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

Επεξεργασία δεδομένων για έξυπνες αγροκαλλιέργειες

Διπλωματικη Εργασια

του

ΙΩΑΝΝΗ ΒΑΛΙΑΔΗ

Επιβλέπων: Βασιλική Καντερέ
Επίκουρη Καθηγήτρια Ε.Μ.Π.

Αθήνα, Οκτώβριος 2022

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

Επεξεργασία δεδομένων για έξυπνες αγροκαλλιέργειες

Διπλωματικη Εργασια

του

ΙΩΑΝΝΗ ΒΑΛΙΑΔΗ

Επιβλέπων: Βασιλική Καντερέ
Επ. Καθηγήτρια Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 17η Οκτωβρίου 2022

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Βασιλική Καντερέ	Νεκτάριος Κοζύρης	Συμεών Παπαβασιλείου
Επ. Καθηγήτρια Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.

Αθήνα, Οκτώβριος 2022

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

(Υπογραφή)

.....

Ιωαννης Βαλιαδης
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Η συλλογή, επεξεργασία και αποθήκευση δεδομένων αισθητήρων σε πραγματικό χρόνο και η αυτοματοποιημένη λήψη αποφάσεων μέσω μιας ενιαίας αρχιτεκτονικής αποτελεί επιτακτική ανάγκη πλέον στην σύχρονη κοινωνία. Η λήψη δεδομένων από γεωργικές καλλιέργειες, όπως η θερμοκρασία, η υγρασία και άλλες χρήσιμες πληροφορίες σχετικά με την υγεία των φυτών, επιτυγχάνεται μέσω εφαρμογών IoT (Internet of Things). Οι αισθητήρες και οι διασυνδεδεμένες συσκευές μαζεύουν μεγάλες ποσότητες δεδομένων, τα οποία στέλνονται ταυτόχρονα σε ένα σύστημα ανταλλαγής μηνυμάτων που βασίζεται στην τεχνολογία publish-subscribe χρησιμοποιώντας ελαφριά και γρήγορα πρωτόκολλα MQTT. Ένα τέτοιο σύστημα είναι το Apache Kafka, το οποίο εγγυάται ελάχιστες απώλειες και μικρές καθυστερήσεις κατά την διαχείρηση μεγάλου όγκου δεδομένων. Τα δεδομένα αποθηκεύονται στη συνέχεια σε μια βάση δεδομένων χρονοσειρών που ονομάζεται InfluxDB. Χρησιμοποιώντας το εργαλείο Grafana, μπορούμε να οπτικοποιήσουμε τα δεδομένα και να χρησιμοποιήσουμε μοντέλα μηχανικής μάθησης για τη λήψη αποφάσεων μεγάλης σημασίας για τη γεωργία.

Λέξεις κλειδιά:
Internet of Things, MQTT, Apache Kafka, InfluxDB, Machine learning,Grafana

Abstract

The collection, processing, storage of real-time sensor data and automated decision making through a common architecture is a growing need worldwide. The extraction of data from agricultural crops, such as temperature, humidity and other useful information about plant health, is achieved through IoT (Internet of Things) applications. Interconnected devices and sensors collect large volumes of data in real-time, which are simultaneously sent using fast and lightweight MQTT protocols to a messaging system based on the publish-subscribe technique. One such system is Apache Kafka which guarantees short delays and minimal losses while handling large volumes of data. The data is then stored in a time series database the InfluxDB. By using Grafana we are provided with the visualization of the data and the ability to use machine learning models for making decisions of major importance for agro-culture.

Keywords:
Internet of Things, MQTT, Apache Kafka, InfluxDB, Machine learning, Grafana

Ευχαριστίες

Με την ολοκλήρωση της διπλωματικής μου εργασίας θα ήθελα να ευχαριστήσω ξεχωριστά όλους όσους συνέβαλαν στην εκπόνησή της. Αρχικά θα ήθελα να ευχαριστήσω θερμά την καθηγήτρια **κυρία Καντερε**, η οποία με εμπιστεύτηκε και μου έδωσε την ευκαιρία να ασχοληθώ με ένα τόσο ενδιαφέρον θέμα. Έμαθα πολλά και χρήσιμα πράγματα και χάρις την καθοδήγηση και την επίβλεψή της κατάφερα να ολοκληρώσω την διπλωματική μου. Θερμές ευχαριστίες θα ήθελα να απευθύνω και στον κύριο **Παρασκευά Κερασιώτη**, που επίσης με καθοδηγούσε και με συμβούλευε σε κάθε μου δυσκολία. Τέλος, θα ήθελα να ευχαριστήσω την οικογένεια μου και τους φίλους μου που με στήριζαν καθ΄όλη την διάρκεια των σπουδών μου και ιδιαίτερα κατα την διάρκεια δημιουργίας της συγκεκριμένης εργασίας. Νιώθω ευγνώμων για όλους.Σας ευχαριστώ και πάλι.

Περιεχόμενα

Κατάλογος Σχημάτων

Κεφάλαιο 1

Εισαγωγή

1.1 Στόχος της εργασίας

Στόχος της παρούσας εργασίας είναι η ανάπτυξη μιας αρχιτεκτονικής για την εξαγωγή, επεξεργασία, αποθήκευση και οπτικοποίηση (visualization) δεδομένων από αισθητήρες.

Ειδικότερα μελετάται η διαδρομή των δεδομένων και των μετρήσεων, από την απόκτηση τους από τους αισθητήρες στις εφαρμογές IoT (Internet of Things), η διάδοσή τους μέσω πρωτοκόλλων ανταλλαγής μηνυμάτων και η παράλληλη επεξεργασία τους από εφαρμογές προκειμένου να ληφθούν συγκεκριμένες αποφάσεις με την χρήση μοντέλων μηχανικής μάθησης. Εκείνη, εν ολίγοις, είναι η διαδρομή που ακολουθεί ένα τεράστιο μέγεθος πραγματικών δεδομένων, με απώτερο στόχο τον έλεγχο και τη διαχείριση των γεωργικών καλλιεργειών.

Η παρούσα εργασία ξεκινά με μια περιγραφή της εξέλιξης του IoT, των πρωτοκόλλων και της αρχιτεκτονικής που χρησιμοποιήται, συνεχίζοντας με τις εφαρμογές του IoT που χρησιμοποιούνται σήμερα. Έπειτα, εξετάζει ευέλικτα εργαλεία και πρωτόκολλα ανταλλαγής μηνυμάτων όπως το MQTT και συστήματα προσωρινής αποθήκευσης όπως το Apache Kafka. Στην συνέχεια δίνεται έμφαση στα εργαλεία που χρησιμοποιούνται για την μεταφόρα των δεδομένων, την αποθήκευση τους καθώς και την οπτικοποιήση τους. Πιο συγκεκριμένα, την μεταφόρα αύτη των δεδόμενων από τον Apache Kafka προς την βάση δεδομένων χρονοσειρών την αναλαμβάνει το Telegraf της Influx Data. Το Telgraf μεταφέρει τα δεδομένα στην βάση δεδομένων χρονοσείρων (Timeseries Database) σε ένα άλλο προιόν της Influx Data την InfluxDB. Τέλος χρησιμοποιήται ένα άλλο εργαλείο της Influx Data για την οπτικοποιήση των δεδομένων το Grafana.

1.2 Γενικό πλαίσιο

Για να επεξεργαστούμε και να απεικονίσουμε τα δεδομένα αισθητήρων, πρέπει πρώτα να βρούμε έναν τρόπο για την αποτελεσματική μετάδοση και αποθήκευση των δεδομένων. Τα δεδομένα μας συλλέγονται από το IoT και τους αισθητήρες, οπότε η συλλογή των πληροφοριών και η μετάδοσή τους πρέπει να είναι γρήγορη και αξιόπιστη, ώστε να αποφεύγονται καθυστερήσεις και απώλειες. Είναι επίσης απαραίτητο να επιλέξουμε έναν τρόπο μετάδοσης

που δεν απαιτεί μεγάλους αποθηκευτικούς πόρους και υψηλή υπολογιστική ισχύ, δεδομένου ότι έχουμε να κάνουμε με αισθητήρες που είναι μικροί και έχουν περιορισμένη υπολογιστική ισχύ. Για την προσωρινή αποθήκευση, χρειαζόμαστε ένα γρήγορο και αξιόπιστο σύστημα που να μπορεί να επεξεργαστεί χιλιάδες δεδομένα σε σύντομο χρονικό διάστημα, ενώ παράλληλα να είναι εξαιρετικά επεκτάσιμο σε περίπτωση που πρέπει να προστεθούν νέοι αισθητήρες στο σύστημα.Για αυτές τις προδιαγραφές, εξετάζουμε το πρωτόκολλο MQTT α τη μετάδοση δεδομένων από τους αισθητήρες και τα μοντέλα publish/subscribe, όπως ο Kafka Server, για την προσωρινή αποθήκευση δεδομένων.

Μόλις εξασφαλιστεί η μετάδοση και η προσωρινή αποθήκευση των δεδομένων, προχωράμε στην επεξεργασία των δεδομένων. Δεδομένου ότι έχουμε να κάνουμε με ένα σενάριο που λαμβάνει χώρα σε πραγματικό χρόνο, η χρονική εξάρτηση και ο χρονισμός πρέπει να είναι καλά καθορισμένα. Για αύτο το λόγο, γίνεται μέλετη των προιόντων της Influx Data που έχουν δημιουργήθει για την ταχέα, αποδοτική και αξιόπιστη λειτουργία του όλου συστήματος σε δεδόμενα που μείζοντα σημασία είναι ο χρόνος.

Τέλος, θέλουμε να επεξεργαστούμε αυτά τα δεδομένα που δημιουργήθηκαν και αποθηκεύτηκαν προκειμένου να λάβουμε. Δηλαδή, το όλο σύστημα θα πρέπει να είναι σε θέση να εκτελεί ερωτήματα στη βάση δεδομένων και να χρησιμοποιεί μοντέλα μηχανικής μάθησης (Machine Learning ML) να δίνονται κάποιες οδηγίες στον γεωργό που διαχειρίζεται την αγροκαλλιέργεια. Αύτα τα μόντελα θα δίνουν την πιο ορθή απόφαση στον γεώργο που θα έχει πάρθει με βάση κάποιον καίριων πληροφορίων όπως τα συστατικά του χώματος, υγρασία, είκονων των φυλλών και άλλων πολλών.

1.3 Περιγραφή του προβλήματος

Στα συστήματα που διαχειρίζονται δεδομένα σε πραγματικό χρόνο καλούμαστε να λύσουμε τρία βασικά προβλήματα. Πρώτον, το σύστημά μας πρέπει να μπορεί να επεξεργάζεται μεγάλο πλήθος πληροφοριών και να είναι γρήγορο, δηλαδή να έχει μικρές καθυστερήσεις, ώστε τα αποτελέσματα που παίρνουμε στην έξοδο να ανταποκρίνονται στην τρέχουσα κατάσταση του συστήματος και να παίρνουμε μετρήσεις και αποτελέσματα που αφορούν το παρελθόν και το μέλλον. Πρέπει επίσης να βεβαιωθούμε ότι οι είσοδοι δεν είναι προκατειλημμένες, καθώς αυτό μπορεί να επηρεάσει τα αποτελέσματα των αποφάσεων που μεταφέρονται στον αγρότη και να δώσει μια λανθασμένη εικόνα του συστήματός μας. Τέλος, όπως συμβαίνει με όλα τα συστήματα, θέλουμε τα δεδομένα μας,τόσο τα δεδομένα που λαμβάνουμε στην είσοδο με τη μορφή ενδείξεων αισθητήρων όσο και τα δεδομένα που παράγονται από τους μετασχηματισμούς του συστήματος, να είναι ασφαλή, ώστε να μην χάνονται σε περίπτωση αποτυχίας οποιουδήποτε στοιχείου του συστήματός μας.

1.4 Αντιμετώπιση του προβλήματος

Στην παρούσα εργασία παρουσιάζουμε μια αρχιτεκτονική που αντιμετωπίζει τα παραπάνω προβλήματα και υλοποιεί ένα ολοκληρωμένο σύστημα για την εξαγωγή, επεξεργασία, απο-

θήκευση και οπτικοποίηση δεδομένων αισθητήρων σε πραγματικό χρόνο και τη λήψη αποφάσεων με βάση αυτά τα δεδομένα.

Όσο αφορά την ταχύτητα και τον όγκο δεδομένων χρησιμοποιήθηκαν πρωτόκολλα και εργαλεία που χαρακτηρίζονται από την ταχύτητα και την ανθεκτικότητά τους σε μεγάλους όγκους δεδομένων. Πιο συγκεκριμένα, για την μετάδοση της πληροφορίας των αισθητήρων χρησιμοποιήθηκε το πρωτόκολλο MQTT το οποίο είναι ιδιαίτερα ταχύ. Επιπλέον, για την ενδιάμεση αποθήκευση χρησιμοποιήσαμε Apache Kafka αντί για παραδοσιακές βάσεις δεδομένων, εργαλείο τύπου publish/subscribe επεξεργάζεται μεγάλους όγκους δεδομένων με μεγάλη ταχύτητα. Στην συνέχεια, τα δεδομένα τα όποια είναι εικόνας αποθηκεύονται με την χρήση Hadoop Distributed File System (HDFS) για την ασφαλή και γρήγορη ανάκτηση τους αλλά και για την ανοχή σε σφάλματα διότι μιλάμε για συστήματα χαμηλής απόδοσης κα αξιοπιστίας.

Έπειτα με την βοήθεια του Telegraf τα δεδομένα μεταφέρονται στην βάση δεδομένων χρονοσειρών για την μόνιμη αποθήκευση τους στην InfluxDB. Εκεί εμφανίζεται και η χρήση του Grafana, για την οπτικοποιήση αυτών των δεδομένων καθώς και των αποφάσεων - κατευθυντήρων οδηγιών που θα παρέχει το σύστημα μας στο γεωργό, που θα έχουν παρθεί από μοντέλα μηχανικής μάθησης που έχουν αναπτυχτεί.

1.5 Δομή της εργασίας

Στο δεύτερο κεφάλαιο περιγράφεται και ορίζεται η έννοια του IoT (Internet of Things), πώς λειτουργεί και γιατί είναι σημαντικό και θα είναι μια τεχνολογία που θα ανθίσει στο μέλλον.

Η επόμενη ενότητα εξετάζει τη διαδικασία ανταλλαγής μηνυμάτων με τη χρήση του πρωτοκόλλου MQTT, τη δομή αυτών των μηνυμάτων και τα πλεονεκτήματα της μεθόδου publish/subscribe.

Στη συνέχεια αναφέρουμε τον Apache Kafka, μια διάσημη οντότητα λογισμικού για διαχείρηση ροών δεδομένων, και αναπτήσεται ο τρόπος λειτουργίας του και τα πλεονεκτήματα που παρέχει στο σύστημα.

Στο πέμπτο κεφάλαιο, παρουσιάζεται η InfluxDB, μια βάση δεδομένων χροσειρών για την μόνιμη αποθήκευση των δεδομένων μας, καθώς και τα πλεονέκτηματα χρησιμοποιήση της.

Έπειτα, παρουσιάζονται τα open source προιόντα της Influx Data, Telgraf & Grafana τα όποια εγίναν χρήση στην αρχιτέκτονικη , για την μεταφόρα δεδόμενων απο τον Kafka και για την οπτικοποιήση τους αντίστοιχα.

Στο έβδομο κεφάλαιο, αναλύθηκε το κατανεμημένο σύστημα αποθήκευσης δεδομένων Hadoop Distributed File System (HDFS) που έγινε χρήση στην αποθήκευση είκονων.

Στο επόμενο κεφάλαιο, ασχολούμαστε με την μηχανική μάθηση (Machine Learning) ένα γενικό θεωρητικό υπόβαθρο του αντικείμενου και ύστερα ειδικότερα στου αλγορίθμους στους οποίους βασίστηκε η εκπαίδευση των μοντέλων που χρησιμοποιούνται στην λήψη των αποφάσεων.

Στο προτελευταίο κεφάλαιο παρουσιάζει την τελική αρχιτεκτονική που επιλέχθηκε και αναλύει τη λειτουργικότητά της. Παρουσιάζονται επίσης αποτελέσματα προσομοίωσης, στα

οποία η παρούσα αρχιτεκτονική δοκιμάζεται σε πραγματικά σενάρια εφαρμογών.

Η παρούσα εργασία ολοκληρώνεται με τα συμπεράσματα μας.

Κεφάλαιο 2

Internet of Things

2.1 Εισαγωγή

Internet of things είναι ο όρος για ένα δίκτυο που αποτελείται από φυσικά αντικείμενα ή αντικείμενα εμφυτευμένα με αισθητήρες, λογισμικό ή γενικά ηλεκτρονικά μέρη που, σε συνδυασμό με τη δυνατότητα σύνδεσης στο Διαδίκτυο, τους επιτρέπουν να συλλέγουν και να ανταλλάσσουν δεδομένα.

Με τον όρο ¨αντικείμενα' εννοούμε μια ποικιλία συσκευών που μπορούν να χρησιμοποιηθούν για ιατρικούς σκοπούς (αισθητήρες για μέτρηση γλυκόζης αίματος, καρδιακού ρυθμού κ.λπ.), σε βιομηχανικό εξοπλισμό (έλεγχος θερμοκρασίας, υγρασίας), σε έξυπνα προϊόντα (κινητά τηλέφωνα, αυτοκίνητα, τηλεοράσεις για μεγαλύτερη ασφάλεια και εξατομικευμένη εμπειρία χρήσης). Το IOT επιτρέπει την επικοινωνία μεταξύ όλων αυτών των αντικειμένων και συσκευών, καθώς και την απομακρυσμένη διαχείριση αυτών των αντικειμένων και συσκευών μέσω ενός υπάρχοντος δικτύου υποδομής. Η δυνατότητα αυτή δημιουργεί πολλές ευκαιρίες για την ενσωμάτωση της πληροφορικής στον κόσμο μας, προσφέροντας μεγαλύτερη αποτελεσματικότητα, ακρίβεια και πολλά οικονομικά οφέλη.

2.2 Ορισμός

Το Διαδίκτυο των Πραγμάτων (IoT) είναι η σύνδεση συσκευών στο Διαδίκτυο και η χρήση αυτής της σύνδεσης για την παροχή κάποιου είδους χρήσιμης απομακρυσμένης παρακολούθησης ή ελέγχου αυτών των συσκευών. Δεν υπάρχει ενιαίος, αυστηρός ορισμός του όρου. Έχουν προταθεί διάφοροι ορισμοί του IoT, κάποιοι από τους οποίους είναι οι παρακάτω:

1. IoT είναι ένα πλαίσιο στο οποίο όλα τα πράγματα έχουν μια εκπροσώπηση και οντότητα στο Διαδίκτυο. Ειδικότερα, το IoT έχει σαν στόχο στην παροχή νέων εφαρμογών και υπηρεσιών που αποσκοπούν στη γεφύρωση του φυσικού και του εικονικού κόσμου, με τη λειτουργία M2M να αποτελεί τη γραμμή επικοινωνίας που επιτρέπει την αλληλεπίδραση μεταξύ πραγμάτων και εφαρμογών σε επίπεδο cloud. (IEEE Communications Magazine)

2. IoT: μια παγκόσμια δόμη της κοινωνίας της πληροφορίας που παρέχει προηγμένες υπηρεσίες μέσω διασυνδεδεμένων (φυσικών και εικονικών) συσκευών (¨πραγμάτων¨) με βάση τις υπάρχουσες και εξελισσόμενες διαλειτουργικές τεχνολογίες πληροφοριών και επικοινωνιών. (Διεθνής Ένωση Τηλεπικοινωνιών - ITU»

2.3 Ιστορική εξέλιξη του IoT

Η πρώτη ιδέα του IoT εμφανίστηκε πριν από σχεδόν δύο δεκαετίες, αλλά οι τεχνολογίες πίσω από υπήρχαν ήδη και βρίσκονταν υπό ανάπτυξη εδώ και πολλά χρόνια. Ας δούμε την ιστορία της εξέλιξης του IoT και των υποστηρικτικών και συναφών τεχνολογιών του σε χρονολογική σειρά:

- 1969: Internet, η κύρια τεχνολογία πίσω από το IoT εμφανίστηκε ως Advanced Research Project Agency Network (ARPANET) το οποίο χρησιμοποιήθηκε κυρίως από ακαδημαϊκή και ερευνητική αδελφότητα για την ανταλλαγή ερευνητικών εργασιών, την ανάπτυξη νέων τεχνικές διασύνδεσης και για τη σύνδεση υπολογιστών με πολλούς υπολογιστές γενικού σκοπού. Υπολογιστικά κέντρα του υπουργείου Άμυνας των Ηνωμένων Πολιτειών, καθώς και σε δημόσιους και ιδιωτικους τομείς.

- 1973: Μια άλλη βασική τεχνολογία για το IoT είναι το RFID (Radio-Frequency Αναγνώριση μέσω ραδιοσυχνοτήτων). Αν και οι ρίζες της RFID μπορούν να εντοπιστούν πίσω στον Β΄ Παγκόσμιο Πόλεμο. και οι εξελίξεις συνεχίστηκαν κατά τη δεκαετία του 1950 και του 1960, αλλά το πρώτο δίπλωμα ευρεσιτεχνίας των ΗΠΑ για ετικέτα RFID με επανεγγράψιμη μνήμη έλαβε ο Mario W. Cardullo το 1973. Ωστόσο, ένας επιχειρηματίας με έδρα την Καλιφόρνια, ο Charles Walton, έλαβε επίσης μια πατέντα το ίδιο έτος για παθητικό αναμεταδότη που ξεκλειδώνει την πόρτα εξ αποστάσεως.

- 1974: Το ενσωματωμένο σύστημα υπολογιστών ήταν μια άλλη σημαντική τεχνολογία για το IoT. Αυτά τα συστήματα υλοποιούνται με τη χρήση υπολογιστών μιας πλακέτας και μικροελεγκτών και ενσωματώνονται στο μεγαλύτερο σύστημα για να αποτελέσουν το αναπόσπαστο μέρος του.

- 1984: Πρώιμη χρήση του IoT χωρίς να έχει βαφτιστεί. Μια μηχανή κόκα κόλας ήταν συνδεδεμένη στο διαδίκτυο για να αναφέρει τη διαθεσιμότητα και τη θερμοκρασία του ποτού.

- 1990: Διάδοση του διαδικτύου στις επιχειρήσεις και στις καταναλωτικές αγορές. Ωστόσο, η χρήση του εξακολουθούσε να είναι περιορισμένη λόγω της χαμηλής απόδοσης της συνδεσιμότητας του δικτύου.

- 1991: Η έννοια της πανταχού παρούσας πληροφορικής προτάθηκε από τον Mark Weiser. Η πανταχού παρούσα πληροφορική έκανε χρήση προηγμένων ενσωματωμένων υπολογιστών ως υπολογιστής να είναι παρών στα πάντα, αλλά αόρατος. Αργότερα, έγινε γνωστή ως πανταχού παρούσα πληροφορική.

- Μέσα της δεκαετίας του 1990: Αναπτύχθηκαν κόμβοι αισθητήρων για την ανίχνευση των δεδομένων από μοναδικά αναγνωρισμένες ενσωματωμένες συσκευές και να ανταλλάσσουν απρόσκοπτα τις πληροφορίες για την υλοποίηση και τη βασική ιδέα του IoT

- 1999: Η επικοινωνία μεταξύ συσκευών εισήχθη από τον Bill Joy στο έργο του ταξινομία του διαδικτύου και ο όρος ¨Διαδίκτυο των πραγμάτων' χρησιμοποιήθηκε για πρώτη φορά από τον Ashton. Εκτός αυτού, η τεχνολογία RFID ενισχύθηκε από την ίδρυση του Auto-ID Center στο Ινστιτούτο Τεχνολογίας της Μασαχουσέτης (MIT) για την παραγωγή ενός φθηνού τσιπ που μπορεί να αποθηκεύσει πληροφορίες και να μπορεί να χρησιμοποιηθεί για τη σύνδεση αντικειμένων με το διαδίκτυο.

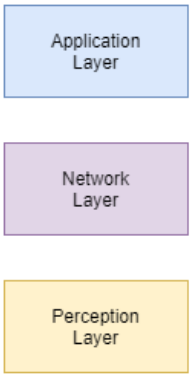
- Από το 2000 και μετά: ως αποτέλεσμα της ψηφιοποίησης, η συνδεσιμότητα στο διαδίκτυο έγινε το κανόνας για πολλές εφαρμογές και όλες οι επιχειρήσεις και τα προϊόντα αναμενόταν να έχουν παρουσία στο διαδίκτυο και να παρέχουν πληροφορίες on-line. Ωστόσο, αυτές οι συσκευές εξακολουθούν να είναι κατά κύριο λόγο πράγματα στο Διαδίκτυο που απαιτούν περισσότερο ανθρώπινη αλληλεπίδραση και παρακολούθηση μέσω εφαρμογών και διεπαφών.

Οι πραγματικές δυνατότητες του IoT έχουν μόλις αρχίσει να συνειδητοποιούνται όταν οι ανεπαίσθητες τεχνολογίες λειτουργούν παρασκηνιακά και ανταποκρίνεται δυναμικά στις προσδοκίες μας ή την ανάγκη των ¨πραγμάτων' να ενεργούν και να συμπεριφέρονται. [16]

2.4 Αρχιτεκτονική του IoT

Το διαδύκτιο των πραγμάτων δεν έχει μια μοναδική αρχιτεκτονική εξαιτίας της μεγάλης ποικιλίας των συσκευών και τον διαφορετικών αναγκών των χρηστών που έχει.
Γενική αναπαράσταση IoT με παραδείγματα [7]:
Device Layer: Συμπρεριλαμβάνει έναν ή περισσότερους αισθητήρες σε χαμηλούς σε κόστος,με μικρή κατανάλωση ενέργειας, μικρής διάστασης.
Communication Layer: Περιέχει Ρεστ πρωτόκολλα και άλλα applications level πρωτοκολλα (MQTT / HTTP)
Bus Layer: Ιδιαίτερο επίπεδο υψηλής σημασίας, περιέχει HTTP Server και συχνά MQTT Broker. Συνδυάζει και μαζεύει την πληροφορία μέσω Gateway καθώς και "αλλάζει" τα δεδομένα έτσι ώστε να μπορούν να χρησιμοποιηθούν από διαφορετικά πρωτόκολλα.
Event Processing and Analytics: Προεπεξεργασία με εφαρμογή cloud και edge computing.
Application Layer: Web based Engine (Web Portal) για επικοινωνία με άλλα API (API Management), επικοινωνεί και με συσκευές εκτός του δικτύου (Dashboard).

2.4.1 Three-layer architecture



Σχήμα 2.1: Αρχιτεκτονική 3 επιπέδων

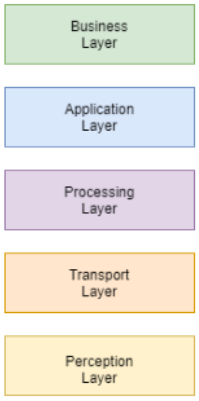
Η πιο βασική μορφή αρχιτεκτονικής είναι 3 στρωμάτων [7][17].

(1) Στρώμα εφαρμογής: Σε αυτό το στρώμα βρίσκεται το σύνολο των εφαρμογών στις οποίες χρησιμοποιείται το IoT. Είναι υπεύθυνο για την διανομή των πληροφοριών, μέσω υπηρεσιών, στους χρήστες που τις διαχειρίζονται και τις χρησιμοποιούν.

(2) Επίπεδο δικτύου: Υπεύθυνο για την σύνδεση των συσκευών που χρησιμοποιούν το IoT και γενικά των συσκευών δικτύου. Σε αυτό το επίπεδο γίνεται ακόμα, η διαβίβαση και η προεπεξεργασία των δεδομένων που συλλέγονται από το παρακάτω στρώμα.

(3) Στρώμα αντίληψης: Επίπεδο αισθητήρων: συγκεντρώνονται πληροφορίες για το περιβάλλον καθώς και τα αντικείμενα που βρίσκονται και αλληλεπιδρούν σε αυτό.

2.4.2 Five-layer architecture



Σχήμα 2.2: Αρχιτεκτονική 5 επιπέδων

Για περισσότερη λεπτομέρεια η αρχιτεκτονική του Σχήματος 2.1 μπορεί να χωριστεί σε μικρότερα κομμάτια και να οδηγήσει σε αυτή των 5 επιπέδων [7][17].

(1) Επιχειρησιακό επίπεδο: Διαχείριση του IoT οικοδομήματος από ένα σύνολο διαφορετικών οπτικών και εξασφάλιση ασφάλειας και ιδιωτικότητας στον χρήστη. Στενά συνδεδεμένο με την εμπορική χρήση του IoT ως τεχνολογία εφαρμοσμένη σε παραγωγικές μονάδες με σκοπό το κέρδος.

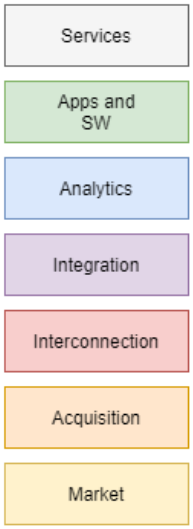
(2) Στρώμα εφαρμογής: Όπως περιγράφεται παραπάνω στην αρχιτεκτονική 3 στρωμάτων.

(3) Στρώμα επεξεργασίας: Αποθηκεύει, αναλύει και επεξεργάζεται των τεράστιο όγκο δεδομένων που παράγονται στα παρακάτω στρώματα, χρησιμοποιώντας τεχνικές όπως Big Data Analysis και cloud computing.

(4) Στρώμα μεταφοράς: Υπεύθυνο για την μεταφορά δεδομένων από τους αισθητήρες ανάμεσα στα στρώματα μέσω του δικτύου (Bluetooth, RFID, NFC,wireless, 3G, 4G, 5G).

(5) Στρώμα αντίληψης: Όπως περιγράφεται παραπάνω στην αρχιτεκτονική 3 στρωμάτων.

2.4.3 Seven-layer architecture



Σχήμα 2.3: Αρχιτεκτονική 7 επιπέδων

Είναι η πλέον σύνηθης αρχιτεκτονική για να εξηγήσει τη δομή ενός συστήματος IoT. [17].

(1) Υπηρεσίες: Υπεύθυνο για την παροχή υπηρεσιών προς το χρήστη, όπως ενεργειακή διαχείριση, εκπαίδευση, μεταφορές κ.α.

(2) Εφαρμογές και τεχνολογία SW: Για να υλοποιηθούν τέτοιες κατανεμημένες εφαρμογές απαιτείται το επίπεδο αυτό, όπως επίσης και λογισμικό όπως SDN (software defined networking), SOA (services oriented architecture) κ.α.

(3) Ανάλυση: Τα συνδυασμένα αυτά δεδομένα στη συνέχεια, αναλύονται, χρησιμοποιώντας τεχνικές μηχανικής μάθησης και εξόρυξης γνώσης από δεδομένα.

Κεφάλαιο 3

Πρωτόκολλο MQTT

3.1 Εισαγωγή

Στο διαδίκτυο οι συσκευές επικοινωνούν μεταξύ τους μέσω κάποιων πρωτοκόλλων επικοινωνίας και διεπαφών (IoT). Για να αλληλεπιδράσουν το ένα επίπεδο με το άλλο, οι συσκευές μπορούν να κάνουν χρήση διάφορων πρωτοκόλλων. Αυτό έχει σαν αποτέλεσμα, το MQTT είναι ένα από τα πιο δημοφιλή πρωτόκωλλο.

MQ Telemetry Transport ή MQTT αποτελεί ένα όχι βαρύ εννιαίο πρωτόκολλο ανταλλαγής μηνυμάτων για την μεταφορά πληροφορίας σε μακρυνές τοποθεσίες όπου χρειάζεται ΄βάρος μικρού κώδικα' ή το εύρος ζώνης του δικτύου να είναι συγκεκριμένο. Τα θετικά αυτά δίνουν την δυνατότητα την εφαρμογή αυτού του πρωτοκόλλου στα συστήματα M2M (Machine to Machine) και IoT (Internet of Things).

Σχήμα 3.1: MQTT logo

Χαρακτηριστικά του MQTT: [6]

1. Πρωτόκολλο ανταλλαγής μηνυμάτων του IoT
2. Σχεδιασμένο για έμπιστη επικοινωνία πάνω σε μη έμπιστα κανάλια
3. Publish / Subscribe model
4. Μικρή επιβάρυνση (Minimal overhead)

5. Ευχρήστο και ευκολονόητο

6. Data Agnostic: Δεν έχει γνώση ή δεν ενδιαφέρεται με ποια μέθοδο η πληροφορία που δέχεται του αποστέλλεται

Βασικά χαρακτηριστικά του πρωτοκόλλου MQTT

1. Μη σύγχρονο πρωτόκολλο

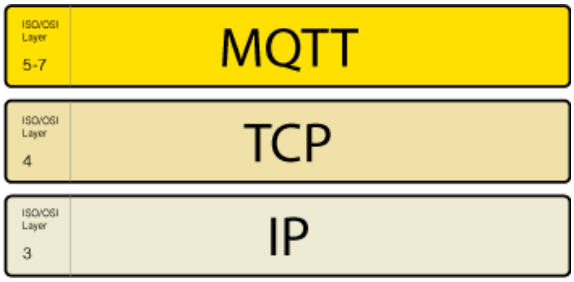
2. Εννιαία μηνύματα

3. Λειτουργία σε κατάσταση κακής σύνδεσης της γραμμής μετάδοσης των δεδομένων

4. Υποστήριξη διαφόρων επιπέδων αξιοπιστίας των μηνυμάτων (QoS)

5. Δεκτικό σε προσθήκη νεών οντοτήτων

Στο στρώμα της εφαρμογής, το πρωτόκολλο MQTT γίνεται χρήση πάνω από το πρωτόκολλο TCP / IP και αξιοποιεί την θύρα 1883 (ή 8883) ως default [4].



Σχήμα 3.2: Επίπεδο λειτουργίας MQTT

3.2 Ανταλλαγή μηνυμάτων στο MQTT

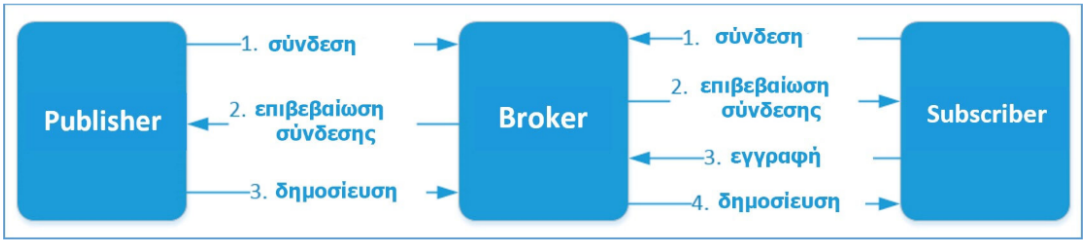
Το πρωτόκολλο MQTT, πραγματοποιεί την ανταλλαγή των μηνυμάτων ανάμεσα του Client, που μπορεί να είναι Publisher ή Subscriber της πληροφορίας και του Broker των μηνυμάτων (π.χ. Mosquitto MQTT) [4]

Ο MQTT broker είναι ένα λογισμικό το οποίο ¨τρέχει' τοπικά ή στο cloud. Ο broker παρομοιάζεται με ένα ταχυδρομείο, αφού εκτελεί την διαχείριση όλων των μηνυνάτων ανάμεσα Publishers και Subscribers. Τα Topics βοηθούν την όμαλη διεξαγωγή της διαχείρησης που καλείται να κάνει το broker. Ο Publisher στέλνει το μήνυμα στο ορισμένο Topic που θέλει και οποιοσδήποτε θέλει να δεχτεί το μηνύμα που τον αφορούν καλείται να κάνει Subscribe στο ίδιο topic. Περισότεροι απο έναν Clients έχουν την δυνατότητα να λαμβάνουν μηνύματα από έναν Publisher (one to many capability) και ένας subscriber έχει την δυνατότητα να δέχεται μηνύματα από πολλούς Publishers (many to one).

Κάθε Client μπορεί να είναι παράλληλα Publisher και Subscriber, πράγμα που διευκολύνει την επεξεργσία και τον έλεγχο των συσκευών και την ανταλαγή δεδομένων.

Οι συσκευές MQTT χρησιμοποιούν συγκεκριμένους τύπους μηνυμάτων για την διεπαφή τους με τον Broker. Εδώ είναι τα κύρια είδη:

- Σύνδεση: δημιουργία σύνδεσης με το Broker μηνυμάτων.
- Αποσύνδεση: διακοπή σύνδεσης με το Broker μηνυμάτων.
- Εγγραφή: εγγραφή σε ένα Topic στο Broker μηνυμάτων.
- Απεγγραφή: διαγραφή του Topic.
- Δημοσίευση: δημοσίευση δεδομένων σχετικά με κάποιο θέμα μέσα στο Broker μηνυμάτων.



Σχήμα 3.3: Αποστολή μηνυμάτων στο MQTT

3.3 Ποιότητα εξυπηρέτησης στο πρωτόκολλο MQTT (QoS)

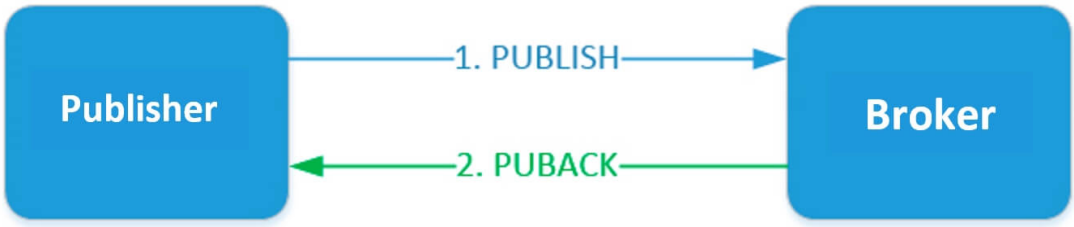
Η ποιότητα υπηρεσίας (QoS) στην ανταλλαγή μηνυμάτων MQTT είναι μια συμφωνία μεταξύ αποστολέα και παραλήπτη σχετικά με την εγγύηση της παράδοσης ενός μηνύματος.

Υπάρχουν τρία επίπεδα QoS:[4].

1. 0 - το πολύ μία φορά

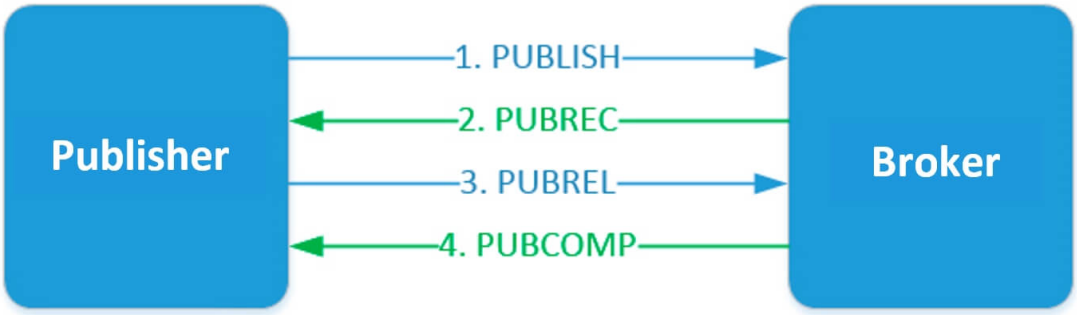
Σχήμα 3.4: MQTT με QoS = 0

2. 1 - τουλάχιστον μία φορά



Σχήμα 3.5: MQTT με QoS = 1

3. 2 - ακριβώς μία φορά



Σχήμα 3.6: MQTT με QoS = 2

Θα τα εξετάσουμε λεπτομερέστερα σε λίγο. Αλλά αξίζει πρώτα να εξηγήσουμε πώς καθορίζεται το QoS.

Όταν ένας πελάτης (π.χ. μια απομακρυσμένη συσκευή IoT) δημοσιεύει σε έναν διαμεσολαβητή, ο πελάτης καθορίζει το επίπεδο QoS για το συγκεκριμένο μήνυμα. Όταν ο broker στέλνει το μήνυμα σε ένα subscriber, χρησιμοποιείται ξανά το QoS που έχει οριστεί από τον πρώτο πελάτη για το συγκεκριμένο μήνυμα.

Έτσι, ουσιαστικά ο αρχικός εκδότης οποιουδήποτε μηνύματος καθορίζει το QoS του μηνύματος σε όλη τη διαδρομή μέχρι τους τελικούς παραλήπτες.

Ας δούμε τι σημαίνουν τα τρία επίπεδα με περισσότερες λεπτομέρειες.

3.3.1 QoS Level 0 - at most once

Αυτή είναι η απλούστερη, με το λιγότερο κόστος, μέθοδος αποστολής ενός μηνύματος. Ο πελάτης απλώς δημοσιεύει το μήνυμα και δεν υπάρχει επιβεβαίωση από τον broker.

3.3.2 QoS Level 1 - at least once

Αυτή η μέθοδος εγγυάται ότι το μήνυμα θα μεταφερθεί επιτυχώς στον διαμεσολαβητή.

Ο διαμεσολαβητής στέλνει μια επιβεβαίωση πίσω στον αποστολέα, αλλά σε περίπτωση που η επιβεβαίωση χαθεί, ο αποστολέας δεν θα αντιληφθεί ότι το μήνυμα πέρασε, οπότε θα στείλει το μήνυμα ξανά. Ο πελάτης θα ξαναστείλει μέχρι να λάβει την επιβεβαίωση του διαμεσολαβητή.

Αυτό σημαίνει ότι η αποστολή είναι εγγυημένη, αν και το μήνυμα μπορεί να φτάσει στον μεσίτη περισσότερες από μία φορές.

3.3.3 QoS Level 2 - exactly once

Αυτό είναι το υψηλότερο επίπεδο υπηρεσίας, στο οποίο υπάρχει μια ακολουθία τεσσάρων μηνυμάτων μεταξύ του αποστολέα και του παραλήπτη, ένα είδος χειραψίας για να επιβεβαιωθεί ότι το κύριο μήνυμα έχει σταλεί και ότι η επιβεβαίωση έχει ληφθεί.

Όταν ολοκληρωθεί η χειραψία, τόσο ο αποστολέας όσο και ο παραλήπτης είναι σίγουροι ότι το μήνυμα στάλθηκε ακριβώς μία φορά.

3.3.4 Χρήση στην κατάληλη περίπτωση

Το QoS 0 έχει το χαμηλότερο κόστος όσον αφορά τον όγκο μεταφοράς δεδομένων. Αυτό είναι κατάλληλο όταν έχετε μια αξιόπιστη σύνδεση μεταξύ συσκευής και διαμεσολαβητή.

Θα πρέπει να εξετάσετε αν η εφαρμογή IoT σας μπορεί να ανεχθεί την απώλεια ενός μηνύματος που και που. Για παράδειγμα, εάν μια συσκευή παρακολουθεί κάτι και στέλνει αθροιστικές μετρήσεις, τότε ο αντίκτυπος ενός χαμένου μηνύματος θα είναι απλώς μια καθυστέρηση στην άφιξη αυτών των μετρήσεων στον διακομιστή. Το QoS 1 είναι μια καλή επιλογή εάν πρέπει να είστε βέβαιοι ότι κάθε μήνυμα φτάνει, αλλά η εφαρμογή IoT μπορεί να ανεχθεί τη λήψη ενός μηνύματος περισσότερες από μία φορές.

Εάν η συσκευή αποστολής μεταδίδει αθροιστικές μετρήσεις δεδομένων από ένα περιουσιακό στοιχείο, τότε το QoS 1 μπορεί να είναι κατάλληλο, επειδή τυχόν περιστατικά λήψης ενός μηνύματος περισσότερες από μία φορές από τον διακομιστή θα πρέπει να έχουν μικρή επίδραση. Εάν όμως οι μετρήσεις που αποστέλλονται είναι οτι θέλουμέ να δούμε μια ¨αλλαγή από την τελευταία μετάδοση' (για παράδειγμα, αριθμός των φορών που πατήθηκε ένα κουμπί), τότε ένα διπλό μήνυμα μπορεί να δημιουργήσει παραπλανητικά δεδομένα στην πύλη.

Ένας τρόπος για να το παρακάμψετε αυτό είναι να διασφαλίσετε ότι οι συσκευές στέλνουν κάθε μήνυμα με μοναδική χρονοσφραγίδα. Το Assetwolf θα αναγνωρίσει ένα διπλότυπο μήνυμα με βάση τη χρονοσφραγίδα και θα το αγνοήσει. Το QoS 2 εγγυάται παράδοση ακριβώς μία φορά, αλλά έχει σχετικά υψηλό κόστος όσον αφορά τη μεταφορά δεδομένων.

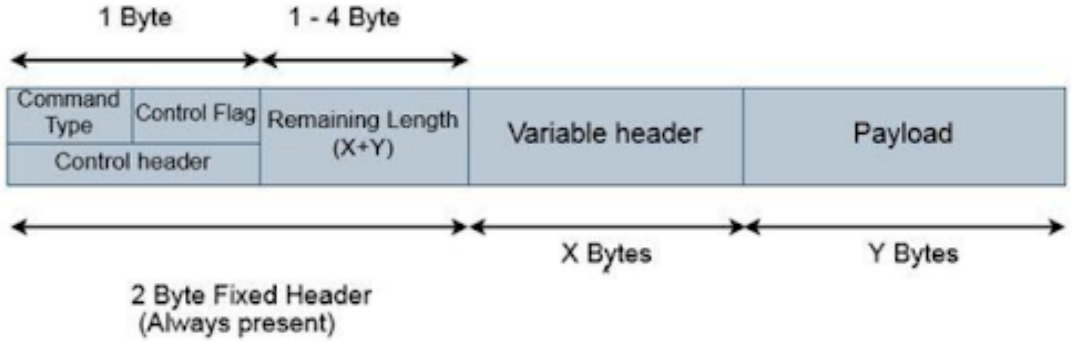
Οι περισσότερες εφαρμογές θα πρέπει να λειτουργούν χωρίς πρόβλημα με QoS 2, αλλά θα πρέπει να εξετάσετε αν θα ήταν κατάλληλο ένα QoS με χαμηλότερο κόστος. Συχνά μέσω προσεκτικής επιλογής των πεδίων στα δεδομένα - χρησιμοποιώντας χρονοσφραγίδες και αθροιστικές αναγνώσεις δεδομένων - είναι ένας εύκολος τρόπος να αποφύγετε την επιβάρυνση του QoS 2.

3.4 Δομή του μηνύματος

Τα μηνύματα πρωτόκολου MQTT αποτελείται από 3 κύρια μέρη:

1. Fixed header
2. Variable header
3. Payload

Η fixed header μεγέθους 2-byte εμπεριέχεται σε όλα τα μηνύματα ενώ τα άλλα δυο μέρη δεν είναι εμφανίζονται πάντα[1].



Σχήμα 3.7: MQTT μήνυμα

Κεφάλαιο 4

Apache Kafka

4.1 Εισαγωγή

Ο Apache Kafka είναι μια κατανεμημένη πλατφόρμα αποθήκευσης συμβάντων και επεξεργασίας ροής. Είναι ένα σύστημα ανοικτού κώδικα που αναπτύχθηκε από το Apache Software Foundation και είναι γραμμένο σε Java και Scala. Στόχος του έργου είναι η παροχή μιας ενοποιημένης πλατφόρμας υψηλής απόδοσης και χαμηλής καθυστέρησης για τον χειρισμό ροών δεδομένων σε πραγματικό χρόνο. Το Kafka μπορεί να συνδεθεί με εξωτερικά συστήματα (για εισαγωγή/εξαγωγή δεδομένων) μέσω του Kafka Connect και παρέχει τις βιβλιοθήκες Kafka Streams για εφαρμογές επεξεργασίας ροής. Το Kafka χρησιμοποιεί ένα δυαδικό πρωτόκολλο βασισμένο στο TCP, το οποίο είναι βελτιστοποιημένο για αποδοτικότητα και βασίζεται σε μια αφαίρεση ¨συνόλου μηνυμάτων' που ομαδοποιεί φυσικά τα μηνύματα μεταξύ τους για να μειώσει την επιβάρυνση του κυκλικού ταξιδιού στο δίκτυο. Αυτό οδηγεί σε μεγαλύτερα πακέτα δικτύου, μεγαλύτερες διαδοχικές λειτουργίες δίσκου, συνεχόμενα μπλοκ μνήμης που επιτρέπει στην Kafka να μετατρέψει μια εκρηκτική ροή τυχαίων εγγραφών μηνυμάτων σε γραμμικές εγγραφές.

Σχήμα 4.1: Apache Kafka Logo

Σε αυτή την τεχνική, ο αποστολέας (publisher) δεν στέλνει δεδομένα απευθείας στον παραλήπτη (subscriber), αλλά τα διαχωρίζει με ετικέτες ανάλογα με το θέμα τους και τα διαβιβάζει χωρίς να γνωρίζει αν υπάρχουν καποίοι που ενδιαφέροντε (subscribers) για να τα δεχτούν. Ομοίως, ένας ενδιαφερόμενος συνδρομητής (subscriber) κάνει subscribe σε ένα θέμα με μηνύματα που θέλει να λάβει, χωρίς να γνωρίζει αν υπάρχει κάποιος στο θέμα αυτό που θα στείλει δεδομένα. Από την ανάλυση του συστήματος καταλήγουμε ότι υπάρχει ένα

κύριο σύστημα (broker) στο οποίο αρχειοθετούνται και αποστέλλονται τα μηνύματα σε σχέση με το θέμα (topic) τους.Το σύστημα αύτο αναλαμβάνει να δέχεται τα μηνύματα όχι ίδιων θεμάτων και να τα διαβιβάζει στους ενδιαφερόμενους χωρίς να συνδέει κάθε αποστολέα με τον παραλήπτη.

Το Apache Kafka αναφέρεται μερικές φορές και ως κατανεμημένο αρχείο καταγραφής συμβάντων (distributed event log) το οποίο έχει αμετάβλητα μηνύματα που αποθηκεύονται και προσωρινά στον δίσκο με βάση την πολιτική που έχει ο broker γεγονός που αποτελεί την κύρια διαφορά μεταξύ του Kafka και άλλων παρόμοιων συστημάτων ανταλλαγής μηνυμάτων [2].

Το Kafka παρουσιάζεται συχνά ως ένα υβρίδιο ενός συστήματος ανταλλαγής μηνυμάτων και μιας βάσης δεδομένων και έχει σχεδιαστεί για να χειρίζεται πραγματικού χρόνου ροές δεδομένων και data pipelines αξιόπιστα και με υψηλή απόδοση. Παρέχει επίσης έναν κοινό χώρο για την αποθήκευση δεδομένων και τη διαχείριση συμβάντων, μειώνοντας το πρόβλημα της ολοκλήρωσης και της ενοποίησης, έχοντας όλα όσα χρειάζεται η εφαρμογή σε ένα κοινό μέρος. Χρησιμοποιείται ακόμη για τη δημιουργία συστημάτων ETL (Extract, Transform, Load), CDC (Change Data Capture) και Big Data Ingest.

4.2 Αρχιτεκτονική

Καθώς οι διάφορες εφαρμογές σχεδιάζουν την αρχιτεκτονική του Kafka ανάλογα, υπάρχουν τα ακόλουθα βασικά μέρη που απαιτούνται για τον σχεδιασμό της αρχιτεκτονικής του Apache Kafka.

1. Οικοσύστημα δεδομένων: Αρκετές εφαρμογές που χρησιμοποιούν τον Apache Kafka αποτελούν ένα οικοσύστημα. Αυτό το οικοσύστημα είναι κατασκευασμένο για την επεξεργασία δεδομένων. Λαμβάνει εισροές με τη μορφή εφαρμογών που δημιουργούν δεδομένα και οι εκροές ορίζονται με τη μορφή μετρήσεων, αναφορών κ.λπ. Το παρακάτω διάγραμμα αναπαριστά ένα κυκλικό οικοσύστημα δεδομένων για την Kafka

2. Συστάδα Kafka: Ένα σύμπλεγμα Kafka είναι ένα σύστημα που αποτελείται από διαφορετικούς μεσίτες, θέματα και τα αντίστοιχα διαμερίσματά τους. Τα δεδομένα γράφονται στο θέμα εντός της συστάδας και διαβάζονται από την ίδια τη συστάδα.

3. Παραγωγοί: Ένας παραγωγός στέλνει ή γράφει δεδομένα/μηνύματα στο θέμα εντός της συστάδας. Προκειμένου να αποθηκευτεί ένας τεράστιος όγκος δεδομένων, διάφοροι παραγωγοί εντός μιας εφαρμογής στέλνουν δεδομένα στο σμήνος Kafka.

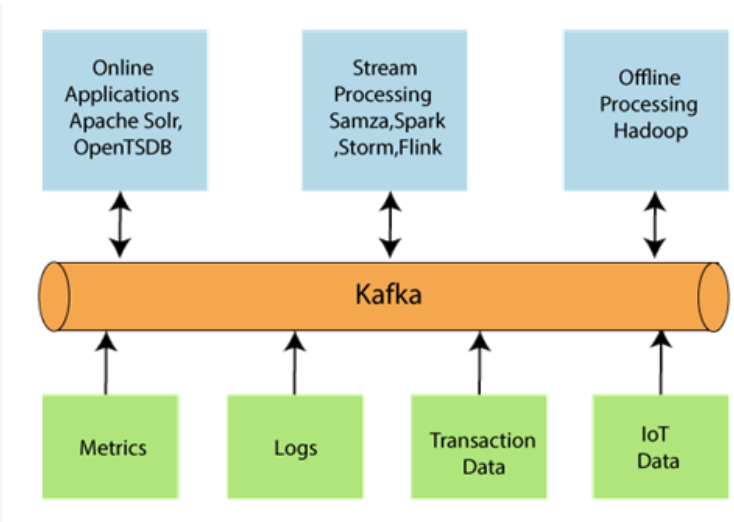
4. Καταναλωτές: Ένας καταναλωτής είναι αυτός που διαβάζει ή καταναλώνει μηνύματα από το σμήνος Kafka. Μπορεί να υπάρχουν διάφοροι καταναλωτές που καταναλώνουν διαφορετικούς τύπους δεδομένων από το σύμπλεγμα. Η ομορφιά του Kafka είναι ότι κάθε καταναλωτής γνωρίζει από πού πρέπει να καταναλώσει τα δεδομένα.

5. Broker: Ένας διακομιστής Kafka είναι γνωστός ως broker. Ένας μεσίτης είναι μια γέφυρα μεταξύ παραγωγών και καταναλωτών. Εάν ένας παραγωγός επιθυμεί να γράψει δεδομένα στη συστάδα, αυτά αποστέλλονται στον διακομιστή Kafka. Όλοι οι μεσίτες βρίσκονται μέσα σε μια συστάδα Kafka. Επίσης, μπορεί να υπάρχουν πολλαπλοί μεσίτες.

6. Θέματα: Είναι ένα κοινό όνομα ή μια επικεφαλίδα που δίνεται για να αναπαραστήσει έναν παρόμοιο τύπο δεδομένων. Στον Apache Kafka, μπορούν να υπάρχουν πολλαπλά θέματα σε μια συστάδα. Κάθε θέμα καθορίζει διαφορετικούς τύπους μηνυμάτων.

7. Partition: Τα δεδομένα ή το μήνυμα χωρίζονται σε μικρά υποτμήματα, γνωστά ως διαμερίσματα. Κάθε κατάτμηση φέρει δεδομένα που έχουν μια τιμή μετατόπισης. Τα δεδομένα εγγράφονται πάντα με διαδοχικό τρόπο. Μπορούμε να έχουμε άπειρο αριθμό διαμερισμάτων με άπειρες τιμές μετατόπισης. Ωστόσο, δεν είναι εγγυημένο σε ποιο διαμέρισμα θα γραφτεί το μήνυμα.

8. ZooKeeper: Ένας ZooKeeper χρησιμοποιείται για την αποθήκευση πληροφοριών σχετικά με τη συστάδα Kafka και λεπτομέρειες για τους πελάτες-καταναλωτές. Διαχειρίζεται τους διαμεσολαβητές διατηρώντας μια λίστα με αυτούς. Επίσης, ένας ZooKeeper είναι υπεύθυνος για την επιλογή ενός ηγέτη για τις κατατμήσεις. Εάν προκύψουν οποιεσδήποτε αλλαγές, όπως ο θάνατος ενός μεσίτη, νέα θέματα κ.λπ., ο ZooKeeper στέλνει ειδοποιήσεις στον Apache Kafka. Ένας ZooKeeper έχει σχεδιαστεί για να λειτουργεί με μονό αριθμό διακομιστών Kafka. Ο Zookeeper έχει έναν κορυφαίο διακομιστή που χειρίζεται όλες τις εγγραφές και οι υπόλοιποι διακομιστές είναι οι ακόλουθοι που χειρίζονται όλες τις αναγνώσεις. Ωστόσο, ένας χρήστης δεν αλληλεπιδρά άμεσα με τον Zookeeper, αλλά μέσω των μεσιτών. Κανένας διακομιστής Kafka δεν μπορεί να λειτουργήσει χωρίς διακομιστή zookeeper. Είναι υποχρεωτική η εκτέλεση του διακομιστή zookeeper.



Σχήμα 4.2: Αρχιτεκτονική Apache Kafka

[12]

4.3 Χαρακτηριστικά Kafka

Retention (Αποθήκευση)

Ο διακομιστής Kafka προσφέρει τη δυνατότητα αποθήκευσης των μηνυμάτων σε ένα θέμα είτε για ορισμένο χρονικό διάστημα (7 ημέρες από προεπιλογή) είτε μέχρι τα μηνύματα να φτάσουν ένα ορισμένο μέγεθος (για παράδειγμα, 1 GB). Όταν πληρούνται και οι δύο προϋποθέσεις, τα μηνύματα διαγράφονται με βάση την ηλικία, ώστε να υπάρχουν ακόμα δεδομένα στο topic. Ανάλογα με το topic οι ρυθμίσεις αποθήκευσης μπορούν να μεταβληθούν, π.χ. για πολύ ενεργά topics το χρονικό όριο αποθήκευσης μπορεί να είναι μερικές ώρες[2].

Reliability Guarantees (Εγγυήσεις Αξιοπιστίας) [2]

1. Οι consumers μπορούν να αναγνωρίζουν μόνο committed μηνύματα. Ο Kafka διαθέτει at-least-once message delivery semantics, επομένως το committed μήνυμα δεν θα εξαφανιστεί αλλά δεν μπορεί να αποφευχθεί το διάβασμα διπλών εγγραφών. Για exactly-once semantics στηριζόμαστε είτε στα Kafka Streams είτε σε εξωτερικά συστήματα .

2. Τα μηνύματα ορίζονται committed σε περίπτωση που αποτυπόνονται στον partition leader και ενημερόνονται όλα τα αντίγραφα (replicas) του partition με το στιγμιότυπο του leader. Ο Kafka επιβεβαιώνει ότι τα committed μηνύματα δεν θα εξαφανιστούν αφού υπάρχει το ελάχιστο μια replica που τα συμπεριλαμβάνει και η πολιτική αποθήκευσης δεν έχει ενεργοποιήσει την διαγραφή τους.

3. Χρονική σειρά των γεγονότων σε ένα partition. Αν το μήνυμα Β γράφεται μετα απο το μήνυμα Α από τον ίδιο producer στο ίδιο partition τότε το μήνυμα Α θα έχει μικρότερο offset από το μήνυμα Β. Δηλαδή ένας consumer θα λάβει πρώτα το μηνυμα Α και ύστερα το Β.

Trade-Offs [2]

Στον Kafka, όπως και σε κάθε κατανεμημένο σύστημα, καλούμαστε να βρούμε τη χρυσή τομή μεταξύ των δυνατοτήτων και των εγγυήσεων που προσφέρονται. Στην προκειμένη περίπτωση, ο χρήστης καλείται να επιλέξει μεταξύ συνέπειας και αξιοπιστίας και υψηλής απόδοσης, διαθεσιμότητας και χαμηλής καθυστέρησης. Σε κάθε περίπτωση, το σύστημα εγγυάται σε μεγάλο βαθμό όλα τα παραπάνω, αλλά αν θέλουμε υψηλότερες εγγυήσεις για ένα μέρος, πρέπει να θυσιάσουμε μια άλλη πτυχή του συστήματός μας. Για παράδειγμα, αν θέλουμε χαμηλή καθυστέρηση κατά την ανάγνωση μηνυμάτων από ένα topic που είναι αντίγραφο ενός topic του cluster, πρέπει να δεχτούμε την πιθανότητα το μήνυμα που λαμβάνουμε να είναι λανθασμένο λόγω έλλειψης χρόνου ενημέρωσης της τιμής του μηνύματος με βάση την τιμή του αρχικού μηνύματος στο leader του topic.

4.4 Πλεονεκτήματα & Μειονεκτήματα

4.4.1 Πλεονεκτήματα

Τα ακόλουθα πλεονεκτήματα του Apache Kafka το καθιστούν αξιόλογο:

1. Χαμηλή καθυστέρηση: Δηλαδή, μέχρι 10 χιλιοστά του δευτερολέπτου. Αυτό οφείλεται στο γεγονός ότι αποσυνδέει το μήνυμα, γεγονός που επιτρέπει στον καταναλωτή να καταναλώσει το εν λόγω μήνυμα ανά πάσα στιγμή.

2. Υψηλή απόδοση: Λόγω της χαμηλής καθυστέρησης, ο Kafka είναι σε θέση να διαχειριστεί μεγαλύτερο αριθμό μηνυμάτων υψηλού όγκου και υψηλής ταχύτητας. Ο Kafka μπορεί να υποστηρίξει χιλιάδες μηνύματα σε ένα δευτερόλεπτο. Πολλές εταιρείες, όπως η Uber, χρησιμοποιούν την Kafka για να φορτώνουν μεγάλο όγκο δεδομένων.

3. Ανοχή σφαλμάτων: Ο Kafka διαθέτει ένα βασικό χαρακτηριστικό που παρέχει αντοχή στην αποτυχία κόμβου/μηχανής εντός της συστάδας.

4. Ανθεκτικότητα: Ο Kafka προσφέρει τη δυνατότητα αντιγραφής, η οποία κάνει τα δεδομένα ή τα μηνύματα να παραμένουν περισσότερο στο σύμπλεγμα σε σχέση με έναν δίσκο. Αυτό το καθιστά ανθεκτικό.

5. Μειώνει την ανάγκη για πολλαπλές ενσωματώσεις: Όλα τα δεδομένα που γράφει ένας παραγωγός περνούν από τον Kafka. Ως εκ τούτου, χρειάζεται απλώς να δημιουργήσουμε μια ολοκλήρωση με τον Kafka, ο οποία μας ενσωματώνει αυτόματα με κάθε σύστημα παραγωγής και κατανάλωσης.

6. Εύκολα προσβάσιμο: Καθώς όλα τα δεδομένα μας αποθηκεύονται στον Kafka, γίνονται εύκολα προσβάσιμα από οποιονδήποτε.

7. Κατανεμημένο σύστημα: Ο Apache Kafka περιέχει μια κατανεμημένη αρχιτεκτονική που τον καθιστά επεκτάσιμο. Η κατάτμηση και η αντιγραφή είναι οι δύο δυνατότητες στο πλαίσιο του κατανεμημένου συστήματος.

8. Χειρισμός σε πραγματικό χρόνο: Ο Apache Kafka είναι σε θέση να χειρίζεται αγωγούς δεδομένων σε πραγματικό χρόνο. Η δημιουργία ενός αγωγού δεδομένων σε πραγματικό χρόνο περιλαμβάνει επεξεργαστές, ανάλυση, αποθήκευση κ.λπ.

9. Προσέγγιση δέσμης: Ο Kafka χρησιμοποιεί περιπτώσεις χρήσης που μοιάζουν με παρτίδες. Μπορεί επίσης να λειτουργήσει ως εργαλείο ETL λόγω της δυνατότητας εμμονής των δεδομένων του.

10. Επεκτασιμότητα: Η ιδιότητα του Kafka να χειρίζεται ταυτόχρονα μεγάλο αριθμό μηνυμάτων το καθιστά ένα κλιμακούμενο προϊόν λογισμικού.

4.4.2 Μειονεκτήματα

Με τα παραπάνω πλεονεκτήματα, υπάρχουν οι ακόλουθοι περιορισμοί/μειονεκτήματα του Apache Kafka:

1. Δεν διαθέτει πλήρες σύνολο εργαλείων παρακολούθησης: Ο Apache Kafka δεν περιέχει ένα πλήρες σύνολο εργαλείων παρακολούθησης καθώς και διαχείρισης. Έτσι, οι νέες νεοσύστατες επιχειρήσεις ή επιχειρήσεις φοβούνται να εργαστούν με τον Kafka.

2. Ζητήματα προσαρμογής μηνυμάτων: Ο διαμεσολαβητής Kafka χρησιμοποιεί κλήσεις συστήματος για την παράδοση μηνυμάτων στον καταναλωτή. Σε περίπτωση που το μήνυμα χρειάζεται κάποια διόρθωση, η απόδοση του Kafka μειώνεται σημαντικά. Έτσι, λειτουργεί καλά εάν το μήνυμα δεν χρειάζεται να αλλάξει.

3. Δεν υποστηρίζει την επιλογή θέματος wildcard: Ο Apache Kafka δεν υποστηρίζει επιλογή θέματος wildcard. Αντ΄ αυτού, ταιριάζει μόνο με το ακριβές όνομα θέματος. Αυτό συμβαίνει επειδή η επιλογή θεμάτων μπαλαντέρ την καθιστά ανίκανη να αντιμετωπίσει ορισμένες περιπτώσεις χρήσης.

4. Έλλειψη ορισμένων παραδειγμάτων μηνυμάτων: Ορισμένα παραδείγματα μηνυμάτων, όπως ουρές από σημείο σε σημείο, αίτημα/απάντηση κ.λπ. λείπουν από την Kafka για ορισμένες περιπτώσεις χρήσης.

[13] [19]

Κεφάλαιο 5

InfluxDB

5.1 Εισαγωγή

Η InfluxDB είναι μια βάση δεδομένων χρονοσειρών ανοικτού κώδικα (TSDB) που αναπτύχθηκε από την εταιρεία InfluxData. Είναι γραμμένη στη γλώσσα προγραμματισμού Go για την αποθήκευση και ανάκτηση δεδομένων χρονοσειρών σε τομείς όπως η παρακολούθηση λειτουργιών, οι μετρήσεις εφαρμογών, τα δεδομένα αισθητήρων του Διαδικτύου των πραγμάτων και η ανάλυση σε πραγματικό χρόνο.

Σχήμα 5.1: InfluxDB Logo

5.2 Time Series & Time Series DBs

5.2.1 Time Series

Χρονοσειρά είναι μια διατεταγμένη ακολουθία τιμών μιας μεταβλητής (π.χ. θερμοκρασία). σε ίσα χρονικά διαστήματα (π.χ. ωριαία). Συνεπώς, είναι μια ακολουθία δεδομένων διακριτού χρόνου. Για παράδειγμα, δεδομένα με χρονοσφραγίδα, όπως αρχεία καταγραφής και οι μετρήσεις IoT συσκευών μπορούν να θεωρηθούν χρονοσειρές. Οι μετρήσεις που συνιστούν μια χρονοσειρά διατάσσονται σε ένα χρονοδιάγραμμα, το οποίο αποκαλύπτει πληροφορίες για τα υποκείμενα πρότυπα. Η διάταξη έχει σημασία, επειδή υπάρχει εξάρτηση μεταξύ του χρόνου και των μετρήσεων και η αλλαγή της σειράς μπορεί να αλλάξει την το νόημα των δεδομένων. Παράδειγμα χρονοσειράς θα μπορούσαν να είναι οι ωριαίες μετρήσεις της θερμοκρασίας σε έναν συγκεκριμένο μετεωρολογικό σταθμό, οι ημερήσιες μετρήσεις των της τιμής κλεισίματος μιας συγκεκριμένης μετοχής, κ.λπ.

Οι χρονοσειρές χρησιμοποιούνται σε διάφορα πεδία, τα πιο συνηθισμένα από τα οποία είναι :

1. **Time Series Analysis**: Η ανάλυση χρονοσειρών χρησιμοποιείται προκειμένου να να διερευνήσει τον τρόπο με τον οποίο μια δεδομένη μεταβλητή μεταβάλλεται με την πάροδο του χρόνου. Για παράδειγμα, η απογραφή, δηλαδή ανάλυση της κοινής γνώμης για ένα συγκεκριμένο θέμα με την πάροδο του χρόνου, π.χ. οι υποψήφιοι πρόεδροι στις αμερικανικές εκλογές, μπορεί να θεωρηθεί μια χρονολογική Analysis Time Series (Ανάλυση χρονοσειρών).

2. **Regression Analysis**: Η ανάλυση παλινδρόμησης μπορεί να χρησιμοποιηθεί για την εξέταση πώς οι αλλαγές που σχετίζονται με μια συγκεκριμένη μεταβλητή μπορούν να προκαλέσουν μετατοπίσεις σε άλλες μεταβλητές κατά την ίδια χρονική περίοδο. Για παράδειγμα, η χρηματιστηριακή αγορά Ανάλυση, δηλαδή πώς οι τιμές μιας μετοχής με την πάροδο του χρόνου επηρεάζουν τις τιμές άλλων μετοχών τιμές ή την ανεργία κατά την ίδια χρονική περίοδο, μπορεί να εξεταστεί μια εργασία ανάλυσης παλινδρόμησης.

3. **Time Series Forecasting**: Η πρόβλεψη χρονοσειρών χρησιμοποιεί πληροφορίες σχετικά με τις ιστορικές τιμές και τα σχετικά πρότυπα για την πρόβλεψη μελλοντικών δραστηριότητα. Για παράδειγμα, οικονομική πρόβλεψη, πρόβλεψη καιρού, Πρόβλεψη σεισμών (σεισμικές χρονοσειρές), πρόβλεψη πωλήσεων, κ.λπ.

5.2.2 Time Series Databases

Ορισμός

Μια βάση δεδομένων χρονοσειρών (TSDB) είναι ένας τύπος βάσης δεδομένων που έχει βελτιστοποιηθεί για χρονοσειρές ή δεδομένα με χρονοσήμανση. Έχει κατασκευαστεί ειδικά για το χειρισμό μετρικών, γεγονότων ή μετρήσεων με χρονοσήμανση. Η TSDB είναι ιδανική για τη αναγνωρήση της αλλαγής με την πάροδο του χρόνου. Μια τέτοια βάση δεδομένων επιτρέπει στους χρήστες της να δημιουργούν, να απαριθμούν, να ενημερώνουν, να καταστρέφουν και να οργανώνουν διάφορες χρονοσειρές με πιο αποτελεσματικό τρόπο. Η βασική διαφορά με τα δεδομένα χρονοσειρών από τα κανονικά δεδομένα είναι ότι ως επί το πλείστον εσείς θέτετε queries σχετικά με την πάροδο του χρόνου. Σήμερα, η πλειονότητα των εταιρειών παράγουν μια παράλογα μεγάλη ροή μετρήσεων και γεγονότων (δεδομένα χρονοσειρών) και, ως εκ τούτου, η ανάγκη για μια βάση δεδομένων TSDB είναι αναπόφευκτη.

Ιδιότητες

Οι κύριες ιδιότητες που διακρίνουν τα δεδομένα χρονοσειρών από τα κανονικά δεδομένα είναι η σύνοψη, η διαχείριση του κύκλου ζωής των δεδομένων και το μεγάλο εύρος σαρώσεις πολλών εγγραφών. Η επισκόπηση ορισμένων από τις απαιτούμενες ιδιότητες ενός TSDB έχει ως εξής:

1. **Data Location**: Εάν τα σχετικά δεδομένα δεν βρίσκονται μαζί στο φυσικό αποθήκευτικο χώρο, τα ερωτήματα δεδομένων μπορεί να είναι πραγματικά αργά και να οδηγήσουν ακόμη και σε χρονοδιακοπές. Επειδή οι μη διαδοχικές λειτουργίες εισόδου/εξόδου εξακολουθούν να είναι πολύ αργές σε σύγκριση με με τις διαδοχικές I/O, ακόμη και όταν χρησιμοποιείται SSD. Μια TSDB τοποθετεί μαζί τα chucks δεδομένων εντός του ίδιου χρονικού εύρους στο ίδιο φυσικό τμήμα του συστημάτος της βάσης δεδομένων και ως εκ τούτου επιτρέπει τη γρήγορη πρόσβαση για ταχύτερη, αποδοτικότερη ανάλυση.

2. **Fast, easy range queries**: Καθώς μια TSDB διατηρεί τα συναφή δεδομένα μαζί, εξασφαλίζει ότι τα ερωτήματα εύρους είναι γρήγορα. Σε πολλές περιπτώσεις τακτικές βάσεις δεδομένων παράγουν σφάλμα ευρετηρίου εκτός μνήμης λόγω της τεράστιας όγκου των δεδομένων χρονοσειρών και στη συνέχεια επηρεάζουν την απόδοση των λειτουργιών ανάγνωσης και εγγραφής. Επιπλέον, θα πρέπει να λαμβάνεται υπόψη μας ότι η γλώσσα ερωτημάτων που χρησιμοποιείται θα πρέπει να διευκολύνει χρήστες να γράφουν τέτοια ερωτήματα.

3. **High write performance**: Πολλές βάσεις δεδομένων δεν είναι σε θέση να εξυπηρετούν αιτήματα προβλέψιμα και γρήγορα κατά τη διάρκεια αιχμής φορτίων. Οι βάσεις δεδομένων TSDB θα πρέπει να εξασφαλίζουν υψηλή διαθεσιμότητα και υψηλές επιδόσεις τόσο για λειτουργίες ανάγνωσης όσο και για εγγραφής κατά τη διάρκεια ώρα αιχμής, επειδή συνήθως είναι σχεδιασμένες να παραμένουν διαθέσιμες ακόμη και κάτω από τις πιο απαιτητικές συνθήκες. Δεδομένα χρονοσειρών καταγράφονται συνήθως κάθε δευτερόλεπτο ή ακόμη και λιγότερο από αυτό, οπότε οι εγγραφές πρέπει να είναι γρήγορες.

4. **Data compression**: Καθώς τα δεδομένα χρονοσειρών καταγράφονται ως επί το πλείστον ανά δευτερόλεπτο ή ακόμη και με μικρότερη λεπτομέρεια, χρειάζονται συνήθως μια καλύτερη τεχνική συμπίεσης δεδομένων. Και καθώς τα δεδομένα μεγαλώνουν σε μεγαλύτερη ηλικία, η υπερανάλυση γίνεται λιγότερο σημαντικές, οπότε οι βάσεις δεδομένων TSDB θα πρέπει να παρέχουν λειτουργικότητα για την εκτέλεση αναδιπλώσεων σε τέτοια σενάρια για συμπίεση δεδομένων.

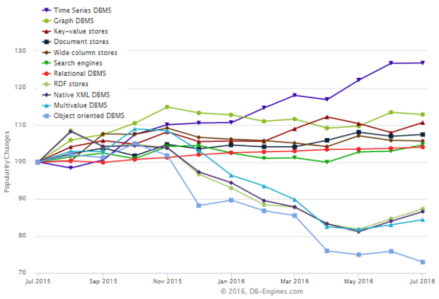
5. **Scalability**: Τα δεδομένα χρονοσειρών αυξάνονται πολύ γρήγορα. Για παράδειγμα, ένα συνδεδεμένο αυτοκίνητο θα στέλνει 25 GB δεδομένων στο cloud κάθε ώρα. Οι συνήθεις βάσεις δεδομένων δεν είναι σχεδιασμένες για να χειρίζονται αυτή την κλιμάκωση. Από την άλλη πλευρά, οι βάσεις δεδομένων χρονοσειρών έχουν σχεδιαστεί για να φροντίζουν η κλιμάκωση αυτή να αντιμετώπιζεται και να μην επηρεάζει την όρθη λειτουργία της αν έχουμε τον χρόνο ως το πρώτο μας μέλημα. Αυτό μπορεί να οδηγήσει σε βελτίωση των επιδόσεων, συμπεριλαμβανομένων: υψηλότερων ρυθμών εισαγωγής, ταχύτερων ερωτημάτων σε κλίμακα και καλύτερων συμπίεση δεδομένων.

6. **Usability**: Οι βάσεις δεδομένων TSDB περιλαμβάνουν συνήθως λειτουργίες που είναι κοινές για την ανάλυση δεδομένων χρονοσειρών. Για παράδειγμα, χρησιμοποιούν δεδομένα πολιτικές διατήρησης, συνεχείς αναζητήσεις, ευέλικτες χρονικές αθροίσεις, εύρος

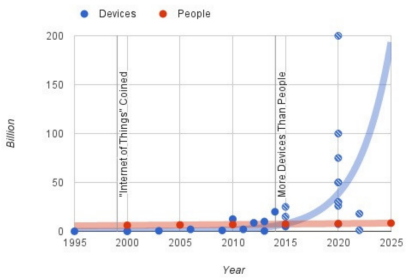
ερωτήματα κ.λπ. Έτσι, αυτό αυξάνει τη χρηστικότητα βελτιώνοντας την εμπειρία του χρήστη σε περίπτωση που ασχολείται με ανάλυση που σχετίζεται με το χρόνο.

Δημοτικότητα

Είναι προφανές ότι οι TSDBs είναι για τη διαχείριση δεδομένων χρονοσειρών, αλλά η δημοτικότητά τους φαίνεται να έχει αυξηθεί με την εμφάνιση του Διαδικτύου των πραγμάτων (IoT). Το IoT είναι ένα δίκτυο φυσικών συσκευών/αντικειμένων με συνδεσιμότητα που επιτρέπει να ανταλλάσσουν και να συλλέγουν δεδομένα. Οι τεχνολογίες αυτές δημιουργούν μεγάλη ποσότητα δεδομένων τα οποία συνήθως φέρουν χρονοσήμανση, οπότε με την αύξηση της δημοτικότητας του IoT, η δημοτικότητα των TSDB αυξήθηκε ακόμη περισσότερο, επειδή μπορούν να χρησιμοποιηθούν για την αποτελεσματική αποθήκευση δεδομένων αισθητήρων και συσκευών σε αυτόν τον τομέα. Ορισμένες άλλες συνήθεις χρήσεις των TSDB είναι η παρακολούθηση DevOps και η ανάλυση δεδομένων σε πραγματικό χρόνο. Σήμερα, πολλές μεγάλες εταιρείες όπως το Facebook, το eBay κ.λπ. χρησιμοποιούν TSDB αντί για σχεσιακές βάσεις δεδομένων, ιδίως για σκοπούς παρακολούθησης δεδομένων. Από το γράφημα στο Σχήμα 5.2 φαίνεται ότι η δημοτικότητα των TSDB αυξάνεται ραγδαία τα τελευταία δύο χρόνια. Από το 2015 έως το 2016 η δημοτικότητα των TSDBs σε αυξήθηκε κατά 26,7% που είναι διπλάσια από ό,τι τα συστήματα διαχείρισης βάσεων δεδομένων γραφημάτων που είναι τα 2α στον κατάλογο. Η δημοτικότητα των TSDBs συνεχίζει να αυξάνεται μέχρι σήμερα. Στην εικόνα 5.3 απεικονίζουμε την την αύξηση της δημοτικότητας των IoT ξεκινώντας επίσης από το 2015, για να δώσουμε μια ιδέα του την ταυτόχρονη ανάπτυξη των δύο τομέων.



Σχήμα 5.2: DB-Engines Ranking, Popularity Trend



Σχήμα 5.3: Σκιαγράφηση δημοτικότητας του Internet of Things

Οφέλη και χρήσεις

Όπως αναφέρθηκε προηγουμένως στην ενότητα 5.2.2, ορισμένες κοινές εφαρμογές των TSDBs είναι IoT, DevOps, Data Analytics κ.λπ. Αλλά το ερώτημα είναι ότι γιατί προτιμάμε τις TSDB έναντι των κανονικών βάσεων δεδομένων σε τέτοιες εφαρμογές· Ορισμένες πρόσθετα πλεονεκτήματα της χρήσης των TSDB είναι τα εξής 3 :

1. Μειωμένος χρόνος διακοπής λειτουργίας: Σε σενάρια πραγματικής ζωής υπάρχουν ορισμένες καταστάσεις όπου η διαθεσιμότητα είναι κρίσιμη ανά πάσα στιγμή, η αρχιτεκτονική μιας βάσης δεδομένων που είναι κατασκευασμένη για δεδομένα χρονοσειρών αποφεύγει οποιαδήποτε διακοπή λειτουργίας των δεδομένων ακόμη και σε περίπτωση κατατμήσεων δικτύου ή βλαβών υλικού.

2. Χαμηλότερο κόστος: Η ευελιξία και το υψηλό όριο αποτυχίας μεταφράζεται σε λιγότερους πόρους που απαιτούνται για τη διαχείριση των διακοπών λειτουργίας. Χρησιμοποιείται υλικό κοινής χρήσης για γρήγορη και εύκολη κλιμάκωση εξασφαλίζει τη μείωση του λειτουργικού κόστους και του κόστους υλικού για την κλιμάκωση προς τα πάνω ή προς τα κάτω.

3. Βελτιωμένες επιχειρηματικές αποφάσεις: Καθώς η TSDB επιτρέπει σε έναν οργανισμό να παρακολουθεί και να αναλύει δεδομένα σε πραγματικό χρόνο, τον βοηθά στη λήψη ταχύτερων και ακριβέστερες προσαρμογές για αλλαγές στην υποδομή, την κατανάλωση ενέργειας, τη συντήρηση των συσκευών ή άλλες σημαντικές αποφάσεις που επηρεάζουν την επιχείρηση.

Ορισμένες περιπτώσεις χρήσης για τις TSDB περιλαμβάνουν την παρακολούθηση συστημάτων λογισμικού, όπως τα εικονικά μηχανές, διάφορες υπηρεσίες ή εφαρμογές, παρακολούθηση φυσικών συστημάτων για για παράδειγμα κάποιο εξοπλισμό ή μηχανήματα, συνδεδεμένες συσκευές, το περιβάλλον, συστήματα οικιακής διαχείρισης, ανθρώπινα σώματα κ.λπ. Μια άλλη χρήση των TSDB είναι σε συστήματα χρηματοοικονομικών συναλλαγών για κλασικούς τίτλους ή σε κρυπτονομίσματα (BitCoin κ.λπ.). Οι TSDB μπορούν επίσης να χρησιμοποιηθούν ως εφαρμογές συμβάντων για την παρακολούθηση δεδομένων αλληλεπίδρασης χρήστη/πελάτη και σε εργαλεία επιχειρηματικής ευφυΐας για την παρακολούθηση βασικών μετρήσεων και της γενικής κατάστασης της επιχείρησης.

Top Time Series Databases

Οι κορυφαίες βάσεις δεδομένων TSDB και η κατάταξή τους φαίνονται στο Σχήμα 5.4 σύμφωνα με DB-Engines Ranking of Time Series DBMS. Η DB-Engines είναι ένας ανεξάρτητος δικτυακός τόπος που κατατάσσει τις βάσεις δεδομένων με βάση τη δημοτικότητα των μηχανών αναζήτησης, την κοινωνική τις αναφορές στα μέσα ενημέρωσης, τον αριθμό των προσφορών εργασίας και τη συχνότητα τεχνικών συζητήσεων. Η Influx DB κατατάσσεται στην πρώτη θέση σε αυτόν τον κατάλογο από τον Οκτώβριο του 2017. Εικόνα 2 δείχνει επίσης τις ιστορικές μεταβολές αυτών των βάσεων δεδομένων και φαίνεται ότι η InfluxDB ήταν επίσης η

κορυφαία TSDB τον Οκτώβριο του 2016 και διατήρησε την κατάταξη. Στην επόμενη ενότητα θα συζητήσουμε λεπτομερώς την InfluxDB. Η δεύτερη στον κατάλογο είναι η RRDtool, η οποία είναι ένα ανοιχτού κώδικα, βιομηχανικό πρότυπο εργαλείο καταγραφής δεδομένων και γραφικών παραστάσεων για δεδομένα χρονοσειρών. Οι υποστηριζόμενες γλώσσες προγραμματισμού είναι C, C#, Java, JavaScript, Lua, Perl, PHP, Python, Ruby . Το Graphite κατατάσσεται τρίτο στη λίστα, το οποίο είναι επίσης ένα εργαλείο καταγραφής δεδομένων ανοικτού κώδικα και εργαλείο γραφικών παραστάσεων το οποίο είναι υλοποιημένο σε Python και υποστηρίζει JavaScript και τις γλώσσες προγραμματισμού Πψτηον. Τόσο το RRDtool όσο και το Γραπηιτε ασχολούνται με αριθμητικές τιμές.

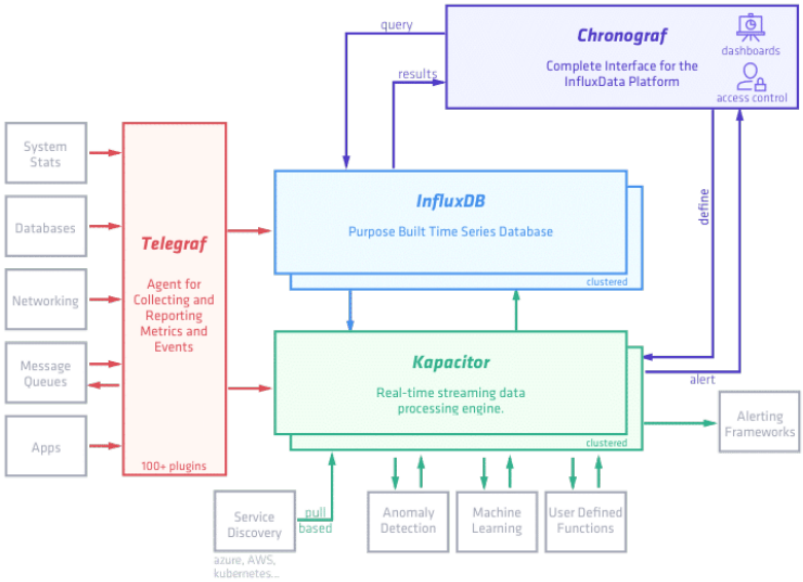
Rank			DBMS	Database Model	Score		
Oct 2017	Sep 2017	Oct 2016			Oct 2017	Sep 2017	Oct 2016
1.	1.	1.	InfluxDB	Time Series DBMS	8.70	+0.22	+3.38
2.	2.	2.	RRDtool	Time Series DBMS	3.12	+0.06	+0.64
3.	3.	3.	Graphite	Time Series DBMS	2.76	+0.21	+0.86
4.	4.	4.	OpenTSDB	Time Series DBMS	1.86	-0.03	+0.39
5.	5.	5.	Kdb+	Multi-model	1.83	+0.06	+0.62
6.	6.	6.	Druid	Time Series DBMS	1.00	+0.02	+0.40
7.	7.	7.	Prometheus	Time Series DBMS	0.74	+0.07	+0.46
8.	8.	8.	KairosDB	Time Series DBMS	0.48	-0.02	+0.21
9.	9.	↑ 10.	eXtremeDB	Multi-model	0.32	-0.02	+0.13
10.	10.	↓ 9.	Riak TS	Time Series DBMS	0.24	-0.02	+0.02
11.	11.	↑ 17.	Axibase	Time Series DBMS	0.18	-0.07	+0.18
12.	12.	↑ 19.	Hawkular Metrics	Time Series DBMS	0.15	+0.02	+0.15
13.	13.	↑ 18.	Blueflood	Time Series DBMS	0.14	+0.03	+0.14
14.	↑ 16.	↑ 15.	Machbase	Time Series DBMS	0.08	+0.03	+0.06
15.	↓ 14.	↓ 13.	Warp 10	Time Series DBMS	0.06	-0.05	+0.01
16.	↓ 15.	16.	TempoIQ	Time Series DBMS	0.04	-0.02	+0.03
17.	17.	↓ 12.	Heroic	Time Series DBMS	0.00	-0.01	-0.05
17.	↑ 18.	↓ 14.	Newts	Time Series DBMS	0.00	±0.00	-0.03
17.	↑ 18.		SiriDB	Time Series DBMS	0.00	±0.00	
17.	↑ 18.	↑ 19.	SiteWhere	Time Series DBMS	0.00	±0.00	±0.00
17.	↑ 18.	↓ 11.	Yanza	Time Series DBMS	0.00	±0.00	-0.06

Σχήμα 5.4: Top 21 Time Series Databases

5.3 Γενικές πληροφορίες & Αρχιτεκτονική

Η InfluxDB είναι μια βάση δεδομένων χρονοσειρών ανοικτού κώδικα με προαιρετική στοιχεία κλειστού κώδικα που αναπτύχθηκαν από την InfluxData. Είναι γραμμένη σε γλώσσα προγραμματισμού Go και είναι βελτιστοποιημένη για να χειρίζεται δεδομένα χρονοσειρών όπως

ορίζονται στην ενότητα 5.2.2. Παρέχει μια γλώσσα ερωτημάτων που μοιάζει με την SQL. Η έκδοση ανοικτού κώδικα, δηλαδή το TICK Stack (βλέπε εικόνα 5.5), παρέχει μια πλήρη χρονοσειρά πλατφόρμα βάσης δεδομένων με διάφορες υπηρεσίες, συμπεριλαμβανομένου του πυρήνα της InfluxDB, και μπορεί να να τρέχει σε νέφος και σε εγκαταστάσεις σε έναν μόνο κόμβο. Οι εκδόσεις κλειστού κώδικα, δηλαδή οι InfluxEnterprise (IE) και InfluxCloud (IC), προσφέρουν πρόσθετες λειτουργίες, όπως υψηλή διαθεσιμότητα, επεκτασιμότητα, δημιουργία αντιγράφων ασφαλείας και επαναφορά, και εκτελούνται είτε στις εγκαταστάσεις (IE) ή στο σύννεφο (IC). Περισσότερες λεπτομέρειες σχετικά με την καταλληλότητα του καθενός έκδοσης για συγκεκριμένες περιπτώσεις χρήσης θα δοθεί στην ενότητα 5.4.



Σχήμα 5.5: The open-source, Time Series Database Platform, TICK stack

Storage Engine

Η InfluxDB χρησιμοποιεί επί του παρόντος την εσωτερική δομή δεδομένων, το Time Structured Merge Tree (TSM Tree). Περισσότερες λεπτομέρειες σχετικά με αυτή τη μορφή αποθήκευσης θα θα δοθούν σύντομα. Ωστόσο, η InfluxDB έχει χρησιμοποιήσει διάφορες μορφές αποθήκευσης σε διάφορες εκδόσεις. Αρχικά χρησιμοποιούσε το LevelDB (μια βάση δεδομένων βασισμένη στο Log Structured Merge Trees (LSM)), η οποία βελτιστοποιεί την απόδοση εγγραφής και προσφέρει ενσωματωμένη συμπίεση. Ωστόσο, η LevelDB δεν παρέχει hot backup λειτουργικότητα, πράγμα που σημαίνει ότι πρέπει να κλείσετε τη βάση δεδομένων για να την αντιγράψετε με ασφάλεια. Για το λόγο αυτό η InfluxDB χρησιμοποίησε παραλλαγές της LevelDB, όπως η RocksDB και η HyperLevelDB, οι οποίες χρησιμοποιούν επίσης LSM Trees. Υπάρχει ένα εγγενές πρόβλημα με τα LSM Trees όμως, η διαγραφή είναι μια ακριβή λειτουργία και μια χρονοσειρά DB απαιτεί διαγραφές σε μεγάλη κλίμακα λόγω της

αυτόματης διατήρησης δεδομένων (βλ. Ενότητα 5.2.2). Αυτός είναι ο λόγος για τον οποίο η InfluxDB μεταπήδησε σε μια εναλλακτική δομή δεδομένων, το map B+Tree. Χρησιμοποίησε την BoltDB ως υποκείμενη μηχανή αποθήκευσης, η οποία μπορεί να έχει ελαφρώς χειρότερες επιδόσεις στις λειτουργίες εγγραφής, αλλά προσφέρει αυξημένη σταθερότητα και αξιοπιστία. Ωστόσο, συνειδητοποίησαν ότι όταν η βάση δεδομένων γινόταν μεγαλύτερη, οι εγγραφές θα άρχιζαν να αυξάνονται στα ύψη Input Output Operations per Second (IOPS). Στη συνέχεια, η ομάδα InfluxDB αποφάσισε να δημιουργήσει τη δική της αποθήκευση μορφή, το TSM Tree. Το TSM Tree είναι παρόμοιο με ένα LSM Tree κατά μία έννοια ότι χρησιμοποιεί αρχεία καταγραφής με δυνατότητα εγγραφής μπροστά, αρχεία ευρετηρίου που είναι μόνο για ανάγνωση και περιστασιακά εκτελεί συμπιέσεις για να συνδυάσει αρχεία ευρετηρίου. Ωστόσο, δεν πάσχει από το πρόβλημα της διαγραφής και προσφέρει καλύτερα ποσοστά συμπίεσης (βελτίωση 45 φορές) στη χρήση του χώρου στο δίσκο σε σύγκριση με ένα δέντρο B+.

5.4 Πελάτες & Use Case

Η InfluxDB έχει περισσότερες από 70000 ενεργές εγκαταστάσεις και προτιμάται από τους πελάτες για DevOps Monitoring (παρακολούθηση υποδομών, παρακολούθηση εφαρμογών, παρακολούθηση cloud), IoT Monitoring και Real-Time Analytics. Οι πελάτες της μεταξύ άλλων είναι οι AXA, Cisco, eBay, IBM και άλλες εταιρείες. Οι ιστορίες επιτυχίας των πελατών ανά κατηγορία μπορούν να βρεθούν παρακάτω.

5.4.1 DevOps Monitoring: The IBM Case

Τα προϊόντα IBM Trusteer βοηθούν στον εντοπισμό και την πρόληψη του πλήρους φάσματος των φορέων επίθεσης που ευθύνονται για την πλειονότητα της διαδικτυακής, κινητής και διακαναλικής απάτης. Προκειμένου να παρέχεται πλήρης προστασία από την ηλεκτρονική απάτη σε ανά πάσα στιγμή, η πλατφόρμα Trusteer πρέπει να διατηρεί υψηλή διαθεσιμότητα. Για το σκοπό αυτό η ομάδα της χρησιμοποιεί τεχνικές παρακολούθησης DevOps που υποστηρίζονται από την InfluxDB, Telegraf (ένα άλλο προϊόν του οικοσυστήματος InfluxData) και Grafana (λογισμικό ανοικτού κώδικα για την ανάλυση χρονοσειρών). Χρησιμοποιούν το Telgraf για τη συλλογή δεδομένα, το InfluxDB για την αποθήκευση τους και το Grafana για την ανάλυση και την οπτικοποίηση. Συλλέγουν δεδομένα σχετικά με τις επιδόσεις των υποδομών και των εφαρμογών, προκειμένου να να παρακολουθούν το σύστημα νέφους τους, το οποίο περιλαμβάνει εκατοντάδες εικονικούς διακομιστές.

5.4.2 IoT Monitoring: The Spiio Case

Το Spiio επιτρέπει την παρακολούθηση πράσινων τοίχων υψηλής αξίας. Σε αύτες τις εγκαταστάσεις φυτών παρέχοντε αισθητήρες και το σχετικό λογισμικό στους κηπουρούς. Οι πράσινοι τοίχοι γίνονται όλο και πιο δημοφιλείς, ιδίως σε μεγάλες πόλεις, φέρνοντας τη φύση πιο κοντά στη ζωή της πόλης. Ωστόσο, η διατήρηση πολλαπλών πράσινων τοίχων σε μια πόλη μπορεί να αποτελέσει πρόβλημα για τους επαγγελματίες της κηπουρικής. Αυτός είναι ο λόγος

ο οποιός η Spiio χρησιμοποιεί αισθητήρες για την κατανόηση της απόδοσης των φυτών από τα δεδομένα, και έτσι μειώνει δραστικά το κόστος συντήρησης, εξασφαλίζοντας τον πλήρη ψηφιακό έλεγχο εκατομμυρίων φυτών. Η Spiio προσπάθησε να υιοθετήσει διάφορες λύσεις πριν από την InfluxDB, δηλαδή πλατφόρμες IoT, όπως η AWS Greengrass και η Azure IOT, βάσεις δεδομένων πολλαπλών χρήσεων , όπως η MySQL, η Cassandra, η Elasticsearch, και ακόμη και βάσεις δεδομένων χρονοσειρών, όπως η OpenTSDB. Ωστόσο, καμία από αυτές τις λύσεις δεν ήταν τόσο ολιστική όσο η InfluxData. Επί του παρόντος, η Spiio χρησιμοποιεί την InfluxDB για την αποθήκευση δεδομένων, Kapacitor για την ανάλυση δεδομένων σε πραγματικό χρόνο και ροής και το Chronograf για δεδομένα οπτικοποίηση δεδομένων. Τόσο το Kapacitor όσο και το Chronograf αποτελούν μέρος του InfluxData οικοσύστημα. Το InfluxData δίνει τη δυνατότητα στους πελάτες της Spiio να έχουν πρόσβαση και να μοιράζονται πρωτόγνωρες γνώσεις σχετικά με τη βελτιστοποίηση της συντήρησης των πράσινων τοίχων, παρακολουθώντας τον αντίκτυπο των παραγόντων που επηρεάζουν την απόδοση των φυτών.

5.4.3 Real-Time Analytics: The eBay Case

Η EBay Inc. είναι παγκόσμιος ηγέτης στο ηλεκτρονικό εμπόριο. Διάφορες ομάδες εντός της eBay χρησιμοποιούν InfluxDB και γενικά το οικοσύστημα InfluxData για παρακολούθηση DevOps και ανάλυση σε πραγματικό χρόνο. Εδώ, θα επικεντρωθούμε στις αναλύσεις σε πραγματικό χρόνο στο εσωτερικό της eBay Experimentation team. Αυτή η ομάδα είναι υπεύθυνη για τον πειραματισμό της eBay πλατφόρμα, η οποία εκτελεί περισσότερα από 1.500 πειράματα και επιτρέπει στους επιχειρηματικούς χρήστες της eBay να αποκτήσουν εικόνα για σημαντικές αναλύσεις και να απαντήσουν σε κρίσιμες επιχειρηματικές ερωτήσεις. Ωστόσο, τα πειράματα μπορεί να παρουσιάσουν ανωμαλίες, όπως η κίνηση διαφθορά, και σε αυτό το σημείο αναλαμβάνει δράση η InfluxDB. Οι ανωμαλίες εντοπίζονται καθημερινά με τη χρήση αλγορίθμων πρόβλεψης ανωμαλιών/κυκλοφορίας, αποθηκεύονται στο InfluxDB και απεικονίζονται στο Grafana. Έχοντας αυτά τα αναλυμένα δεδομένα που αναπαρίστανται σε μορφή χρονοσειράς είναι καθοριστικής σημασίας και τους επιτρέπει να τα παρουσιάζουν στο Grafana τους dashboard. Αυτός ο συνδυασμός επιτρέπει στην πλατφόρμα πειραματισμού να είναι επεκτάσιμη και αυτάρκης, δηλαδή με τη δημιουργία νέων ταμπλό και συνόλων δεδομένων αυτόματα.

5.5 Πλεονεκτήματα & Μειονέκτημα

5.5.1 Πλεονεκτήματα

1. Προσαρμοσμένη για δεδομένα χρονοσειρών: Η InfluxDB έχει σχεδιαστεί για να χειρίζεται πιο αποτελεσματικά δεδομένα χρονοσειρών. Έχει σχεδιαστεί για να έχει εντυπωσιακές ταχύτητες εγγραφής και ανάγνωσης.

2. Λύσεις για κάθε ανάγκη: Η InfluxData παρέχει ένα σύστημα ανοικτού κώδικα πυρήνα που περιλαμβάνει τα InfluxDB, Kapacitor, Telegraf και Chronograf (το TICK Stack)

δωρεάν. Παρέχει επίσης μια λύση SaaS, δηλαδή το InfluxCloud, που προσφέρει υψηλή διαθεσιμότητα, επεκτασιμότητα και προηγμένη λειτουργίες δημιουργίας αντιγράφων ασφαλείας και επαναφοράς για χρήστες με μεγαλύτερες ανάγκες και περιορισμένη υποδομή. Το InfluxCloud αναπτύσσει διακομιστές στις ΗΠΑ, Καναδά και την Ευρώπη. Υπάρχει επίσης μια εσωτερική λύση του InfluxCloud, δηλαδή το InfluxEnterprise, για πελάτες που θέλουν να αξιοποιήσουν τη δική τους υποδομή ή υπηρεσίες cloud. Με άλλα λόγια, η InfluxDB προσφέρει διάφορες λύσεις που ανταποκρίνονται σε κάθε πιθανή επιχειρηματική ανάγκη.

3. Ολιστική λύση: Το InfluxDB έχει σχεδιαστεί για να λειτουργεί άριστα μαζί με το υπόλοιπο οικοσύστημα InfluxData, δηλαδή με τα Kapacitor, Telegraf και Chronogra. Υπό αυτή την έννοια είναι κάτι πολύ περισσότερο από ένα απλό βάση δεδομένων. Αποτελεί μέρος μιας ολιστικής λύσης που προσφέρει συσσώρευση, ανάλυση και οπτικοποίηση, όλα σε ένα πακέτο.

4. Διάφορα πρόσθετα εισόδου: Η InfluxDB δεν περιορίζεται σε ένα ή δύο μεθόδους εισόδου, όπως άλλες TSDB, αλλά προσφέρει διάφορα plugins εισόδου δωρεάν. Εκτός από το HTTP API, προσφέρει ένα πρόσθετο UDP, Graphite plugin, το οποίο επιτρέπει την είσοδο στο πρωτόκολλο γραμμής Graphite μορφή, το πρόσθετο CollectD, το οποίο επιτρέπει την είσοδο σε εγγενή μορφή collectd, OpenTSDB plugin, το οποίο επιτρέπει Telnet και HTTP OpenTSDB πρωτόκολλο. Αυτό σημαίνει ότι το InfluxDB μπορεί να λειτουργήσει ως αντικαταστάτης του για ένα σύστημα OpenTSDB.

5. Υποστήριξη Grafana: Η Grafana είναι το κατάλληλο λογισμικό για την ανάλυση χρονοσειρών, με πολύ περισσότερες από 100.000 ενεργές εγκαταστάσεις. Η Grafana έχει εισαγάγει ένα πρόσθετο για το InfluxDB ως πηγή δεδομένων για τις αναλύσεις της ταμπλό.

6. Εκτεταμένη υποστήριξη γλωσσών προγραμματισμού: Η InfluxDB προσφέρει υποστήριξη για διάφορες γλώσσες προγραμματισμού, μεταξύ άλλων σε: .Net, Java, Perl, PHP, Python, R, Ruby, Scala και άλλες.

7. Γλώσσα ερωτημάτων που μοιάζει με SQL: Η InfluxDB συνοδεύεται από μια γλώσσα ερωτημάτων που μοιάζει με SQL γλώσσα, την InfluxQL, που σημαίνει ότι δεν έχει απότομη καμπύλη εκμάθησης και είναι πιο εύκολο να γραφτεί και να κατανοηθεί και από μη τεχνικούς ανθρώπους σε σύγκριση, για παράδειγμα, με το OpenTSDB που δεν παρέχει τέτοια γλώσσα. Αυτό είναι εξαιρετικά σημαντικό όταν πρόκειται για τον κόσμο των επιχειρήσεων, όπου τα δεδομένα παίζουν σημαντικό ρόλο και τα χειρίζονται άνθρωποι με διαφορετικό υπόβαθρο.

8. Συνεχής υποστήριξη ερωτημάτων: Τα συνεχή ερωτήματα (CQ) είναι ερωτήματα InfluxQL που εκτελούνται αυτόματα και περιοδικά σε πραγματικό χρόνο δεδομένα και αποθηκεύουν τα αποτελέσματα των ερωτημάτων σε μια καθορισμένη μέτρηση. Τα CQs επιτρέπουν την δειγματοληψία (roll-up) των δεδομένων που ζητούνται συχνά, υψηλής

ακρίβειας σε χαμηλότερη ακρίβειας. Τα ερωτήματα σε δεδομένα με χαμηλότερη κοκκομετρία απαιτούν λιγότερους πόρους και είναι ταχύτερα από τα ερωτήματα με υψηλότερη κοκκομετρία.

9. Εύκολη εγκατάσταση: Η InfluxDB μεταγλωττίζεται σε ένα ενιαίο δυαδικό αρχείο με καμία εξάρτηση, γεγονός που καθιστά εξαιρετικά εύκολη την εγκατάσταση της.

10. Αυτόματη διαγραφή: Τα δεδομένα χρονοσειρών ενδέχεται να καταστούν λιγότερο σημαντικά ή ακόμη και άχρηστα, ανάλογα με την εφαρμογή, καθώς περνάει ο καιρός. Η InfluxDB με την χρήση των πολιτικών διατήρησης, όπως συζητήθηκε στην ενότητα 2.1.1, επιτρέπει την αυτόματη λήξη των παρωχημένων δεδομένων.

11. Απεριόριστα πεδία: Η νέα μηχανή αποθήκευσης της InfluxDB, η TSM Tree, όπως συζητήθηκε στην προηγούμενη υποενότητα, έχει μορφή στήλης, πράγμα που σημαίνει ότι ο αριθμός των πεδίων δεν επηρεάζει αρνητικά την απόδοση των ερωτημάτων. Κατά συνέπεια, ο αριθμός των πεδίων στη μέτρηση δεν δεν έχει επίσης περιορισμούς.

12. Ενσωματωμένη διεπαφή διαχειριστή στο διαδίκτυο: Όπως η SQL, έτσι και η InfluxDB παρέχει μια ενσωματωμένη διαδικτυακή διεπαφή για τους χρήστες που δεν είναι εξοικειωμένοι με διεπαφές γραμμής εντολών και θα προτιμούσαν μια πιο διαισθητική λύση.

13. Εκτεταμένη τεχνίκη υποστήριξη: Η InfluxData παρέχει έναν εκτενή οδηγό τεκμηρίωσης για το InfluxDB από την εγκατάσταση έως τα σύνθετα ερωτήματα και βελτιστοποίηση σχήματος.

5.5.2 Μειονέκτημα

1. Η επεκτασιμότητα ως χαρακτηριστικό στενής πηγής: Όταν η InfluxDB ανακοίνωσε ότι η ομαδοποίηση δεν θα περιλαμβανόταν στην έκδοση ανοικτού κώδικα, δέχθηκε κατακραυγή από την κοινότητα. Επί του παρόντος, τα χαρακτηριστικά υψηλής διαθεσιμότητας και επεκτασιμότητας του InfluxDB είναι στενού κώδικα. Ωστόσο, η InfluxData παρέχει μια λύση αντιγραφής ανοικτού κώδικα για υψηλή διαθεσιμότητα, ενώ πολλοί χρήστες χρησιμοποιούν το sharding ως μια λύση-παράκαμψη για την υπόλοιπη λειτουργικότητα ομαδοποίησης στην έκδοση ανοικτού κώδικα. Αυτό, για παράδειγμα, μπορεί να κάνει το InfluxDB να φαίνεται μη ελκυστικό για νεοσύστατες επιχειρήσεις με περιορισμένο προϋπολογισμό. Ωστόσο, η InfluxData παρέχει διάφορα πακέτα που ξεκινούν από 149 δολάρια το μήνα (ή 249 δολάρια το μήνα για την έκδοση cloud), ένα κόστος που δεν είναι απαγορευτικό ακόμη και για μικρότερες εταιρείες που θέλουν να επενδύσουν σε μια καλή κλιμακούμενη λύση.

2. Κοινοτικά ζητήματα: Δεδομένου ότι το InfluxDB είναι ένα σχετικά νέο προϊόν, η κοινότητά του είναι σχετικά μικρή σε σύγκριση με λύσεις όπως η Cassandra. Αυτό σημαίνει ότι μια απλή αναζήτηση στο Google μπορεί να μην επιφέρει το επιθυμητό αποτέλεσμα για κάθε πιθανό πρόβλημα. Ωστόσο, δεδομένου ότι η InfluxDB δημοτικότητα αυξάνεται , η κοινότητά της αναμένεται να αυξηθεί.

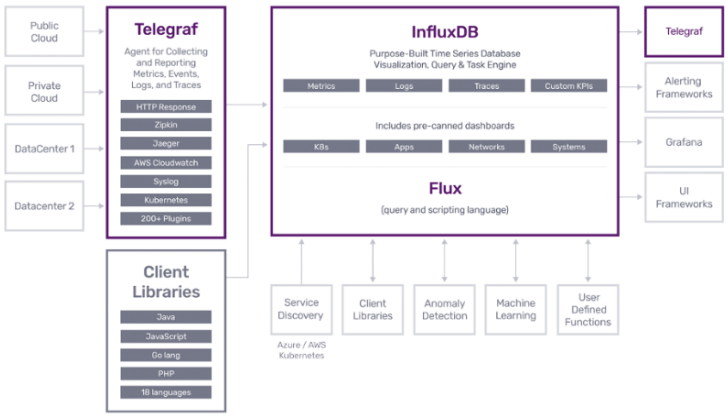
[20]

Κεφάλαιο 6

Telegraf & Grafana

6.1 Εισαγωγή

Για την ομαλή λειτουργία του συστήματος IoT θα πρέπει τα δεδομένα από μια βάση που χρησιμοποιηούμαι ή άπο καποιά ομάδα sensor να μεταφέρονται και να αναπαραστούνται τα δεδομένα χρονοσειρών από τις συσκευές αύτες στην InfluxDB. Τα δεδομένα αυτά μπορεί να παράγονται από συσκευές IoT και αισθητήρες στο πεδίο ή από τους διακομιστές σας που φιλοξενούνται ήδη σε ένα κέντρο δεδομένων AWS. Ένας από τους ευκολότερους τρόπους για την εισαγωγή δεδομένων είναι η χρήση του Telegraf και για αναπαράσταση είναι το Grafana. Ένα παράδειγμα αρχιτεκτονικής InfluxDB με Telegraf και Grafana είναι αυτό που αναπαραστάται στο σχήμα 6.1.



Σχήμα 6.1: Αρχιτεκτονική InfluxDB με Telegraf και Grafana

Το Telegraf είναι ένας πράκτορας διακομιστή ανοικτού κώδικα που διαθέτει πρόσθετα για τη συλλογή δεδομένων από περισσότερες από 250 διαφορετικές πηγές δεδομένων και πρωτόκολλα δικτύου. Τα δεδομένα μπορούν στη συνέχεια να σταλούν σε 50 διαφορετικές επιλογές εξόδου για αποθήκευση ή επεξεργασία.

Ως εργαλείο απεικόνισης, το Grafana είναι ένα δημοφιλές εργαλείο για παρακολούθησης δεδομένων, το οποίο χρησιμοποιείται συχνά σε συνδυασμό με βάσεις δεδομένων χρονοσειρών

όπως οι InfluxDB, Prometheus και Graphite, πλατφόρμες παρακολούθησης όπως οι Sensu, Icinga, Checkmk, Zabbix, Netdata και PRTG,SIEM όπως οι Elasticsearch και Splunk και άλλες πηγές δεδομένων.

6.2 Telegraf

6.2.1 Εισαγωγή

Το Telegraf, είναι ένας πράκτορας που βασίζεται σε διακομιστή, συλλέγει και αποστέλλει μετρήσεις και συμβάντα από βάσεις δεδομένων, συστήματα και αισθητήρες IoT. Γραμμένο σε Go, το Telegraf μεταγλωττίζεται σε ένα ενιαίο δυαδικό αρχείο χωρίς εξωτερικές εξαρτήσεις - απαιτώντας ελάχιστη μνήμη.Τέλος έχει την δυνατότητα εσωτερικής αναπαράστασης των δεδομένων.

Σχήμα 6.2: Telegraf Logo

6.2.2 Δυνατότητες του Telegraf

Η χρήση του Telegraf στο συστήμα σου, παρέχει κάποιος δυνατότητες και ασφάλεια στο σύστημα IoT.

Μια απο αύτες είναι ότι για περιπτώσεις χρήσης IoT σε άκραίες καταστάσεις όπου οι διακομιστές σας χάνουν τη σύνδεση δικτύου, το Telegraf έχει τη δυνατότητα να αποθηκεύει τοπικά τις μετρήσεις στη μνήμη μέχρι να αποκατασταθεί η σύνδεση και στη συνέχεια να στέλνει όλα τα συσσωρευμένα δεδομένα στον καθορισμένο προορισμό. Το Telegraf μπορεί να εγκατασταθεί σε δικό του διακομιστή ή τοπικά σε μια συσκευή IoT για μέγιστη ευελιξία. Η εγκατάσταση είναι εύκολη, επειδή το Telegraf μπορεί να αναπτυχθεί ως ένα ενιαίο δυαδικό αρχείο χωρίς εξαρτήσεις.

Αλλη μία δυνατότητα είναι ότι μπορείτε πριν τοποθετήσετε τα δεδομένα χρονοσειρών σας σε μακροπρόθεσμη αποθήκευση, ίσως θελήσετε να τα εμπλουτίσετε με δεδομένα από άλλες πηγές ή να τα επεξεργαστείτε γρήγορα για να αποκτήσετε πληροφορίες σε πραγματικό χρόνο. Εάν χρησιμοποιείτε το Telegraf, μπορείτε να το κάνετε αυτό εκμεταλλευόμενοι κάποια από τα ενσωματωμένα από την κοινότητα πρόσθετα επεξεργασίας ή ακόμη και να δημιουργήσετε τον δικό σας προσαρμοσμένο επεξεργαστή για να μετασχηματίσετε ή να εμπλουτίσετε τα δεδομένα σας. Ορισμένοι ευρέως χρησιμοποιούμενοι επεξεργαστές περιλαμβάνουν τον επεξεργαστή regex για το μετασχηματισμό των τιμών ετικετών και πεδίων. Το πρόσθετο του

επεξεργαστή Execd, το οποίο σας επιτρέπει να εκτελείτε οποιοδήποτε εξωτερικό πρόγραμμα για τον μετασχηματισμό των δεδομένων σας, όπως για παράδειγμα ένα σενάριο Python.

6.2.3 Μετρικές του Telegraf

Οι μετρικές του Telegraf είναι η εσωτερική αναπαράσταση που χρησιμοποιείται για τη μοντελοποίηση των δεδομένων κατά την επεξεργασία. Αυτές οι μετρικές βασίζονται στενά στο μοντέλο δεδομένων του InfluxDB και περιέχουν τέσσερα κύρια στοιχεία:

1. Όνομα μέτρησης: Περιγραφή και χώρος ονομάτων για τη μέτρηση.

2. Ετικέτες: Ζεύγη συμβολοσειρών κλειδιού/τιμής και συνήθως χρησιμοποιούνται για τον προσδιορισμό της μέτρησης.

3. Πεδία: Ζεύγη Key/Value που είναι τυποποιημένα και συνήθως περιέχουν τα δεδομένα της μέτρησης.

4. Χρονοσφραγίδα: Ημερομηνία και ώρα που σχετίζονται με τα πεδία.

Αυτός ο τύπος μετρικής υπάρχει μόνο στη μνήμη και πρέπει να μετατραπεί σε συγκεκριμένη αναπαράσταση προκειμένου να μεταδοθεί ή να προβληθεί. Το Telegraf παρέχει μορφές δεδομένων εξόδου (γνωστές και ως σειριοποιητές) για αυτές τις μετατροπές. Ο προεπιλεγμένος σειριοποιητής του Telegraf μετατρέπει σε InfluxDB Line Protocol, το οποίο παρέχει υψηλή απόδοση και απευθείας αντιστοίχιση ένα προς ένα από τις μετρικές του Telegraf.

6.2.4 Εξωτερικά plugins

Τα εξωτερικά πρόσθετα (plugins) είναι εξωτερικά προγράμματα που έχουν δημιουργηθεί εκτός του Telegraf και μπορούν να εκτελεστούν μέσω ενός πρόσθετου execd. Αυτά τα εξωτερικά πρόσθετα επιτρέπουν μεγαλύτερη ευελιξία σε σύγκριση με τα εσωτερικά πρόσθετα του Telegraf. Τα οφέλη από τη χρήση εξωτερικών πρόσθετων περιλαμβάνουν:

1. Πρόσβαση σε βιβλιοθήκες που δεν είναι γραμμένες σε Go

2. Χρήση αδειοδοτημένου λογισμικού (που δεν είναι διαθέσιμο στην κοινότητα ανοικτού κώδικα)

3. Συμπερίληψη μεγάλων εξαρτήσεων που διαφορετικά θα φούσκωναν το Telegraf

4. Χρήση του εξωτερικού σας πρόσθετου αμέσως χωρίς να περιμένετε τη δημοσίευση από την ομάδα του Telegraf

5. Εύκολη μετατροπή των πρόσθετων μεταξύ εσωτερικών και εξωτερικών χρησιμοποιώντας το shim

Χρήση του shim 'execd'

Το shim διευκολύνει την εξαγωγή ενός εσωτερικού plugin εισόδου, επεξεργαστή ή εξόδου από το κύριο repo του Telegraf σε ένα αυτόνομο repo. Αυτό επιτρέπει σε οποιονδήποτε να το κατασκευάσει και να το εκτελέσει ως ξεχωριστή εφαρμογή χρησιμοποιώντας ένα από τα πρόσθετα execd: inputs.execd processors.execd outputs.execd.

Χρήση εξωτερικού πρόσθετου

Έχετε την δυνατότητα να ρυθμίσετε το πρόσθετό σας ώστε να το χρησιμοποιείτε με το execd. Για τα καταχωρημένα εξωτερικά πρόσθετα, ο συγγραφέας του εξωτερικού πρόσθετου είναι επίσης υπεύθυνος για τη συντήρηση και την ανάπτυξη χαρακτηριστικών των εξωτερικών πρόσθετων.

[11]

6.3 Grafana

6.3.1 Εισαγωγή

Το Grafana είναι ένα δημοφιλές εργαλείο ερωτημάτων, απεικόνισης και ειδοποίησης δεδομένων χρονοσειρών ανοικτού κώδικα το οποίο αναπτύχθηκε από τον Torkel Ödegaard το 2014. Διαθέτει ένα μοντέλο πηγής δεδομένων το οποίο είναι εξαιρετικά επεκτάσιμο και υποστηρίζει πολλαπλές πηγές δεδομένων που βασίζονται σε χρονοσειρές, όπως το Προμετηευς, InfluxDB και OpenTSDB, καθώς και βάσεις δεδομένων SQL όπως η MySQL και η Postgres. Ανεξάρτητα από το πού αποθηκεύονται τα δεδομένα, σας επιτρέπει να κάνετε ερωτήματα στα δεδομένα χρησιμοποιώντας ερωτήματα editor, να τα απεικονίσετε χρησιμοποιώντας πίνακες οργάνων, και να ειδοποιήσετε σχετικά με αυτά χρησιμοποιώντας τη λειτουργία ειδοποίησης.

Σχήμα 6.3: Grafana Logo

6.3.2 Χαρακτηριστικά του Grafana

Το Grafana αναπτύχθηκε ως εναλλακτική λύση στο Kibana (μέρος της στοίβας ELK) ως εργαλείο dashboarding με βάση χρονοσειρές που επικεντρώνεται σε γραφήματα. Σχεδιάστηκε πρωτίστως για να βλέπει καλά στα μονιτορ σε ένα δωμάτιο ελέγχου επιχειρήσεων με τη σκέψη ότι ένας χειριστής θα περάσει περισσότερο χρόνο κοιτάζοντάς το παρά ρυθμίζοντάς το. Όταν ξεκίνησε, συνήθιζε να υποστηρίζει InfluxDB και Graphite ως τις δύο μοναδικές πηγές δεδομένων. Τώρα, το Grafana είναι πιθανώς το μοναδικό εργαλείο που υποστηρίζει το συνδυασμό δεδομένων από πολλές διαφορετικές πηγές σε ένα ενιαίο ταμπλό. Χαρακτηριστικά του Grafana:

1. Οπτικοποίηση: Ο πρωταρχικός σκοπός της χρήσης του Grafana είναι η οπτικοποίηση των μετρήσεων είτε με τη μορφή γραφημάτων, heatmaps ή ιστογραμμάτων. Το Grafana παρέχει πολυάριθμες επιλογές για την οπτικοποίηση των δεδομένων σας με επιλογή θεμάτων και ζωνών ώρας.

2. Ενοποιήση: Μπορείτε να χρησιμοποιήσετε περισσότερες από 30 πηγές δεδομένων που υποστηρίζει το Grafana για να συνδυάσετε όλα τα δεδομένα που παράγει το περιβάλλον σας για να δημιουργήσετε ουσιαστικές ταμπλό από αυτά. Εκτός του ότι διαθέτει ενσωματωμένη υποστήριξη για το Graphite, InfluxDB, Prometheus, Elasticsearch κ.τ.λ., παρέχει ένα εκτεταμένο σύνολο πρόσθετων προγραμμάτων πηγών δεδομένων μέσω πρόσθετων προγραμμάτων.

3. Επεκτείνετε: Το Grafana διαθέτει ακμάζουσα υποστήριξη από την κοινότητα, η οποία σας βοηθάει να επεκτείνετε πέρα από τα έτοιμα χαρακτηριστικά σε άλλα εργαλεία τρίτων και πλατφόρμες

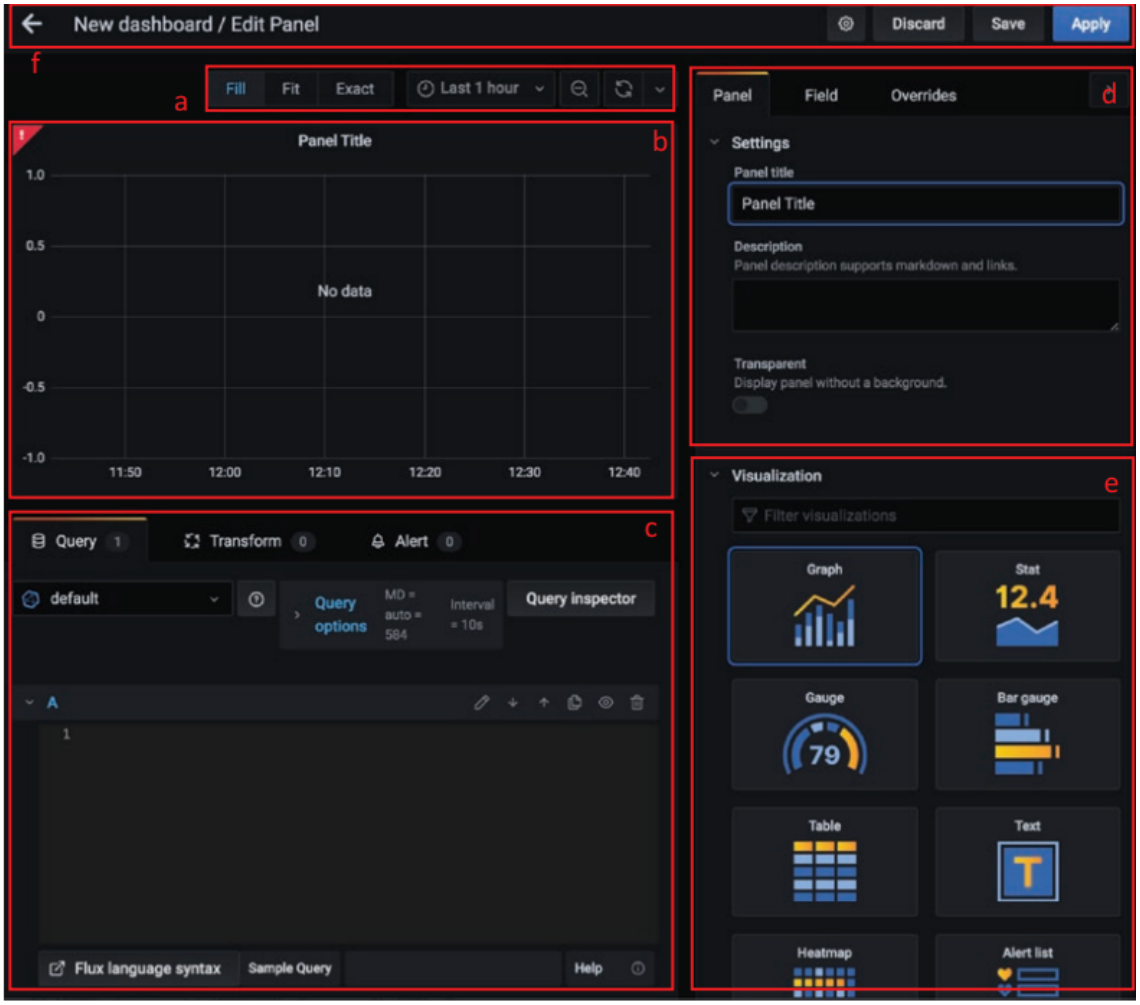
4. Συνεργασία: Μπορείτε να χρησιμοποιήσετε το σύνδεσμο για να μοιραστείτε τα ταμπλό Grafana με ομάδες ή να τις δημοσιεύσετε.

5. Ειδοποίηση: Ορίζοντας κατώτατα όρια ειδοποίησης, μπορείτε να ειδοποιείτε για κρίσιμες καταστάσεις με το Grafana.

Το Grafana δεν είναι διαθέσιμη μόνο ως λογισμικό ανοικτού κώδικα αλλά και ως Grafana Εντερπρισε και Grafana Cloud. Το Grafana Cloud είναι μια πλήρως διαχειρίσιμη περίπτωση Grafana που εκτελείται σε δημόσιο cloud και παρέχει μια εξαιρετικά διαθέσιμη και εξαιρετικά κλιμακούμενη επιλογή για την την ανάπτυξή σας μαζί με μακροπρόθεσμη διατήρηση. Εάν προτιμάτε να τρέξετε και να διαχειρίζεστε το Grafana μόνοι σας με πρόσβαση σε χαρακτηριστικά Enterprise-grade, τότε το Grafana Enterprise είναι η κατάλληλη επιλογή για εσάς.[15]

6.3.3 Panel Grafana

Μόλις προσθέσετε ένα πάνελ, θα μοιάζει με το σχέδιο 6.4.. Τα διάφορα τμήματα έχουν σημειωθούν κατάλληλα για την καλύτερη κατανόηση από το a έως το f.



Σχήμα 6.4: Grafana Panel

a) Προεπισκόπηση απεικόνισης
b) Τίτλος πίνακα, άξονες και υπόμνημα
c) Τμήμα δεδομένων
d) Ρυθμίσεις πίνακα, πεδίου και παρακάμψεων
e) Οπτική απεικόνιση
f) Επικεφαλίδα

6.4 Πλεονεκτήματα & Μειονέκτημα

6.4.1 Πλεονεκτήματα

Τα πλεονεκτήματα του Grafana περιλαμβάνουν:

1. Cloud agnostic: Οπτικοποιεί μετρήσεις από δύο κορυφαίους παρόχους cloud το AWS και το Microsoft Azure.

2. Επιλογές ενσωμάτωσης: Ενσωματώνεται με ένα ευρύ σύνολο πηγών δεδομένων, όπως το Προμετηευς, το InfluxDB, η MySQL, το Elasticsearch και το Splunk.

3. Ποικιλία οπτικών επιλογών: Προσφέρει διάφορες επιλογές πάνελ απεικόνισης που είναι πλούσιες σε χαρακτηριστικά, εξαιρετικά παραμετροποιήσιμες και προσαρμόσιμες ώστε να ανταποκρίνονται στις απαιτήσεις ενός οργανισμού.

6.4.2 Μειονέκτημα

Τα μειονεκτήματα του Grafana περιλαμβάνουν:

1. Περιορισμοί στην οργάνωση και τον σχεδιασμό του πίνακα οργάνων: Τα πάνελ οπτικοποίησης περιορίζονται σε αυτά που διατίθενται από την Grafana Labs και την κοινότητά της. Η οργάνωση και το φιλτράρισμα των πάνελ ενδέχεται να μην προσφέρουν αρκετή ευελιξία για το σχεδιασμό των ταμπλό, αλλά σε αυτά τα χαρακτηριστικά παρατηρούνται βελτιώσεις με τις νέες εκδόσεις εκδόσεων.

2. Δεν υπάρχει συλλογή και αποθήκευση δεδομένων: Το Grafana μπορεί να χρησιμοποιήσει δεδομένα μόνο από εξωτερικά συστήματα όπως το Prometheus, η MySQL, το Azure Monitor και το Amazon CloudWatch. Αυτό σημαίνει ότι το Grafana δεν έχει κανένα μέσο για να συλλέγει δεδομένα από μόνο του, μέσω πρακτόρων ή άλλων αγωγών δεδομένων, και έτσι εξαρτάται από άλλα συστήματα για την παροχή δεδομένων.

3. Περιορισμένες απεικονίσεις τύπου δεδομένων: Υποστηρίζει κυρίως δεδομένα χρονοσειρών, αλλά η κοινότητα ανοικτού κώδικα της Grafana δημιουργεί σταδιακά πρόσθετα(plugins) που επιτρέπουν στους χρήστες να απεικονίζουν οποιοδήποτε είδος δεδομένων. Για παράδειγμα, οι χρήστες μπορούν να δουν τα αποτελέσματα του Azure Resource Graph μέσω ενός πρόσθετου.

[18]

Κεφάλαιο 7

Hadoop Distributed File System (HDFS)

7.1 Εισαγωγή

Το Hadoop του Apache είναι μια εφαρμογή ανοικτού κώδικα του πλαισίου Map/Reduce της Google. Επιτρέπει κατανεμημένα, δεδομένα εντατικές και παράλληλες εφαρμογές, αποσυνθέτοντας μία τεράστια εργασία σε μικρότερες εργασίες και ένα τεράστιο σύνολο δεδομένων σε μικρότερες κατατμήσεις, έτσι ώστε κάθε εργασία να επεξεργάζεται μια διαφορετική κατάτμηση παράλληλα. Οι κύριες αφαιρέσεις είναι οι εξής:

1. Εργασίες MAP που επεξεργάζονται τις κατατμήσεις ενός συνόλου δεδομένων χρησιμοποιώντας ζεύγη κλειδιών/τιμών για τη δημιουργία ενός σύνολο ενδιάμεσων αποτελεσμάτων και

2. REDUCE εργασίες που συγχωνεύουν όλες τις ενδιάμεσες τιμές που σχετίζονται με το ίδιο ενδιάμεσο κλειδί.

Το Hadoop χρησιμοποιεί το κατανεμημένο σύστημα αρχείων Hadoop (HDFS) το οποίο είναι μια υλοποίηση του συστήματος αρχείων της Google (GFS) για την αποθήκευση δεδομένων. Το κατανεμημένο σύστημα αρχείων Hadoop (HDFS) είναι ένα κατανεμημένο σύστημα αρχείων που έχει σχεδιαστεί για να τρέχει σε κοινά προϊόντα υλικό. Παρόλο που υπάρχουν πολλές ομοιότητες με το υπάρχοντα κατανεμημένα συστήματα αρχείων, είναι πολύ διαφορετικά. Το HDFS έχει υψηλό βαθμό ανοχής σε σφάλματα και έχει αναπτυχθεί για να να αναπτυχθεί σε υλικό χαμηλού κόστους. Το HDFS παρέχει αποτελεσματική πρόσβαση σε δεδομένα εφαρμογών και είναι κατάλληλο για εφαρμογές με μεγάλα σύνολα δεδομένων.

Το HDFS cluster έχει αρχιτεκτονική master/slave με ένα μόνο Name Node ως κύριο διακομιστή που διαχειρίζεται το αρχείο χώρο ονομάτων του συστήματος και ρυθμίζει την πρόσβαση των πελατών στα αρχεία. Οι σκλάβοι είναι ένας αριθμός κόμβων δεδομένων, συνήθως ένας ανά κόμβο σε συστάδα, οι οποίοι διαχειρίζονται τον αποθηκευτικό χώρο που συνδέεται με τους κόμβους που τρέχουν. Το HDFS παρέχει ένα χώρο ονομάτων του συστήματος αρχείων και επιτρέπει την αποθήκευση των δεδομένων των χρηστών σε αρχεία. Ένα αρχείο χωρίζεται σε

ένα ή περισσότερα μπλοκ και αυτά τα μπλοκ αποθηκεύονται σε ένα σύνολο από Data κόμβων. Οι λειτουργίες του χώρου ονομάτων του συστήματος αρχείων, όπως το άνοιγμα, κλείσιμο και μετονομασία αρχείων και καταλόγων πραγματοποιούνται από το Name Node. Αποφασίζει επίσης την αντιστοίχιση των μπλοκ σε Data Nodes. Οι κόμβοι δεδομένων είναι υπεύθυνοι για την εξυπηρέτηση της ανάγνωσης και εγγραφής από τους πελάτες του συστήματος αρχείων, τη δημιουργία μπλοκ, αντιγραφής και διαγραφής κατόπιν οδηγιών από τον κόμβο Name Node.[5]

Σχήμα 7.1: HDFS Logo

7.2 Ιστορική Αναδρομή

Ο σχεδιασμός του HDFS βασίστηκε στο σύστημα αρχείων της Google. Αρχικά κατασκευάστηκε ως υποδομή για το έργο Apache Nutch για τη μηχανή αναζήτησης στο διαδίκτυο, αλλά έκτοτε έγινε μέλος του οικοσυστήματος Hadoop.

Στα πρώτα χρόνια του διαδικτύου, άρχισαν να εμφανίζονται οι μηχανές αναζήτησης ιστού ως ένας τρόπος για να αναζητούν οι άνθρωποι πληροφορίες σε ιστοσελίδες. Αυτό δημιούργησε διάφορες μηχανές αναζήτησης, όπως οι Yahoo και Google.

Δημιουργήθηκε επίσης μια άλλη μηχανή αναζήτησης με την ονομασία Nutch, η οποία ήθελε να διανέμει δεδομένα και υπολογισμούς σε πολλούς υπολογιστές ταυτόχρονα. Η Nutch στη συνέχεια μεταφέρθηκε στη Yahoo και χωρίστηκε σε δύο. Το Apache Spark και το Hadoop αποτελούν πλέον τις δικές τους ξεχωριστές οντότητες. Εκεί που το Hadoop έχει σχεδιαστεί για να χειρίζεται την επεξεργασία παρτίδων, το Spark έχει φτιαχτεί για να χειρίζεται αποτελεσματικά δεδομένα σε πραγματικό χρόνο.

Σήμερα, η δομή και το πλαίσιο του Hadoop διαχειρίζονται από το ίδρυμα λογισμικού Apache, το οποίο είναι μια παγκόσμια κοινότητα προγραμματιστών λογισμικού και συνεισφερόντων.

Το HDFS γεννήθηκε από αυτό και έχει σχεδιαστεί για να αντικαταστήσει τις λύσεις αποθήκευσης υλικού με μια καλύτερη, πιο αποτελεσματική μέθοδο - ένα εικονικό σύστημα αρχειοθέτησης. Όταν πρωτοεμφανίστηκε στη σκηνή, το MapReduce ήταν η μόνη μηχανή κατανεμημένης επεξεργασίας που μπορούσε να χρησιμοποιήσει το HDFS. Πιο πρόσφατα, τα

εναλλακτικά στοιχεία υπηρεσιών δεδομένων του Hadoop, όπως η HBase και η Solr, χρησιμοποιούν επίσης το HDFS για την αποθήκευση δεδομένων.

7.3 Αρχιτεκτονική

7.3.1 Εισαγωγή

Το HDFS (Hadoop Distributed File System) σχεδιάστηκε κυρίως για να λειτουργεί σε συσκευές μικρού κόστους και μικρών επιδόσεων, λειτουργώντας σε σχεδιασμό κατανεμημένου συστήματος αρχείων. Το HDFS έχει σχεδιαστεί με τέτοιο τρόπο ώστε να υποστηρίζει περισσότερο στην αποθήκευση των δεδομένων σε ένα μεγάλο κομμάτι μπλοκ παρά στην αποθήκευση μικρών μπλοκ δεδομένων.

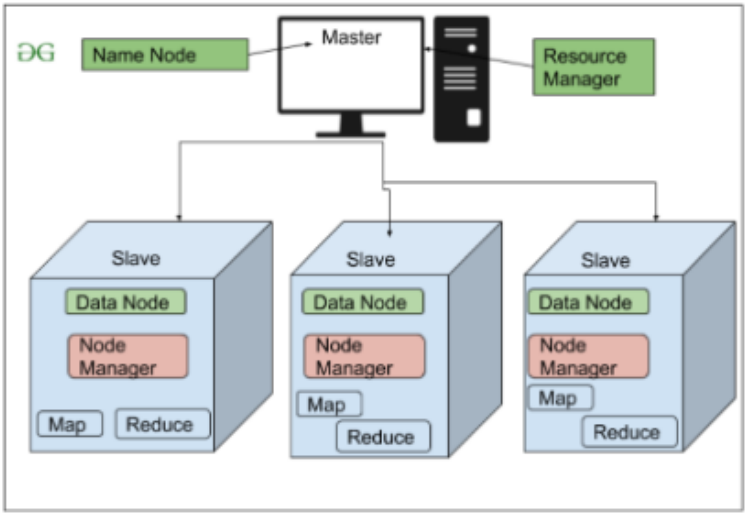
Το HDFS στο Hadoop παρέχει ανοχή σε σφάλματα και υψηλή διαθεσιμότητα στο επίπεδο αποθήκευσης και στις άλλες συσκευές που υπάρχουν σε αυτό το σύμπλεγμα Hadoop. Κόμβοι αποθήκευσης δεδομένων στο HDFS:

1. NameNode: Ο NameNode λειτουργεί ως Master σε μια συστάδα Hadoop που καθοδηγεί τους Datanode(Slaves). Ο Nameode χρησιμοποιείται κυρίως για την αποθήκευση των μεταδεδομένων, δηλαδή των δεδομένων σχετικά με τα δεδομένα. Τα μεταδεδομένα μπορεί να είναι τα αρχεία καταγραφής συναλλαγών που παρακολουθούν τη δραστηριότητα του χρήστη σε μια συστάδα Hadoop.

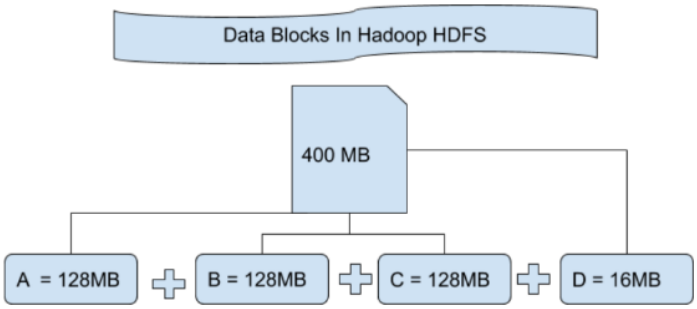
   Μεταδεδομένα μπορεί επίσης να είναι το όνομα του αρχείου, το μέγεθος και οι πληροφορίες σχετικά με τη θέση (αριθμός μπλοκ, block ids) του Datanode που αποθηκεύει ο Namenode για να βρει τον πλησιέστερο κόμβο δεδομένων για ταχύτερη επικοινωνία. Ο Namenode δίνει οδηγίες στους DataNodes με τις λειτουργίες όπως delete, create, Replicate, κ.λπ.

2. DataNode: Τα DataNodes λειτουργούν ως Slave. Τα DataNodes χρησιμοποιούνται κυρίως για την αποθήκευση των δεδομένων σε μια συστάδα Hadoop, ο αριθμός των DataNodes μπορεί να είναι από 1 έως 500 ή και περισσότερο από αυτό. Όσο μεγαλύτερος είναι ο αριθμός των DataNode, το σύμπλεγμα Hadoop θα μπορεί να αποθηκεύει περισσότερα δεδομένα. Επομένως, συνιστάται ο κόμβος δεδομένων να έχει υψηλή αποθηκευτική ικανότητα για την αποθήκευση μεγάλου αριθμού μπλοκ αρχείων.

7.3.2 Αρχιτεκτονική υψηλού επιπέδου του Hadoop



Μπλοκ αρχείων στο HDFS: Τα δεδομένα στο HDFS αποθηκεύονται πάντα σε μπλοκ. Έτσι, το ενιαίο μπλοκ δεδομένων χωρίζεται σε πολλαπλά μπλοκ μεγέθους 128MB, το οποίο είναι προεπιλεγμένο και μπορείτε επίσης να το αλλάξετε χειροκίνητα.



Ας κατανοήσουμε αυτή την έννοια της διάσπασης του αρχείου σε μπλοκ με ένα παράδειγμα. Ας υποθέσουμε ότι έχετε ανεβάσει ένα αρχείο 400MB στο HDFS σας, τότε αυτό που συμβαίνει είναι ότι αυτό το αρχείο χωρίζεται σε μπλοκ μεγέθους 128MB+128MB+128MB+16MB = 400MB. Αυτό σημαίνει ότι δημιουργούνται 4 μπλοκ το καθένα των 128MB εκτός από το τελευταίο. Το Hadoop δεν γνωρίζει ή δεν ενδιαφέρεται για το ποια δεδομένα είναι αποθηκευμένα σε αυτά τα μπλοκ, οπότε θεωρεί τα τελευταία μπλοκ του αρχείου ως ένα μερικό αρχείο, καθώς δεν έχει καμία ιδέα σχετικά με αυτό. Στο σύστημα αρχείων Linux, το μέγεθος ενός μπλοκ αρχείου είναι περίπου 4KB, το οποίο είναι πολύ μικρότερο από το προεπιλεγμένο μέγεθος των μπλοκ αρχείων στο σύστημα αρχείων Hadoop. Όπως όλοι γνωρίζουμε, το Hadoop έχει διαμορφωθεί κυρίως για την αποθήκευση δεδομένων μεγάλου μεγέθους που είναι σε petabyte, αυτό είναι που κάνει το σύστημα αρχείων Hadoop να διαφέρει από άλλα συστήματα αρχείων καθώς μπορεί να κλιμακωθεί, σήμερα τα μπλοκ αρχείων των 128MB έως 256MB θεωρούνται στο Hadoop.

Η αντιγραφή στο HDFS: Η αντιγραφή εξασφαλίζει τη διαθεσιμότητα των δεδομένων.

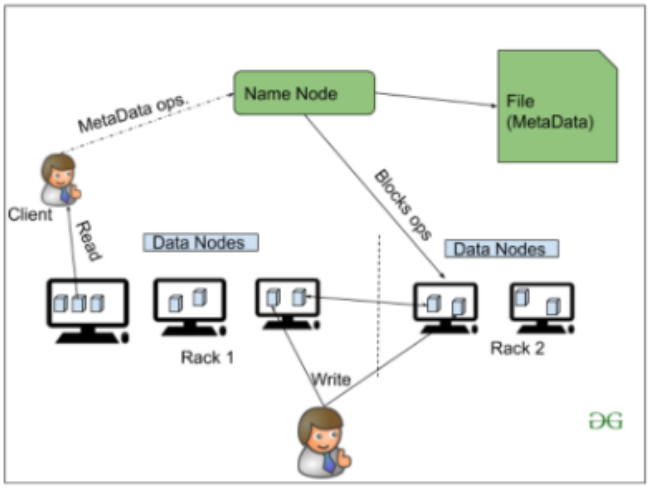
Η αντιγραφή είναι η δημιουργία ενός αντιγράφου από κάτι και ο αριθμός των φορών που κάνετε ένα αντίγραφο αυτού του συγκεκριμένου πράγματος μπορεί να εκφραστεί ως συντελεστής αντιγραφής. Όπως είδαμε στα μπλοκ αρχείων ότι το HDFS αποθηκεύει τα δεδομένα με τη μορφή διαφόρων μπλοκ, την ίδια στιγμή που το Hadoop είναι επίσης ρυθμισμένο να κάνει ένα αντίγραφο αυτών των μπλοκ αρχείων.

Από προεπιλογή, ο συντελεστής αντιγραφής για το Hadoop έχει οριστεί σε 3, ο οποίος μπορεί να ρυθμιστεί σημαίνει ότι μπορείτε να τον αλλάξετε χειροκίνητα σύμφωνα με τις απαιτήσεις σας, όπως στο παραπάνω παράδειγμα που έχουμε κάνει 4 μπλοκ αρχείων, πράγμα που σημαίνει ότι γίνονται 3 αντίγραφα κάθε μπλοκ αρχείων, δηλαδή συνολικά $4 \times 3 = 12$ μπλοκ για το σκοπό της δημιουργίας αντιγράφων ασφαλείας.

Αυτό οφείλεται στο γεγονός ότι για την εκτέλεση του Hadoop χρησιμοποιούμε commodity hardware (φθηνό υλικό συστήματος) το οποίο μπορεί να καταρρεύσει ανά πάσα στιγμή. Δεν χρησιμοποιούμε υπερυπολογιστή για την εγκατάσταση του Hadoop. Γι΄ αυτό χρειαζόμαστε ένα τέτοιο χαρακτηριστικό στο HDFS το οποίο μπορεί να δημιουργήσει αντίγραφα των εν λόγω μπλοκ αρχείων για σκοπούς δημιουργίας αντιγράφων ασφαλείας, αυτό είναι γνωστό ως ανοχή σφάλματος.

Τώρα, ένα πράγμα που πρέπει επίσης να παρατηρήσουμε είναι ότι μετά τη δημιουργία τόσων αντιγράφων των μπλοκ αρχείων μας σπαταλάμε τόσο πολύ από τον αποθηκευτικό μας χώρο, αλλά για τους μεγάλους οργανισμούς με εμπορικά σήματα τα δεδομένα είναι πολύ πιο σημαντικά από τον αποθηκευτικό χώρο, οπότε κανείς δεν ενδιαφέρεται για αυτόν τον επιπλέον αποθηκευτικό χώρο. Μπορείτε να ρυθμίσετε τον παράγοντα αντιγραφής στο αρχείο hdfs-site.xml.

Rack Awareness Το rack δεν είναι τίποτε άλλο παρά η φυσική συλλογή κόμβων στο σύμπλεγμα Hadoop μας (ίσως 30 έως 40). Μια μεγάλη συστάδα Hadoop αποτελείται από τόσα πολλά Racks . Με τη βοήθεια αυτής της πληροφορίας Racks το Namenode επιλέγει τον πλησιέστερο κόμβο Datanode για να επιτύχει τη μέγιστη απόδοση κατά την εκτέλεση των πληροφοριών ανάγνωσης/εγγραφής που μειώνει την κίνηση δικτύου. [9]



7.4 Πλεονεκτήματα & Μειονεκτήματα

7.4.1 Πλεονεκτήματα

Ως υποέργο ανοικτού κώδικα στο πλαίσιο του Hadoop, το HDFS προσφέρει πέντε βασικά πλεονεκτήματα κατά την επεξεργασία μεγάλων δεδομένων:

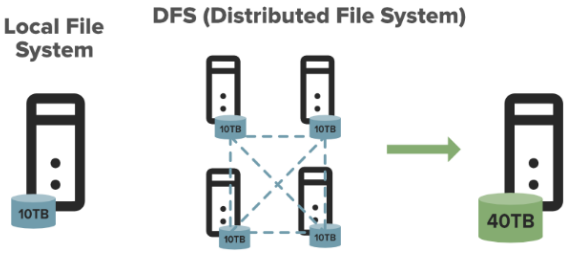
1. **Ανοχή σφαλμάτων** Το HDFS έχει σχεδιαστεί για να ανιχνεύει σφάλματα και να ανακάμπτει αυτόματα και γρήγορα, εξασφαλίζοντας συνέχεια και αξιοπιστία.

2. **Ταχύτητα**, λόγω της αρχιτεκτονικής του σε συστάδες, μπορεί να διατηρήσει 2 GB δεδομένων ανά δευτερόλεπτο.

3. **Πρόσβαση** σε περισσότερους τύπους δεδομένων, συγκεκριμένα σε δεδομένα ροής. Λόγω του σχεδιασμού του για τη διαχείριση μεγάλων ποσοτήτων δεδομένων για επεξεργασία παρτίδων, επιτρέπει υψηλούς ρυθμούς μετάδοσης δεδομένων, καθιστώντας το ιδανικό για την υποστήριξη δεδομένων ροής.

4. **Συμβατότητα και φορητότητα** Το HDFS έχει σχεδιαστεί ώστε να είναι φορητό σε μια ποικιλία ρυθμίσεων υλικού και συμβατό με διάφορα υποκείμενα λειτουργικά συστήματα παρέχοντας τελικά στους χρήστες την προαιρετικότητα να χρησιμοποιούν το HDFS με τη δική τους προσαρμοσμένη ρύθμιση. Αυτά τα πλεονεκτήματα είναι ιδιαίτερα σημαντικά όταν έχουμε να κάνουμε με μεγάλα δεδομένα και έγιναν δυνατά με τον ιδιαίτερο τρόπο με τον οποίο το HDFS χειρίζεται τα δεδομένα.

   Αυτό το γράφημα δείχνει τη διαφορά μεταξύ ενός τοπικού συστήματος αρχείων και του HDFS.



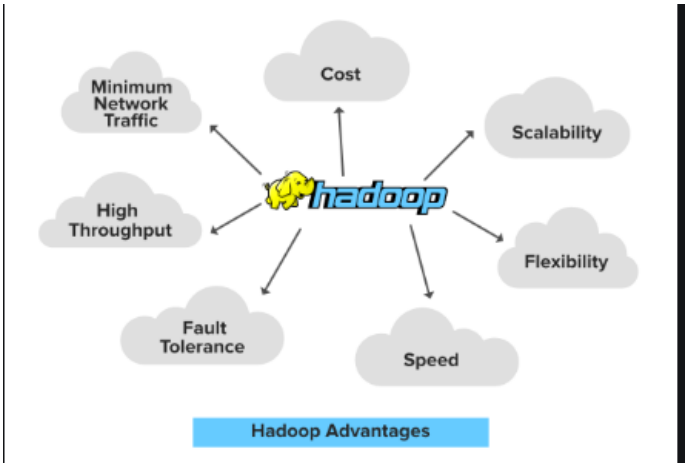
5. **Επεκτασιμότητα** Μπορείτε να κλιμακώσετε τους πόρους ανάλογα με το μέγεθος του συστήματος αρχείων σας. Το HDFS περιλαμβάνει μηχανισμούς κάθετης και οριζόντιας κλιμάκωσης.

6. **Εντοπισμός δεδομένων** Όταν πρόκειται για το σύστημα αρχείων Hadoop, τα δεδομένα βρίσκονται στους κόμβους δεδομένων, σε αντίθεση με το να μετακινούνται τα δεδομένα εκεί όπου βρίσκεται η υπολογιστική μονάδα. Μειώνοντας την απόσταση μεταξύ των δεδομένων και της υπολογιστικής διαδικασίας, μειώνεται η συμφόρηση του δικτύου και το σύστημα γίνεται πιο αποτελεσματικό και αποδοτικό.

7. **Οικονομικά αποδοτικό** Αρχικά, όταν σκεφτόμαστε τα δεδομένα μπορεί να σκεφτόμαστε ακριβό υλικό και καταληστευμένο εύρος ζώνης. Όταν παρουσιάζεται βλάβη στο υλικό, μπορεί να είναι πολύ δαπανηρή η επιδιόρθωσή της. Με το HDFS, τα δεδομένα αποθηκεύονται ανέξοδα, καθώς είναι εικονικά, γεγονός που μπορεί να μειώσει δραστικά το κόστος αποθήκευσης μεταδεδομένων και δεδομένων του συστήματος αρχείων και του χώρου ονομάτων του συστήματος αρχείων. Επιπλέον, επειδή το HDFS είναι ανοικτού κώδικα, οι επιχειρήσεις δεν χρειάζεται να ανησυχούν για την καταβολή τέλους αδειοδότησης.

8. **Αποθηκεύει μεγάλες ποσότητες δεδομένων** Η αποθήκευση δεδομένων είναι αυτό που αφορά το HDFS - δηλαδή δεδομένα όλων των ποικιλιών και μεγεθών - αλλά ιδιαίτερα μεγάλες ποσότητες δεδομένων από εταιρείες που δυσκολεύονται να τα αποθηκεύσουν. Αυτό περιλαμβάνει τόσο δομημένα όσο και αδόμητα δεδομένα.

9. **Ευέλικτο** Σε αντίθεση με ορισμένες άλλες πιο παραδοσιακές βάσεις δεδομένων αποθήκευσης, δεν υπάρχει ανάγκη επεξεργασίας των δεδομένων που συλλέγονται πριν από την αποθήκευση. Είστε σε θέση να αποθηκεύσετε όσα δεδομένα θέλετε, με τη δυνατότητα να αποφασίσετε τι ακριβώς θα θέλατε να κάνετε με αυτά και πώς θα τα χρησιμοποιήσετε αργότερα. Αυτό περιλαμβάνει επίσης μη δομημένα δεδομένα, όπως κείμενο, βίντεο και εικόνες.



7.4.2 Μειονεκτήματα

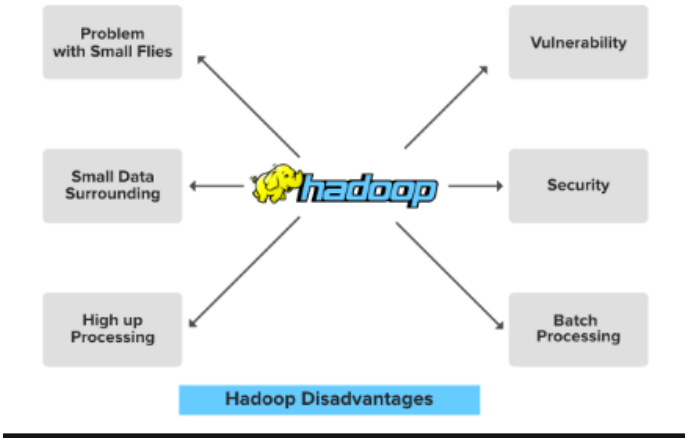
1. **Πρόβλημα με μικρά αρχεία** Το HDFS μπορεί να εκτελέσει αποτελεσματικά έναν μικρό αριθμό αρχείων μεγάλου μεγέθους. Το HDFS αποθηκεύει το αρχείο με τη μορφή μπλοκ αρχείων τα οποία έχουν μέγεθος από 128MB(από προεπιλογή) έως 256MB. Το HDFS αποτυγχάνει όταν χρειάζεται να προσπελάσει το αρχείο μικρού μεγέθους σε μεγάλο αριθμό. Αυτό το τόσο μεγάλο πλήθος μικρών αρχείων επιβαρύνει τον Namenode και δυσκολεύει την εργασία του.

2. **Ευπάθεια** Το HDFS είναι ένα πλαίσιο που είναι γραμμένο σε java και η java είναι μία από τις πιο συχνά χρησιμοποιούμενες γλώσσες προγραμματισμού, γεγονός που το καθιστά πιο ανασφαλές, καθώς μπορεί εύκολα να αξιοποιηθεί από οποιονδήποτε εγκληματία του κυβερνοχώρου.

3. **Έλλειψη ασφάλειας** Τα δεδομένα είναι τα πάντα για έναν οργανισμό, εξ ορισμού η λειτουργία ασφαλείας στο HDFS καθίσταται μη διαθέσιμη. Έτσι, ο οδηγός δεδομένων πρέπει να είναι προσεκτικός με αυτό το πρόσωπο ασφαλείας και θα πρέπει να λάβει τα κατάλληλα μέτρα σχετικά με αυτό. Το HDFS χρησιμοποιεί το Kerberos για το χαρακτηριστικό ασφαλείας το οποίο δεν είναι εύκολο να διαχειριστεί. Η κρυπτογράφηση αποθήκευσης και δικτύου λείπει από το Kerberos, γεγονός που μας κάνει να ανησυχούμε περισσότερο γι΄ αυτό.

4. **Υψηλή επεξεργασία** Η λειτουργία ανάγνωσης/εγγραφής στο HDFS είναι υπέρμετρη αφού έχουμε να κάνουμε με δεδομένα μεγάλου μεγέθους που είναι σε TB ή PB. Στο Hadoop, η ανάγνωση ή η εγγραφή δεδομένων γίνεται από το δίσκο, γεγονός που καθιστά δύσκολη την εκτέλεση υπολογισμών στη μνήμη και οδηγεί σε επιβάρυνση επεξεργασίας ή High up processing.

5. **Υποστηρίζει μόνο την επεξεργασία παρτίδων** Η διεργασία δέσμης δεν είναι παρά οι διεργασίες που εκτελούνται στο παρασκήνιο και δεν έχουν κανενός είδους αλληλεπίδραση με τον χρήστη. Οι μηχανές που χρησιμοποιούνται για αυτές τις διεργασίες μέσα στον πυρήνα του HDFS δεν είναι τόσο αποδοτικές. Η παραγωγή της εξόδου με χαμηλή καθυστέρηση δεν είναι δυνατή με αυτήν.



[10]

Κεφάλαιο 8

Machine Learning(ML)

8.1 Εισαγωγή

Η Μηχανική Μάθηση (Machine Learning - ML) είναι ένα πεδίο έρευνας που ασχολείται με την κατανόηση και την κατασκευή μεθόδων που ¨μαθαίνουν¨, δηλαδή μεθόδων που αξιοποιούν δεδομένα για να βελτιώσουν την απόδοση σε κάποιο σύνολο εργασιών. Θεωρείται μέρος της τεχνητής νοημοσύνης. Οι αλγόριθμοι μηχανικής μάθησης δημιουργούν ένα μοντέλο με βάση δειγματικά δεδομένα, γνωστά ως δεδομένα εκπαίδευσης, προκειμένου να κάνουν προβλέψεις ή να λαμβάνουν αποφάσεις χωρίς να είναι ρητά προγραμματισμένοι να το κάνουν. Οι αλγόριθμοι μηχανικής μάθησης χρησιμοποιούνται σε μια ευρεία ποικιλία εφαρμογών, όπως στην ιατρική, στο φιλτράρισμα ηλεκτρονικού ταχυδρομείου, στην αναγνώριση ομιλίας και στην όραση υπολογιστών, όπου είναι δύσκολο ή ανέφικτο να αναπτυχθούν συμβατικοί αλγόριθμοι για την εκτέλεση των απαιτούμενων εργασιών. Ένα υποσύνολο της μηχανικής μάθησης σχετίζεται στενά με την υπολογιστική στατιστική, η οποία επικεντρώνεται στην πραγματοποίηση προβλέψεων με τη χρήση υπολογιστών, αλλά δεν είναι όλη η μηχανική μάθηση στατιστική μάθηση. Η μελέτη της μαθηματικής βελτιστοποίησης παρέχει μεθόδους, θεωρία και τομείς εφαρμογής στον τομέα της μηχανικής μάθησης. Η εξόρυξη δεδομένων είναι ένα συναφές πεδίο μελέτης, που επικεντρώνεται στη διερευνητική ανάλυση δεδομένων μέσω μη επιβλεπόμενης μάθησης. Ορισμένες εφαρμογές της μηχανικής μάθησης χρησιμοποιούν δεδομένα και νευρωνικά δίκτυα με τρόπο που μιμείται τη λειτουργία ενός βιολογικού εγκεφάλου. Στην εφαρμογή της σε επιχειρηματικά προβλήματα, η μηχανική μάθηση αναφέρεται επίσης ως προγνωστική ανάλυση. [21]

8.2 Διαδικασίες Μάθησης

8.2.1 Εισαγωγή

Τα συστήματα με ευφυΐα, ακριβώς όπως και οι άνθρωποι, έχουν ποικίλα τρόπους μάθησης. Οι μάθηση αύτη μπορούμε να την κατηγοριοποιήσουμε σε δύο μεγάλες κατηγορίες,επιβλεπόμενη μάθηση δηλαδή με κάποιον καθοδηγητή και μη επιβλεπόμενη μάθηση. Στην παρούσα ενότητα

θα αναλύσουμε μόνο την επιβλεπόμενη μάθηση που χρησιμοποιείται κατα κόρων στα συστήματα μηχανικής μάθησης.

8.2.2 Επιβλεπόμενη Μάθηση

Η επιβλεπόμενη μάθηση, επίσης γνωστή ως επιβλεπόμενη μηχανική μάθηση, είναι μια υποκατηγορία της μηχανικής μάθησης και της τεχνητής νοημοσύνης. Ορίζεται από τη χρήση επισημειωμένων συνόλων δεδομένων για την εκπαίδευση αλγορίθμων που ταξινομούν δεδομένα ή προβλέπουν αποτελέσματα με ακρίβεια. Καθώς τα δεδομένα εισόδου τροφοδοτούνται στο μοντέλο, αυτό προσαρμόζει τα βάρη του έως ότου το μοντέλο προσαρμοστεί κατάλληλα, κάτι που συμβαίνει στο πλαίσιο της διαδικασίας διασταυρούμενης επικύρωσης. Η επιβλεπόμενη μάθηση βοηθά τους οργανισμούς να επιλύουν μια ποικιλία προβλημάτων του πραγματικού κόσμου σε κλίμακα, όπως η ταξινόμηση των ανεπιθύμητων μηνυμάτων σε ξεχωριστό φάκελο από τα εισερχόμενά σας.

Η μάθηση με επίβλεψη χρησιμοποιεί ένα σύνολο εκπαίδευσης για να διδάξει τα μοντέλα να αποδίδουν την επιθυμητή έξοδο. Αυτό το σύνολο δεδομένων εκπαίδευσης περιλαμβάνει εισόδους και σωστές εξόδους, οι οποίες επιτρέπουν στο μοντέλο να μαθαίνει με την πάροδο του χρόνου. Ο αλγόριθμος μετράει την ακρίβειά του μέσω της συνάρτησης απωλειών, προσαρμόζοντας μέχρι να ελαχιστοποιηθεί επαρκώς το σφάλμα. Η επιβλεπόμενη μάθηση μπορεί να διαχωριστεί σε δύο τύπους προβλημάτων κατά την εξόρυξη δεδομένων - ταξινόμηση και παλινδρόμηση:

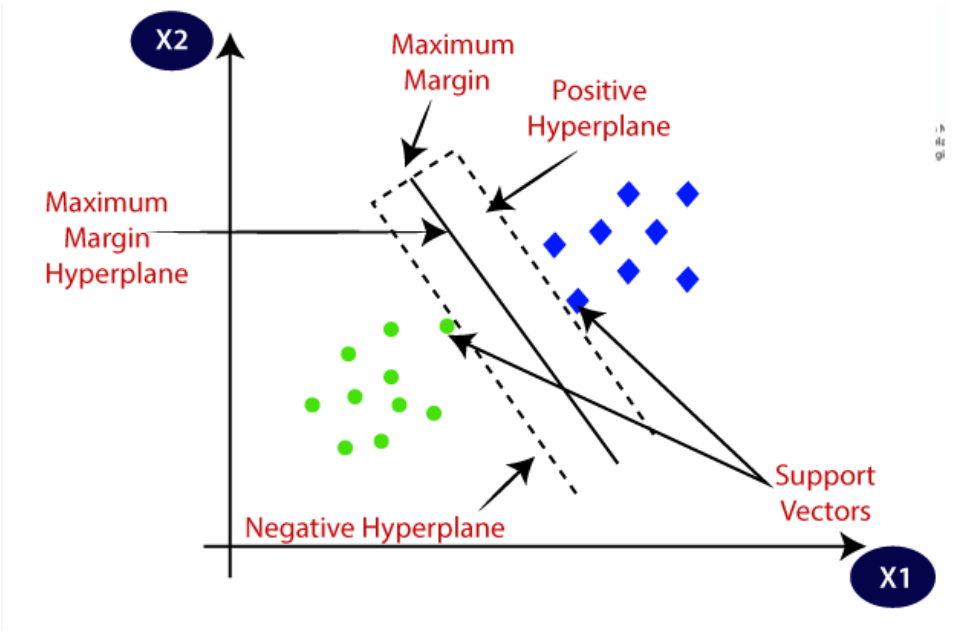
1. Η ταξινόμηση (classification) χρησιμοποιεί έναν αλγόριθμο για την ακριβή ανάθεση δεδομένων δοκιμής σε συγκεκριμένες κατηγορίες. Αναγνωρίζει συγκεκριμένες οντότητες μέσα στο σύνολο δεδομένων και προσπαθεί να βγάλει κάποια συμπεράσματα σχετικά με τον τρόπο με τον οποίο αυτές οι οντότητες θα πρέπει να επισημανθούν ή να οριστούν. Οι συνήθεις αλγόριθμοι ταξινόμησης είναι οι γραμμικοί ταξινομητές, οι μηχανές διανυσμάτων υποστήριξης (SVM), τα δέντρα απόφασης, ο k-near γείτονας και το τυχαίο δάσος, οι οποίοι περιγράφονται λεπτομερέστερα παρακάτω.

2. Η παλινδρόμηση (regression) χρησιμοποιείται για την κατανόηση της σχέσης μεταξύ εξαρτημένων και ανεξάρτητων μεταβλητών. Χρησιμοποιείται συνήθως για την πραγματοποίηση προβλέψεων, όπως για τα έσοδα από τις πωλήσεις μιας συγκεκριμένης επιχείρησης. Η γραμμική παλινδρόμηση, η λογιστική παλινδρόμηση και η πολυωνυμική παλινδρόμηση είναι δημοφιλείς αλγόριθμοι παλινδρόμησης.

8.3 Support Vector Machine Algorithm

Η Μηχανή Διανυσμάτων Υποστήριξης ή SVM είναι ένας από τους πιο δημοφιλείς αλγορίθμους επιβλεπόμενης μάθησης, ο οποίος χρησιμοποιείται για προβλήματα ταξινόμησης και παλινδρόμησης. Ωστόσο, κατά κύριο λόγο, χρησιμοποιείται για προβλήματα Ταξινόμησης στη Μηχανική Μάθηση. Ο στόχος του αλγορίθμου SVM είναι να δημιουργήσει την καλύτερη γραμμή ή όριο απόφασης που μπορεί να διαχωρίσει τον n-διάστατο χώρο σε κλάσεις, ώστε

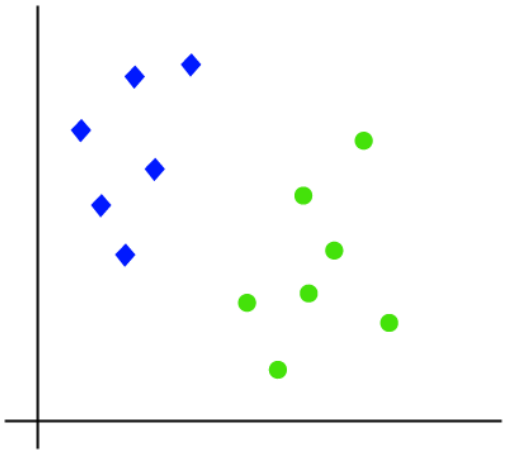
να μπορούμε εύκολα να τοποθετήσουμε το νέο σημείο δεδομένων στη σωστή κατηγορία στο μέλλον. Αυτό το καλύτερο όριο απόφασης ονομάζεται υπερεπίπεδο.Το SVM επιλέγει τα ακραία σημεία/διανύσματα που βοηθούν στη δημιουργία του υπερεπιπέδου. Αυτές οι ακραίες περιπτώσεις ονομάζονται διανύσματα υποστήριξης και ως εκ τούτου ο αλγόριθμος ονομάζεται μηχανή διανύσματος υποστήριξης. Σκεφτείτε το παρακάτω διάγραμμα στο οποίο υπάρχουν δύο διαφορετικές κατηγορίες που ταξινομούνται με τη χρήση ενός ορίου απόφασης ή υπερεπιπέδου:



Σχήμα 8.1: Διαχωρισμός δυο κλάσεων με SVM

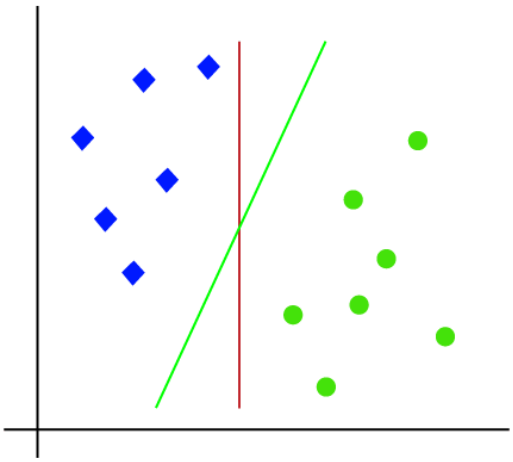
8.3.1 Linear SVM

Η λειτουργία του αλγορίθμου SVM μπορεί να γίνει κατανοητή με τη χρήση ενός παραδείγματος. Ας υποθέσουμε ότι έχουμε ένα σύνολο δεδομένων που έχει δύο ετικέτες (πράσινο και μπλε) και το σύνολο δεδομένων έχει δύο χαρακτηριστικά x1 και x2. Θέλουμε έναν ταξινομητή που να μπορεί να ταξινομήσει το ζεύγος(x1, x2) των συντεταγμένων είτε στο πράσινο είτε στο μπλε. Σκεφτείτε την παρακάτω εικόνα:



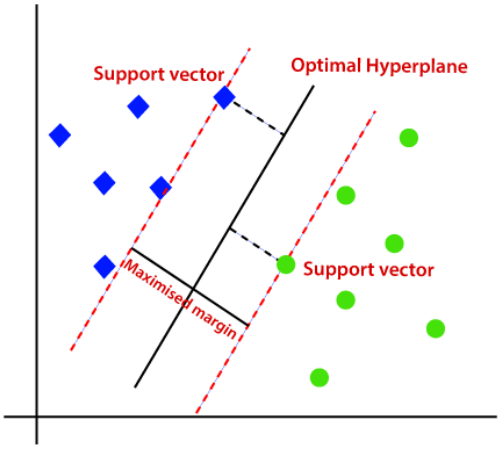
Σχήμα 8.2: Παράδειγμα για την κατανόηση SVM

Καθώς λοιπόν πρόκειται για δισδιάστατο χώρο, χρησιμοποιώντας απλώς μια ευθεία γραμμή, μπορούμε εύκολα να διαχωρίσουμε αυτές τις δύο κατηγορίες. Αλλά μπορεί να υπάρχουν πολλαπλές γραμμές που μπορούν να διαχωρίσουν αυτές τις κλάσεις. Σκεφτείτε την παρακάτω εικόνα:



Σχήμα 8.3: Παράδειγμα για την κατανόηση SVM

Ως εκ τούτου, ο αλγόριθμος SVM βοηθά στην εύρεση της καλύτερης γραμμής ή του καλύτερου ορίου απόφασης- αυτό το καλύτερο όριο ή περιοχή ονομάζεται υπερεπίπεδο. Ο αλγόριθμος SVM βρίσκει το πλησιέστερο σημείο των γραμμών και από τις δύο κλάσεις. Τα σημεία αυτά ονομάζονται διανύσματα υποστήριξης. Η απόσταση μεταξύ των διανυσμάτων και του υπερεπιπέδου ονομάζεται περιθώριο. Και ο στόχος του SVM είναι η μεγιστοποίηση αυτού του περιθωρίου. Το υπερεπίπεδο με το μέγιστο περιθώριο ονομάζεται βέλτιστο υπερεπίπεδο.



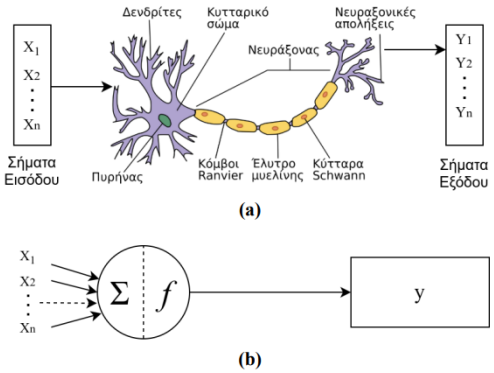
Σχήμα 8.4: Παράδειγμα για την κατανόηση SVM

[14]

8.4 Νευρωνικά Δίκτυα

8.4.1 Εισαγωγή

Ένα νευρωνικό δίκτυο είναι ένα δίκτυο ή κύκλωμα βιολογικών νευρώνων ή, με τη σύγχρονη έννοια, ένα τεχνητό νευρωνικό δίκτυο, που αποτελείται από τεχνητούς νευρώνες ή κόμβους. Έτσι, ένα νευρωνικό δίκτυο είναι είτε ένα βιολογικό νευρωνικό δίκτυο, που αποτελείται από βιολογικούς νευρώνες, είτε ένα τεχνητό νευρωνικό δίκτυο, που χρησιμοποιείται για την επίλυση προβλημάτων τεχνητής νοημοσύνης (AI). Οι συνδέσεις του βιολογικού νευρώνα μοντελοποιούνται στα τεχνητά νευρωνικά δίκτυα ως βάρη μεταξύ των κόμβων. Ένα θετικό βάρος αντικατοπτρίζει μια διεγερτική σύνδεση, ενώ αρνητικές τιμές σημαίνουν ανασταλτικές συνδέσεις. Όλες οι είσοδοι τροποποιούνται με ένα βάρος και αθροίζονται. Αυτή η δραστηριότητα αναφέρεται ως γραμμικός συνδυασμός. Τέλος, μια συνάρτηση ενεργοποίησης ελέγχει το πλάτος της εξόδου. Για παράδειγμα, ένα αποδεκτό εύρος εξόδου είναι συνήθως μεταξύ 0 και 1 ή θα μπορούσε να είναι -1 και 1. Αυτά τα τεχνητά δίκτυα μπορούν να χρησιμοποιηθούν για προγνωστική μοντελοποίηση, προσαρμοστικό έλεγχο και εφαρμογές όπου μπορούν να εκπαιδευτούν μέσω ενός συνόλου δεδομένων. Η αυτομάθηση που προκύπτει από την εμπειρία μπορεί να συμβεί μέσα στα δίκτυα, τα οποία μπορούν να εξάγουν συμπεράσματα από ένα πολύπλοκο και φαινομενικά άσχετο σύνολο πληροφοριών.



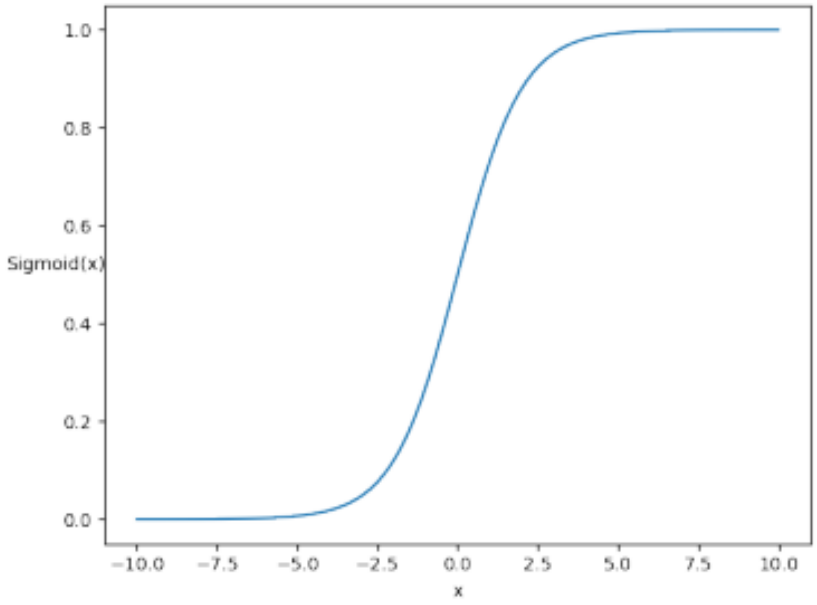
Σχήμα 8.5: Η δομή ενός φυσικού νευρώνα του ανθρώπινου εγκεφάλου (α) σε σύγκριση με την μαθηματική δομή του τεχνητού νευρώνα (β) ο οποίος χρησιμοποιείται στα νευρωνικά δίκτυα.

8.4.2 Συναρτήσεις ενεργοποίησης

Στα τεχνητά νευρωνικά δίκτυα, η συνάρτηση ενεργοποίησης ενός κόμβου καθορίζει την έξοδο του εν λόγω κόμβου δεδομένης μιας εισόδου ή ενός συνόλου εισόδων. Ένα τυπικό ολοκληρωμένο κύκλωμα μπορεί να θεωρηθεί ως ένα ψηφιακό δίκτυο συναρτήσεων ενεργοποίησης που μπορεί να είναι 'ON' (1) ή 'OFF' (0), ανάλογα με την είσοδο. Αυτό είναι παρόμοιο με το γραμμικό perceptron στα νευρωνικά δίκτυα. Ωστόσο, μόνο οι μη γραμμικές συναρτήσεις ενεργοποίησης επιτρέπουν στα δίκτυα αυτά να υπολογίζουν μη τετριμμένα προβλήματα χρησιμοποιώντας μόνο ένα μικρό αριθμό κόμβων, και οι συναρτήσεις ενεργοποίησης αυτές ονομάζονται μη γραμμικές. Οι πιο δημοφιλείς οικογένειες συναρτήσεων ενεργοποίησης είναι οι εξής:

Sigmoid or Logistic

Η σιγμοειδής συνάρτηση είναι μια μαθηματική συνάρτηση που έχει χαρακτηριστική καμπύλη σχήματος 'S' ή σιγμοειδή καμπύλη.Ένα συνηθισμένο παράδειγμα σιγμοειδούς συνάρτησης είναι η λογιστική συνάρτηση που φαίνεται στο πρώτο σχήμα και ορίζεται από τον τύπο ορίζονται ως $f(x) = \frac{1}{1+e^{-x}}$ και φράσουν την είσοδο στο διάστημα [0,1].



Σχήμα 8.6: Γραφική παράσταση της Σιγμοειδούς συνάρτησης.

Softmax

Η συνάρτηση softmax είναι μια συνάρτηση που μετατρέπει ένα διάνυσμα από K πραγματικές τιμές σε ένα διάνυσμα από K πραγματικές τιμές που έχουν άθροισμα 1. Οι τιμές εισόδου μπορεί να είναι θετικές, αρνητικές, μηδενικές ή μεγαλύτερες του 1, αλλά η softmax τις μετατρέπει σε τιμές μεταξύ 0 και 1, ώστε να μπορούν να ερμηνευθούν ως πιθανότητες. Εάν μια από τις εισόδους είναι μικρή ή αρνητική, το softmax τη μετατρέπει σε μικρή πιθανότητα, και εάν μια είσοδος είναι μεγάλη, τότε τη μετατρέπει σε μεγάλη πιθανότητα, αλλά θα παραμένει πάντα μεταξύ 0 και 1.

Η συνάρτηση softmax ονομάζεται μερικές φορές συνάρτηση softargmax ή λογιστική παλινδρόμηση πολλαπλών κλάσεων. Αυτό συμβαίνει επειδή η softmax είναι μια γενίκευση της λογιστικής παλινδρόμησης που μπορεί να χρησιμοποιηθεί για ταξινόμηση πολλαπλών κλάσεων και ο τύπος της μοιάζει πολύ με τη σιγμοειδή συνάρτηση που χρησιμοποιείται για τη λογιστική παλινδρόμηση. Η συνάρτηση softmax μπορεί να χρησιμοποιηθεί σε έναν ταξινομητή μόνο όταν οι κλάσεις είναι αμοιβαία αποκλειόμενες. Η συνάρτηση Softmax ορίζεται μαθηματικά από τον παρακάτω τύπο:

$$softmax(z) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

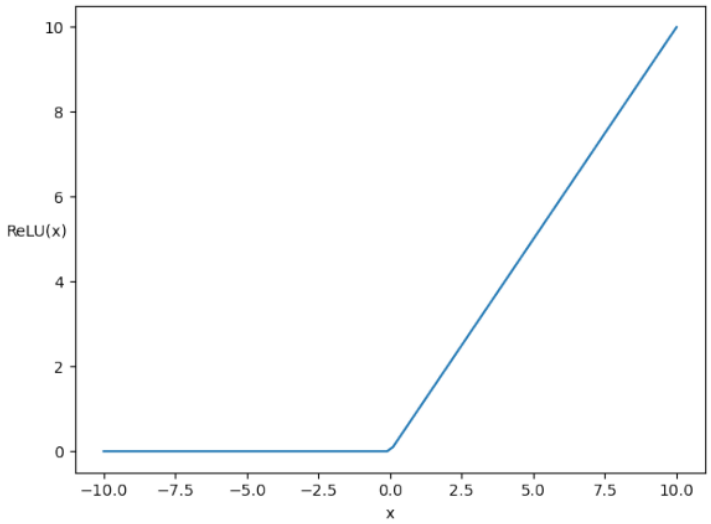
για i = 1, . . . , K και z = z_1, . . . , z_K [3]

ReLU

Η Rectifier Linear Unit (ReLU) ορίζεται μαθηματικά ως

$$ReLU(x) = max(0, x)$$

Είναι μια σχετικά άπλη συνάρτηση που παρολά αυτά έχει μικρό χρόνο μάθησης. Με συνέπεια, στις μέρες μας χρησιμοποιείται λόγω της μικρής πολυπλοκότητας αλλά και της τεράστιας αποτελεσματικότητας.

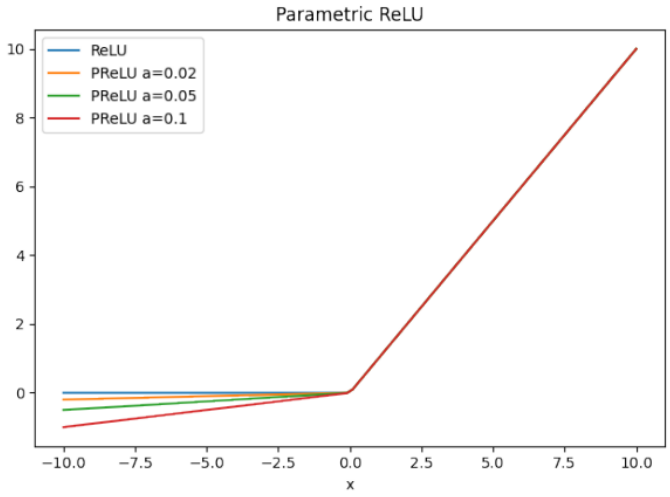


Σχήμα 8.7: Γραφική παράσταση της συνάρτησης ReLU.

Η ReLU βρίσκεται σε μια ομάδα συναρτήσεων, οι οποίες διαφέρουν ως προς την λειτουργία στις αρνητικές τιμές εισόδου. Για παράδειγμα, πολλοί ερευνητές δεν δεχόντουσαν οτι το μη θετικό μέρος των σημάτων, πρακτικά εξαφανίζετε. Για αύτο τον λόγο, χρησιμοποιούνται οι Leaky συναρτήσεις που μείωνουν αντί να διαγράφουν εξ ολοκλήρου την αρνητική συνιστώσα. Μάλιστα, ο βαθμός μείωσης μπορεί να είναι και παράμετρος του δικτύου, όπως η μεταβλητή α στην Parametric ReLU

$$PReLU(x) = max(0, x) + min(a * x, 0)$$

.



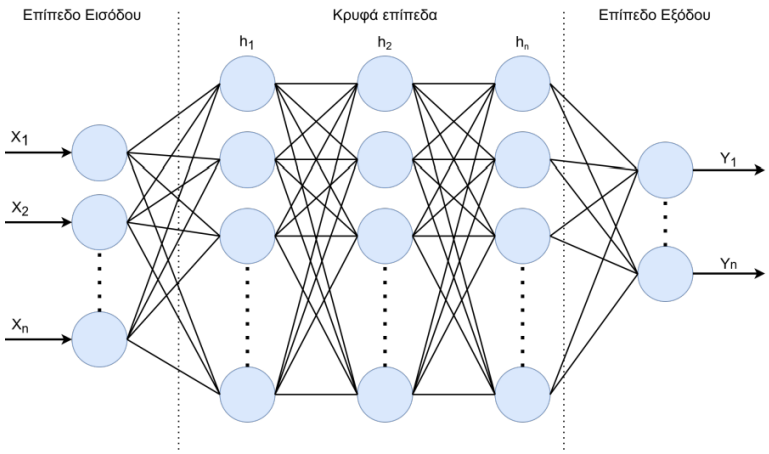
Σχήμα 8.8: Γραφική παράσταση της συνάρτησης ReLU σε σύγκριση με διάφορες περιπτώσεις της παραμετρικής ReLU ανάλογα με την τιμή της παραμέτρου α.

8.4.3 Αρχιτεκτονικές Νευρωνικών Δικτύων

Όπως συμβαίνει με τα πραγματικά νευρωνικά κυκλώματα στον εγκέφαλο, οι αρχιτεκτονικές των τεχνητών νευρωνικών δικτύων συχνά περιγράφονται ως εμπρόσθιες ή αναδρομικές. Τα feedforward νευρωνικά δίκτυα επεξεργάζονται σήματα με μονόδρομη κατεύθυνση και δεν έχουν εγγενή χρονική δυναμική. Έτσι, συχνά περιγράφονται ως στατικά. Αντίθετα, τα αναδρομικά δίκτυα έχουν βρόχους και μπορούν να θεωρηθούν ως ένα δυναμικό σύστημα του οποίου η κατάσταση διατρέχει έναν χώρο καταστάσεων και διαθέτει σταθερές και ασταθείς ισορροπίες.

1. **Δίκτυα πρόσθιας τροφοδότησης (feedforward neural network):**

   Το Νευρωνικό Δίκτυο Προώθησης είναι ένα τεχνητό νευρωνικό δίκτυο στο οποίο οι συνδέσεις μεταξύ των κόμβων δεν σχηματίζουν κύκλο. Το αντίθετο του νευρωνικού δικτύου feed forward είναι το επαναλαμβανόμενο νευρωνικό δίκτυο, στο οποίο ορισμένες διαδρομές ανακυκλώνονται. Το μοντέλο feed forward είναι η απλούστερη μορφή νευρωνικού δικτύου, καθώς οι πληροφορίες επεξεργάζονται μόνο προς μία κατεύθυνση. Ενώ τα δεδομένα μπορεί να περάσουν από πολλούς κρυμμένους κόμβους, κινούνται πάντα προς μία κατεύθυνση και ποτέ προς τα πίσω.



Σχήμα 8.9: Η δομή ενός πλήρους συνδεδεμένου νευρωνικού δικτύου.

Το "κρυφό" του σχήματος 8.9 αναφέρεται στο γεγονός ότι αυτό το μέρος του νευρωνικού δικτύου, δεν είναι ορατό από την είσοδο και την έξοδο του δικτύου. Η ύπαρξή του, είναι να παράγουν πολύτιμα ενδιάμεσα αποτελέσματα κάνοντας υπολογισμούς, βοηθώντας το δίκτυο να παράγει μεγαλύτερης τάξης πληροφορία από τα δεδομένα που εισέλθανε. Ένα νευρωνικό δίκτυο λέγεται πλήρως συνδεδεμένο, όταν κάθε κόμβος σε κάθε επίπεδο, συνδέεται με κάθε άλλο κόμβο του επόμενου επιπέδου.

2. **Αναδρομικά δίκτυα (recurrent neural network):**

Ένα επαναλαμβανόμενο νευρωνικό δίκτυο (RNN) είναι ένας τύπος τεχνητού νευρωνικού δικτύου που χρησιμοποιεί διαδοχικά δεδομένα ή δεδομένα χρονοσειράς. Αυτοί οι αλγόριθμοι βαθιάς μάθησης χρησιμοποιούνται συνήθως για τακτικές ή χρονικές καταστάσεις, όπως η μετάφραση γλωσσών, η επεξεργασία φυσικής γλώσσας (nlp), η αναγνώριση ομιλίας και η απόδοση λεζάντας σε εικόνες- ενσωματώνονται σε δημοφιλείς εφαρμογές όπως το Siri, η φωνητική αναζήτηση και το Google Translate. Όπως τα νευρωνικά δίκτυα τροφοδότησης και τα νευρωνικά δίκτυα συνελίξεων (CNN), τα επαναλαμβανόμενα νευρωνικά δίκτυα χρησιμοποιούν δεδομένα εκπαίδευσης για να μάθουν. Διακρίνονται από τη ¨μνήμη' τους, καθώς λαμβάνουν πληροφορίες από προηγούμενες εισόδους για να επηρεάσουν την τρέχουσα είσοδο και έξοδο. Ενώ τα παραδοσιακά βαθιά νευρωνικά δίκτυα υποθέτουν ότι οι είσοδοι και οι έξοδοι είναι ανεξάρτητες μεταξύ τους, η έξοδος των αναδρομικών νευρωνικών δικτύων εξαρτάται από τα προηγούμενα στοιχεία εντός της ακολουθίας. Ενώ τα μελλοντικά γεγονότα θα ήταν επίσης χρήσιμα για τον καθορισμό της εξόδου μιας δεδομένης ακολουθίας, τα μονόδρομα αναδρομικά νευρωνικά δίκτυα δεν μπορούν να λάβουν υπόψη αυτά τα γεγονότα στις προβλέψεις τους.

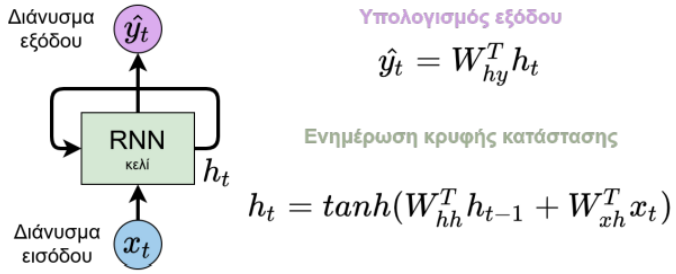
Τα αναδρομικά αύτα δίκτυα εκφράζονται μαθηματικά από την σχέση

$$h_t = f_W(h_{t-1}, x_t)$$

η οποία αντιστοιχεί στις πράξεις που γίνονται σε έναν κλασικό τεχνικό νευρώνα. Πολλαπλασιάζουμε τις εισόδους με έναν πίνακα βαρών W και περνάμε το αποτέλεσμα από

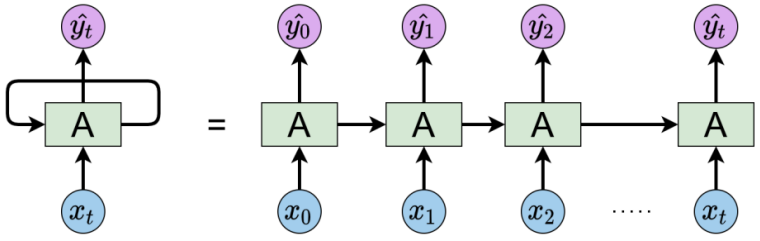
μια μη-γραμμικότητα.

Η κύρια διαφορά είναι στο ότι αφού έχουμε δύο εισόδους, την παλαιά κατάσταση και την τωρινή είσοδο, χρειαζόμαστε έναν πίνακα βαρών για την κάθε μία, όπως παρουσιάζεται και στο Σχ. 8.7. Η τελική έξοδος του κελιού, μπορεί να υπολογιστεί από την ανανεωμένη κατάσταση, πολλαπλασιάζοντάς την με έναν ξεχωριστό πίνακα βαρών.



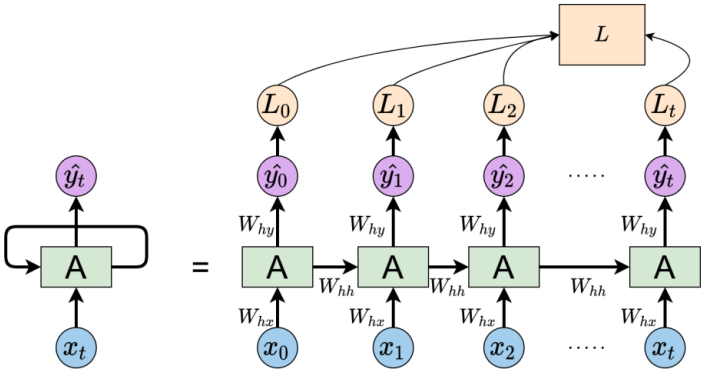
Σχήμα 8.10: Μαθηματικές σχέσεις ενός κελιού RNN.

Το τεράστιο κέρδος των αναδρομικών δικτύων, όταν έχουμε να κάνουμε με χρονοσειρές διαδοχικών δεδομένων φαίνεται στο παρακάτω σχήμα, όπου έχουμε αναπαραστήσει όλες τις καταστάσεις του δικτύου διαδοχικά.



Σχήμα 8.11: Το αναδρομικό δίκτυο, "ξετυλιγμένο" στο χρόνο..

Κάθε κελί μεταφέρει πληροφορία στο επόμενο (μέσω του υπολογισμού του h_t). Όπως βλέπουμε στο Σχ. 8.9, σε κάθε στιγμή οι πίνακες βαρών W παραμένουν ίδιοι και υπολογίζονται διαφορετικές έξοδοι για κάθε είσοδο. Για κάθε τιμή εξόδου, υπολογίζεται μια τιμή συνάρτησης κόστους και για τον υπολογισμό του συνολικού κόστους, προστίθονται οι επιμέρους τιμές



Σχήμα 8.12: Γραφική απεικόνιση της εμπρόσθιας τροφοδότησης ενός αναδρομικού δικτύου.

8.5 Μοντέλα Βαθιάς Μάθησης (Deep Learning)

8.5.1 Εισαγωγή

Η βαθιά μάθηση είναι μια τεχνική μηχανικής μάθησης που διδάσκει στους υπολογιστές να κάνουν αυτό που είναι φυσικό για τους ανθρώπους δηλαδή να μαθαίνουν από το παράδειγμα. Η βαθιά μάθηση είναι μια βασική τεχνολογία πίσω από τα αυτοκίνητα χωρίς οδηγό, που τους επιτρέπει να αναγνωρίζουν ένα σήμα στοπ ή να διακρίνουν έναν πεζό από μια κολόνα. Είναι το κλειδί για τον φωνητικό έλεγχο σε καταναλωτικές συσκευές όπως τηλέφωνα, tablet, τηλεοράσεις και ηχεία hands-free. Η βαθιά εκμάθηση λαμβάνει μεγάλη προσοχή τελευταία και για καλό λόγο. Επιτυγχάνει αποτελέσματα που δεν ήταν εφικτά πριν.

Στη βαθιά μάθηση, ένα μοντέλο υπολογιστή μαθαίνει να εκτελεί εργασίες ταξινόμησης απευθείας από εικόνες, κείμενο ή ήχο. Τα μοντέλα βαθιάς μάθησης μπορούν να επιτύχουν κορυφαία ακρίβεια, που μερικές φορές ξεπερνά τις επιδόσεις σε ανθρώπινο επίπεδο. Τα μοντέλα εκπαιδεύονται χρησιμοποιώντας ένα μεγάλο σύνολο επισημασμένων δεδομένων και αρχιτεκτονικές νευρωνικών δικτύων που περιέχουν πολλά επίπεδα.

8.5.2 Συνελικτικά δίκτυα

Στη βαθιά μάθηση, ένα νευρωνικό δίκτυο συνέλιξης (CNN ή ConvNet) είναι μια κατηγορία τεχνητών νευρωνικών δικτύων (ANN), που εφαρμόζεται συνήθως για την ανάλυση οπτικών εικόνων. Τα CNN είναι επίσης γνωστά ως τεχνητά νευρωνικά δίκτυα με αναλλοίωτη μετατόπιση ή αναλλοίωτο χώρο (SIANN), με βάση την αρχιτεκτονική των πυρήνων ή φίλτρων συνέλιξης με κοινό βάρος που ολισθαίνουν κατά μήκος των χαρακτηριστικών εισόδου και παρέχουν μεταφορικές αποκρίσεις, γνωστές ως χάρτες χαρακτηριστικών. Αντίθετα, τα περισσότερα νευρωνικά δίκτυα συνέλιξης δεν είναι αναλλοίωτα στη μετάφραση, λόγω της λειτουργίας υποδειγματοληψίας που εφαρμόζουν στην είσοδο. Έχουν εφαρμογές στην αναγνώριση εικόνων και βίντεο, στα συστήματα συστάσεων, στην ταξινόμηση εικόνων, στην κατάτμηση

εικόνων, στην ανάλυση ιατρικών εικόνων, στην επεξεργασία φυσικής γλώσσας, στις διεπαφές εγκεφάλου-υπολογιστή και στις χρηματοοικονομικές χρονοσειρές.

Τα CNN είναι κανονικοποιημένες εκδόσεις των πολυεπίπεδων perceptrons. Τα πολυεπίπεδα perceptrons συνήθως σημαίνουν πλήρως συνδεδεμένα δίκτυα, δηλαδή κάθε νευρώνας σε ένα επίπεδο συνδέεται με όλους τους νευρώνες στο επόμενο επίπεδο. Η ¨πλήρης συνδεσιμότητα' αυτών των δικτύων τα καθιστά επιρρεπή στην υπερπροσαρμογή των δεδομένων. Οι τυπικοί τρόποι κανονικοποίησης ή αποτροπής της υπερπροσαρμογής περιλαμβάνουν: την τιμωρία των παραμέτρων κατά την εκπαίδευση (όπως η αποσύνθεση των βαρών) ή την περικοπή της συνδεσιμότητας (παραλείψεις συνδέσεων, διακοπή συνδέσεων κ.λπ.) Τα CNN ακολουθούν μια διαφορετική προσέγγιση προς την κανονικοποίηση. Πιο συγκεκριμένα εκμεταλλεύονται το ιεραρχικό μοτίβο των δεδομένων και συναρμολογούν μοτίβα αυξανόμενης πολυπλοκότητας χρησιμοποιώντας μικρότερα και απλούστερα μοτίβα που είναι αποτυπωμένα στα φίλτρα τους. Επομένως, σε μια κλίμακα συνδεσιμότητας και πολυπλοκότητας, τα CNN βρίσκονται στο χαμηλότερο άκρο.

Τα CNN χρησιμοποιούν σχετικά λίγη προεπεξεργασία σε σύγκριση με άλλους αλγορίθμους ταξινόμησης εικόνων. Αυτό σημαίνει ότι το δίκτυο μαθαίνει να βελτιστοποιεί τα φίλτρα (ή πυρήνες) μέσω αυτοματοποιημένης μάθησης, ενώ στους παραδοσιακούς αλγορίθμους αυτά τα φίλτρα σχεδιάζονται με το χέρι. Αυτή η ανεξαρτησία από την προηγούμενη γνώση και την ανθρώπινη παρέμβαση στην εξαγωγή χαρακτηριστικών είναι ένα σημαντικό πλεονέκτημα.

Για τον υπολογισμό της πράξης της συνέλιξης μεταξύ εικόνων, χρησιμοποιείται ο τύπος της δισδιάστατης συνέλιξης

$$C(i,j) = \sum_{m=0}^{M_a-1} \sum_{n=0}^{N_a-1} A(m,n) * B(i-m, j-n)$$

ο οποίος εφαρμόζεται μεταξύ δύο πινάκων από pixels (εικόνων) A και B και παράγει έναν νέο πίνακα C. Ο πρώτος πίνακας αντιστοιχεί στην εικόνα εισόδου, ο δεύτερος, είναι ένα φίλτρο (kernel) το οποίο "περνάει" πάνω από την εικόνα, φιλτράροντάς την και παράγοντας τον πίνακα εξόδου C.

Η αρχιτεκτονική των συνελικτικών νευρωνικών δικτύων,απαρτίζεται από ένα στρώμα εισόδου και εξόδου, καθώς και από περισσότερα κρυφά επίπεδα. Κάθε κρυφό επίπεδο, απαρτίζεται από συνεχόμενα επίπεδα συνέλιξης, συνάρτησης ενεργοποίησης, υποδειγματοληψίας της εικόνας καθώς και από ένα ή περισσότερα πλήρως συνδεδεμένα επίπεδα (fully connected). Τα συνεχόμενα αυτά στώματα έχουν τις εξής εργασίες:

1. **Συνελικτικά (Convolutional):** Αποσκοπούν στον υπολογισμό της εικόνας εξόδου ή χάρτη χαρακτηριστικών (feature map), μετά από την εφαρμογή του φίλτρου (kernel) του οποίου οι τιμές είναι εκπαιδεύσιμες (trainable) από το δίκτυο και το πέρασμα του αποτελέσματος από μια συνάρτηση ενεργοποίησης. Πρακτικά, η ορθή ρύθμιση αυτών των παραμέτρων είναι αυτό που οδηγεί το δίκτυο στην σωστή ανάλυση της εικόνας, επιτυνγχανοντάς το επιθυμητό αποτέλεσμα.

2. **Υποδειγματοληψία (Pooling):** Συρικνόνει τις διαστάσεις του πίνακα εισόδου συνδυάζοντας τις εξόδους του προηγούμενου επιπέδου. Συνδυάζει pixels περιοχών της

εικόνας εισόδου, τυπικά 2 × 2, υπολογίζοντας το μέγιστο (max pooling) ή το μέσο (average pooling) των τιμών.

3. **Πλήρως συνδεδεμένο επίπεδο (fully connected layer):** Οι τελικοί χάρτες χαρακτηριστικών που προκύπτουν ύστερα από τις πολλές αλλαγές συνελικτικών και pooling επιπέδων, περνούν από ένα πλήρως συνδεδεμένο νευρωνικό δίκτυο, όπως αυτό που περιγράφεται στην προηγούμενη ενότητα, για την εξαγωγή του τελικού αποτελέσματος.

Τα συνελικτικά νευρωνικά δίκτυα, χρησιμοποιούνται με επιτυχία και για προβλήματα ανάλυσης χρονοσειρών, αφού τα δεδομένα αυτά , μπορούν να θεωρηθούν ως πίνακες εισόδου και άρα το συνελικτικό δίκτυο θα μπορέσει να ανιχνεύσει πρότυπα αυτών.

8.6 Transfer Learning

Με τον όρο Transfer Learning (Μεταφοράς Μάθησης) αναφερόμαστε στην επαναχρησιμοποίηση ενός προ-εκπαιδευμένου μοντέλου σε ένα νέο (παρόμοιο) πρόβλημα.

Η εκμάθηση ενός καινούργιου μοντέλου για κάθε διαφορετικό πρόβλημα απαιτεί πληθώρα δεδομένων με ετικέτες (labeled data). Για να επιτύχουν ικανοποιητικά αποτελέσματα, τα μοντέλα αυτά τυπικά χρειάζεται να εκπαιδευτούν σε δεδομένα της τάξεως των χιλιάδων. Συχνά τέτοιο πλήθος δεδομένων δεν είναι διαθέσιμο, συνεπώς η μοναδική λύση είναι η μεταφορά βασικής γνώσης απο ενα ήδη εκπαιδευμένο μοντέλο στο νέο πρόβλημα. Η μεταφορά μάθησης συνήθως οδηγεί σε γρηγορότερη σύγκλιση και υψηλότερη απόδοση από αυτήν που θα είχε το μοντέλο, αν είχε εκπαιδευτεί μόνο σε ένα μικρό σύνολο δεδομένων.

Η διαίσθηση πίσω από τη μεταφορά μάθησης για την ταξινόμηση εικόνων είναι ότι αν ένα μοντέλο εκπαιδευτεί σε ένα αρκετά μεγάλο και γενικό σύνολο δεδομένων, αυτό το μοντέλο θα έχει μάθει να αναγωνρίζει βασικά μοτίβα του οπτικού κόσμου. Συνεπώς μπορούμε να επωφεληθούμε από αυτούς τους μαθημένους χάρτες χαρακτηριστικών (π.χ. αναγνώρηση βασικών σχημάτων) χωρίς να χρειάζεται να ξεκινήσουμε από το μηδέν εκπαιδεύοντας ένα μεγάλο μοντέλο σε ένα μεγάλο σύνολο δεδομένων.

Στον τομέα της υπολογιστικής όρασης, τα νευρωνικά δίκτυα συνήθως προσπαθούν να ανιχνεύσουν ακμές στα πρώτα στρώματα του νευρωνικού, σχήματα στα μεσαία στρώματα και κάποια χαρακτηριστικά που αφορούν το συγκεκριμένο πρόβλημα στα επόμενα στρώματα. Στη μεταφορά μάθησης, χρησιμοποιούνται οι τιμές των βαρών των πρώτων και μεσαίων στρώματων και επανεκπαιδεύουμε μόνο τα τελευταία στρώματα με τα (λίγα) δεδομένα που διαθέτουμε για το πρόβλημά μας. [8]

Κεφάλαιο 9

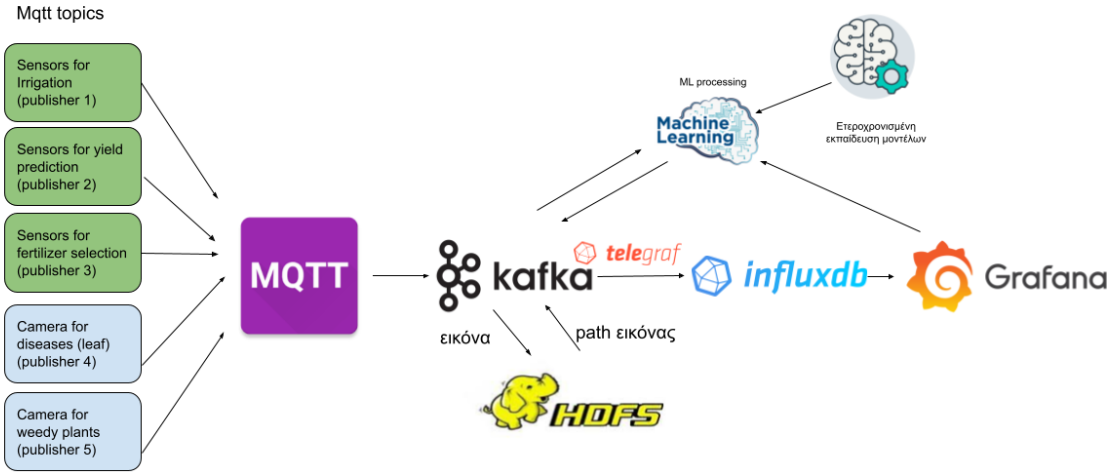
Παρουσίαση Επιλεγμένης Αρχιτεκτονικής και Εφαρμογή της

9.1 Εισαγωγή

Στα προηγούμενα κεφάλαια, αναπτύχθηκε όλη η θεωρία για τα πρωτόκολλα και τις τεχνολογίες που απαιτούντε για την δημιουργία του συστήματος που αναπτύξαμε. Στο παρόν κεφάλαιο θα παρουσιαστεί η εφαρμογή αύτη σε ένα σύστημα το όποιο θα αξιοποιείται από γεωργούς για την εύρυθμη λειτουργία της αγροκαλλιέργειας τους. Η γλώσσα προγραμματισμού που χρησιμοποιήθηκε είναι η Python. Σε όλα τα επίπεδα εξασφαλίζεται η κρυπτογράφηση των μηνυμάτων για λόγους ασφάλειας και τηρούνται όλα τα απαραίτητα μέτρα για την ασφάλεια από κακόβουλες επιθέσεις στο σύστημα μας. Όλα τα μηνύματα έχουν Transport Layer Security (TLS) πιστοποιητικό καθώς και Secured Sockets Layer (SSL)

9.2 Αρχιτεκτονική

Η αρχιτεκτονική που βασίστηκε η υλοποίηση ακολουθεί το σχεδιάγραμμα του σχήματος 9.1.



Σχήμα 9.1: Αρχιτεκτονική που υλοποιήθηκε.

Πιο συγκεκριμένα, στην αρχή του συστήματος μας έχουμε μια ομαδα απο sensors και κάμερες οι οποίοι μαζεύουν δεδομένα και αποστέλλονται με το MQTT πρωτόκολλο στο Apache Kafka. Τα δεδομένα που είναι εικόνες τα όποια έχουν μεταφερθεί υπο την μορφή byte array ξαναγίνονται εικονες και αποθηκεύονται με την χρήση του Hadoop Distributed File System (HDFS) κατανεμημένα και επιστρέφετε το path το οποίο ¨αποθηκεύτηκαν¨. Το ίδιο συμβάνει και για όλα τα άλλα δεδομένα για την μόνιμη αποθήκευση τους. Στην συνέχεια ο Apache Kafka με την βοήθεια του Telegraf μεταφέρει τα δεδομένα στην βάση δεδομένων χρονοσειρών την InfluxDB όπου και αποθηκεύονται. Τέλος, το Grafana μας παρέχει οπτικοποίηση των δεδομένων. Στο επίπεδο του Grafana έχουμε και την εμφανιση των μοντέλων μηχανικής μάθησης. Τα μοντέλα αυτά εκπαιδεύτηκαν ετεροχρονισμένα από πραγματικά δεδομένα. Τα μοντέλα λαμβάνουν το μήνυμα δημιουργίας καποιάς πρόβλεψης απο το Grafana με αποτέλσματα να δέχεται δεδομένα απο τον Kafka για την παραγωγή της εξόδου. Την έξοδο αύτη την στέλνει στον Kafka πάλι για να ακολουθήσει τον ίδιο δρόμο με τα υπόλοιπα δεδομένα απο αισθητήρες. Στην συνέχεια του κεφαλαίου αυτού θα αναλυθούν κάθε κομμάτι του συστήματος ξεχωριστά.

9.3 Sensors

Η παραγωγή των δεδομένων γίνεται από κάποιους sensors οι οποίοι έχουν τοποθετηθεί στην αγροκαλλιέργεια. Αυτοι οι sensors χωρίζονται σε πέντε κατηγορίες ανάλογα ποιό μοντέλο μηχανικής μάθησης τροφοδοτούν εν τέλει με τα δεδομένα τους. Δηλαδή για κάθε ομάδα τέτοιων sensor έχει φτιαχτεί και εκπαιδευτεί ένα μοντέλο για την λήψη μιας απόφασης. Πιο συγκεκριμένα έχουμε:

1. Sensors για άρδευση: αύτη η ομάδα μαζεύει δεδομένα για να μπορέσει το μοντέλο της άρδευσης να αποφανθεί άμα ο γεωργός πρέπει να ποτίσει την αγροκαλλιέργεια ή οχι.

2. Sensors για την πρόβλεψη της σοδειάς: αύτη η ομάδα μαζεύει δεδομένα για να μπορέσει

το μοντέλο της πρόβλεψης να αποφανθεί την αναμενόμενη ποσότητα της σοδειάς.

3. Sensors για την υπόδειξη του κατάλληλου λιπάσματος: αυτή η ομάδα μαζεύει δεδομένα για να μπορέσει το μοντέλο της επιλογής λιπάσματος να επιλέξει το πλέον κατάλληλο λίπασμα σύμφωνα με τις απαιτήσεις της αγροκαλλιέργειας.

4. Camera για τον εντοπισμό κάποιου ιού στα φυτά: αυτή η ομάδα μαζεύει εικόνες των φύλλων των φυτών για τον εντοπισμό κάποιου ιού για την εγκαιρη αγωγή του κατάλληλων αντιβιοτικών στην σοδειά.

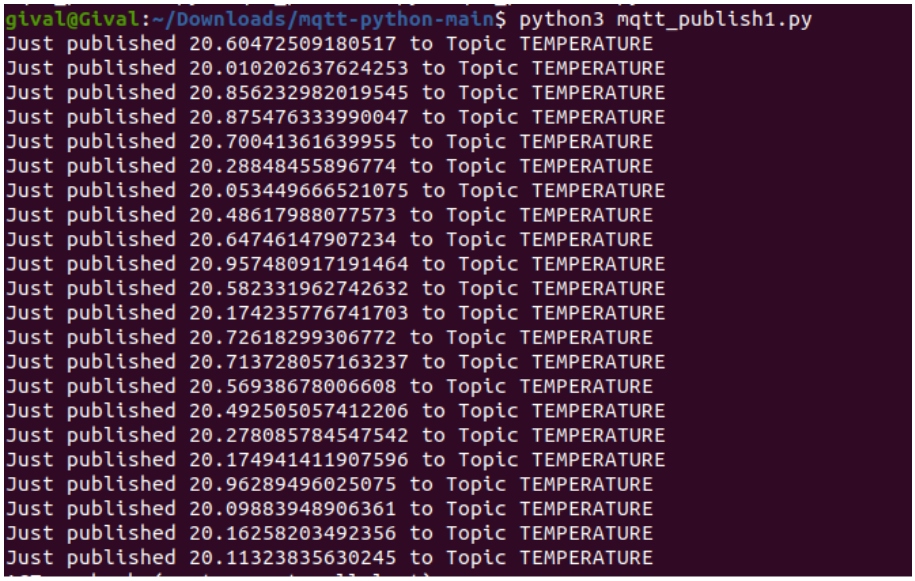
5. Camera για τον εντοπισμό ζιζανίων φυτών: αυτή η ομάδα μαζεύει εικόνες ριζών των φυτών για τον εντοπισμό ζιζανίων που δυσκολεύουν την ανάπτυξη τους.

Αυτή η παραγωγή δεδομένων δεν γίνεται με την ίδια συχνότητα, διαφέρει απο ομαδα σε ομαδα. Ειδικότερα η ομάδα Sensors για άρδευση παράγει δεδομένα σε πιο τακτα διαστήματα από την ομάδα Sensors για την υπόδειξη του κατάλληλου λιπάσματος.

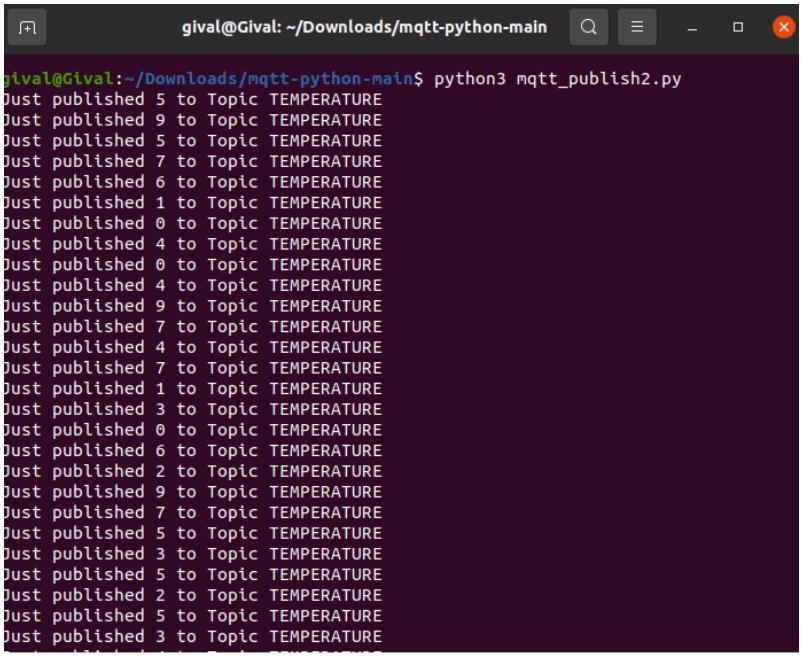
Για την ανάγκη του πειράματος μας τα δεδομένα αυτά αποκτήθηκαν από πραγματικά δεδομένα που βρέθηκαν διαθετιμένα στην πλατφόρμα του Kaggle είτε από εταιρίες είτε από ερευνητικές ομάδες. Η ροη των δεδομένων για να αναπαραστεί κανονικές συνθήκες, έγινε από κάποια scripts σε Python που αναπτύχθηκαν από εμάς ακριβώς για αυτόν τον σκοπό. Τα δεδομένα αυτά μεταφέρονται με την βοήθεια του πρωτοκόλλου MQTT το οποίο θα αναπτυχθεί στην συνέχεια.

9.4 MQTT

Οι ομάδες των sensors που αναφέρθηκαν στην προηγούμενη ενότητα ανταλλάσουν μηνύματα με το πρωτόκολλο του MQTT. Πιο συγκεκριμένα, εγινε χρησιμοποιήσει του Eclipse Mosquitto broker που έγινε εγκατάσταση στο default port 1883, επιπλέον ο χρόνος ζώης(Time to Live (TTL)) των μηνυμάτων μέσα στο MQTT έχει οριστεί στην default τιμή των 60 δευτερόλεπτων δηλαδή ενός λεπτού. Οι sensors αποτελούν τους publisher, για κάθε ένα έχει δημιουργηθεί ένα topic με το αντίστοιχο όνομα (π χ. για την θερμοκρασία έχει δημιουργηθεί ένα topic με όνομα temperature). Τα μηνύματα αυτά ανταλάσονται με ποιότητα εξυπηρέτησης Qos(Quility of service) νούμερο 2 για την παραλαβή του μηνύματος στο broker ακριβώς μια φορά. Στα παρακάτω στιγμιότυπο οθόνης αναπαριστάται η διαδικασία publish δύο sensors θερμοκρασίας.

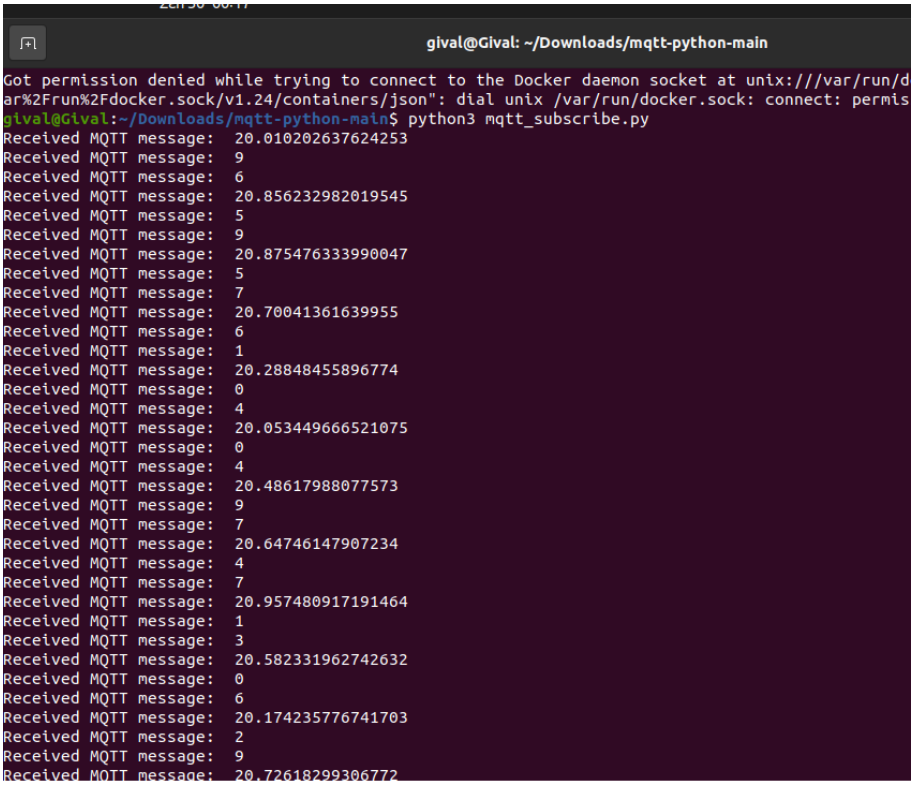


Σχήμα 9.2: Παραγωγή δεδομένων θερμοκρασίας publisher1.



Σχήμα 9.3: Παραγωγή δεδομένων θερμοκρασίας publisher2.

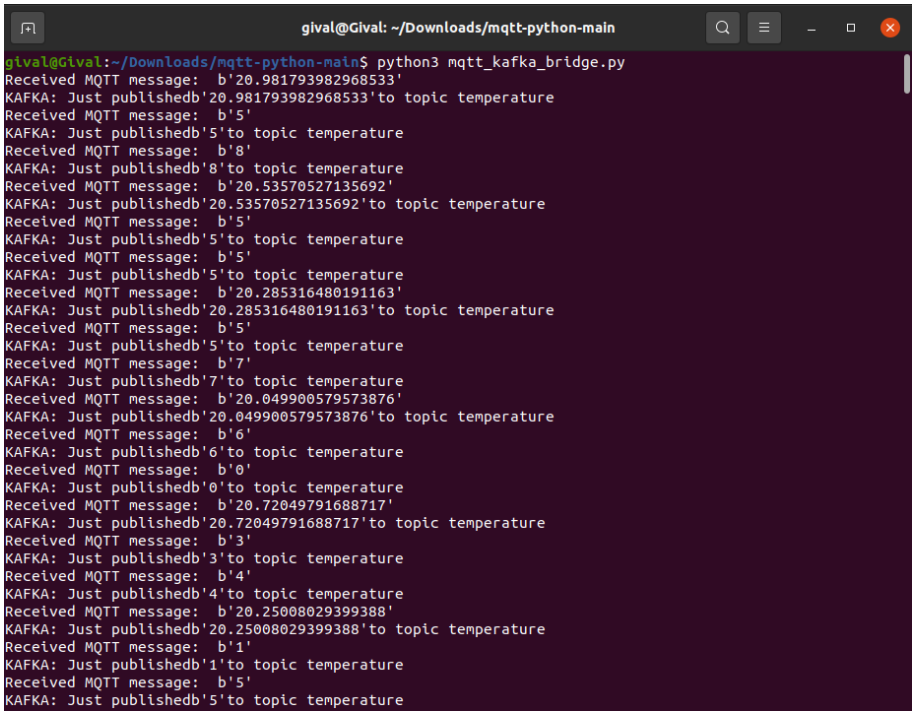
Στην συνέχεια με την βοήθεια του broker είναι στην άλλη πλευρά ένας consumer που δέχεται αυτο τα μηνύματα που στην περίπτωση μας είναι ο Apache Kafka. Στο επόμενο στιγμιότυπο αποτυπώνεται σε πρώτη φάση ένας απλός Consumer που αποδέχεται τα μηνύματα στο topic Temperature που στέλνουν οι προηγούμενοι publisher.



Σχήμα 9.4: Consumer που λαμβάνει τα δεδομένα.

9.5 Apache Kafka

Ο Apache Kafka με default port :9092 έχει εγκατασταθεί στο σύστημά μας,επιπλέον ο χρόνος ζώης(Time to Live (TTL)) των μηνυμάτων μέσα στον Apache Kafka είναι 168 ώρες δηλαδή μια εβδομάδα. Αφού είδαμε την παραπάνω λειτουργία αντί για τον προηγούμενο Mqtt Consumer χρησιμοποιούμε στο σύστημα μας ένα script το όποιο λέγεται mqqt_kafka_bridge.py. Αυτο το script στην ουσια υποδέχεται το μήνυμα από το topic temprature του MQTT και κάνει publish το ίδιο μήνυμα στο topic του Apache Kafka που ονομάζεται με την σειρά του temperature.



Σχήμα 9.5: Kafka bridge που λαμβάνει τα δεδομένα από το MQTT και τα κανει publish στον topic του Kafka temperature.

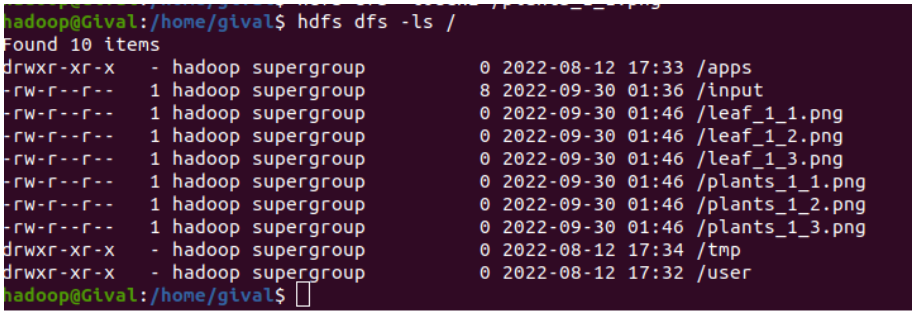
Στη συνέχεια βλέπουμε στο σχήμα 9.6 ότι τα δεδομένα που είχαμε στείλει από την αρχή του συστήματος (στο επίπεδο των sensors) υποδέχονται από έναν KAFKA consumer. Πρέπει να επισημανθεί ότι κάθε είδος δεδομένου και στον Apache Kafka όπως έγινε και στο MQTT επίπεδο θα έχουν δικό του topic (π.χ. ένα topic για την θερμοκρασία, ένα topic για υγρασία κ.τ.λ.). Για λόγους πληρότητας θα αναφερθεί έδω ότι λαμβάνει και τις εξόδους των μοντέλων μηχανικής μάθησης για να ακολουθήσουν την ίδια πορία με τα άλλα δεδομένα, το πως γίνεται αυτό θα το περιγράψουμε παρακάτω.

```
gival@Gival:~/Downloads/mqtt-python-main$ python3 kafka_cons
Topic Name=temperature, Message=b"b'0'"
Topic Name=temperature, Message=b"b'20.226104111588896'"
Topic Name=temperature, Message=b"b'7'"
Topic Name=temperature, Message=b"b'8'"
Topic Name=temperature, Message=b"b'20.384587036960983'"
Topic Name=temperature, Message=b"b'2'"
Topic Name=temperature, Message=b"b'1'"
Topic Name=temperature, Message=b"b'20.59073378918842'"
Topic Name=temperature, Message=b"b'5'"
Topic Name=temperature, Message=b"b'0'"
Topic Name=temperature, Message=b"b'20.804885327356985'"
Topic Name=temperature, Message=b"b'4'"
Topic Name=temperature, Message=b"b'5'"
Topic Name=temperature, Message=b"b'20.867401029512088'"
Topic Name=temperature, Message=b"b'6'"
Topic Name=temperature, Message=b"b'4'"
Topic Name=temperature, Message=b"b'20.79285238076708'"
Topic Name=temperature, Message=b"b'6'"
Topic Name=temperature, Message=b"b'1'"
Topic Name=temperature, Message=b"b'20.96837037479328'"
Topic Name=temperature, Message=b"b'6'"
Topic Name=temperature, Message=b"b'3'"
Topic Name=temperature, Message=b"b'20.71409225786835'"
Topic Name=temperature, Message=b"b'3'"
Topic Name=temperature, Message=b"b'8'"
Topic Name=temperature, Message=b"b'20.94030388896446'"
Topic Name=temperature, Message=b"b'1'"
Topic Name=temperature, Message=b"b'2'"
```

Σχήμα 9.6: Consumer που λαμβάνει τα δεδομένα.

9.6 Hadoop Distributed File System (HDFS)

Όταν ο Consumer του Kafka υποδεχτεί τα αρχεία, τα αρχεία που έρχονται από topics που προορίζεται για την συλλογή εικόνων, τα μετατρέπει από bytes array σε εικόνες. Αυτές τις εικόνες τις αποθηκεύει σε ενάν path το οποίο διέπεται από την χρήση του HDFS δίνοντας του τον επόμενο αύξοντα αριθμός (π.χ. οι εικόνες που είναι από την ςαμερα για τους ιους, το όνομα τους έχει την μορφή leaf_1_x, όπου x ο αύξοντας αριθμός). Στην συνέχεια λαμβάνει το path της εικονας για να προωθησει αυτο ο KAFKA consumer και όχι τα δεδομένα που έλαβε. Τα υπολοίπα δεδομένα (δεδομένα από αισθητήρες και αποτέλεσμα των μοντέλων μηχανικής μάθησης) απλά τα αποθηκεύει χωρίς καμία άλλη επεξεργασία για την μόνιμη αποθήκευση τους. Όπως φαίνεται από το σχήμα 9.7



Σχήμα 9.7: Αποθήκευση των εικόνων σε HDFS στο πείραμα μας

9.7 Telegraf & InfluxDB

Έγινε εγκατάσταση του InlfuxDB καθώς και η δημιουργία Table όσων και των topics του Apache Kafka με ακριβώς τα ίδια ονόματα. Στην συνεχεια ρυθμίστηκαν οι κατάλληλοι παράμετροι στο αρχείο telegraf.conf ετσι ώστε να επιτευχθεί η ζεύξη μεταξύ του Kafka και της InfluxDB. Το Telegraf ότι πληροφορία λαμβάνει ο Kafka Consumer θα το προώθει και θα το εισάγει στο αντίστοιχο table της βάσεις με ακριβώς το ίδιο όνομα. Το Telegraf εχει ρυθμιστεί να λειτουργεί σε Consumer Group έτσι ώστε να επιτυγχάνεται η παραλληλοποιήση αποστόλης δεδομένων δηλαδή να διαβάζει διαφορετικά μηνύματα απο το ίδιο topic του Apache Kafka και να τα προώθει στην InfluxDB. Τέλος, τα δεδομένα που στέλνονται στην InfluxDB έχουν διάρκεια ζωής ενός μήνα μέτα απο αύτο το διάστημα διαγράφονται. Σε περίπτωση που κάποιος θέλει να τα ανακτήσει μπορεί να ανατρέξει στο HDFS άφου έχει γίνει μόνιμη αποθηκεύση τους έκει.

9.8 Machine Learning

Τα μοντελα που χρησιμοποιήθηκαν εκπαιδευτηκαν ετεροχρονισμένα με πραγματικά δεδομένα που βρέθηκαν στην πλατφόρμα του Kaggle. Αυτά τα μοντέλα χωρίζονται σε δύο κατηγορίες ανάλογα με το τύπο των δεδομένων, εικόνες και tabular. Επιπλέον τα μοντέλα χωρίζονται ανάλογα με τον στόχο σε Classification και Regression. Στην συνέχεια θα αναλύσουμε τους αλγόριθμους που χρησιμοποιήθηκαν σε κάθε περίπτωση.

9.8.1 Μοντέλο άδρευσης

Στο συγκεκριμένο μοντέλο χρησιμοποιήθηκε το μοντέλο Multi-layer Perceptron classifier το οποίο είναι μοντέλο βαθιάς μάθησης. Του εισάγαμε τέσσερις εισόδους η οποίες ήταν :

1. υγρασία εδάφους
2. θερμοκρασία
3. υγρασία ατμόσφαιρας
4. ώρα

	Soil Moisture	Temperature	Humidity	Time
0	54	22	70	21
1	12	20	40	104
2	34	26	35	62
3	7	44	44	93
4	50	38	23	92
...	...	...	...	...
99995	74	3	29	83
99996	58	10	46	82
99997	4	35	39	17
99998	83	36	56	15
99999	43	45	45	33

100000 rows × 4 columns

Σχήμα 9.8: Dataset που έγινε χρήση.

και αναμέναμε εξοδο που μας όριζε αν η αγροκαλλιέργεια χρειάζεται πότισμα ή όχι. Το μοντέλο αυτό έχει (150X150X150 νευρώνες) τρία κρυφά επίπεδα. Το Accuracy του μοντέλου είναι στο 94% ενώ το Precision και Recall ήταν 96% και 93% αντίστοιχα.

9.8.2 Μοντέλο πρόβλεψη της σοδειά

Σε αυτή την περίπτωση φτιάξαμε την δικιά μας αρχιτεκτονική χρησιμοποιώντας την βιβλιοθήκη Keras. Συγκεκριμένα φτιάξαμε ένα νευρωνικό με 5 εισόδους:

1. ετήσια παραγωγή
2. μ.ο. βροχών που είχε φέτος
3. άζωτο της ατμόσφαιρας
4. ποτάσα που έχει το έδαφος
5. έκταση της αγροκαλλιέργειας

```
      ANNUAL    avg_rain  Nitrogen  POTASH  RICE_PRODUCTION
287    700.0   54.166667     38806   20134           128.44
1343   390.3   22.408333     91199    5370           196.03
2123  1911.0  139.858333     29459    7720           819.25
804   1392.8   81.966667     33386    4202           734.97
1782  1570.6   89.525000       326      22            18.65
...      ...         ...       ...     ...              ...
960    594.9   36.500000     41371     801            38.18
905    498.7   29.216667      6751     318             0.02
1096   552.9   22.500000    113564    8279          1264.65
235    876.5   48.800000    121694   19952          1011.54
1061   583.1   35.275000     91133    3766           672.34

[2282 rows x 5 columns]       RICE_YIELD
287      1973.87
1343     2072.08
2123     1590.53
804      1982.65
1782     1085.56
...          ...
960      1977.21
905      2000.00
1096     4685.62
235      3718.49
1061     3745.63

[2282 rows x 1 columns]
```

Σχήμα 9.9: Dataset που έγινε χρήση.

και αναμέναμε εξοδο που μας ορίζει την συνολική παραγωγή ρυζιού. Το μοντέλο αυτό έχει (42X7X3 νευρώνες) τρία κρυφά επίπεδα. Επειδή αυτό το μοντέλο έχει λιγότερους νευρώνες έχει μικρότερο Accuracy απο την προηγούμενη περίπτωση.

9.8.3 Μοντέλο πρόβλεψης λιπάσματος

Στην συγκεκριμένη περίπτωση τα δεδομένα που είχαμε ήταν γραμμικά διαχωρίσιμα επομένως χρησιμοποιήσαμε Μηχανές Διανυσμάτων Υποστήριξης (Support Vector Machines SVM). Αυτό το μοντέλο είχε για είσοδο:

1. Ca
2. Mg
3. K
4. S
5. N
6. Lime
7. C
8. P

9. υγρασία

# Ca	# Mg	# K	# S	# N
0.7	0.6	0.8	0.8	0.7
0.5	0.5	0.4	0.3	0.5
0.6	0.8	0.1	0.3	0.7
0.7	0.7	0.7	0.5	0.8
0.8	0.8	0.2	0.3	0.5
0.7	0.5	0.6	0.7	0.8
0.7	0.5	0.8	0.8	0.2
0.1	0.4	0.7	0.5	0.6
0.5	0.8	0.3	0.4	0.7
0.4	0.2	0.6	0.6	0.6
0.8	0.7	0.5	0.6	0.1
0.6	0.6	0.7	0.5	0.2

Σχήμα 9.10: Dataset που έγινε χρήση.

# Lime	# C	# P	# Moisture	# class
0.8	0.3	0.1	0.9	4
0.7	0.5	0.7	0.8	2
0.5	0.5	0.6	0.6	2
0.7	0.4	0.1	0.7	4
0.5	0.7	0.8	0.5	2
0.8	0.3	0.4	0.8	4
0.4	0.7	0.5	0.7	3
0.7	0.5	0.6	0.9	1
0.7	0.8	0.6	0.7	2
0.5	0.8	0.7	0.5	1
0.4	0.7	0.6	0.7	3

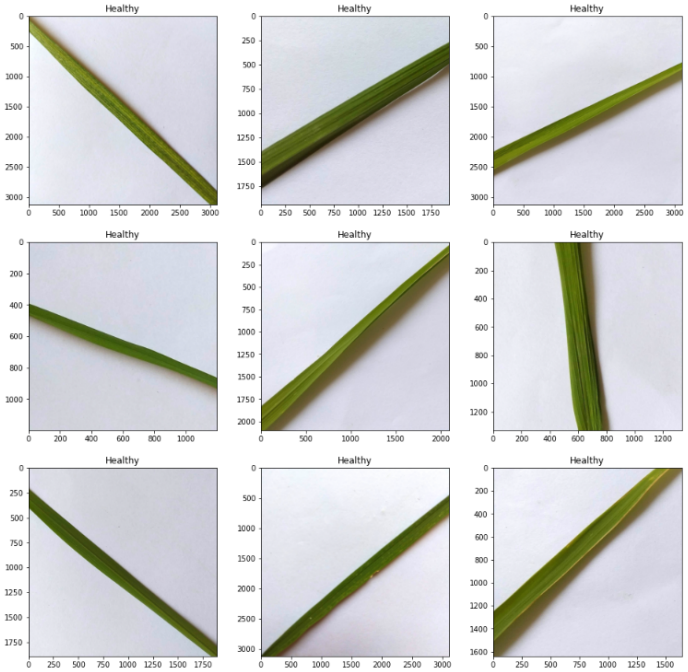
Σχήμα 9.11: Dataset που έγινε χρήση.

και αναμέναμε εξοδο που να κάνει επιλογή ενός λιπάσματος ανάμεσα από τέσσερα. Λόγω των δεδομένων το Accuracy βγήκε 100%.

9.8.4 Μοντέλο αναγνώρισης ιου

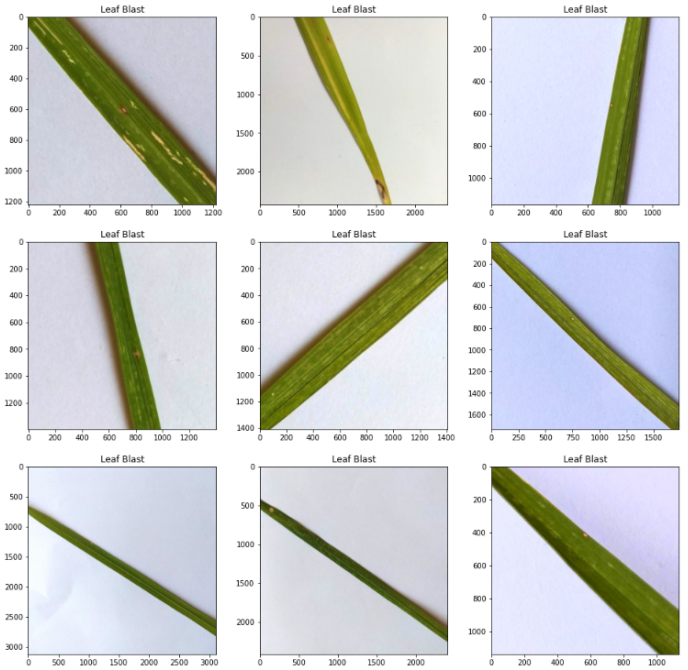
Αυτό το μοντέλο που εκπαιδεύσαμε καλείτε με είσοδο μια εικόνα ενός φύλλου του φυτού να κατηγοροποιήση το φυτο ως:

1. υγιές



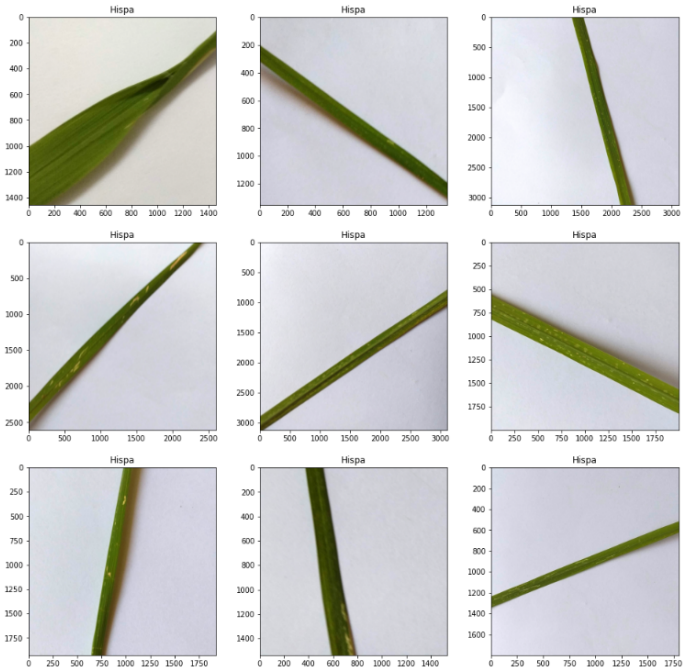
Σχήμα 9.12: Dataset που έγινε χρήση.

2. ότι νοσεί από Leaf Blast



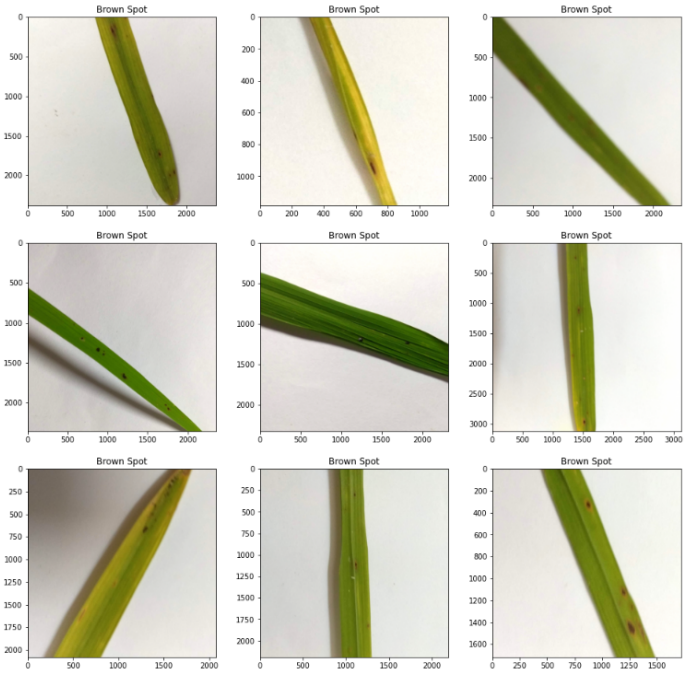
Σχήμα 9.13: Dataset που έγινε χρήση.

3. ότι νοσεί από Hispa



Σχήμα 9.14: Dataset που έγινε χρήση.

4. ότι νοσεί από Brown Spot



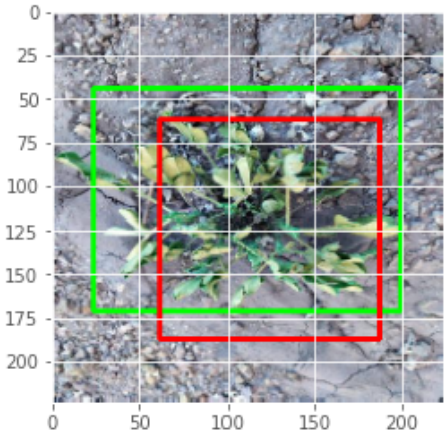
Σχήμα 9.15: Dataset που έγινε χρήση.

Στην περίπτωσή μας το απλό νευρωνικό δίκτυο δεν αρκεί διότι αναγνωρίσει τοπικά μοτίβα πάνω σε μια εικόνα. Για αυτό τον λόγο χρησιμοποιούμε συνελικτικά νευρωνικά. Η εικόνα περνάει από τέσσερα φίλτρα το κάθε φίλτρο μειώνει τις διαστάσεις της εικόνας, στόχος της

εκπαίδευσης είναι η εκμάθηση των τιμών του κάθε φίλτρου που χρησιμοποιείται. Τα μεγέθη των φίλτρων καθορίζονται ως εξής [32, 32, 128], [64, 64, 256], [128, 128, 512], [256,256, 1024] με την σειρα που ειπώθηκαν. Έπειτα η έξοδος του τελευταίου φίλτρου ισοπεδώνεται (flatten) και περνάει από ένα πλήρες συνδεδεμένο νευρωνικό με εξόδους όσες και οι κλάσεις δηλαδή τέσσερις. Το μοντέλο πετυχαίνει Accuracy 82%.

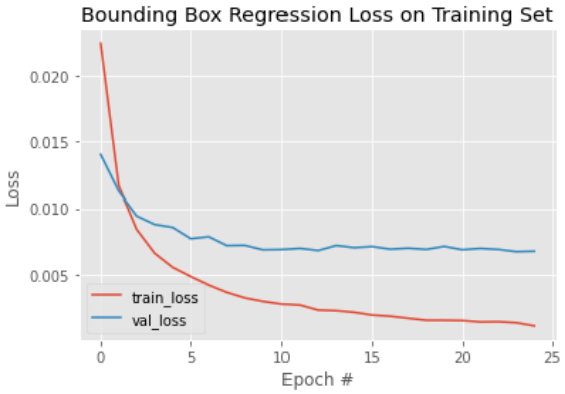
9.8.5 Μοντέλο εντοπισμού ζιζανίων

Αυτό το μοντέλο διαφέρει από τα άλλα γιατί θέλει να προβλέπει τέσσερις πραγματικές τιμές και όχι να κατηγοριοποίηση την είσοδο σε κάποια κλάση (Regression). Λόγω χαμηλού πλήθους πραγματικών δεδομένων και της δυσκολίας του προβλήματος η εκπαίδευση ενός συνεληξιακού μοντέλου όπως του προηγούμενου δεν θα έφερνε τα επιθυμητά αποτελέσματα. Για αυτό τον λόγο εφαρμόζουμε την τεχνική της μεταφοράς μάθησης (Transfer Learning) η οποία χρησιμοποιεί ένα προεκπαιδευμένο μοντέλο το οποίο έχει εκπαιδευτεί σε εκατομμύρια εικόνες και έχει μάθει να αναγνωρίζει μοτίβα σε αυτές.Για να πετύχουμε την εκπαίδευση στα δικά μας δεδομένα ¨παγώνουμε' όλα εκτός από τα τελευταία επίπεδα του προεκπαιδευμένου μοντέλου και προσθέτουμε νέα τεχνητά επίπεδα των οποίων τα βάρη θέλουμε να μάθουμε με βάση τα δικά μας δεδομένα.



Σχήμα 9.16: Με πράσσινο χρώμα το πραγματικό bounding box και με κοκκινο η πρόβλεψη του μοντέλου μας.

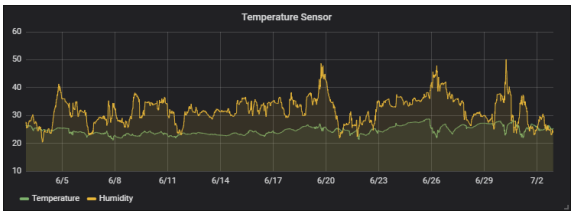
Συγκεκριμένα χρησιμοποιούμε το προεκπαιδευμένο μοντέλο VGG16 το οποίο έχει εκπαιδευτεί σε μια πολύ μεγάλη βάση εικόνων (image net). Αν και αυτη η βάση δεν περιέχει φυτά όπως στην δική μας βάση τα αρχικά του επίπεδα έχουν μάθει να αναγνωρίζουν βασικά σχήματα που βρίσκονται σε μια εικόνα οπότε θα μπορεί για παράδειγμα να αναγνωρίσει την μορφή ενός φύλλου. Όπως και στο προηγούμενο μοντέλο η έξοδος του τελευταίου επιπέδου του VGG16 ισοπεδώνεται (flatten) και προσθέτουμε ένα πλήρες συνδεδεμένο νευρωνικό με [128,64,32,4] νευρώνες οπότε έτσι παράγονται οι τέσσερις τιμές που θέλουμε για να δημιουργηθεί το παραλληλόγραμμο του bounding box. Μια χαρακτηριστική έξοδο του μοντέλου απεικονίζεται στο σχέδιο 9.2.



Σχήμα 9.17: Διάγραμμα loss function για τα δεδομένα εκπαίδευσης και validation.

9.9 Grafana & Machine Learning

Μετά την εγκατάσταση του Grafana και την κατάλληλη σύνδεση του με την βάση InfluxDB. Το Grafana οπτικοποιει τα δεδομένα που υπάρχουν στην βάση. Ένα πιθανό dashboard του συστήματος μας μπορεί να είναι το σχέδιο 9.10.



Σχήμα 9.18: Grafana dashboard.

Στην συνέχεια,με χρησιμοποιούμε το κατάλληλου plugin (Loud ML app) που δίνει την δυνατότητα στο Grafana να ζητάει μια πρόβλεψη απο το μοντέλο μηχανικής μάθησης που την παράγει. Εκεί μπορούμε να έχουμε την αλληλεπίδραση μαζί τους και να βλέπουμε σε πραγματικό χρόνο την απόφαση που παίρνει. Το μοντέλο όταν λάβει το μήνυμα οτι πρέπει να παράγει κάποια απόφαση τότε παίρνει απο τον Apache Kafka τα δεδόμενα που θέλει σε πραγματικό χρόνο. Όταν λάβει την πληροφορία που χρειάζεται και έχει την έξοδο που του ζητήθηκε τότε την στέλνει την έξοδο του στον Kafka, για να ακολουθήσει την πορεία των δεδομένων των αισθητήρων. Δηλαδή αρχικά θα αποθηκευτεί αυτή η έξοδος σε HDFS και μετα μέσω Telgraf θα πάει στην InfluxDB για να έχουμε την οπτικοποιήση του αποτελέσματος μέσω του Grafana.

Κεφάλαιο 10

Σύνοψη και συμπεράσματα

Από όλα αυτά τα δεδομένα και τις αναλύσεις που παρουσιάζονται σε κάθε κεφάλαιο, είναι σαφές ότι υπάρχει ένα δυνητικό σύστημα που ανταποκρίνεται στις απαιτήσεις επεξεργασίας δεδομένων για την έξυπνη γεωργία. Το σύστημα αυτό θα πρέπει να διαθέτει τεχνολογίες παρόμοιες με το MQTT, Apache Kafka, InfluxDB.

Σε κάθε περίπτωση, τα κύρια πλεονεκτήματα του συστήματος είναι, πρώτον, ότι μπορεί να καλύψει τις απαιτήσεις σχεδόν όλων των πιθανών περιπτώσεων χρήσης που σχετίζονται με την επεξεργασία δεδομένων για έξυπνες καλλιέργειες με ελάχιστες αλλαγές και, δεύτερον, ότι είναι εξαιρετικά επεκτάσιμο, ώστε να μπορεί να διαχειρίζεται μεγάλες και αυξανόμενες ποσότητες πληροφοριών με ελάχιστες αλλαγές στο software.

Βιβλιογραφία

[1] MQTT Packet Format. https://openlabpro.com/guide/mqtt-packet-format/.

[2] Finematics. Apache Kafka Explained. https://www.youtube.com/watch?v=JalUUBKdcA0&ab_channel=Finematics, 2019.

[3] Thomas Wood. Softmax Function. https://deepai.org/machine-learning-glossary-and-terms/softmax-layer, 2017.

[4] Assetwolf. Quality of Service (QoS) Π. https://assetwolf.com/learn/mqtt-qos-understanding-quality-of-service.

[5] databricks. Hadoop Distributed File System (HDFS). https://www.databricks.com/glossary/hadoop-distributed-file-system-hdfs, 2017.

[6] Dominik Obermaier. Lightweight and scalable IoT Messaging with MQTT. https://www.slideshare.net/dobermai/lightweight-and-scalable-iot-messaging-with-mqtt, 2019.

[7] Edureka. Internet of Things (IoT) Architecture. https://www.youtube.com/watch?v=FRxRT0DjE7A&fbclid=IwAR3fvbogcKDE5MLumkgc7ClrpWg5XzwOJWzzLx8KXXtiPU5CipnfRJi_wpg&ab_channel=edureka, 2018.

[8] Emanouil Pangiotou. Σχεδίαση, υλοποίηση και εκπαίδευση μοντέλων Μηχανικής Μάθησης σε ενεργειακά και μετεωρολογικά δεδομένα για την πρόβλεψη φορτίου. http://artemis.cslab.ece.ntua.gr:8080/jspui/bitstream/123456789/17795/1/thesis_panagiotou.pdf, 2020.

[9] GEEKSFORGEEKS. Hadoop Distributed File System (HDFS) architecture. https://www.geeksforgeeks.org/hadoop-architecture/, 2017.

[10] GEEKSFORGEEKS. Hadoop Distributed File System (HDFS) pros and cons. https://www.geeksforgeeks.org/hadoop-pros-and-cons/, 2017.

[11] InfluxData. Telegraf Documentation. https://docs.influxdata.com/telegraf/v1.21, 2017.

[12] javapoint. Apache Kafka Architecture. https://www.javatpoint.com/apache-kafka-architecture, 2017.

[13] javapoint. Kafka: Advantages and Disadvantages. https://www.javatpoint.com/apache-kafka-advantages-and-disadvantages, 2017.

[14] javapoint. Support Vector Machine Algorithm. https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm, 2017.

[15] Mainak Chakraborty. Monitoring Cloud-Native Applications. https://docs.influxdata.com/telegraf/v1.21, 2017.

[16] Neha Sharma. The history, present and future with Iot. https://link.springer.com/chapter/10.1007/978-3-030-04203-5_3/, 2017.

[17] Pallavi Sethi, Smruti R. Sarangi. Internet of Things: Architectures, Protocols, and Applications. https://www.hindawi.com/journals/jece/2017/9324035/, 2017.

[18] Prateek Singh. Consider Grafana vs. Prometheus for your time-series tools. https://www.techtarget.com/searchitoperations/tip/Consider-Grafana-vs-Prometheus-for-your-time-series-tools, 2017.

[19] Stefanos Souliotis. Εξόρυξη, Επεξεργασία, Αποθήκευση και Οπτικοποίηση δεδομένων από αισθητήρες σε πραγματικό χρόνο (Live). http://artemis.cslab.ece.ntua.gr:8080/jspui/bitstream/123456789/17872/1/thesis_souliotis.pdf, 2020.

[20] Syeda Noor Zehra Naqvi. (Time Series Databases and InfluxDB). https://cs.ulb.ac.be/public/_media/teaching/influxdb_2017.pdf, 2017.

[21] WikiPedia. Kafka: Advantages and Disadvantages. https://en.wikipedia.org/wiki/Machine_learning, 2017.