγή]

- **Ακρίβεια (Precision)**

Η ακρίβεια μετρά το ποσοστό των σωστών προβλέψεων

Τύπος Υπολογισμού: Η ακρίβεια (P) είναι ο αριθμός των ορθά θετικών προβλέψεων (TP) προς το σύνολο των ορθά θετικών (TP) και των λανθασμένων θετικών (FP).

$$P = \frac{T_P}{T_P + F_P}$$

- **Ανάκληση (Recall)**

Η ανάκληση μετρά το ποσοστό των πραγματικών αντικειμένων που εντοπίστηκαν σωστά.

Τύπος Υπολογισμού: Η ανάκληση (R) είναι ο αριθμός των ορθά θετικών προβλέψεων (TP) προς το σύνολο των ορθά θετικών (TP) και των λανθασμένων αρνητικών (FN).

$$R = \frac{T_P}{T_P + F_N}$$

- **Mean Average Precision**

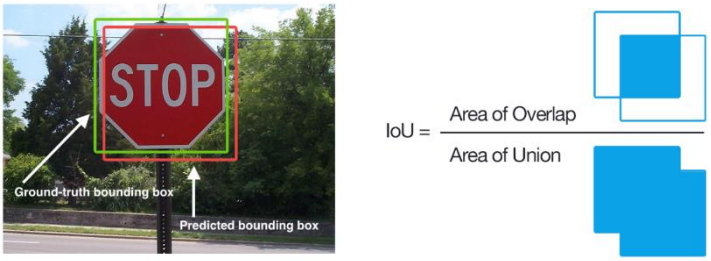
Η μέση αντιπροσωπευτική ακρίβεια (mAP) θεωρείται ως το πιο αξιόπιστο μέτρο αξιολόγησης, παρέχοντας μία ολοκληρωμένη εικόνα της απόδοσης του μοντέλου. Υπολογίζεται σε διάφορες οριακές τιμές του "IoU", οι οποίες κυμαίνονται από 0.5 έως 0.95.

Τύπος υπολογισμού: Ο υπολογισμός του mAP προαπαιτεί τον υπολογισμό της μέσης ακρίβειας (AP) για όλες τις κλάσεις του συνόλου δεδομένων.

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i$$

- **Διασταύρωση επί της ένωσης (Intersection over Union)**

Το "IoU" υπολογίζει το βαθμό επικάλυψης μεταξύ δύο οριοθετημένων πλαισίων εντοπίζονται σε μία εικόνα. Όσο υψηλότερη είναι τιμή του IoU, τόσο πιο κοντά είναι οι συντεταγμένες του πλαίσιού της πρόβλεψης με τις συντεταγμένες του πραγματικού πλαισίου.



Εικόνα 14: Intersection Over Union [πηγή]

Κεφάλαιο 2: Βιβλιογραφική Αναφορά

Πριν την ανάλυση, το σχεδιασμό και την εκπαίδευσης του μοντέλου, είναι απαραίτητη η μελέτη σχετικών άρθρων και δημοσιεύσεων. Για το σκοπό αυτό, αξιοποιήθηκε η μηχανή αναζήτησης Google Schola, η οποία παρέχει πρόσβαση σε ακαδημαϊκά άρθρα, έρευνες από ποικίλους επιστημονικούς κλάδους [19].

Αρχικά, χρησιμοποιήθηκε η φράση κλειδί "parking spot detection", για την εύρεση αποτελεσμάτων που περιλαμβάνουν και διαφορετικές προσεγγίσεις από αυτή της εργασίας. Η μηχανή αναζήτησης εντόπισε 65.500 αποτελέσματα. Στη συνέχεια, χρησιμοποιήθηκε η φράση κλειδί "automatic parking spot detection" και τα αποτελέσματα μειώθηκαν σε 39.400

Λόγω του τεράστιου όγκου διαθέσιμων άρθρων, η επιλογή και ο σχολιασμός ακόμα και με τη χρήση φίλτρων, καθίσταται δύσκολη έως αδύνατη. Ως εκ τούτου κρίνεται αναγκαία η υιοθέτηση διαφορετικής προσέγγισης.

Η νέα προσέγγιση εστιάζει αρχικά στον χαρακτηρισμό των διαφορετικών τύπων αυτόματων συστημάτων αναγνώρισης θέσεων στάθμευσης. Σύμφωνα με το [20], τα συστήματα κατηγοριοποιούνται σε τρεις κύριες:

- **Συστήματα Μετρητών (Counter-Based Systems):** Τα συστήματα μετρητών βασίζονται στη χρήση αισθητήρων στις εισόδους και εξόδους των χώρων στάθμευσης. Η διαθεσιμότητα των χώρων στάθμευσης υπολογίζεται με την αφαίρεση του αριθμού των οχημάτων που εξέρχονται από τον αριθμό των οχημάτων που εισέρχονται. Είναι ιδανικά για χώρους στάθμευσης με ελεγχόμενές εισόδους και εξόδους. Ωστόσο, ένα βασικό μειονέκτημα είναι ότι δεν παρέχονται πληροφορίες για την διαθεσιμότητα κάθε μεμονωμένης θέσης στάθμευσης.

- **Συστήματα Αισθητήρων (Sensor-Based Systems)**: Τα συστήματα αισθητήρων αξιοποιούν αισθητήρες τοποθετημένους είτε στα δάπεδα είτε στις οροφές των χώρων στάθμευσης για τον εντοπισμό οχημάτων. Κάθε θέση στάθμευσης διακατέχεται με αισθητήρες όπως υπέρυθρων (Infrared), υπερήχων (Ultrasonic) ή μαγνητικού πεδίου (Magnetic).

- **Συστήματα Όρασης (Vision-Based Systems):** Τα συστήματα όρασης αξιοποιούν οπτικούς αισθητήρες, όπως κάμερες, σε συνδυασμό με μοντέλα βαθιάς μάθησης, για την αυτόματη ανίχνευση ελευθέρων και κατειλημμένων θέσεων στάθμευσης.

Επιστρέφοντας λοιπόν στο πλαίσιο της εργασίας, πραγματοποιήθηκε αναζήτηση στο Scholar για συστήματα Αισθητήρων και Όρασης σε χώρους στάθμευσης. Τα συστήματα μετρητών παραλήφθηκαν καθώς η χρήση τους έχει περιοριστεί και δεν εστιάζουν στο αντικείμενο της εργασίας όσο τα υπόλοιπα συστήματα.

Η [21] προτείνει μία λύση διαδικτύου των πραγμάτων που βασίζεται σε σύστημα αισθητήρων. Το προτεινόμενο σύστημα περιλαμβάνει ελεγκτές, αισθητήρες, έναν διακομιστή (server) και το νέφος (Cloud). Δύο αισθητήρες υπερήχων, τοποθετημένοι στις οροφές, αποστέλλουν δεδομένα στον διακομιστή για επεξεργασία και αποθήκευση στο νέφος. Η μετάδοση δεδομένων στις δοκιμές λειτούργησε άψογα. Η ανάκτηση δεδομένων από τους αισθητήρες και η παράδοση τους σε smartphones, πραγματοποιείται σε 42 δευτερόλεπτα.

Η [22] προτείνει μία ακόμα IoT πραγματικού χρόνου λύση, για έξυπνες πανεπιστημιουπόλεις. Η υλοποίηση πραγματοποιείται με τη χρήση ενός Raspberry Pi ως IOT-Gateway, αισθητήρων υπέρηχων για κάθε θέση στάθμευσης, GPS module και καμερών που καλύπτουν ένα σύνολο θέσεων. Τα δεδομένα σχετικά με τη διαθεσιμότητα, λαμβάνονται μέσω μίας εφαρμογής Blink, η οποία διαθέτει ένα γραφικό περιβάλλον χρήστη (GUI). Το γραφικό περιβάλλον παρέχει ένα dashboard με πληροφορίες σχετικά με την πληρότητα του χώρου, καθώς επίσης και οδηγίες για την εύρεση διαθέσιμης θέσης.

Η [23] προτείνει την χρήση του αλγορίθμου βαθιάς μάθησης YOLO, για την αναγνώριση και τον διαχωρισμό των ελευθέρων και κατειλημμένων θέσεων στάθμευσης. Για χρήση με το μοντέλο πραγματοποιήθηκε εκπαίδευσης μοντέλων με διαφορετικές εποχές (Epochs), σε ένα custom σύνολο δεδομένων αποτελούμενο από 135 εικόνες. Η ακρίβεια ανήλθες στο 6.9% και 41.3% για 50 και 300 εποχές εκπαίδευσης αντίστοιχα. Σύμφωνα με τους συγγραφείς, τα αποτελέσματα δεν ήταν ικανοποιητικά. Προτείνουν ωστόσο, περαιτέρω έρευνα με διαφορετικές αρχιτεκτονικές και παραλλαγές συνελικτικών νευρωνικών δικτύων, καθώς και με ένα πιο εμπλουτισμένο σύνολο δεδομένων.

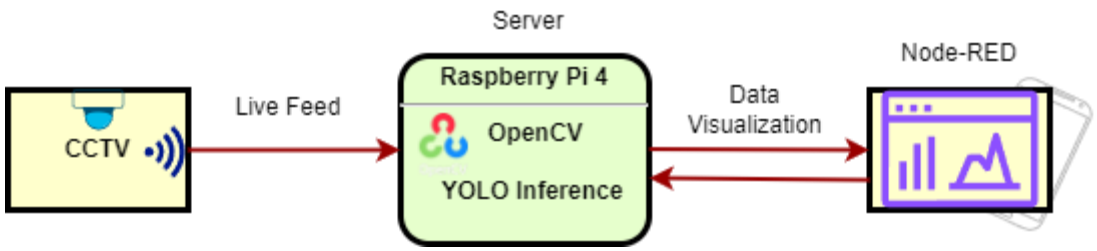
Κεφάλαιο 3: Υλοποίηση Συστήματος

Όπως προαναφέρθηκε, στόχος είναι η υλοποίηση ενός συστήματος αναγνώρισης θέσεων στάθμευσης σε πραγματικό χρόνο με τη χρήση της μηχανικής όρασης. Πριν την υλοποίηση του συστήματος, είναι απαραίτητο να καθοριστούν οι βασικές του απαιτήσεις.

3.1 Περιγραφή Συστήματος

- **Βασικές Απαιτήσεις:**

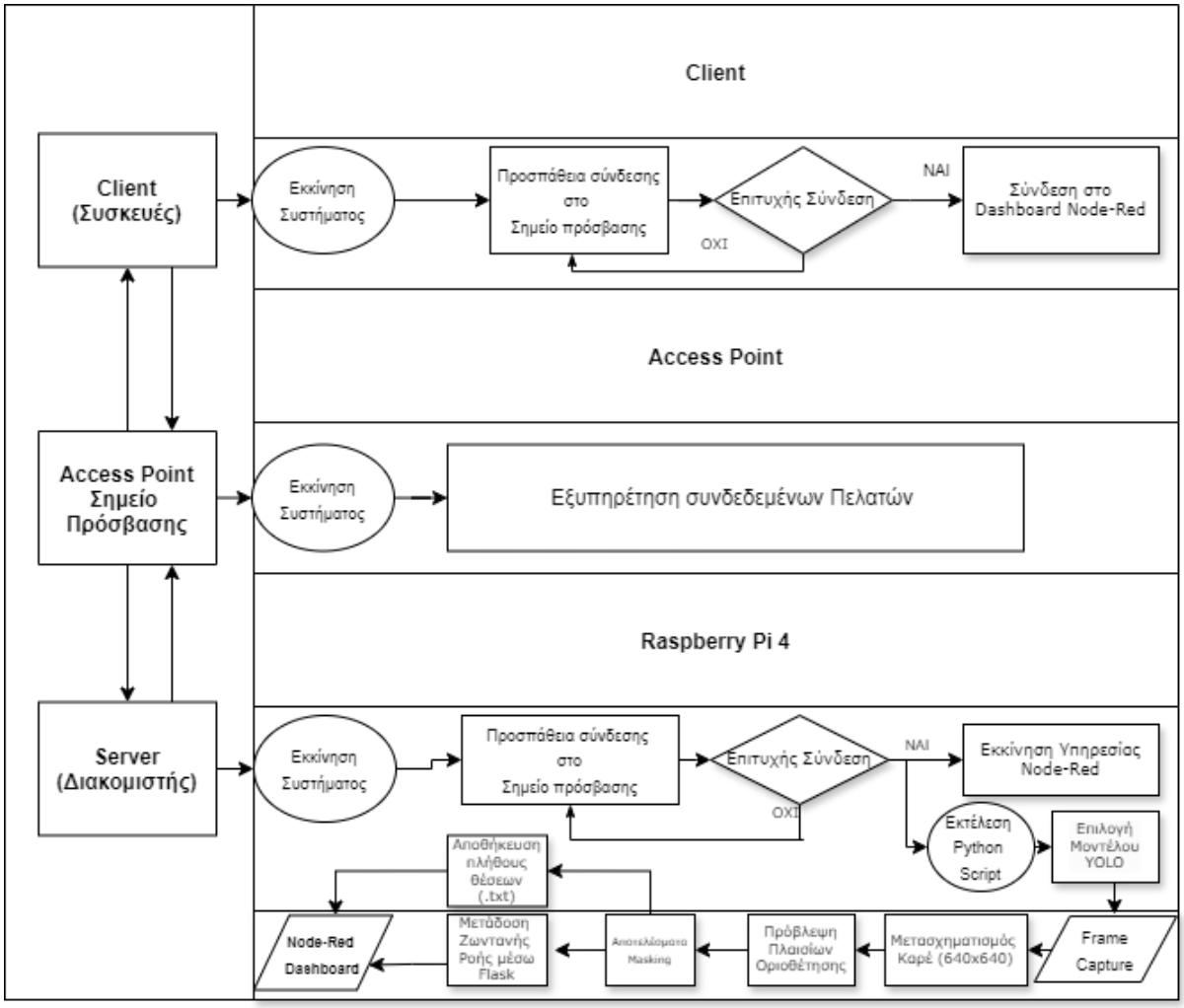
1. Ζωντανή ροή: Το σύστημα θα πρέπει να μεταδίδει ή να λαμβάνει ζωντανή ροή.
2. Αναγνώριση θέσεων στάθμευσης: Το σύστημα θα πρέπει να μπορεί να εντοπίζει με ακρίβεια τις διαθέσιμες και κατειλημμένες θέσεις στάθμευσης.
3. Οπτικοποίηση δεδομένων: Το σύστημα πρέπει να παρουσιάζει τα δεδομένα σχετικά με την κατάσταση των θέσεων στάθμευσης με ευκρινή και κατανοητό τρόπο.



Εικόνα 15: Αρχιτεκτονική Συστήματος

- **Περιγραφή συστήματος υλοποίησης**

Στην αναγνώριση αντικειμένων απαραίτητο στοιχείο είναι η χρήση ένας οπτικού αισθητήρα, όπως μία κάμερα, που αλληλοεπιδρά με το περιβάλλον και καταγράφει τη ζωντανή ροή. Η επεξεργασία της ζωντανής ροής είναι απαραίτητη για την οπτικοποίηση των δεδομένων την πραγματοποίηση προβλέψεων. Επομένως απαιτείται η χρήση υπολογιστικού συστήματος με την ελάχιστη απαραίτητη υπολογιστική ισχύς. Επιπλέον, το επιλεγμένο υπολογιστικό σύστημα οφείλει να υποστηρίζεται από τις απαραίτητες βιβλιοθήκες και σχετικό λογισμικό.

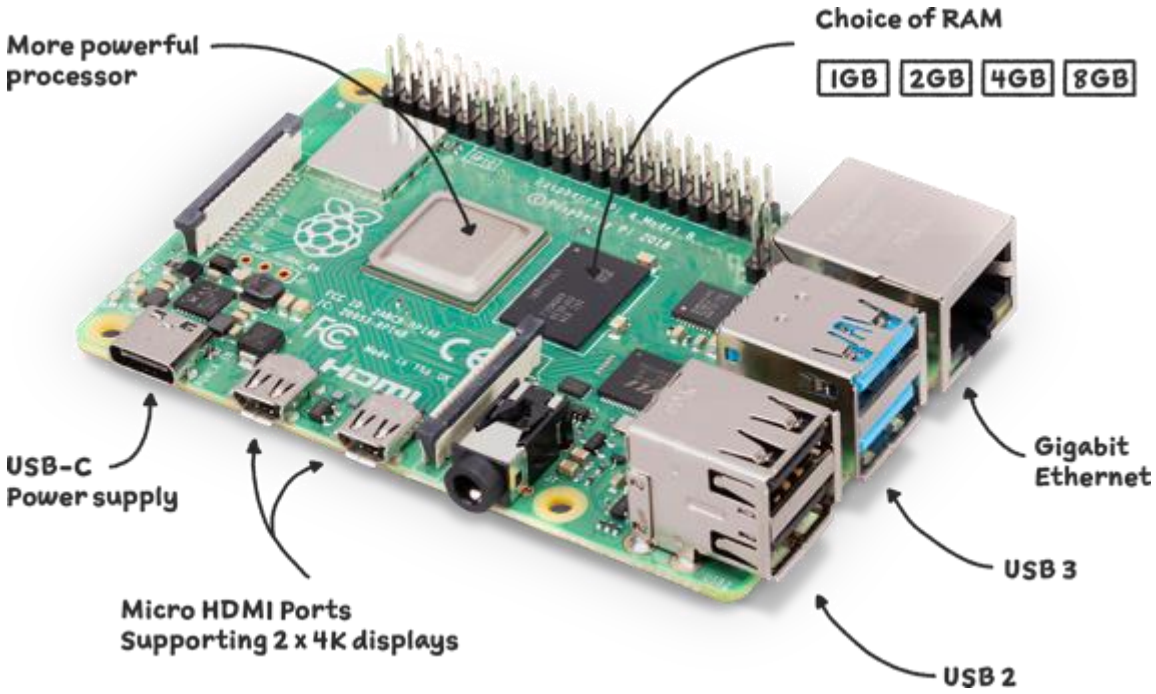


Εικόνα 16: Αναλυτικό Διάγραμμα Ροής της υλοποίησης

3.2 Υλικό – Hardware

3.2.1 Raspberry Pi

Το Raspberry Pi αποτελεί έναν υπολογιστή μονής πλακέτας (SBC) ή ενσωματωμένο σύστημα, με βασικά χαρακτηριστικά τη χαμηλή κατανάλωση ενέργειας, το μικρό μέγεθος και τις υψηλές του αποδόσεις. Τα Raspberry Pi's κυκλοφόρησαν για πρώτη φορά το 2012 με δημιουργό το Raspberry Pi Foundation και χώρα παραγωγής το Ηνωμένο Βασίλειο [24]. Στόχος του ιδρύματος είναι η προώθηση της πληροφορικής και των υπολογιστικών επιστημών στο ευρύ κοινό.



Εικόνα 17: Raspberry Pi 4 Model B [πηγή]

Σχόλιο: Ο μικροϋπολογιστής που επιλέχθηκε είναι το Raspberry Pi 4, καλύπτοντας επαρκώς τις υπολογιστικές απαιτήσεις.

3.2.2 Τεχνικά Χαρακτηριστικά του Raspberry Pi 4 [25]

3.2.2.1 Σύστημα σε ένα Τσιπ (System on a Chip)

Η πλακέτα διαθέτει ένα SoC της Broadcom BCM2711, το οποίο ενσωματώνει Κεντρική Μονάδα Επεξεργασίας (CPU) και Μονάδα Επεξεργασίας Γραφικών (GPU). Ο τετραπύρινος επεξεργαστής Cortex-A72, 64 bit, χρονισμένος στα 1.5 GHz, προσφέρει έως και τριπλάσια ταχύτητα σε σχέση με προηγούμενα μοντέλα καθιστώντας τον ιδανικό για απαιτητικές εργασίες, όπως εφαρμογές Διαδικτύου των Πραγμάτων (IoT) και εφαρμογές μηχανικής μάθησης.

3.2.2.2 Αποθηκευτικός Χώρος

Ένα μειονέκτημα των Raspberry Pi, είναι η έλλειψη ενσωματωμένου αποθηκευτικού χώρου, καθιστώντας απαραίτητη την χρήση μίας κάρτας μνήμης micro-SD. Συγκριτικά με άλλα μέσα αποθήκευσης, οι micro-SD παρουσιάζουν μειωμένη και αρκετά μειωμένη διάρκεια ζωής, ενώ η ταχύτητα εγγραφής και ανάγνωσης δεδομένων ενδέχεται να επηρεάσει αρνητικά την απόδοση του Raspberry Pi.

3.2.2.3 Μνήμη RAM

Οι διαθέσιμες εκδόσεις μνήμης RAM είναι, 1GB, 2GB, 4GB και 8GB LPDDR4. Η μνήμη RAM LPDDR4 είναι έως και 8 φορές ποιο γρήγορη από παλαιότερα μοντέλα. Για την παρούσα εργασία επιλέχθηκε το μοντέλο με 4GB μνήμης RAM.

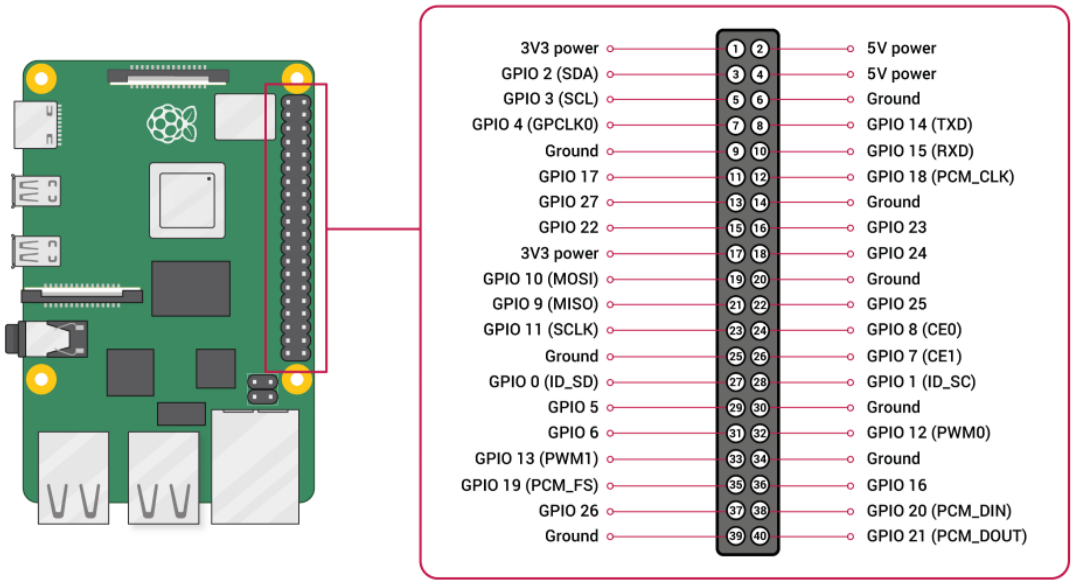
3.2.2.4 Συνδεσιμότητα

Το Raspberry Pi 4, διαθέτει Wi-Fi 2.4 & 5GHz, Bluetooth και Gigabit Ethernet με ταχύτητες έως 1000 Mbps. Υποστηρίζει έως και δύο οθόνες με μέγιστη ανάλυση 4K@60Hz, ενώ διαθέτει και θύρες USB για σύνδεση περιφερειακών.

3.2.2.5 Ακροδέκτες γενικής χρήσης εισόδου-εξόδου (GPIO):

Τα Raspberry Pi διαθέτουν 40 ακροδέκτες γενικής χρήσης εισόδου-εξόδου (General-Purpose Input-Output) [26]. Όλοι οι ακροδέκτες, εκτός από της τροφοδοσία μπορούν να προγραμματιστούν ως είσοδοι ή έξοδοι. Οι ακροδέκτες τροφοδοσίας παρέχουν τάσεις 5V και 3.3V καθώς και γειώσεις.

Στους ακροδέκτες εισόδου μπορούν να συνδεθούν συσκευές, όπως αισθητήρες και κουμπιά, οι οποίες παράγουν ψηφιακά σήματα που κυμαίνονται στα 1.8V-3.3V. Οι ακροδέκτες εξόδου, όταν ενεργοποιηθούν παράγουν σήμα με τάση 3.3V.



Εικόνα 18: Ακροδέκτες Εισόδου-Εξόδου Γενικής Χρήσης του Raspberry Pi 4 [πηγή]

Ορισμένοι ακροδέκτες μπορούν να χρησιμοποιηθούν και για ειδικές λειτουργίες. Στην παραπάνω εικόνα, βλέπουμε τις λειτουργίες των ακροδεκτών.

Υποστηριζόμενες λειτουργίες;

- Έλεγχος:
    - Διαμόρφωση πλάτους παλμού (PWM): Χρησιμοποιώντας τους ακροδέκτες, μπορεί να ελεγχθεί η ταχύτητα ενός ανεμιστήρα, η φωτεινότητα LED και άλλα.

- Επικοινωνία του Raspberry Pi με εξωτερικές συσκευές:
    - SPI: Διατίθενται δύο κανάλια, με το κάθε κανάλι να καταλαμβάνει 5 ακροδέκτες στο σύνολο. (MOSI, MISO, SCLK, CE0, CE1)
    - UART: Σειριακή επικοινωνία, μέσω σειριακής κονσόλας. Απαιτούνται 2 ακροδέκτες μετάδοσης σήματος και ακροδέκτης γείωσης (RX, TX, GND)
    - I2C: Διατίθενται δύο κανάλια, με το κάθε κανάλι να καταλαμβάνει 2 ακροδέκτες. (SDA, SCL)

Τέλος ο προγραμματισμός των ακροδεκτών επιτυγχάνεται με τη κάποιας από τις υποστηριζόμενες γλώσσες προγραμματισμού (π.χ Python) και των διαθέσιμων βιβλιοθηκών της γλώσσας.

3.2.2.6 Τροφοδοσία

Για την τροφοδοσία του Raspberry Pi απαιτείται η χρήση τροφοδοτικού με μέγιστη ελάχιστη ισχύς εξόδου στα 13.5 Watts και βύσμα USB Type-C,. Επιλέχθηκε το επίσημο τροφοδοτικό Raspberry Pi Power Supply.

Εικόνα 19: Raspberry Pi Official Power Supply [πηγή]

3.2.2 Pi Camera Module 3

Στην παρούσα υποενότητα, θα γίνει αναφορά στην κάμερα του συστήματος, η οποία επιτρέπει στο Raspberry Pi να αλληλοεπιδρά με τον πραγματικό κόσμο. Στην υλοποίηση χρησιμοποιήθηκε, η κάμερα “Pi Camera Module 3”, μία νέα έκδοση της σειράς Pi Camera, από το Raspberry Pi Foundation [27].

Εικόνα 20: Raspberry Pi Camera Module 3 [πηγή]

3.2.2.1 Τεχνικά Χαρακτηριστικά του Pi Camera Module 3

Αισθητήρας: Το Module 3 χρησιμοποιεί τον αισθητήρα IMX708 της Sony, οποίος προσφέρει ανάλυση 4608 x 2592 pixels και διαγώνιο 7.4mm.

Λειτουργίες: Λήψη Φωτογραφιών και Λήψη Βίντεο (1080p50Hz, 720p100Hz, 480p120Hz)

Συμβατότητα:

- **Υλικό:** Το Module 3 είναι συμβατό με όλα τα μοντέλα Raspberry Pi. Η σύνδεση της κάμερας με το Raspberry Pi, πραγματοποιείται μέσω μίας καλωδιοταινίας FFC 15x1mm.
- **Λογισμικό:** Η διεπαφή μεταξύ του μικροϋπολογιστή και της κάμερας, επιτυγχάνεται μέσω της προεγκατεστημένης βιβλιοθήκης libcamera.

Λοιπά χαρακτηριστικά: Ενσωματωμένο φίλτρο αποκοπής IR

3.3 Λογισμικό - Software

Για την υλοποίηση του συστήματος, χρησιμοποιήθηκαν τα ακόλουθα λογισμικά. Θα παρουσιαστούν και θα αναλυθούν κάθε ένα από αυτά

3.3.1 Λειτουργικό Σύστημα Raspberry Pi OS

Από τους δημιουργούς των Raspberry Pi προσφέρεται δωρεάν ένα λειτουργικό σύστημα με την ονομασία "Raspberry Pi OS", παλαιότερα γνωστό και ως Raspbian OS [28]. Το Raspberry Pi OS βασίζεται στη διανομή ανοιχτού κώδικα Debian GNU/Linux, με δημιουργό τον Ian Mudrock και ημερομηνία κυκλοφορίας, στις 16 Αυγούστου του 1993 [29].

Πρόκειται για ένα ειδικά διαμορφωμένο λειτουργικό σύστημα που περιλαμβάνει πάνω από 35 χιλιάδες προμεταγλωτισμένα πακέτα λογισμικού και είναι συμβατό με όλα τα μοντέλα των Raspberry Pi, εκτός από τους μικροελεγκτές.

Υπάρχουν δύο κύριες εκδόσεις του Raspberry Pi OS, η έκδοση Desktop και η έκδοση Lite, για 32-bit και 64-bit αρχιτεκτονικές. Η έκδοση Desktop περιλαμβάνει ένα γραφικό περιβάλλον εργασίας (GUI), διανέμεται με ή χωρίς προ-εγκατεστημένα προγράμματα και ενδείκνυται για αρχάριους χρήστες. Η έκδοση Lite, διαθέτει τερματική κονσόλα, καταναλώνει λιγότερους πόρους και προορίζεται για λιγότερο ισχυρά μοντέλα ή για εφαρμογές όπου δεν απαιτείται γραφικό περιβάλλον εργασίας.

Εκτός του Raspberry Pi OS, υπάρχουν και άλλες διαθέσιμες Linux διανομές, οι οποίες υποστηρίζονται από επίσημους φορείς ή από την ενεργή κοινότητα [30].

3.3.2 Γλώσσα Προγραμματισμού Python

Πρόκειται για μια ανοιχτού κώδικα, γλώσσα προγραμματισμού, γενικού σκοπού (General-Purpose) & υψηλού επιπέδου (High-Level) [31]. Διαθέτει ενσωματωμένες δομές δεδομένων, όπως δυναμικές λίστες (Lists), πλειάδες (Tuples) και πίνακες κατακερματισμού (Dictionaries). Υπάρχουν δομές που καθορίζονται από το χρήστη, όπως πίνακες (Arrays), στοίβες (Stacks), ουρές (Queues), γραφικές παραστάσεις (Graphs), Deques και Hashmaps [32]. Ακόμη οι βιβλιοθήκες που τη συνοδεύουν είναι αρκετά εμπλουτισμένες και δεν υστερούν σε σχέση με τις αντίστοιχες βιβλιοθήκες άλλων γλωσσών προγραμματισμού όπως για παράδειγμα των Java, C και C++.

Η Python χρησιμοποιείται ευρέως στην ανάπτυξη εφαρμογών υπολογιστικής όρασης, μηχανικής μάθηση, ανάλυσης δεδομένων και λογισμικού. Σύμφωνα με το Δείκτη Δημοτικότητας Γλώσσας Προγραμματισμού (Popularity of Programming Language Index), η Python, συγκαταλέγεται στις πλέον δημοφιλείς γλώσσες, με περίπου 8.5 εκατομμύρια χρήστες [33].

Ο Guido van Rossum, ο δημιουργός της Python, ξεκίνησε να ασχολείται με την ανάπτυξη της, γλώσσας στις αρχές της δεκαετίας του 1990 [32]. Η δημιουργία της Python, οφείλεται στην προσπάθεια του να επιλύσει τα προβλήματα αλλά και τις ελλείψεις των γλωσσών προγραμματισμού εκείνης της εποχής. Θεωρείται διάδοχος της γλώσσας ABC, από την οποία έχει αντλήσει έμπνευση για αρκετές λειτουργίες της.

. Η πρώτη έκδοση της Python, η Python 0.9, κυκλοφόρησε το 1991. Στα τέλη του 2000, κυκλοφόρησε η Python 2.0, μία ποιο εμπλουτισμένη έκδοση, με νέα χαρακτηριστικά, όπως υποστήριξη Unicode, μέτρηση αναφορών (reference counting) και ανίχνευση κύκλου για συλλογή απορριμμάτων (cycle-detecting garbage collection).

Το 2001, την ανάπτυξη της Python ανέλαβε το μη κερδοσκοπικό ίδρυμα, Python Software Foundation. Το 2008 κυκλοφόρησε η Python 3.0, διορθώθηκαν σφάλματα προηγουμένων εκδόσεων, με αποτέλεσμα να διακοπεί η συμβατότητα με προηγούμενες εκδόσεις.

3.3.3 Περιβάλλον Ανάπτυξης - PyCharm CE

Το PyCharm αποτελεί ένα ολοκληρωμένο περιβάλλον ανάπτυξης (IDE), ειδικά σχεδιασμένο για την ανάπτυξη εφαρμογών με τη γλώσσα προγραμματισμού Python. Δημιουργήθηκε από την εταιρία JetBrains και κυκλοφόρησε για πρώτη φορά σε δοκιμαστική Beta έκδοση το 2010. Τρία χρόνια αργότερα, το 2013 κυκλοφόρησαν δύο εκδόσεις: η ανοιχτού κώδικα έκδοση PyCharm Community Edition και η επί πληρωμή έκδοση PyCharm Professional [34].

Το PyCharm προσφέρει πλήθος εργαλείων που διευκολύνουν και απλοποιούν τη διαδικασία της συγγραφής κώδικα. Τα συντακτικά σφάλματα επισημαίνονται με διαφορετικό χρώμα και γραμματοσειρές, ενώ προτεινόμενες εντολές εμφανίζονται σε αναδυόμενα παράθυρα. Η τεχνητή νοημοσύνη προσφέρει λειτουργίες που ενισχύουν την παραγωγικότητα και τη εμπειρία των χρηστών προβλέποντας και συμπληρώνοντας αυτόματα γραμμές κώδικα.

Η εύρεση μεταβλητών ή τμημάτων κώδικα, είναι εφικτή με τη βοήθεια του εργαλείου αναζήτησης και η πλοήγηση προς αυτών γίνεται μέσω του εύχρηστου γραφικού περιβάλλον χρήστη. Στο στάδιο της αποσφαλμάτωσης κώδικα (Debugging), ο χρήστης μπορεί να εντοπίσει και να διορθώσει σφάλματα, που εμφανίζονται κατά την εκτέλεση του προγράμματος.

Σχόλιο: Στο πλαίσιο της εργασίας, χρησιμοποιήσαμε την τελευταία έκδοση του PyCharm Community Edition για την συγγραφή, επεξεργασία και αποσφαλμάτωση κώδικα Python.

3.3.4 Jupyter Notebook

Το Jupyter Notebook ή αλλιώς IPython Notebook είναι μία εφαρμογή ιστού που χρησιμοποιείται ευρέως στην επιστήμη των δεδομένων, σε εφαρμογές ανάλυσης δεδομένων και μηχανικής μάθησης, αλλά και για εκπαιδευτικούς σκοπούς. Κυκλοφόρησε το 2014, από τον δημιουργό του IPython Project, Fernando Perez [35].

Το Jupyter είναι ένα ανοιχτού κώδικα διαδραστικό περιβάλλον ανάπτυξης, που επιτρέπει τη δημιουργία και τον διαμοιρασμό εγγραφών JSON. Στη δημιουργία του Jupyter Notebook, χρησιμοποιήθηκαν βιβλιοθήκες ανοιχτού κώδικα. Βασικό χαρακτηριστικό των σημειωματάριων Jupyter, είναι ότι τα περιεχόμενα του σημειωματάριου χωρίζονται σε κελιά.

Το περιεχόμενο ενός σημειωματάριου, μπορεί να είναι μικτό. Υπάρχουν δύο τύποι κελιών, τα κελιά κώδικα και κειμένου.

- Τα κελιά κειμένου μπορεί να περιέχουν εικόνες, μαθηματικές συναρτήσεις, πίνακες, γραφήματα και επεξηγηματικά υπομνήματα.

- Τα κελιά κώδικα, αποτελούνται από τρεις περιοχές, της περιοχής εισόδου, περιοχής εξόδου και περιοχής εμφάνισης. Η περιοχή εισόδου, περιέχει εκτελέσιμο προγραμματιστικό κώδικα. Υποστηρίζονται πάνω από εκατό γλώσσες προγραμματισμού. Κατά την εκτέλεση κώδικα, εμφανίζεται ένα νέο κελί, με το αποτέλεσμα της εκτέλεσης.

3.3.5 Google Colaboratory

Το Google Colaboratory ή αλλιώς Colab, αποτελεί μία έκδοση του Jupiter Notebook, με τη μορφή λογισμικού ως υπηρεσίας νέφους (SaaS). Αναπτύχθηκε από την Google και κυκλοφόρησε για πρώτη φορά το 2017 [36]. Το Colab είναι κατάλληλο για εκπαιδευτικούς σκοπούς, εφαρμογές μηχανικής μάθησης (Machine Learning) και της επιστήμης των δεδομένων (Data Science).

Η υπηρεσία διατίθεται δωρεάν και παρέχει πρόσβαση σε υπολογιστικούς πόρους, όπως κάρτες γραφικών (GPU) και μονάδες επεξεργασίας Tensor (TPU). Ακόμα διαθέτει ένα ευρύ φάσμα προ-εγκατεστημένων βιβλιοθηκών και μέσω του περιβάλλοντος προγραμματισμού, επιτρέπεται η εκτέλεση κώδικα γραμμένο σε γλώσσα προγραμματισμού Python. Η χρήση της υπηρεσίας, υπόκεινται σε περιορισμούς. Ορισμένοι περιορισμοί μπορούν να παρακαμφθούν, με την απόκτηση πακέτων συνδρομής, όπως σε περιπτώσεις, όπου οι διαθέσιμοι υπολογιστικοί πόροι της δωρεάν έκδοσης δεν είναι αρκετοί. Άλλοι περιορισμοί που δεν μπορούν να παρακαμφθούν, αφορούν τις δραστηριότητες, για τις οποίες η χρήση του Colab δεν επιτρέπεται.

Στην εργασία, το Colab χρησιμοποιήθηκε στην διαδικασία εκπαίδευσης των μοντέλων. Η εκπαίδευση είναι μια χρονοβόρα διαδικασία και ο χρόνος ολοκλήρωσης επηρεάζεται από πολλούς παράγοντες. Ο απαιτούμενος χρόνος μπορεί να μειωθεί σε ένα βαθμό, με τη χρήση ισχυρού υπολογιστικού υλικού. Οι προσφερόμενοι υπολογιστικοί πόροι της δωρεάν έκδοσης, που αναφέρονται στον Πίνακα 1, κάλυψαν επαρκώς τις υπολογιστικές ανάγκες και επίσπευσαν σημαντικά την διαδικασία εκπαίδευσης.

CPU	Intel Xeon CPU @ 2.20GHz
GPU	Tesla K80, 12GB GGDR5 VRAM
RAM	12GB RAM
HDD	25GB

Πίνακας 1: Χαρακτηριστικά του υπολογιστικού υλικού της Δωρεάν έκδοσης του Google Colab

3.3.6 Node-Red

Το Node-Red αποτελεί ένα ευρέως διαδεδομένο εργαλείο για την ανάπτυξη εφαρμογών Διαδικτύου των Πραγμάτων (IoT). Προσφέρει ένα περιβάλλον προγραμματισμού βασισμένο σε ροές (Flow-Based), το οποίο επιτρέπει την περιγραφή της λειτουργίας μίας εφαρμογής μέσω ενός δικτύου κόμβων. Επιπλέον βασίζεται στο Node.js και καθίσταται εφικτή η εκτέλεση κώδικα JavaScript. Κυκλοφόρησε για πρώτη φορά από την IBM το 2013, ενώ από το 2016 διανέμεται με νέα άδεια χρήσης ως λογισμικό ανοιχτού κώδικα [37].

Οι χρήστες μπορούν να αποκτήσουν πρόσβαση στο γραφικό περιβάλλον του Node-Red μέσω ενός προγράμματος περιήγησης ιστού. Το γραφικό περιβάλλον μεταξύ άλλων, περιλαμβάνει και τον επεξεργαστή ροών, ένα εργαλείο που επιτρέπει στους χρήστες να δημιουργούν και να διαχειρίζονται τις ροές τους. Η τοποθέτηση κόμβων στον επεξεργαστή ροών και η μεταξύ τους διασύνδεση, πραγματοποιείται μέσω “drag and drop”. Οι διαθέσιμοι κόμβοι εντοπίζονται στην παλέτα κόμβων (Palette). Η παλέτα κόμβων παρέχει μια συλλογή από προ-καθορισμένους κόμβους, οι οποίοι εκτελούν διάφορες λειτουργίες, όπως της αποσφαλμάτωσης, εκτέλεσης κώδικα JavaScript, καθώς της χειροκίνητης ενεργοποίησης της ροής. Δίνεται επίσης η δυνατότητα προσθήκης νέων κόμβων μέσω του διαχειριστή παλέτας (Palette Manager). Οι ροές που προκύπτουν είναι αρχεία με προεκτάσεις JSON.

3.3.7 Βιβλιοθήκες

Στη συγγραφή του απαραίτητου του κώδικα, έγινε χρήση βιβλιοθηκών, να αποκτήσουμε πρόσβαση σε προχωρημένες λειτουργίες, αλλά και να επισπεύσουμε τη διαδικασία συγγραφής του εκτελέσιμου προγράμματος. Παρακάτω θα γίνει παρουσίαση των βασικών βιβλιοθηκών που χρησιμοποιήθηκαν.

3.3.7.1 OpenCV

Η βιβλιοθήκη OpenCV (Open Computer Vision Library) αποτελεί μία ανοιχτού κώδικα βιβλιοθήκη που δημιουργήθηκε με σκοπό την ανάπτυξη μιας κοινής υποδομής σε εφαρμογές που βασίζονται στην υπολογιστική όραση και τη μηχανική μάθηση.

Προσφέρει περισσότερους από 2500 βελτιστοποιημένους αλγορίθμους, οι οποίοι χρησιμοποιούνται σε πλήθος εφαρμογών όπως η αναγνώριση αντικειμένων, η αναγνώριση προσώπων, η αναγνώριση κινήσεων, η αναγνώριση χειρονομιών και η επαυξημένη πραγματικότητα. Ακόμη, η OpenCV επιτρέπει την εξαγωγή τρισδιάστατων μοντέλων από εικόνες και βίντεο, καθώς και τον εντοπισμό εικόνων με παρόμοιο θέμα σε βάσεις δεδομένων [38].

Η OpenCV αναπτύχθηκε στο κέντρο ερευνών της Intel στη Ρωσία το 1999. Η πρώτη έκδοση άλφα της βιβλιοθήκης παρουσιάστηκε το 2000, στο συνέδριο IEEE, με κεντρικό θέμα την αναγνώριση προτύπων με την χρήση της υπολογιστικής όρασης. Στη διάρκεια της πρώτης δεκαετίας του 2000, κυκλοφόρησαν σημαντικές ενημερώσεις.

Η έκδοση OpenCV 2, κυκλοφόρησε τον Οκτώβριο του 2009, με σημαντικές αλλαγές και βελτιώσεις. Έκτοτε, ανά τακτά χρονικά διαστήματα κυκλοφορούν ενημερώσεις και βελτιώσεις. Το μη κερδοσκοπικό ίδρυμα OpenCV ανέλαβε την υποστήριξη της βιβλιοθήκης το 2012 [39].

Εικόνα 21: Λογότυπο βιβλιοθήκης OpenCV [πηγή]

3.3.7.2 NumPy

Η βιβλιοθήκη NumPy (Numerical Python), αποτελεί μία ανοιχτού κώδικα βιβλιοθήκη για τη γλώσσα προγραμματισμού Python [40]. Είναι μία από τις ποιο δημοφιλείς βιβλιοθήκες της γλώσσας και χρησιμοποιούμενη στην επιστήμη των υπολογιστών. Προσφέρει υποστήριξη για πολυδιάστατους πίνακες, συνοδευόμενη από μέσω μία εμπλουτισμένης συλλογή μαθηματικών συναρτήσεων υψηλού επιπέδου. Δημιουργήθηκε από τον Travis Oliphant το 2005 και η πρώτη έκδοση NumPy 1.0 κυκλοφόρησε το 2006, ενσωματώνοντας στοιχεία της βιβλιοθήκης Numeric. Η βιβλιοθήκη δέχεται τακτικά ενημερώσεις [41].

Εικόνα 22: Λογότυπο βιβλιοθήκης NumPy [πηγή]

3.3.7.3 Matplotlib

Η Matplotlib, είναι μια ανοιχτού κώδικα βιβλιοθήκη για τη γλώσσα Python, η οποία προσφέρει αντικειμενοστραφή διεπαφή και εργαλεία για την οπτικοποίηση δεδομένων και την δημιουργία γραφημάτων [42]. Δημιουργήθηκε από τον John D. Hunter, το 2002 και κυκλοφόρησε το 2003, με άδεια ανοιχτού λογισμικού τύπου BSD. Σήμερα, η συντήρηση της Matlplotlib πραγματοποιείται χάρη στην ενεργή κοινότητα [43].

Εικόνα 23: Λογότυπο βιβλιοθήκης Matplotlib [πηγή]

3.3.7.4 PyTorch

Η PyTorch αποτελεί μία ανοιχτού κώδικα βιβλιοθήκη για εφαρμογές μηχανικής μάθησης (Machine Learning) και επιστήμης των δεδομένων (Data Science) [44]. Βασίζεται στην Torch, μια βιβλιοθήκη, που παρέχει ένα υπολογιστικό πλαίσιο (Framework) γραμμένο στη γλώσσα προγραμματισμού C.

Η PyTorch διαθέτει δύο σημαντικά χαρακτηριστικά υψηλού επιπέδου. Προσφέρει εργαλεία υλοποίησης και εκπαίδευσης σύνθετων νευρωνικών δικτύων, καθώς και πολυδιάστατους πίνακες δεδομένων που ονομάζονται Tensors. Οι Tensors φέρουν ομοιότητες, με τους πίνακες της βιβλιοθήκης NumPy και ο υπολογισμός τους, επιταχύνεται σημαντικά με τη χρήση μονάδων επεξεργασίας γραφικών (GPU).

Η PyTorch κυκλοφόρησε το 2016 από την Meta AI, πρώην Facebook AI και διανέμεται με άδεια ανοιχτού λογισμικού τύπου BSD. Από το 2022 και έπειτα η PyTorch βρίσκεται υπό την αιγίδα του PyTorch Foundation, μία θυγατρική του μη κερδοσκοπικού ιδρύματος Linux Foundation.

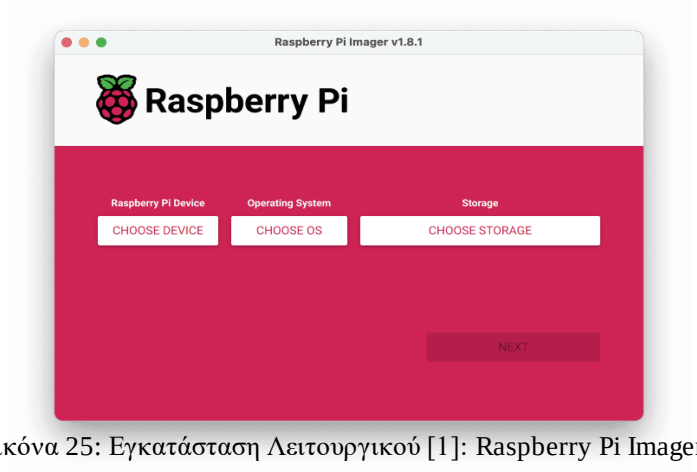
Εικόνα 24: Λογότυπο βιβλιοθήκης PyTorch [πηγή]

3.4 Υλοποίηση Συστήματος

3.4.1 Εγκατάσταση Λειτουργικού συστήματος

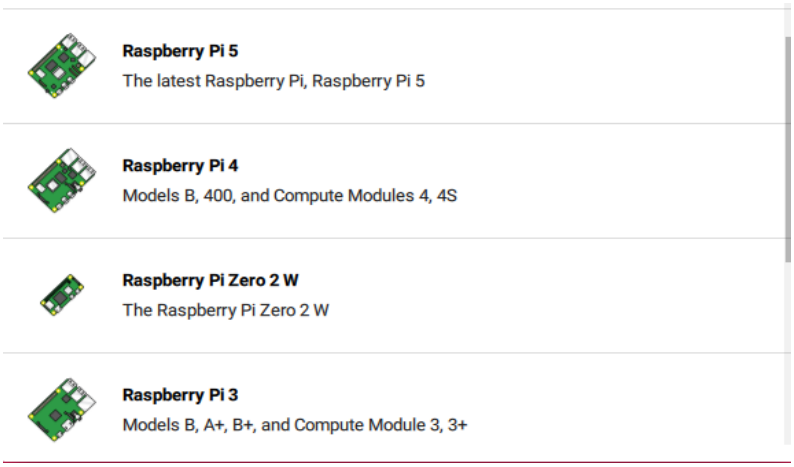
Οι δημιουργοί του Raspberry Pi, θέλοντας να διευκολύνουν τους νέους χρήστες δημιούργησαν ένα δωρεάν λογισμικό για την διαμόρφωση και την εγγραφή του λειτουργικού συστήματος, με ονομασία "Raspberry Pi Imager" [45]. Το πρόγραμμα απλοποιεί τη διαδικασία εγγραφής του λειτουργικού συστήματος, μέσω ενός φιλικού προς τον χρήστη, γραφικού περιβάλλοντος.

Αφού εγκαταστήσουμε το πρόγραμμα, το εκτελούμε και εμφανίζεται το παρακάτω παράθυρο:



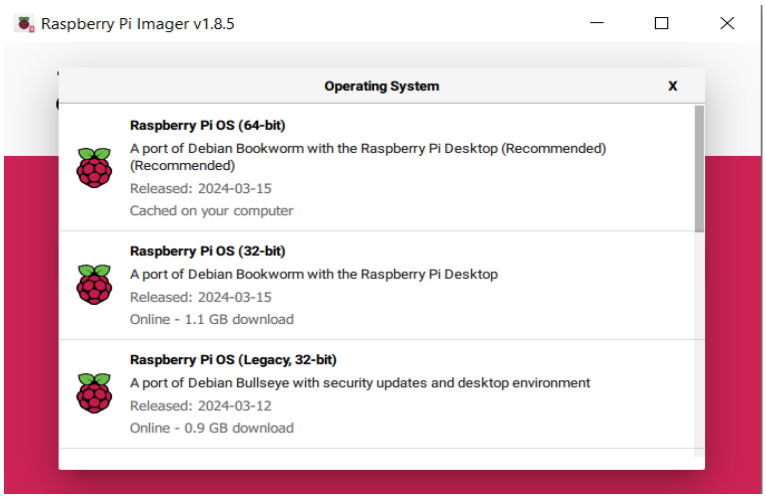
Εικόνα 25: Εγκατάσταση Λειτουργικού [1]: Raspberry Pi Imager

Στη συνέχεια, πατάμε στο "CHOOSE DEVICE" και επιλέγουμε από τη λίστα το Raspberry Pi 4.



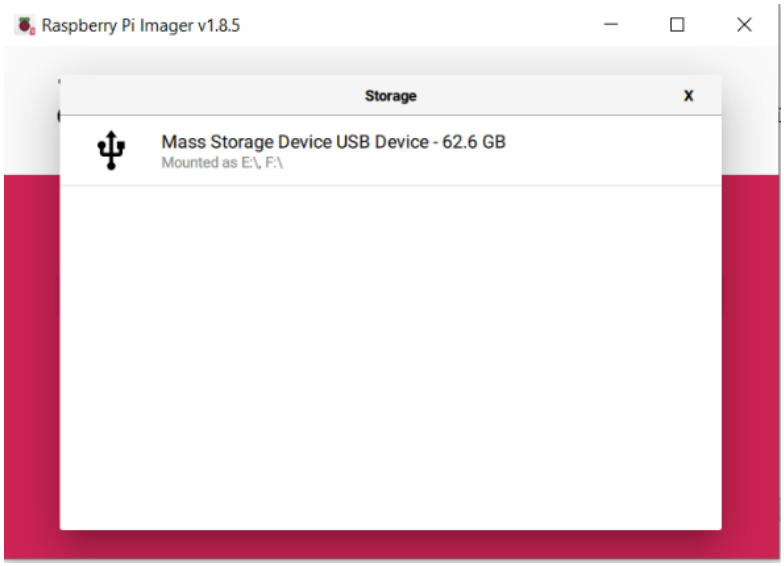
Εικόνα 26: Εγκατάσταση Λειτουργικού [2]: Επιλογή του Raspberry Pi 4

Στη συνέχεια, επιλέγουμε το λειτουργικό σύστημα. Πατώντας στο "CHOOSE OS", επιλέγουμε την έκδοση Raspberry Pi OS (64-bit).



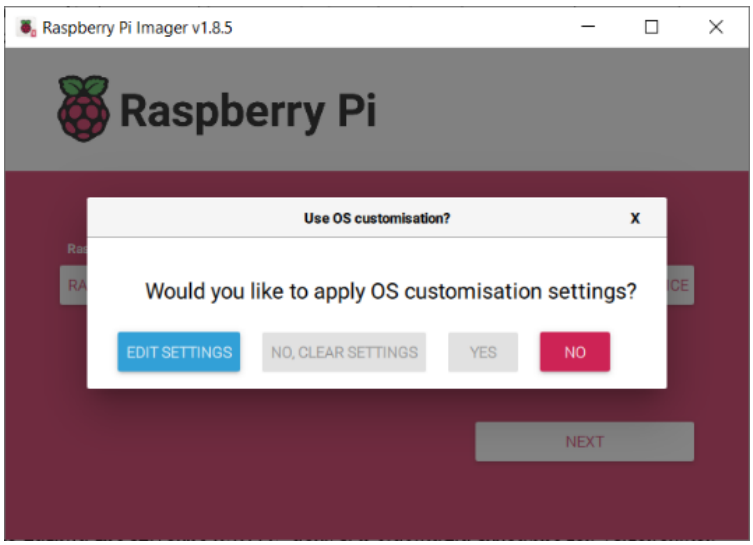
Εικόνα 27: Εγκατάσταση Λειτουργικού [3]: Επιλογή του Raspberry Pi OS (64-bit)

Έπειτα τοποθετούμε την κάρτα μνήμης στον υπολογιστή και κάνουμε κλικ στο "CHOOSE STORAGE". Από τη λίστα που εμφανίζεται, επιλέγουμε την κάρτα μνήμης micro-SD ως το αποθηκευτικό μέσο στο οποίο θα εγγραφεί το λειτουργικό σύστημα.



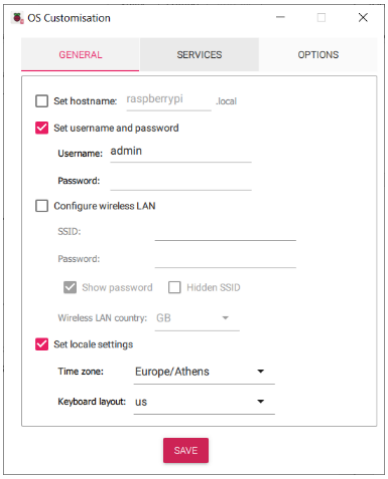
Εικόνα 28: Εγκατάσταση Λειτουργικού [4]: Επιλογή αποθηκευτικού μέσου

Στη συνέχεια, πατώντας το κουμπί "NEXT", προχωράμε στην επιλογή των ρυθμίσεων εξατομίκευσης.



Εικόνα 29: Εγκατάσταση Λειτουργικού [5]: Παραμετροποιήσεις Λειτουργικού Συστήματος 1/2

Συμπληρώνουμε τα πεδία της καρτέλας "General", του παραθύρου "OS Customization" Εικόνας 30. Επίσης από τη καρτέλα "Services" (Εικόνα 30), ενεργοποιούμε την υπηρεσία SSH για απομακρυσμένη πρόσβαση στο Raspberry Pi.



Εικόνα 30: Εγκατάσταση Λειτουργικού [6]: Παραμετροποιήσεις Λειτουργικού συστήματος 2/2

Τέλος, πατώντας στο "WRITE", πραγματοποιείται η εγγραφή του λειτουργικού συστήματος στην κάρτα μνήμης.

3.4.2 Εγκατάσταση Βιβλιοθηκών Python

Έχοντας ολοκληρώσει με επιτυχία τη διαδικασία εγγραφής του λειτουργικού και πραγματοποιήσει την πρώτη εκκίνηση του Pi, προχωρούμε στην εγκατάσταση των απαραίτητων πακέτων και βιβλιοθηκών.

Πρώτα ανοίγουμε το τερματικό και εκτελούμε τις παρακάτω εντολές:

sudo apt-get update && sudo apt-get upgrade -y

Μέσω των παραπάνω εντολών, πραγματοποιείται έλεγχος στα αποθετήρια του συστήματος για τυχών ενημερώσεις των εγκατεστημένων πακέτων του συστήματος.

Με το λειτουργικό πλήρως ενημερωμένο, εκτελούμε τη εντολή που ακολουθεί και ελέγχουμε αν η Python είναι εγκατεστημένη στο σύστημα.

python --version

Στην έξοδο του τερματικού εμφανίζεται η έκδοση 3.11.2 της Python.

Εικόνα 31: Έλεγχος έκδοσης Python

Στη συνέχεια ακολουθεί η δημιουργία ενός εικονικού περιβάλλοντος, χρησιμοποιώντας το προ εγκατεστημένο πακέτο Virtualenv. Η χρήση ενός εικονικού περιβάλλοντος είναι μία πρακτική που συνίσταται για την αποφυγή τυχών προβλημάτων ασυμβατότητας. Το εικονικό περιβάλλον είναι ένας απομονωμένος χώρος με ανεξάρτητες βιβλιοθήκες και διερμηνείς Python από του συστήματος.

Εκτελώντας τις εντολές που ακολουθούν, δημιουργήσαμε ένα εικονικό περιβάλλον

python -m venv /path/venv

Ενεργοποίησή του εικονικού περιβάλλοντος:

source /bin/activate

Κλωνοποίηση αποθετηρίου yolov5:

git clone https://github.com/ultralytics/yolov5

Μετάβαση στο τοπικό ευρετήριο:

cd yolov5

Εγκατάσταση απαιτούμενων πακέτων:

pip install -r requirements.txt

3.4.3 Εγκατάσταση Node-Red

Η εγκατάσταση του Node-Red πραγματοποιείται μέσω της παρακάτω εντολής:

bash <curl -sL https://raw.githubusercontent.com/node-red/linux-installers/master/deb/update-nodejs-and-nodered)

Μέσω της εντολής κατεβαίνει και εκτελείται ένα bash script που εγκαθιστά τα απαραίτητα πακέτα για την εκτέλεση του Node-Red, συμπεριλαμβανομένων των Node.js και npm.

Στη συνέχεια ενεργοποιούμε την υπηρεσία: nodered.service, μέσω της εντολής:

sudo systemctl enable nodered.service

3.4.3.1 Παραμετροποίηση Node-Red

Για τη δημιουργία του "Dashboard" και την υλοποίηση ορισμένων λειτουργειών, απαιτείται η εγκατάσταση των παρακάτω μη προ-εγκατεστημένων κόμβων. Η εγκατάσταση πραγματοποιείται με την εντολή:

npm install [package-name]

Απαραίτητα πακέτα – κόμβοι:

- **node-red-dashboard:** Για τη δημιουργία και διαμόρφωση του Dashboard είναι απαραίτητη η χρήση αυτού του πακέτου, Μέσω των χρήσιμων κόμβων που παρέχονται από μία σειρά από κόμβους, επιτρέπεται η δημιουργία του Dashboard, καθώς και η διαμόρφωση ανάλογα «με τα θέλω» και τις ανάγκες και τις απαιτήσεις της υλοποίησης.

- **node-red-ui-iframe:** Αυτό το πακέτο περιλαμβάνει τον κόμβο iframe, έναν κόμβο που επιτρέπει την ενσωμάτωση ιστοσελίδων στο Dashboard. Στη περίπτωση μας, χρησιμοποιείται για την ενσωμάτωση της ζωντανής ροής της κάμερας με τα bounding boxes των αντικειμένων που ανιχνεύθηκαν.

- **node-red-contrib-pythonshell:** Αυτό το πακέτο επιτρέπει την εκτέλεση Python scripts μέσα στο Node-Red.

- **node-red-contrib-ui-digital-display:** Ένας από τους κόμβους περιέχονται επιτρέπει τη χρήση ενός widget, μιας ψηφιακής οθόνης με ψηφία για χρήση σε συνδυασμό με το Dashboard.

  Εγκατάσταση των κόμβων:

  - **npm install node-red-dashboard**
  - **npm install node-red-ui-iframe**
  - **npm install node-red-contrib-pythonshell**
  - **npm install node-red-contrib-ui-digital-display**

3.4.4 Υλοποίηση του Πίνακα Dashboard

Σκοπός της χρήσης του Node-Red είναι η υλοποίηση ενός πίνακα οργάνων που θα παρέχει πληροφορίες σχετικά με την πληρότητα του χώρου στάθμευσης.

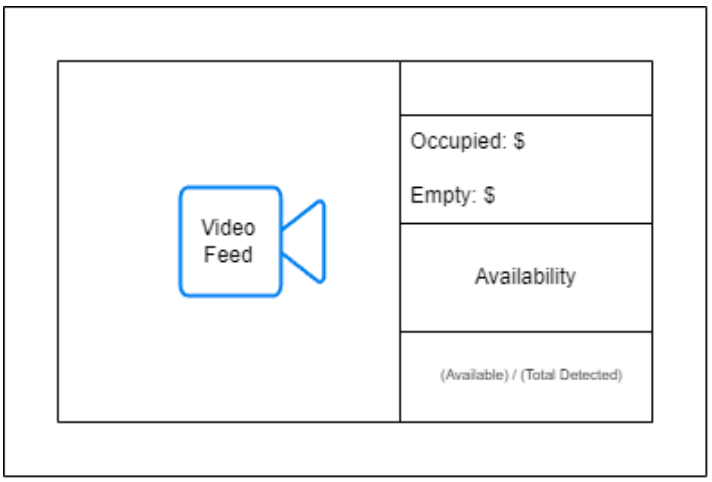
3.4.4.1 Περιγραφή του Πίνακα Dashboard

Αρχικά, το Dashboard θα περιλαμβάνει δύο βασικά πλαίσια:

1. **Ζωντανή ροή:** Σε αυτό το πλαίσιο θα εμφανίζεται η ζωντανή ροή εικόνων ή βίντεο από τον χώρο στάθμευσης, πάνω στην οποία θα παρουσιάζονται οι ελεύθερες και κατειλημμένες θέσεις που ανιχνεύθηκαν.
2. **Προβολή αποτελεσμάτων:** Σε αυτό το πλαίσιο θα εμφανίζονται τα αποτελέσματα που προέκυψαν από την ανάλυση, δηλαδή ο αριθμός των ελευθέρων και των κατειλημμένων θέσεων.

Σχόλιο: Πρέπει να σημειωθεί ότι ορισμένες θέσεις ενδέχεται να μην ανιχνευθούν, ακόμη και αν απεικονίζονται στη ζωντανή ροή. Σε επόμενο κεφάλαιο θα γίνει σχετική αναφορά.

Η τελική μορφή του Dashboard θα έχει μορφή παρόμοια με της εικόνας που ακολουθεί:



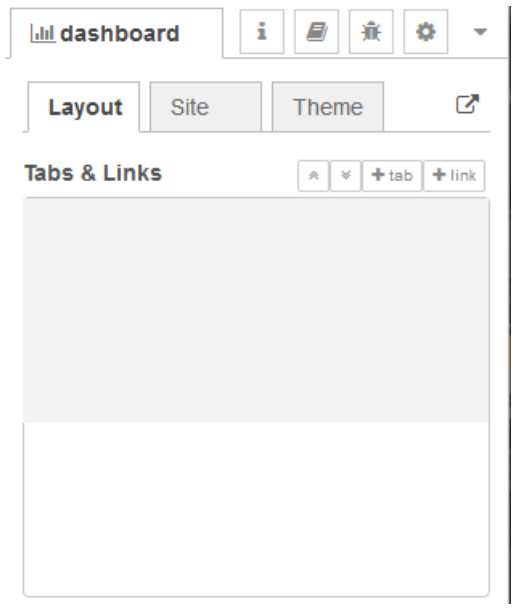
Εικόνα 32: Σχέδιο του πίνακα Dashboard

3.4.4.2 Δημιουργία Πίνακα Dashboard

Έχοντας λάβει την απόφαση για τη διάταξη του Πίνακα, ακολουθεί η υλοποίηση του, σύμφωνα με τα παρακάτω βήματα:

1. Αρχικά, κάνουμε "κλικ" στην επιλογή "Dashboard" που βρίσκεται στη πλαϊνή μπάρα (Sidebar) του χώρου εργασίας (Workspace).

2. Στο παράθυρο που εμφανίζεται, πατάμε στην επιλογή "Tab" για να δημιουργήσουμε ένα νέο Dashboard.



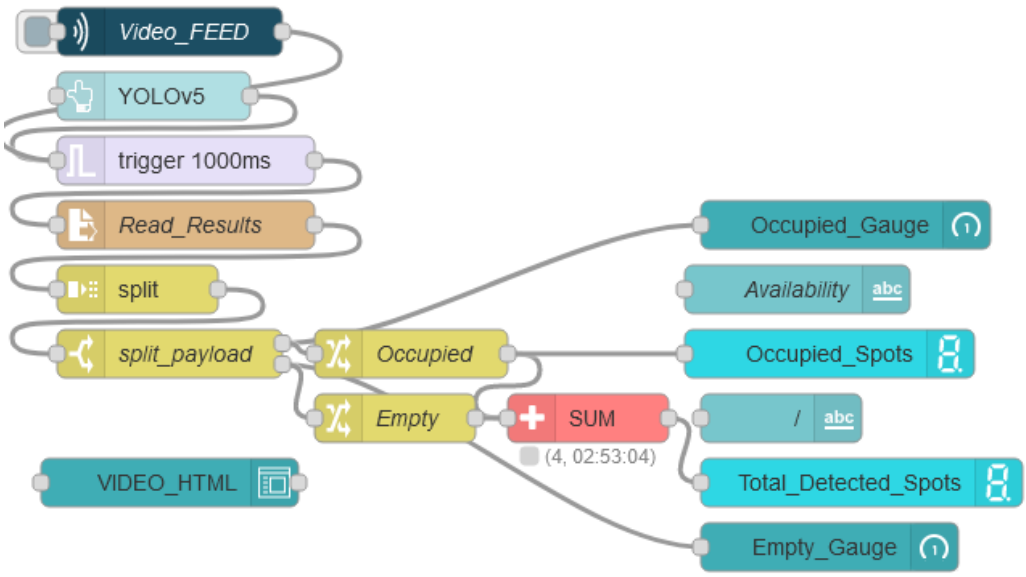
Εικόνα 33: Node-Red [1] – Δημιουργία νέου Dashboard

Στη συνέχεια, πατάμε στην επιλογή "Group" και δημιουργούμε δύο νέες ομάδες:

- Πρώτη Ομάδα: Σε αυτή την ομάδα ανήκει ένας μόνο κόμβος, μέσω του οποίου προβάλλεται η μεταδιδόμενη ροή.

- Δεύτερη Ομάδα: Σε αυτή την ομάδα ανήκουν οι κόμβοι που παρέχουν τις σχετικές πληροφορίες, χρησιμοποιώντας τα αντίστοιχα widget.

Έπειτα τοποθετούμε τους απαραίτητους κόμβους για τη δημιουργία της ροής, όπως φαίνεται στη παρακάτω εικόνα:



Εικόνα 34: Node-Red [2] Flow – Dashboard

Ακολουθεί μία σύντομη επεξήγηση της ροής:

Ο κόμβος Video_Feed επιτρέπει την εκτέλεση του Python Script YOLOv5.py, ενεργοποιώντας τη μετάδοση ροής και το Inference μέσω του YOLO. Η περιγραφή της μετάδοσης ακολουθεί στο Κεφάλαιο 5.

Ο κόμβος Read_Results, ενεργοποιείται αυτόματα και διαβάζει το περιεχόμενο ενός αρχείου με μία γραμμή της μορφής [x,y]. Όπου περιλαμβάνονται τα πλήθη των ελευθέρων και κατειλημμένων θέσεων.

Οι κόμβοι Split και Split_Payload διαχωρίζουν τα δεδομένα σε δύο εξόδους, η κάθε μία περιλαμβάνει μία αλφαριθμητική τιμή (String). Οι κόμβοι Occupied και Empty μετατρέπουν τις αλφαριθμητικές τιμές σε αριθμητικές.

Τα widgets Occupied_Gauge, Empty_Gauge, Occupied_Spots και Total_Detected_Spots εμφανίζουν με διάφορους τρόπους τα πλήθη των κενών και των κατειλημμένων θέσεων.

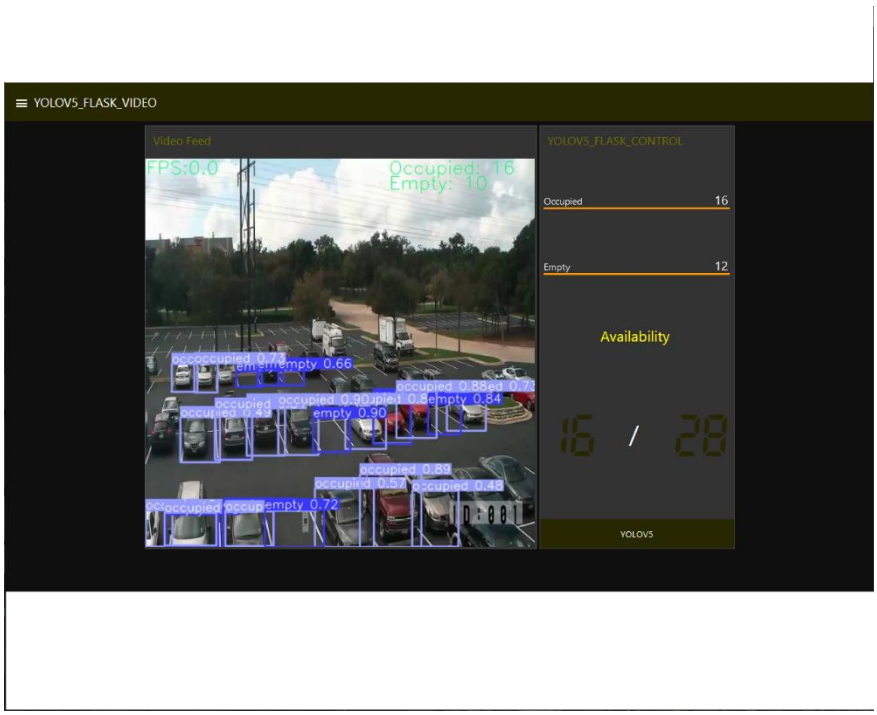
Ο Video_HTML περιλαμβάνει τη URL διεύθυνση όπου γίνεται μετάδοση της ροής.

Παρακάτω ακολουθεί η διάταξη των widget και του Dashboard:

Εικόνα 35: Node-Red [3] - Layout

Μετά το "Deployment" της ροής και την πρόσβαση στην ιστοσελίδα: **http://127.0.0.1:1880/ui,** εμφανίζεται το τελικό αποτέλεσμα του Dashboard.

Τελική μορφή του Dashboard:



Εικόνα 36: Node-Red [4] - Τελικό Dashboard

Κεφάλαιο 4: Εκπαίδευση Μοντέλων

Στο παρόν κεφάλαιο, θα εξερευνήσουμε την εκπαίδευση μοντέλων βασιζόμενων σε διάφορες εκδόσεις του αλγορίθμου YOLO, με στόχο την επιλογή του μοντέλου με την υψηλότερη ακρίβεια για μετέπειτα χρήση με το Raspberry Pi. Πριν ξεκινήσουμε τη διαδικασία εκπαίδευσης, θα αναφερθούμε στα βασικά βήματα που θα ακολουθήσουμε, όπως η επιλογή του κατάλληλου συνόλου δεδομένων και η προετοιμασία των σημειωματάριων Jupyter για χρήση με το Colab.

4.1 Σύνολο Δεδομένων – Dataset

Τα σύνολα δεδομένων αποτελούν απαραίτητο στοιχείο στην εκπαίδευση μοντέλων μηχανικής όρασης. Αποτελούνται από πλήθος εικόνων, που επιτρέπουν στο μοντέλο να αναγνωρίζει μοτίβα και να πραγματοποιεί προβλέψεις σε νέα, άγνωστα δεδομένα.

Υπάρχουν διάφοροι τύποι συνόλων δεδομένων για χρήση στη μηχανική όραση, όπως τα σύνολα εικόνων (Image Dataset), σύνολα βίντεο (Video Dataset), τρισδιάστατα σύνολα δεδομένων (3D Datasets) και τα συνθετικά σύνολα δεδομένων (Synthetic Dataset).

Σε αυτό το σημείο επίσης είναι αναγκαία η λήψη απόφασης που αφορά την επιλογή μεταξύ ενός έτοιμου συνόλου δεδομένων ή της δημιουργίας.

Η δημιουργία ενός νέου συνόλου δεδομένων απαιτεί τη συλλογή και οργάνωση σχετικών δεδομένων, με το πρόβλημα που καλείται να λύσει το εκπαιδευμένο μοντέλο. Στην περίπτωση μας, το σύνολο δεδομένων πρέπει να περιέχει μεγάλο πλήθος από εικόνες με θέμα, πολλούς και διαφορετικούς χώρους στάθμευσης, με διαφορετικές γωνίες λήψης και καιρικές συνθήκες. Πρόκειται για μία χρονοβόρα και περίπλοκη διαδικασία στην οποία απαιτείται η μετακίνηση και η πρόσβαση δημόσιους και ιδιωτικούς χώρους στάθμευσης με αποτέλεσμα να τίθενται ζητήματα νομικής φύσεως όπως της προστασία των προσωπικών δεδομένων καθώς επίσης και της αδειοδότησης για την πρόσβαση στους χώρους.

Προς το παρόν θα αγνοηθούν οι προαναφερθείσες προκλήσεις και θα παρουσιαστούν τα απαιτούμενα βήματα για τη δημιουργία ενός συνόλου δεδομένων, με στόχο την αναγνώριση θέσεων στάθμευσης

4.1.1 Συλλογή εικόνων για τη δημιουργία Dataset

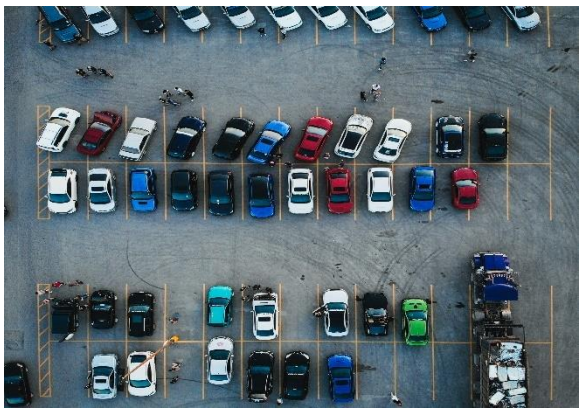
Η διαδικασία συλλογής εικόνων, όπως προαναφέρθηκε απαιτεί τη λήψη πολλών εικόνων υπό διαφορετικές καιρικές συνθήκες και γωνίες λήψεις. Μέρος της διαδικασίας μπορεί να αυτοματοποιηθεί και ενδεχομένως να επιταχυνθεί σε μικρό βαθμό με τη χρήση ενός Python Script, όπως αυτό που ακολουθεί, το οποίο επιτρέπει τη λήψη εικόνων με ρυθμό 30 λεπτών.

dataset_collection.py

```
from picamera2 import Picamera2, Preview
from libcamera import Transform
from os import system
from time import sleep

camera = Picamera2()
camera.resolution = (1920, 1080)
camera.start()
for i in range(10):
    filename = f"image_{i+1}.jpg"
    camera.capture_file(filename)
    sleep(1000)
```

Εικόνα 37: Σύνολο Δεδομένων [1] - Ενδεικτικό Script ImageCap.py - Λήψη εικόνων με το Pi Camera Module 3



Εικόνα 38: Σύνολο Δεδομένων [2] Ενδεικτική Εικόνα 1/2

Εικόνα 39: Σύνολο Δεδομένων [3] Ενδεικτική Εικόνα 2/2

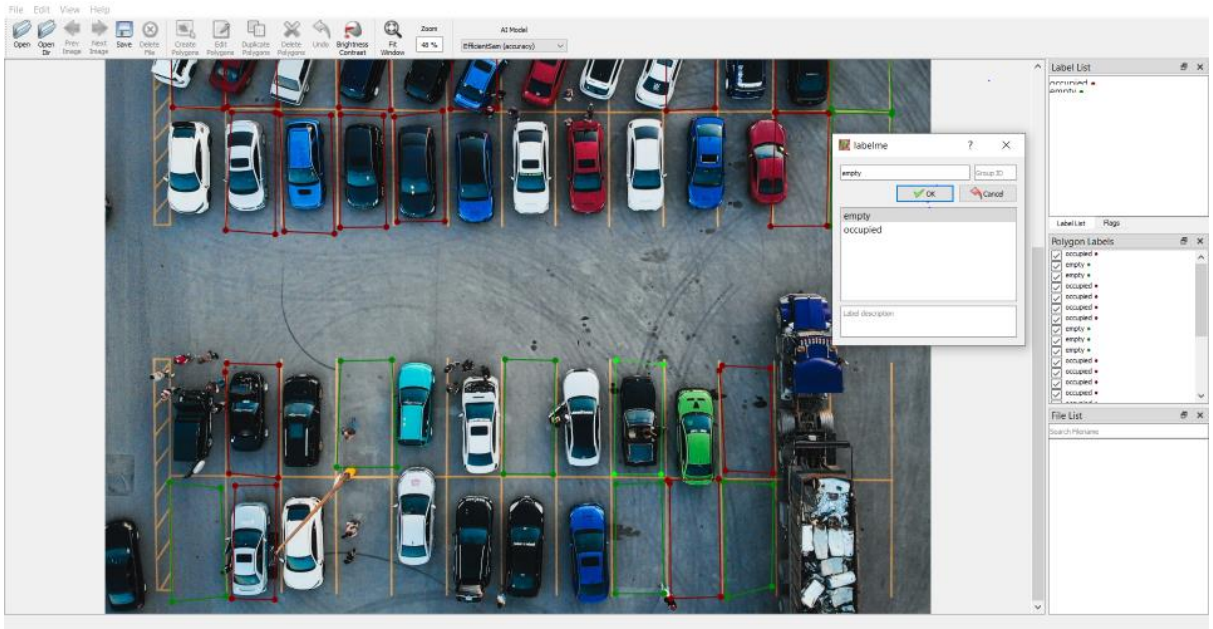
4.1.2 Επισήμανση Εικόνων (Image Annotation)

Η επισήμανση των εικόνων αποτελεί μία χρονοβόρα διαδικασία που πραγματοποιείται μετά την λήψη των εικόνων. Η διαδικασία αυτή περιλαμβάνει τον προσδιορισμό των αντικείμενων που επιθυμούμε το μοντέλο να αναγνωρίζει μέσω της εκπαίδευσης.

Η επισήμανση των αντικειμένων πραγματοποιείται, με τη χρήση εργαλείων επισήμανσης, τα οποία επιτρέπουν τη δημιουργία πλαισίων οριοθέτησης (Bounding Boxes) ή πολυγώνων γύρω από τα αντικείμενα.

Ένα από τα πιο γνωστά εργαλεία επισήμανσης είναι το LabelImg, ένα ανοιχτού κώδικα εργαλείο επισήμανσης, που επιτρέπει τη σχεδίαση πλαισίων οριοθέτησης και τη δημιουργία αρχείων επισήμανσης συμβατών με τον αλγόριθμο YOLO.

Στην περίπτωση της εργασίας στις υπό-επεξεργασία εικόνες του Dataset, απαιτείται η δημιουργία πλαισίων οριοθέτησης τόσο στις κενές όσο και στις κατειλημμένες θέσεις.

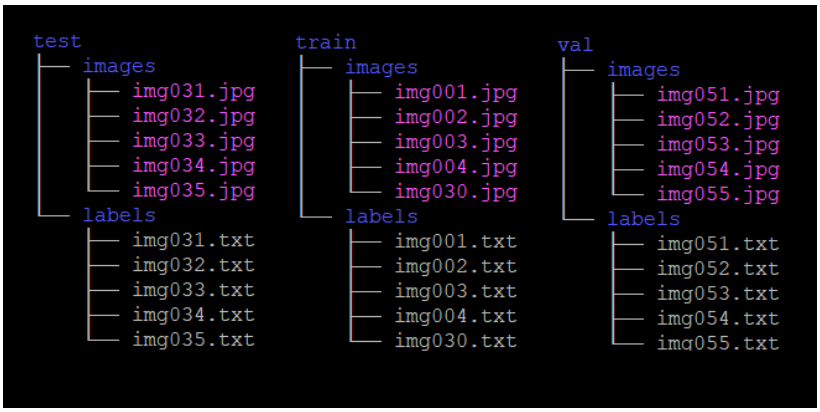


Σύνολο Δεδομένων [4] – Ενδεικτικό παράδειγμα επισήμανσης σε εικόνα του συνόλου δεδομένων

4.1.3 Διαχωρισμός Συνόλου Δεδομένων

Πριν από την εκπαίδευση του δικτύου, πραγματοποιείται ο διαχωρισμός του Dataset στα υποσύνολα Εκπαίδευσης (Train), Επικύρωσης (Validation) και Αξιολόγησης (Test). Συνήθης πρακτική αποτελεί, ο διαχωρισμός του συνόλου δεδομένων με ποσοστά 64% για το Train, 16% για το Validation και 20% για το Test.

Επιπλέον, είναι απαραίτητη η δημιουργία ενός νέου φακέλου, για κάθε υποσύνολο, ο οποίος θα περιλαμβάνει τα αρχεία με τις συντεταγμένες των επισημασμένων αντικειμένων.



Εικόνα 40: Σύνολο Δεδομένων [5] - Διαχωρισμός σε υποσύνολα

Συνοψίζοντας: Στη δημιουργία ενός συνόλου δεδομένων:

Απαιτείται η συλλογή πολλών εικόνων που απεικονίζουν το αντικείμενο που επιθυμούμε να αναγνωρίζει το μοντέλο. Στη συνέχεια πραγματοποιείται η επισήμανση των αντικειμένων στις εικόνες μέσω πλαισίων οριοθέτησης. Τέλος οι εικόνες χωρίζονται σε τρία υποσύνολα που χρησιμοποιούνται για εκπαίδευση, έλεγχο και αξιολόγηση του μοντέλου. Λαμβάνοντας υπόψιν τον υψηλό χρόνο που απαιτείται για τη δημιουργία ενός συνόλου δεδομένων, καθώς επίσης και τα πιθανά νομικά ζητήματα, καταλήξαμε στη χρήση ενός έτοιμου συνόλου δεδομένων.

4.1.4 Αναζήτηση συνόλου δεδομένων

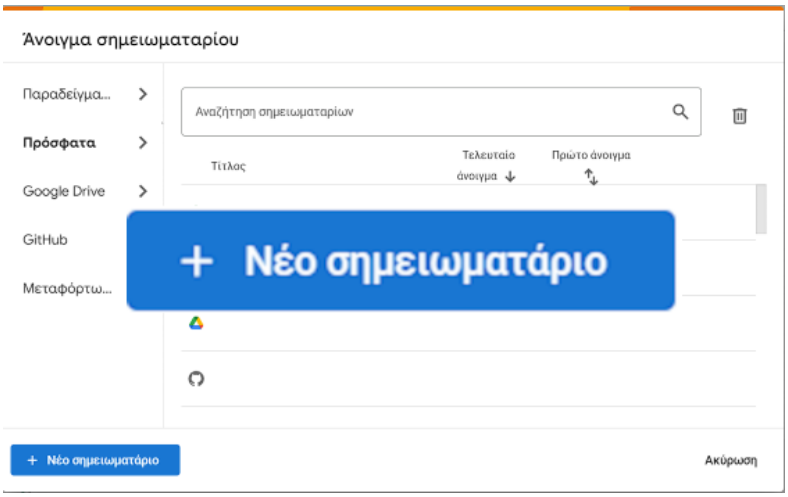
Το σύνολο δεδομένων που κληθήκαμε να εντοπίσουμε, πρέπει να αποτελείται από εικόνες με χώρους στάθμευσης. Η αναζήτηση σχετικών συνόλων δεδομένων πραγματοποιήθηκε στα αποθετήρια Roboflow και Kaggle, χρησιμοποιώντας τη φράση «parking spot detection» ως βασικό όρο αναζήτησης. Έπειτα από σχετική αναζήτηση, επιλέχθηκε το dataset με ονομασία «parking-spot-detector-1» [46]. Το σύνολο δεδομένων αποτελείται από 1599 επισημασμένες εικόνες, από τις οποίες οι 1379 (86,2%) ανήκουν στο υποσύνολο εκπαίδευσης, 199 στο υποσύνολο επικύρωσης και 21 στο υποσύνολο αξιολόγησης.

4.2 Προετοιμασία Google Colab

Στις προηγούμενες υποενότητες του κεφαλαίου, επιλέξαμε το Dataset που θα χρησιμοποιήσουμε για την εκπαίδευση του μοντέλου. Σε αυτή την υποενότητα, θα πραγματοποιήσουμε τις απαραίτητες παραμετροποιήσεις στο σημειωματάριο Jupyter και θα εκπαιδεύσουμε τα μοντέλα.

4.2.1 Δημιουργία σημειωματάριου Jupyter & εγκατάσταση εξαρτήσεων

Πρώτα, μεταβαίνουμε στην επίσημη ιστοσελίδα του Colab. Στη συνέχεια ακολουθούμε τις οδηγίες για να δημιουργήσουμε τρία νέα σημειωματάρια με ονομασίες YOLOv5, YOLOv8, YOLOv10.

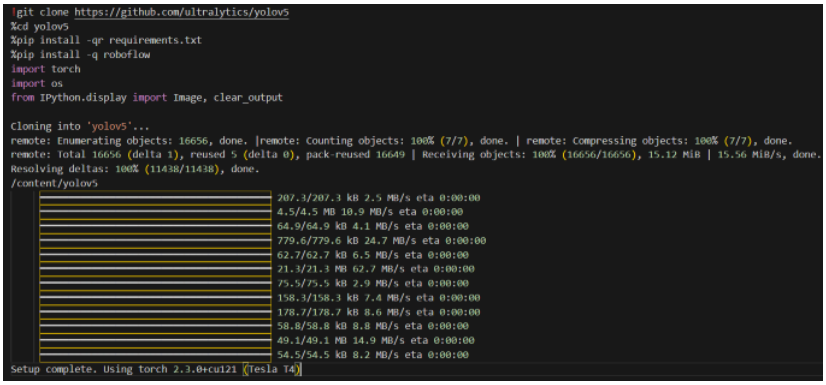


Εικόνα 41: Colab - Δημιουργία σημειωματάριων Jupyter

4.2.2 Εγκατάσταση εξαρτήσεων

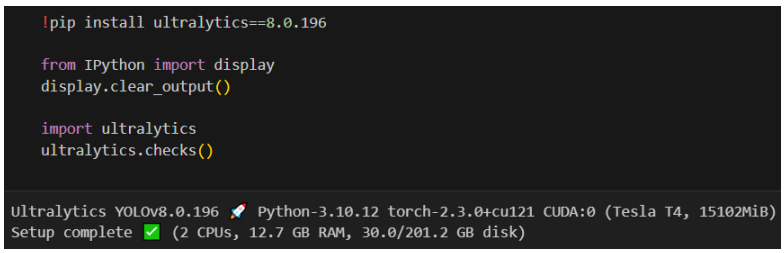
Στη συνέχεια ακολουθεί η εγκατάσταση των εξαρτήσεων και πακέτων στο Colab για κάθε μία από τις εκδόσεις που επιλέξαμε. Η Ultralytics, δημιουργός των YOLOv5 & YOLOv8, προσφέρει πέρα από τις βιβλιοθήκες που χρησιμοποιήθηκαν και σημειωματάρια Jupyter, από τα οποία θα αντλήσουμε ορισμένα τμήματα κώδικα.

YOLOv5: [47]



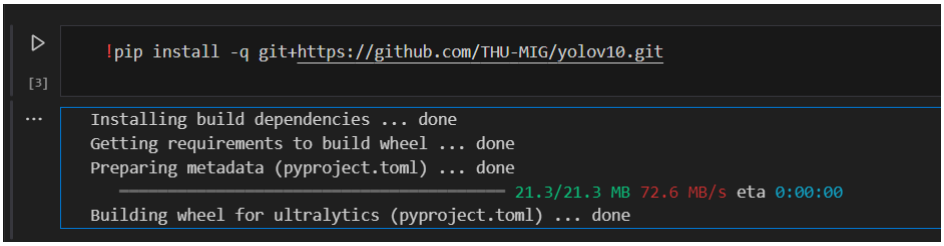
Εικόνα 42: Εγκατάσταση εξαρτήσεων [1] – YOLOv5

YOLOv8: [48]



Εικόνα 43: Εγκατάσταση εξαρτήσεων [2] – YOLOv8

YOLOv10: [49]



Εικόνα 44: Εγκατάσταση εξαρτήσεων [2] – YOLOv10

4.2.3 Μεταφόρτωση του Dataset στο Colab

Η μεταφόρτωση του συνόλου δεδομένων που επιλέχθηκε πραγματοποιείται μέσω του παρακάτω τμήματος κώδικα. Αφού προηγηθεί εγκατάσταση της βιβλιοθήκης Roboflow, και μέσω του μοναδικού κλειδιού πραγματοποιείται σύνδεση στο API της Roboflow και γίνεται λήψη του συνόλου δεδομένων.

```
!pip install roboflow

from roboflow import Roboflow
rf = Roboflow(api_key="                              ")
project = rf.workspace("car-parking-space").project("parking-spot-detector-a84ql")
version = project.version(1)
dataset = version.download("yolov8")
```

Εικόνα 45: Μεταφόρτωση Συνόλου Δεδομένων στο Colab [1]

Στην έξοδο του τερματικού λαμβάνουμε, ότι πραγματοποιήθηκε με επιτυχία η λήψη του συνόλου δεδομένων στο περιβάλλον εργασίας:

```
Collecting roboflow
  Downloading roboflow-1.1.30-py3-none-any.whl (75 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 75.5/75.5 kB 873.2 kB/s eta 0:00:00
Collecting certifi==2023.7.22 (from roboflow)
  Downloading certifi-2023.7.22-py3-none-any.whl (158 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 158.3/158.3 kB 2.7 MB/s eta 0:00:00
Collecting chardet==4.0.0 (from roboflow)
  Downloading chardet-4.0.0-py2.py3-none-any.whl (178 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 178.7/178.7 kB 10.4 MB/s eta 0:00:00
Collecting cycler==0.10.0 (from roboflow)
  Downloading cycler-0.10.0-py2.py3-none-any.whl (6.5 kB)
Collecting idna==2.10 (from roboflow)
  Downloading idna-2.10-py2.py3-none-any.whl (58 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 58.8/58.8 kB 8.7 MB/s eta 0:00:00
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.10/dist-packages (from roboflow) (1.4.5)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (from roboflow) (3.7.1)
Requirement already satisfied: numpy>=1.18.5 in /usr/local/lib/python3.10/dist-packages (from roboflow) (1.25.2)
Collecting opencv-python-headless==4.8.0.74 (from roboflow)
  Downloading opencv_python_headless-4.8.0.74-cp37-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (49.1 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 49.1/49.1 MB 14.9 MB/s eta 0:00:00
Requirement already satisfied: Pillow>=7.1.2 in /usr/local/lib/python3.10/dist-packages (from roboflow) (9.4.0)
Requirement already satisfied: python-dateutil in /usr/local/lib/python3.10/dist-packages (from roboflow) (2.8.2)
Collecting python-dotenv (from roboflow)
  Downloading python_dotenv-1.0.1-py3-none-any.whl (19 kB)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from roboflow) (2.31.0)
...
    Found existing installation: certifi 2024.2.2
    Uninstalling certifi-2024.2.2:
      Successfully uninstalled certifi-2024.2.2
Successfully installed certifi-2023.7.22 chardet-4.0.0 cycler-0.10.0 idna-2.10 opencv-python-headless-4.8.0.74 python-dotenv-1.0.1 python-magic-0.4.27 requests-toolbelt-1.0.0 roboflow-
```

Εικόνα 46: Μεταφόρτωση Συνόλου Δεδομένων στο Colab [2]

4.3 Διαδικασία Εκπαίδευσης

Η εκπαίδευση των μοντέλων πραγματοποιήθηκε με την εκτέλεση του σχετικού Python Script με ονομασία train.py, για κάθε μία από τις εκδόσεις του YOLO που επιλέχθηκε.

Κατά την εκτέλεση του train.py, στην είσοδο του ορίζονται οι παρακάτω παράμετροι:

- **Img:** Το μέγεθος των εικόνων εισόδου.
- **Batch size:** Το μέγεθος παρτίδας
- **Epochs:** Τον αριθμό των εποχών της εκπαίδευσης
- **Data:** Η διαδρομή προς το αρχείο data.yaml του συνόλου δεδομένων
- **Weights:** Η διαδρομή προς το αρχείο με τα προ-εκπαιδευμένα βάρη

4.3.1 Εκπαίδευση YOLOv5

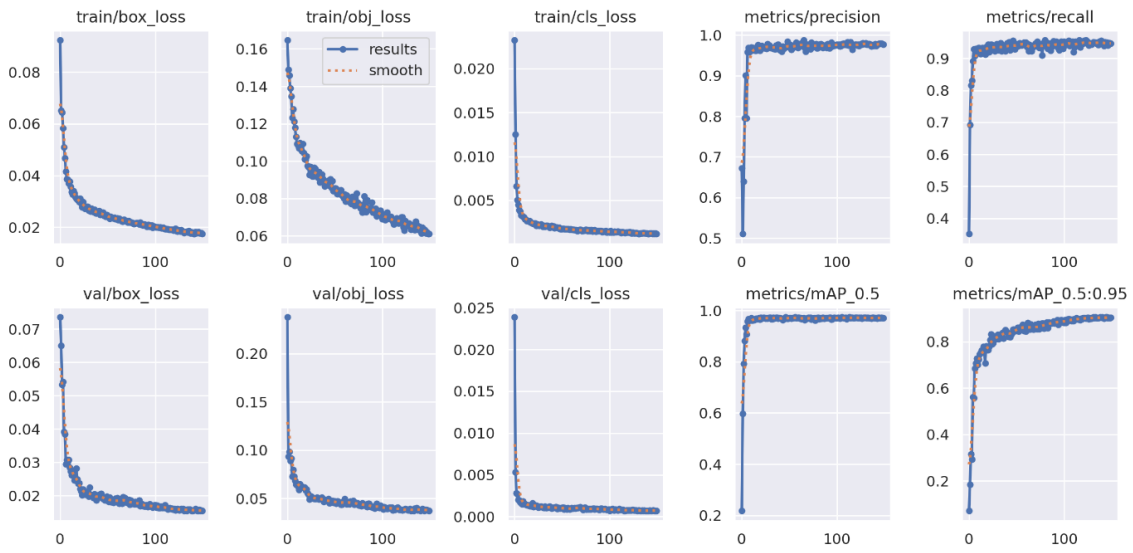
Εντολή για την εκπαίδευση των YOLOv5n & YOLOv5s

```
!python train.py --img 640 --batch 32 --epochs 150 --data {dataset.location}/data.yaml --weights yolov5n.pt --cache
```

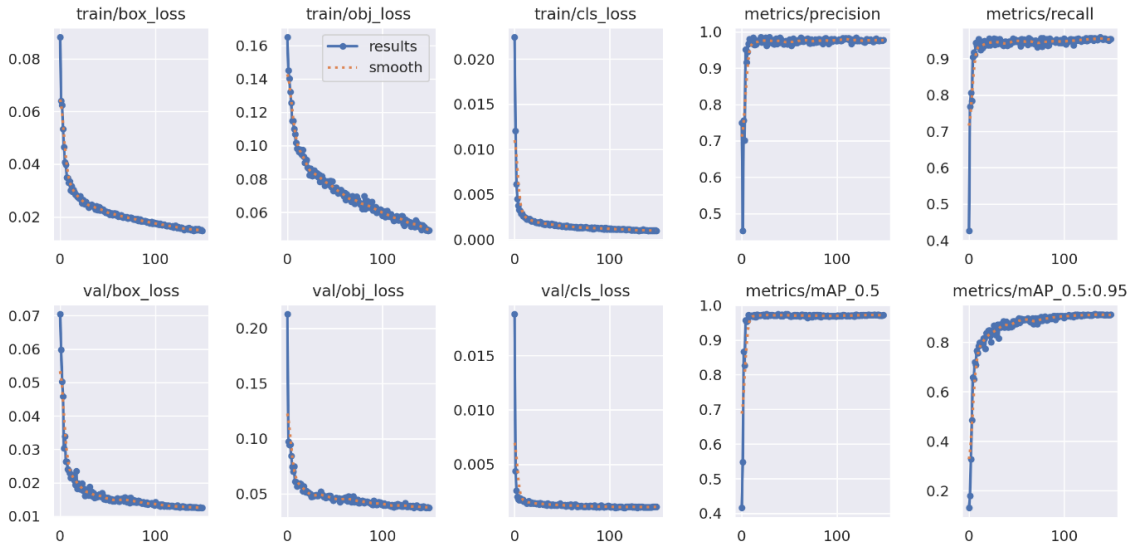
Εικόνα 47: Εκπαίδευση μοντέλων [1]: YOLOv5

4.3.1.1 Αποτελέσματα Εκπαίδευσης YOLOv5

Στις εικόνες 48 και 49 παρουσιάζονται τα μετρικά της εκπαίδευσης των μοντέλων YOLOv5n & YOLOv5s, τα οποία εκπαιδεύτηκαν με τον ίδιο αριθμό εποχών, μέγεθος παρτίδας και σύνολο δεδομένων. Για το YOLOv5n, η ακρίβεια άγγιξε το 97%, η ανάκληση το 94% και το mAP το 97.11%. Συγκριτικά το YOLOv5s παρουσιάζει παρόμοια αποτελέσματα με το YOLO5n, με ακρίβεια 97%, ανάκληση 95% και mAP 97.17%. Είναι αξιοσημείωτο ότι σε κανένα από τα δύο μοντέλα δεν παρατηρήθηκε φαινόμενο της υπερπροσαρμογής (overfitting) μέχρι και την τελευταία εποχή της εκπαίδευσης.

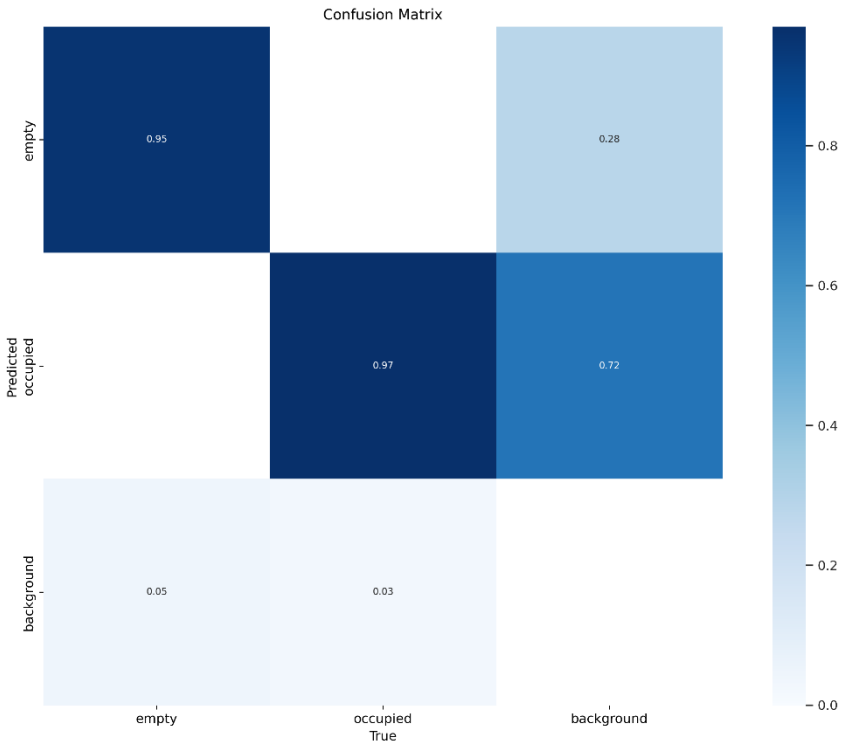


Εικόνα 48: Αποτελέσματα εκπαίδευσης - YOLOv5n [1]

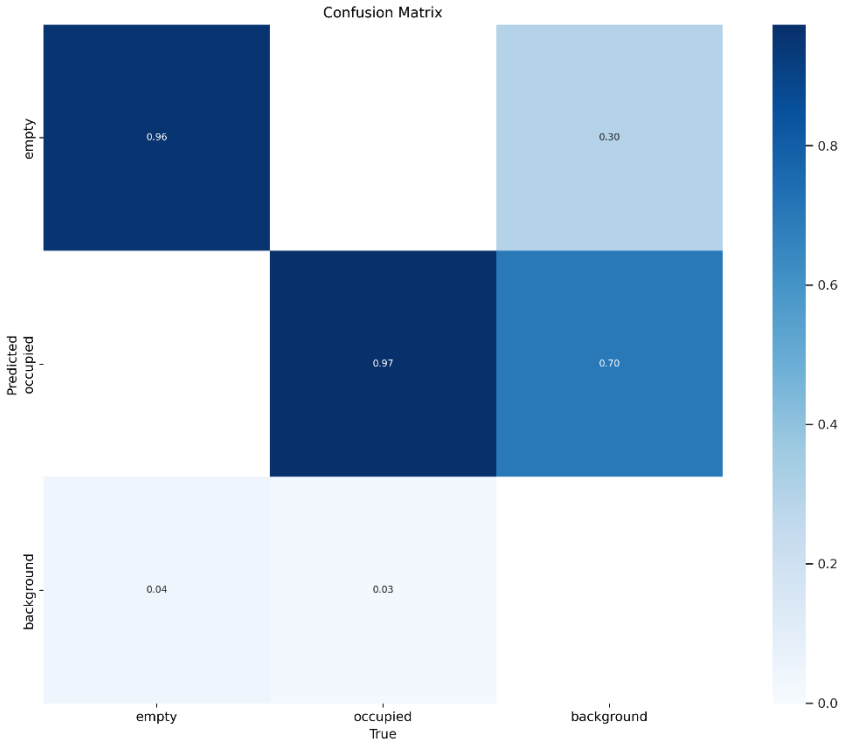


Εικόνα 49: Αποτελέσματα εκπαίδευσης - YOLOv5s [1]

Στις Εικόνες 50 και 51, παρουσιάζονται οι πίνακες σύγχυσης των μοντέλων YOLOv5n & YOLOv5s.

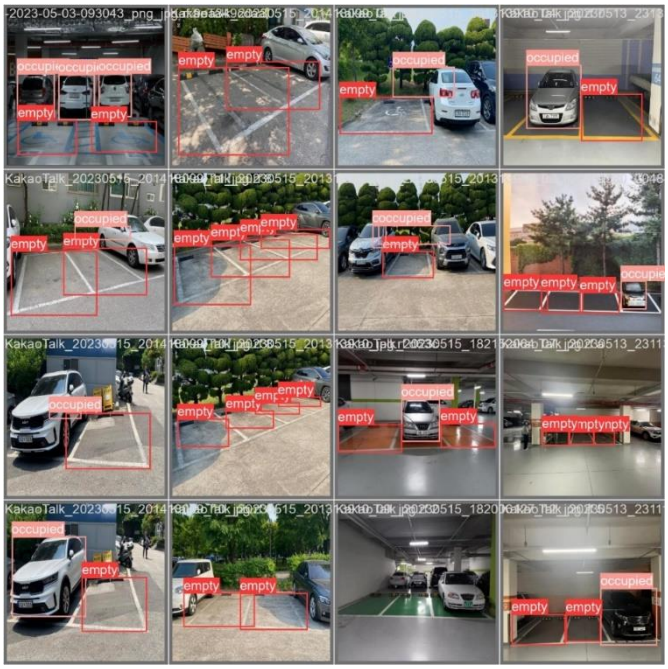


Εικόνα 50: Πίνακας Σύγχυσης - YOLOv5n [2]

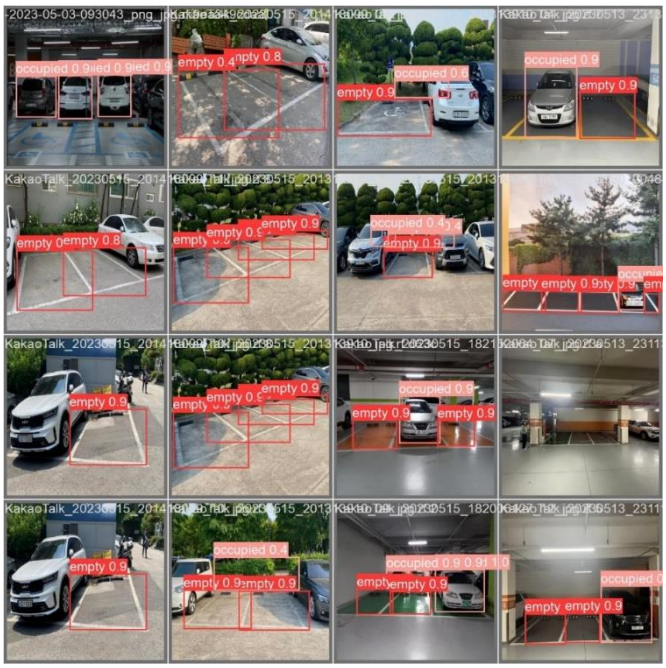


Εικόνα 51: Πίνακας Σύγχυσης - YOLOv5s [2]

Στις Εικόνες 52 & 53, παρουσιάζονται τα αποτελέσματα της ανίχνευσης στις εικόνες του υποσύνολού αξιολόγησης.



Εικόνα 52: Αποτελέσματα ανίχνευσης στο υποσύνολο Αξιολόγησης - YOLOv5n [3]



Εικόνα 53: Αποτελέσματα ανίχνευσης στο υποσύνολο Αξιολόγησης - YOLOv5s [3]

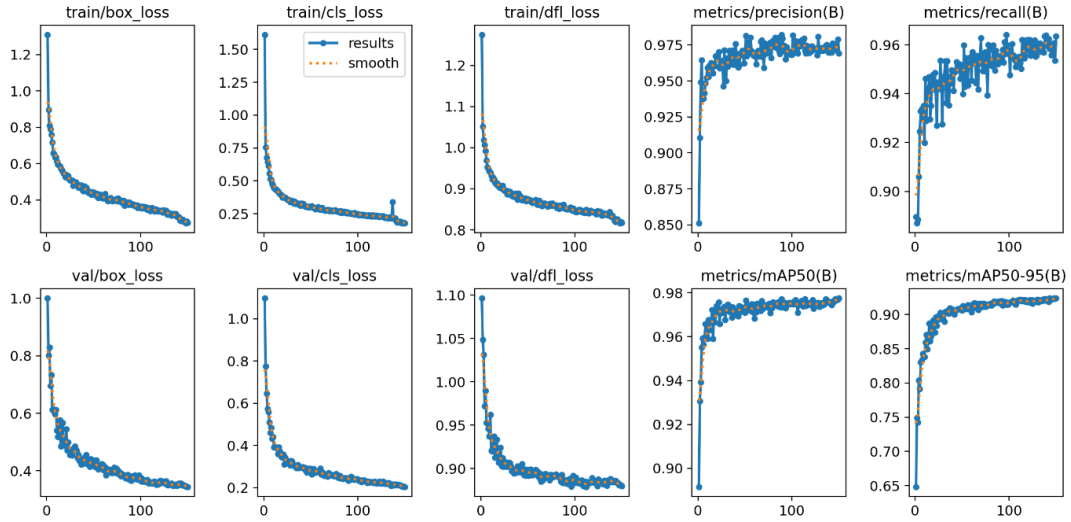
4.3.2 Εκπαίδευση YOLOv8

```
!yolo task=detect mode=train model=yolov8n.pt data=/content/datasets/parking-spot-detector-1/data.yaml epochs=150 imgsz=640 plots=True
```

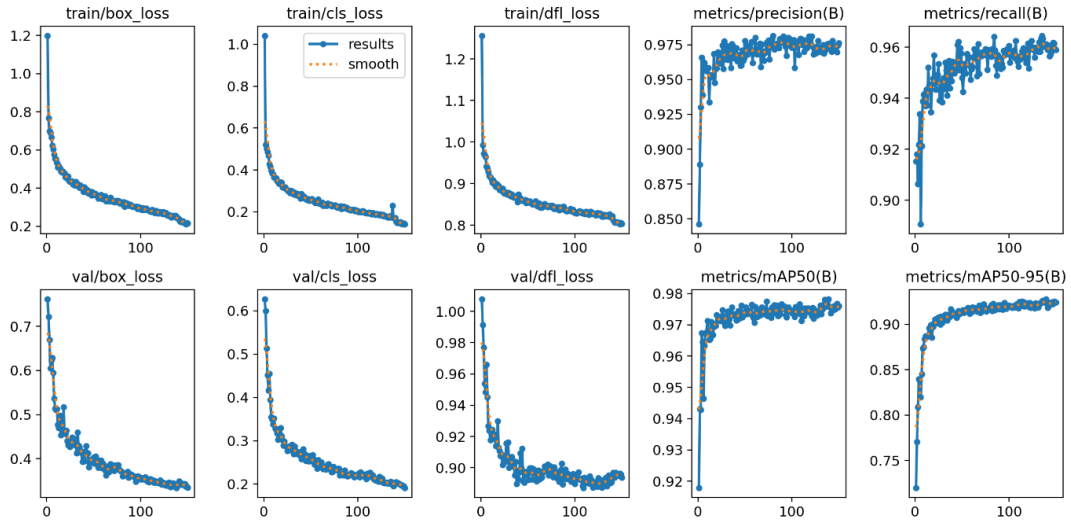
Εικόνα 54: Εκπαίδευση μοντέλων [2]: YOLOv8

4.3.2.1 Αποτελέσματα Εκπαίδευσης YOLOv8

Στις εικόνες 55 και 56 παρουσιάζονται τα μετρικά από την εκπαίδευση των μοντέλων YOLOv8n & YOLOv8s. Το YOLOv8n εμφάνισε ακρίβεια 96.92%, ανάκληση 95.88% και mAP στο 97.75%. Το YOLOv8s, από την άλλη, εμφάνισε ακρίβεια 97.33%, ανάκληση 95.88% και mAP 97.6%.

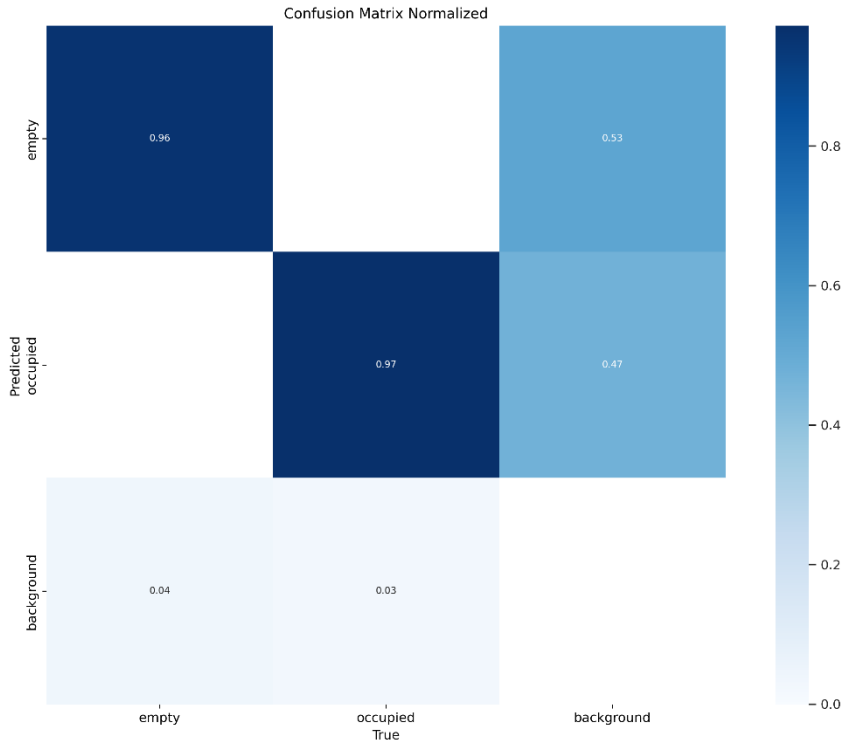


Εικόνα 55: Αποτελέσματα Εκπαίδευσης – YOLOv8n [1]

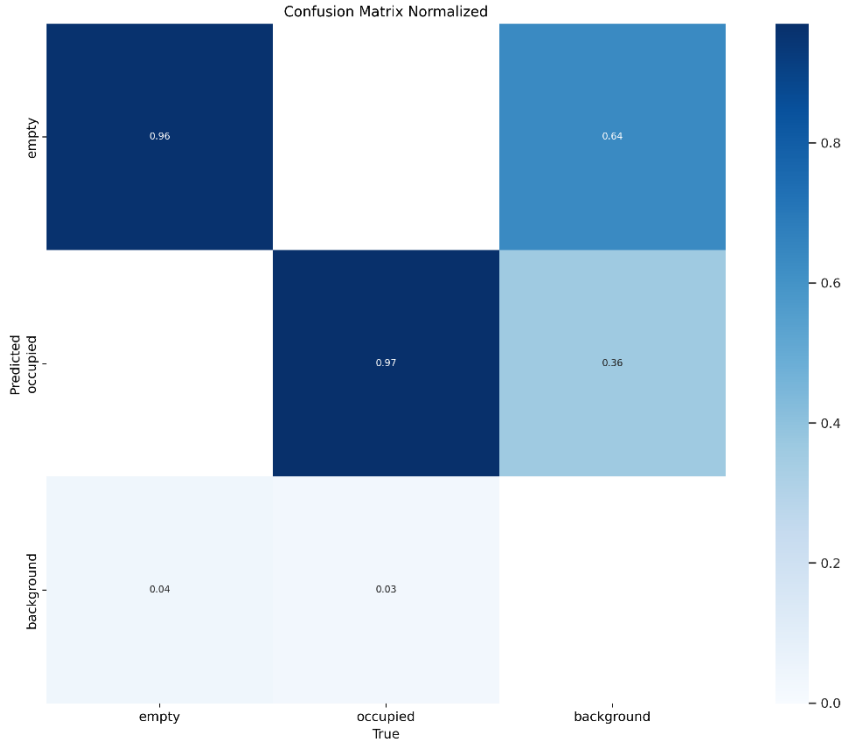


Εικόνα 56: Αποτελέσματα εκπαίδευσης – YOLOv8s [1]

Στις Εικόνες 57 και 58, παρουσιάζονται οι πίνακες σύγχυσης των YOLOv8n & YOLOv8s.

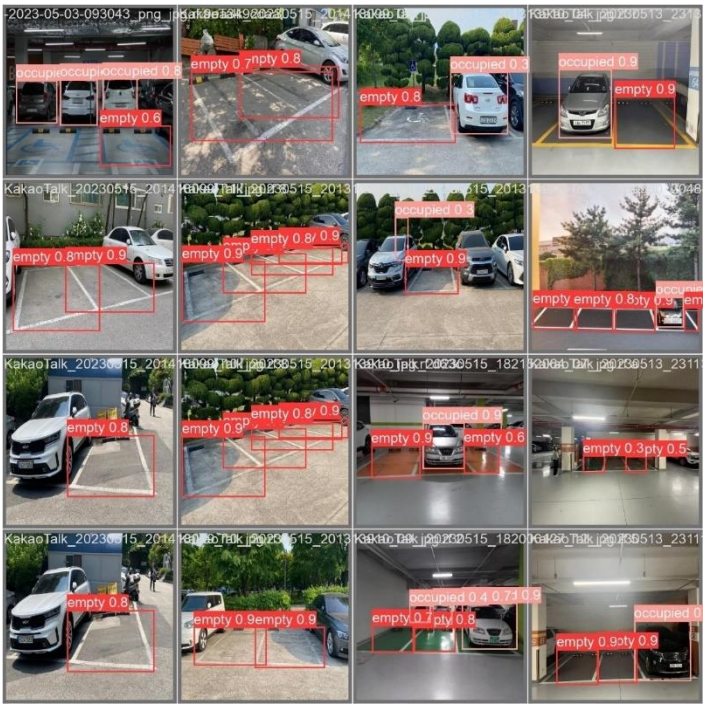


Εικόνα 57: Πίνακας Σύγχυσης - YOLOv8n [2]

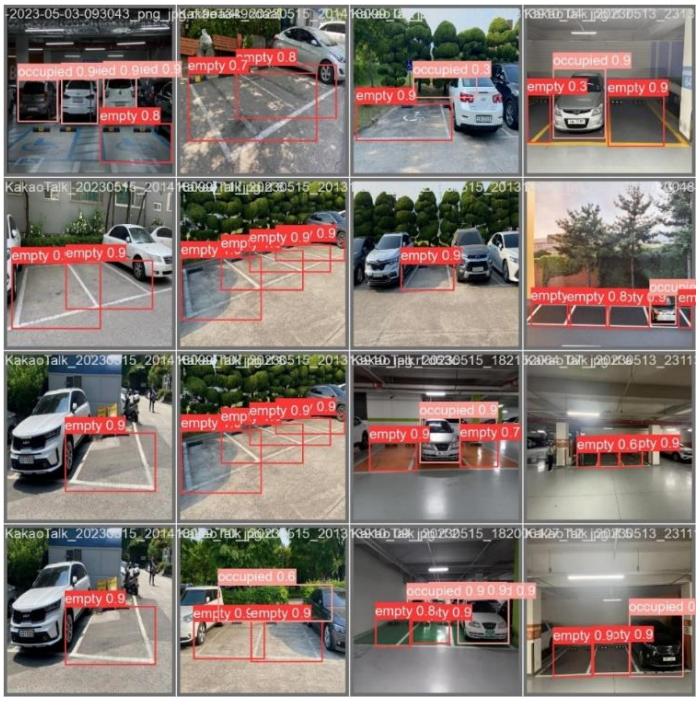


Εικόνα 58: Πίνακας Σύγχυσης - YOLOv8s [2]

Στις Εικόνες 59 & 60, απεικονίζονται τα αποτελέσματα της ανίχνευσης στις εικόνες του υποσύνολού αξιολόγησης.



Εικόνα 59: Αποτελέσματα ανίχνευσης στο υποσύνολο Αξιολόγησης - YOLOv8n [3]



Εικόνα 60: Αποτελέσματα ανίχνευσης στο υποσύνολο Αξιολόγησης - YOLOv8s [3]

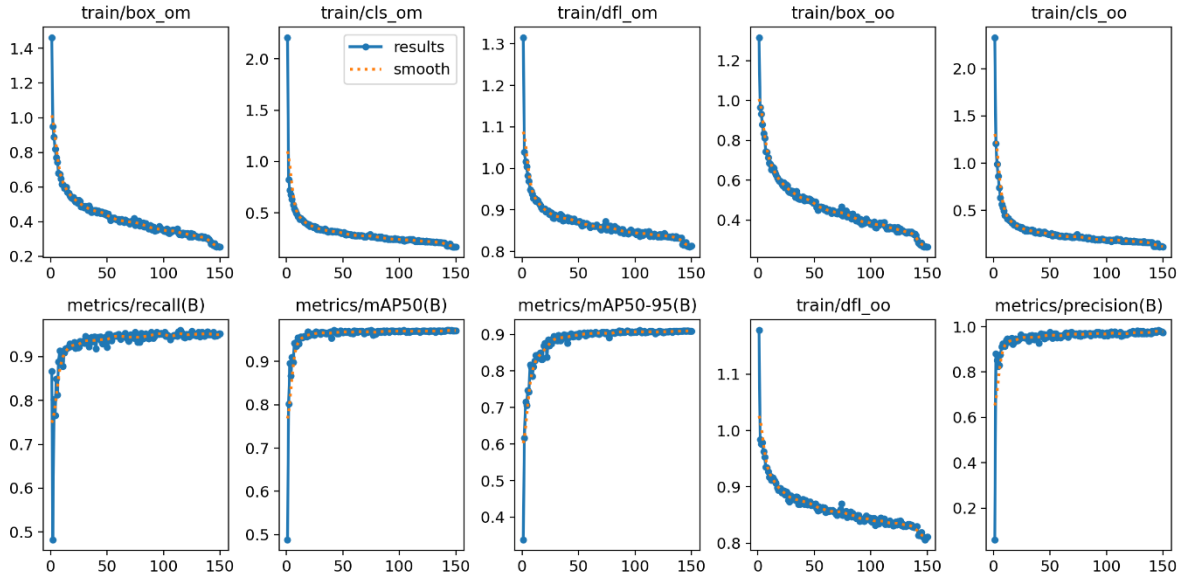
4.3.3 Εκπαίδευση YOLOv10

```
!yolo task=detect mode=train epochs=150 batch=32 plots=True model={HOME}/weights/yolov10s.pt data= /content/parking-spot-detector-1/data.yaml
```

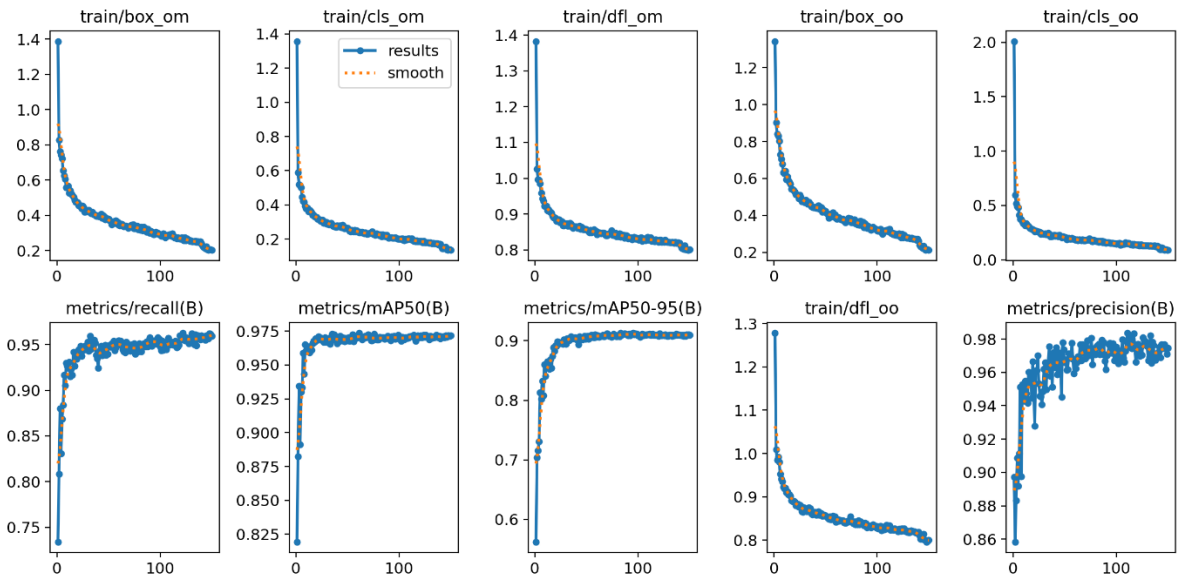
Εικόνα 61: Εκπαίδευση μοντέλων [3]: YOLOv10

4.3.3.1 Αποτελέσματα Εκπαίδευσης YOLOv10

Στις εικόνες 62 και 63, παρουσιάζονται τα μετρικά της εκπαίδευσης των μοντέλων YOLOv10n & YOLOv10s. Στο YOLOv10n, η ακρίβεια άγγιξε 97.3%, ανάκληση 95.2% και mAP 97.08%, χωρίς να παρουσιαστεί overfitting. Το YOLOv10s παρουσίασε παρόμοια αποτελέσματα, με ακρίβεια 97.4%, ανάκληση 95.2% και mAP 97.15%.

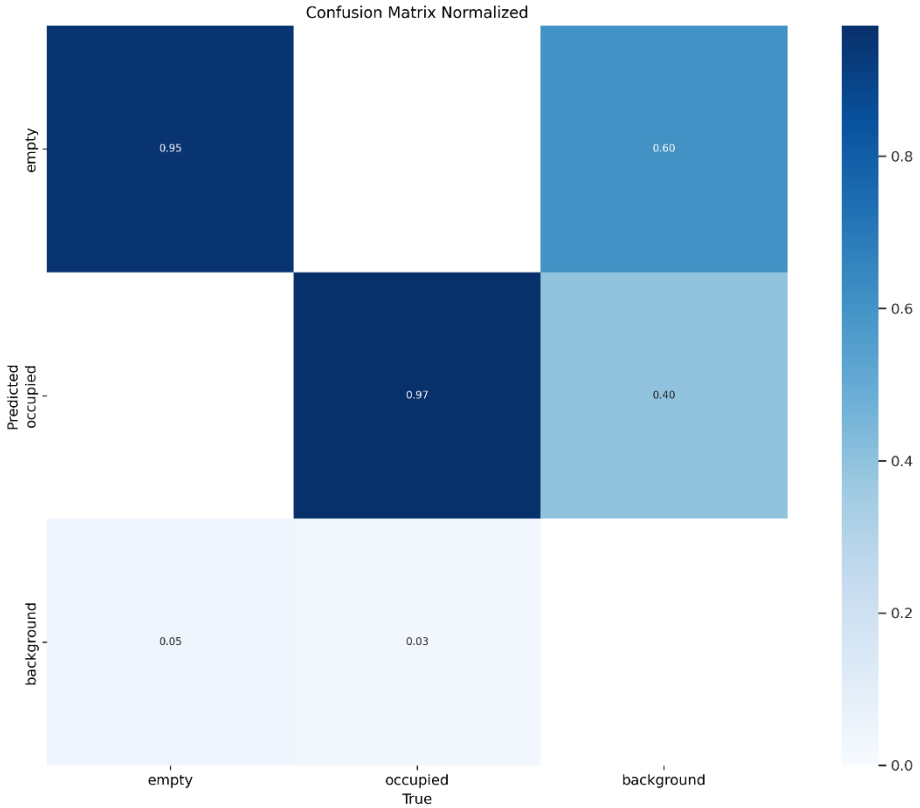


Εικόνα 62: Αποτελέσματα εκπαίδευσης – YOLOv10n [1]

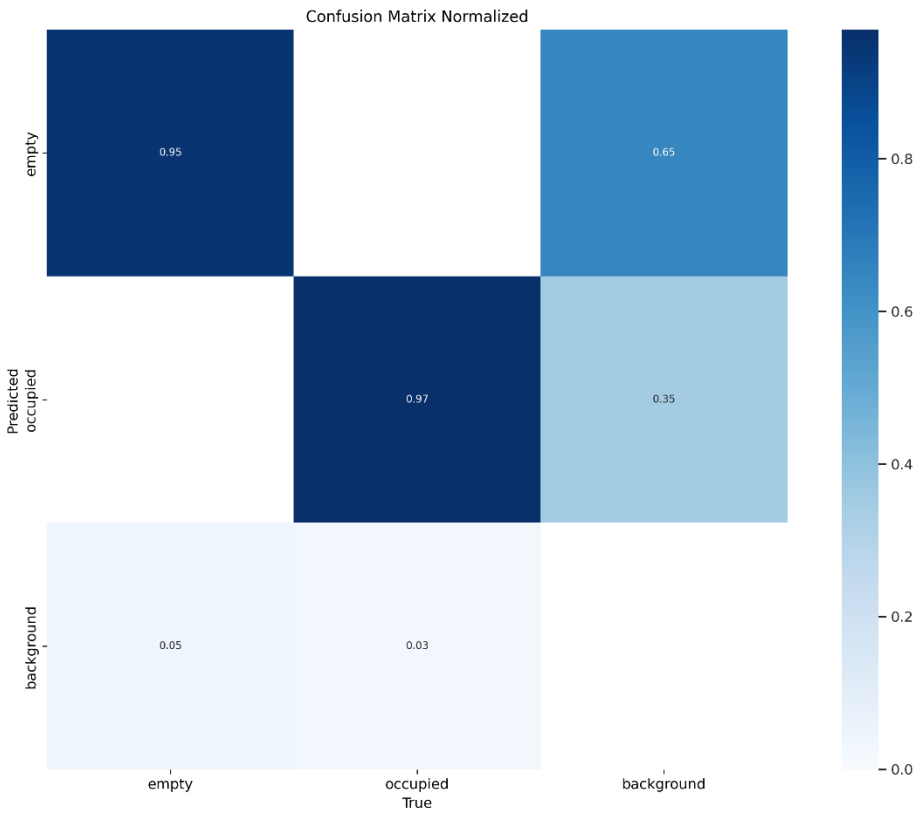


Εικόνα 63: Αποτελέσματα εκπαίδευσης – YOLOv10s [1]

Στις Εικόνες 64 και 65, παρουσιάζονται οι πίνακες σύγχυσης των μοντέλων YOLOv10n & YOLOv10s.

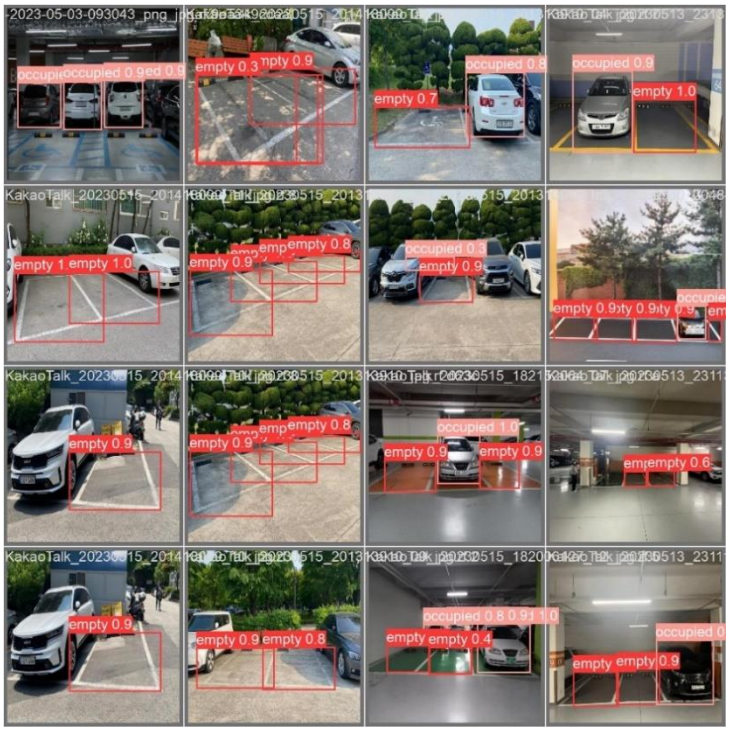


Εικόνα 64: YOLOv10n [3] – Πίνακας Σύγχυσης

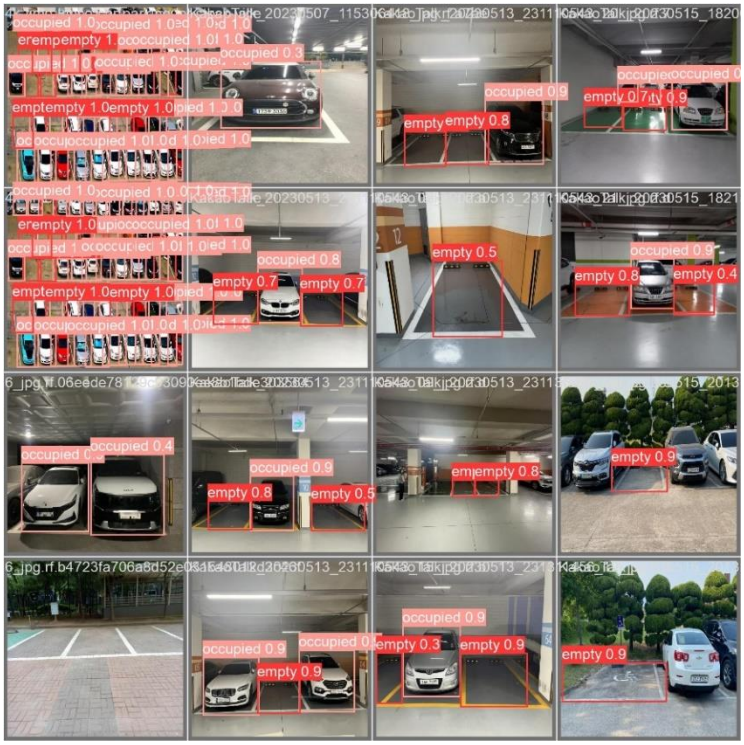


Εικόνα 65: YOLOv10s [2] – Πίνακας Σύγχυσης

Στις Εικόνες 66 και 67 απεικονίζονται τα αποτελέσματα της ανίχνευσης στις εικόνες του υποσύνολού αξιολόγησης.



Εικόνα 66: YOLOv10n [4] – Αποτελέσματα ανίχνευσης στο υποσύνολο Αξιολόγησης



Εικόνα 67: YOLOv10s [3] – Αποτελέσματα ανίχνευσης στο υποσύνολο Αξιολόγησης

4.3.4 Συνοπτικός Πίνακας: Επιδόσεις Εκπαίδευσης

	mAP_50	mAP_0.5:0.95	Precision	Recall	F1-Score
YOLOv5n	0.9711	0.9223	0.9766	0.9465	0.97
YOLOv5s	0.9717	0.9111	0.9763	0.9540	0.97
YOLOv8n	0.9775	0.9218	0.9692	0.9635	0.97
YOLOv8s	0.9760	0.9240	0.9762	0.9588	0.97
YOLOv10n	0.9708	0.9091	0.9733	0.9522	0.96
YOLOv10s	0.9715	0.9100	0.9746	0.9595	0.96

Πίνακας 2: Συνοπτικός Πίνακας: Επιδόσεις Εκπαίδευσης

Στον Πίνακα 2, παρατηρούμε ότι τα μοντέλα και οι παραλλαγές που εκπαιδεύτηκαν με το ίδιο σύνολο δεδομένων παρουσιάζουν πολύ υψηλά και συγκρίσιμα αποτελέσματα.

Η ακρίβεια όπως αναφέρθηκε προηγουμένως, μετρά το ποσοστό των ορθά θετικών προβλέψεων προς το σύνολο των ορθά θετικών προβλέψεων του μοντέλου. Με ακρίβεια δύο δεκαδικών ψηφίων όλα τα μοντέλα εμφανίζουν την ίδια ακρίβεια (Recall). Το ίδιο ισχύει και για τα mAP και F1-Score όπου όλα τα μοντέλα παρουσιάζουν ίδιες τιμές mAP και F1-Score. Την υψηλότερη ανάκληση παρουσιάζει το μοντέλο YOLOv8n με σχετικά μικρή διαφορά από τα άλλα μοντέλα. Τέλος, υπενθυμίζοντας ότι το mAP_0.5:0.95 θεωρείται ως το πιο αξιόπιστο μετρικό αξιολόγησης, καθώς παρέχει μία συνολική εικόνα της απόδοσης του μοντέλου. Το μοντέλο YOLOv8s παρουσίασε το υψηλότερο mAP_0.5:0.95, ξεπερνώντας το YOLO5n κατά 0.2%

Χρόνος	YOLOv5n	YOLOv5s	YOLOv8n	YOLOv8s	YOLOv10n	YOLOv10s
Ώρες	1.397	1.539	1.754	1.891	1.824	1.955

Πίνακας 3: Χρόνος Εκπαίδευσης Μοντέλων

Στον Πίνακα 3, παρουσιάζεται ο χρόνος εκπαίδευσης των μοντέλων. παρατηρούμε ότι ο χρόνος εκπαίδευσης του μοντέλου YOLOv5n ολοκληρώθηκε σε συντομότερο χρονικό διάστημα από τα υπόλοιπα μοντέλα. Τον υψηλότερο χρόνο εκπαίδευσης παρουσίασε το μοντέλο YOLOv10s.

Size	YOLOv5n	YOLOv8n	YOLOv10n	YOLOv5s	YOLOv8s	YOLOv10s
MB	3.73	5.97	5.5	13.7	20.22	14.78

Πίνακας 4: Συνοπτικός Πίνακας: Μέγεθος των μοντέλων

Στον Πίνακα 4, παρατηρούμε ότι το μοντέλο YOLOv5n διαθέτει το μικρότερο μέγεθος σε σχέση με όλα τα υπόλοιπα που εκπαιδεύτηκαν. Επιπλέον παρατηρούμε ότι τα μοντέλα των παραλλαγών Nano, τα οποία διαθέτουν λιγότερες παραμέτρους, παρουσιάζουν μικρότερο μέγεθος σε σύγκριση με τα μοντέλα Small.

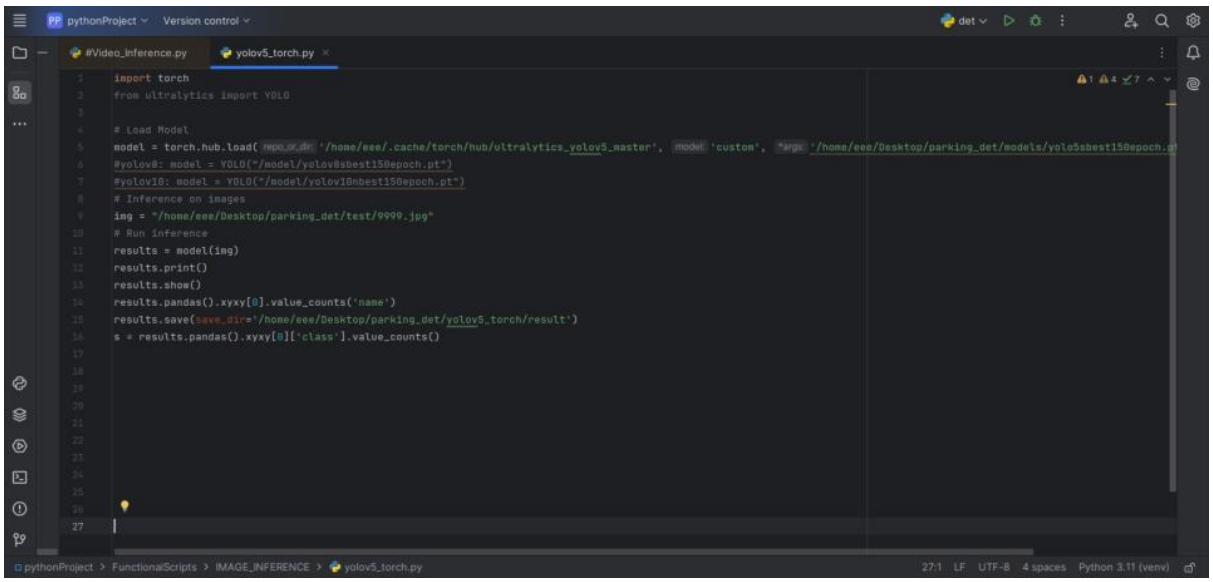
Κεφάλαιο 5: Εφαρμογή Συστήματος

Στο παρόν κεφάλαιο, θα πραγματοποιηθούν δοκιμές στα μοντέλα που εκπαιδεύτηκαν στο προηγούμενο κεφάλαιο, με στόχο την επιλογή του μοντέλου με την υψηλότερη ακρίβεια και τις καλύτερες επιδόσεις. Θα μελετηθούν δύο περιπτώσεις, της ανίχνευσης αντικειμένων σε εικόνα και της ανίχνευσης αντικειμένων σε βίντεο.

Συγκρίνοντας τα αποτελέσματα από τις δύο αυτές περιπτώσεις, θα επιλεχθεί το μοντέλου που παρουσιάζει τις βέλτιστες επιδόσεις και θα χρησιμοποιηθεί για την ανίχνευση αντικειμένων σε πραγματικό χρόνο.

5.1 Εξαγωγή Συμπερασμάτων από Εικόνες (Image Inference)

Στην περίπτωση αυτή, τα μοντέλα καλούνται να ανιχνεύσουν τις διαθέσιμες θέσεις στάθμευσης σε μία δοσμένη εικόνα.



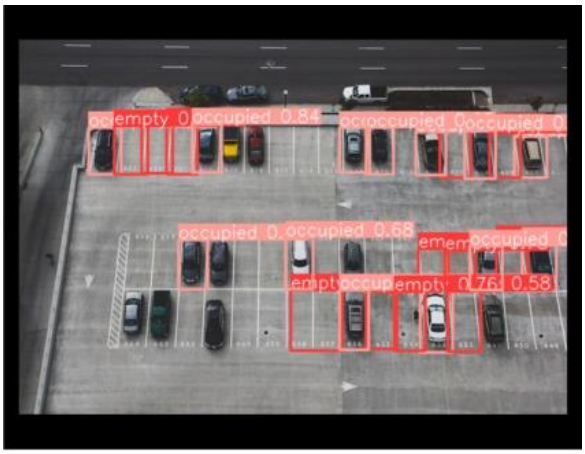
Εικόνα 68: Python Script: Imageinference.py

Η διαδικασία εξαγωγής συμπερασμάτων πραγματοποιείται μέσω του αρχείου Imageinference.py. Μετά την ολοκλήρωση της διαδικασίας, τα αποτελέσματα προβάλλονται.

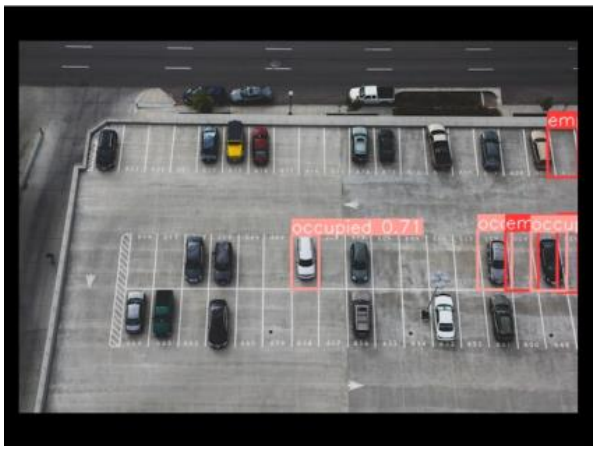
Στις Εικόνες 69 έως 74 που ακολουθούν, απεικονίζονται τα αποτελέσματα της εξαγωγής συμπερασμάτων για κάθε μοντέλο που εκπαιδεύτηκε, σε μία εικόνα εισόδου.

Εικόνα 69:YOLOv5n-[1] Image Inference

Εικόνα 70:YOLOv5s-[1] Image Inference

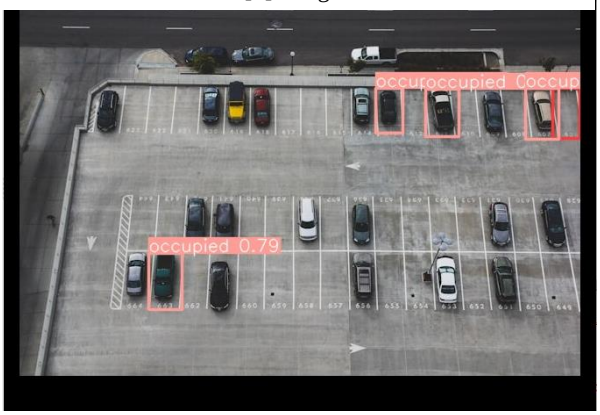


Εικόνα 71: YOLOv8n-[1] Image Inference

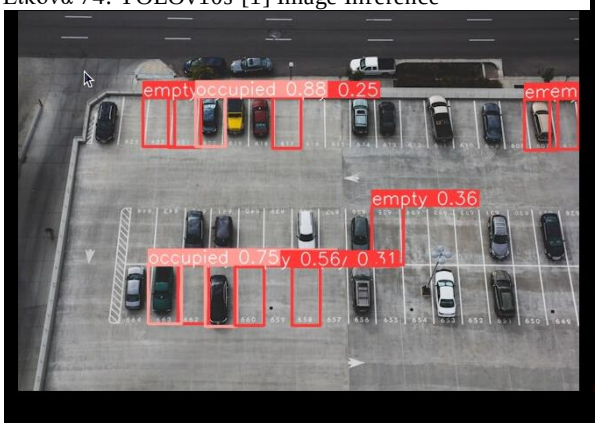


Εικόνα 72: YOLOv8s-[1] Image Inference

Εικόνα 73: YOLOv10n-[1] Image Inference



Εικόνα 74: YOLOv10s-[1] Image Inference



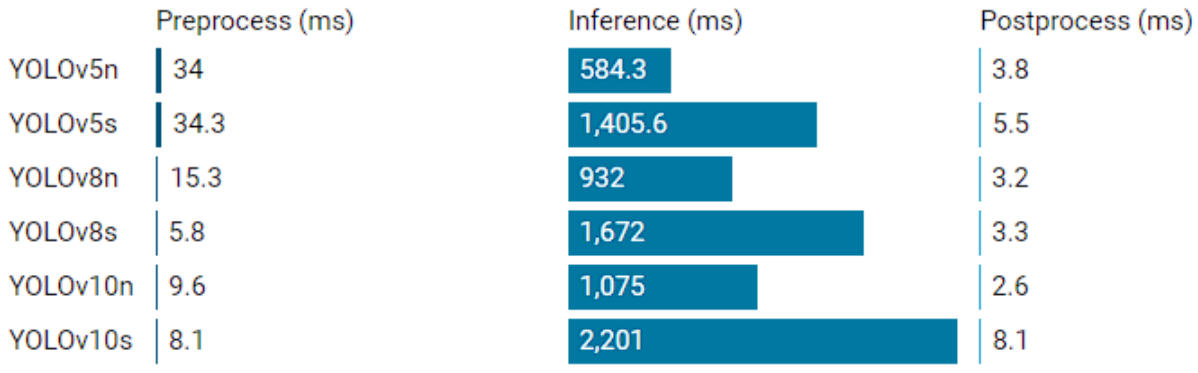
~	YOLOv5n	YOLOv5s	YOLOv8n	YOLOv8s	YOLOv10n	YOLOv10s
Postprocess (ms)	3.8	5.5	3.2	3.3	2.6	8.1
Inference (ms)	584.3	1405.6	932	1672	1075	2201
Preprocess (ms)	34	34.3	15.3	5.8	9.6	8.1
Occupied/Empty	32/24	26/16	13/12	3/3	4/1	3/9

Πίνακας 5: Συγκεντρωτικός Πίνακας Μετρικών [1] – Image Inference

Από τον παραπάνω πίνακα συμπεραίνουμε ότι:

- Χρόνος συμπεράσματος: Ο χρόνος που απαιτείται για την ολοκλήρωση της εξαγωγής των συμπερασμάτων στα μοντέλα Small είναι αρκετά υψηλότερος, αγγίζοντας σχεδόν το διπλάσιο σε σχέση με τα μοντέλα Nano.

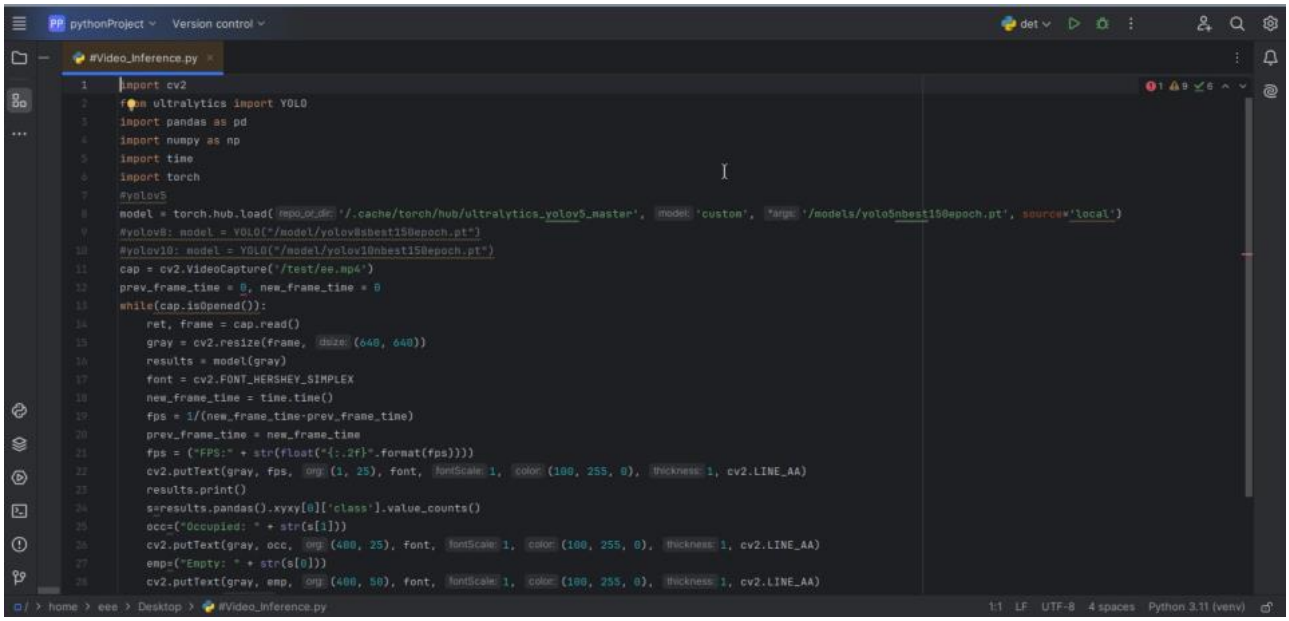
- Ανίχνευση θέσεων στάθμευσης: Στην περίπτωση της εξαγωγής συμπερασμάτων από εικόνες, το μοντέλο YOLOv5n κατάφερε να ανιχνεύσει τις περισσότερες θέσεις στάθμευσης σε σχέση με τα υπόλοιπα μοντέλα. Επιπλέον, η διαδικασία του της εξαγωγής συμπερασμάτων ολοκληρώθηκε σε μικρότερο χρονικό διάστημα συγκριτικά με τα άλλα μοντέλα.



Εικόνα 75: Οπτικοποίηση Αποτελεσμάτων: Εξαγωγή Συμπερασμάτων από Εικόνες (Image Inference)

5.2 Εξαγωγή Συμπερασμάτων από Βίντεο (Video Inference)

Σε αυτή την υποενότητα, θα δοκιμάσουμε τα εκπαιδευμένα μοντέλα χρησιμοποιώντας ένα προγεγραμμένο βίντεο που απεικονίζει έναν χώρο στάθμευσης.



Εικόνα 76: Python Script: Videoinference.py

Η διαδικασία της εξαγωγής συμπερασμάτων πραγματοποιείται μέσω του Videoinference.py.

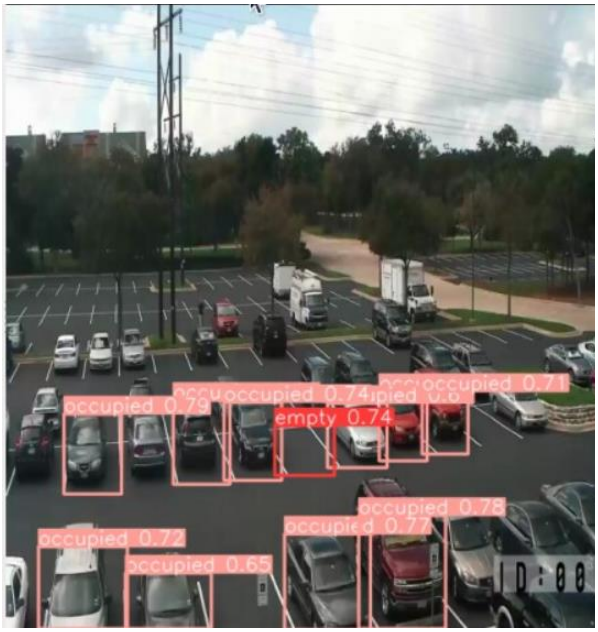
Στις Εικόνες 77 έως 82 που ακολουθούν, παρουσιάζονται τα αποτελέσματα ανίχνευσης κενών θέσεων στάθμευσης για κάθε ένα από τα μοντέλα που εκπαιδεύτηκαν. Όπως φαίνεται από τις εικόνες που ακολουθούν η κάμερα μέσω της οποίας πραγματοποιήθηκε η αρχική λήψη του βίντεο, είναι τοποθετημένη με τρόπο ώστε να επιτηρεί έναν χώρο στάθμευσης με σχετικά μεγάλη έκταση και περίπου 180 θέσεις στάθμευσης. Από τα παρακάτω αποτελέσματα παρατηρούμε ότι τα μοντέλα, ανεξάρτητα από το σύνολο των θέσεων στάθμευσης που ανίχνευσαν, αδυνατούν να εντοπίσουν τις κενές θέσεις που βρίσκονται στο βάθος της ροής. Ένας πιθανός λόγος της αδυναμίας αναγνώρισης, ενδέχεται να προέρχεται από το σύνολο δεδομένων που επιλέχθηκε για την εκπαίδευση των μοντέλων. Ακόμη ένας λόγω είναι το μικρό μέγεθος των θέσεων στάθμευσης συναρτήσει με τις μπροστινές θέσεις της εικόνας, πράγμα που επηρεάζει σημαντικά την απόδοση των μοντέλων.

Εικόνα 77: YOLOv5n [2] Video Inference

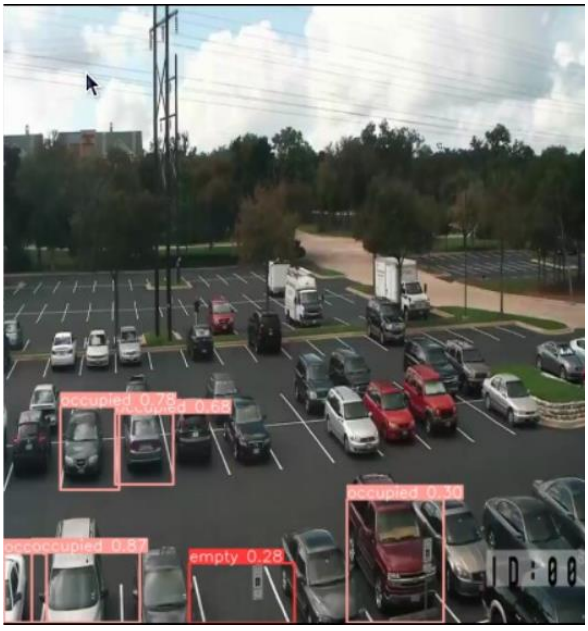
Εικόνα 78: YOLOv5s [2] Video Inference

Εικόνα 79: YOLOv8n-[2] Video Inference

Εικόνα 80: YOLOv8s-[2] Video Inference



Εικόνα 81: YOLOv10n-[2] Video Inference

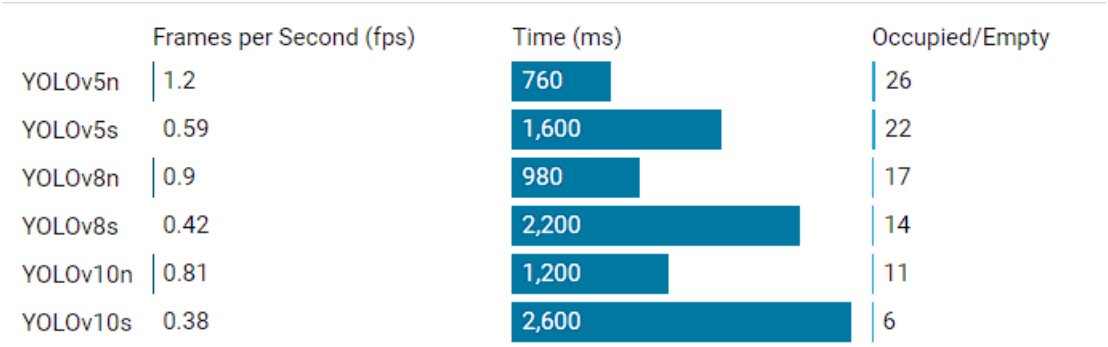


Εικόνα 82: YOLOv10s-[2] Video Inference

~	YOLOv5n	YOLOv5s	YOLOv8n	YOLOv8s	YOLOv10n	YOLOv10s
Frames per Second (fps)	1.2	0.59	0.9	0.42	0.81	0.38
Time (ms)	760	1600	980	2200	1200	2600
Occupied/Empty	15-11	17-5	14-3	12-2	10-1	5-1

Πίνακας 6: Συγκεντρωτικός Πίνακας Μετρικών [2] – Video Inference

Με βάση τα αποτελέσματα των Πινάκων 3 έως 6 και λαμβάνοντας υπόψη ότι το επιλεγμένο μοντέλο θα χρησιμοποιηθεί σε εφαρμογή πραγματικού χρόνου για την εξαγωγή συμπερασμάτων σε μικροϋπολογιστή με περιορισμένη υπολογιστική ισχύ, επιλέχθηκε το YOLOv5n. Το YOLOv5n παρουσιάζει τον ταχύτερο χρόνο εξαγωγής συμπερασμάτων, ξεπερνώντας τα υπόλοιπα μοντέλα. Επιπλέον, αναγνώρισε συνολικά τις περισσότερες θέσεις στάθμευσης, σε όλες τις δοκιμές που διεξήχθησαν σε εικόνες, αναπαραγωγής βίντεο και ζωντανή ροή μέσω κάμερας.



Εικόνα 83: Οπτικοποίηση Αποτελεσμάτων: Εξαγωγή Συμπερασμάτων από Βίντεο (Video Inference)

5.3 Εξαγωγή Συμπερασμάτων πραγματικού χρόνου (Real-Time Inference)

Έχοντας επιλέξει το μοντέλο YOLOv5n ως το πλέον κατάλληλο για χρήση με τη συσκευή, στη συνέχεια θα εφαρμοστεί το μοντέλο και η οπτικοποίηση των δεδομένων σχετικών με τη διαθεσιμότητα θέσεων στάθμευσης για έως και δύο χώρους στάθμευσης.

5.3.1 Ανασκόπηση επιδόσεων

Στο κεφάλαιο 3.4.4, δημιουργήθηκε ένα Dashboard, για την οπτικοποίηση των αποτελεσμάτων της ανίχνευσης σε έναν μόνο χώρο στάθμευσης. Στην Εικόνα 34, απεικονίζεται η ροή του Node-Red, ενώ στην Εικόνα 36 παρουσιάζεται το Dashboard.

Επιπλέον, στον πίνακα 5, η ταχύτητα του YOLOv5n άγγιξε τα 1.2 καρέ ανά δευτερόλεπτο (fps).

Η εκτέλεση μίας δεύτερης ροής παράλληλα διπλασίασε τον χρόνο εξαγωγής συμπερασμάτων, επηρεάζοντας αρνητικά τις επιδόσεις του συστήματος. Ως αποτέλεσμα, τα καρέ ανά δευτερόλεπτο μειώθηκαν σε περίπου 0.60 fps για κάθε ροή.

5.3.2 Βελτίωση επιδόσεων

Λαμβάνοντας υπόψη τα παραπάνω και με στόχο την ομαλή λειτουργία του συστήματος κατά την ταυτόχρονη εξαγωγή συμπερασμάτων σε δύο ροές δεδομένων, κρίθηκε απαραίτητη η βελτίωση της απόδοσης του μοντέλου, αντί για την αντικατάσταση του Raspberry Pi 4 με άλλο πιο υπολογιστικά ισχυρότερο μικροϋπολογιστή.

Υπάρχουν διάφορες προσεγγίσεις που συμβάλλουν στην βελτίωση της απόδοσης του μοντέλου, μειώνοντας ταυτόχρονα τον χρόνο εξαγωγής συμπερασμάτων, όπως αναφέρονται στο [50]:

1. **Εξαγωγή Μοντέλου (Model Export):** Εξαγωγή του μοντέλου σε διαφορετική μορφή, βελτιστοποιημένη για χρήση με συστήματα που διαθέτουν περιορισμένη υπολογιστική ισχύ.

2. **Κλάδεμα Μοντέλου (Model Pruning):** Αφαιρώντας μη απαραίτητους νευρώνες και παραμέτρους από το δίκτυο, το μέγεθος του μοντέλου μειώνεται, οδηγώντας σε ταχύτερους υπολογισμούς. Το κλάδεμα του μοντέλου μπορεί να πραγματοποιηθεί πριν και μετά την εκπαίδευση.

3. **Κβαντισμός Μοντέλου (Quantization):** Η ακρίβεια των βαρών του μοντέλου μειώνεται από υψηλής ακρίβειας των 32bit σε χαμηλής ακρίβειας των 8 bit, οδηγώντας σε γρηγορότερους υπολογισμούς.

Στο πλαίσιο της παρούσας εργασίας, ακολουθήθηκε η πρώτη προσέγγιση, της μετατροπή του μοντέλου PyTorch στην ανοιχτή μορφή ONNX [51].

5.3.2.1 Μετατροπή του PyTorch μοντέλου σε Onnx

Προϋπόθεση για την μετατροπή του μοντέλου είναι η πρόσβαση στο σημειωματάριο Jupyter που χρησιμοποιήθηκε για την εκπαίδευση του μοντέλου.

Στη συνέχεια, απαιτείται η εκτέλεση της εντολής:

```
!python /content/yolov5/export.py --weights /content/yolov5/models/yolov5nbest.pt --include onnx
```

Εικόνα 84: Μετατροπή PyTorch σε Onnx [1]

```
export: data=data/coco128.yaml, weights=['/content/yolov5/models/yolov5nbest.pt'], imgsz=[640, 640], batch_size=1, device=cpu, half=False, inplace=False, keras=False, c
YOLOv5 🚀 v7.0-338-gff063284 Python-3.10.12 torch-2.3.0+cu121 CPU

Fusing layers...
Model summary: 157 layers, 1761871 parameters, 0 gradients, 4.1 GFLOPs

PyTorch: starting from /content/yolov5/models/yolov5nbest.pt with output shape (1, 25200, 7) (3.7 MB)

ONNX: starting export with onnx 1.16.1...
ONNX: export success ✅ 0.8s, saved as /content/yolov5/models/yolov5nbest.onnx (7.1 MB)

Export complete (1.2s)
Results saved to /content/yolov5/models
Detect:          python detect.py --weights /content/yolov5/models/yolov5nbest.onnx
Validate:        python val.py --weights /content/yolov5/models/yolov5nbest.onnx
PyTorch Hub:     model = torch.hub.load('ultralytics/yolov5', 'custom', '/content/yolov5/models/yolov5nbest.onnx')
```

Εικόνα 85: Μετατροπή PyTorch σε Onnx [2]

~	Frames per Second (fps)	Inference Time (ms)	Occupied/Empty
PyTorch	1.2	760ms	15/11
Onnx	2.8	280ms	15/11

Πίνακας 7: Αποτελέσματα μετατροπής PyTorch σε Onnx

Όπως φαίνεται στον Πίνακα 7, η μετατροπή του μοντέλου σε μορφή Onnx, μείωσε τον χρόνο συμπερασμάτων κατά περίπου 36%, χωρίς να επηρεάσει τον αριθμό των αναγνωριζόμενων συνολικά θέσεων.

Στη συνέχεια, το Onnx μοντέλο εφαρμόστηκε στην ανίχνευση πραγματικού χρόνου για δύο ταυτόχρονες ροές και τα αποτελέσματα στον Πίνακα 8.

PyTorch				
~	Frames per Second (fps)	Inference Time (ms)	NMS (ms)	Occupied/Empty
Ροή 1 [52]	0.58	1600	6.7	15/11
Ροή 2 [53]	0.59	1673	5.0	1/9
Onnx				
~	Frames per Second (fps)	Inference Time (ms)	NMS (ms)	Occupied/Empty
Ροή 1 [52]	1.25	799	10.6	15/11
Ροή 2 [53]	1.33	750	9.5	1/9

Πίνακας 8: Αποτελέσματα εφαρμογής του μοντέλου YOLOv5n ONNX σε διπλή ροή.

5.4 Οπτικοποίηση αποτελεσμάτων ροής

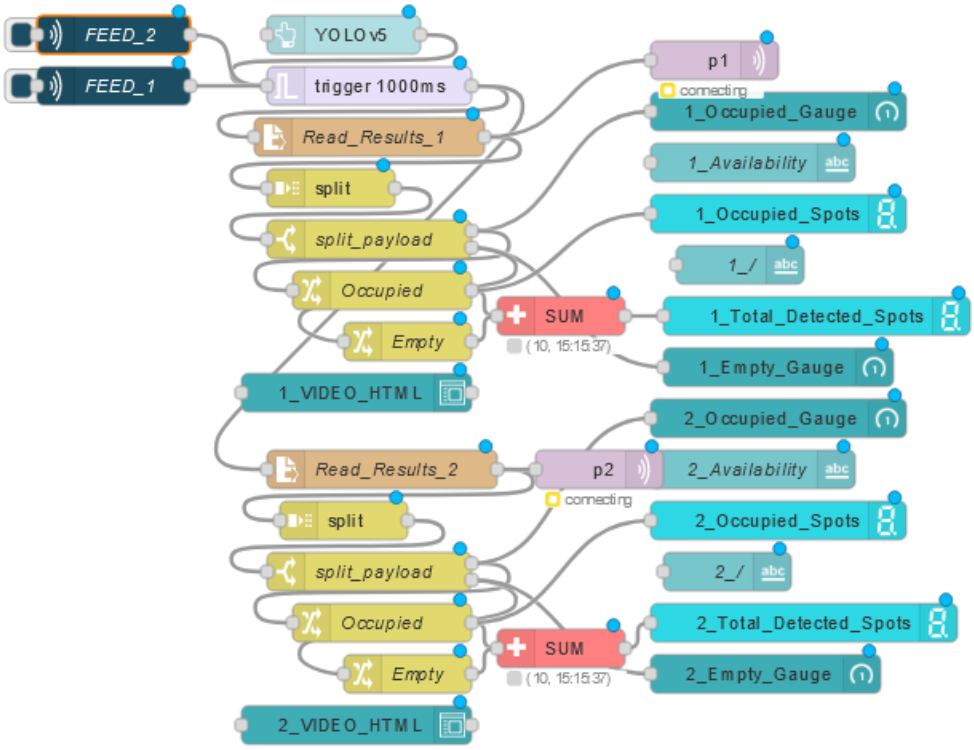
5.4.1 Node-Red Dashboard

Στην παρούσα υποενότητα θα τροποποιήσουμε:

- Την ροή Node-Red του Κεφαλαίου 3.4.4
- Το αρχείο εξαγωγής συμπερασμάτων βίντεο

Στόχος των τροποποιήσεων είναι η ταυτόχρονη, ζωντανή εξαγωγή αποτελεσμάτων και η οπτικοποίηση των δεδομένων που αφορούν τη διαθεσιμότητα των δύο χώρων στάθμευσης.

5.4.1.1 Τροποποίηση Dashboard



Εικόνα 86: Node-Red Flow – Dashboard 2 ροών

Στην Εικόνα 86, παρουσιάζεται η τελική μορφή της ροής Node-Red. Πραγματοποιήθηκε αντιγραφή και επικόλληση των κόμβων της Εικόνας 34. Τα νέα widgets ανατέθηκαν σε μία νέα ομάδα (Group) του Layout με όνομα "Parking Lot 2". Η ομάδα δημιουργήθηκε με βάση τα βήματα του Κεφαλαίου 4.3.

5.4.1.2 Τροποποιήσεις στα Python Scripts

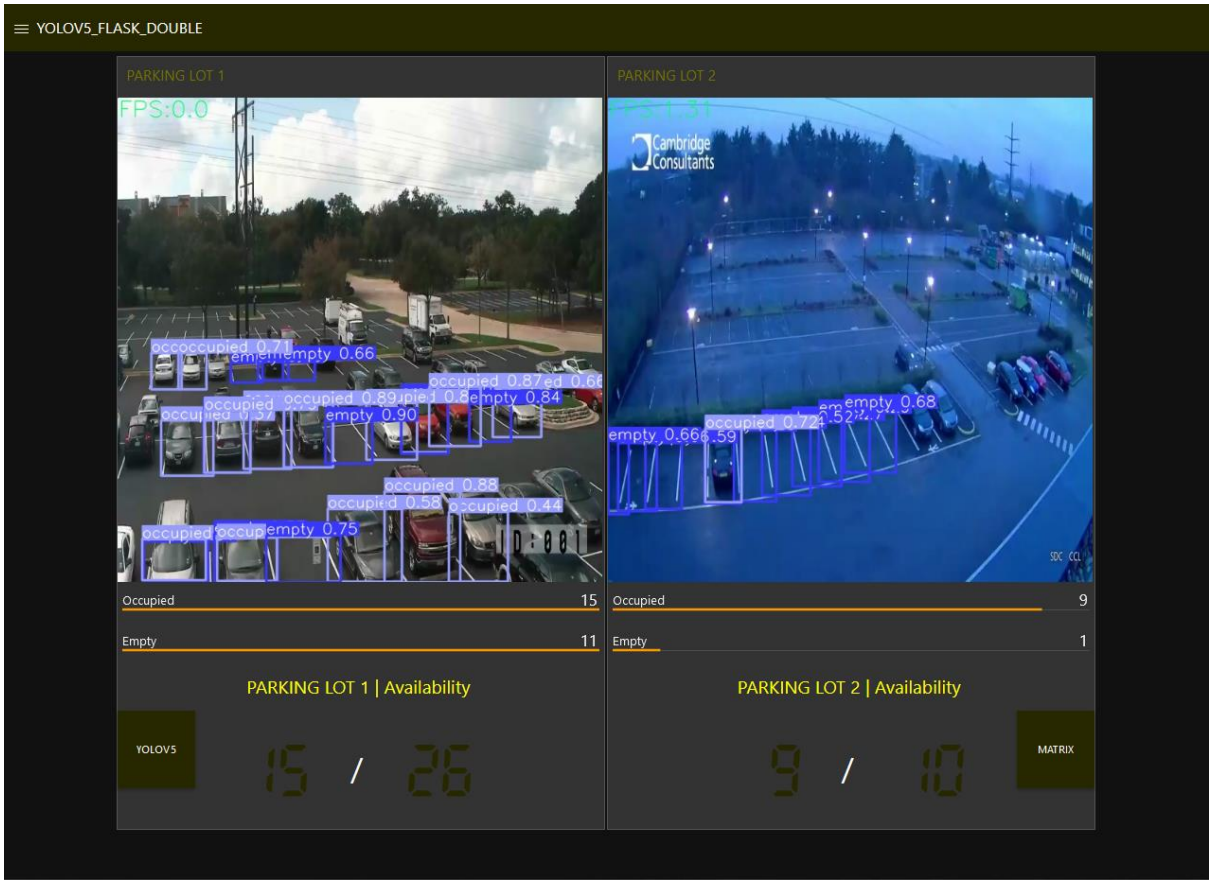
scratch.py

```python
import cv2 import time import torch
from flask import Flask, Response
model = torch.hub.load(/parking_det/models/yolo5nbest150epoch.onnx', source='local')
app = Flask(__name__)
video = cv2.VideoCapture('/parking_det/test/ee.mp4')
def detect(frame):
    frame = cv2.resize(frame, dsize: (640, 640))
    results = model(frame)
    results.render()
    s = results.pandas().xyxy[0]['class'].value_counts()  # show
    print(s)  # class 0 = empty, class 1 = occupied
    return frame
def gen(video):
        success, frame = video.read()
        frame = detect(frame)
        ret, jpeg = cv2.imencode( ext: '.jpg', frame)
        frame = jpeg.tobytes()
        yield (b'--frame\r\n'
               b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n\r\n')
@app.route('/')
def index():
    global video
    return Response(gen(video),
                    mimetype='multipart/x-mixed-replace; boundary=frame')
if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000, threaded=True)
```

Εικόνα 87: Python Script: YOLOv5Flask.py

Η ταυτόχρονη εξαγωγή συμπερασμάτων από δύο ροές πραγματοποιείται μέσω του αρχείου YOLOv5Flask.py, το οποίο βασίζεται σε μία τροποποιημένη έκδοση του Videoinference.py. Αρχικά λαμβάνεται μία ροή βίντεο. Έπειτα πραγματοποιείται εξαγωγή συμπερασμάτων από την ροή χρησιμοποιώντας το επιλεγμένο YOLO. Τέλος μέσω του Flask, αναμεταδίδεται η επεξεργασμένη ροή.

Για τη διπλή μετάδοση βίντεο ή ροών από κάμερες, απαιτείται η χρήση ενός δεύτερου πανομοιότυπου αρχείου που να περιλαμβάνει τη πηγή της δεύτερης ροής και διαφορετική θύρα (port) μετάδοσης.



Εικόνα 88: Node Red - Dashboard Διπλή Ροή

Στην παραπάνω εικόνα απεικονίζεται το τελικό Dashboard της υλοποίησης, το οποίο περιλαμβάνει δύο διαφορετικές ροές στις οποίες γίνεται ζωντανή εξαγωγή χαρακτηριστικών. Στο Dashboard, απεικονίζονται οι δύο ροές, μια για κάθε ροή από χώρο στάθμευσης. Κάτω από τις ροές υπάρχουν ενδείξεις που αφορούν την διαθεσιμότητα του εκάστοτε χώρου.

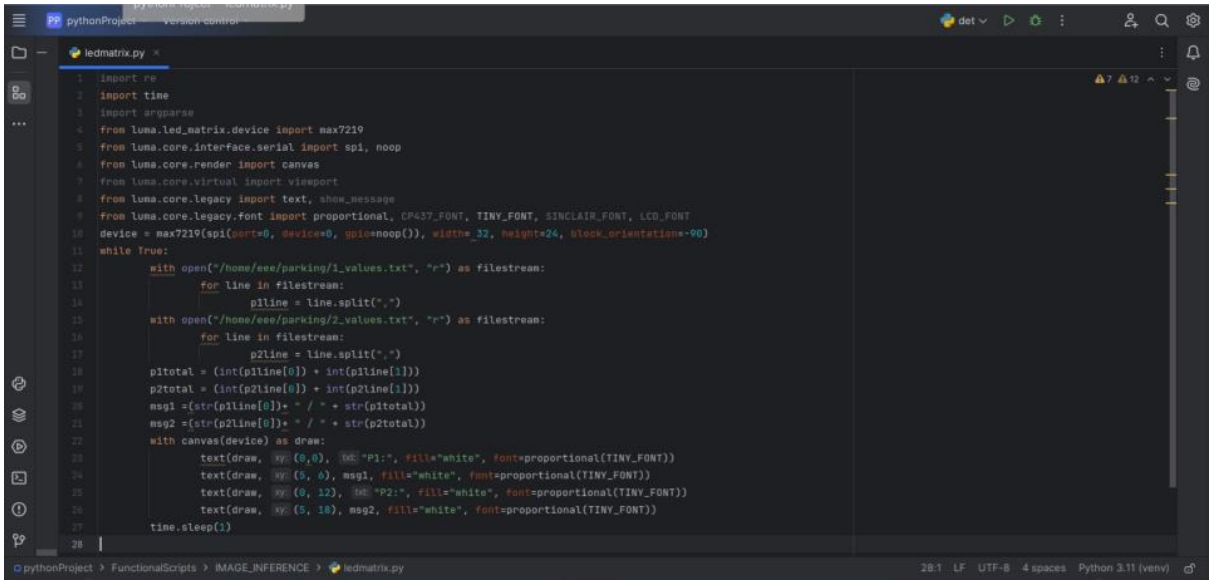
5.4.2 LED Matrix

Εκτός από το Dashboard, υλοποιήθηκε ένας ακόμη τρόπος οπτικοποίησης των πληροφοριών, κάνοντας χρήση τριών Led Matrix Displays.

Στην χρήση του πίνακα Led απαιτείται η εγκατάσταση της Python βιβλιοθήκης, **luma.led.matrix** μέσω της εντολής:

pip install luma.led.matrix.

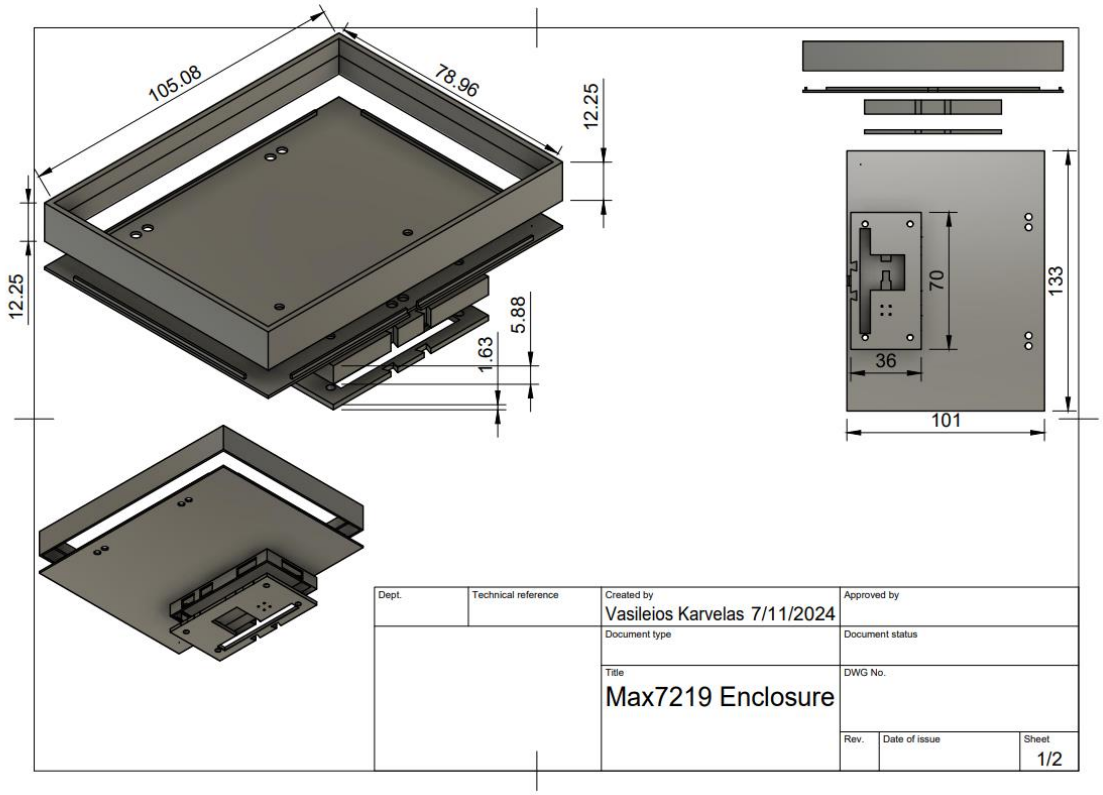
Παρακάτω ακολουθεί ο κώδικας Ledmatrix.py, ο οποίος παρέχει την προβολή των πληροφοριών διαθεσιμότητας στη οθόνη LED Matrix, η οποία συνδέεται μέσω SPI και του οδηγού MAX7219.



Εικόνα 89: Python Script: Ledmatrix.py

Στην Εικόνα 90 παρουσιάζεται ο υλοποιημένος πίνακας Led, ο οποίος αποτελείται από τρία modules σε παράλληλη διάταξη. Κάθε module φέρει τέσσερα πάνελ, με το καθένα να διαθέτει 64 LED σε διάταξη 8x8. Για τη σύνδεση των modules μεταξύ τους, καθώς και για τη σύνδεση με το Raspberry Pi, απαιτήθηκε η συγκόλληση καλωδίων. Επιπλέον, σχεδιάστηκε και εκτυπώθηκε μία 3D θήκη για την στέγαση των modules και για την προστασία των συνδέσεων. Στην Εικόνα 91 παρουσιάζεται το τεχνικό σχέδιο της θήκης.

Εικόνα 90: 32x24 Led Matrix – MAX7219



Εικόνα 91: Τεχνικό σχέδιο [1]: 3D Enclosure των LED Display Matrix και Raspberry Pi Zero 2 Access Point

Κεφάλαιο 6: Συμπεράσματα και Μελλοντικές Βελτιώσεις

6.1 Συμπεράσματα

Η αυξανόμενη χρήση οχημάτων, σε συνδυασμό με την έλλειψη υποδομών στάθμευσης, αποτελεί ένα σημαντικό πρόβλημα, που επηρεάζει αρνητικά την ποιότητα ζωής των ανθρώπων και του περιβάλλοντος. Η τεχνητή νοημοσύνη, η οποία έχει εισχωρήσει πλέον στην καθημερινότητα μας, μπορεί να προσφέρει λύσεις για την αντιμετώπιση του προβλήματος.

Στο πλαίσιο της παρούσας διπλωματικής εργασίας, ασχοληθήκαμε με την εκπαίδευση μοντέλων βασιζόμενων σε τρεις από τις εκδόσεις του αλγορίθμου ανίχνευσης αντικειμένων YOLO. Πριν από την εκπαίδευση, προβήκαμε στην επιλογή κατάλληλου συνόλου δεδομένων, από ένα αποθετήριο. Αρχικά, παρουσιάσαμε τα βήματα για την δημιουργία ενός συνόλου και εξηγήσαμε τους λόγους που μας οδήγησαν στην επιλογή ενός υφιστάμενου αντί για την δημιουργία ενός δικού μας.

Μετά το πέρας της εκπαίδευσης συγκρίναμε τα αποτελέσματα της εκπαίδευσης και λαμβάνοντας επίσης τα αποτελέσματα από την δοκιμή των μοντέλων στην εξαγωγή συμπερασμάτων από εικόνες και βίντεο, επιλέξαμε το κατάλληλο μοντέλου. Το επιλεγμένο μοντέλο YOLOv5n από τα υπόλοιπα, ήταν το πιο γρήγορο, με το μικρότερο μέγεθος και παρουσίαζε τις καλύτερες επιδόσεις στην αναγνώριση των θέσεων στάθμευσης.

Ακολούθησε η εφαρμογή του μοντέλου στο Raspberry Pi 4 για την εξαγωγή συμπερασμάτων από μία ροή βίντεο. Το σύστημα ανταποκρίθηκε ικανοποιητικά. Έπειτα παρουσιάστηκε ένα πιο ρεαλιστικό σενάριο που περιλάμβανε τη χρήση πολλαπλών καμερών, δοκιμάζοντας ταυτόχρονα τις αντοχές του συστήματος. Το σύστημα δεν ανταποκρίθηκε επαρκώς.

Για την αντιμετώπιση της αδυναμίας του συστήματος, πραγματοποιήθηκε η μετατροπή του μοντέλου από PyTorch σε ONNX. Η μετατροπή οδήγησε σε μείωση του χρόνου εξαγωγής συμπερασμάτων μειώθηκε στο μισό. Τα αποτελέσματα που προέκυψαν από τη μετατροπή, κρίθηκαν ικανοποιητικά καθώς δε παρατηρήθηκε μείωση της ακρίβειας στην ανίχνευσης. Δεν πραγματοποιήθηκαν περαιτέρω προσπάθειες για επιπλέον βελτιώσεις.

Η συσκευή που υλοποιήθηκε, αποτελούμενη από το Raspberry Pi 4, αποτελεί μία αξιόπιστη λύση για την αναγνώριση των θέσεων στάθμευσης.

6.2 Μελλοντικές Βελτιώσεις

- Επιλογή διαφορετικού συνόλου δεδομένων ή δημιουργία σύνθετου.
- Εφαρμογή και των υπολοίπων προαναφερθέντων προσεγγίσεων για την βελτίωσης της απόδοσης της συσκευής όπως του κβαντισμού και του κλαδέματος του μοντέλου.
- Χρήση νεότερου και πιο ισχυρού υπολογιστικού συστήματος όπως το Raspberry Pi 5, σε συνδυασμό με έναν επιταχυντή Google Coral, στοχεύοντας στην μείωση του χρόνου εξαγωγής συμπερασμάτων.
- Εκπαίδευση μοντέλων από διαφορετικούς αλγόριθμους ανίχνευσης αντικείμενων και σύγκριση των αποτελεσμάτων με της παρούσας εργασίας.

Βιβλιογραφία – Αναφορές – Διαδικτυακές Πηγές

[1]. M. Neelam, Neelam MahaLakshmi. (2021) *Aspects of Artificial Intelligence In Karthikeyan.J,* Su-Hie Ting and Yu-Jin Ng (eds), *"Learning Outcomes of Classroom Research"* pp:250-256.

[2]. Wikipedia Contributors. *"History of Artificial Intelligence"*, Πρόσβαση: Ιούλιος 10, 2024. [Online], Διαθέσιμο: https://en.wikipedia.org/wiki/History_of_artificial_intelligence#Birth_of_artificial_intelligence_(1941-56).

[3]. Rancho Labs. *"6 Major Sub-Fields of Artificial Intelligence – Rancho Labs"* Medium, Πρόσβαση: Ιούλιος 10, 2024. [Online], Διαθέσιμο: https://rancholabs.medium.com/6-major-sub-fields-of artificial-intelligence-77f6a5b28109.

[4]. A.Burkov, *"The Hundred-Page Machine Learning Book, Draft", Andriy Burkov,* 2019.

[5]. J. P. Mueller, L. Massaron. *"Deep Learning for Dummies",* For Dummies, 1st ed, 2019.

[6]. S. Haykin. *"Neural Networks and Learning Machines",* 3rd ed, Hamilton, Ontario, Canada: Pearson, 2009.

[7]. K. O'Shea, R. Nash. (2015, Δεκέμβριος), *"An Introduction to Convolutional Neural Networks",* arXiv preprint, pp:3-9, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://arxiv.org/pdf/1511.08458

[8]. DeepAI. (2019, May. 17). *"Feedforward Neural Network",* Πρόσβαση: Ιούλιος 10, 2024 [Online]. Διαθέσιμο: https://deepai.org/machine-learning-glossary-and-terms/feed-forward-neural-network

[9]. GeeksforGeeks. (2018, Οκτώβριος. 03). *"Introduction to Recurrent Neural Network",* Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://www.geeksforgeeks.org/introduction-to-recurrent-neural-network

[10]. GeeksforGeeks. (2024, Μάϊος. 03). *"Activation Functions in Neural Networks",* Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://www.geeksforgeeks.org/activation-functions-neural-networks

[11]. Wikipedia Contributors, (2003, Μάρτιος, 26). *"Gradient Descent",* Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://en.wikipedia.org/wiki/Gradient_descent

[12]. Wikipedia Contributors, (2005, Ιανουάριος, 4). *"Backpropagation",* Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://en.wikipedia.org/wiki/Backpropagation

[13]. Wikipedia Contributors, (2011, Ιούλιος, 20). *"Deep Learning",* Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://en.wikipedia.org/wiki/Deep_learning

[14]. R. Szeliski, (2010, Σεπτέμβριος, 3). *"Computer Vision: Algorithms and Applications",* Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://szeliski.org/Book/

[15]. Wikipedia Contributors, (2008, Φεβρουάριος, 18). *"Object Detection",* Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://en.wikipedia.org/wiki/Object_detection

[16]. M. Carranza-García, J. Torres-Mateo, P. Lara-Benítez, and J. García-Gutiérrez, *"On the Performance of One-Stage and Two-Stage Object Detectors in Autonomous Vehicles Using Camera Data" Remote Sensing*, vol. 13, no. 1, Dec. 2020, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: doi: 10.3390/rs13010089.

[17]. V7Labs. (2023, Ιανουάριος. 17). *"YOLO: Algorithm for Object Detection Explained"* Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://www.v7labs.com/blog/yolo-object-detection

[18]. M. Al Rabiani-Aliif, M Hussain *"YOLOv1 to YOLOv10: A comprehensive review of YOLO variants and their application in the agricultural domain",* Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://doi.org//arXiv.2406.10139

[19]. Google Scholar. *"About Google Scholar"*, Πρόσβαση: Ιούλιος 10, 2024, [Online], Διαθέσιμο: https://scholar.google.com/intl/en/scholar/about.html

[20]. S. N. Murami, *"Automated Car Parking Space Detection Using Deep Learning"*, Thesis, Dept. Comp. Inform., Nairobi Univ., 2019.

[21]. A. E. Dwiputra, H. Khoswanto, R. Sutjiadi and R. Lim, *"IoT-Based Car's Parking Monitoring System"*, in *"Matec Web of Conferences"*, Vol. 164. EDP Sciences, 2018, https://repository.petra.ac.id/17850/1/Publikasi1_91024_4054.pdf

[22]. W. A. Jabar, C. Wen Wei, N. A. Atiqah Ainna M. Azmi and N. A. Haironnazli, *"An IoT Raspberry Pi-Based Parking Management System For Smart Campus", Internet of Things*, Vol. 14, 2021: 100387, https://doi.org/10.1051/matecconf/201816401002

[23]. M. Ogawa, T. Arnon and E. Gruber *"Identifying Parking Lot Occupancy with YOLOv5" Journal of Student Research*, Vol. 12, no. 4, 2023 doi:10.47.611/jsr. v12i4.2280

[24]. Wikipedia Contributors, (2011, Μάιος, 6). *"Raspberry Pi"*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://en.wikipedia.org/wiki/Raspberry_Pi

[25]. Raspberry Pi, (2019, Ιούνιος, 24). *"Raspberry Pi Tech Specs"*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/

[26]. Raspberry Pi, (2024, Ιούνιος). *"Raspberry Pi GPIO and the 40-pin header"*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://www.raspberrypi.com/documentation/computers/raspberry-pi.html#gpio-and-the-40-pin-header

[27]. Raspberry Pi, (2020, Μάϊος, 15). *"Raspberry Pi Camera Module 3"*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://datasheets.raspberrypi.com/camera/camera-module-3-product-brief.pdf

[28]. Wikipedia Contributors, (2024, Μάϊος, 14). *"Debian"*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://en.wikipedia.org/wiki/Debian

[29]. Raspberry Pi, (2024, Ιούνιος). *"Raspberry Pi OS"*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://www.raspberrypi.com/documentation/computers/os.html

[30]. Wikipedia Contributors, (2024, Ιούλιος, 5). *"Raspberry Pi Operating Systems"*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://en.wikipedia.org/wiki/Raspberry_Pi#Operating_Systems

[31]. Python, (2024). *"General Python FAQ"*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://docs.python.org/3/faq/general.html#what-is-python

[32]. Wikipedia Contributors, (2024, Ιούλιος, 4). *"Python (Programming Language)"*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://en.wikipedia.org/wiki/Python_(programming_language)

[33]. PYPL, (2024, Ιούλιος, 7). *"PYPL Popularity of Programming Language"*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://pypl.github.io/PYPL.html

[34]. Wikipedia Contributors, (2024, Ιούνιος, 8). *"PyCharm)"*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://en.wikipedia.org/wiki/PyCharm

[35]. Wikipedia Contributors, (2024, Απρίλιος, 20). *"IPython"*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://en.wikipedia.org/wiki/IPython

[36]. Jupyter4Edu. "Chapter 5 Jupyter Notebook ecosystem", Teaching and Learning with Jupyter

[37]. Node-Red, (2024). *"About: Node-Red"*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://nodered.org/about/#history

[38]. OpenCV, (2024). *"About OpenCV"*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://opencv.org/about/

[39]. Wikipedia Contributors, (2024, Φεβρουάριος, 19). *"OpenCV"*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://en.wikipedia.org/wiki/OpenCV

[40]. NumPy, (2024). *"About Us"*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://numpy.org/about/

[41]. Wikipedia Contributors, (2024, Απρίλιος, 2). *"NumPy"*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://en.wikipedia.org/wiki/NumPy

[42]. Matplotlib, (2024). *"Matplotlib: Visualization with Python"*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://matplotlib.org/

[43]. Wikipedia Contributors, (2024, Μάϊος, 9). *"Matplotlib"*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://en.wikipedia.org/wiki/Matplotlib

[44]. Wikipedia Contributors, (2024, Ιούνιος, 13). *"Torch (Machine Learning)"*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://en.wikipedia.org/wiki/Torch_(machine_learning)

[45]. Raspberry Pi, (2024). *"Introducing Raspberry Pi Imager, our new imaging Utility"*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://www.raspberrypi.com/news/raspberry-pi-imager-imaging-utility/

[46]. Car Parking Space, (2023, Δεκέμβριος), *"Parking Spot Detector Dataset", Roboflow*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://universe.roboflow.com/car-parking-space/parking-spot-detector-a84ql

[47]. G. Jocher, B. Qaddoumi, (2023, Νοέμβριος, 12). *"YOLOv5: Train Custom Data"*, in *Ultralytics Docs*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://docs.ultralytics.com/yolov5/tutorials/train_custom_data/#next-steps

[48]. G. Jocher, B. Qaddoumi, F. C. Akyon and L. Q., (2023, Νοέμβριος, 12). *"YOLOv5: Train Custom Data"*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://docs.ultralytics.com/yolov5/tutorials/train_custom_data/#next-steps

[49]. J. Gallagher, P. Skalski, (2024, Μάϊος 24), *"How to train YOLOv10 Model on a Custom Dataset"*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://blog.roboflow.com/yolov10-how-to-train/

[50]. G. Jocher, A.Vina, (2024, Ιούλιος 5), *"Best Practices for Model Deployment"*, *Ultralytics*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://docs.ultralytics.com/guides/model-deployment-practices/

[51]. G. Jocher, A.Vina, (2024, Ιούλιος 5), *"ONXX Export", Ultralytics*, Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://docs.ultralytics.com/integrations/onnx

Οπτικοακουστικά μέσα που χρησιμοποιήθηκαν στην εξαγωγή συμπερασμάτων:

[52]. Cambridge Consultants. *"New Multi-Storey Car Park – Construction Time-Lapse"*. (2017, Σεπτέμβριος 15). Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://www.youtube.com/watch?v=4mpD7XEyqbw

[53]. Supercircuits. *"Security Camera Parking Lot Surveillance Video"*. (2012, Οκτώβριος 23). Πρόσβαση: Ιούλιος 10, 2024 [Online], Διαθέσιμο: https://www.youtube.com/watch?v=U7HRKjlXK-Y

Παράρτημα Α – Κώδικες

Κώδικας 1: Python Script: ImageCap.py – Κεφάλαιο 4.1.1: Συλλογή εικόνων για την δημιουργία Dataset

```
# Εισαγωγή απαραίτητων κλάσεων από τις Βιβλιοθήκες: Picamera2,libcamera, os, time.
from picamera2 import Picamera2, Preview
from libcamera import Transform
from os import system
from time import sleep
# Δημιουργία νέου αντικειμένου με όνομα camera, που επιτρέπει την αλληλεπίδραση με την κάμερα PiModule3:
camera = Picamera2()
# Ρύθμιση Ανάλυσης:
camera.resolution = (1920, 1080)
# Εκκίνηση Κάμερας:
camera.start()
# Βρόχος για την λήψη και αποθήκευση 100 φωτογραφιών με ρυθμό λήψης τα 30 λεπτά
for i in range(100):
    filename = f"image_{i+1}.jpg"
    camera.capture_file(filename)
    sleep(1800)
```

Κώδικας 2: Python Script: ImageInference.py – Κεφάλαιο 5.1 – Σελίδα 97

```
# Εισαγωγή απαραίτητων κλάσεων από τις Βιβλιοθήκες: torch και YOLO:
import torch
from ultralytics import YOLO
#Επιλογή εκπαιδευμένου μοντέλου:
model = torch.hub.load('/home/eee/.cache/torch/hub/ultralytics_yolov5_master',
'custom', '/models/yolo5sbest150epoch.pt', source='local')
#yolov8: model = YOLO("/model/yolov8sbest150epoch.pt")
#yolov10: model = YOLO("/model/yolov10nbest150epoch.pt")
#Εικόνα εισόδου:
img = "/test/test.jpg"
#Εξαγωγή Χαρακτηριστικών – Ανίχνευση αντικειμένων:
results = model(img)
#Εμφάνιση αποτελεσμάτων
results.print()
results.pandas().xyxy[0].value_counts('name')
```

Κώδικας 3: Python Script: Videoinference.py – Κεφάλαιο 5.2 – Σελίδα 100

```
# Εισαγωγή απαραίτητων κλάσεων από τις Βιβλιοθήκες: torch και YOLO:
import cv2
from ultralytics import YOLO
import pandas as pd
import numpy as np
import time
import torch
# Επιλογή εκπαιδευμένου μοντέλου:
model = torch.hub.load('/home/eee/.cache/torch/hub/ultralytics_yolov5_master',
'custom', '/home/eee/Desktop/parking_det/models/yolo5nbest150epoch.pt',
source='local')
# Δημιουργία νέου αντικειμένου για το βίντεο
cap = cv2.VideoCapture('/home/eee/Desktop/parking_det/test/ee.mp4')
# Αρχικοποίηση μεταβλητών υπολογισμού FPS
prevtime = 0
newtime = 0
while(cap.isOpened()):
# Άνοιγμα νέου παραθύρου με την ροή της κάμερας & Λήψη καρέ
    ret, frame = cap.read()
# Μετασχηματισμός καρέ
    results = model(cv2.resize(frame, (640, 640)))
# Επιλογή της γραμματοσειράς
    font = cv2.FONT_HERSHEY_SIMPLEX
    newtime = time.time()
# Υπολογισμός των καρέ ανά δευτερόλεπτο
    fps = 1/(newtime-prevtime)
    prevtime = newtime
# Προβολή FPS στην οθόνη
    fps = ("FPS:" + str(float("{:.2f}".format(fps))))
    cv2.putText(gray, fps, (1, 25), font, 1, (100, 255, 0), 1, cv2.LINE_AA)
# Προβολή των πλαισίων οριοθέτησης - bounding boxes
    results.render()
    s=results.pandas().xyxy[0]['class'].value_counts()
# Εκτύπωση στο τερματικό: Διαθέσιμες και Κατειλημμένες θέσεις
    print(s)
# Εμφάνιση στην ροή: Πλήθος κατειλημμένων θέσεων
    Occ = ("Occupied: " + str(s[1]))
    cv2.putText(gray, occ, (400, 25), font, 1, (100, 255, 0), 1, cv2.LINE_AA)
    emp = ("Empty: " + str(s[0]))
# Εμφάνιση στην ροή: Πλήθος ελευθέρων θέσεων
    cv2.putText(gray, emp, (400, 50), font, 1, (100, 255, 0), 1, cv2.LINE_AA)
# Κλείσιμο Ροής
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
```

Κώδικας 4: Python Script: YOLOv5Flask.py – Κεφάλαιο 5.4.1 – Σελίδα 108

```
# Εισαγωγή απαραίτητων κλάσεων από τις Βιβλιοθήκες: torch, numpy, cv2, ultralytics,
time, panda και flask
import cv2
from ultralytics import YOLO
import pandas as pd
import numpy as np
import time
import torch
from flask import Flask, Response
#Επιλογή εκπαιδευμένου μοντέλου:
model = torch.hub.load('/home/eee/.cache/torch/hub/ultralytics_yolov5_master',
'custom', '/home/eee/Desktop/parking_det/models/yolo5nbest150epoch.onnx',
source='local')
#Δημιουργία αντικειμένου Flask
app = Flask(__name__)
# Αρχικοποίηση μεταβλητών υπολογισμού FPS
prev_frame_time = 0
new_frame_time = 0
#Δημιουργία αντικειμένου του βίντεο / κάμερας
video = cv2.VideoCapture('/home/eee/Desktop/parking_det/test/ee.mp4')
#Έναρξη συνάρτησης detect()
#Η detect() είναι υπεύθυνη για την εξαγωγή συμπερασμάτων και την προβολή των
πλαισίων οριοθέτησης για κάθε ένα καρέ του μέσου εισόδου
def detect(frame):
        global prev_frame_time,new_frame_time
        # Μετασχηματισμός καρέ
        frame = cv2.resize(frame, (640, 640))
        results = model(frame)
        results.render()
        results.print()
        # Επιλογή της γραμματοσειράς
        font = cv2.FONT_HERSHEY_SIMPLEX
        # Υπολογισμός των καρέ ανά δευτερόλεπτο
        prev_frame_time = 0
        new_frame_time = time.time()
        fps = 1/(new_frame_time-prev_frame_time)
        prev_frame_time = new_frame_time
        # Προβολή FPS στην οθόνη
        fps = ("FPS:" + str(float("{:.2f}".format(fps))))        # render the FPS
count on the frame
        cv2.putText(frame, fps, (1, 25), font, 1, (100, 255, 0), 1, cv2.LINE_AA)
        # Προβολή αποτελεσμάτων inference στο τερματικό
        s=results.pandas().xyxy[0]['class'].value_counts() #show
        print(s) #class 0 = empty, class 1 = occupied
        #Αποθήκευση σε αρχείο .txt
        s = ",".join(map(str, s.values))
        with open("/home/eee/Desktop/parking_det/NodeRED/yolov5_torch/YOLOV5-
Flask_DOUBLE/1_values.txt", "w") as f:
            f.write(str(s))
        # Εμφάνιση στην ροή: Πλήθος ελευθέρων θέσεων
        #occ=("Occupied: " + str(s[1]))
        #cv2.putText(frame, occ, (400, 25), font, 1, (100, 255, 0), 1, cv2.LINE_AA)
        #emp=("Empty: " + str(s[0]))
        #cv2.putText(frame, emp, (400, 50), font, 1, (100, 255, 0), 1, cv2.LINE_AA)
        return frame
#Έναρξη συνάρτησης gen().
def gen(video):
    while True:
        # Λήψη καρέ από την ροή του βίντεο ή της κάμερας
        success, frame = video.read()
        frame= detect(frame)
        #Χρήση της συνάρτησης cv2.imencode() της βιβλιοθήκης OpenCV, για την
κωδικοποίηση του κάρε σε μορφή .JPG
```

```
        ret, jpeg = cv2.imencode('.jpg', frame)
        #Μετατροπή του κωδικοποιημένου καρέ σε ακολουθία byte(string) για την
μετάδοση της ροής εισόδου με τα πλαίσια οριοθέτησης (μέσω του Flask)
        frame = jpeg.tobytes()
        #Αποστολή του κωδικοποιημένου Jpeg καρέ για την δημιουργία ροής, προσβάσιμη
στον περιηγητή ιστού.
        yield (b'--frame\r\n'
                  b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n\r\n')
#Συνάρτηση αντικειμένου Flask
@app.route('/')
#διεύθυνση URL του διακομιστή Flask, όπου γίνεται προβολή της ροής
def index():
    global video
    return Response(gen(video),
                    mimetype='multipart/x-mixed-replace; boundary=frame')

#Εκκίνηση διακομιστή web Flask στο Raspberry Pi 4.
if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000, threaded=True)
#port=5000: Η θύρα του διακομιστή Flask
#threaded=True: Ενεργοποίηση της ταυτόχρονης διαχείρησης πολλαπλών νημάτων.
#host='0.0.0.0': Ο διακομιστής Flask(ροή εισόδου) είναι προσβάσιμος από όλες τις
συσκευές του δικτύου
```

Κώδικας 5: Python Script: Ledmatrix.py – Κεφάλαιο 5.3.2 – Σελίδα 104

```
import re
import time
import argparse
# Εισαγωγή module από τη βιβλιοθήκη luma-led-matrix & παραμετροποίηση
from luma.led_matrix.device import max7219
from luma.core.interface.serial import spi, noop
from luma.core.render import canvas
from luma.core.virtual import viewport
from luma.core.legacy import text, show_message
from luma.core.legacy.font import proportional, CP437_FONT, TINY_FONT,
SINCLAIR_FONT, LCD_FONT
serial = spi(port=0, device=0, gpio=noop())
device = max7219(serial, width= 32, height=24, block_orientation=-90) #
blocks_arranged_in_reverse_order=False) #cascaded=4
while True:
# Φόρτωση αρχείων με την διαθεσιμότητα των χώρων στάθμευσης
        with open("/home/eee/parking/1_values.txt", "r") as filestream:
                for line in filestream:
                        p1line = line.split(",")
        with open("/home/eee/parking/2_values.txt", "r") as filestream:
                for line in filestream:
                        p2line = line.split(",")
        p1total = (int(p1line[0]) + int(p1line[1]))
        p2total = (int(p2line[0]) + int(p2line[1]))
        msg1 =(str(p1line[0])+ " / " + str(p1total))
        msg2 =(str(p2line[0])+ " / " + str(p2total))
# Εμφάνιση της διαθεσιμότητας
        with canvas(device) as draw:
                text(draw, (0,0), "P1:", fill="white",
font=proportional(TINY_FONT))
                text(draw, (5, 6), msg1, fill="white",
font=proportional(TINY_FONT))
                text(draw, (0, 12), "P2:", fill="white",
font=proportional(TINY_FONT))
                text(draw, (5, 18), msg2, fill="white",
font=proportional(TINY_FONT))
        time.sleep(1)
```

Κώδικας 6: Python Script USBcamera.py – Εξαγωγή συμπερασμάτων από USB Camera

```
#Εισαγωγή των απαραίτητων βιβλιοθηκών
import numpy as np
import cv2
import time
import torch
#Επιλογή εκπαιδευμένου μοντέλου
model = torch.hub.load('/home/*/.cache/torch/hub/ultralytics_yolov5_master', 'custom',
'/home/*/Desktop/yolo5n.pt', source='local')
# Δημιουργία νέου αντικειμένου για την χρήση της εξωτερικής USB κάμερας
cap = cv2.VideoCapture("/dev/v4l/by-path/xxxxxxxxxxx-video-index0")
# Αρχικοποίηση μεταβλητών υπολογισμού FPS
prevtime = 0
newtime = 0
while(cap.isOpened()):
# Άνοιγμα νέου παραθύρου με την ροή της κάμερας & Λήψη καρέ
    ret, frame = cap.read()
# Μετασχηματισμός καρέ
    results = model(cv2.resize(frame, (640, 640)))
# Επιλογή της γραμματοσειράς
    font = cv2.FONT_HERSHEY_SIMPLEX
    newtime = time.time()
# Υπολογισμός των καρέ ανά δευτερόλεπτο
    fps = 1/(newtime-prevtime)
    prevtime = newtime
# Προβολή FPS στην οθόνη
    fps = ("FPS:" + str(float("{:.2f}".format(fps))))
    cv2.putText(gray, fps, (1, 25), font, 1, (100, 255, 0), 1, cv2.LINE_AA) # Προβολή των
πλαισίων οριοθέτησης - bounding boxes
    results.render()
    s=results.pandas().xyxy[0]['class'].value_counts()
# Εκτύπωση στο τερματικό: Διαθέσιμες και Κατειλημμένες θέσεις
    print(s)
# Εμφάνιση στην ροή: Πλήθος κατειλημμένων θέσεων
    Occ = ("Occupied: " + str(s[1]))
    cv2.putText(gray, occ, (400, 25), font, 1, (100, 255, 0), 1, cv2.LINE_AA)
    emp = ("Empty: " + str(s[0]))
# Εμφάνιση στην ροή: Πλήθος ελευθέρων θέσεων
    cv2.putText(gray, emp, (400, 50), font, 1, (100, 255, 0), 1, cv2.LINE_AA)
# Δημιουργία νέου αρχείου με όνομα values.txt, όπου αποθηκεύονται οι τιμές των ελευθέρων και
κατειλημμένων θέσεων για χρήση με τη Node-Red
    s = ",".join(map(str, s.values))
    with open("/home/***/Desktop/*.txt", "w") as f:
        f.write(str(s))
# Εμφάνιση της ροής σε ένα παράθυρο με όνομα 'Frame'
cv2.imshow('frame', gray)
```

Παράρτημα Β – Node-Red: FLOWS

Ροή Dashboard [1] – Κεφάλαιο 3 – Σελίδα 76


```
[{"id":"5bedd9252247fe89","type":"tab","label":"YOLOV5_FLASK_SINGLE_FEED","disabled":false,"info":"","env":
[]},{"id":"1daf63736c5a7e9b","type":"ui_button","z":"5bedd9252247fe89","name":"","group":"df340af30f142da7"
,"order":12,"width":6,"height":1,"passthru":false,"label":"YOLOv5","tooltip":"","color":"","bgcolor":"","cl
assName":"","icon":"","payload":"","payloadType":"str","topic":"topic","topicType":"msg","x":560,"y":80,"wi
res":[["3447692cc57d162b"]]},{"id":"84a6adebb2c761bb","type":"ui_artlessgauge","z":"5bedd9252247fe89","grou
p":"df340af30f142da7","order":2,"width":0,"height":0,"name":"Occupied_Gauge","icon":"","label":"Occupied","
unit":"","layout":"linear","decimals":0,"differential":false,"minmax":false,"colorTrack":"#555555","style":
"","colorFromTheme":true,"property":"payload","secondary":"secondary","inline":false,"animate":true,"sector
s":[{"val":0,"col":"#ff9900","t":"min","dot":0},{"val":10,"col":"#ff9900","t":"max","dot":0}],"lineWidth":3
,"bgcolorFromTheme":true,"diffCenter":"","x":990,"y":160,"wires":[]},{"id":"2a67a3d2dba61206","type":"ui_ar
tlessgauge","z":"5bedd9252247fe89","group":"df340af30f142da7","order":4,"width":0,"height":0,"name":"Empty_
Gauge","icon":"","label":"Empty","unit":"","layout":"linear","decimals":0,"differential":false,"minmax":fal
se,"colorTrack":"#555555","style":"","colorFromTheme":true,"property":"payload","secondary":"secondary","in
line":false,"animate":true,"sectors":[{"val":0,"col":"#ff9900","t":"min","dot":0},{"val":10,"col":"#ff9900"
,"t":"max","dot":0}],"lineWidth":3,"bgcolorFromTheme":true,"diffCenter":"","x":980,"y":360,"wires":[]},{"id
":"3447692cc57d162b","type":"trigger","z":"5bedd9252247fe89","name":"","op1":"","op2":"0","op1type":"nul","
op2type":"str","duration":"1000","extend":false,"overrideDelay":false,"units":"ms","reset":"","bytopic":"al
l","topic":"topic","outputs":1,"x":580,"y":120,"wires":[["5a0efcf2ece46810"]]},{"id":"dbc18b0ec8308840","ty
pe":"split","z":"5bedd9252247fe89","name":"","splt":",","spltType":"str","arraySplt":1,"arraySpltType":"len
","stream":false,"addname":"","x":550,"y":200,"wires":[["855a7d3ac303fadf"]]},{"id":"5a0efcf2ece46810","typ
e":"file          in","z":"5bedd9252247fe89","name":"Read_Results","filename":"/NodeRED/yolov5_torch/YOLOV5-
Flask/values.txt","filenameType":"str","format":"utf8","chunk":false,"sendError":false,"encoding":"none","a
llProps":false,"x":580,"y":160,"wires":[["dbc18b0ec8308840"]]},{"id":"855a7d3ac303fadf","type":"switch","z"
:"5bedd9252247fe89","name":"split_payload","property":"parts.index","propertyType":"msg","rules":[{"t":"eq"
,"v":"0","vt":"str"},{"t":"eq","v":"1","vt":"str"}],"checkall":"true","repair":false,"outputs":2,"x":570,"y
":240,"wires":[["84a6adebb2c761bb","5ca3002ef9ff7f13"],["2a67a3d2dba61206","e8e93de21b697257"]]},{"id":"a9a
703ac8235d2e3","type":"pythonshell
in","z":"5bedd9252247fe89","name":"Video_FEED","pyfile":"/NodeRED/Scripts/YOLOV5-
Flask/FLASK_MP4_YOLOv5.py","virtualenv":"/home/eee/Desktop/yolov8_ultralytics/venv","continuous":false,"std
InData":false,"x":570,"y":40,"wires":[["3447692cc57d162b"]]},{"id":"1ad11229c3a14b5f","type":"ui_iframe","z
":"5bedd9252247fe89","group":"217d4a9051b6e33e","name":"VIDEO_HTML","order":1,"width":12,"height":12,"url":
"http://parkingdet.local:5000/video_feed","origin":"*","scale":100,"x":570,"y":320,"wires":[[]]},{"id":"5ca
3002ef9ff7f13","type":"change","z":"5bedd9252247fe89","name":"Occupied","rules":[{"t":"set","p":"payload","
pt":"msg","to":"$number(payload)","tot":"jsonata"},{"t":"set","p":"topic","pt":"msg","to":"a","tot":"str"}]
,"action":"","property":"","from":"","to":"","reg":false,"x":720,"y":240,"wires":[["d7b43819a53386d6","9789
d6371618fd05"]]},{"id":"e8e93de21b697257","type":"change","z":"5bedd9252247fe89","name":"Empty","rules":[{"
t":"set","p":"payload","pt":"msg","to":"$number(payload)","tot":"jsonata"},{"t":"set","p":"topic","pt":"msg
","to":"b","tot":"str"}],"action":"","property":"","from":"","to":"","reg":false,"x":710,"y":280,"wires":[[
"d7b43819a53386d6"]]},{"id":"d7b43819a53386d6","type":"SumUltimate","z":"5bedd9252247fe89","name":"SUM","pr
operty":"payload","math":"sum","subtractstartfrom":"","x":830,"y":280,"wires":[["98dccbed05abc703"]]},{"id"
:"9789d6371618fd05","type":"ui_digital_display","z":"5bedd9252247fe89","name":"Occupied_Spots","group":"df3
40af30f142da7","order":8,"width":2,"height":3,"digits":"2","decimals":"0","x":980,"y":240,"wires":[]},{"id"
:"98dccbed05abc703","type":"ui_digital_display","z":"5bedd9252247fe89","name":"Total_Detected_Spots","group
":"df340af30f142da7","order":10,"width":2,"height":3,"digits":"2","decimals":"0","x":1000,"y":320,"wires":[
]},{"id":"b81788b3fa81a610","type":"ui_text","z":"5bedd9252247fe89","group":"df340af30f142da7","order":9,"w
idth":2,"height":3,"name":"","label":"/","format":"{{msg.payload}}","layout":"col-
center","className":"","style":true,"font":"","fontSize":"40","color":"#ffffff","x":950,"y":280,"wires":[]}
,{"id":"dd7e81888b1b623c","type":"ui_text","z":"5bedd9252247fe89","group":"df340af30f142da7","order":6,"wid
th":6,"height":1,"name":"Availability","label":"Availability","format":"{{msg.payload}}","layout":"col-
center","className":"","style":true,"font":"","fontSize":"24","color":"#ffff00","x":960,"y":200,"wires":[]}
,{"id":"dc2afbada208627b","type":"pythonshell
in","z":"5bedd9252247fe89","name":"Video_USB_CAMERA","pyfile":"/NodeRED/Scripts/YOLOV5-
Flask/FLASK_USB_CAMERA_YOLOv5.py","virtualenv":"/home/eee/Desktop/yolov8_ultralytics/venv","continuous":fal
se,"stdInData":false,"x":320,"y":80,"wires":[[]]},{"id":"064585870560e5ad","type":"pythonshell
in","z":"5bedd9252247fe89","name":"Video_PI_CAMERA","pyfile":"/NodeRED/Scripts/YOLOV5-
Flask/FLASK_PI_CAMERA_YOLOv5.py","virtualenv":"/home/eee/Desktop/yolov8_ultralytics/venv","continuous":fals
e,"stdInData":false,"x":330,"y":40,"wires":[[]]},{"id":"843d7ac2245d2f1d","type":"ui_spacer","z":"5bedd9252
247fe89","name":"spacer","group":"df340af30f142da7","order":1,"width":6,"height":1},{"id":"8747f653c4f7758c
","type":"ui_spacer","z":"5bedd9252247fe89","name":"spacer","group":"df340af30f142da7","order":3,"width":6,
"height":1},{"id":"7a8892b2389e5818","type":"ui_spacer","z":"5bedd9252247fe89","name":"spacer","group":"df3
40af30f142da7","order":5,"width":6,"height":1},{"id":"90eb3174ea024f39","type":"ui_spacer","z":"5bedd925224
7fe89","name":"spacer","group":"df340af30f142da7","order":7,"width":6,"height":1},{"id":"beda6a8286c8a85a",
"type":"ui_spacer","z":"5bedd9252247fe89","name":"spacer","group":"df340af30f142da7","order":11,"width":6,"
height":1},{"id":"df340af30f142da7","type":"ui_group","name":"YOLOV5_FLASK_CONTROL","tab":"7a85cb07d7584649
","order":2,"disp":true,"width":"6","collapse":false,"className":""},{"id":"217d4a9051b6e33e","type":"ui_gr
oup","name":"Video
Feed","tab":"7a85cb07d7584649","order":1,"disp":true,"width":"12","collapse":false,"className":""},{"id":"7
a85cb07d7584649","type":"ui_tab","name":"YOLOV5_FLASK_VIDEO","icon":"dashboard","disabled":false,"hidden":f
alse}]
```

Ροή Dashboard [2] – Κεφάλαιο 5 – Σελίδα 106


```
[{"id":"76e5fc2d90810236","type":"tab","label":"YOLOV5_FLASK_DOUBLE_FEEDS","disabled":false,"info":"","e
nv":[]},{"id":"1ac320a7ec12d5c5","type":"ui_button","z":"76e5fc2d90810236","name":"","group":"0b3cf3503f
d78df8","order":5,"width":2,"height":2,"passthru":false,"label":"YOLOv5","tooltip":"","color":"","bgcolo
r":"","className":"","icon":"","payload":"","payloadType":"str","topic":"topic","topicType":"msg","x":10
60,"y":120,"wires":[["2e05c45bc6d8b511"]]},{"id":"bd45586865dcfcca","type":"ui_artlessgauge","z":"76e5fc
2d90810236","group":"0b3cf3503fd78df8","order":2,"width":0,"height":0,"name":"1_Occupied_Gauge","icon":"
","label":"Occupied","unit":"","layout":"linear","decimals":0,"differential":false,"minmax":false,"color
Track":"#555555","style":"","colorFromTheme":true,"property":"payload","secondary":"secondary","inline":
false,"animate":true,"sectors":[{"val":0,"col":"#ff9900","t":"min","dot":0},{"val":10,"col":"#ff9900","t
":"max","dot":0}],"lineWidth":3,"bgcolorFromTheme":true,"diffCenter":"","x":1490,"y":200,"wires":[]},{"i
d":"825885320629e632","type":"ui_artlessgauge","z":"76e5fc2d90810236","group":"0b3cf3503fd78df8","order"
:3,"width":0,"height":0,"name":"1_Empty_Gauge","icon":"","label":"Empty","unit":"","layout":"linear","de
cimals":0,"differential":false,"minmax":false,"colorTrack":"#555555","style":"","colorFromTheme":true,"p
roperty":"payload","secondary":"secondary","inline":false,"animate":true,"sectors":[{"val":0,"col":"#ff9
900","t":"min","dot":0},{"val":10,"col":"#ff9900","t":"max","dot":0}],"lineWidth":3,"bgcolorFromTheme":t
rue,"diffCenter":"","x":1480,"y":400,"wires":[]},{"id":"2e05c45bc6d8b511","type":"trigger","z":"76e5fc2d
90810236","name":"","op1":"","op2":"0","op1type":"nul","op2type":"str","duration":"1000","extend":false,
"overrideDelay":false,"units":"ms","reset":"","bytopic":"all","topic":"topic","outputs":1,"x":1080,"y":1
60,"wires":[["75bd505296a62035","fbc151070282e5b7"]]},{"id":"74237d7cdfa6d17c","type":"split","z":"76e5f
c2d90810236","name":"","splt":",","spltType":"str","arraySplt":1,"arraySpltType":"len","stream":false,"a
ddname":"","x":1050,"y":240,"wires":[["c427988fade80e29"]]},{"id":"75bd505296a62035","type":"file
in","z":"76e5fc2d90810236","name":"Read_Results_1","filename":"/NodeRED/yolov5_torch/YOLOV5-
Flask_DOUBLE/1_values.txt","filenameType":"str","format":"utf8","chunk":false,"sendError":false,"encodin
g":"none","allProps":false,"x":1080,"y":200,"wires":[["74237d7cdfa6d17c"]]},{"id":"c427988fade80e29","ty
pe":"switch","z":"76e5fc2d90810236","name":"split_payload","property":"parts.index","propertyType":"msg"
,"rules":[{"t":"eq","v":"0","vt":"str"},{"t":"eq","v":"1","vt":"str"}],"checkall":"true","repair":false,
"outputs":2,"x":1070,"y":300,"wires":[["bd45586865dcfcca","7e8817efdd8a8a96"],["825885320629e632","d06bc
1e6926056f5"]]},{"id":"ce69a58005d64f77","type":"pythonshell
in","z":"76e5fc2d90810236","name":"FEED_1","pyfile":"/NodeRED/Scripts/YOLOV5-
Flask_DOUBLE/1_FLASK_MP4_YOLOv5.py","virtualenv":"/home/eee/Desktop/yolov8_ultralytics/venv","continuous
":false,"stdInData":false,"x":1060,"y":80,"wires":[["2e05c45bc6d8b511","540b006af1015094"]]},{"id":"19d8
faf22b1830e2","type":"ui_iframe","z":"76e5fc2d90810236","group":"0b3cf3503fd78df8","name":"1_VIDEO_HTML"
,"order":1,"width":12,"height":12,"url":"http://parkingdet.local:5000/video_feed","origin":"*","scale":1
00,"x":1040,"y":360,"wires":[[]]},{"id":"7e8817efdd8a8a96","type":"change","z":"76e5fc2d90810236","name"
:"Occupied","rules":[{"t":"set","p":"payload","pt":"msg","to":"$number(payload)","tot":"jsonata"},{"t":"
set","p":"topic","pt":"msg","to":"a","tot":"str"}],"action":"","property":"","from":"","to":"","reg":fal
se,"x":1220,"y":280,"wires":[["978d90aa74df43ed","616120d34bf6140d"]]},{"id":"d06bc1e6926056f5","type":"
change","z":"76e5fc2d90810236","name":"Empty","rules":[{"t":"set","p":"payload","pt":"msg","to":"$number
(payload)","tot":"jsonata"},{"t":"set","p":"topic","pt":"msg","to":"b","tot":"str"}],"action":"","proper
ty":"","from":"","to":"","reg":false,"x":1210,"y":320,"wires":[["978d90aa74df43ed"]]},{"id":"978d90aa74d
f43ed","type":"SumUltimate","z":"76e5fc2d90810236","name":"SUM","property":"payload","math":"sum","subtr
actstartfrom":"","x":1330,"y":320,"wires":[["eddd6fa7be09ad88"]]},{"id":"616120d34bf6140d","type":"ui_di
gital_display","z":"76e5fc2d90810236","name":"1_Occupied_Spots","group":"0b3cf3503fd78df8","order":7,"wi
dth":2,"height":3,"digits":"2","decimals":"0","x":1480,"y":280,"wires":[]},{"id":"eddd6fa7be09ad88","type
e":"ui_digital_display","z":"76e5fc2d90810236","name":"1_Total_Detected_Spots","group":"0b3cf3503fd78df8
","order":9,"width":2,"height":3,"digits":"2","decimals":"0","x":1510,"y":360,"wires":[]},{"id":"f94c0ac
92b87eb72","type":"ui_text","z":"76e5fc2d90810236","group":"0b3cf3503fd78df8","order":8,"width":2,"heigh
t":3,"name":"1_/","label":"/","format":"{{msg.payload}}","layout":"col-
center","className":"","style":true,"font":"","fontSize":"40","color":"#ffffff","x":1450,"y":320,"wires"
:[]},{"id":"2eb3cb64b971c0b3","type":"ui_text","z":"76e5fc2d90810236","group":"0b3cf3503fd78df8","order"
:4,"width":12,"height":1,"name":"1_Availability","label":"PARKING LOT 1 |
Availability","format":"{{msg.payload}}","layout":"col-
center","className":"","style":true,"font":"","fontSize":"24","color":"#ffff00","x":1460,"y":240,"wires"
:[]},{"id":"5c8ad120c0bd198e","type":"pythonshell
in","z":"76e5fc2d90810236","name":"FEED_2","pyfile":"/NodeRED/Scripts/YOLOV5-
Flask_DOUBLE/2_FLASK_MP4_YOLOv5.py","virtualenv":"/home/eee/Desktop/yolov8_ultralytics/venv","continuous
":false,"stdInData":false,"x":1060,"y":40,"wires":[["2e05c45bc6d8b511","b530c61a499dc849"]]},{"id":"fbc1
51070282e5b7","type":"file
in","z":"76e5fc2d90810236","name":"Read_Results_2","filename":"/NodeRED/yolov5_torch/YOLOV5-
Flask_DOUBLE/2_values.txt","filenameType":"str","format":"utf8","chunk":false,"sendError":false,"encodin
g":"none","allProps":false,"x":1060,"y":440,"wires":[["83649d719c47aa24"]]},{"id":"83649d719c47aa24","ty
pe":"split","z":"76e5fc2d90810236","name":"","splt":",","spltType":"str","arraySplt":1,"arraySpltType":"
len","stream":false,"addname":"","x":1030,"y":480,"wires":[["615dd375cc8b82b5"]]},{"id":"615dd375cc8b82b
5","type":"switch","z":"76e5fc2d90810236","name":"split_payload","property":"parts.index","propertyType"
:"msg","rules":[{"t":"eq","v":"0","vt":"str"},{"t":"eq","v":"1","vt":"str"}],"checkall":"true","repair":
false,"outputs":2,"x":1050,"y":520,"wires":[["e3b47751a8649b33","602e946095333fa2"],["9f6fdb44baaf8a5a",
"eee47ed9cfa67fad"]]},{"id":"e3b47751a8649b33","type":"change","z":"76e5fc2d90810236","name":"Occupied",
"rules":[{"t":"set","p":"payload","pt":"msg","to":"$number(payload)","tot":"jsonata"},{"t":"set","p":"to
pic","pt":"msg","to":"a","tot":"str"}],"action":"","property":"","from":"","to":"","reg":false,"x":1200,
"y":520,"wires":[["80c08ded679ea68a","376cd453e911f438"]]},{"id":"9f6fdb44baaf8a5a","type":"change","z":
"76e5fc2d90810236","name":"Empty","rules":[{"t":"set","p":"payload","pt":"msg","to":"$number(payload)","
tot":"jsonata"},{"t":"set","p":"topic","pt":"msg","to":"b","tot":"str"}],"action":"","property":"","from
":"","to":"","reg":false,"x":1190,"y":560,"wires":[["80c08ded679ea68a"]]},{"id":"80c08ded679ea68a","type
":"SumUltimate","z":"76e5fc2d90810236","name":"SUM","property":"payload","math":"sum","subtractstartfrom
":"","x":1330,"y":560,"wires":[["3dba1415c71f7b8c"]]},{"id":"18ce7c1c4c9b92f8","type":"ui_text","z":"76e
5fc2d90810236","group":"21128524394897ef","order":7,"width":2,"height":3,"name":"2_/","label":"/","forma
t":"{{msg.payload}}","layout":"col-
center","className":"","style":true,"font":"","fontSize":"40","color":"#ffffff","x":1490,"y":680,"wires"
:[]},{"id":"29d82c0a3068d226","type":"ui_iframe","z":"76e5fc2d90810236","group":"21128524394897ef","name
":"2_VIDEO_HTML","order":1,"width":12,"height":12,"url":"http://parkingdet.local:5001/video_feed","origi
```

```
n":"*","scale":100,"x":1060,"y":700,"wires":[[]]},{"id":"602e946095333fa2","type":"ui_artlessgauge","z":
"76e5fc2d90810236","group":"21128524394897ef","order":2,"width":0,"height":0,"name":"2_Occupied_Gauge","
icon":"","label":"Occupied","unit":"","layout":"linear","decimals":0,"differential":false,"minmax":false
,"colorTrack":"#555555","style":"","colorFromTheme":true,"property":"payload","secondary":"secondary","i
nline":false,"animate":true,"sectors":[{"val":0,"col":"#ff9900","t":"min","dot":0},{"val":10,"col":"#ff9
900","t":"max","dot":0}],"lineWidth":3,"bgcolorFromTheme":true,"diffCenter":"","x":1530,"y":560,"wires":
[]},{"id":"eee47ed9cfa67fad","type":"ui_artlessgauge","z":"76e5fc2d90810236","group":"21128524394897ef",
"order":3,"width":0,"height":0,"name":"2_Empty_Gauge","icon":"","label":"Empty","unit":"","layout":"line
ar","decimals":0,"differential":false,"minmax":false,"colorTrack":"#555555","style":"","colorFromTheme":
true,"property":"payload","secondary":"secondary","inline":false,"animate":true,"sectors":[{"val":0,"col
":"#ff9900","t":"min","dot":0},{"val":10,"col":"#ff9900","t":"max","dot":0}],"lineWidth":3,"bgcolorFromT
heme":true,"diffCenter":"","x":1520,"y":760,"wires":[]},{"id":"376cd453e911f438","type":"ui_digital_disp
lay","z":"76e5fc2d90810236","name":"2_Occupied_Spots","group":"21128524394897ef","order":6,"width":2,"he
ight":3,"digits":"2","decimals":"0","x":1530,"y":640,"wires":[]},{"id":"3dba1415c71f7b8c","type":"ui_dig
ital_display","z":"76e5fc2d90810236","name":"2_Total_Detected_Spots","group":"21128524394897ef","order":
8,"width":2,"height":3,"digits":"2","decimals":"0","x":1550,"y":720,"wires":[]},{"id":"27f83674ce259095"
,"type":"ui_text","z":"76e5fc2d90810236","group":"21128524394897ef","order":4,"width":12,"height":1,"nam
e":"2_Availability","label":"PARKING LOT 2 | Availability","format":"{{msg.payload}}","layout":"col-
center","classNam":"","style":true,"font":"","fontSize":"24","color":"#ffff00","x":1510,"y":600,"wires"
:[]},{"id":"540b006af1015094","type":"debug","z":"76e5fc2d90810236","name":"debug
1","active":true,"tosidebar":true,"console":false,"tostatus":false,"complete":"false","statusVal":"","st
atusType":"auto","x":1260,"y":80,"wires":[]},{"id":"b530c61a499dc849","type":"debug","z":"76e5fc2d908102
36","name":"debug
2","active":true,"tosidebar":true,"console":false,"tostatus":false,"complete":"false","statusVal":"","st
atusType":"auto","x":1260,"y":40,"wires":[]},{"id":"da29c44ec2c25b21","type":"ui_button","z":"76e5fc2d90
810236","name":"","group":"21128524394897ef","order":10,"width":2,"height":2,"passthru":false,"label":"M
ATRIX","tooltip":"","color":"","bgcolor":"","className":"","icon":"","payload":"","payloadType":"str","t
opic":"topic","topicType":"msg","x":1040,"y":800,"wires":[["fbc92c3c4b674500"]]},{"id":"fbc92c3c4b674500
","type":"trigger","z":"76e5fc2d90810236","name":"","op1":"","op2":"0","op1type":"nul","op2type":"str","
duration":"1000","extend":false,"overrideDelay":false,"units":"ms","reset":"","bytopic":"all","topic":"t
opic","outputs":1,"x":1060,"y":840,"wires":[["77c42f5b5a173642"]]},{"id":"77c42f5b5a173642","type":"pyth
onshell in","z":"76e5fc2d90810236","name":"Matrix","pyfile":"/NodeRED/Scripts/YOLOV5-
Flask_DOUBLE/MATRIX.py","virtualenv":"/home/eee/Desktop/yolov8_ultralytics/venv","continuous":false,"std
InData":false,"x":1030,"y":880,"wires":[[]]},{"id":"4c808f6fc7c4b400","type":"ui_spacer","z":"76e5fc2d90
810236","name":"spacer","group":"0b3cf3503fd78df8","order":6,"width":1,"height":1},{"id":"a571505b10a84c
a2","type":"ui_spacer","z":"76e5fc2d90810236","name":"spacer","group":"0b3cf3503fd78df8","order":10,"wid
th":3,"height":1},{"id":"3a8d530a2bbd78ca","type":"ui_spacer","z":"76e5fc2d90810236","name":"spacer","gr
oup":"0b3cf3503fd78df8","order":11,"width":1,"height":1},{"id":"cad77755078533d6","type":"ui_spacer","z"
:"76e5fc2d90810236","name":"spacer","group":"0b3cf3503fd78df8","order":12,"width":3,"height":1},{"id":"7
d20e316e56edf04","type":"ui_spacer","z":"76e5fc2d90810236","name":"spacer","group":"0b3cf3503fd78df8","o
rder":13,"width":3,"height":1},{"id":"c75b3c0bd3465ea8","type":"ui_spacer","z":"76e5fc2d90810236","name"
:"spacer","group":"0b3cf3503fd78df8","order":14,"width":3,"height":1},{"id":"dfa51b8ec6e9ed97","type":"u
i_spacer","z":"76e5fc2d90810236","name":"spacer","group":"21128524394897ef","order":5,"width":3,"height"
:1},{"id":"7002a83f79978b2c","type":"ui_spacer","z":"76e5fc2d90810236","name":"spacer","group":"21128524
394897ef","order":9,"width":1,"height":1},{"id":"5f0709d316b29a31","type":"ui_spacer","z":"76e5fc2d90810
236","name":"spacer","group":"21128524394897ef","order":11,"width":3,"height":1},{"id":"817551538fee0f29
","type":"ui_spacer","z":"76e5fc2d90810236","name":"spacer","group":"21128524394897ef","order":12,"width
":1,"height":1},{"id":"d22d4bdbfb881e15","type":"ui_spacer","z":"76e5fc2d90810236","name":"spacer","grou
p":"21128524394897ef","order":13,"width":3,"height":1},{"id":"e0a598f8033785c4","type":"ui_spacer","z":"
76e5fc2d90810236","name":"spacer","group":"21128524394897ef","order":14,"width":3,"height":1},{"id":"0b3
cf3503fd78df8","type":"ui_group","name":"PARKING LOT
1","tab":"17d0e016ea97d0ac","order":1,"disp":true,"width":"12","collapse":false,"className":""},{"id":"2
1128524394897ef","type":"ui_group","name":"PARKING LOT
2","tab":"17d0e016ea97d0ac","order":3,"disp":true,"width":"12","collapse":false,"className":""},{"id":"1
7d0e016ea97d0ac","type":"ui_tab","name":"YOLOV5_FLASK_DOUBLE","icon":"dashboard","order":5,"disabled":fa
lse,"hidden":false}]
```