Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων
Τομεας Τεχνολογιας Πληροφορικης και Υπολογιστων

Surveillance system for Mask detection using AI techniques

Διπλωματικη Εργασια

του

ΠΕΤΡΟΥ ΝΤΟΥΝΗ

Επιβλέπων: Δημήτριος Τσουμάκος
Αναπληρωτής Καθηγήτής Ε.Μ.Π.

Αθήνα, Οκτώβριος 2023

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών

Surveillance system for Mask detection using AI techniques

Διπλωματικη Εργασια

του

ΠΕΤΡΟΥ ΝΤΟΥΝΗ

Επιβλέπων: Δημήτριος Τσουμάκος
Αναπληρωτής Καθηγήτής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 18η Οκτωβρίου 2023

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Δημήτριος Τσουμάκος	Συμεών Παπαβασιλείου	Γεώργιος Γκούμας
Αν. Καθηγήτής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.

Αθήνα, Οκτώβριος 2023

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών

(Υπογραφή)

.....

Πετρος Ντουνης
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Το 2019 ξεκίνησε η πιο πρόσφατη πανδημία, αυτή του COVID-19, η οποία κλόνισε τη παγκόσμια κοινότητα. Ένα απο τα πιό αποτελεσματικά μέτρα που ελήφθησαν ήταν αυτό της υποχρεωτικοποίησης της μάσκας σε δημόσιους χώρους, εμποδίζοντας την εξάπλωσή του. Σε πολλούς χώρους οπου ο συνωστισμός ήταν αναπόφευκτος όπως αεροδρόμια ή αποβάθρες τρένων η χρήση της μάσκας θα μπορούσε να γίνει monitor με χρήση αυτοματοποιημένου συστήματος χρησιμοποιώντας τεχνολογίες Internet Of things , machine learning. Η χρήση του live feed απο τις κάμερες για προσδιορισμό του ποσοστού των ατόμων που φορούν την μάσκα είναι και ένα πρόβλημα επιστημονικού ενδιαφέροντος λόγω της πολυπλοκότητας του προβλήματος για να γίνει real-time. Η παρούσα διπλωματική εργασία σκοπεύει να παρουσιάσει ένα ολοκληρωμένο μοντέλο real time ανάλυσης του live feed.

Λέξεις κλειδιά:
Internet of Things, MQTT, Apache Kafka, InfluxDB, Machine learning, YOLO, Object Tracking

Abstract

2019 was the year of the most recent pandemic of COVID-19 which shocked global community. One of the best measures against the spreading of the disease was the use of mask in public spaces. In many of those public spaces monitoring the percentage of people wearing a mask can be implemented as an autonomous system using technologies from branches of Internet of things and Machine learning. Using live-feed from surveillance cameras to determine the percentages of people who wear and people who don't is an interesting scientific problem due to its inherit difficulty in order to be real-time. The purpose of this thesis is to present an end to end system to solve this exact problem.

Keywords:
Internet of Things, MQTT, Apache Kafka, InfluxDB, Machine learning, YOLO, Object Tracking

Ευχαριστίες

Θα ήθελα να ευχαριστήσω ξεχωριστά όλους όσους συνέβαλαν στην πραγματοποίηση της διπλωματικής μου εργασίας με τη συμβολή τους. Θα ήθελα να ευχαριστήσω θερμά την καθηγήτρια **κυρία Καντερέ**, η οποία μου έδωσε την ευκαιρία και με εμπιστεύτηκε να ασχοληθώ με μια εφαρμογή τόσο πρακτικού όσο και θεωρητικού ενδιαφέροντος. Χάρις την καθοδήγηση και την επίβλεψή της κατάφερα να εκπονήσω την διπλωματική μου εργασία. Επίσης θερμά θα ήθελα να ευχαριστήσω τον καθηγητή **κύριο Τσουμάκο** ο οποίος ήταν στην συνέχεια επιβλέπων μετά την αποχώρηση της κυρίας Καντερε. Πολλές ευχαριστίες θα ήθελα να απευθύνω και στον κύριο **Παρασκευά Κερασιώτη**, που η βοήθεια του ήταν πολύτιμη και χωρίς αυτή δε θα είχα καταφέρει να φέρω εις πέρας την παρούσα εργασία. Τέλος, θα ήθελα να ευχαριστήσω την οικογένεια και τους φίλους μου που με στήριξαν καθ'όλη την διάρκεια των σπουδών μου και ιδιαίτερα κατα την διάρκεια πραγμάτωσης της συγκεκριμένης εργασίας. Σας ευχαριστώ όλους!

Περιεχόμενα

Κατάλογος Σχημάτων

Κεφάλαιο 1

Εισαγωγή

1.1 Σκοπός της εργασίας

Η παρούσα διπλωματική εργασία έχει ως στόχο την μελέτη και ανάπτυξη ενός ολοκληρωμένου συστήματος αναγνωρισης ανθρώπων που φορούν ή δεν φορούν μάσκα σε ζωντανή ροη βίντεο κατανεμημένα σε κάμερες και την απόδοση χαρακτηρηστικών (ID) σε καθέναν απο αυτούς για καλύτερη εκτίμηση του αριθμού των ατόμων που φορούν και δεν φορούν μάσκα. Η απόδοση (ID) και η ακολούθηση των ατόμων κατα τη διάρκεια που βρίσκονται στο οπτικό πεδίο της κάμερας βοηθά στην καλύτερη εκτίμηση, αφού αν κάποιος εμφανιστεί σε ένα πλάνο με μάσκα και στη συνέχεια γυρίσει και το μοντέλο αναγνώρισης δεν καταφέρει να αναγνψρίσει εκ νέου οτι φοράει μάσκα, η πληροφορία αυτή θα έχει παραμείνει απο τον αλγόριθμο παρακολούθησης αφού ο άνθρωπος θα είναι σημειωμένος στη βάση δεδομένων ως κάποιος που φοράει μάσκα ακόμη και αν δεν μπορεί να γίνει η αναγνώριση της στο συγκεκριμένο καρέ. Από εκεί οι πληροφορίες περι μασκών και ατόμον πρέπει να μεταφέρονται στα ενδιάμεσα συστήματα εως ότου καταλήξουν στην βάση δεδομένων, εν αρμονία πάντοτε με την ζωντανή ροή που απαιτείται.

1.2 Γενικό πλαίσιο

Νοηματικά υπάρχουν δύο μεγάλοι τομείς με τους οποίους ασχολείται η παρούσα εργασία. Ο πρώτος τομέας αφορά στη μηχανική μάθηση και είναι αυτός του προβλήματος υπολογισμού του αριθμού των ατόμων με μάσκα και χωρίς και της ακολούθησής τους στο βίντεο για καλύτερη ακρίβεια του εν λόγω υπολογισμού. Εδώ πέρα απο την γενική οριοθέτηση του προβλήαμτος υπάρχει και η πρόκληση του live-feed που σημαίνει ότι τα μοντέλα θα πρέπει πέρα απο αποδοτικά να είναι και ελαφριά προκείμενου να μπορούν να λειτουργήσουν σε ζωντανή ροή βίντεο, σε ενα σύστημα παρακολούθησης. Ο δέυτερος τομέας αφορά στις παραδοσιακές προκλήσεις του Internet of things, συγκεκριμένα στο κατα πόσο είναι εφικτή η μετάδοση της πληροφορίας σε ζωντανό χρόνο, έμπιστα και με ακρίβεια απο το endpoind που υπολογίζονται και επιπλέον η έμπιστη και έγκυρη αποθήκευσή τους απο πολλές κάμερες στην ίδια βάση δεδομένων δημιουργόντας ένα ακέραιο κατανεμημένο σύστημα.

1.3 Δομή της εργασίας

Η παρούσα εργασία είναι χωρισμένη σε δύο μέρη: το θεωρητικό και το πρακτικό, Στο θεωρητικό μέρος αναπτύσσεται όλο το απαιτούμενο υπόβαθρο στα πλαίσια της εργασίας χωρίς υπερβολικά τεχνικά σημεία χωρισμένο νοηματικά ανα κεφάλαια. Το κάθε κεφάλαιο στο θεωρητικό μέρος αποτελεί μια ενδελεχή εισαγωγή στον εκάστοτε τομέα που υλοποιήθηκε και χρησιμοποιήθηκε για την επίτυξη του στόχου της εργασίας. Σε αντιστοιχία το πρακτικό μέρος αποτελέιται απο δύο κεφάλαια τα οποία είναι νοηματικά χωρισμένα ώς εξής: το κεφάλαιο 7 περιγράφει τα αποτελέσματα των μοντέλων μηχανικής μάθησης μεά την υλοποίηση τους καθώς και την εκπαίδευση και τις μετρικές απόδοσής τους, ενώ το κεφάλαιο 8 περιγράφει την ροή της υπόλοιπης πληροφορίας απο τη στιγμή που υπολογίζεται, απο το πρωτόκολλο επικοινωνίας MQTT με το οποίο αποστέλονται αρχικά έως της κατάληξής τους στη βάση δεδομένων.

1.4 Επίλυση του προβλήματος

Περιγράφουμε στα κεφάλαια της εργασίας τόσο θεωρητικά, όσο και πειραματικά τα επι μέρους σημεία που οδηγούν στην κατασκευή του εν λόγω συστήματος. Συγκεκριμένα περιγράφονται τα μοντέλα μηχανικής μάθησης που αφορούν τόσο την αναγνώριση μάσκας όσο και την παρακολούθηση και απόδοση αναγνωριστικων τα οποία χρησιμοποιούνται για τον υπολογισμό των ζητούμενων δεδομένων. Στη συνέχεια περιγράφεται το MQTT που είναι ένα πρωτόκολλο επικοινωνίας μεταξύ των συσκευών IoT , ή στην περίπτωσή μας των καμερών. Το άλλο άκρο του MQTT το καταλαμβάνει ο Apache Kafka που χρησιμοποιείται για την σταθεροποίηση των δεδομένων και της αύξησης του scalability του συστήματος. Από τον Apache Kafka λαμβάνει τα δεδομένα το εργαλείο Telegraf και τα εναποθέτει στον τελικό τους προορισμο, την (βελτιστοποιημένη για εφαρμογές IoT) βάση δεδομένων InfluxDB

Μέρος I

Θεωρητικό Μέρος

Κεφάλαιο 2

Internet of Things (IoT)

2.1 Εισαγωγή

Το (IoT) αναφέρεται σε ένα δίκτυο φυσικών συσκευών, οχημάτων, και άλλων φυσικών αντικειμένων που διαθέτουν αισθητήρες, λογισμικό και συνδεσιμότητα δικτύου που τους επιτρέπει να συλλέγουν και να μοιράζονται δεδομένα. Αυτές οι συσκευές - γνωστές και ως ¨έξυπνα αντικείμενα' - μπορεί να κυμαίνονται από απλές συσκευές ¨έξυπνου σπιτιού¨, όπως έξυπνοι θερμοστάτες, μέχρι wearables, όπως έξυπνα ρολόγια και ρούχα με δυνατότητα RFID, μέχρι πολύπλοκα βιομηχανικά μηχανήματα και συστήματα μεταφορών. Σήμερα, οραματίζονται ακόμη και ολόκληρες ¨έξυπνες πόλεις' που βασίζονται σε τεχνολογίες IoT.

Το IoT επιτρέπει σε αυτές τις έξυπνες συσκευές να επικοινωνούν μεταξύ τους και με άλλες συσκευές με δυνατότητα σύνδεσης στο διαδίκτυο, όπως τα smartphones και τα gateways, δημιουργώντας ένα τεράστιο δίκτυο διασυνδεδεμένων συσκευών που μπορούν να ανταλλάσσουν δεδομένα και να εκτελούν αυτόνομα μια ποικιλία εργασιών. Αυτό μπορεί να περιλαμβάνει τα πάντα, από την παρακολούθηση των περιβαλλοντικών συνθηκών σε αγροκτήματα, έως τη διαχείριση της κυκλοφορίας με έξυπνα αυτοκίνητα και άλλες έξυπνες συσκευές αυτοκινήτων, τον έλεγχο μηχανών και διαδικασιών σε εργοστάσια, την παρακολούθηση αποθεμάτων και αποστολών σε αποθήκες.

Οι πιθανές εφαρμογές του IoT είναι τεράστιες και ποικίλες και ο αντίκτυπός του είναι ήδη αισθητός σε ένα ευρύ φάσμα βιομηχανιών, όπως η μεταποίηση, οι μεταφορές, η υγειονομική περίθαλψη και η γεωργία. Καθώς ο αριθμός των συνδεδεμένων στο διαδίκτυο συσκευών συνεχίζει να αυξάνεται, το IoT είναι πιθανό να διαδραματίσει ολοένα και πιο σημαντικό ρόλο στη διαμόρφωση του κόσμου μας και να μεταμορφώσει τον τρόπο με τον οποίο ζούμε, εργαζόμαστε και αλληλεπιδρούμε μεταξύ μας.

Σε επιχειρησιακό πλαίσιο, οι συσκευές IoT χρησιμοποιούνται για την παρακολούθηση ενός ευρέος φάσματος παραμέτρων, όπως η θερμοκρασία, η υγρασία, η ποιότητα του αέρα, η κατανάλωση ενέργειας και η απόδοση των μηχανημάτων. Τα δεδομένα αυτά μπορούν να αναλυθούν σε πραγματικό χρόνο για τον εντοπισμό μοτίβων, τάσεων και ανωμαλιών που μπορούν να βοηθήσουν τις επιχειρήσεις να βελτιστοποιήσουν τις λειτουργίες τους και να βελτιώσουν το τελικό τους αποτέλεσμα. [10]

Σχήμα 2.1: Δημοτικότητα Internet of Things

2.2 Ορισμός

Το Internet of Things θεωρείται ή υποδομή που επιτρέπει σε μια πληθώρα συσκευών να επικοινωνούν μεταξύ τους με βασικό και απαραίτητο χαρακτηριστικό της επικοινωνίας να αποτελεί η σύνδεση στο ίντερνετ. Οι συσκευές αυτές μπορεί να περιλαμβάνουν τόσο συσκευές όσο και υπηρεσίες δικτύπου.Μερικά τυπικά παραδείγματα των συσκεών χρησιμοποιούνται για ιατρικούς σκοπούς (αισθητήρες για οξυγόνο, ελέγχου καρδιακού παλμού), σε βιομηχανικές εγκαταστάσεις (για τον έλεγχο θερμοκρασιών, για ασφαλείας), σε έξυπνα προϊόντα (τηλεοράσεις, smartphones).Ανά τα χρόνια έχουν δωθεί διαφορετικοι ορισμοί για το τι περιγράφει ακριβως με την παραπάνω να είναι πάντα η βασική ιδέα. Ενδεικτικά παρατίθενται μερικοί απο αυτούς:

1. Internet of Things είναι ένα πλαίσιο στο οποίο όλες οι συσκευές έχουν μια εκπροσώπηση και οντότητα στο Διαδίκτυο. Ειδικότερα, το IoT έχει ως στόχο στην παροχή νέων εφαρμογών και υπηρεσιών που αποσκοπούν στην έπικοινωνία του φυσικού και του εικονικού κόσμου, με το Machine to Machine (M2M) να αποτελεί τη γραμμή επικοινωνίας που επιτρέπει την αλληλεπίδραση μεταξύ πραγμάτων και εφαρμογών σε επίπεδο σλουδ. (IEEE Communications Magazine)

2. Το IoT (ουσιαστικό): Η διασύνδεση, μέσω του διαδικτύου, υπολογιστικών συσκευών. Η διασύνδεση αυτή είναι ενσωματωμένη σε καθημερινά αντικείμενα, παρέχοντάς τους τη δυνατότητα να στείλουν και να λάβουν δεδομένα. (Oxford Λεξικό, 2015)

3. IoT: Μια παγκόσμια υποδομή για την κοινωνία των πληροφοριών, παρέχοντας προηγμένες υπηρεσίες μέσω διασυνδεδεμένων (φυσικών και εικονικών) συσκευών (“πράγμα-

τα") που βασίζονται στις υφιστάμενες και εξελισσόμενες διαλειτουργικές τεχνολογίες πληροφόρησης και επικοινωνίας. (Διεθνής Ένωση Τηλεπικοινωνιών - ITU)

2.3 Ιστορία του IoT

Αν και παραδείγματα διασυνδεδεμένων ηλεκτρονικών συσκευών υπάρχουν ήδη από τις αρχές του 19ου αιώνα, με την εφεύρεση του τηλέγραφου και την ικανότητά του να μεταδίδει πληροφορίες με κωδικοποιημένο σήμα σε απόσταση, οι απαρχές του IoT χρονολογούνται στα τέλη της δεκαετίας του 1960. Τότε, μια ομάδα διακεκριμένων ερευνητών άρχισε να διερευνά τρόπους σύνδεσης υπολογιστών και συστημάτων. Κορυφαίο παράδειγμα αυτής της εργασίας ήταν το ARPANET, το δίκτυο που δημιουργήθηκε από την Υπηρεσία Προηγμένων Ερευνητικών Προγραμμάτων (ARPA) του Υπουργείου Άμυνας των ΗΠΑ - το δίκτυο αυτό αποτέλεσε πρόδρομο του σημερινού Διαδικτύου. Στα τέλη της δεκαετίας του 1970 επιχειρήσεις, κυβερνήσεις και καταναλωτές άρχισαν να διερευνούν τρόπους σύνδεσης προσωπικών υπολογιστών (PC) και άλλων μηχανημάτων μεταξύ τους. Μέχρι τη δεκαετία του 1980 τα τοπικά δίκτυα (LAN) παρείχαν έναν αποτελεσματικό και ευρέως χρησιμοποιούμενο τρόπο επικοινωνίας και κοινής χρήσης εγγράφων, δεδομένων και άλλων πληροφοριών σε μια ομάδα υπολογιστών σε πραγματικό χρόνο.

Μέχρι τα μέσα της δεκαετίας του 1990 το Διαδίκτυο επέκτεινε αυτές τις δυνατότητες σε παγκόσμιο επίπεδο και οι ερευνητές και οι τεχνολόγοι άρχισαν να διερευνούν τρόπους με τους οποίους άνθρωποι και μηχανές θα μπορούσαν να συνδεθούν καλύτερα. Το 1997 ο Βρετανός τεχνολόγος Κέβιν Άστον, συνιδρυτής του Auto-ID Center στο , άρχισε να διερευνά ένα τεχνολογικό πλαίσιο, την αναγνώριση ραδιοσυχνοτήτων (RFID), που θα επέτρεπε σε φυσικές συσκευές να συνδέονται μέσω μικροτσίπ και ασύρματων σημάτων, και ήταν σε μια ομιλία του το 1999 που ο Άστον επινόησε τη φράση 'Internet of Things¨. Μέσα σε λίγα χρόνια τα smartphones, το cloud computing, οι εξελίξεις στην επεξεργαστική ισχύ και οι βελτιωμένοι αλγόριθμοι λογισμικού είχαν δημιουργήσει ένα πλαίσιο για τη συλλογή, αποθήκευση, επεξεργασία και ανταλλαγή δεδομένων με πιο ισχυρό τρόπο. Ταυτόχρονα, εμφανίστηκαν εξελιγμένοι αισθητήρες που μπορούσαν να μετρήσουν την κίνηση, τη θερμοκρασία, τα επίπεδα υγρασίας, την κατεύθυνση του ανέμου, τον ήχο, το φως, τις εικόνες, τις δονήσεις και πολλές άλλες συνθήκες - μαζί με τη δυνατότητα εντοπισμού ενός ατόμου ή μιας συσκευής μέσω γεωγραφικού εντοπισμού. Αυτές οι εξελίξεις κατέστησαν δυνατή τη δυνατότητα επικοινωνίας τόσο με ψηφιακές συσκευές όσο και με φυσικά αντικείμενα σε πραγματικό χρόνο. Για παράδειγμα, με την προσθήκη ενός τσιπ εντοπισμού σε ένα αντικείμενο, όπως ένα πορτοφόλι ή μια βαλίτσα, είναι δυνατή η προβολή της θέσης του. Το ίδιο τσιπ ενσωματωμένο σε μια ψηφιακή συσκευή μπορεί να εντοπίσει την τοποθεσία της σε περίπτωση απώλειας ή κλοπής. Στη συνέχεια, με την ευρεία υιοθέτηση κινητών συσκευών όπως τα smartphones και τα tablets και την εισαγωγή της πανταχού παρούσας ασύρματης συνδεσιμότητας, κατέστη δυνατή η σύνδεση ανθρώπων

και πραγμάτων με σχεδόν πανταχού παρόντα τρόπο. Ως αποτέλεσμα, τα έξυπνα δίκτυα κυκλοφορίας, οι συνδεδεμένες δεξαμενές αποθήκευσης και τα συστήματα βιομηχανικής ρομποτικής έγιναν ο κανόνας.

Το IoT συνεχίζει να εξελίσσεται. Σήμερα υποστηρίζει μια σειρά από περιπτώσεις χρήσης, όπως η τεχνητή νοημοσύνη που χρησιμοποιείται για σύγχρονες προσομοιώσεις, συστήματα ανίχνευσης που ανιχνεύουν ρύπους σε αποθέματα νερού και συστήματα που παρακολουθούν αγροτικά ζώα και καλλιέργειες. Για παράδειγμα, είναι πλέον δυνατή η παρακολούθηση της θέσης και της υγείας των ζώων και η εφαρμογή εξ αποστάσεως των βέλτιστων επιπέδων νερού, λιπασμάτων και φυτοφαρμάκων στις καλλιέργειες. Παρακάτω εμφανίζονται καποια 1999: Ο όρος 'Internet of Things' επινοήθηκε από τον Kevin Ashton. Ο Ashton, ενώ εργαζόταν πάνω στην τεχνολογία RFID, οραματίστηκε ένα μέλλον όπου καθημερινά αντικείμενα θα μπορούσαν να συνδεθούν στο διαδίκτυο και να επικοινωνούν μεταξύ τους, οδηγώντας στη γέννηση της έννοιας του Διαδικτύου των Πραγμάτων.

Από τα τέλη του 20ου αιώνα και μετά: Συνεχείς εξελίξεις στην τεχνολογία αισθητήρων, με σημαντική πρόοδο στη δεκαετία του 2000 και μετά. Οι αισθητήρες έγιναν μικρότεροι, πιο προσιτοί και πιο αποτελεσματικοί, επιτρέποντας τη συλλογή δεδομένων από φυσικά αντικείμενα και περιβάλλοντα. Αυτή η πρόοδος έπαιξε ζωτικό ρόλο στην ανάπτυξη του IoT, παρέχοντας τα θεμέλια για την ανίχνευση και την παρακολούθηση διαφόρων παραμέτρων. Παρακάτω εμφανίζονται χρονολογικά τα σημαντικότερα ορόσημα του την τελευταια 25ετία:

- 1998: Εισαγωγή του IPv6, παρέχοντας τα θεμέλια για την ευρεία συνδεσιμότητα των συσκευών IoT. Η υιοθέτηση του IPv6 διεύρυνε τον χώρο διευθύνσεων που είναι διαθέσιμος για τις συσκευές, διευκολύνοντας τη σύνδεση μεγάλου αριθμού συσκευών IoT στο διαδίκτυο. Αυτό επέτρεψε την απρόσκοπτη επικοινωνία και μεταφορά δεδομένων μεταξύ των συσκευών IoT και του ςλουδ.

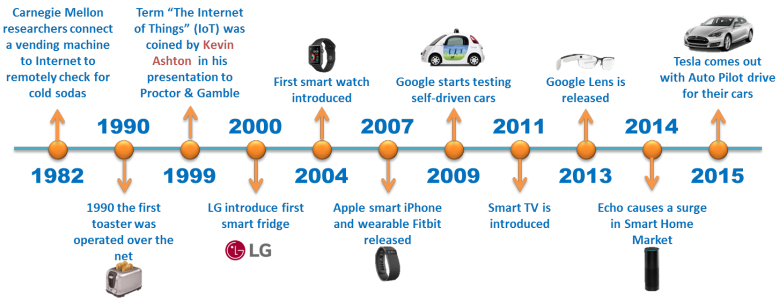
- Δεκαετίες του 2000 και 2010: Ανάπτυξη και τελειοποίηση των πλατφορμών και προτύπων IoT, συμπεριλαμβανομένης της εμφάνισης πρωτοκόλλων και πλαισίων. Διάφοροι οργανισμοί, συμπεριλαμβανομένων βιομηχανικών κοινοπραξιών και φορέων τυποποίησης, εργάστηκαν για τον καθορισμό πρωτοκόλλων επικοινωνίας, μορφοτύπων δεδομένων και πλαισίων ασφαλείας. Αυτή η τυποποίηση διευκόλυνε τη διαλειτουργικότητα μεταξύ διαφορετικών συσκευών και συστημάτων IoT, διευκολύνοντας την ανάπτυξη και την εγκατάσταση λύσεων IoT.

- Τέλη της δεκαετίας του 2000 και αρχές της δεκαετίας του 2010: Ραγδαία ανάπτυξη του βιομηχανικού IoT (IIoT) και των έξυπνων πόλεων, μεταμορφώνοντας τις βιομηχανίες και τα αστικά περιβάλλοντα μέσω εφαρμογών IoT. Το IIoT έφερε επανάσταση στη μεταποίηση και τα logistics, επιτρέποντας την παρακολούθηση σε πραγματικό χρόνο, την προγνωστική συντήρηση και την αυτοματοποίηση των διαδικασιών. Οι έξυπνες πόλεις αξιοποίησαν τις τεχνολογίες IoT για την ενίσχυση των αστικών υποδομών, τη βελτιστοποίηση της διαχείρισης των πόρων, τη βελτίωση της δημόσιας ασφάλειας και την παροχή καλύτερων υπηρεσιών στους πολίτες.

- Δεκαετία του 2010 και μετά: Αυξανόμενη εστίαση στην αντιμετώπιση των προκλήσεων ασφάλειας και προστασίας της ιδιωτικής ζωής στο διευρυνόμενο τοπίο του IoT, με συνεχείς προσπάθειες για την ανάπτυξη ισχυρών μέτρων και πλαισίων. Ο πολλαπλασιασμός των συσκευών IoT και η ποικιλομορφία των εφαρμογών δημιούργησαν νέα τρωτά σημεία ασφαλείας. Ως αποτέλεσμα, δόθηκε ολοένα και μεγαλύτερη έμφαση στην εφαρμογή μέτρων ασφαλείας, όπως ισχυρή πιστοποίηση ταυτότητας, κρυπτογράφηση και ασφαλείς ενημερώσεις υλικού λογισμικού, για την προστασία των συσκευών και των δεδομένων του IoT από απειλές στον κυβερνοχώρο.

- 2020: Επιτάχυνση της υιοθέτησης του IoT λόγω των εξελίξεων στην τεχνολογία και της αυξημένης συνδεσιμότητας παγκοσμίως. Το έτος 2020 παρατηρείται έξαρση της υιοθέτησης του IoT σε όλες τις βιομηχανίες και τους τομείς. Η πανδημία COVID-19 ανέδειξε περαιτέρω τη σημασία του IoT στην υγειονομική περίθαλψη, την απομακρυσμένη παρακολούθηση και τη διατήρηση βασικών υπηρεσιών. Οι οργανισμοί και τα άτομα αξιοποίησαν λύσεις IoT για τη βελτίωση της αποδοτικότητας, την αυτοματοποίηση των διαδικασιών και την ενίσχυση της συνδεσιμότητας.

- Συνεχείς εξελίξεις: Η ιστορία του IoT εξακολουθεί να γράφεται, με συνεχείς καινοτομίες και εξελίξεις που διαμορφώνουν το μέλλον του οικοσυστήματος IoT. Οι αναδυόμενες τεχνολογίες όπως η τεχνητή νοημοσύνη (AI) και η συνδεσιμότητα 5G, αναμένεται να έχουν αντίκτυπο στο τοπίο του IoT, επιτρέποντας ταχύτερη επεξεργασία δεδομένων, πιο έξυπνη λήψη αποφάσεων στην άκρη και μεγαλύτερες δυνατότητες συνδεσιμότητας. Αυτές οι εξελίξεις θα διευρύνουν περαιτέρω τις δυνητικές εφαρμογές και δυνατότητες του IoT σε διάφορους τομείς



Σχήμα 2.2: Συσκευές, ορόσημα του Internet of Things

2.4 Αρχιτεκτονική του Internet of Things

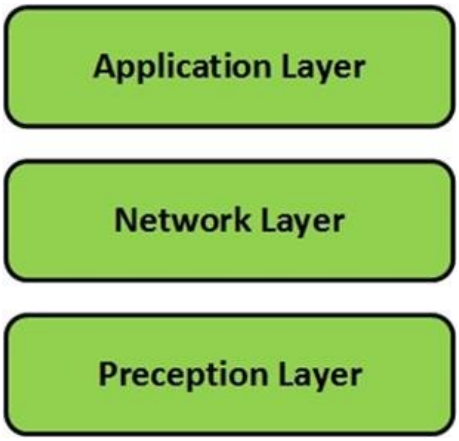
Δεν υπάρχει ενιαία συναίνεση σχετικά με την αρχιτεκτονική για το IoT, η οποία συμφωνείται καθολικά. Διαφορετικές αρχιτεκτονικές έχουν προταθεί από διαφορετικούς ερευνητές.Τις

βασικότερες απο αυτές συνιστούν οι Αρχιτεκτονικές τριών και πέντε επιπέδων. Η πιο βασική αρχιτεκτονική είναι η αρχιτεκτονική τριών επιπέδων , όπως φαίνεται στο Σχήμα 2.3 . Εισήχθη στα πρώτα στάδια της έρευνας στην στον τομέα αυτό. Διαθέτει τρία επίπεδα,

1. Το στρώμα αντίληψης (Perception Layer) είναι το φυσικό στρώμα, το οποίο έχει αισθητήρες για την ανίχνευση και τη συλλογή πληροφοριών σχετικά με το περιβάλλον. Αντιλαμβάνεται ορισμένες φυσικές παραμέτρους ή εντοπίζει άλλα έξυπνα αντικείμενα στο περιβάλλον.

2. Το στρώμα δικτύου (Network Layer) είναι υπεύθυνο για τη σύνδεση με άλλα έξυπνα αντικείμενα, συσκευές δικτύου και διακομιστές. Τα χαρακτηριστικά του χρησιμοποιούνται επίσης για τη μετάδοση και την επεξεργασία δεδομένων αισθητήρων.

3. Το επίπεδο εφαρμογής (Application Layer) είναι υπεύθυνο για την παροχή υπηρεσίες συγκεκριμένης εφαρμογής στον χρήστη. Καθορίζει διάφορες εφαρμογές στις οποίες το Διαδίκτυο των πραγμάτων μπορεί να αναπτυχθεί, για παράδειγμα, έξυπνα σπίτια, έξυπνα πόλεις και έξυπνη υγεία.



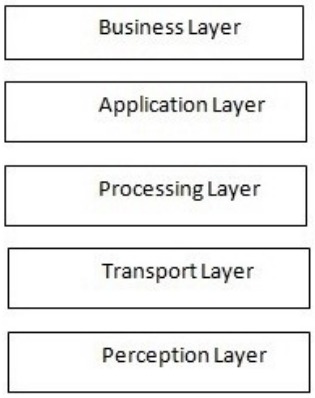
Σχήμα 2.3: 3 Layer architecture Internet of Things

Η αρχιτεκτονική τριών επιπέδων ορίζει την κύρια ιδέα του Internet of Things, αλλά δεν είναι επαρκής για την έρευνα σε IoT, διότι η έρευνα συχνά επικεντρώνεται σε λεπτότερες πτυχές τουInternet of Things. Για το λόγο αυτό, έχουμε πολλές περισσότερες πολυεπίπεδες αρχιτεκτονικές που προτείνονται στη βιβλιογραφία. Μία από αυτές είναι η αρχιτεκτονική πέντε επιπέδων, η οποία περιλαμβάνει επιπλέον την επεξεργασία και τα επιχειρηματικά επίπεδα . Τα πέντε στρώματα είναι η αντίληψη, μεταφορά, επεξεργασία, εφαρμογή και επιχειρηματικά στρώματα. Ο ρόλος των στρωμάτων αντίληψης και εφαρμογής είναι ο ίδιος με την αρχιτεκτονική με τρία στρώματα. Περιγράφουμε τη λειτουργία των υπόλοιπων τριών στρωμάτων.

1. Το στρώμα μεταφοράς (Transport Layer) μεταφέρει τα δεδομένα του αισθητήρα από το στρώμα αντίληψης στο στρώμα επεξεργασίας και αντίστροφα μέσω δικτύων όπως ασύρματα, 3G, LAN, Bluetooth, RFID και NFC.

2. Το στρώμα επεξεργασίας (Processing Layer) είναι επίσης γνωστό ως ενδιάμεσο λογισμικό στρώμα. Αποθηκεύει, αναλύει και επεξεργάζεται τεράστιες ποσότητες δεδομένων που προέρχονται από το στρώμα μεταφοράς. Μπορεί να να διαχειρίζεται και να παρέχει μια ποικιλομορφία υπηρεσιών στο χαμηλότερα στρώματα. Χρησιμοποιεί πολλές τεχνολογίες απο ενότητες όπως βάσεις δεδομένων, cloud computing και big data.

3. Το επιχειρηματικό επίπεδο (Business Layer) διαχειρίζεται ολόκληρο το σύστημα IoT, συμπεριλαμβανομένων των εφαρμογών, των επιχειρηματικών και κερδοσκοπικών μοντέλων, και το απόρρητο των χρηστών.



Σχήμα 2.4: 5 Layer architecture Internet of Things

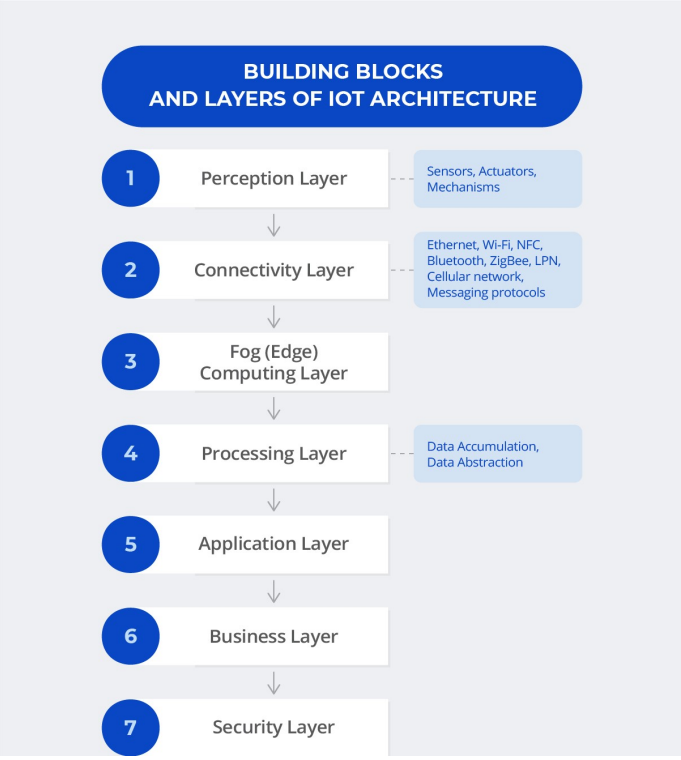
Τέλος σε παρόμοια σημασία με την αρχιτεκτονική των 5 επιπέδων είναι αυτή των 7 επιπέδων οπου σε σχέση με την πρώτη έχουμε τις εξής δύο ειδοποιές διαφορές:

1. Το στρώμα μεταφοράς (Transport Layer) διαχωρίζεται σε δύο επι μέρους στρώματα, το στρώμα σύνδεσης (Connectivity layer) το οποίο αποτελείται εξόλοκλήρου απο τα πρωτόκολλα μεταφοράς που χρησιμοποιούται για τη μεταφορά της πληροφορίας όπως Wi-Fi, NFC, MQTT και το στρώμα (Edge Computing Layer) αποτελείται από το φυσικό υλικό των συσκευών , το ενσωματωμένο λειτουργικό σύστημα που διαχειρίζεται τις διεργασίες στη συσκευή και το λειτουργικό σύστημα της συσκευής, το οποίο είναι το λογισμικό και οι οδηγίες που έχουν προγραμματιστεί στις συσκευές IoT.

2. Το στρώμα ασφάλειας (Security Layer) είναι το στρώμα το οποίο εξασφαλίζει την ασφάλεια ολόκληρου του συστήματος IoT και δραστηριποιείται κυρίως σε 3 τομείς: την

ασφάλεια που αφορα στις συσκευές του IoT, αυτήν που αφορά στην επικοινωνία της πληροφορίας και τέλος την ασφάλεια σε επίπεδο cloud.

[17]



Σχήμα 2.5: 7-layer Architecture of Internet of Things

Μια άλλη αρχιτεκτονική που προτάθηκε από τους Νινγ και Ωανγ είναι εμπνευσμένη από τα επίπεδα επεξεργασίας στον ανθρώπινο εγκέφαλο. Είναι εμπνευσμένη από τη νοημοσύνη και την ικανότητα των ανθρώπων να σκέφτονται, να αισθάνονται, να θυμούνται, να λαμβάνουν αποφάσεις και να αντιδρούν στο φυσικό περιβάλλον. Αποτελείται από τρία μέρη. Το πρώτο είναι ο ανθρώπινος εγκέφαλος, ο οποίος είναι ανάλογος με την επεξεργασία και την μονάδα διαχείρισης δεδομένων ή το κέντρο δεδομένων. Το δεύτερο είναι η σπονδυλική στήλη. ο νωτιαίος μυελός, ο οποίος είναι ανάλογος με το κατανεμημένο δίκτυο δεδομένων των κόμβων επεξεργασίας δεδομένων και των ευφυών πυλών. Τρίτον, το δίκτυο των νεύρων, το οποίο αντιστοιχεί στα στοιχεία δικτύωσης και τους αισθητήρες.

2.5 Επικοινωνία στο IoT

Καθώς το Internet of Things αναπτύσσεται με πολύ γρήγορους ρυθμούς, υπάρχει μεγάλος αριθμός έξυπνων συσκευών που συνδέονται στο Διαδίκτυο. Οι συσκευές IoT τροφοδοτούνται

από μπαταρίες, με ελάχιστους υπολογιστικούς και αποθηκευτικούς πόρους. Λόγω των περιορισμλων της φύσης τους, υπάρχουν διάφορες προκλήσεις επικοινωνίας, όπως:

- Διευθυνσιοδότηση και ταυτοποίηση: δεδομένου ότι εκατομμύρια έξυπνες πραγμάτων θα συνδεθούν στο Διαδίκτυο, θα θα πρέπει να ταυτοποιούνται μέσω μιας μοναδικής διεύθυνσης, στο βάσει της οποίας θα επικοινωνούν μεταξύ τους. Για αυτό, χρειαζόμαστε έναν μεγάλο χώρο διευθύνσεων και μια μοναδική διεύθυνση για κάθε έξυπνο αντικείμενο.

- Επικοινωνία χαμηλής ισχύος: η επικοινωνία δεδομένων μεταξύ των συσκευών είναι ένα έργο που καταναλώνει ενέργεια, ειδικά, η ασύρματη επικοινωνία. Ως εκ τούτου, χρειαζόμαστε μια λύση που να διευκολύνει την επικοινωνία με χαμηλή ισχύ κατανάλωση.

- Πρωτόκολλα δρομολόγησης με χαμηλή απαίτηση μνήμης και αποδοτικά πρότυπα επικοινωνίας.

- Υψηλή ταχύτητα και επικοινωνία χωρίς απώλειες.

- Κινητικότητα : Οι συσκευές IoT συνήθως συνδέονται στο Διαδίκτυο μέσω IP (πρωτόκολλο διαδικτύου). Αυτό το πρωτόκολλο είναι πολύ περίπλοκο και απαιτεί μεγάλη ποσότητα ισχύος και μνήμης από τις συσκευές σύνδεσης. Οι συσκευές IoT μπορούν επίσης να συνδεθούν τοπικά μέσω άλλων δικτύων, τα οποία καταναλώνουν λιγότερη ενέργεια, και να συνδεθούν στο Διαδίκτυο μέσω μιας έξυπνης πύλης. Τέτοια κανάλια επικοινωνίας όπως το Bluetooth, το RFID και το NFC είναι αρκετά δημοφιλή, αλλά είναι περιορισμένης εμβέλειας (έως ένα μερικά μέτρα). Ως εκ τούτου, οι εφαρμογές τους περιορίζονται σε μικρές προσωπικά δίκτυα.

Τα πρωτόκολλα επικοινωνίας τα οποία κυριαρχούν παγκοσμίως σε εφαρμογές IoT ενσωματώνουν όλες τις παραπάνω απαιτήσεις και αυτό οφείλεται στην επιτυχία τους. Παρακάτω παρουσιάζονται συνοπτικά κάποια απο αυτά:

1. **HyperText Transfer Protocol (HTTP)**: Αυτό το πρωτόκολλο αποτέλεσε την απαρχή της επικοινωνίας δεδομένων για τον Παγκόσμιο Ιστό (WWW), οπότε είναι λογικό να χρησιμοποιείται και στον κόσμο του IoT. Ωστόσο, δεν είναι βελτιστοποιημένο γι΄ αυτό λόγω του ότι έχει φτιαχτεί για δύο συστήματα που επικοινωνούν μεταξύ τους κάθε φορά, όχι για περισσότερα, οπότε η σύνδεση πολλών αισθητήρων για τη λήψη πληροφοριών είναι χρονοβόρα και ενεργοβόρα. Επιπλέον το HTTP είναι φτιαγμένο για ένα σύστημα (πελάτης) που στέλνει ένα μήνυμα σε ένα άλλο (διακομιστής). Αυτό καθιστά αρκετά δύσκολη την κλιμάκωση μιας λύσης IoT. Τέλος υπάρχει και η κατανάλωση ενέργειας: Το HTTP βασίζεται στο πρωτόκολλο ελέγχου μετάδοσης (TCP), το οποίο απαιτεί πολλούς υπολογιστικούς πόρους, οπότε δεν είναι κατάλληλο για εφαρμογές που λειτουργούν με μπαταρία.

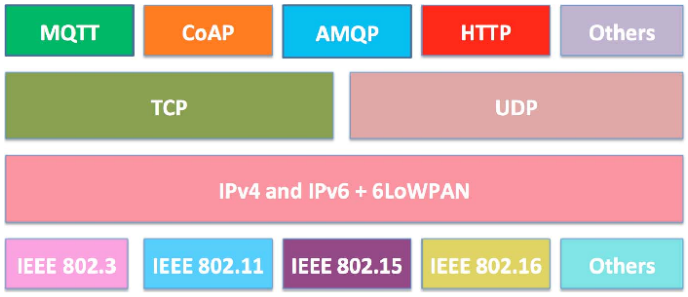
2. **Message Queue Telemetry Transport (MQTT)**: Σχεδιασμένο για να είναι ελαφρύ, ώστε να μπορεί να λειτουργεί σε δίκτυα με πολύ χαμηλό εύρος ζώνης, το MQTT

επιτρέπει την επικοινωνία μεταξύ κόμβων τόσο σε αξιόπιστα όσο και σε αναξιόπιστα δίκτυα. Το MQTT ακολουθεί μια αρχιτεκτονική subscribe / publish, που σημαίνει ότι υπάρχουν κόμβοι (brokers) που καθιστούν τις πληροφορίες διαθέσιμες, ενώ άλλοι (clients) μπορούν να διαβάσουν τις διαθέσιμες πληροφορίες αφού εγγραφούν, αποκτώντας πρόσβαση στην αντίστοιχη διεύθυνση URL.

3. **Data Distribution Service (DDS)**: Παρόμοια με το MQTT, το DDS ακολουθεί μια μεθοδολογία subscribe / publish, με την κύρια διαφορά ότι δεν υπάρχουν brokers. Αυτό σημαίνει ότι όλοι οι publishers (π.χ. αισθητήρες θερμοκρασίας) και οι subscribers (π.χ. κινητά τηλέφωνα) είναι συνδεδεμένοι στο ίδιο δίκτυο. Το δίκτυο αυτό είναι γνωστό ως Global Data Space (GDS) και διασυνδέει κάθε κόμβο με όλους τους άλλους για να αποφεύγονται οι συμφορήσεις.

4. **Constrained Application Protocol (CoAP)**: Το CoAP είναι ένα πρωτόκολλο διαδικτυακής μεταφοράς που χρησιμοποιείται σε περιορισμένα δίκτυα με χαμηλό εύρος ζώνης και χαμηλή διαθεσιμότητα. Ακολουθεί μια αρχιτεκτονική πελάτη/εξυπηρετητή και είναι δομημένο παρόμοια με το HTTP, υποστηρίζοντας το μοντέλο REST: οι διακομιστές καθιστούν διαθέσιμους πόρους με μια διεύθυνση URL και οι πελάτες μπορούν να κάνουν αιτήσεις τύπου GET, POST, PUT και DELETE. Οι σύνδεσμοι επικοινωνίας CoAP είναι 1:1 και βασίζονται στο UDP, οπότε η παράδοση δεν είναι εγγυημένη. Το CoAP είναι φτιαγμένο για να λειτουργεί σε δίκτυα με μεγάλη συμφόρηση, όπου οι κόμβοι δεν διαθέτουν ευφυΐα και δεν λειτουργούν πάντα.

5. **WebSocket**: Συνδεδεμένη με το πρωτόκολλο HTTP, η τεχνολογία WebSocket εγκαθιστά μια σύνδεση TCP μεταξύ ενός προγράμματος περιήγησης και ενός διακομιστή και στη συνέχεια και οι δύο ανταλλάσσουν πληροφορίες μέχρι να κλείσει η σύνδεση.



Σχήμα 2.6: Internet of Things Protocols

[13]

2.6 Εφαρμογές του IoT

Υπάρχουν ποικίλοι τομείς στους οποίους έχουν αναπτυχθεί ευφυείς εφαρμογές. Η προκαταρκτική έρευνα δείχνει τις δυνατότητες του IoT για τη βελτίωση της ποιότητας ζωής στην κοινωνία μας. Ορισμένες χρήσεις των εφαρμογών του IoT είναι ο οικιακός αυτοματισμός, παρακολούθηση της φυσικής κατάστασης, παρακολούθηση της υγείας, προστασία του περιβάλλοντος, έξυπνες πόλεις και βιομηχανικές εφαρμογές.

2.6.1 Έξυπνα σπίτια

Τα έξυπνα σπίτια γίνονται όλο και πιο δημοφιλή σήμερα για δύο λόγους. Πρώτον, οι αισθητήρες και οι τεχνολογίες ενεργοποίησης μαζί με τα ασύρματα δίκτυα έχουν ωριμάσει σημαντικά. Δεύτερον, οι άνθρωποι σήμερα εμπιστεύονται την τεχνολογία για την αντιμετώπιση των ανησυχιών τους σχετικά με την ποιότητα των ζωής και την ασφάλεια των σπιτιών τους. Στα έξυπνα σπίτια, αναπτύσσονται διάφοροι αισθητήρες, οι οποίοι παρέχουν έξυπνες και αυτοματοποιημένες υπηρεσίες στον χρήστη. Βοηθούν στην αυτοματοποίηση των καθημερινών εργασιών και συμβάλλουν στη διατήρηση μιας ρουτίνας για άτομα που έχουν την τάση να ξεχνούν. Βοηθούν στην εξοικονόμηση ενέργειας με το σβήσιμο των φώτων και των ηλεκτρονικών γκάτζετ αυτόματα. Συνήθως χρησιμοποιούμε αισθητήρες κίνησης για αυτόν τον σκοπό. Οι αισθητήρες κίνησης μπορούν επιπλέον να χρησιμοποιηθούν για ασφάλεια.

Η εξοικονόμηση ενέργειας στα έξυπνα σπίτια επιτυγχάνεται συνήθως μέσω αισθητήρων και επίγνωσης του περιβάλλοντος. Οι αισθητήρες συλλέγουν δεδομένα από το περιβάλλον (φως, θερμοκρασία, υγρασία, αέριο και συμβάντα πυρκαγιάς). Αυτά τα δεδομένα από ετερογενείς αισθητήρες διοχετεύονται σε έναν συγκεντρωτή , ο οποίος τα προωθεί στη μηχανή υπηρεσιών με επίγνωση του πλαισίου.Για παράδειγμα, μια εφαρμογή μπορεί να ενεργοποιήσει αυτόματα το κλιματιστικό όταν η υγρασία αυξάνεται. Ή, όταν υπάρχει διαρροή αερίου, μπορεί να σβήσει όλα τα φώτα.

2.6.2 Έξυπνες Πόλεις

Οι έξυπνες πόλεις αποτελούν έναν όρο που αναφέρεται στο σύνολο των εφαρμογών IoT που δραστηριοποιούνται σε αυτη με σκοπό την αυτοματοποιημένη βελτίωση των συνθηκών ζωής των πολιτών σε διάφορους τομείς, μερικοί εκ των οποίων αποτελούν:

1. Έξυπνες μεταφορές. Οι εφαρμογές έξυπνων μεταφορών μπορούν να διαχειρίζονται την καθημερινή κυκλοφορία στις πόλεις χρησιμοποιώντας αισθητήρες και ευφυή συστήματα επεξεργασίας πληροφοριών. Ο κύριος στόχος των ευφυών μεταφορών συστημάτων είναι η ελαχιστοποίηση της κυκλοφοριακής συμφόρησης, η εξασφάλιση της εύκολης και απρόσκοπτης στάθμευσης και η αποφυγή ατυχημάτων με τη σωστή δρομολόγηση της κυκλοφορίας και τον εντοπισμό των επικύνδυνων οδηγών. Οι αισθητήρες που διέπουν αυτού του είδους τις εφαρμογές είναι οι τεχνολογίες GPS για την τοποθεσία, επιταχυνσιόμετρα για την ταχύτητα RFID για την αναγνώριση οχημάτων, αισθητήρες υπερύθρων

για την καταμέτρηση επιβατών και οχημάτων και κάμερες για την καταγραφή της κίνησης οχημάτων και της κυκλοφορίας

2. Έξυπνα συστήματα νερού. Δεδομένης της επικρατούσας λειψυδρίας στα περισσότερα μέρη του κόσμου, είναι πολύ σημαντική η αποτελεσματική διαχειριση των υδάτινων πόρων. Κατά συνέπεια, οι περισσότερες πόλεις επιλέγουν έξυπνες λύσεις που τοποθετούν πολλούς μετρητές στις γραμμές ύδρευσης και στους αγωγούς ομβρίων. Αυτοί οι μετρητές μπορούν να χρησιμοποιηθούν για τη μέτρηση του βαθμού εισροής και εκροής νερού για τον εντοπισμό πιθανών διαρροών. Τέτοια συστήματα χρησιμοποιούνται επίσης σε συνδυασμό με δεδομένα από μετεωρολογικούς δορυφόρους και αισθητήρες νερού ποταμών βοηθώντας στην έγκαιρη και εμπεριστατωμένη πρόληψη απο πλημμύρες.

3. Εξοικονόμηση ενέργειας: Το smart grid είναι πληροφορίες και τεχνολογία επικοινωνίας που επιτρέπει τη σύγχρονη παραγωγή, μεταφορά, διανομή και κατανάλωση ηλεκτρικής ενέργειας. Για να γίνει η παραγωγή, η μεταφορά και η διανομή, η έννοια των έξυπνων δικτύων προσθέτει νοημοσύνη σε κάθε βήμα και επιτρέπει επίσης την αμφίδρομη ροή τοψ ρεύματος (από τον καταναλωτή στον προμηθευτή). Αυτό μπορεί εξοικονομήσει πολλή ενέργεια και να βοηθήσει τους καταναλωτές να κατανοήσουν καλύτερα τη ροή της ενέργειας και τη δυναμική τιμολόγηση. Σε ένα έξυπνο δίκτυο, η παραγωγή ενέργειας είναι κατανεμημένη. Υπάρχουν αισθητήρες που αναπτύσσονται σε όλο το σύστημα για την παρακολούθηση των πάντων. Είναι στην πραγματικότητα ένα κατανεμημένο δίκτυο μικροδικτύων . Τα μικροδίκτυα παράγουν ενέργεια για την κάλυψη των απαιτήσεων των τοπικών περιοχών και μεταδίδουν πίσω την πλεονάζουσα ενέργεια στο κεντρικό δίκτυο. Τα μικροδίκτυα μπορούν επίσης να ζητήσουν ενέργεια από το κεντρικό δίκτυο σε περίπτωση έλλειψης.

Σχήμα 2.7: Smart City

Κεφάλαιο 3

MQTT Protocol

3.1 Εισαγωγή

Το MQTT είναι ένα προτώκολλο επικοινωνίας για το IoT (Internet of things). Έχει σχεδιαστεί ως μια εξαιρετικά ελαφρή επικοινωνία μηνυμάτων της μορφής Publish/Subscribe, το οποίο το καθιστά ιδανικό για απομακρυσμένες συσκευές όπως σένσορες ή στην περίπτωσή μας κάμερες με μικρή επεξεργαστική ισχύς και ελάχιστο bandwidth δικτύου.

Το MQTT σήμερα χρησιμοποιείται σε μια μεγάλη γκάμα βιομηχανιών όπως αυτοκινήτων, κατασκευαστική, πετρελαίου, αερίου. Είναι σαφές ότι στα πλαίσια του Internet of things είναι απαραίτητο ένα τέτοιο προτώκολλο επικοινωνίας το οποίο έχει κατασκευαστεί ειδικά για τέτοιου είδους εφαρμογές.

Ο Andy Stanford-Clark και ο Arlen Nipper ήταν οι πρώτοι οι οποίοι κατασκεύασαν την πρώτη μορφή του MQTT (Messaging Queue Telemetry Transport) με στόχο την παρακολούθηση ροών πετρελαίου. Στόχος τους ήταν ένα bandwidth-efficient πρωτόκολλο επικοινωνίας ελαφρύ το οποίο δεν θα απαιτούσε μεγάλη χρήση ισχύος, αφού οι συσκευές συνδέονταν μέσω επικοινωνίας με δορυφόρο, το οποίο εκείνη την εποχή ήταν αρκετά ακριβό. Παρόλαυτά στη συνέχεια άλλαξε η μορφή επικοινωνίας σε publish-subscribe καναλιού οπότε δέν ήταν πλέον Messaging Queue, παρόλαυτα το όνομα έχει παραμείνει.

Σχήμα 3.1: MQTT logo

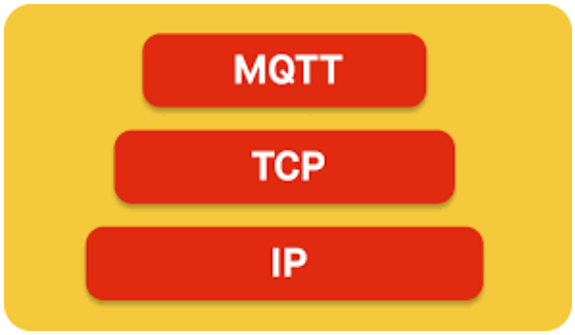
Χαρακτηριστικά του MQTT που το καθιστούν ιδανικό για IoT εφαρμογές:

1. Minimal Overhead για επίτευξή μικρού bandwidth
2. Publish / Subscribe model
3. Σχεδιασμένο για έμπιστη επικοινωνία πάνω σε μη έμπιστα κανάλια επικοινωνίας ή κακής ποιότητας σύνδεσης
4. Ευχρήστο και εύκολα κατανοητό
5. Data Agnostic δηλαδή μπορεί να χρησιμοποιηθεί για να μεταφέρει οποιουδήποτε είδους πληροφορία

3.2 Βασικά στοιχεία του MQTT

3.2.1 Στρώμα εφαρμογής

Αρχικά το πρωτόκολλο επικοινωνίας MQTT στο στρώμα της εφαρμογής γίνεται πάνω απο το πρωτόκολλο TCP/IP και η συνήθης θύρα επικοινωνίας έιναι η 1883 (ή πιο σπάνια η 8883)



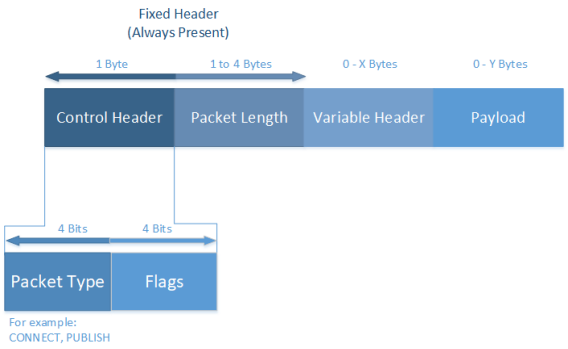
Σχήμα 3.2: MQTT στρώμα εφαρμογής

3.2.2 Δομή Μηνύματος

Ένα μήνυμα στο πρωτόκολλο MQTT ενθυλακώνεται δημιουργώντας τα εξής μέρη:

1. Control Header το οποίο έχει μέγεθος ενός byte χωρίζεται σε δυο επι μέρους σημεία
   (α′) Packet Type μεγέθους 4bit που υποδηλώνει το action π.χ. Connect,Publish
   (β′) Flags μεγέθους 4bit που ανάλογα το action δηλώνουν την επιπρόσθετη λειτουργία
2. Το μήκος του πακέτου, μεγέθους 1-4 bytes
3. Την μεταβλητή επικεφαλίδα η οποία μπορεί να διαφέρει το μεγεθός της

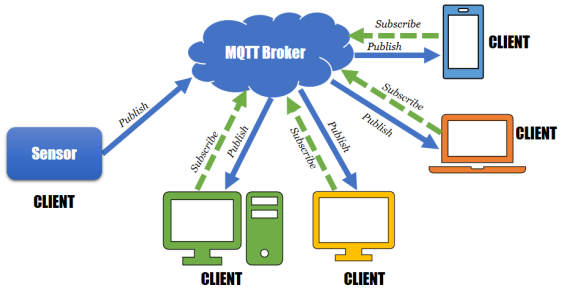
4. Το payload, την ουσιαστική πληροφορία δηλαδή που θέλουμε να μεταφέρουμε.



Σχήμα 3.3: MQTT Δομή μηνύματος

3.3 Τρόπος ανταλλαγής μηνυμάτων

Τα βασικά μέρη τα οποία συμμετέχουν στο πρωτόκολλο MQTT είναι οι Clients και ο Broker. Η βασική λειτουργία των Client είναι είτε να γράφουν είτε να διαβάζουν απο συγκεκριμένα κανάλια επικοινωνίας γνωστά και ως Topics, ενώ η βασική λειτουργικότητα του broker είναι να αναλαμβάνει τη διαχείριση των μηνυμάτων στα topics και να εξασφαλίζει τόσο την σωστή επικοινωνία μεταξύ εφαρμογών που χρησιμοποιούν το ίδιο κανάλι επικοινωνίας όσο και να αναλαμβάνει τη διεχείριση προβλημάτων όπως αδυναμία σύνδεσης στο διαδίκτυο. Παρακάτω αναλύουμε καλύτερα τις επι μέρους λειτουργίες:



Σχήμα 3.4: MQTT βασική λειτουργία

3.3.1 MQTT Broker

Ένας MQTT Broker είναι μια κεντρική οντότητα που δρα όπως ένας μεσίτης, δηλαδή αρχικά κοιτάζει το background των ενδιαφερόμενων και αφού συμπεράνει ότι όλοι οι κανόνες

είναι εντάξει ξεκινάει τη συνναλαγή. Ο MQTT Broker κάνει ακριβώς το ίδιο αλλα αντί για συναλλαγές με χρήματα αφορά σε ανταλλαγές μηνυμάτων. Συγκεκριμένα ένας MQTT Broker διανέμει τις συναλλαγές ως εξής:

1. Επιτρέπει στις συσκευές γνωστές και ως clients να ζητήσουν σύνδεση.
2. Αυθεντικοποιεί τις συσκευές βασισμένο στις πληροφορίες σύνδεσης που του αποστέλλουν
3. Μόλις η συσκευή αυθεντικοποιηθεί ο MQTT Broker φροντίζει ώστε να μπορούν να στέλνουν ή να διαβάζουν μηνύματα σε ένα topic ασφαλώς χρησιμοποιώντας transport Layer Security (TLS)
4. Αποθηκεύει τα μηνύματα στον server ώστε σε περίπτωη ανεπιτυχούς σύνδεσης μίας συσκευής ανάγνωσης να μην χαθεί η πληροφορία

Η σημαντικότερη λειτουργία ενός MQTT Broker είναι το decoupling των συσκευών που επικοινωνούν προσθέτοντας ένα ενδιάμεσο στρώμα με αρχιτεκτονική η οποία είναι scalable για μεγάλο αριθμό συσκευών.Οι πιο απαιτητικές λειτουργίες γίνονται στον MQTT Broker με αποτέλεσμα στους clients να επαφίεται το κομμάτι ελάχιστης υπολογιστικής πολυπλοκότητας με ελάχιστο bandwidth.

Οι MQTT Broker διακρίνονται σε δυο ειδών κατηγορίες

1. Managed Broker στους οποίους δεν απαιτείται κανένα τοπικό στήσιμο ώστε να μπορούν να επικοινωνούν με MQTT, αφού χρησιμοποιεί ο χρήστης την εκάστοτε υπηρεσία παροχής τους του.
2. Self-Hosted Broker, όπου όπως αναφέρεται και στο όνομα εγκαθίσταται στον τοπικό σέρβερ με την IP του χρήστη. Η διαδικασία αν και δεν είναι δύσκολη απαιτεί βαθιά κατανοήση του συστήματος για την εφαρμογή όλων των παραμέτρων ορθά. Δημοφιλείς τέτοιοι MQTT Broker είναι το Moquitto και το HiveMQ

Παρακάτω φαίνεται μια λίστα με τους πιο δημοφιλείς MQTT Broker και των δύο κατηγοριών.

3.3.2 MQTT Client

Ένας MQTT Client μπορεί να είναι οποιαδήποτε smart device όπως ένα έξυπνο ρολόι, μια κάμερα ή ένας αισθητήρας. Οι δυνατότητες ενός MQTT Client στα πλαίσια του MQTT πρωτοκόλλου είναι οι εξής:

1. Συνδέονται με τον Broker μέσω ενός ονόματος χρήστη και ενός κωδικού
2. Μπορούν να κάνουν publish σε ένα topic δηλαδή να γράψουν κάποια πληροφορία σε αυτό.

Type	Address and Port	WebSocket Support	SSL Support	Scalability	
AWS IoT Core MQTT	Managed	Dynamically assigned	Yes, port=443	Yes, port=8883	Auto Scale
Mosquitto	Self-hosted and Managed	test.mosquitto.org	Yes, port=8081,8080	Yes, port=8883,8884	Scale horizontally making bridges
Mosca/Aedes	Self-hosted and Managed	test.mosca.io	Yes, port=3000	Yes, port=8883	Horizontally and vertically
HiveMQ	Self-hosted and Managed	broker.hivemq.com	Yes, port=8000,443	Yes, port=8883	Yes, Broker-Clustering
VerneMQ	Self-hosted and Managed	self-assigned	Yes, port=9001,9002	Yes, port=8883	Both horizontal and vertical
Azure IoT Hub	Managed	Dynamically assigned	Yes, port=443	Yes, port=8883	Auto Scale
EMQ X	Self-hosted	Self-assigned	Yes, port=8083, 8084	Yes, port=8883	Clustering
ejabberd	Self-hosted and managed	Self-assigned and dynamically allocated	Yes	Yes	Clustering

Σχήμα 3.5: Λίστα δημοφιλών MQTT Broker

3. Μπορούν να κάνουν subscribe σε ένα topic δηλαδή να διαβάσουν την πληροφορία που έγραψε κάποιος άλλος MQTT Client

Ο όρος topic αναφέρεται σε ένα κανάλι επικοινωνίας στο οποίο γράφουν ή διαβάζουν αντίστοιχα οι MQTT Clients. Ο broker χωρίζει τα κανάλια και όποτε σε ένα εξ αυτών γίνεται publish ένα μήνυμα, αναλαμβάνει την αποστολή αυτου του μηνύματος σε όλους τους subscribers του topic.

[7]

3.4 Παράμετροι επικοινωνίας

Όπως και σε κάθε πρωτόκολλο επικοινωνίας στο οποίο επικοινωνούν μεταξύ τους πολλές διαφορετικές συσκευές πρέπει να οριστούν κάποιες παράμετροι της επικοινωνίας τους ώστε να εναρμονίζονται όλα τα μέρη. Στην περίπτωση τοψ MQTT οι πιο βασικές παράμετροι επικοινωνίας είναι η αυθεντικοποίηση του client, η θύρα στην οποία γίνεται η επικοινωνία (συνήθως 1883) και η διεύθυνση του broker. Παρακάτω θα δούμε καποιες άλλες παραμέτρους οι οποίες χρησιμοποιούνται για τον καθορισμό της σύνδεσης:

3.4.1 MQTT QoS (Quality of Service)

Η Quality of Service ή εν συντομία QoS αφορά στην ποιότητα εξυπηρέτησεις μηνυμάτων κατα την αποστολή στο πρωτόκολλο και χωρίζεται σε τρείς βασικές κατηγορίες. Το ποιά εξ αυτων θα χρησιμοποιηθεί το καθορίζει ο client τη στιγμή που κάνει subscribe στο εκάστοτε topic και είναι σε αύξουσα σειρά εμπιστοσύνης ή ασφάλειας παράδοσης.

QoS=0 (At Most Once)

Η πιο απλή περίπτωση, κατα την οποία ο subscriber αποστέλλει την πληροφορία στον broker μην έχοντας κάποια περαιτέρω δέσμευση (Fire and Forget)

Σχήμα 3.6: Λίστα δημοφιλών MQTT Broker

QoS=1 (At Least Once)

Σε αυτή την περίπτωση ο broker επιβεβαιώνει ότι έλαβε την αποστολή του μηνύματος απο τον subscriber αποστέλλοντας τουλάχιστον ένα μήνυμα επιβεβαίωσης ACK Message με το αντίστοιχο πακέτο PUBACK.

Σχήμα 3.7: Λίστα δημοφιλών MQTT Broker

QoS=2 (Excactly Once)

Σε αυτή την περίπτωση, η παράδοση του μηνύματος στον subscriber απο τον broker είναι εγγυημένη, χωρίς την επαναποστολή αντιγράφων. Ο Publisher αποστέλλει μήνυμα στον Broker με συγκεκριμένο ID. Ο Publisher αποθηκεύει το μήνυμα που εως ότου λάβει απάντηση PUBREC από τον Broker. Όταν ο Broker απαντήσει με το PUBREC που περιέχει το ίδιο ID που έλαβε. Αφού λάβει το μήνυμα, ο Publisher στέλνει το PUBREL με το ίδιο ID. Ο Broker αναλαμβάνει να αποθηκεύσει το αντίγραφο του μηνύματος μέχρι να λάβει το μήνυμα PUBREL.

Μόλις ο Βροκερ το λάβει έχει το δικαίωμα να διαγράψει το αντίγραφο του μηνύματος και αποστέλλει το τελικό μήνυμα PUBCOMP για την ολοκλήρωση της συναλλαγής.

Σχήμα 3.8: Λίστα δημοφιλών MQTT Broker

3.4.2 Keep-Alive

Το Keep-Alive ορίζεται και αυτό απο την μεριά του client και αφορά στο μέγιστο διάστημα για το οποίο θα παραμένει ανοικτή η επικοινωνία μεταξυ client και broker. Πιό συγκεκριμένα ορίζεται ως το μέγιστο δυνατό χρονικό διάστημα το οποίο μπορεί να παρέλθει απο τη στιγμή που ο Client στέλνει ένα πακέτο ελέγχου (το οποίο γίνετια σε οποιαδήποτε αλληλεπίδραση του client και του broker) μέχρι την αποστολή οτυ επόμενου. Άρα είναι ένα cut-off time στο οποίο αν ο client δεν έχει πραγματοποιήσει κάποια επικοινωνία με τον broker η σύνδεση τους κλείνει απο την μεριά του broker.

3.4.3 LWT (Last Will and Testament)

Το LWT είναι ένα PUBLISH μήνυμα που στέλνει ο Publisher στον Broker το οποίο περιγράφει τις ενέργεις που θα ακολουθηθούν σε περίπτωση που ο Publisher χάνει τη συνδεσή του με τον Broker , πιθανώς από πρόβλημα σύνδεσης δικτύου. Σε αυτή τη περίπτωση ανάλογα με την πληροφορία ο Broker αποστέλλει το LWT μήνυμα σε όλους τους Clients που έχυν εγγραφεί στο topic που έγραφε ο αποσυνδεδεμένος. Τέλος να επισημανθεί ότι ανάλογα με την χρισιμότητα της εφαρμογής μέσα στο LWT μπορεί να γραφεί το τι QoS θα ακολουθηθεί. [6]

Κεφάλαιο 4

Machine Learning

4.1 Εισαγωγή

Machine Learning ή μηχανική μάθηση είναι υποπεδίο της επιστήμης των υπολογιστών, που αναπτύχθηκε από τη μελέτη της αναγνώρισης προτύπων και της υπολογιστικής θεωρίας μάθησης στην τεχνητή νοημοσύνη. Η μηχανική μάθηση διερευνά τη μελέτη και την κατασκευή αλγορίθμων που μπορούν να μαθαίνουν από τα δεδομένα και να κάνουν προβλέψεις σχετικά με αυτά. Τέτοιοι αλγόριθμοι λειτουργούν κατασκευάζοντας μοντέλα από πειραματικά δεδομένα, προκειμένου να κάνουν προβλέψεις βασιζόμενες στα δεδομένα ή να εξάγουν αποφάσεις που εκφράζονται ως το αποτέλεσμα.

4.2 Ορισμός

Ο Tom M. Mitchell πρότεινε έναν πιο επίσημο ορισμό που χρησιμοποιείται ευρέως: «Ένα πρόγραμμα υπολογιστή λέγεται ότι μαθαίνει από εμπειρία E ως προς μια κλάση εργασιών T και ένα μέτρο επίδοσης Π, αν η επίδοσή του σε εργασίες της κλάσης T, όπως αποτιμάται από το μέτρο P, βελτιώνεται με την εμπειρία E. Αυτή η οριοθέτηση των εργασιών που αφορούν τη μηχανική μάθηση προσφέρει μια ουσιαστικά λειτουργική οριοθέτηση αντί να ορίζει τον τομέα σε γνωστικούς όρους. Αυτό ακολουθεί την πρόταση του Alan Turing στη δική του εργασία Computing Machinery and Intelligence, στην οποία η ερώτηση ¨Μπορούν οι μηχανές να σκέφτονται·' αντικαθίσταται από την ερώτηση ¨Μπορούν οι μηχανές να κάνουν αυτό που κάνουμε εμείς (ως σκεπτόμενα οντα);' [19]

4.3 Ιστορια του Machine Learning

Μέχρι τις αρχές της δεκαετίας του 1960, είχε αναπτυχθεί από την εταιρεία Raytheon μια πειραματική ¨μηχανή μάθησης' με μνήμη στοιχείων που είχαν εισαχθεί μέσω τρυπητής ταινίας, η οποία ονομαζόταν CyberTron, για να αναλύει σήματα ηχοβολίας, ηλεκτροκαρδιογραφήματα και μοτίβα ομιλίας χρησιμοποιώντας αρχές αναδραστικής μάθησης. Εκπαιδευόταν επανειλημμένα από έναν ανθρώπινο χειριστή/δάσκαλο για την αναγνώριση μοτίβων και εξοπλιζόταν με ένα

κουμπί ¨λάθος' για να το κάνει να αναθεωρήσει τις λανθασμένες αποφάσεις. Ένας βιβλίο, σημείο αναφοράς για την έρευνα στην μηχανική μάθηση κατά τη διάρκεια της δεκαετίας του 1960 ήταν το βιβλίο του Nilsson για τις Μηχανές Μάθησης, που ασχολείται κυρίως με τη μηχανική μάθηση για ταξινόμηση.Το ενδιαφέρον που σχετίζεται με την αναγνώριση προτύπων συνέχισε και στη δεκαετία του 1970, όπως περιγράφεται από τους Duda και Hart το 1973. Το 1981 δόθηκε μια αναφορά για χρήση στρατηγικών εκπαίδευσης έτσι ώστε το νευρωνικό δίκτυο να μπορεί να αναγνωρίζει 40 χαρακτήρες απο το terminal.

Η μηχανική μάθηση στις μέρες μας έχει δύο στόχους, ένας είναι η ταξινόμηση δεδομένων βασισμένη σε μοντέλα που έχουν αναπτυχθεί, ο άλλος σκοπός είναι η πρόβλεψη μελλοντικών αποτελεσμάτων βασισμένων σε αυτά τα μοντέλα. Ένας υποθετικός αλγόριθμος στοχευμένος στην ταξινόμηση δεδομένων μπορεί να χρησιμοποιεί την υπολογιστική όραση για κρεατοελιές και σε συνδυασμό με transfer learning μπορεί να εκπαιδευτεί προκειμένου να τις ταξινομεί σε καρκινογενείς και μη. Ένα ακόμη παράδειγμα είναι ένας αλγόριθμος μηχανικής μάθησης για την συναλλαγές μετοχών μπορεί να βοηθά τον χρήστη για μελλοντικές πρόβλεψεις τιμών των μετοχών ή ακόμη και επενδυτικών στρατηγικών. [12]

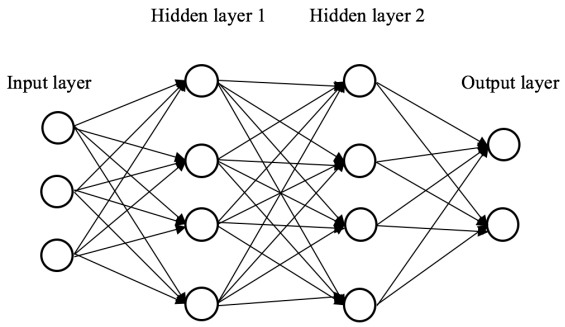
4.4 Νευρωνικά Δίκτυα

4.4.1 Γενικά Στοιχεία για τα Νευρωνικά Δίκτυα

Τα νευρωνικά δίκτυα, επίσης γνωστά ως τεχνητά νευρωνικά δίκτυα (Artificial Neural Networks) ή προσομοιωμένα νευρωνικά δίκτυα (Simulated Neural Networks), είναι ένα υποσύνολο της μηχανικής μάθησης και βρίσκονται στην καρδιά των αλγορίθμων βαθιάς μάθησης. Το όνομά τους και η δομή τους είναι εμπνευσμένα από τον ανθρώπινο εγκέφαλο, μιμούμενα τον τρόπο με τον οποίο οι βιολογικοί νευρώνες αλληλεπιδρούν.

Τα νευρωνικά δίκτυα αποτελούνται από επίπεδα κόμβων, περιλαμβάνοντας ένα επίπεδο εισόδου, ένα ή περισσότερα κρυφά επίπεδα και ένα επίπεδο εξόδου. Κάθε κόμβος ή τεχνητός νευρώνας συνδέεται με έναν άλλον και έχει ένα βάρος και ένα threshold. Εάν η έξοδος οποιουδήποτε ατομικού κόμβου είναι άνω της καθορισμένης τιμής του threshold, αυτός ο κόμβος ενεργοποιείται και στέλνει δεδομένα στο επόμενο επίπεδο του δικτύου. Διαφορετικά, δεν περνά κανένα δεδομένο στο επόμενο επίπεδο του δικτύου.

Τα νευρωνικά δίκτυα βασίζονται στα δεδομένα εκπαίδευσης για να ¨μάθουν' και να βελτιώσουν την ακρίβειά τους με την πάροδο του χρόνου. Μόλις αυτοί οι αλγόριθμοι ,βελτιστοποιήσουν τις τιμές τών βαρών, μπορούν να αναγνωρίζουν με ακρίβεια μοτίβα στα οποία εκπαιδεύτηκαν με αποτέλεσμα να γίνονται ισχυρά εργαλεία στην επιστήμη υπολογιστών, επιτρέποντάς μας να ταξινομούμε και να ομαδοποιούμε δεδομένα με μεγάλη ταχύτητα. Οι εργασίες στην αναγνώριση φωνής ή εικόνας μπορούν να διαρκέσουν λεπτά αντί ωρών σε σύγκριση με



Σχήμα 4.1: Νευρωνικό δίκτυο με 2 κρυφά επίπεδα

τη αναγνώριση από εμπειρογνώμονες. Ένα από τα πιο γνωστά νευρωνικά δίκτυα, που χρησιμοποιούμε καθημερινά είναι ο αλγόριθμος αναζήτησης της Google PageRank.

4.4.2 Λειτουργία Νευρωνικών Δικτύων

Η λειτουργία του νευρωνικού δικτύου προκύπτει αν σκεφτούμε κάθε κόμβο του σαν γραμμικό μοντέλο, που όπως αναφέρθηκε προηγουμένως, η είσοδός του προκύπτει απο τις εξόδους του προηγούμενου στρώματος πολλαπλασίασμένες με το βάρος του, προσθέτοντας το threshold (ή bias) για να καταλήξουμε στον τύπο:

$$input = \sum w_i x_i + b$$

Η έξοδος του Νευρωνικού δικτύου προκύπτει απο μία σύνάρτηση f της εισόδου του. Ανάλογα με την χρήση του Νευρωνικού Δικτύου η συνάρτηση αυτή f μπορέι να πάρει διάφορες τιμές. Μία είναι να την θεωρήσουμε σαν την step function δηλαδή:

$$f(x) = \begin{cases} 0 & x \leq 0 \\ 1 & 0 \leq x \end{cases}$$

Μια άλλη συνάρτηση εξόδου η οποία χρησιμοποιείται ειδικά σε περιπτώσεις όπου θέλουμε να χρησιμοποιήσουμε αναλυτικές μεθόδους για την εύρεση βέλτιστων βαρών όπως τον αλγόριθμο back propagation όπου για τη χρήση παραγώγων του αλγορίθμου χριαζόμαστε μια συνεχή συνάρτηση είναι η σιγμοειδής συνάρτηση:

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

Τα δεδομένα του εκάστοτε προβλήματος (μέσω κατάλληλης μετάφρασης σε αριθμούς στο διαστημα $[0, 1]$) δίνονται στο νευρωνικό δίκτυο σαν τιμές output του πρώτου στρώματος και η απάντηση ή τα ζητούμενα προκύπτουν απο τις τιμές του τελευταίου στρώματος κόμβων (μέσω

πάλι κατάλληλης μετάφρασης στη μορφή που μας είναι κατανοητή).
Η εκπαίδευση του κάθε κόμβου του Νευρωνικού Δικτύου , η εύρεση ιδανικών βαρών για την επίλυση του εκάστοτε προβλήματος , με βάση ένα προσχεδιασμένο dataset γίνεται μέσω της διαδικασίας ελαχιστοποίησης μια συνάρτησης κόστους οπως ονομάζεται. Αυτή η συνάρτηση κόστους μας δείχνει πόσο κοντά είναι η πρόβλεψη του Νευρωνικού Δικτύου σε σχέση με τα πραγματικά δεδομένα, και για την βελτίωση των βαρών χρησιμοποιύνται αλγόριθμοι όπως back propagation που προαναφέρθηκε. Μια συνιθησμένη συνάρτηση κόστους είναι η διασπορά των εξόδων του νευρωνικού στις γνωστές εισόδους με τις γνωστες (και σωστές) εξόδους.

$$CostFunction = \frac{1}{m}\sum_{1}^{m}(y_i - y_i')^2$$

4.4.3 Τύποι Νευρωνικών Δικτύων

1. Multi-layer Perceptrons στα οποία αναφέρονται τα παραπάνω παραδείγματα με τη σιγμοειδή συνάρτηση ενεργοποίησης αφού τα περισσότερα πραγματικά προβλήματα δεν είναι γραμμικά. Η χρήση τους γίνεται μεταξύ άλλων για υπολογιστική όραση, αναγνώριση προτύπων, επεξεργασία φυσικής γλώσσας.

2. Convolutional neural networks (CNNs) ή συνελικτικά νευρωνικά δίκτυα. Είναι παρόμοια με τα παραπάνω με βασική τους διαφορά τη χρήση της μαθηματικής πράξης της συνέλιξης αντί για τον πολλαπλασιασμό πινάκων σε τουλάχιστον ένα απο τα επίπεδά τους. Οι εφαρμογές τους είναι κυρίως σε αναγνωριση εικόνας και βίντεο, συστήματα προτάσεων και κατηγοριοποίηση εικόνων. Για αυτό η χρήση τους αποτελείται απο την μετάφραση εικόνας σε πίνακα pixel στα οποία αντείθεται ένας αριθμός ανάλογα με το χρώμα και στη συνέχεια σε κάποιο απο τα επίπεδα εφαρμόζεται συνέλιξη αντί πολλαπλασιασμού και για αυτό έχουν ακι το όνομά τους. Αξίζει να σημειωθεί ότι αυτή η καταλληλότητά τους για εφαρμογές εικόνων κάνουν τα συνελικτικά δίκτυα βασικό μέρος του μοντέλου μηχανικής μάθησης που χρησιμοποιούμε για την αναγνώριση μάσκας σε βίντεο.

3. Recurrent neural networks (RNNs) ή αναδρομικά νευρωνικά δίκτυα των οποίων η ειδοποιώς διαφορά είναι τα feedback loops στα οποία η έξοςδος ενός μετέπειτα στρώματος κόμβων ξαναδίνεται σαν είσοδος σε προηγούμενα. Αυτό επιτρέπει δυναμική συμπεριφορά στο νευρωνικό δίκτυο κάτι που το κάνει κατάλληλο για αναγνώριση χρονοδιαγραμμάτων σαν είσοδο , π.χ. για χρηματιστηριακές προβλέψεις ή πρόβλεψη πωλήσεων αν και η χρήση τους δέν περιορίζεται εκεί αφού χρησιμοποιείται για αναγνώριση γραφικού χαρακτήρα ή ομιλίας.

4.4.4 Convolutional neural networks (CNNs)

Τα συνελικτικά νευρωνικά δίκτυα διακρίνονται από άλλα νευρωνικά δίκτυα χάρη στην ανώτερη απόδοσή τους σε εισόδους εικόνας, ομιλίας ή ήχου. Διαθέτουν τρία βασικά είδη

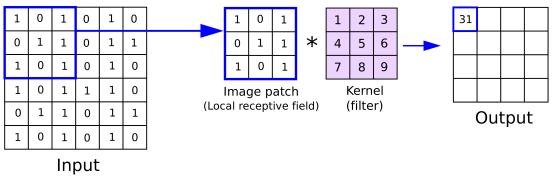
στρωμάτων, τα οποία είναι:

1. Συνελικτικό στρώμα
2. Στρώμα συσσώρευσης (Pooling)
3. Πλήρως συνδεδεμένο (Fully-connected) στρώμα

Το συνελικτικό στρώμα είναι το πρώτο στρώμα ενός συνελικτικού δικτύου. Τα συνελικτικά στρώματα μπορούν να ακολουθούνται από επιπλέον συνελικτικά στρώματα ή στρώματα συσσώρευσης, παρόλα αυτά το πλήρως συνδεδεμένο στρώμα είναι το τελευταίο στρώμα. Με κάθε στρώμα, το CNN αυξάνει την πολυπλοκότητά του, αναγνωρίζοντας μεγαλύτερα τμήματα της εικόνας. Τα πρώτα στρώματα επικεντρώνονται σε απλά χαρακτηριστικά, όπως τα χρώματα και οι άκρες. Καθώς η εικονική δεδομένα προχωρούν μέσα από τα στρώματα του CNN, αναγνωρίζονται μεγαλύτερα στοιχεία ή σχήματα του αντικειμένου μέχρι να αναγνωριστεί το επιθυμητό αντικείμενο.

1. Convolutional Layer
   Το βασικό σημείο του ενός συνελικτικού στρώματος είναι ο τρόπος με τον οποίο μεταφράζεται ο πίνακας των pixel στο επόμενο στρώμα κόμβων και αυτό γίνεται μέσω της μαθηματικής πράξης της συνέλιξης του πίνακα εισόδου με τον Kernel πίνακα ο οποίος παίρνει την τιμή του ανάλογα με τα στοιχεία της εικόνας που θέλουμε να δώσουμε έμφαση. Αν ο πίνακας f αντισοιχεί στην είσοδο του συνελικτικού στρώματος (αν είναι το πρώτο στρώμα θα είναι η ίδια η εικόνα) και ο πίνακας h αντιστοιχεί στο επιλεγμένο Kernel τότε η έξοδος του συνελικτικού στρώματος θα δίνεται απο:

$$G[m,n] = (f * h)[m,n] = \sum_{j=1}^{m} \sum_{k=1}^{n} h[j,k] f[m-j, n-k]$$



Σχήμα 4.2: Συνελικτικό Στρώμα CNN

2. Pooling Layer Τα στρώματα συσσώρευσης (Pooling), επίσης γνωστά ως στρώματα μείωσης διαστάσης (downsampling), εκτελούν μείωση διαστασης, μειώνοντας τον αριθμό των παραμέτρων της εισόδου. Παρόμοια με το συνελικτικό στρώμα, η λειτουργία συσσώρευσης διασχίζει ένα φίλτρο σε όλη την είσοδο, αλλά η διαφορά είναι ότι αυτό το φίλτρο δεν έχει κανένα βάρος. Αντ άυτού, το Kernel εφαρμόζει μια συναθροιστική

συνάρτηση στις τιμές μέσα στα frame, γεμίζοντας τον πίνακα εξόδου. Υπάρχουν δύο βασικοί τύποι συσσώρευσης:
-Μέγιστη συσσώρευση (Max pooling): Καθώς το φίλτρο κινείται στην είσοδο, επιλέγει το εικονοστοιχείο με τη μέγιστη τιμή και το στέλνει στον πίνακα εξόδου. Αυτή η προσέγγιση τείνει να χρησιμοποιείται συχνότερα σε σχέση με τη μέση συσσώρευση.
-Μέση συσσώρευση (Average pooling): Καθώς το φίλτρο κινείται στην είσοδο, υπολογίζει τη μέση τιμή μέσα στο frame και το στέλνει στον πίνακα εξόδου.

3. Fully Connected Layer
   Το όνομα του στρώματος πλήρους σύνδεσης περιγράφει το τελικό στρώμα του CNN Οι τιμές των εικονοστοιχείων της εισόδου δεν συνδέονται άμεσα με το στρώμα εξόδου στα μερικώς συνδεδεμένα στρώματα. Ωστόσο, στο στρώμα πλήρους σύνδεσης, κάθε κόμβος στο στρώμα εξόδου συνδέεται άμεσα με έναν κόμβο στο προηγούμενο στρώμα.

4.5 Transfer Learning

Το Transfer Learning (TL) είναι μια τεχνική στη μηχανικής μάθησης κατά την οποία η γνώση που αποκτήθηκε από μια εκπαίδευση ενος μοντέλου μηχανικής μάθησης επαναχρησιμοποιείται για να ενισχύσει την απόδοση στο μοντέλο για ένα παρεμφερές πρόβλημα .Για παράδειγμα, για την ταξινόμηση εικόνων, η γνώση που αποκτήθηκε κατά την εκμάθηση της αναγνώρισης αυτοκινήτων θα μπορούσε να εφαρμοστεί κατά την προσπάθεια αναγνώρισης φορτηγών. Το θέμα αυτό σχετίζεται με την βιβλιογραφία για τη μεταφορά της μάθησης, αν και οι πρακτικοί δεσμοί μεταξύ των δύο πεδίων είναι περιορισμένοι. Η επαναχρησιμοποίηση/μεταφορά πληροφοριών από εργασίες που έχουν διδαχθεί προηγουμένως σε νέες εργασίες έχει τη δυνατότητα να βελτιώσει σημαντικά την αποτελεσματικότητα της μάθησης

4.5.1 Ιστορικά στοιχεία τουTransfer Learning

Το 1976, οι Bozinovski και Fulgosi δημοσίευσαν μια εργασία που ασχολήθηκε με το Transfer Learning στην εκπαίδευση νευρωνικών δικτύων. Το 1981, μια έκθεση εξέτασε την εφαρμογή της μάθησης μεταφοράς σε ένα σύνολο δεδομένων από εικόνες που αναπαριστούσαν γράμματα τερματικών υπολογιστών, αποδεικνύοντας πειραματικά τη θετική και την αρνητική μάθηση μεταφοράς.

Το 1993, ο Pratt διατύπωσε τον αλγόριθμο μεταφοράς με βάση τη διακριτική ικανότητα (DBT discriminability-based transfer) .Το 1997, οι Pratt και Thrun φιλοξενούμενοι επιμελήθηκαν ένα ειδικό τεύχος του περιοδικού Machine Learning αφιερωμένο στο Transfer Learning και μέχρι το 1998, το πεδίο είχε εξελιχθεί και περιελάμβανε μάθηση πολλαπλών εργασιών, μαζί με πιο επίσημες θεωρητικές βάσεις.Το Learning to Learn, που επιμελήθηκαν οι Thrun , Pratt, είναι μια ανασκόπηση του θέματος το 1998.Η μάθηση μεταφοράς έχει εφαρμοστεί στη γνωστική επιστήμη. Ο Pratt ήταν φιλοξενούμενος εκδότης ενός τεύχους του Connection Science για την επαναχρησιμοποίηση νευρωνικών δικτύων μέσω μεταφοράς το 1996.

Ο Ng δήλωσε στο σεμινάριό του στο NIPS 2016 ότι το Transfer Learning θα γίνει ο επόμενος μοχλός εμπορικής επιτυχίας της μηχανικής μάθησης μετά την επιβλεπόμενη μάθηση (Supervised Learning).

4.5.2 Μαθηματικό υπόβαθρο του Transfer Learning

Ένας τομέας D ορίζεται από δύο μέρη, έναν χώρο χαρακτηριστικών X και μια αθροιστική κατανομή πιθανότητας $P(X)$, όπου $X = \{x_1, ..., x_n\} \in X$. Για παράδειγμα, εάν η εφαρμογή μηχανικής μάθησης έχει ως στόχο την ταξινόμηση ελαττωμάτων μονάδων λογισμικού και κάθε μετρική των μοναδων αυτών λαμβάνεται ως χαρακτηριστικό γνώρισμα, τότε x_i είναι το i-οστό διάνυσμα χαρακτηριστικών (instance) που αντιστοιχεί στην i-οστή μονάδα λογισμικού, n είναι ο αριθμός των διανυσμάτων χαρακτηριστικών στο X, X είναι ο χώρος όλων των πιθανών διανυσμάτων χαρακτηριστικών και x είναι ένα συγκεκριμένο δείγμα μάθησης. Για έναν δεδομένο τομέα D, μια διαδικασία μάθησης T ορίζεται από δύο μέρη, έναν χώρο ετικετών Y και μια συνάρτηση πρόβλεψης $f(-)$, η οποία μαθαίνεται από το διάνυσμα χαρακτηριστικών και τα ζεύγη ετικετών x_i, y_i όπου $x_i \in X$ και $y_i \in Y$. Αναφερόμενοι στην εφαρμογή ταξινόμησης ελαττωμάτων μονάδας λογισμικού, το Y είναι το σύνολο των ετικετών , τα y_i παίρνουν τιμές true ή false, και $f(x)$ είναι η συνάρτηση του μοντέλου που προβλέπει την τιμή της ετικέτας για τη μονάδα λογισμικού x. Από τους παραπάνω ορισμούς, ένας τομέας $D = \{X, P(X)\}$ και μια εργασία $T = Y, f(-)$. Τώρα, το D_S ορίζεται ως τα δεδομένα του πηγαίου τομέα όπου $D_S = \{x_{S1}, y_{S1}, ..., x_{Sn}, y_{Sn}\}$, όπου $x_{Si} \in X_S$ είναι η i-οστή περίπτωση δεδομένων του D_S και $y_{Si} \in YS$ είναι η αντίστοιχη ετικέτα κλάσης για το x_{Si}. Με τον ίδιο τρόπο, το D_T ορίζεται ως τα δεδομένα του τομέα-στόχου, όπου $D_T = \{x_{T1}, y_{T1}, ..., x_{Tn}, y_{Tn}\}$, όπου $x_{Ti} \in X_T$ είναι το i-οστό δεδομένοπερίπτωση του D_T και$y_{Ti} \in Y_T$ είναι η αντίστοιχη ετικέτα κλάσης για το x_{Ti}. Επιπλέον, η πηγή εργασία συμβολίζεται ως T_S, η εργασία-στόχος ως T_T , η συνάρτηση πρόβλεψης πηγής ως $f_S(\dot{})$, και η συνάρτηση πρόβλεψης στόχου ως $f_T(\dot{})$.

Τώρα το Transfer Learning μπορεί να οριστεί επίσημα : Δεδομένου ενός πεδίου πηγήςD_S με μια αντίστοιχη εργασία πηγής T_S και ενός πεδίου στόχου D_T με μια αντίστοιχη εργασία T_T , το Transfer Learning είναι η διαδικασία βελτίωσης της συνάρτησης πρόβλεψης στόχου $f_T(\dot{})$ με τη χρήση των σχετικών πληροφοριών από τα D_S και T_S, όπου $DS \neq DTTS \neq TT$. Ο τομέας πηγής (input) που ορίζεται εδώ μπορεί να επεκταθεί σε πολλαπλούς τομείς πηγής. Δεδομένου ότι $D_S = \{X_S, P(X_S)\}$ και $D_T = \{X_T, P(X_T)\}$, η συνθήκη $D_S \neq D_T$ σημαίνει ότι $XS \neq XT$ και/ή $P(X_S) \neq P(X_T)$. Η περίπτωση όπου$X_S \neq X_T$ οσον αφορά το το Transfer Learning ορίζεται ως ετερογενές το Transfer Learning. Η περίπτωση όπου $X_S = X_T$ αντίστοιχα ορίζεται ως ομοιογενής μάθηση μεταφοράς. Επιστρέφοντας στο παράδειγμα της ταξινόμησης ελαττωμάτων μονάδων λογισμικού, η ετερογενής μάθηση μεταφοράς είναι η περίπτωση όπου το έργο λογισμικού προέλευσης έχει διαφορετικές μετρικές (χαρακτηριστικά) από το έργο λογισμικού στόχου. Εναλλακτικά, η ομοιογενής μάθηση μεταφοράς είναι όταν οι μετρικές λογισμικού είναι ίδιες τόσο για το έργο λογισμικού προέλευσης όσο και για το έργο λογισμικού στόχου. Συνεχίζοντας με τον ορισμό της μάθησης μεταφοράς, η περίπτωση όπου $P(X_S) \neq P(X_T)$ σημαίνει ότι οι οριακές κατανομές στους χώρους εισόδου είναι διαφορετικές

μεταξύ των έργων προέλευσης και των πεδίων-στόχων.

Ο Shimodaira απέδειξε ότι ένα μοντέλο μάθησης που εκπαιδεύεται με ένα δεδομένο πεδίο πηγής δεν θα αποδώσει βέλτιστα σε ένα πεδίο στόχου όταν οι οριακές κατανομές των πεδίων εισόδου είναι διαφορετικές. Ένα άλλο πιθανό συμπέρασμα του Transfer Learning (από τον παραπάνω ορισμό) είναι οτι $TS \neq TT$, και επιπλέον ισχύει ότι $T = Y, f()$ ή $T = Y, P(Y|X)$. Επομένως, σε ένα περιβάλλον μεταφοράς μάθησης, είναι δυνατόν να ισχύει $Y_S \neq Y_T$ και/ή $P(Y_S|X_S) \neq P(Y_T|X_T)$. Η περίπτωση όπου $P(Y_S|X_S) \neq P(Y_T|X_T)$ σημαίνει ότι οι δεσμευμένες πιθανότητες, μεταξύ των τομέων πηγής και στόχου είναι διαφορετικές. Ένα παράδειγμα μιας υπό συνθήκη κατανομής αναντιστοιχίας είναι όταν μια συγκεκριμένη ενότητα λογισμικού αποδίδει διαφορετικά αποτελέσματα επιρρεπή σε σφάλματα στους τομείς πηγής και στόχου. Η περίπτωση $Y_S \neq Y_T$ αναφέρεται σε αναντιστοιχία στον χώρο των κλάσεων. Ένα παράδειγμα αυτής της περίπτωσης είναι όταν το έργο λογισμικού πηγής έχει ένα δυαδικό χώρο ετικετών true για επιρρεπή σε ελαττώματα και false για μη επιρρεπή σε ελαττώματα, και ο τομέας-στόχος έχει μια ετικέτα χώρο που ορίζει πέντε επίπεδα μονάδων επιρρεπών σε σφάλματα. Μια άλλη περίπτωση που μπορεί να προκαλέσει υποβάθμιση του διακριτικού ταξινομητή είναι όταν $P(Y_s) \neq P(Y_T)$, η οποία προκαλείται από ένα μη ισορροπημένο σύνολο δεδομένων με ετικέτες μεταξύ των τομέων πηγής και στόχου. Η περίπτωση των παραδοσιακών μοντέλων μηχανικής μάθησης είναι $D_S = D_T$ και $T_S = T_T$. [16]

4.6 YOLO (You Only Look Once)

4.6.1 Εισαγωγή

Στην επίλυση του προβλήματος που ασχολείται η παρούσα διπλωματική εργασία ο στόχος είναι η αναγνώριση της ύπαρξης η μη μασκας απο live feed βίντεο, οπότε το μοντέλο αναγνώρισης θα πρέπει να επεξεργάζεται το βίντεο με ελάχιστη καθυστέρηση ώστε να έχουν νόημα οι προβλέψεις. Δεδομένου ότι μια συνηθισμένη κάμερα βγάζει βίντεο σε τουλάχιστον 45fps αυτό δεν είναι εύκολο task.

Το μοντέλο που χρησιμοποιήθηκε είναι η πιο πρόσφατη έκδοση του YOLO (You Only Look Once), το YOLOv7. Το μοντέλο αυτό κατασκευάστηκε με σκοπό την γρήγορη κατηγοριοποίηση εικόνων με αρκετά μεγάλη ακρίβεια για την ταχύτητά του. Το βασικό δίκτυο μπορεί να τρέχει σε 45 frames per second χωρίς batch processing σε μια Titan X GPU και η γρήγορη (αλλα όχι τόσο ακριβής) έκδοση στα 150fps. Αυτό συνεπάγεται ότι μπορεί να επεξεργαστεί real time video με λιγότερα απο 25 εκατομυριοστά του δευτερολέπτου καθυστέρηση. Επιπλέον έχει καταφέρει να πετύχει διπλάσιο mean average precision σε σχέση με άλλα μοντέλα αναγνώρισης πραγματικού χρόνου.

Κάποια διαφορετικά συστήματα κατηγοριοποίησης χρησιμοποιούν έτοιμους classifiers του

εκάστοτε αντικειμένου και τον τρέχουν σε διαφορετικές περιοχές τις εικόνας με σκοπό αναγνώριση με μεγάλη εμπιστοσύνη. Τέτοια μοντέλα όπως το Deformable Parts Models (DPM) δεν βοηθούν ιδιαίτερα στο πρόβλημα της περιπτωσής μας λόγω των καθυστερήσεών τους.

Η λειτουργικότητα του YOLO που δίνει αυτό το πλεονέκτημα στη ταχύτητα είναι αυτό που ισχυρίζεται και το όνομά του: κοιτάς μία φορά! Σε αντίθεση λοιπόν με αντίστοιχα μοντέλα, το YOLO με ένα CNN ταυτόχρονα προβλέπει τα περικλειόμενα ορθωγώνια που μπορεί να βρίσκεται το αντικείμενο κάποιας κλάσης, μαζί με τη πιθανότητα το αντικείμενο μέσα στο ορθογώνιο να έιναι της εκάστοτε κλάσης. Αυτό λοιπόν το γεγονός δίνει την μεγάλη υπεροχή του στη ταχύτητα αναγνώρισης.

4.6.2 Unified Detection

Η λειτουργικότητα του μοντέλου της ταυτόχρονης αναγνώρισης γίνεται με τον εξής τρόπο: Αρχικά, χωρίζεται η εικόνα σε ένα SxS grid. Αν το κέντρο ενός αντικειμένου πέσει σε κάποιο στοιχείο του grid αυτό είναι υπεύθυνο για την αναγνώρισή του.

Κάθε τέτοιο κελί προβλέπει B περικλείοντα κουτιά και το ποσοστό εμπιστοσύνης για αυτά. Το ποσοστό εμπιστοσύνης ορίζεται επίσημα ως:

$$pred = Pr(Object) \times IOU_{pred}^{truth}$$

Αν δεν υπάρχει αντικείμενο στο αντίστοιχο περικλειόμενο κουτί θα πρέπει η αντίστοιχη εμπιστοσύνη να είναι 0. Το κάθε κουτί περιγράφεται απο 5 μεταβλητές $x, y, w, h, pred$. Το (x, y) αντιπρωσοπεύει το κέντρο του κουτιού και τα w, h το ύψος και το πλάτος του αντίστοιχα. Επίσης προβλέπεται και η πιθανότητα το κάθε αντικέιμενο να έιναι αντίστοιχης κλάσης σύμφωνα με τον τύπο:

$$Pr(Class_i|Object)$$

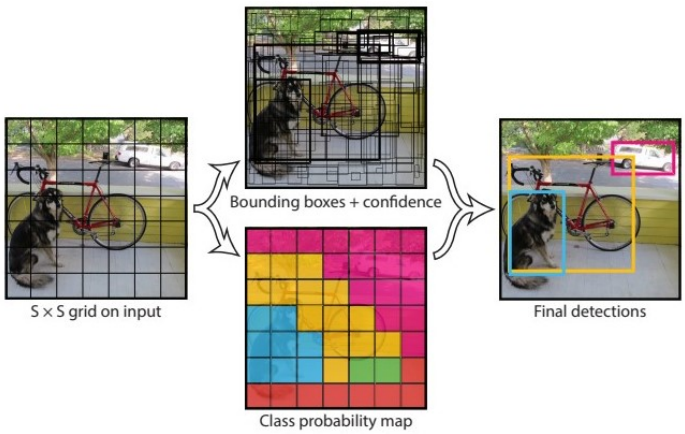
Υπολογίζεται μόνο ένα σύνολο πιθανοτήτων αν κελί του grid ανεξάρτητα αοι των αριθμό των κουτιών που αντιστοιχούν σε αυτό. Στην περίπτωση του test πολλαπλασιάζονται οι αντίστοιχες δεσμευμένες πιθανότητες και η εμπιστοσύνη του κάθε κουτιού, λαμβάνοντας έτσι υπ΄ όψιν και τη πιθανότητα ύπαρξης αντικειμένου κλάσης στο κουτί όσο και το πόσο καλά περικλείει το κουτί το αντικείμενο. Ο αντίστοιχος τύπος είναι ο

$$Pr(Class_i|Object) \times Pr(Object) \times IOU_{pred}^{truth} = Pr(Class_i) \times IOU_{pred}^{truth}$$

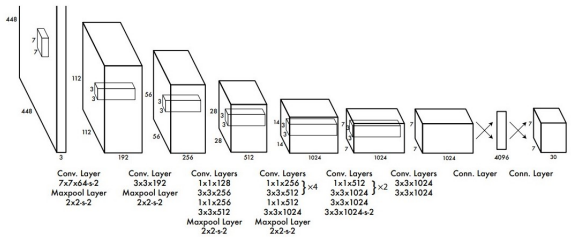
[11]

4.6.3 Εσωτερική αρχιτεκτονική του μοντέλου

Η αρχιτεκτονική του μοντέλου περιγράφεται απο 24 στρώματα συνελικτικών δικτύων τα οποία ακολουθούνται απο 2 πλήρως συνδεδεμένα στρώματα. Χρησιμοποιούνται $1x1$ στρώματα



Σχήμα 4.3: Ταυτόχρονος υπολογισμός κλάσσης και περικλειόμενου κουτιού



Σχήμα 4.4: Εσωτερική αρχιτεκτονική του μοντέλου YOLO

μείωσης τα οποία ακολουθούνται απο $3x3$ συνελικτικά στρώματα νευρώνων. Το τελικό output είναι ένας $7x7x30$ τένσορας όπως φαίνεται στην παρακλάτω εικόνα.

Το τελικό πλήρως συνδεδεμένο στρώμα προβλέπει τις τειλκές πιθανότητες για τα κουτιά και τις κλάσεις. Κανονικοποιείται η εικόνα σε διαστάσεις στο διάστημα $x, y \in [0, 1]$, Ενώ η συνάρτηση ενεργοποίσης που χρισημοποιείται είναι γραμμική και ισούται μέ

$$f(x) = \begin{cases} 0.1x & x \leq 0 \\ x & otherwise \end{cases}$$

4.6.4 Εκπαίδευση του μοντέλου

Για την επίτευξη μέγιστης απόδοσης το μοντέλο του YOLO γίνονται pretrain τα πρώτα 20 στρώματα συνελικτικού δικτύου στο 1000-class competition dataset της ImageNet ακολουθούμενα απο ενα average pooling layer και ένα στρώμα πλήρους διασύνδεσης. Εκπαιδεύεται για περίπου μια εβδομάδα και καθόλη τη διαδικασία χρησιμοποιείται το Darknet framework για την εκπαίδευση και τα συμπεράσματα.

Η loss function που χρησιμοποιείται είναι η κλασσική συνάρτηση διασποράς, λόγω της ευκολίας βελτιστοποίησης, όμως δεν συμπίπτει ακριβώς με τον στόχο της μεγιστοποίησης του

μέσου precision. Αυτό συμβαίνει διότι εξισώνει τη σημαντικότητα του λάθους εντοπισμού σωστής κλάσης με το λάθος επιλογής σωστου περικλειόμενου ορθογωνίου. Επιπλέον σε κάθε εικόνα τα περισσότερα κελιά δεν περιέχουν κάποιο αντικείμενο , γεγονός που μπορεί να κάνει την εκπάιδευση του μοντέλου να αποκλίνει. Για την επίλυση αυτών των προβλημάτων αυξήθηκε χειροκείνητα η απώλεια λόγω λάθους τοποθέτησης του κουτιού ενώ μειώθηκε η αντίστοιχη απώλεια λόγω λάθους κατηγοριοποίησης του αντικειμένου μέσω δύο παραμέτρων $\lambda_{coord} = 5$ και $\lambda_{noobj} = 0.5$. Επιπλέον η συνάρτηση απώλειας πρέπει να λαμβάνει υπ όψιν ότι ένα λάθος στο περικλειόμενο κουτί ενός μεγάλου αντικειμένου είναι λιγότερο σημαντικό απο το ίδιο σε απολυτη τιμή λάθος ενός μικρότερου. Συνοψίζοντας τα παραπάνω οι δημιουργοί του μοντέλου κατέληξαν στην εξής συνάρτηση απώλειας:

$$
\begin{aligned}
&\lambda_{\textbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\
&+ \lambda_{\textbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right] \\
&+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2 \\
&+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2 \\
&+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2 \quad (
\end{aligned}
$$

Όπου η 1_i^{obj} ειναι η δείκτρια συνάρτηση του άν ένα αντικείμενο εφανίζεται στο κελί i και η 1_{ij}^{obj} ειναι η δείκτρια συνάρτηση του άν ένα αντικείμενο εφανίζεται στο j-οστό περικλειόμενο κουτί του κελιού i.

[15]

4.6.5 Προεκπαίδευση του μοντέλου YOLO, COCO dataset

Το μοντέλο του YOLO που χρησιμοποιήθηκε στα πλαίσια της αναγνώρισης μάσκας ήταν το YOLOv7 προεκπαιδευμένο μοντέλο. Όπως αναφέρθηκε και σε προηγούμενη ενότητα το μοντέλο που εκπαιδεύσαμε ήταν ήδη εκπαιδευμένο σε ένα παρεμφερές πρόβλημα, συγκεκριμένα της αναγνώρισης αντικειμένων σε εικόνα, χρησιμοποιήθηκε δηλαδή το Transfer Learning. Το συγκεκριμένο μοντέλο ήταν προεκπαιδευμένο σε ένα απο τα γνωστότερα dataset, συγκεκριμένα το COCO (Common Objects in Context)

Το σύνολο δεδομένων COCO (Common Objects in Context) είναι ένα μεγάλης κλίμακας dataset αναγνώρισης εικόνας για την ανίχνευση αντικειμένων, την κατάτμηση (segmentation) και την απόδοση κειμένου. Περιέχει πάνω από 330.000 εικόνες, καθεμία από τις οποίες είναι σχολιασμένη με 89 κατηγορίες αντικειμένων, οι οποίες εμφανίζονται στον παρακάτω πίνακα. Το COCO dataset χρησιμοποιείται ευρέως στην έρευνα της όρασης υπολογιστών και έχει χρησιμοποιηθεί για την εκπαίδευση και αξιολόγηση πολλών μοντέλων ανίχνευσης και τμημα-

τοποίησης αντικειμένων τελευταίας τεχνολογίας. Το dataset αποτελείται από δύο κύρια μέρη: τις εικόνες και τα labels τους.

Οι εικόνες είναι οργανωμένες σε μια ιεραρχία καταλόγων, με τον κατάλογο ανώτατου επιπέδου να περιέχει υποκαταλόγους για τα σύνολα εκπαίδευσης, επικύρωσης και δοκιμής, τα train, validation, test. Τα labels παρέχονται σε μορφή JSON, με κάθε αρχείο να αντιστοιχεί σε μία εικόνα. Κάθε label στο σύνολο δεδομένων περιλαμβάνει τις ακόλουθες πληροφορίες:

- Όνομα αρχείου εικόνας
- Μέγεθος εικόνας (πλάτος και ύψος)
- Λίστα αντικειμένων με τις ακόλουθες πληροφορίες: Κατηγορία αντικειμένου (π.χ. ¨πρόσωπο¨, ¨αυτοκίνητο¨)
- Συντεταγμένες του κουτιού της αναγνώρισης(bounding box) (x, y, πλάτος, ύψος)-
- Segmentation mask (πολυγωνική ή σε μορφή RLE)
- Βασικά σημεία και θέσεις τους (εάν υπάρχουν)
- Πέντε λεζάντες που περιγράφουν τη σκηνή

Το MS COCO προσφέρει επιπλέον και διάφορους τύπους annotations, όπως

- **Object detection** με συντεταγμένες bounding box και πλήρεις μάσκες τμηματοποίησης για 89 διαφορετικές κλάσεις αντικείμενων.
- **Stuff image segmentation** με χάρτες pixel που εμφανίζουν 91 κατηγορίες περιοχών φόντου
- **Panoptic segmentation** προσδιορίζει αντικείμενα σε εικόνες
- **Dense pose** με πάνω από 39.000 φωτογραφίες που περιλαμβάνουν πάνω από 56.000 άτομα με labels με αντιστοίχιση μεταξύ pixels και ενός πρότυπου τρισδιάστατου μοντέλου και περιγραφές φυσικής γλώσσας (NLP) για κάθε εικόνα
- **Keypoint annotations** για πάνω από 250.000 άτομα με σχόλια σε σημεία-κλειδιά, όπως το δεξί μάτι, η μύτη και ο αριστερός γοφός.

4.7 Μοντέλα tracking

4.7.1 Εισαγωγή στο πρόβλημα tracking

Το tracking στο deep learning είναι το έργο της πρόβλεψης των θέσεων των αντικειμένων σε ένα βίντεο χρησιμοποιώντας τα χωρικά και τα χρονικά χαρακτηριστικά τους. Πιο τεχνικά, το tracking είναι η λήψη του αρχικού συνόλου ανιχνεύσεων που έγιναν απο κάποιο μοντέλο , η

ID	OBJECT	ID	OBJECT	ID	OBJECT
1	person	31	handbag	61	cake
2	bicycle	32	tie	62	chair
3	car	33	suitcase	63	couch
4	motorcycle	34	frisbee	64	potted plant
5	airplane	35	skis	65	bed bed
6	bus	36	snowboard	66	mirror
7	train	37	sports	67	dining table
8	truck	38	kite	68	window
9	boat	39	baseball	69	desk
10	traffic	40	baseball	70	toilet
11	fire	41	skateboard	71	door
12	street	42	surfboard	72	tv
13	stop	43	tennis racket	73	laptop
14	parking meter	44	bottle	74	mouse
15	bench	45	plate	75	remote
16	bird	46	wine glass	76	keyboard
17	cat	47	cup	77	cell phone
18	dog	48	fork	78	microwave
19	horse	49	knife	79	oven
20	sheep	50	spoon	80	toaster
21	cow	51	bowl	81	sink
22	elephant	52	banana	82	refrigerator
23	bear	53	apple	83	blender
24	zebra	54	sandwich	84	book
25	giraffe	55	orange	85	clock
26	hat	56	broccoli	86	vase
27	backpack	57	carrot	87	scissors
28	umbrella	58	hot dog	88	teddy
29	shoe	59	pizza	89	hair
30	eye glasses	60	donut		

4.1: MS COCO Dataset Classes

ανάθεση μοναδικών αναγνωριστικών και η παρακολούθησή τους σε όλα τα καρέ της ροής βίντεο διατηρώντας τα αναγνωριστικά που έχουν ανατεθεί. Η ανίχνευση είναι γενικά μια διαδικασία δύο βημάτων απο δυο μονάδες:

- Μια μονάδα ανίχνευσης για τον εντοπισμό του στόχου: Η μονάδα που είναι υπεύθυνη για την ανίχνευση και τον εντοπισμό του αντικειμένου στο καρέ χρησιμοποιώντας κάποιον Object Detector όπως το YOLOv4, το CenterNet κ.λπ.

- Ενας προγνώστης κίνησης: Αυτή η μονάδα είναι υπεύθυνη για την πρόβλεψη της μελλοντικής κίνησης του αντικειμένου χρησιμοποιώντας τις πληροφορίες του παρελθόντος, τόσο χωρικές όσο και χρονικές.

4.7.2 Προκλήσεις στο tracking

Κατά την κατασκευή ενός αλγορίθμου παρακολούθησης αντικειμένων προκύπτουν μερικές προκλήσεις. Η παρακολούθηση ενός αντικειμένου σε έναν δρόμο ή σε ένα απλό περιβάλλον είναι εύκολη. Σε ένα πραγματικό περιβάλλον, το αντικείμενο ενδιαφέροντος θα έχει πολλούς παράγοντες που το επηρεάζουν, καθιστώντας την παρακολούθηση του αντικειμένου δύσκολη. Η επίγνωση αυτών των κοινών προκλήσεων είναι το πρώτο βήμα για να μπορέσετε να αντιμετωπίσετε αυτά τα ζητήματα κατά τη σχεδίαση αλγορίθμων εντοπισμού αντικειμένων. Μερικές από τις κοινές προκλήσεις της ανίχνευσης αντικειμένων είναι οι εξής:

- **Παρεμβολές φόντου**: Είναι δύσκολο να εξαχθούν χαρακτηριστικά, να ανιχνευθεί ή ακόμη και να εντοπιστεί ένα αντικείμενο-στόχος όταν το φόντο είναι πυκνοκατοικημένο, καθώς εισάγει περισσότερες περιττές πληροφορίες ή θόρυβο, καθιστώντας το δίκτυο λιγότερο ευαίσθητο σε σημαντικά χαρακτηριστικά.

- **Διαφοροποίηση φωτισμού**: Σε ένα πραγματικό σενάριο, ο φωτισμός σε ένα αντικείμενο ενδιαφέροντος μεταβάλλεται δραστικά καθώς το αντικείμενο κινείται, καθιστώντας τον εντοπισμό του πιο δύσκολο.

- **Απόκρυψη**: Καθώς διάφορα αντικείμενα εισέρχονται και εξέρχονται από το καρέ, το αντικειμένου-στόχος συχνά αποκρύπτεται εμποδίζοντας τον αλγόριθμο να το εντοπίσει και να το παρακολουθήσει, καθώς το φόντο ή το προσκήνιο παρεμβάλλονται. Αυτό συμβαίνει συχνά όταν τα πλαίσια οριοθέτησης πολλαπλών αντικειμένων έρχονται πολύ κοντά μεταξύ τους, με αποτέλεσμα οι αλγόριθμοι να μπερδεύονται και να αναγνωρίζουν το αντικείμενο που παρακολουθείται ως νέο αντικείμενο.

- **Αλλαγή του σχήματος του στόχου** Σε όλες τις εικόνες και τα καρέ, το σχήμα του αντικειμένου ενδιαφέροντος μπορεί να περιστραφεί, να μειωθεί σε μέγεθος ή να παραμορφωθεί.Αυτό μπορεί να οφείλεται σε πολλούς παράγοντες, όπως η μεταβολή του σημείου θέασης ή οι αλλαγές στην κλίμακα του αντικειμένου, και συχνά επηρεάζει τη διαίσθηση εντοπισμού αντικειμένων του αλγορίθμου.

- **Χαμηλή ανάλυση**: Ανάλογα με την ανάλυση, ο αριθμός των εικονοστοιχείων μέσα στο πλαίσιο οριοθέτησης του συνόλου δεδομένων εκπαίδευσης μπορεί να είναι πολύ μικρός για να είναι συνεπής η παρακολούθηση αντικειμένων.

Στα πλαίσια αυτής της εργασίας, στην προσπάθεια επίλυσης των προκλήσεων του προβλήματος tracking σε πραγματικό χρόνο υλοποιήθηκαν δύο μοντέλα τα οποία χρησιμοποιούν μεταξύ τους διαφορετικές προσεγγίσεις στο πρόβλημα.

4.7.3 Το μοντέλο παρακολούθησης DeepSort

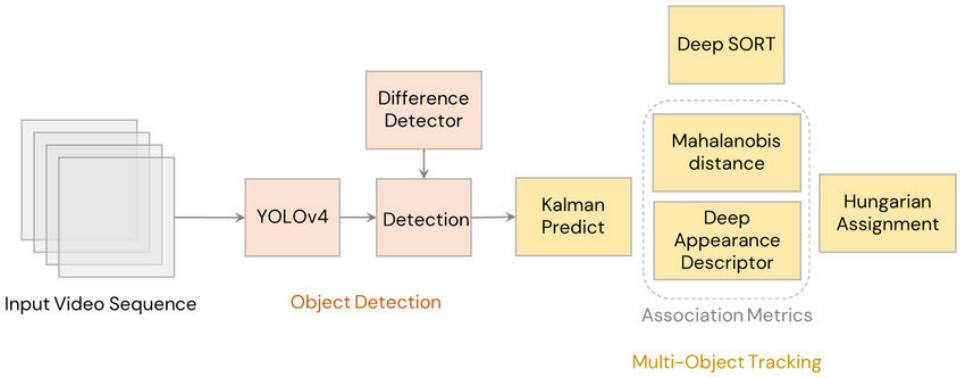
Ο DeepSort είναι ένας γνωστός αλγόριθμος εντοπισμού αντικειμένων. Αποτελεί επέκταση του Simple Online Real-time Tracker (SORT), ενός online αλγορίθμου εντοπισμού. Το SORT είναι μια μέθοδος που εκτιμά τη θέση ενός αντικειμένου με βάση την προηγούμενη θέση του χρησιμοποιώντας το φίλτρο Kalman. Το φίλτρο Kalman μία μέθοδος αρκετά αποτελεσματική για την αντιμετώπιση του προβλήματος της απόκρυψης. Ουσιαστικά προσδίδει την έννοια της κίνησης του αντικειμένου μεταξύ διαδιχικών καρέ, δίνοντας στο μοντέλο την δυνατότητα προσεγγιστικου υπολογισμού της θέσης ενος αντικείμενου, ακόμη και όταν αυτό εμφανίζεται μερικώς ή καλύπτεται πλήρως σε ένα καρέ.

Το SORT αποτελείται από τρία στοιχεία:

- **Ανίχνευση**: Πρώτον, ανιχνεύεται το αρχικό αντικείμενο ενδιαφέροντος.
- **Πρόβλεψη**: Η μελλοντική θέση του αντικειμένου ενδιαφέροντος προβλέπεται με τη χρήση του φίλτρου Kalman.
- **Συσχέτιση**: Οι προσεγγιστικές θέσεις του αντικειμένου-στόχου που έχει προβλεφθεί πρέπει να βελτιστοποιηθούν. Αυτό γίνεται συνήθως με την ανίχνευση της θέσης στο μέλλον χρησιμοποιώντας τον Hungarian algorithm.

Οι αλγόριθμοι βαθιάς μάθησης χρησιμοποιούνται για τη βελτίωση των αλγορίθμων SORT. Επιτρέπουν στο SORT να εκτιμήσει τη θέση του αντικειμένου με πολύ μεγαλύτερη ακρίβεια, επειδή τα δίκτυα αυτά μπορούν πλέον να προβλέψουν τα χαρακτηριστικά της εικόνας-στόχου. Ο ταξινομητής του συγκεκριμένο συνελικτικού νευρωνικού δικτύου (CNN) ουσιαστικά εκπαιδεύεται σε ένα σύνολο δεδομένων μέχρι να επιτύχει υψηλή ακρίβεια. Μόλις επιτευχθεί, ο ταξινομητής αφαιρείται, αφήνοντας απλώς τα χαρακτηριστικά που συλλέγονται από το σύνολο δεδομένων. Αυτά τα εξαγόμενα χαρακτηριστικά χρησιμοποιούνται στη συνέχεια από τον αλγόριθμο SORT για τον εντοπισμό στοχευμένων αντικειμένων.

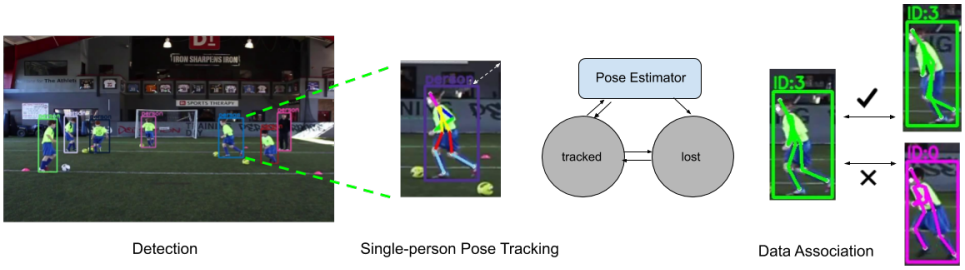
Ο DeepSORT λειτουργεί στα 20Hz, με τη δημιουργία των χαρακτηριστικών να καταλαμβάνει σχεδόν το μισό χρόνο εξαγωγής συμπερασμάτων. Επομένως, με δεδομένη μια σύγχρονη GPU, το σύστημα παραμένει υπολογιστικά αποδοτικό και λειτουργεί σε πραγματικό χρόνο.



Σχήμα 4.5: DeepSort Architecture

4.7.4 Το μοντέλο παρακολούθησης LightTrack

Το Pose Tracking είναι το ππρόβλημα του υπολογισμού των ανθρώπινων πόζων πολλαπλών προσώπων σε βίντεο και της ανάθεσης μοναδικών αναγνωριστικών για κάθε σημείο κλειδί σε όλα τα καρέ.



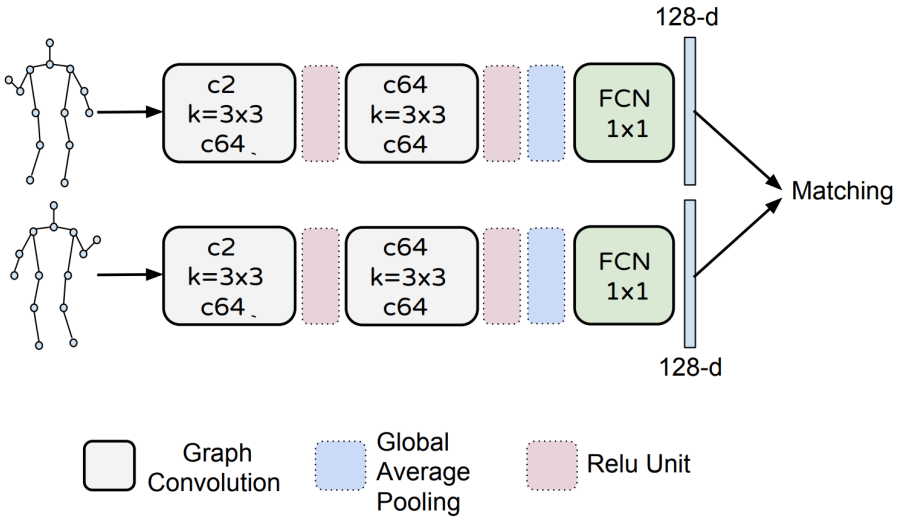
Σχήμα 4.6: Pose Tracking

Το μοντέλο παρακολούθησης LightTrack προτείνει ένα νέο πλαίσιο εντοπισμού πόζας top-down. Έχει αποδειχθεί ότι η ανθρώπινη στάση μπορεί να χρησιμοποιηθεί για την καλύτερη εξαγωγή συμπερασμάτων σχετικά με τις ανθρώπινες θέσεις. Παρατηρούμε ότι, σε μια top-down προσέγγιση , οι ακριβείς ανθρώπινες θέσεις διευκολύνουν επίσης την εκτίμηση των ανθρώπινων στάσεων. Μελετάμε περαιτέρω τις σχέσεις μεταξύ αυτών των δύο επιπέδων πληροφορίας: Η προσεγγιστική θέση του ατόμου μπορεί να αποσταλεί σε βασικά σημεία του σώματος με μια εκτιμητή πόζας ενός ατόμου. Η θέση των ανθρώπινων αρθρώσεων μπορεί να χρησιμοποιηθεί ευθέως για να υποδείξει αδρές θέσεις υποψήφιων ανθρώπων. Έτσι, η επαναλαμβανόμενη εκτίμηση είναι μια εφικτή στρατηγική για το πρόβλημα Single-person Tracking (SPT). Το πρόβλημα Pose Tracking για πολλά αντικείμενα δεν ρίναι αποδοτικό να επιλυθεί σαν άθροισμα πολλών SPT. Για αυτό το μοντέλο LightTrack αντιμετωπίζει πρώτα κάθε υποψήφιο άνθρωπο ξεχωριστά, έτσι ώστε η αντίστοιχη ταυτότητά του να διατηρείται σε όλα τα καρέ. Με αυτόν τον τρόπο, παρακάμπτεται η χρονοβόρα διαδικασία βελτιστοποίησης εκτός σύνδεσης. Σε περίπτωση που ο εντοπισμένος υποψήφιος χαθεί λόγω απόκρυψης ή μετατόπισης της κάμερας, καλείται στη συνέχεια η μονάδα ανίχνευσης για αναβιώση των υποψηφίων και να το συσχετίσμό τους με τους εντοπισμένους στόχους από το προηγούμενο καρέ μέσω αντιστοίχισης στάσεων. Με αυτόν τον τρόπο, επιτυγχάνεται η ανίχνευση πόζας πολλαπλών στόχων. με μια διαδικασία SPT και μια διαδικασία αντιστοίχισης πόζας μεταξύ των καρέ.

Συγκεκριμένα, το πλαίσιο οριοθέτησης του ατόμου στο επόμενο καρέ συνάγεται από τις αρθρώσεις που εκτιμώνται από την πόζα από το τρέχον καρέ. Εντοπίζονται οι ελάχιστες και μέγιστες συντεταγμένες και διευρύνεται αυτή η περιοχή ενδιαφέροντος κατά 20% σε κάθε πλευρά. Το διευρυμένο πλαίσιο οριοθέτησης αντιμετωπίζεται ως η εντοπισμένη περιοχή για αυτό το άτομο στο επόμενο καρέ. Εάν η μέση βαθμολογία εμπιστοσύνης $\hat{s}$ από τις εκτιμώμενες αρθρώσεις είναι χαμηλότερη από το όριο τ_s, αυτό αντανακλά ότι ο στόχος έχει χαθεί δεδομένου ότι οι αρθρώσεις δεν είναι πιθανό να εμφανιστούν στο πλαίσιο οριοθέτησης που βρέθηκαν αν ανήκουν στο στόχο. Η κατάσταση του στόχου ορίζεται ως εξής:

$$state = \begin{cases} tracked & \hat{s} > \tau_s \\ lost & otherwise \end{cases}$$

Δεδομένων των ακολουθιών των αρθρώσεων του σώματος στο με τη μορφή 2D συντεταγμένων, κατασκευάζεται ένα χωρικό γράφημα με τις αρθρώσεις ως κόμβους του γραφήματος και τις συνδέσεις του ανθρώπινου σώματος ως ακμές του γράφου. Η είσοδος στον γράφο είναι τα διανύσματα των συντεταγμένων των αρθρώσεων στους κόμβους του γράφου. Είναι ανάλογο με τα CNN που βασίζονται στην εικόνα, όπου η είσοδος σχηματίζεται από διανύσματα έντασης εικονοστοιχείων που βρίσκονται στο δισδιάστατο πλέγμα της εικόνας. Η είσοδος στο συνελικτικό νευρωνικό δίκτυο είναι οι συντεταγμενες-διανύσματα των συνδέσεων της ανθρώπινης στάσης.



Σχήμα 4.7: LightTrack Architecture

Αξιολόγηση μοντέλου

Η αξιολόγηση περιλαμβάνει ακρίβεια εκτίμησης της πόζας και ακρίβεια εντοπισμού της πόζας. Η ακρίβεια εκτίμησης της πόζας αξιολογείται με τη χρήση της τυπικής μετρικής mAP, ενώ η αξιολόγηση της παρακολούθησης της πόζας γίνεται σύμφωνα με τις σαφείς μετρικές MOT που αποτελούν το πρότυπο για την αξιολόγηση της παρακολούθησης πολλαπλών στόχων.

[9]

Κεφάλαιο 5

Apache Kafka

5.1 Εισαγωγή

Με την ονομασία Apache Kafka είναι γνωστή πλατφόρμα λογισμικού για επεξεργασία ροών δεδομένων. Αναπτύχθηκε αρχικά από την εταιρεία πίσω από το κοινωνικό δίκτυο LinkedIn και κατόπιν δόθηκε ως δωρεά στο Ίδρυμα Λογισμικού Apache. Είναι γραμμένη στις γλώσσες προγραμματισμού Scala και Java, ενώ πρόκειται για λογισμικό κώδικα ελεύθερου προς ανάπτυξη από όλους. Το έργο αποσκοπεί να παρέχει μια ενιαία πλατφόρμα για χειρισμό ροών δεδομένων σε πραγματικό χρόνο, με χαρακτηριστικά την υψηλή απόδοση και ελάχιστες περιόδους αδράνειας. Στην αρχιτεκτονική του το επίπεδο αποθήκευσης είναι ουσιαστικά μια ουρά δημοσίευσης και κατανάλωσης μηνυμάτων, με τεράστια επιδεκτικότητα διεύρυνσης, σχεδιασμένη σαν ένα κατανεμημένο αρχείο καταγραφής συναλλαγών.Αυτό την καθιστά ιδιαίτερα πολύτιμη για εφαρμογές που απευθύνονται σε επιχειρήσεις και βιομηχανίες με την ανάγκη να επεξεργάζονται ροές δεδομένων. Επιπροσθέτως, η πλατφόρμα μπορεί να συνδεθεί σε εξωτερικά συστήματα (για εισαγωγή/εξαγωγή δεδομένων) με μια από τις συνιστώσες του, την Kafka Connect, ενώ προσφέρει και μια βιβλιοθήκη ζωντανής επεξεργασίας των εισερχόμενων δεδομένων, τα Kafka Streams. [1]

5.2 Event Streaming

Το Event Streaming μπορεί να παρομοιαστεί σαν το ψηφιακο αντίστοιχο του εγκεφάλου στο ανθρώπινο σώμα. Από τεχνική σκοπιά το Event Streaming είναι η πρακτική λήψης ζωντανών δεδομένων απο πηγές όπως βάσεις δεδομένων, σενσορες, κινητά, υπηρεσίες cloud ή άλλες εφαρμογές με τη μορφή streams ή events, η ασφαλής αποθήκευση αυτών για να χρησιμοποιηθούν αργότερα ή να επεξεργάστουν σε ζωντανό χρόνο για τη λήψη αποφάσεων με σκοπό την ομαλή λειτουργία όλης της εφαρμογής. Τα Event Streams λοιπόν διασφαλίζουν την συνεχή ροή των δεδομένων και των εφαρμογών που δρουν σε αυτα με συνέχεια και ασφάλεια ώστε να μη χαθεί καποια πληροφορία.

Σχήμα 5.1: Apache Kafka logo

Η χρήση των Event Streams είναι πολυποίκιλη και χρησιμοποιούνται σε μια πληθώρα οργανισμών και βιομηχανιών. Μερικές απο τις χρήσεις τους συνοψίζονται παρακάτω:

1. Για την επεξεργασία πληρωμών και χρηματοοικονομικών συναλλαγών σε πραγματικό χρόνο, όπως σε χρηματιστήρια, τράπεζες και ασφαλιστικές εταιρείες.

2. Για την παρακολούθηση αυτοκινήτων, φορτηγών, στόλων και αποστολών σε πραγματικό χρόνο, όπως στα logistics και την αυτοκινητοβιομηχανία.

3. Για τη συνεχή καταγραφή και ανάλυση δεδομένων αισθητήρων από συσκευές IoT ή άλλον εξοπλισμό, όπως σε εργοστάσια και πάρκα αιολικής ενέργειας.

4. Για τη συλλογή και άμεση ανταπόκριση σε αλληλεπιδράσεις και παραγγελίες πελατών, όπως στον τομέα του λιανικού εμπορίου, των ξενοδοχειών και του ταξιδιωτικού κλάδου.

5. Για την παρακολούθηση ασθενών σε νοσοκομειακές μονάδες και την πρόβλεψη αλλαγών στην κατάστασή τους για να διασφαλιστεί έγκαιρη αγωγή σε περιπτώσεις έκτακτης ανάγκης.

6. Για τη σύνδεση, αποθήκευση και διάθεση δεδομένων που παράγονται από διάφορες τμήματα μιας εταιρείας.

7. Για την υποστήριξη πλατφορμας δεδομένων, αρχιτεκτονικών εξυπηρετούμενων από γεγονότα και μικρουπηρεσιών.

Σχήμα 5.2: Event Streaming

Ο Apache Kafka συνδιάζει τρεις δυαντότητες - κλειδιά για την υλοποίηση end to end event streams σε μία εφαρμογή:

1. Publish (εγγραφή) και Subscribe (ανάγνωση) των event streams περιλαμβάνοντας παράλληλα και είσοδο/έξοδο των αρχείων σε άλλα συστήματα.

2. Αποθήκευση των streams με σιγουριά και σταθερότητα για μεγάλο χρονικό διάστημα

3. Επεξεργασία των streams όπως προκύπτουν ή και σε πραγματικό χρόνο

Εκτός αυτών όλη η λειτουργικότητα του Apache Kafka είναι υλοποιημένη σε περιβάλλον κατανεμημένο, scalable με ανοχή σε λάθη και ασφάλεια. Μπορεί να γίνει deploy σε υπολογιστές, εικονικά μηχανήματα ή και στο cloud.
[2]

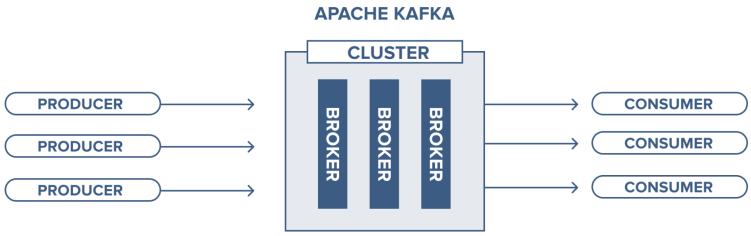
5.3 Λειτουργία του Kafka

Ο Kafka είναι ένα κατανεμημένο σύστημα που αποτελείται από διακομιστές (servers) και πελάτες (clients) οι οποίοι επικοινωνούν μέσω ενός TCP πρωτοκόλλου δικτύου υψηλής απόδοσης. Μπορεί να εγκατασταθεί σε φυσικό υλικό ή εικονικές μηχανές και containers τόσο σε εγκαταστάσεις εντός της επιχείρησης όσο και σε περιβάλλοντα cloud.

Διακομιστές (servers): Ο Kafka λειτουργεί ως ένα cluster ενός ή περισσότερων διακομιστών που μπορεί να καλύπτει πολλαπλά κέντρα δεδομένων . Μερικοί από αυτούς τους διακομιστές αποτελούν το επίπεδο αποθήκευσης, που ονομάζονται brokers. Άλλοι διακομιστές εκτελούν το Kafka Connect για την συνεχή εισαγωγή και εξαγωγή δεδομένων ως Event Streams για την ενσωμάτωση του Kafka με τα περιφερειακά συστήματα όπως σχεσιακές βάσεις δεδομένων, καθώς και άλλα συστήματα Kafka. Για να επιτρέπει την υλοποίηση μεγάλων απαιτήσεων επεξεργαστικής ισχύος, το cluster Kafka είναι highly scalable και ανθεκτικό σε σφάλματα: αν οποιοσδήποτε από τους διακομιστές αποτύχει, οι άλλοι διακομιστές

αναλαμβάνουν την εργασία τους για να διασφαλίσουν συνεχή λειτουργία χωρίς απώλεια δεδομένων.

Πελάτες(clients): Επιτρέπουν την ανάπτυξη κατανεμημένων εφαρμογών και μικρουπηρεσιών που διαβάζουν, γράφουν και επεξεργάζονται ροές γεγονότων παράλληλα, σε κλίμακα και με ανθεκτικότητα σε σφάλματα, ακόμη και σε περιπτώσεις προβλημάτων δικτύου ή αποτυχίας μηχανής. Ο Kafka παρέχει ορισμένους clients που συμπεριλαμβάνει αυτούς για τους οποίους επεκτείνουν δεκάδες clients που παρέχονται από την κοινότητα του Kafka: υπάρχουν πελάτες για Java ,Scala, συμπεριλαμβανομένης της υψηλού επιπέδου βιβλιοθήκης Kafka Streams, για Go, Python, C/C++ και πολλές άλλες γλώσσες προγραμματισμού, καθώς και REST APIs.



Σχήμα 5.3: Λετουργία του Kafka

5.4 Βασική ορολογία του Kafka

Ένα event καταγράφει το γεγονός ότι ¨κάτι συνέβη' στον κόσμο ή στην επιχείρησή. Ονομάζεται επίσης εγγραφή ή μήνυμα. Όταν διαβάζονται ή γράφονται δεδομένα στον Kafka, γίνονται σε μορφή event. Ένα event έχει ένα κλειδί, μια τιμή, ένα χρονικό σήμα και προαιρετικές επικεφαλίδες μεταδεδομένων. Ένα παράδειγμα γεγονότος είναι το παρακάτω:

- Κλειδί γεγονότος: Άλίκη'
- Τιμή γεγονότος: Κατέβαλε πληρωμή ύψους 200 στον Μομπ'
- Χρονικό σήμα γεγονότος: '25 Ιουνίου 2023, 2:06 μ.μ.'

Οι **παραγωγοί (producers)** είναι topics που δημοσιεύουν (produce) γεγονότα στον Kafka, και οι **καταναλωτές (consumers)** είναι εκείνοι που εγγράφονται (subscribe) διαβάζουν και επεξεργάζονται αυτά τα γεγονότα. Στον Kafka, οι παραγωγοί και οι καταναλωτές είναι πλήρως ανεξάρτητοι μεταξύ τους, πράγμα που είναι σημαντικό στοιχείο σχεδίασης για να

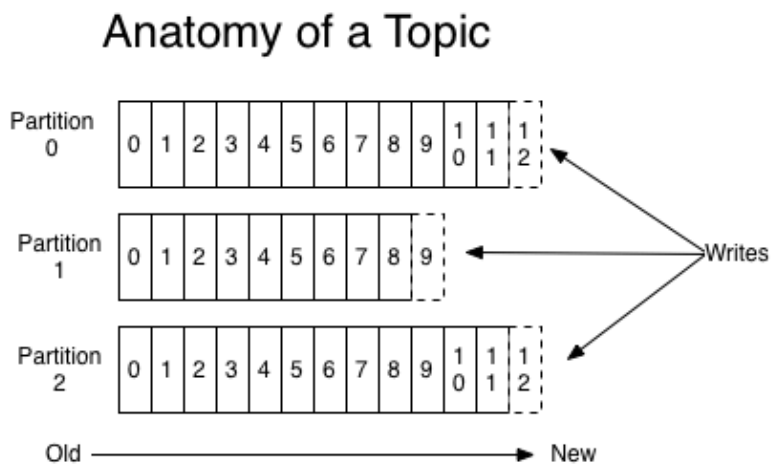
επιτευχθεί η υψηλή κλιμακωσιμότητα για την οποία είναι γνωστός ο Kafka. Για παράδειγμα, οι παραγωγοί δεν χρειάζεται να περιμένουν τους καταναλωτές. Ο Kafka παρέχει διάφορες εγγυήσεις, όπως η δυνατότητα επεξεργασίας γεγονότων με ακρίβεια.

Τα **events** οργανώνονται και αποθηκεύονται με ανθεκτικό τρόπο σε θέματα (topics). Απλούστατα, ένα topic είναι παρόμοιο με έναν φάκελο σε ένα αρχείο συστήματος, και τα events είναι τα αρχεία σε αυτόν τον φάκελο. Ένα παράδειγμα ονόματος topic μπορεί να είναι ¨πληρωμές¨. Στα topics υπάρχουν συνήθως πολλαπλοί producers και πολλαπλοί consumers: ένα topic μπορεί να έχει μηδέν, έναν ή πολλούς producers που γράφουν events σε αυτό, καθώς και μηδέν, έναν ή πολλούς consumers που λαμβάνουν πληροφορία για αυτά τα events. Τα events σε ένα topic μπορούν να διαβαστούν όσο συχνά χρειάζεται - σε αντίθεση με τα παραδοσιακά συστήματα μηνυμάτων, τα events δεν διαγράφονται μετά την κατανάλωσή τους. Αντ΄ αυτού, καθορίζετε πόσο καιρό ο Καφκα θα διατηρήσει τα γεγονότα μέσω μιας ρύθμισης ανά topic , μετά την οποία τα παλιά events θα απορρίπτονται. Η απόδοση του Kafka είναι σταθερή ως προς το μέγεθος των δεδομένων, οπότε η αποθήκευση δεδομένων για μεγάλο χρονικό διάστημα είναι εξασφαλισμένη.

Τα **topics** είναι χωρισμένα σε **partitions**, πράγμα που σημαίνει ότι ένα θέμα topic σε έναν αριθμό ¨κάδων' που βρίσκονται σε διάφορους διακομιστές **Kafka broker**. Αυτή η κατανομή των δεδομένων είναι πολύ σημαντική για την κλιμακωσιμότητα, διότι επιτρέπει στους topics να διαβάζουν και να γράφουν δεδομένα από/προς πολλούς διακομιστές ταυτόχρονα. Όταν ένα νέο event δημοσιεύεται σε ένα θέμα, προσαρτάται στην πραγματικότητα σε έναν από τα partition του topic. Γεγονότα με το ίδιο κλειδί (π.χ., αναγνωριστικό πελάτη ή οχήματος) γράφονται στον ίδιο κατανομέα, και ο Kafka εγγυάται ότι οποιοσδήποτε consumer ενός συγκεκριμένου topic-partition θα διαβάζει πάντα τα γεγονότα αυτού του topic-partition με ακριβώς την ίδια σειρά που γράφονταν.

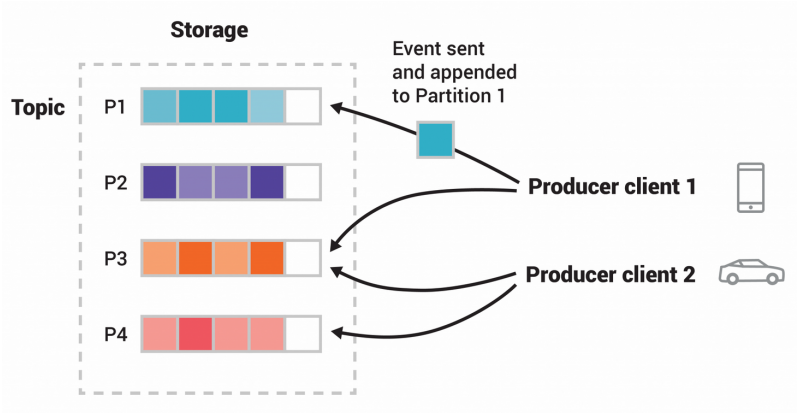
Ο **ZooKeeper** αποθηκεύει μεταδεδομένα για τους brokers του Kafka. Λειτουργεί ως διαμεσολαβητής μεταξύ του broker και των consumers, επιτρέποντας την επικοινωνία των κατανεμημένων διεργασιών μεταξύ τους μέσω ενός κοινού κεντρικού namespace δεδομένων που ονομάζεται znodes (καταχωρητές δεδομένων).

Ένας **broker** είναι ένας μεμονωμένος διακομιστής Kafka. Οι brokers λαμβάνουν μηνύματα από τους producers, τους αναθέτουν μοναδικές θέσεις (offsets) και αποθηκεύουν τα μηνύματα σε δίσκο. Ένα offset είναι μια μοναδική ακέραια τιμή που ο Kafka αυξάνει και προσθέτει σε κάθε μήνυμα κατά τη δημιουργία του. Τα offsets είναι κρίσιμα για τη διατήρηση της συνέπειας των δεδομένων σε περίπτωση αποτυχίας ή διακοπής, καθώς οι καταναλωτές χρησιμοποιούν τα offsets για να επιστρέψουν στο τελευταίο μήνυμα που είδαν μετά από μια αποτυχία. Οι



Σχήμα 5.4: Topic and partitions

brokers ανταποκρίνονται σε αιτήματα κλήσης partition από τους καταναλωτές και επιστρέφουν μηνύματα που έχουν καταχωριστεί στο δίσκο. Ένας μεμονωμένος broker λειτουργεί με βάση το συγκεκριμένο hardware και τις λειτουργικές του ιδιότητες. [2]



Σχήμα 5.5: Kafka broker: Topic and Event management

5.5 Kafka Core Enhancements

Πέρα από την κεντρική αρχιτεκτονική του Kafka, υπάρχουν δύο βασικές επεκτάσεις που επεκτείνουν τις δυνατότητες του Kafka: το Kafka Streams και το Kafka Connect.

- Kafka Streams: Το Kafka Streams είναι μια client βιβλιοθήκη που ενσωματώνει την επεξεργασία ροής με το Kafka. Επιτρέπει στους προγραμματιστές να δημιουργούν εφαρμογές και μικροϋπηρεσίες πραγματικού χρόνου όπου τα δεδομένα εισόδου και εξόδου

αποθηκεύονται σε Kafka Clusters. Με το Kafka Streams, γίενται δυνατή ή επεξεργασία ροής δεδομένων stateless και stateful. Τρέχει εντός της εφαρμογής ή το microservice, ξεχωριστά από τους Kafka brokers, και παρέχει ισχυρές δυνατότητες επεξεργασίας ροής με ελάχιστο latency

- Kafka Connect: Το Kafka Connect είναι ένα πλαίσιο που διευκολύνει τη σύνδεση του Kafka με εξωτερικές πηγές και αποδέκτες δεδομένων, όπως βάσεις δεδομένων και frameworks.

5.5.1 Kafka Streams

Το Kafka Streams είναι μια client βιβλιοθήκη με διεπαφή API για τη δημιουργία εφαρμογών και microservices, όπου τα δεδομένα εισόδου και εξόδου αποθηκεύονται σε Kafka Clusters. Οι εργασίες του Kafka Streams δεν εκτελούνται στους μεσίτες Kafka· αντίθετα, τρέχουν μέσα στην εφαρμογή ή το microservice σας.

Η προσέγγιση τουKafka Streams στην παράλληλη επεξεργασία μοιάζει με αυτή του Kafka:
Κάθε partition είναι μια πλήρως ταξινομημένη ακολουθία εγγραφών δεδομένων που αντιστοιχεί σε ένα topic partition Kafka. Μια εγγραφή δεδομένων στη ροή αντιστοιχεί σε ένα μήνυμα Kafka από αυτό το topic. Τα κλειδιά των εγγραφών δεδομένων καθορίζουν την κατανομή των δεδομένων τόσο στον Kafka όσο και στο Kafka Streams - δηλαδή, πώς δρομολογούνται τα δεδομένα σε συγκεκριμένα partition εντός των topic. Το Kafka Streams επεξεργάζεται μηνύματα σε πραγματικό χρόνο (δηλαδή όχι σε μικρά πακέτα), με χαμηλό latency. Υποστηρίζειstateless και stateful επεξεργασία και windows operations. Το Kafka Streams περιλαμβάνει δύο ειδικούς επεξεργαστές: έναν Source Processor και έναν Sink Processor.

- Ο Source Processor παράγει μια ροή διαβάζοντας εγγραφές από ένα ή περισσότερα Kafka topics και προωθεί τις εγγραφές αυτές στους Sink Processor.
- Ο Sink Processor στέλνει τυχόν ληφθέντα μηνύματα από τους Source Processor σε ένα καθορισμένο Kafka topic.Τα επεξεργασμένα αποτελέσματα μπορούν είτε να επιστρέψουν πίσω στο Καφκα είτε να γραφούν σε ένα εξωτερικό σύστημα.

5.5.2 Kafka Connect

Το Kafka Connect παρέχει ένα scalable μέσο μεταφοράς δεδομένων μεταξύ του Kafka και άλλων αποθηκών. Παρέχει APIs και ένα runtime για το σχεδιασμό και τη λειτουργία των προσθέτων συνδέσεων, τα οποία είναι frameworks που το Kafka Connect εκτελεί και σκοπεύει στη μετακίνηση δεδομένων.

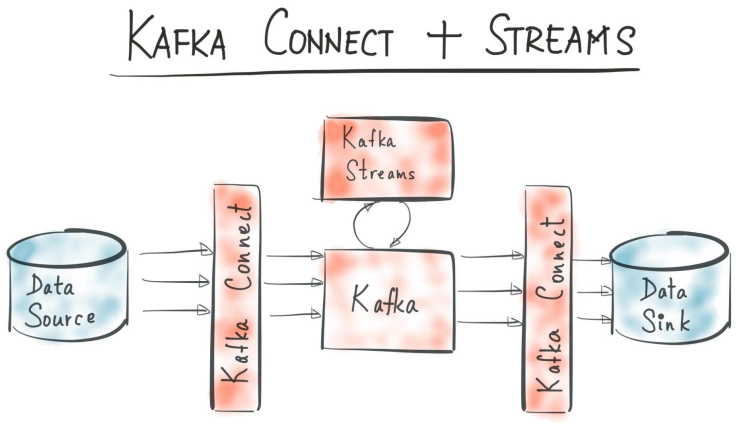
Ο connector ελέγχει:

- Τον ορισμό και τον αριθμό των tasks που λειτουργούν για τον connector.
- Τον τρόπο διαχωρισμού της αντιγραφής δεδομένων μεταξύ των tasks.

- Τη λήψη task configuration από τους workers και την παραπομπή τους στον προορισμό.

Οι connectors ξεκινούν tasks για τη μεταφορά δεδομένων παράλληλα σε κλίμακα και εκμεταλλεύονται βέλτιστα τους υπάρχοντες πόρους στους κόμβους των workers.

- Οι Source connector καταναλώνουν δεδομένα από το σύστημα προέλευσης και αποστέλλουν Kafka Connect data στους workers.
- ΟιSink connector λαμβάνουν δεδομένα Kafka Connector από τους workers και τα γράφουν στην αποθήκη δεδομένων του στόχου.



Σχήμα 5.6: Kafka connect and Kafka Streams

[14]

Κεφάλαιο 6

Time series και InfluxDB

6.1 Time series

Μια χρονοσειρά (Time series) είναι μια ακολουθία τιμών μιας μεταβλητής (π.χ. θερμοκρασία) τα οποία σχετίζονται με μία χρονική στιγμή η οποία δίνει νόημα στην ακολουθία π.χ. την στιγμή που γίνεται η μέτρηση της θερμοκρασίας σε ένα θερμίστορ. Κάποια παραδείγματα δεδομένων με χρονοσήμανση που μπορούν να θεωρηθούν χρονοσειρές, όπως αρχεία καταγραφής και μετρήσεις IoT συσκευών, μπορούν να θεωρηθούν χρονοσειρές. Οι μετρήσεις που απαρτίζουν μια χρονοσειρά για να παρουσιαστούν και να γίνουν αντιληπτές συνήθως τοποθετούνται σε ένα χρονοδιάγραμμα που αποκαλύπτει πληροφορίες για τα πιθανά μοτίβα, ή να αναδείξει κίνδυνο απο κάποια ασυμμετρία σε αυτά. Η διάταξη ανάλογα με τη χρονοσήμασνη έχει σημασία, καθώς υπάρχει εξάρτηση ανάμεσα στον χρόνο και τις μετρήσεις, και η αλλαγή χρονικής σειράς μπορεί να επηρεάσει την σημασία των δεδομένων. Παραδείγματα χρονοσειρών μπορεί να περιλαμβάνουν τις ωριαίες μετρήσεις θερμοκρασίας σε έναν συγκεκριμένο μετεωρολογικό σταθμό ή τις ημερήσιες μετρήσεις κλεισίματος μιας συγκεκριμένης μετοχής.

Σχήμα 6.1: Η τιμή της μετοχής της Microsoft αποτελεί ένα παράδειγμα χρονοσειράς

Οι χρονοσειρές χρησιμοποιούνται σε διάφορους τομείς, με τους πιο συνηθισμένους να περιλαμβάνουν:

- Πρόβλεψη χρονοσειρών (Time Series Forcasting): Η πρόβλεψη χρονοσειρών λαμβάνει πληροφορίες από τις ιστορικές τιμές της χρονοσειράς και αναλύει τα σχετικά μοτίβα για να κάνει μια εμπεριστατωμένη πρόβλεψη για τη μελλοντική της δραστηριότητα. Παραδείγματα χρήσης τέτοιων προβλέψεων περιλαμβάνουν η πρόβλεψη του καιρού, η πρόβλεψη πωλήσεων για το επόμενο τετράμηνο μιας εταιρίας ή ακόμη και η πρόβλεψη στην τιμή μιας μετοχής.

- Ανάλυση χρονοσειρών (Time Series Analysis): Η ανάλυση χρονοσειρών χρησιμοποιείται για να βγάλει πορίσματα για τον τρόπο με τον οποίο μια συγκεκριμένη μεταβλητή αλλάζει με την πάροδο του χρόνου. Για παράδειγμα, μπορεί να περιλαμβάνει την ανάλυση των δημοσκοπήσεων των προηγούμενων χρόνων για την ανάδειξη κοινωνικών μοντέλων με την πάροδο του χρόνου.

- Ανάλυση παλινδρόμησης (Regression Analysis): Η ανάλυση παλινδρόμησης μπορεί να χρησιμοποιηθεί για να εξετάσει πώς οι αλλαγές σε μια συγκεκριμένη μεταβλητή μπορούν να προκαλέσουν αλλαγές σε άλλες μεταβλητές κατά την ίδια χρονική περίοδο. Για παράδειγμα, μπορεί να περιλαμβάνει την οικονομική ανάλυση για το πώς οι τιμές των σιτηρών επηρεάζουν τις τιμές των ζωικών προιόντων και αντίστροφα.

Οι χρονοσειρές αποτελούν ένα σημαντικό εργαλείο για την κατανόηση των δεδομένων που εξελίσσονται με τον χρόνο και την πρόβλεψη μελλοντικών τάσεων και προβλέψεων. Η κατανόηση αυτών των αρχετύπων μπορεί να οδηγήσει σε πολλαπλές εφαρμογές και ανακαλύψεις σε ποικίλους τομείς της επιστήμης και της κοινωνίας.

6.1.1 Χαρακτηρηστικά των time series

Οι χρονοσειρές όπως αναφέρθηκε έχουν σίγουρα ώς μεταβλητή το χρόνο και μάλιστα δίνει συνήθως νόημα στην τιμή της μεταβλητής. Παρόλαυτα υπάρχουν και άλλες ιδιότητες που εμφανίζονται και χαρακτηρίζουν μια χρονοσειρά:

- Timestamp: Η πηγή των time series data προκαλείται από έναν προκαθορισμένο χρονοδιακόπτη ή ένα γεγονός, και όταν οι συσκευές συλλέγουν δεδομένα χρονοσειρών, κάθε εγγραφή συνδέεται πάντα με ένα χρονικό σημείο. Έτσι, τα δεδομένα χρονοσειρών μπορούν να ταξινομηθούν με βάση το χρονο· Σε μια time series σε κάθε εγγραφή δεδομένων, το timestamp αποτελεί το κλειδί για την ανάλυση και την επεξεργασία των δεδομένων

- Δομή: Τα time series data που παράγονται από συσκευές είναι πάντα σε δομές όπως json, συχνά έχοντας ένα προκαθορισμένο τύπο που συνήθως συμβαδίζει με την αντίστοιχη βάση χρονοσειρών.

- Ροή: Τα δεδομένα χρονοσειρών από συνδεδεμένες συσκευές μπορούν να θεωρηθούν ως μια ροή δεδομένων, συλλέγονται συνεχώς και ρέουν στη βάση δεδομένων. Αυτές οι ροές δεδομένων είναι ανεξάρτητες μεταξύ τους αφου η κάθε μία αφορά διαφορετικό πεδίο.

- Σταθερότητα: Παρόλο που η κλίμακα είναι μεγάλη, η συνολική κίνηση σε μια χρονοσειρά θα παραμείνει σταθερή και μπορεί να προβλεφθεί και υπολογιστεί με βάση τον αριθμό των συσκευών και την περίοδο δειγματοληψίας.

- Υψηλή αναλογία εγγραφής/ανάγνωσης: Γενικά, τα ακατέργαστα δεδομένα χρονοσειρών διαβάζονται σπάνια από λογισμικό ανάλυσης και άλλα παρόμοια εργαλεία για τη δημιουργία αναφορών και την εκτέλεση αλγορίθμων, αλλά γράφονται διαρκώς στη βάση δεδομένων.

- Αμεταβλητότητα: Τα συγκεκριμένα αρχεία δεδομένων χρονοσειρών σχεδόν ποτέ δεν ενημερώνονται ή διαγράφονται. Τα δεδομένα χρονοσειρών που παράγονται από συσκευές συνηθίζεται απλά να προστίθενται, παρόμοια με τα αρχεία καταγραφής.

- Retention policy: Στην πλειοψηφία των σεναρίων, καθορίζεται ένας κύκλος ζωής για τα συλλεγμένα δεδομένα χρονοσειρών, μετά τον οποίο διαγράφονται για μείωση των λειτουργικών εξόδων αποθήκευσης. Είναι αρκετά ασυνήθιστο να αποθηκεύονται τα ακατέργαστα δεδομένα για πάντα.

- Real-time computing: Για να ανταποκριθούν στις απαιτήσεις της επιχείρησης, τα συστήματα επεξεργασίας δεδομένων πρέπει να είναι σε θέση να εκτελούν λειτουργίες σε δεδομένα χρονοσειρών σε πραγματικό χρόνο - για παράδειγμα, ένα σύστημα παρακολούθησης πρέπει να είναι σε θέση να ενεργοποιείται αμέσως όταν πληρούνται οι συνθήκες για έναν συναγερμό.

- Aggregation: Οι περισσότερες αναζητήσεις στη βάση πραγματοποιούνται σε ένα καθορισμένο εύρος χρόνου, όχι σε όλα τα ιστορικά δεδομένα. Επιπλέον, η συγχώνευση δεδομένων απαιτείται πάντα για ένα χρόνικό διάστημα για όλες ή ένα υποσύνολο των συσκευών.

[3]

6.2 Time Series Databases

Μια Time Series Database (TSDB) είναι μια βάση δεδομένων που έχει κατασκευαστεί και βελτιστοποιηθεί για τα δεδομένα σε μορφή χρονοσειρών , όπως για παράδειγμα δεδομένα που λαμβάνονται απο έναν αισθητήρα. Τα δεδομένα αυτά μπορεί να αφορούν μετρήσεις σε κάποιον server, παρακολούθηση του performance κάποιας εφαρμογής, δεδομένα δικτύου, δεδομένα αισθητήρων, γεγονότα ή ακόμη και συναλλαγές σε κάποια αγορά τα οποία διαχειρίζονται χρονικά βέλτιστα και υπολογιστικά ελάχιστα απο μια Time Series Database.

Μια Time Series Database είναι κατασκευασμένη ειδικά για τον χειρισμό μετρήσεων και γεγονότων με χρονικές σημάνσεις και έχει βελτιστοποιηθεί για τη μέτρηση των αλλαγών με την πάροδο του χρόνου. Χαρακτηριστικά που καθιστούν τα δεδομένα χρονοσειρών πολύ διαφορετικά από άλλους τύπους δεδομένων, π.χ. στατικά ή παραμετρικά δεδομένα είναι ο διαχειρισμός του κύκλου ζωής τους και οι μεγάλες αναζητήσεις πολλών εγγραφών συνήθως με γνώμονα το χρόνο. Η χρήση των Time Series Databases ουσιαστικά αξιοποιείται σε δεδομένα με συχνές ανανεώσεις σε γνωστές τιμές όπως μετρήσεις τιμής μιας μετοχής ή δεδομένα πολυμορφικά που έχει νοήμα ο χρονος αναγνώρισης τους όπως η καταγραφή μιας παραβίασης ενός συναγερμού σε ένα δωμάτιο ενός σπιτιού. Σήμερα, η πλειονότητα των εταιρειών παράγουν μια μεγάλη ροή μετρήσεων και γεγονότων δεδομένα χρονοσειρών ,επομένως η ανάγκη για μια βάση δεδομένων TSDB είναι αναπόφευκτη

Οι Time Series Database πρώτης γενιάς επικεντρώθηκαν κυρίως στην ανάλυση χρηματοοικονομικών δεδομένων, στην αλληλεπίδραση της αγοράς των μετοχών και σε συστήματα που κατασκευάστηκαν για να επιλύσουν προβλήματα συναλλαγών. Ωστόσο, οι χρονοσειρές δεν αφορούν πλέον μόνο εφαρμογές στη χρηματοοικονομική ανάλυση - στην πραγματικότητα, αποτελούν μόνο μία από τις πολυάριθμες εφαρμογές σε διάφορους κλάδους της βιομηχανίας. Οι θεμελιώδης θεώρηση του data processing έχει αλλάξει δραματικά την τελευταία δεκαετία. Όλα έχουν γίνει διαμερισμένα, οι μονολιθικοί υπολογιστές έχουν εξαφανιστεί και έχουν αντικαταστάθει από υπολογιστές χωρίς διακομιστή, μικροδιακομιστές και containers.

[5]

6.2.1 Χρήση των Time Series Databases

Η χρήση των TSDB έχει αυξηθέι εκθετικά τα τελευταια χρόνια με ολοένα και περισσότερες εταιρίες να τις συμπεριλαμβάνουν στα συστήματα τους για την αποθήκευση χρονοσειρών. Η αύξηση αυτή οφείτεται κυρίως στην βελτιστοποίησή τους και την καλύτερη απόδοσή τους σε ταχύτητα και latency σε σχέση με τις παραδοσιακές σχεσιακές βάσεις δεδομένων. Αυτό οφείλεται κατά κύριο μέρος στις παρακάτω ιδιότητες των TSDB:

- Μειωμένη διάρκεια αποκλεισμού λειτουργίας: Σε πραγματικά σενάρια, όπου η διαθεσιμότητα είναι κρίσιμη ανά πάσα στιγμή, η αρχιτεκτονική ενός TSDB αποφεύγει οποιαδήποτε διακοπή λειτουργίας των δεδομένων ακόμη και σε περιπτώσεις δικτυακών διαμερισμάτων ή βλαβών στο υλικό.

- Χαμηλότερο κόστος: Η ευελιξία και η υψηλή ανοχή σφάλματος των χρονοσειρών βάσεων δεδομένων μεταφράζονται σε μείωση των απαιτούμενων πόρων για τη διαχείριση των διακοπών λειτουργίας. Χρησιμοποιείται κοινόχρηστο υλικό για γρήγορη και εύκολη κλιμάκωση, με αποτέλεσμα τη μείωση του λειτουργικού κόστους και του κόστους υλικού για την αναπροσαρμογή προς τα πάνω ή προς τα κάτω.

- Βελτιωμένες επιχειρηματικές αποφάσεις: Επιτρέποντας σε μια οργάνωση να παρακολουθεί και να αναλύει δεδομένα σε πραγματικό χρόνο, η TSDB τη βοηθά να λαμβάνει ταχύτερες και πιο ακριβείς προσαρμογές για αλλαγές στην υποδομή, την κατανάλωση

ενέργειας, τη συντήρηση των συσκευών ή άλλες σημαντικές αποφάσεις που επηρεάζουν την επιχείρηση.

Οι χρονοσειρών βάσεις δεδομένων μπορούν επίσης να χρησιμοποιηθούν ως εφαρμογές πραγματοποίησης γεγονότων για την παρακολούθηση δεδομένων αλληλεπίδρασης χρήστη/πελάτη και σε εργαλεία επιχειρηματικής νοημοσύνης για την παρακολούθηση βασικών μετρήσεων και τη γενική κατάσταση της επιχείρησης.

6.2.2 Popular Time Series Databases

Από το 2013 ο ιστότοπος https://db-engines.com/ παρακολουθέι την χρήση των TSBD σε παγκόσμια κλίμακα ως ένας ανεξάρτητος παράγοντας. Για την ανάδειξη της κατάταξης των TSBDs λαμβάνει υπόψιν τις εξής παραμέτρους:

- Τον αριθμό των αναζητήσεων και των αναφορών σε αυτές στο διαδίκτυο με τα στοιχεία να λαμβάνονται απο τις πλατφόρμες Google και Bing
- Το γενικό ενδιαφέρον για την εκάστοτε βάση δεδομένων (στοιχεία για αυτό λαμβάνονται απο τα Google Trends)
- Την συχνότητα των τεχνικών συζητήσεων που αφορούν στην εκάστοτε βάση δεδομένων σε ιστοσελίδες όπως StackOverflow, DBA StackExchange
- Τον αριθμό των θέσεων εργασίας που απαιτούν ειδίκευση σε κάποια απο αυτές απο μεγάλες πλατφόρμες όπως LinkedIn και Indeed
- Το πλήθος των σχετικών αναφορών στα μέσα κοινωνικής δικτύωσης

Τα αποτελέσματα έως τον Ιούνιο του 2023 είναι τα εξής:

Όπως βλέπουμε και απο το παρακάτω χρονοδιάγραμμα το οποίο διαθέτει η ίδια ιστοσελίδα σύγκρισης η InfluxDB παραμένει σταθερά η πιο δημοφιλής τα τελευταια χρόνια και αυτήν θα δούμε αναλυτικότερα στην επόμενη ενότητα.

6.3 InfluxDB

6.3.1 Εισαγωγή

Το InfluxDB είναι μιαTSDB που έχει αναπτυχθεί από την εταιρεία InfluxData. Είναι γραμμένο στη γλώσσα προγραμματισμού Go και χρησιμοποιείται για την αποθήκευση και

Rank			DBMS	Database Model	Score		
Jun 2023	May 2023	Jun 2022			Jun 2023	May 2023	Jun 2022
1.	1.	1.	InfluxDB	Time Series, Multi-model	31.26	+1.35	+1.40
2.	3.	3.	Prometheus	Time Series	8.01	+0.59	+1.69
3.	2.	2.	Kdb	Multi-model	8.00	-0.03	-1.12
4.	4.	4.	Graphite	Time Series	6.05	-0.21	+0.69
5.	5.	5.	TimescaleDB	Time Series, Multi-model	4.70	-0.03	+0.14
6.	7.	9.	DolphinDB	Time Series, Multi-model	3.88	+0.46	+2.24
7.	6.	7.	RRDtool	Time Series	3.69	+0.08	+1.26
8.	8.	6.	Apache Druid	Multi-model	3.26	+0.19	+0.32
9.	9.	15.	TDengine	Time Series, Multi-model	2.91	-0.04	+1.95
10.	12.	12.	QuestDB	Time Series, Multi-model	2.49	+0.25	+1.25
11.	10.	8.	OpenTSDB	Time Series	2.36	-0.14	+0.50
12.	11.	11.	GridDB	Time Series, Multi-model	2.26	-0.11	+0.98
13.	13.	10.	Fauna	Multi-model	1.80	-0.08	+0.47
14.	15.	18.	VictoriaMetrics	Time Series	1.38	+0.16	+0.82
15.	14.	13.	Amazon Timestream	Time Series	1.29	+0.02	+0.27
16.	16.	22.	M3DB	Time Series	1.16	-0.02	+0.75
17.	18.	14.	CratcDB	Multi modcl	0.96	0.04	0.03
18.	17.	16.	KairosDB	Time Series	0.95	-0.10	+0.30
19.	19.	17.	eXtremeDB	Multi-model	0.91	+0.01	+0.27
20.	20.	30.	ITTIA	Time Series, Multi-model	0.79	-0.05	+0.72

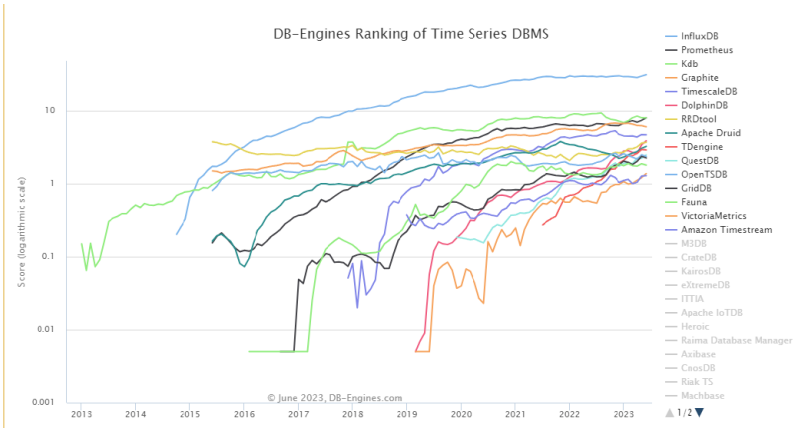
Σχήμα 6.2: Λίστα των 20 δημοφιλέστερων TSDB

ανάκτηση δεδομένων χρονοσειρών σε πεδία όπως η παρακολούθηση, οι μετρήσεις εφαρμογών, τα δεδομένα αισθητήρων (Internet of Things) και η αναλυτική επεξεργασία σε πραγματικό χρόνο. Υποστηρίζει επίσης την επεξεργασία δεδομένων από το Graphite.

Η εταιρεία Errplane, υποστηριζόμενη από την Y Combinator, ξεκίνησε την ανάπτυξη του InfluxDB ως ένα ανοικτού κώδικα έργο στα τέλη του 2013 για την παρακολούθηση απόδοσης και την αποστολή ειδοποιήσεων. Η Errplane συγκέντρωσε χρηματοδότηση σειράς A ύψους 8,1 εκατομμυρίων δολαρίων που οδήγησε η Mayfield Fund και η Trinity Ventures τον Νοέμβριο του 2014. Στα τέλη του 2015, η Errplane άλλαξε επίσημα την επωνυμία της σε InfluxData Inc. Η InfluxData συγκέντρωσε χρηματοδότηση σειράς B ύψους 16 εκατομμυρίων δολαρίων τον Σεπτέμβριο του 2016, ενώ Τον Φεβρουάριο του 2018, ολοκλήρωσε μια χρηματοδότηση σειράς ˜ ύψους 35 εκατομμυρίων δολαρίων με επικεφαλής επενδυτή την Sapphire Ventures.Ανακοινώθηκε επίσης μια νέα χρηματοδότηση ύψους 60 εκατομμυρίων δολαρίων το 2019.

Η InfluxDB δεν έχει εξωτερικά dependencies και παρέχει μια γλώσσα παρόμοια με την SQL, η οποία συνήθως λειτουργεί στο port 8086, με ενσωματωμένες λειτουργίες που επικεντρώνονται στον χρόνο εκτέλεσης ενός query που αποτελείται από μετρήσεις (measurements), σειρές τιμών και σημεία points. Κάθε point αποτελείται από αρκετά ζεύγη κλειδιού-τιμής που ονομάζονται σύνολο πεδίων (fieldset) και timestamp. Όταν ομαδοποιούνται μαζί από ένα σύνολο ζεύγους κλειδιού-τιμής που ονομάζεται σύνολο ετικετών (tagset), αυτά καθορίζουν μια χρονοσειρά. Τελικά, οι σειρές ομαδοποιούνται μαζί από ένα identification number για να σχηματίσουν μια μέτρηση (measurement). [4]

Μια βάση δεδομένων χρονοσειρών (Time Series Database) είναι εντελώς διαφορετική από μια σχεσιακή βάση δεδομένων (όπως η MySQL ή η PostgreSQL), καθώς δεν δημιουργεί



Σχήμα 6.3: Δημοτικότητα TSDB την τελευταία δεκαετία

συσχετίσεις μεταξύ των δεδομένων. Οι σχεσιακές βάσεις δεδομένων είναι ιδανικές για την αποθήκευση δεδομένων με στατικά δεδομένα όπως χρήστες και μηνύματα. Σε τέτοιες βάσεις δεδομένων, συνδέονται τα μηνύματα με τους χρήστες, έτσι ώστε να απαντώνται τα queries αναζήτησης. Οι Time Series Databases δεν έχουν δημιουργηθεί με σκοπό να εκτελέσουν τέτοιου είδους την εργασίες.

Σχήμα 6.4: InfluxDB logo

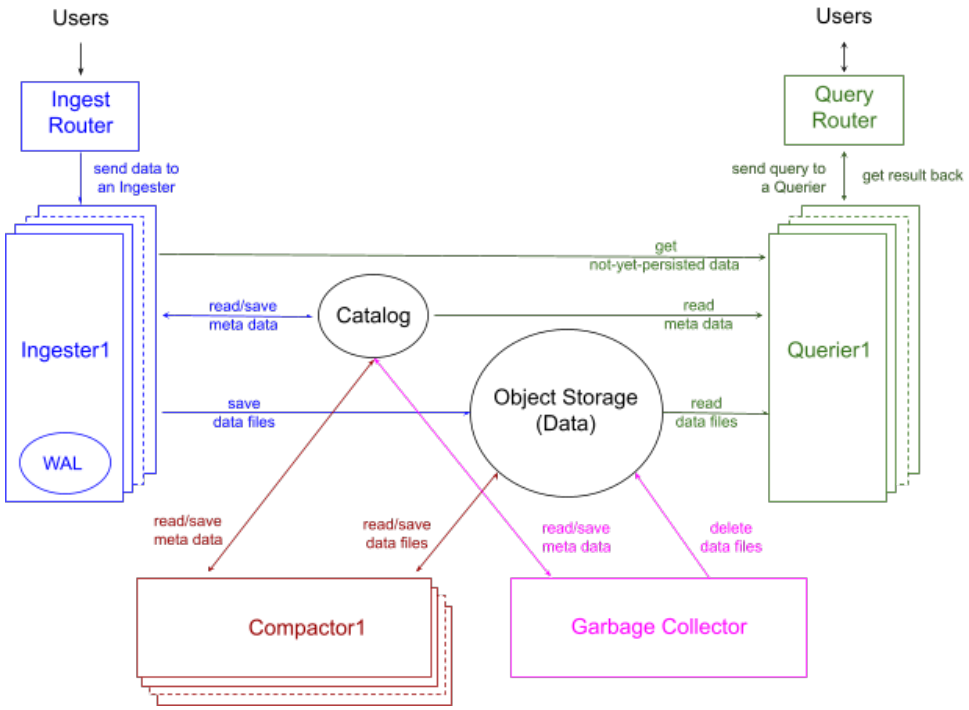
6.3.2 Αρχιτετονική της InfluxDB 3.0

Η InfluxDB 3.0 (προηγουμένως γνωστό ως InfluxDB IOx) είναι μια scalable (στο cloud) βάση δεδομένων που προσφέρει υψηλή απόδοση τόσο στη φόρτωση όσο και στην αναζήτηση (queries) των δεδομένων και επικεντρώνεται στις περιπτώσεις χρήσης χρονοσειρών. Η InfluxDB 3.0 περιλαμβάνει τέσσερις κύριες συνιστώσες και δύο κύριες αποθήκες. Οι τέσσερις συνιστώσες λειτουργούν σχεδόν ανεξάρτητα και είναι υπεύθυνες για:

- Φόρτωση δεδομένων (data ingestion)
- Data querying

- Συμπίεση δεδομένων (data compaction)
- Απομάκρυνση αχρήστων δεδομένων

Αυτές οι συνιστώσες απεικονίζονται με μπλέ, πράσινο, κόκκινο και ροζ αντίστοιχα στο παρακάτω διάγραμμα της αρχιτεκτονικής της InfluxDB :



Σχήμα 6.5: InfluxDB Architecture

Για τους δύο τύπους αποθήκευσης, ο ένας αφορά τα metadata του συστήματος, γνωστό ως κατάλογος (Catalog), ενώ ο άλλος είναι πολύ μεγαλύτερος και αποθηκεύει τα πραγματικά δεδομένα και ονομάζεται Αποθήκευση Αντικειμένων (Object Storage). Εκτός από αυτές τις κύριες τοποθεσίες αποθήκευσης, υπάρχουν πολύ μικρότερα αποθηκευτικά δεδομένων που ονομάζονται Καταγραφή Προτιμήσεων (Write Ahead Log - WAL) που χρησιμοποιούνται μόνο από τη συνιστώσα φόρτωσης δεδομένων και χρησιμοποιούνται για την ανάκτηση μετά από δυσλειτουργία του συστήματος κατά τη φόρτωση.

Τα βέλη στο διάγραμμα δείχνουν την κατεύθυνση της ροής των δεδομένων. Για τα ήδη αποθηκευμένα δεδομένα, σχεδιάστηκε το σύστημα έτσι ώστε ο Κατάλογος και η Αποθήκευση Αντικειμένων να είναι οι μοναδικοί καταχωρητες κατάστασης και να επιτρέπουν σε κάθε συνιστώσα να διαβάζει μόνο αυτές χωρίς την ανάγκη επικοινωνίας με άλλες συνιστώσες. Για τα δεδομένα που δεν έχουν αποθηκευτεί ακόμη, η συνιστώσα φόρτωσης δεδομένων διαχειρίζεται

την κατάσταση για να αποστείλει στη συνιστώσα querying όταν ένα ερώτημα φτάνει. Παρακάτω θα εξετάσουμε πιο αναλυτικά τις επιμέρους συνιστώσες της αρχιτεκτονικής:

Data ingestion

Οι χρήστες γράφουν δεδομένα στον δρομολογητή εισόδου (Ingest Router), ο οποίος διαχωρίζει τα δεδομένα σε έναν από τους ingesters. Ο αριθμός των ingesters στη συστοιχία μπορεί να κλιμακωθεί προς τα πάνω ή προς τα κάτω ανάλογα με το φόρτο εργασίας των δεδομένων. Αυτές οι αρχές κλιμάκωσης χρησιμοποιούνται για τον διαχωρίσμό των δεδομένων. Κάθε ingester διαθέτει έναν προσαρτημένο αποθηκευτικό χώρο, που χρησιμοποιείται ως αρχείο καταγραφής προγενέστερης εγγραφής (Write Ahead Log - WAL) για την ανάκτηση των δεδομένψν σε περίπτωση δυσλειτουργίας.

Κάθε ingester εκτελεί τα εξής σημαντικά βήματα:

- Προσδιορίζει τους πίνακες των δεδομένων: Σε αντίθεση με πολλές άλλες βάσεις δεδομένων, οι χρήστες δεν χρειάζεται να ορίσουν τους πίνακες και το σχήμα των στηλών τους πριν από τη φόρτωση δεδομένων στην InfluxDB. Θα ανακαλυφθούν και θα προστεθούν από τον ingester.

- Επικύρωση του σχήματος των δεδομένων: Οι τύποι δεδομένων που παρέχονται στην εγγραφή ενός χρήστη επικυρώνονται σύγχρονα με την αίτηση εγγραφής. Αυτό αποτρέπει τη διάδοση των συγκρούσεων τύπων στο υπόλοιπο σύστημα και παρέχει στον χρήστη άμεση ανατροφοδότηση.

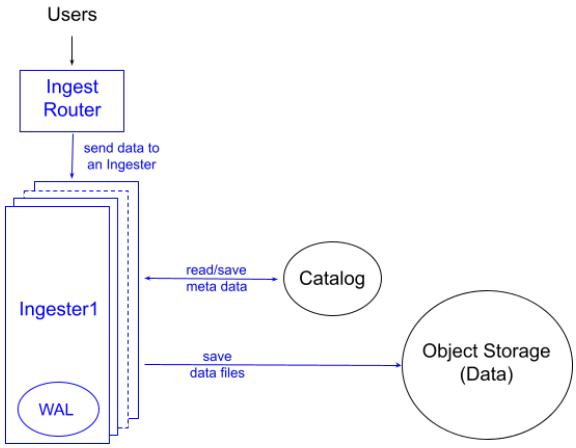
- Κατάτμηση των δεδομένων: Σε μια βάση δεδομένων μεγάλης κλίμακας όπως η InfluxDB, η κατάτμηση των δεδομένων έχει πολλά πλεονεκτήματα. Ο ingester είναι υπεύθυνος για την εργασία διαμέρισης και διαμερίζει τα δεδομένα ανά ημέρα στη στήλη 'time¨. Εάν τα δεδομένα εισόδου δεν έχουν στήλη time, ο Ingest Router την προσθέτει και ορίζει την τιμή της ως τον χρόνο φόρτωσης των δεδομένων.

- Αποδιπλασιασμός των δεδομένων: Σε περιπτώσεις χρήσης χρονοσειρών, είναι σύνηθες να βλέπουμε τα ίδια δεδομένα να εισάγονται πολλές φορές, οπότε η InfluxDB 3.0 εκτελεί τη διαδικασία αποδιπλασιασμού. Ο ingester δημιουργεί ένα αποδοτικό σχέδιο συγχώνευσης πολλαπλών στηλών για την εργασία αποδιπλασιασμού. Επειδή η InfluxDB χρησιμοποιεί το DataFusion για την εκτέλεση queries και το Arrow για την εσωτερική αναπαράσταση δεδομένων, η δημιουργία ενός σχεδίου συγχώνευσης περιλαμβάνει απλώς την τοποθέτηση των τελεστών ταξινόμησης και συγχώνευσης του DataFusion.

- Διατηρήση δεδομένων: Τα επεξεργασμένα και ταξινομημένα δεδομένα στη συνέχεια παραμένουν μόνιμα ως αρχείο Parquet. Επειδή τα δεδομένα κωδικοποιούνται/συμπιέζονται πολύ αποτελεσματικά εάν ταξινομούνται στις στήλες με το μικρότερο cardinality, ο ingester εντοπίζει και επιλέγει τις στήλες με το μικρότερο cardinality για τη σειρά της προαναφερθείσας ταξινόμησης. Ως αποτέλεσμα, το μέγεθος του αρχείου είναι συχνά 10-100 φορές μικρότερο από την ακατέργαστη μορφή του.

- Ενημέρωση του καταλόγου: Στη συνέχεια, ο ingester ενημερώνει τον κατάλογο σχετικά με την ύπαρξη του νεοδημιουργηθέντος αρχείου. Αυτό είναι ένα σήμα για να ενημερώσει τις άλλες δύο συνστώσες, Querier και Compactor, ότι έφτασαν νέα δεδομένα.

Παρόλο που ο ingester εκτελεί πολλά βήματα, η InfluxDB 3.0 βελτιστοποιεί τη διαδρομή της κάθε εγγραφής, διατηρώντας την καθυστέρηση ελάχιστη, της τάξης των χιλιοστών του δευτερολέπτου. Αυτό μπορεί να οδηγήσει σε πολλά μικρά αρχεία στο σύστημα. Ωστόσο, δεν διατηρούνται για πολύ καιρό. Οι compactors συμπιέζουν αυτά τα αρχεία στο παρασκήνιο.



Σχήμα 6.6: Data Ingestion

Data querying

Οι χρήστες στέλνουν ένα SQL ή ένα InfluxQL query στον Query Router, ο οποίος τα προωθεί σε έναν querier, ο οποίος με τη σειρά του διαβάζει τα απαραίτητα δεδομένα, δημιουργεί ένα πλάνο για το query, το εκτελεί και επιστρέφει το αποτέλεσμα στους χρήστες. Ο αριθμός των queriers μπορεί να κλιμακωθεί προς τα πάνω και προς τα κάτω ανάλογα με το φόρτο εργασίας του query, χρησιμοποιώντας τις ίδιες αρχές κλιμάκωσης που χρησιμοποιήθηκαν στο σχεδιασμό των ingester.

Κάθε querier εκτελεί τις εξής κύριες εργασίες:

- Cache Metadata: Για την αποτελεσματική υποστήριξη υψηλού φόρτου εργασίας query, ο querier συγχρονίζει συνεχώς την metadata cache του με τον κεντρικό κατάλογο, ώστε να έχει ενημερωμένους πίνακες και τα metadata που έχουν ενσωματωθεί.

- Ανάγνωση και προσωρινή αποθήκευση δεδομένων: Όταν φτάνει ένα query, εάν τα δεδομένα του δεν είναι διαθέσιμα στην cache του querier, αυτός διαβάζει πρώτα τα δεδομένα στην cache , επειδή από τα στατιστικά στοιχεία προκύπτει ότι τα ίδια αρχεία θα διαβαστούν πολλές φορές. Ο querier αποθηκεύει στην προσωρινή μνήμη μόνο το περιεχόμενο

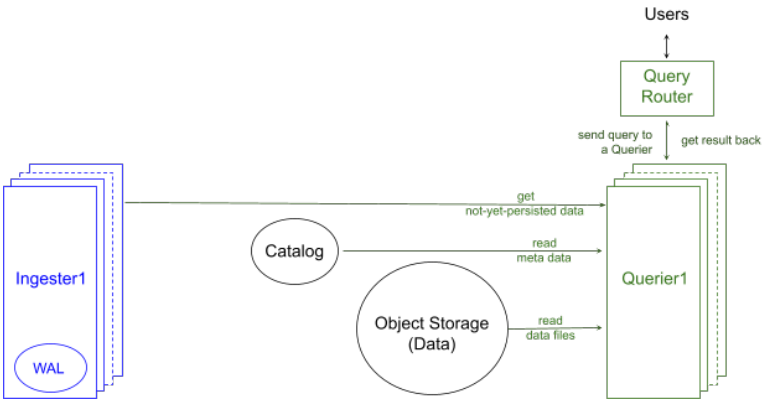
του αρχείου που απαιτείται για την απάντηση του query- το άλλο μέρος του αρχείου που δεν χρειάζεται το query με βάση τη στρατηγική κλαδέματος του querier δεν αποθηκεύεται ποτέ στην cache .

- Λήψη δεδομένων που δεν έχουν ακόμη αποθηκευτεί από τους ingester: Επειδή μπορεί να υπάρχουν δεδομένα στους ingester που δεν έχουν ακόμη ενσωματωθεί στο Object Storage, ο querier πρέπει να επικοινωνήσει με τους αντίστοιχους ingester για να λάβει αυτά τα δεδομένα. Από αυτή την επικοινωνία, ο querier μαθαίνει επίσης από τον ingester αν υπάρχουν νεότεροι πίνακες και δεδομένα που πρέπει να ακυρώσει και να ενημερώσει τις cache του, ώστε να έχει μια ενημερωμένη εικόνα ολόκληρου του συστήματος.

- Κατασκευάζει και εκτελεί ένα βέλτιστο σχέδιο query: Όπως πολλές άλλες βάσεις δεδομένων, ο querier της InfluxDB 3.0 περιέχει έναν Query Optimizer. Ο querier κατασκευάζει το καταλληλότερο σχέδιο ερωτήματος (ή αλλιώς βέλτιστο σχέδιο) που εκτελείται στα δεδομένα από την cache και τους ingester και ολοκληρώνεται στον ελάχιστο δυνατό χρόνο. Παρόμοια με το σχεδιασμό του ingester, ο querier χρησιμοποιεί το DataFusion και το Arrow για τη δημιουργία και εκτέλεση προσαρμοσμένων σχεδίων query για SQL και InfluxQL. Ο querier εκμεταλλεύεται την κατάτμηση δεδομένων που γίνεται στον ingester για να παραλληλοποιήσει το σχέδιο query και να αφαιρέσει τα περιττά δεδομένα πριν από την εκτέλεση του σχεδίου. Ο querier εφαρμόζει επίσης κοινές τεχνικές predicate και projection pushdown για να αφαιρέσει περαιτέρω τα δεδομένα το συντομότερο δυνατό.

Παρόλο που τα δεδομένα σε κάθε αρχείο δεν περιέχουν τα ίδια αντίγραφα, τα δεδομένα σε διαφορετικά αρχεία και τα δεδομένα που δεν έχουν ακόμη ενσωματωθεί και αποστέλλονται στον querier από τους ingester ενδέχεται να περιλαμβάνουν αντίγραφα. Έτσι, η διαδικασία αποδιπλασιασμού είναι επίσης απαραίτητη κατά τη στιγμή του query. Παρόμοια με τον ingester, ο querier χρησιμοποιεί τους ίδιους τελεστές συγχώνευσης ταξινόμησης πολλαπλών στηλών που περιγράφηκαν παραπάνω για την εργασία αποδελτίωσης. Σε αντίθεση με το σχέδιο που κατασκευάστηκε για τον ingester, αυτοί οι τελεστές αποτελούν απλώς μέρος ενός μεγαλύτερου και πιο σύνθετου σχεδίου query που κατασκευάστηκε για την εκτέλεση του query. Αυτό διασφαλίζει ότι τα δεδομένα ρέουν μέσω του υπόλοιπου σχεδίου μετά τον αποδιπλασιασμό.

Αξίζει να σημειωθεί ότι ακόμη και με έναν προηγμένο τελεστή συγχώνευσης ταξινόμησης πολλαπλών στηλών, το κόστος εκτέλεσής του δεν είναι ασήμαντο. Ο querier βελτιστοποιεί περαιτέρω το σχέδιο ώστε να αποδιπλασιάζει μόνο τα επικαλυπτόμενα αρχεία στα οποία μπορεί να συμβούν διπλότυπα. Επιπλέον, για να παρέχει υψηλή απόδοση ερωτημάτων στον querier, η InfluxDB 3.0 αποφεύγει όσο το δυνατόν περισσότερο τον αποδιπλασιασμό κατά τη διάρκεια του χρόνου query, συμπιέζοντας τα δεδομένα εκ των προτέρων.



Σχήμα 6.7: Data Querying

Data compaction

Για να μειωθεί η καθυστέρηση εισαγωγής, η ποσότητα των δεδομένων που επεξεργάζεται και εμμένει σε κάθε αρχείο από έναν ingester είναι ελάχιστη. Αυτό αφήνει πολλά μικρά αρχεία αποθηκευμένα στο Object Storage, τα οποία με τη σειρά τους δημιουργούν σημαντικό I/O κατά τη διάρκεια της υποβολής query και μειώνουν την απόδοση τους. Επιπλέον, τα επικαλυπτόμενα αρχεία ενδέχεται να περιέχουν αντίγραφα που χρειάζονται αποδιπλασιασμό κατά τη διάρκεια του χρόνου υποβολής query, γεγονός που μειώνει την απόδοση των query. Η δουλειά της συμπίεσης δεδομένων είναι να συμπιέζει πολλά μικρά αρχεία που εισέρχονται από τους ingester σε λιγότερα, μεγαλύτερα και μη επικαλυπτόμενα αρχεία για να κερδίσει σε απόδοση.

Το σχήμα 6.8 απεικονίζει την αρχιτεκτονική της συμπίεσης δεδομένων, η οποία περιλαμβάνει έναν ή πολλούς compactor. Κάθε compactor εκτελεί μια εργασία παρασκηνίου που διαβάζει τα πρόσφατα εισερχόμενα αρχεία και τα συμπιέζει σε λιγότερα, μεγαλύτερα και μη επικαλυπτόμενα αρχεία. Ο αριθμός των compactor μπορεί να κλιμακωθεί προς τα πάνω ή προς τα κάτω ανάλογα με τον φόρτο εργασίας συμπίεσης, ο οποίος είναι συνάρτηση του αριθμού των πινάκων με νέα αρχεία δεδομένων, του αριθμού των νέων αρχείων ανά πίνακα, του μεγέθους των αρχείων, του αριθμού των υφιστάμενων αρχείων με τα οποία επικαλύπτονται τα νέα αρχεία και του εύρους ενός πίνακα (τον αριθμό στήλων του πίνακα).

Οι λεπτομερείς εργασίες ενός συμπιεστή αφορούν:

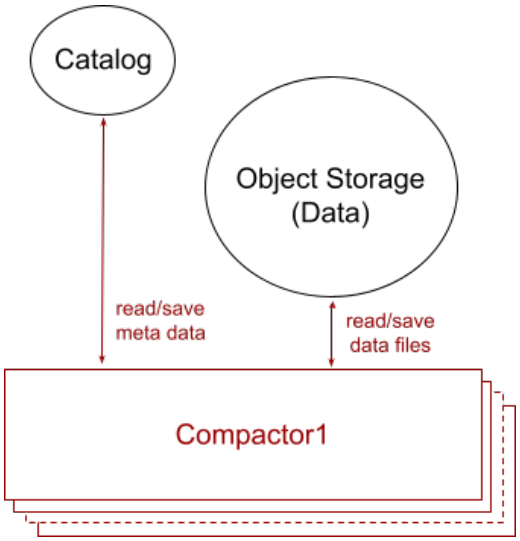
1. Στο πώς κατασκευάζει ένα βελτιστοποιημένο σχέδιο αποσυμπίεσης που συγχωνεύει αρχεία δεδομένων

2. Στη σειρά ταξινόμησης των αρχείων με διαφορετικές στήλες που βοηθά στην αποσυμπίεση

3. Στη χρήση επιπέδων συμπίεσης για την επίτευξη μη επικαλυπτόμενων αρχείων με ταυτόχρονη ελαχιστοποίηση των επανασυμπιέσεων και τη δημιουργία ενός βελτιστοποιημένου σχεδίου αποσυμπίεσης σε ένα μείγμα μη επικαλυπτόμενων και επικαλυπτόμενων αρχείων στον querier.

Όπως και ο σχεδιασμός του ingester και του querier, ο compactor χρησιμοποιεί το DataFusion και το Arrow για τη δημιουργία και εκτέλεση προσαρμοσμένων σχεδίων query. Στην πραγματικότητα, και τα τρία στοιχεία μοιράζονται το ίδιο υπο-σχέδιο συμπίεσης που καλύπτει τόσο τον αποδιπλασιασμό δεδομένων όσο και τη συγχώνευση.

Τα μικρά ή/και επικαλυπτόμενα αρχεία που συμπιέζονται σε μεγαλύτερα και μη επικαλυπτόμενα αρχεία πρέπει να διαγραφούν για την ανάκτηση χώρου. Για να αποφευχθεί η διαγραφή ενός αρχείου που διαβάζεται από ένα query, ο compactor δεν διαγράφει ποτέ πλήρως κανένα αρχείο. Αντ' αυτού, επισημαίνει τα αρχεία ως soft διαγραμμένα στον κατάλογο και μια άλλη υπηρεσία παρασκηνίου που ονομάζεται Garbage Collector διαγράφει τελικά τα soft διαγραμμένα αρχεία για να ανακτήσει χώρο.



Σχήμα 6.8: Data compaction

Garbage collection

Το Σχήμα 6.9 απεικονίζει το σχεδιασμό της συλλογής σκουπιδιών (Garbage collection) της InfluxDB 3.0 που είναι υπεύθυνη για τη διατήρηση δεδομένων και την ανάκτηση χώρου. Ο garbage collector εκτελεί εργασίες παρασκηνίου που προγραμματίζουν την soft και hard διαγραφή δεδομένων.

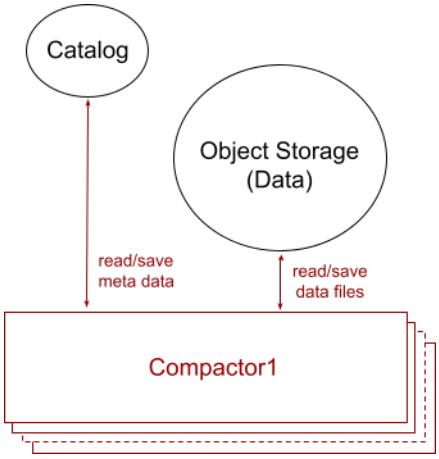
Διατήρηση δεδομένων:

Η InfluxDB 3.0 παρέχει στους χρήστες τη δυνατότητα να ορίζουν την πολιτική διατήρησης (retention policy) δεδομένων τους και να την αποθηκεύουν στον κατάλογο. Η προγραμματισμένη εργασία παρασκηνίου του garbage collector διαβάζει τον κατάλογο για πίνακες που βρίσκονται εκτός της περιόδου διατήρησης και σημειώνει τα αρχεία τους ως soft deleted στον κατάλογο. Αυτό σηματοδοτεί στους queriers και τους compactors ότι τα αρχεία αυτά δεν

είναι πλέον διαθέσιμα για αναζήτηση και συμπίεση, αντίστοιχα.
Ανάκτηση χώρου:

Μια άλλη προγραμματισμένη εργασία παρασκηνίου του garbage collector διαβάζει τον κατάλογο για τα metadata των αρχείων που διαγράφηκαν μαλακά πριν από ορισμένο χρονικό διάστημα. Στη συνέχεια αφαιρεί τα αντίστοιχα αρχεία δεδομένων από το Object Storage και αφαιρεί επίσης τα metadata από τον κατάλογο.

Τα soft deleted αρχεία προέρχονται από διαφορετικές πηγές: συμπιεσμένα αρχεία που διαγράφηκαν από τους compactors, αρχεία εκτός της περιόδου διατήρησης που διαγράφηκαν από τον ίδιο τον garbage collector και αρχεία που διαγράφηκαν μέσω μιας εντολής delete. Η εργασία hard delete δεν χρειάζεται να γνωρίζει από πού προέρχονται τα soft delete και τα αντιμετωπίζει όλα με τον ίδιο τρόπο.



Σχήμα 6.9: Garbage collection

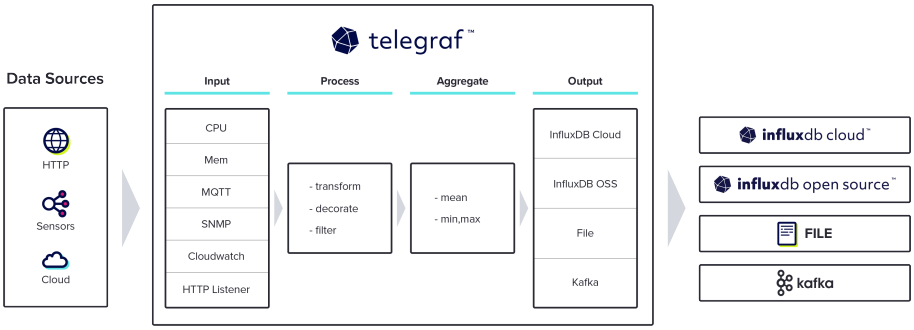
[8]

6.4 Telegraf

6.4.1 Εισαγωγή

Το Telegraf είναι ένας πράκτορας βασισμένος σε διακομιστή για τη συλλογή και αποστολή όλων των μετρήσεων και συμβάντων από βάσεις δεδομένων, συστήματα και αισθητήρες IoT. Το Telegraf είναι γραμμένο στην γλώσσα προγραμαμτισμού Go και μεταγλωττίζεται σε ένα ενιαίο δυαδικό αρχείο χωρίς εξωτερικές εξαρτήσεις, ενώ απαιτεί ελάχιστο αποτύπωμα μνήμης.

Το Telegraf είναι ένας πράκτορας βασισμένος σε διακομιστή ανοικτού κώδικα που διαθέτει plugins για τη συλλογή δεδομένων από περισσότερες από 250 διαφορετικές πηγές δεδομένων σαν input και πρωτόκολλα δικτύου. Τα δεδομένα μπορούν στη συνέχεια να αποδεχθούν απο 50 διαφορετικές επιλογές εξόδου είτε για αποθήκευση είτε για επεξεργασία.



Σχήμα 6.10: Garbage collection

[16]

6.4.2 Χρήση του Telegraf

Το Telegraf χρησιμοποιείται στις εφαρμογές IoT ως ενδιάμεσος φορέας δεδομένων εξασφαλίζοντας έμπιστη μεταφορά δεδομένων απο ένα σύστημα σε ένα άλλο και βοηθώντας στην ομαλή μεταφορά των δεδομένων χωρίς αλλοιώσεις ή χωρίς προβλήματα στη μορφή ή την αποθήκευση των δεδομένων. Αποτελείται από την κύρια διαδικασία και ένα βολικό οικοσύστημα πρόσθετων λειτουργιών που συνδυάζει υπηρεσίες εισόδου και εξόδου. Για παράδειγμα, μπορεί να χρησιμοποιήθούν συμβατά plugins για τη συλλογή μετρήσεων και να την αποστολή τους σε συμβατές εξόδους, όπως άλλες βασεις δεδομένων ή υπηρεσίες. Αυτά τα plugins χρησιμοποιούν τη μορφή γραμμής πρωτοκόλλου της Influx, η οποία ορίζει μια απλή αλλά λειτουργική μορφή για την εργασία με μετρικά σημεία. Ο πράκτορας Telegraf ενεργεί στη συνέχεια ως προσαρμογέας και διοχετεύει μετρήσεις από διάφορες πηγές σε καταχωρημένες εξόδους. Τα plugins του Telegraf χωρίζονται σε 4 βασικές κατηγορίες, συγκεκριμένα:

1. Input Plugins, τα οποία είναι υπέυθυνα για την συλλογή και ρο φιλτράρισμα των στοιχείων απο μια γκάμα διεργασιών και υπηρεσιών.

2. Process Plugins, τα οποία είναι υπέυθυνα για την ενδιάμεση επεξεργασία των δεδομένων που μεταφέρονται

3. Aggregate Plugins, τα οποία δημιουργούν συγκεντρωτικές μετρήσεις, για παράδειγμα το μέσο όρο, το ελάχιστο η μέγιστο από τις μετρήσεις που έχουν συλλέχθει και επεξεργαστεί.

4. Output Plugins, τα οποία γράφουν τςι μετρήσεις σε μια πληθώρα βάσεων δεδομένων, services και ουρών, όπως InfluxDB, Graphite, OpenTSDB, Datadog, Kafka, MQTT, NSQ

Μέρος II

Πρακτικό Μέρος

Κεφάλαιο 7

Υλοποίηση του συστήματος πρόβλεψης

7.1 Εισαγωγή

Η πληροφορία του συστήματος αποτελείται απο τον προσδιορισμό τριών παραμέτρων απο τη ζωντανή ροή της κάμερας, του αριθμού των ατόμων στο εκάστοτε frame, τον αριθμό των ατόμων που φορούν μάσκα και τον αρθμό των ατόμων που δεν φορουν. Αυτή η πληροφορία, όπως θα δούμε και στο επόμενο κεφάλαιο θα μεταφερθεί εως ότου φτάσει στη βάση δεδομένων. Σε αυτό το κεφάλαιο θα αναλυθεί η διαδικασία και τα μοντέλα που χρησιμοποιήθηκαν στα πλαίσια ανάπτυξης της συγκεκριμένης διπλωματικής εργασίας. Συνολικά έχουν γίνει δυο υλοποιήσεις, όλες με Machine Learning μοντελα που μπορούν να μας δώσουν αυτήν την πληροφορία απο την ζωντανή ροή της κάθε κάμερας. Η πρώτη υλοποίηση είναι και η πιο straightforward, συγκεκριμένα αποτελειται απο δύο μοντέλα αναγνώρισης, βασισμένα στο YOLO που αναλύθηκε στην ενότητα 4.6 που τρέχουν παράλληλα σε κάθε frame. Το ένα μοντέλο αναγνωρίζει τον αριθμό των προσώπων με μάσκα και αυτών χωρίς, ενώ το δεύτερο αναγνωρίζει τον αριθμό των ατόμων στο frame. Αυτή η μέθοδος αν και γρήγορη έχει διάφορα προβλήματα όπως το γεγονός ότι το μοντέλο αναγνώρισης μάσκας μπορεί να μην εντοπίζει καθόλη την κίνηση του ατόμου στο οπτικό πεδίο της κάμερας την ύπαρξη η μη μάσκας, παρα μόνο σε μερικά πλάνα. Αυτό έχει ως αποτέλεσμα να μην αποστέλεται ο σωστος αριθμός των ατόμων που φορούν και δεν φορύν μάσκα στο σύστημα. Στην πρόσπάθεια για την αντιμετώπιση αυτού του προβλήματος έχουν γίνει οι άλλες δύο υλοποιήσεις που στηρίζονται στο πρόβλημα Multiple Object Tracking (MOT). Το πρόβλημα Multiple Object Tracking (MOT) είναι το πρόβλημα της ανάθεσης αναγνωριστικών σε κάθε αντικείμενο αναγνώρισης (άτομο στην περίπτωσή μας) σε ένα βίντεο και την διατήρηση των ίδιων αναγνωριστικών για τα ίδια αντικείμενα σε όλη τη διάρκεια του. Αυτό δίνει μια νέα οπτική, προσθέτοντας μνήμη στο σύστημα, αφού μπορούμε να αναθέτουμε σε εναν πίνακα το αν φοράει η όχι μάσκα κάποιο άτομο μόνο όταν γίνεται εντοπισμός μάσκας ή όχι μασκας και να διατηρούμε αυτή την πληροφορία, εως ότου γίνει εκ νέου αναγνώριση μασκας ή μη μάσκας για το ίδιο άτομο. Με τον τρόπο αυτό μειώνονται κατα πολύ οι αυξομειώσεις στον αριθμό των ατόμων που φορούν και δε

φορούν μάσκα και το αποτέλεσμα είναι πιό ακριβές. Σαφώς όμως όπως και θα παρουσιάσουμε υπάρχει και το trade-off ταχύτητας και ακρίβειας, οπότε η ταχύτητα αυτών των μοντέλων είναι μειωμένη σε σχέση με την αρχική, απλούστερη προέγγιση.

7.2 Πρώτη προσέγγιση με YOLOv8

Όπως αναφέρθηκε η πρώτη προσέγγιση αποελείται απο δύο μοντέλα βασισμένα στο YOLO (You Only Look Once), ένα για την αναγνώριση ατόμων και ένα για την αναγνώριση προσώπου χωρις μασκα και προσώπου με μάσκα. Το πρώτο μοντέλο είναι το προεκπεδευμένο μοντέλο στο COCO dataset YOLOv8m το οποίο έχει προεκπεδευτεί στην ανανώριση ανθρώπων , συγκεκριμένα είναι η κλάση με δεικτη 0 οπότε η διαδικασία που έγινε είναι το φιλτράρισμα των αναγνωρίσεων στην κλάση με δείκτη 0 του προεκπεδευμένου μοντέλου και στη συνέχεια το aggregation των προβλέψεων καταλήγοντας στο συνολικό αριθμό ανθρώπων που εντοπίστηκαν στην κάθε εικόνα ή frame. Από το [18] λαμβάνουμε και τον παρακάτω πίνακα μετρικών που αφορούν όλα τα μοντέλα YOLOv8:

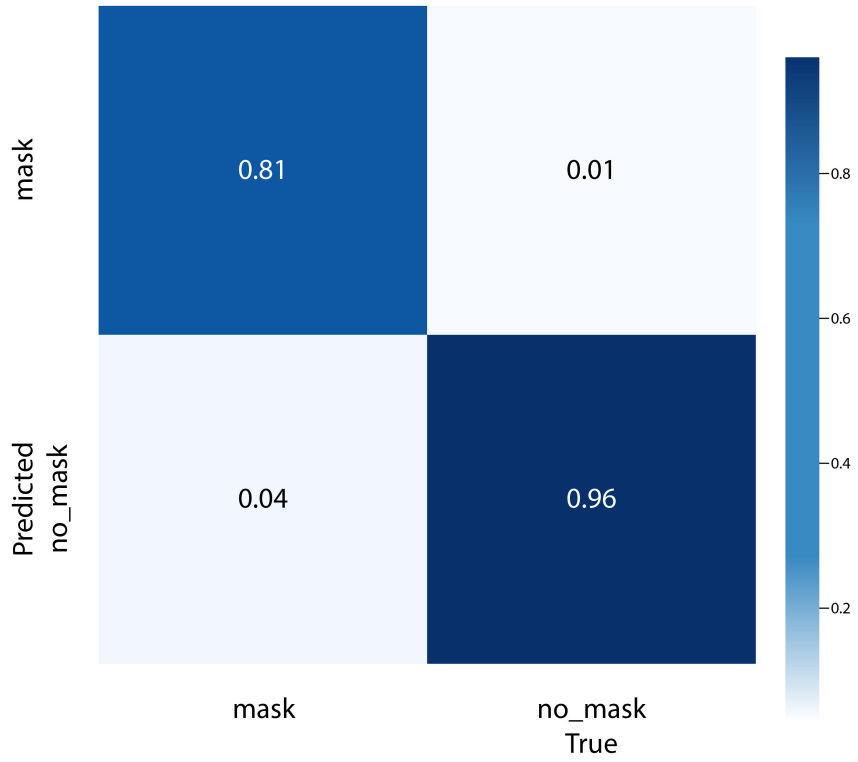
Model	size (pixels)	mAP^{val} 50-95	Speed CPU ONNX (ms)	Speed A100 TensorRT (ms)	params (M)	FLOPs (B)
YOLOv8n	640	37.3	80.4	0.99	3.2	8.7
YOLOv8s	640	44.9	128.4	1.20	11.2	28.6
YOLOv8m	640	50.2	234.7	1.83	25.9	78.9
YOLOv8l	640	52.9	375.2	2.39	43.7	165.2
YOLOv8x	640	53.9	479.1	3.53	68.2	257.8

Σχήμα 7.1: Μετρικές μοντέλων YOLOv8 για το COCO Dataset

Το δεύτερο μοντέλο είναι και το πιο σημαντικό γιατί χρησιμοποιήθηκε σε όλες τις υλοποιήσεις για την αναγνώριση μάσκας. Στην παρούσα υλοποίηση χρησιμοποιείται για τον απ΄ ευθείας υπολογισμό του πλήθους των ατόμων που φορούν και των ατόμων που δε φορούν μάσκα, ενώ στις υπόλοιπες υλοποιήσεις έμμεσα. Το μοντέλο για την αναγνώριση μάσκας, λοιπόν, βασιστήκε στο Transfer learning. Χρησιμοποιήθηκε πάλι το YOLOv8m ως προεκπεδευμένο μοντέλο και στη συνέχεια επανεκπαιδεύτηκε σε καινούριο dataset που αφορούσε την ύπαρξη η μή μάσκας. Συγκεκριμένα λοιπόν το dataset που χρησιμοποιήθηκε το `ηττπς://ωωω.καγγλε.cομ/δατασετς/αδιτψα276/φαcε-μασκ-δατασετ-ψολο-φορματ` το οποίο περιέχει 1840 φωτογραφίες με μάσκες η χωρίς και τα αντίσοτιχα annotations σε YOLO format, δηλαδή με τις συντεταγμένες των bounding boxes καθώς και την κλάση που αντιστοιχεί σε αυτά. Πρόκειται για ένα αρκετά καλο dataset με ευρος διαφορετικών οπτικών γωνιών αλλα

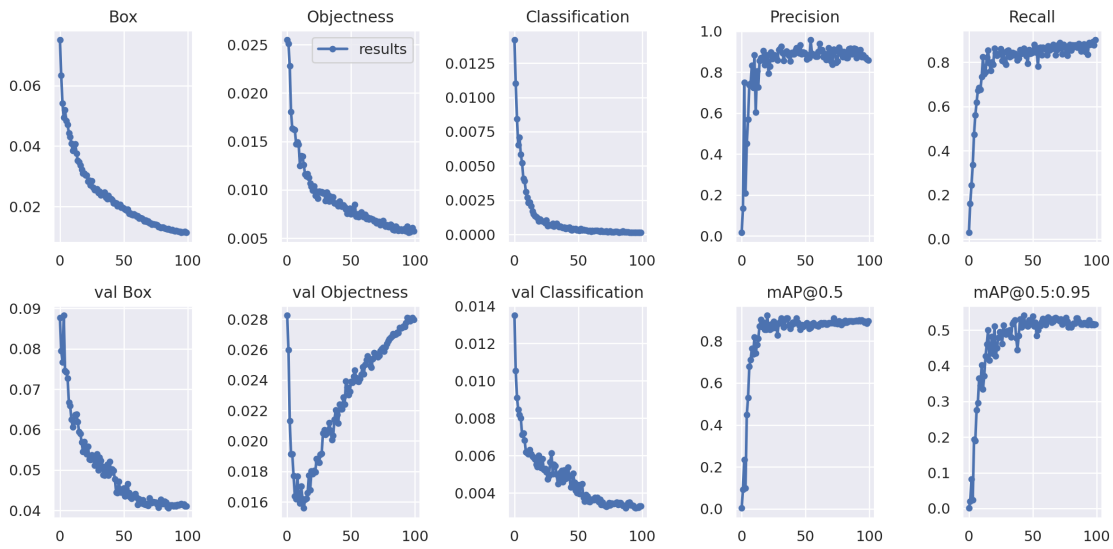
και ανθρώπων. Τα δεδομένα είναι χωρισμένα σε αναλογίες train - test - validation 76% - 13% - 11% όπως και διατηρήθηκαν. Η εκπαίδευση έγινε σε 100 εποχές θεωρώντας εικόνες 640 pixels κανοντας resizing όπου ήταν απαραίτητο. Παρακάτω προκύπτουν και εικονικά τα αποτελέσματα της επανεκπαίδευσης του μοντέλου.

Αρχικά βλεπουμε το confusion matrix που μας εμφανίζει στον κάθετο άξονα τις προβλέψεις και στον οριζόντιο τα πραγματικά δεδομένα καθώς και τα ποσοστά πρόβλεψης, ευρίσκοντας έτσι τα True Positive , True Negative, False Positive, False Negative (TP,TN,FP,FN). Απο αυτά προκύπτουν και οι μετρικές $Precision = \frac{TP}{TP+FP}$, $Recall = \frac{TP}{TP+FN}$ και $F_1 = 2\frac{Precision \times Precision}{Precision+Precision}$, των οποίων η βελτίωση ανα εποχή φαίνεται στα παρακάτω γραφήματα.



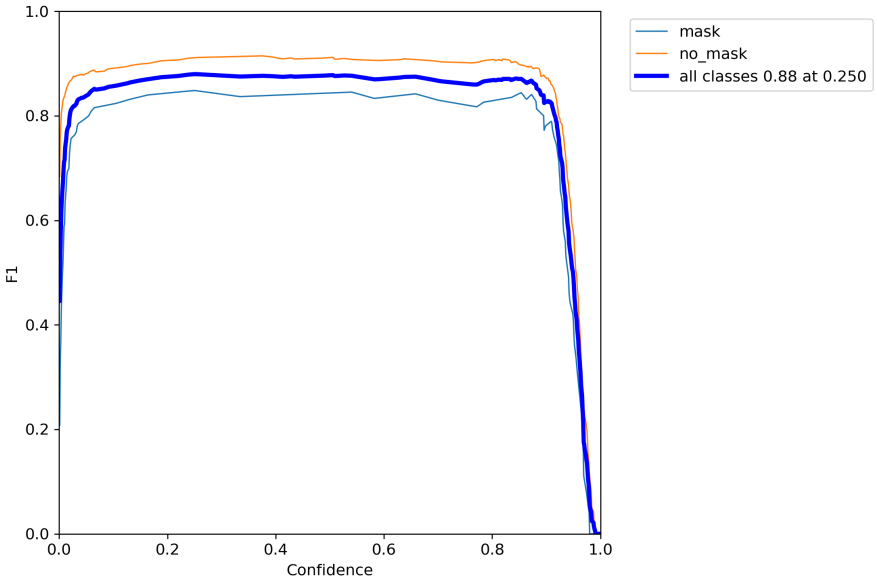
Σχήμα 7.2: Confusion Matrix

Παραπάνω φαίνονται αρκετές σημαντικές μετρικές του μοντέλου κατα τη διάρκεια της εκπαίδευσής του. Ξεκινώντας πάνω αριστερά έχουμε το box loss, μια μετρική του πόσο κοντά είναι τα προβλεπόμενα περικλειόμενα κουτιά σε σχέση με τα πραγματικά. Στην τρίτη θέση υπάρχει το Classification Loss που μετρά το πόσο σωστά κατηγοριοποίησε το περικλειόμενο



Σχήμα 7.3: Αποτελέσματα Training

κουτί το μοντέλο ως μάσκα η όχι μάσκα. Στην κάτω σειρά έχουμε μια επίσης εξίσου σημαντική μετρική, το mAP (Mean Average Precision) σε δύο εκφάνσεις, με όριο Intersection over Union 0.5 και απο 0.5 εώς 0.95.



Σχήμα 7.4: F1 curve

Μπορούμε να πούμε ότι το μοντέλο δίνει αρκετά καλές προβλέψεις και ότι οι 100 εποχές ήταν απαραίτητες για την καλή εκπαίδευσή του αφού είδαμε σημαντικές βελτιώσεις στις αντίστοιχες μετρικές. Τέλος αξίζει να αναφέρουμε ότι το παραπάνω μοντέλο σε λειτουρ-

γία βίντεο μπορεί να τρέξει σε 30fps το οποίο θεωρείται καλό για ζωντανό χρόνο το οποίο επιθυμούμε.

7.3 Deep Sort και YOLOv8

Η αρχική υλοποίηση έδωσε κάποια ικανοποιητικα αποτελεσματα οσον αφορά στην ακρίβεια των μετρήσεων καθως και στην ταχύτητα που το μοντελο τρεχει. Παρατηρώντας αρκετά παραδειγματα βιντεο στα οποία έγινε test το μοντέλο εντοπίστηκε ενα αξιοσημείωτο πρόβλημα: υπάρχουν μεγάλες αυξομοιώσεις στον αριθμό των ανθρώπων που φορουν μασκα καθως και αυτων που δε φορουν. Αυτό συμβαίνει διότι σε διαδοχικά frames τυχαίνει στο ένα να γίνεται αναγνώριση μάσκας καπο το μοντέλο αναγνώρισης και στο άλλο να μην γίνεται. Η λύση που προτείνεται για την αντιμετώπιση αυτού είναι η χρήση tracking, δηλαδή απόδοση χαρακτηριστικών (ID) σε κάθε άτομο της εικόνας και διατήρηση του χαρακτηριστικού του καθ'ολη τη διάρκεια του βιντεο. Έτσι κρατάμε σε κάθε frame τη λίστα των IDs που είναι ενεργά (εμφανίζονται οι αντίστοιχοι άνθρωποι στο βίντεο), παράλληλα γίνεται αναγνώριση μασκών και για κάθε ID για το οποίο γίνεται αναγνώριση μάσκας (η μη μάσκας) γίνεται ανανέωση ενός map απο αναγνωριστικά σε ύπαρξη η μή μάσκας. Αυτό έχει ώς αποτέλεσμα αν δεν γίνει στο εκάστοτε frame αναγνώριση μάσκας η μη για κάποιο χαρακτηρηστικό, θεωρείται ότι φορά η δεν φορά μάσκα ανάλογα με την τελευταία φορα που παρατηρήθηκε με μασκα η χωρίς μάσκα. Με τον τρόπο αυτό μειώνονται σημαντικά οι αυξομειώσεις και καταλήγουμε σε μεγαλύτερης ακρίβειας αποτελέσματα.

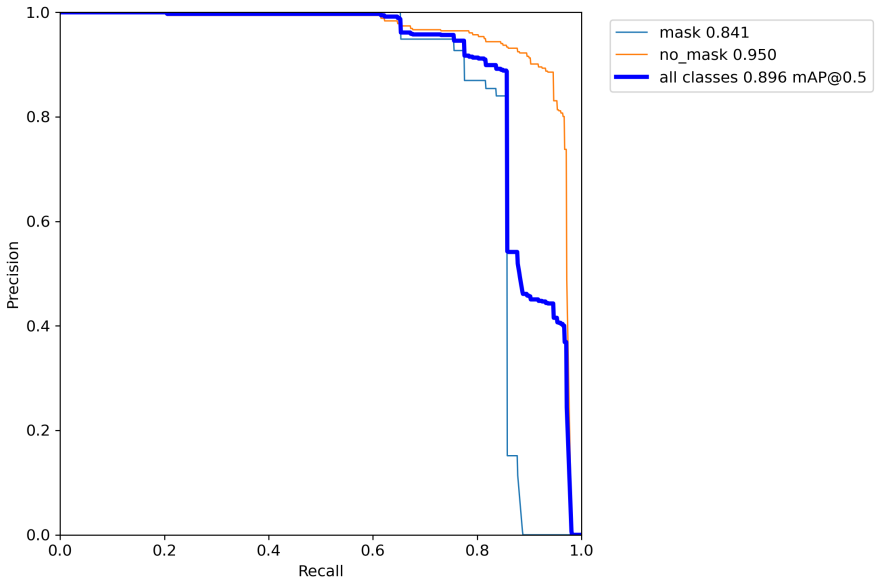
Για την επίλυση του Τρασκινγ χρησιμοποιήθηκε το μοντέλο Deep SORT το οποίο περιγράφηκε στο κεφάλαιο 4. Το τελικό μοντέλο αποτελείται απο 4 βασικές διαδικασίες που γίνονται σε κάθε frame:

1. Ανίχνευση ανθρώπων στο frame
2. Απόδοση αναγνωριστικών με βάση τις προβλέψεις του προηγούμενου frame
3. Πρόβλεψη θέσης ανθρώπων στο επόμενο frame με χρήση του Kalman Filter
4. Αναγνώριση προσώπων με και χωρις μασκα και αντιστοίχιση τους με τα αναγνωριστικά ανανεώνοντας το map των μασκών

Στο σχήμα 7.5 φαίνεται ένα instance του μοντέλου. Πάνω αριστερά σε κάθε περικλειόμενο κουτί ανθρώπου εμφανίζεται το αναγνωριστικό του. Παρατηρούμε ότι ενώ στο frame υπάρχουν 3 αναγνωρισμένες μάσκες η μέτρηση δίνει 4 διότι ο άνθρωπος αριστερά του έχει ήδη αποδοθεί μάσκα οπότε προσμετρείται ακόμη και αν δεν γίνεται αναγνώριση στο συγκεκριμένο frame, αναδεικνύοντας τη σημαντικότητα του tracking στο πρόβλημά μας. Αξίζει να σημειωθεί ότι το μοντέλο YOLOv8 για την αναγνώριση προσώπου με μάσκα ταυτίζεται με αυτό που αναφέρθηκε στην προηγούμενη ενότητα.

Σχήμα 7.5: DeepSORT + YOLOv8 υλοποίηση

Το τελικό μοντέλο μπορεί να τρέξει σε ενα μέσο ρυθμο 20fps σε ενα custom dataset με 150 βιντεο μικρής διάρκειας 1-5 λεπτών, το οποίο δεν είναι ιδανικό για real-time εφαρμογή. Η μετρική που χρησιμοποιήθηκε είναι το Mean Average Precision (mAP) για confidence level = 0.5 (Intersection over Union Score) που όπως φαίνεται και στην καμπύλη Precision - Recall στο σχήμα 7.6 έχουμε mAP = 0.896



Σχήμα 7.6: Precision - Recall γραφική απεικόνιση για το μοντελο DeeepSort YOLO

Κεφάλαιο 8

Υλοποίηση της μεταφορας της πληροφορίας

8.1 Εισαγωγή

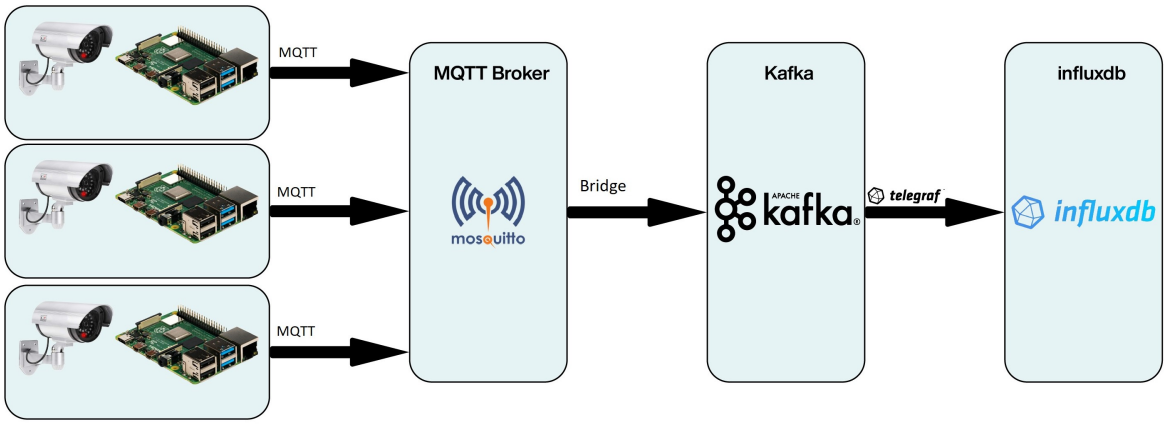
Σε αυτή την ενότητα θα δούμε πως η πληροφορία του αριθμού των ατόμων που φοράνε και δεν φοράνε μάσκα στο πλάνο μεταφέρεται έμπιστα και άμεσα απο τις κάμερες στη βάση δεδομένων απο όπου μπορούν να διαχειριστούν να μελετηθούν και ίσως να ληφθούν μέτρα αν η κατάσταση θεωρηθεί ως επισφαλής. Η μεταφορά της πληροφορίας απο τους σένσορες στη βάση αποτέλεί πρόβλημα προς επίλυση σχεδόν σε κάθε εφαρμογή IoT. Στα πλαίσια της εργασίας λήφθηκε υπόψιν η απαίτηση κλιμακωσης της εφαρμογής σε ολόκληρο σύστημα καμερών, οπότε δόθηκε προσοχή πέρα απο την εγκυρότητα της πληροφορίας, η ταχύτητα που θα διατηρείται ζωντανού χρόνο και σε πιθανή κλιμάκωση της εφαρμογής. Η διαδικασία που ακολουθήθηκε με συνοπτικά βήματα (τα οποία θα αναλυθούν περεταίρω στις επόμενες υποενότητες) είναι τα εξής:

- Μεταφορά απο κάμερα σε MQTT Broker (Eclipse Mosquitto) μέσω πρωτοκόλλου MQTT
- Μεταφορά πληροφορίας απο τον Mosquitto στον Apache Kafka μέσω γέφυρας ζωντανού χρόνου.
- Τελική μεταφορά απο Apache Kafka στην βάση δεδοένων (InfluxDB) Μέσω του εργαλείου Telegraf

Η παραπάνω ροή της πληροφορίας συνοψίζεται στο διάγραμμα 8.1:

8.2 MQTT

Για το πρώτο στάδιο της μεταφοράς της πληροφορίας χρησιμοποιήθηκε το πρωτόκολλο MQTT. Η επιλογή δεν έχγινε τυχαία, όπως αναλύθηκε στο Κεφάλαιο 3 το MQTT αποτελεί



Σχήμα 8.1: Αρχιτεκτονική συστήματος

το ιδανικό πρωτόκολλο για συστήματα IoT γιατι μπορεί να διαχειριστεί άμεσα μεγάλο όγκο δεδομένων απο πολλούς αισθητήρες. Ο MQTT Broker που χρησιμοποιήθηκε στα πλαίσια της εργασίας ήταν ο Mosquitto της Eclipse, ένας απο τους πιο γνωστούς και αξιόπιστους.

Σχήμα 8.2: Mosquitto Logo

Η ονοματοδοσία των topics έγινε ως εξής: κάθε κάμερα με αναγνωριστικό αριθμό k γράφει στο κανάλι masks/camerak. Πρακτικά αυτό σημαίνει ότι κάθε κάμερα αντιστοιχεί σε διαφορετικό κανάλι με αποτέλεσμα να μην υπάρχει σύγχυση ως προς τη χρηση του καναλιού ταυτόχρονα απο πολλές κάμερες. Χρησιμοποιήθηκε QoS(Quality of Service)=1 που απαιτέι την λήψη απάντησης οτι το μήνυμα ελήφθη, ως επιπλέον μηχανισμός για τη διασφάλιση της διατήρησης της πληροφορίας.
Παρακάτω βλέπουμε αποτυπώσεις της οθόνης με την επικοινωνία ενός subscriber σε ένα template κανάλι και του αντίστοιχου publisher μέσω του Mosquitto:

Πλέον λοιπόν έχει εισαχθεί η πληροφορία στον MQTT Broker απο όπου θα συνεχίσει το

Σχήμα 8.3: MQTT μήνυμα απο Raspberry pi στον Mosquitto

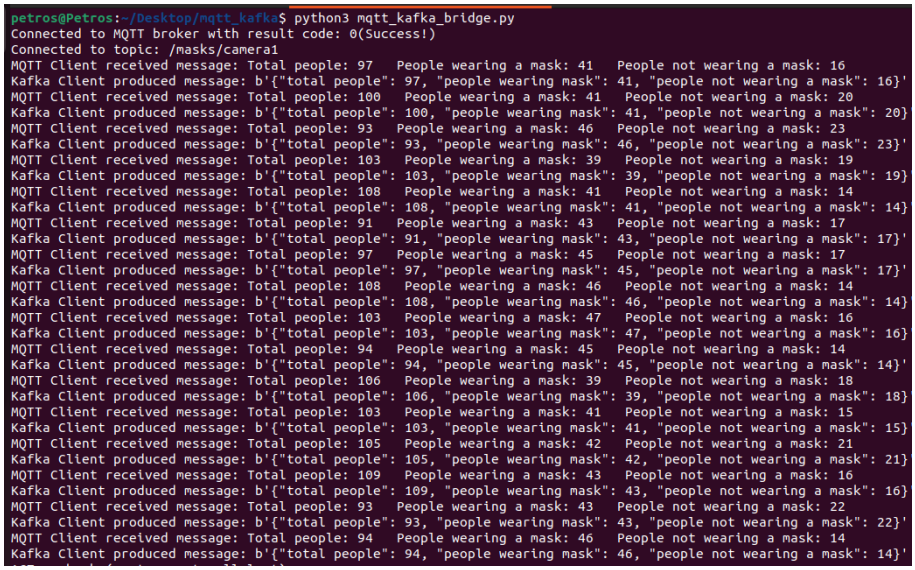
ταξίδι της προς την βάση δεδομένων.

8.3 Apache Kafka

Το επόμενο κομματι της μεταφοράς της πληροφορίας αφορά στον Apache Kafka. Όπως εξηγήθηκε στο Κεφάλαιο 5 ο Apache Kafka είναι μια πλατφόρμα λογισμικού για επεξεργασία ροών δεδομένων ο οποίος χρησιμοποιείται για την σιγουρη και ασφαλή μεταφορά ενος μεγάλου όγκου δεδομένων. Στην παρούσα εφαρμογή χρησιμοποιήθηκε ως ενδιάμεσο στρωμα αποθήκευσης της πληροφορίας ανάμεσα στον Mosquitto (MQTT Broker) και τη βάση δεδομένων. Η χρήση αυτού ως ενδιάμεσο στρώμα λειτουργεί με σκοπό την ασφάλεια μεταφοράς της πληροφορίας καθώς σε αντίθεση με τον MQTT Broker ο Apache Kafka προσφέρει ασφαλή αποθήκευση της πληροφορίας μέσω των partition που διαθέτει και σίγουρη αποστολή της στον επόμενο σταθμό που είναι η βάση δεδομένων. Επιπρόσθετα σε περίπτωση χρήσης για πολλούς χώρους με πολλές κάμερες το scalbility της εφαρμογής είναι άμεσο, συγκεκριμένα χρησιμοποιώντας έναν MQTT Broker για κάθε χώρο και αποστολή των πληροφοριών απο κάθε Broker σε έναν κοινό Apache Kafka για τη ταυτόχρονη αποθήκευση και διαχείρισή τους, αφού έχει πολύ υψηλή αντοχή στην real-time ταυτόχρονη επικοινωνία μεγάλου όγκου δεδομένων.
Η μεταφορά της πληροφορίας απο τον broker στον Kafka γίνεται μέσω μιας γέφυρας που σχεδιάστηκε στα πλαίσια της εργασίας. Αυτή η ¨γεφυρα' ουσιαστικά αποτελείται απο μια λούπα που διαθέτει έναν MQTT subscriber και έναν Kafka producer. Η γέφυρα ουσιαστικά αναμένει το εκάστοτε topic εως ότου ληφθέι κάποιο μήνυμα απο μια κάμερα στον MQTT subscriber και όταν γίνει αυτό ο Kafka producer το δημοσιεύει στο αντίστοιχο (και συνονόματο) topic του Kafka. Ένα στιγμιότυπο χρήσης της γέφυρας φαίνεται παρακάτω στο Σχημα 8.3:

Όπως παρατηρούμε και στην εικόνα η γέφυρα, πέρα απο την μεταφορά της πληροφορίας κάνει και την απαραίτητη επεξεργασία και μετατρέπει το εισερχόμενο μήνυμα απο τύπο *String* σε *Json* με σκοπό την εύκολη αναγνώρισή του απο το Telegraf και την εισαγωγή του στη βάση.

Αξίζει να σημειωθεί ότι στην περίπτωση μιας μεγάλης εφαρμογής η γέφυρα δεν θα επαρκούσε αφου θα δημιουργούσε καθυστερήσεις λόγω του όγκου των δεδομένων. Σε εκείνη την περίπτωση θα ήταν ιδανική η χρήση του Kafka Connect , το οποίο είδαμε ήδη στο κεφάλαιο



Σχήμα 8.4: Λειτουργία γέφυρας

5, για την σύνδεση πολλών MQTT Broker στον Kafka. Το εργαλείο Kafka Connect εξειδικεύεται σε αυτήν ακριβώς την διαδικασία, την μεταφορά δηλαδή με ασφάλεια και συνέπεια μεγάλου όγκου πληροφορίας απο διάφορες πηγές (MQTT Brokers στην περιπτωσή μας) στον Kafka.
Μετά απο τη γέφυρα τα δεδομενα βρίσκονται πλέον αποθηκευμένα στα αντίστοιχα topic του Kafka και απο εκεί θα φτάσουν εν τέλει στη βάση δεδομένων.

8.4 Telegraf και InfluxDB

Τα δεδομένα βρίσκονται στον Apache Kafka και απο εκεί θα πρέπει να μεταφερθούν στη βάση δεδομένων. Το εργαλείο που θα πραγματοποιήσει τη συγκεκριμένη διαδικασία είναι το Telegraf που περιγράφηκε στο κεφάλαιο 6. Το εργαλείο αυτό είναι υπεύθυνο για τη μεταφορά δεδομένων από μια μεγάλη ποικιλία εισόδων σε μια ποικιλία εξόδων που συνήθως είναι η InfluxDB. Για τον προσδιορισμό της προέλευσης και της κατεύθυνσης των δεδομένων καθώς και του τύπου που αναμένεται να είναι αυτά το Telegraf έχει ενα configuration αρχείο. Στην περιπτωσή μας το configuration αρχείο που χρησιμοποιήθηκε ήταν το εξής:

Με αυτήν την τελευταία μεταφορά δεδομένων λοιπόν τα δεδομένα μας βρίσκονται στη βάση δεδομένων. Χωρίζονται σε 3 πεδία ανάλογα τη μέτρηση, συγκεκριμένα συνολικοί άνθρωποι τη δεδομένη χρονική στιγμή, αριθμός ατόμων που επιβεβαιωμένα φορούν μάσκα και αριθμός ατόμων που επιβεβαιωμένα δεν φορούν μασκα όπως φαίνεται και στην παρακάτω εικόνα:

Αυτό λοιπόν αποτελεί και την τελευταία μεταφορά της πληροφορίας αφου τα δεδομένα καταλήγουν στην Time series database (InfluxDB) απο όπου μπορεί να γίνει η επεργασία

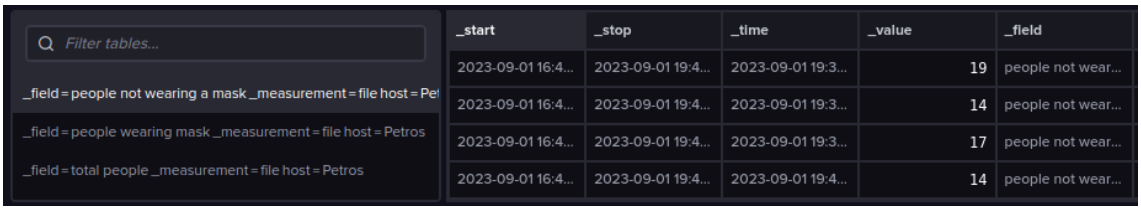
```
# Configuration for telegraf agent
[agent]
  interval = "10s".
  round_interval = true
  metric_buffer_limit = 10000
  collection_jitter = "0s"
  flush_interval = "10s"
  flush_jitter = "0s"
  precision = ""
  hostname = ""
  omit_hostname = false

[[outputs.influxdb_v2]]
  urls = ["http://localhost:8086"]
  token = "$INFLUX_TOKEN"
  organization = "diplwmatiki_org"
  bucket = "Cameras"

[[inputs.kafka_consumer]]
  ## Kafka brokers.
  brokers = ["localhost:9092"]
  topics = ["camera1"]
  max_message_len = 1000000
  data_format = "json"
  tag_keys = ["camera","total people","people wearing mask"]

  [[outputs.file]]
  ## Files to write to, "stdout" or "stderr".
  files = ["stdout"]
  data_format = "json"
```

Σχήμα 8.5: Telegraf configuration file



Σχήμα 8.6: Δεδομένα στην InfluxDB

του, όπως η αναζήτηση μη τήρησης μέτρων ασφαλείας στο χώρο ανάλογα με τις τιμές των μεταβλητών μπορούν να οριστούν αντίστοιχες συναρτήσεις που καθιστούν τον έλεγχο της ασφαάλειας. Τέλος μπορούν επίσης να τρέξουν στα ιστορικά δεδομένα έλεγχοι που πιθανώς κρίνουν τις ώρες που προκύπτει η μεγαλύτερη επισφάλεια του χώρου ή και αθροιστικές συναρτήσεις απο διαφορετικές (μη επικαλυπτόμενες) κάμερες, βγάζοντας πορίσματα για ολόκληρη την εγκαταάσταση. Σε κάθε περίπτωση το ταξίδι των δεδομένων που ξεκίνησε απο κάθε κάμερα και το Rasberry pi που είναι συνδεδεμένο με αυτην έχει ολοκληρωθει και τα δεδομένα βρίσκονται πλέον ασφαλή για οποιαδήποτε πιθανή χρήση στην βάση δεδομένων!

Κεφάλαιο 9

Σύνοψη, συμπεράσματα και Future Work

Απο τις διερευνήσεις της παραπάνω εργασίας σμπαιραίνουμε ότι ένα σύστημα παρακολούθησης και ανίχνευσης μασκών με τη χρήση κάμερας, Raspberry Pi, Mosquitto για την μεταφορά της πληροφορίας απο το Raspberry Pi με MQTT, Apache Kafka και telgraf, InfluxDB καλύπτει πλήρως τις προδιαγραφές του συστήματος ζωντανού χρόνου που θέλαμε να κατασκευάσουμε. Η βελτίωση του τρόπου εύρεσης των δεδομένων βελτιώνεται με τη χρήση tracking στο μοντελο οδηγώντας σε ακριβέστερα αποτελέσματα, παρόλα αυτά υπάρχει το trade-off όπου το μοντέλο πέφτει στα 20fps το οποίο δεν είναι ιδανικό για ζωντανού χρόνου σύστημα.

Σαν Future Work για βελτίωση των υφιστάμενων δυνατοτήτων του συστήματος έχουμε τις εξής προτάσεις:

1. Αντικατάσταση της γέφυρας μεταξύ MQTT broker και Kafka με το Kafka connect που δημιουργεί μεγαλύτερη ασφάλεια των δεδομένων.

2. Βελτίωση του αλγορίθμου tracking. Αντί για το DeepSort μπορεί να χρησιμοποιηθεί κάποιος αλγόριθμος λίγο ελαφρύτερος ώστε να φτάσει το σύστημα στα 30fps όπου θα είναι ιδανικό για σύστημα ζωντανού χρόνου.

3. Το μοντέλο Lightrack μπορεί να χρησιμοποιηθεί ως εναλλακτική, λόγω της καλύτερης ακρίβειας που προσφέρει σαν αλγόριθμος tracking, αν ξεπεραστεί η πολύ χαμηλή ταχύτητα. Σε δοκιμές στα πλαίσια της εργασίας το αποτέλεσμα ήταν να τρέχει το σύστημα σε μόλις 1 fps.

Βιβλιογραφία

[1] Apache Kafka. https://el.wikipedia.org/wiki/Apache_Kafka.

[2] Apache Kafka Documentation. https://kafka.apache.org/documentation/#connect.

[3] Characteristics of Time-Series Data. https://tdengine.com/tsdb/characteristics-of-time-series-data/.

[4] InfluxDB. https://en.wikipedia.org/wiki/InfluxDB.

[5] InfluxDB: Time series database Technical Paper. https://www.influxdata.com/time-series-database/.

[6] MQTT guide. https://www.catchpoint.com/network-admin-guide/mqtt-broker.

[7] Τι είναι το MQTT και γιατί το χρειαζόμαστε. https://cy.ipc2u.com/articles/articles-and-reviews/ti-einai-to-mqtt-kai-giati-to-chreiazomaste-sto-iiot/, 2017.

[8] Nga Tran, Paul Dix, Andrew Lamb, Marko Mikulicic . InfluxDB 3.0: System Architecture. https://www.influxdata.com/blog/influxdb-3-0-system-architecture/, 2023.

[9] Guanghan Ning , Heng Huang. LightTrack: A Generic Framework for Online Top-Down Human Pose Tracking. https://arxiv.org/pdf/1905.02822.pdf, 2019.

[10] IBM. What is internet of things? https://www.ibm.com/topics/internet-of-things.

[11] IBM. What are convolutional neural networks? https://www.ibm.com/topics/convolutional-neural-networks, 2017.

[12] IBM. What is a neural network? https://www.ibm.com/topics/neural-networks, 2018.

[13] Ignacio de Mendizábal. IoT Communication Protocols—IoT Data Protocols. https://www.allaboutcircuits.com/technical-articles/internet-of-things-communication-protocols-iot-data-protocols/, 2022.

[14] Jerry Franklin. Apache Kafka Architecture: What You Need to Know. https://www.upsolver.com/blog/apache-kafka-architecture-what-you-need-to-know, 2022.

[15] Joseph Redmon,Santosh Divvala, Ross Girshick, Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection. https://arxiv.org/pdf/1506.02640.pdf, 2017.

[16] Karl Weiss* , Taghi M. Khoshgoftaar and DingDing Wang. A survey of transfer learning. Weiss et al. J Big Data, 2016.

[17] Pallavi Sethi and Smruti R. Sarangi. Internet of Things: Architectures, Protocols, and Applications. https://downloads.hindawi.com/journals/jece/2017/9324035.pdf?_gl=1*106e464*_ga*MjA4Njg3NzE1MC4xNjg4ODAwNjc4*_ga_NF5QFMJT5V*MTY4ODgwMDY3Ny4xLjAuMTY4ODgwMDY3Ny42MC4wLjA.&_ga=2.77459602.621472554.1688800678-2086877150.1688800678, 2017.

[18] ultralytics. Ultralytics YOLOv8 Github. https://github.com/ultralytics/ultralytics#documentation, 2023.

[19] Wikipedia. Machine Learning General Facts. https://en.wikipedia.org/wiki/Machine_learning.