ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

Ευφυής Διαχείριση Κατανεμημένων Ροών Δεδομένων σε Πραγματικό Χρόνο.

Φουντάς Παναγιώτης

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΥΠΕΥΘΥΝΟΣ

Σταμούλης Γεώργιος
Καθηγητής

ΣΥΝΕΠΙΒΛΕΠΩΝ

Κολομβάτσος Κωνσταντίνος
Διδάσκων

Λαμία 19 Φεβρουαρίου έτος 2020

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

Ευφυής Διαχείριση Κατανεμημένων Ροών Δεδομένων σε Πραγματικό Χρόνο.

Φουντάς Παναγιώτης

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΥΠΕΥΘΥΝΟΣ

Σταμούλης Γεώργιος
Καθηγητής

ΣΥΝΕΠΙΒΛΕΠΩΝ

Κολομβάτσος Κωνσταντίνος
Διδάσκων

Λαμία 19 Φεβρουαρίου έτος 2020

UNIVERSITY OF THESSALY

SCHOOL OF SCIENCE

DEPARTMENT OF COMPUTER SCIENCE & TELECOMMUNICATIONS

Intelligent Management of Data Streams in Real Time.

Fountas Panagiotis

FINAL THESIS

ADVISOR

Stamoulis George
Proffesor

CO ADVISOR

Kolomvatsos Konstantinos
Instructor

Lamia 19 February year 2020

«Με ατομική μου ευθύνη και γνωρίζοντας τις κυρώσεις [(1)], που προβλέπονται από της διατάξεις της παρ. 6 του άρθρου 22 του Ν. 1599/1986, δηλώνω ότι:

*1. Δεν παραθέτω κομμάτια βιβλίων ή άρθρων ή εργασιών άλλων αυτολεξεί **χωρίς να τα περικλείω σε εισαγωγικά** και χωρίς να αναφέρω το συγγραφέα, τη χρονολογία, τη σελίδα. Η αυτολεξεί παράθεση χωρίς εισαγωγικά χωρίς αναφορά στην πηγή, είναι λογοκλοπή. Πέραν της αυτολεξεί παράθεσης, λογοκλοπή θεωρείται και η παράφραση εδαφίων από έργα άλλων, συμπεριλαμβανομένων και έργων συμφοιτητών μου, καθώς και η παράθεση στοιχείων που άλλοι συνέλεξαν ή επεξεργάσθηκαν, χωρίς αναφορά στην πηγή. Αναφέρω πάντοτε με πληρότητα την πηγή κάτω από τον πίνακα ή σχέδιο, όπως στα παραθέματα.*

*2. Δέχομαι ότι η αυτολεξεί **παράθεση χωρίς εισαγωγικά**, ακόμα κι αν συνοδεύεται από αναφορά στην πηγή σε κάποιο άλλο σημείο του κειμένου ή στο τέλος του, είναι αντιγραφή. Η αναφορά στην πηγή στο τέλος π.χ. μιας παραγράφου ή μιας σελίδας, δεν δικαιολογεί συρραφή εδαφίων έργου άλλου συγγραφέα, έστω και παραφρασμένων, και παρουσίασή τους ως δική μου εργασία.*

3. Δέχομαι ότι υπάρχει επίσης περιορισμός στο μέγεθος και στη συχνότητα των παραθεμάτων που μπορώ να εντάξω στην εργασία μου εντός εισαγωγικών. Κάθε μεγάλο παράθεμα (π.χ. σε πίνακα ή πλαίσιο, κλπ), προϋποθέτει ειδικές ρυθμίσεις, και όταν δημοσιεύεται προϋποθέτει την άδεια του συγγραφέα ή του εκδότη. Το ίδιο και οι πίνακες και τα σχέδια

4. Δέχομαι όλες τις συνέπειες σε περίπτωση λογοκλοπής ή αντιγραφής.

Ημερομηνία: ……/…../20……

Ο – Η Δηλ.

(1) «Όποιος εν γνώσει του δηλώνει ψευδή γεγονότα ή αρνείται ή αποκρύπτει τα αληθινά με έγγραφη υπεύθυνη δήλωση
του άρθρου 8 παρ. 4 Ν. 1599/1986 τιμωρείται με φυλάκιση τουλάχιστον τριών μηνών. Εάν ο υπαίτιος αυτών των πράξεων
σκόπευε να προσπορίσει στον εαυτόν του ή σε άλλον περιουσιακό όφελος βλάπτοντας τρίτον ή σκόπευε να βλάψει άλλον, τιμωρείται με κάθειρξη μέχρι 10 ετών.»

ΠΕΡΙΛΗΨΗ

Η αυξημένη χρήση του Διαδικτύου των Πραγμάτων(IoT) για την ανάπτυξη έξυπνων εφαρμογών χρησιμοποιώντας τεράστιους όγκους δεδομένων ανοίγει νέες ευκαιρίες στην εξαγωγή συμπερασμάτων από δεδομένα και στην υποστήριξη για την αποτελεσματική λήψη αποφάσεων. Για το λόγο αυτό πολλές εφαρμογές έχουν αναπτυχθεί για συλλογή και επεξεργασία δεδομένων. Ένα μεγάλο μέρος αυτών των εφαρμογών έχει αναπτυχθεί σχετικά με τις απαιτήσεις που ορίζουν οι τεράστιες υποδομές του IoT. Ωστόσο οι εφαρμογές πραγματικού χρόνου είναι επιρρεπής στην εμφάνιση ελλιπών τιμών (missing values) γεγονός που αποτελεί μεγάλο πρόβλημα ειδικά σε τέτοιου είδους εφαρμογές. Η ύπαρξη των ελλιπών τιμών μπορεί να επηρεάσει αρνητικά τα αποτελέσματα οποιασδήποτε μορφής επεξεργασίας με αποτέλεσμα να περιορίσουν την απόδοση οποιασδήποτε εφαρμογής IoT. Στην παρούσα εργασία μελετώντας πέρα από τη σχετική βιβλιογραφία προτείνουμε έναν αλγόριθμο αντικατάστασης των ελλιπών τιμών σε ένα σύνολο δεδομένων βασιζόμενοι στα δεδομένα που κατέγραψαν άλλοι κόμβοι σε ένα δίκτυο αποτελούμενο από IoT συσκευές και edge κόμβους. Στόχος μας είναι να υποστηρίξουμε τον υπολογισμό των ελλιπών τιμών με την χρήση της «γνώσης» που παρέχει μια ομάδα IoT συσκευών μέσω τιμών που καταγράφει για διάφορα φαινόμενα. Ο αλγόριθμος που παρουσιάζεται στην παρούσα μελέτη υιοθετεί μια συνεχή μέθοδο ανίχνευσης συσχετισμού ανάμεσα σε δεδομένα πραγματικού χρόνου που παράγονται από τις εμπλεκόμενες συσκευές. Έτσι κάθε τιμή που λείπει μπορεί να αντικατασταθεί από την τελική τιμή που θα εξαχθεί από τα δεδομένα που παρέχονται από τις συσχετισμένες IoT συσκευές. Η μελέτη αυτή περιέχει την περιγραφή του αλγορίθμου που προτείνουμε και την αξιολόγηση του μέσα από ένα μεγάλο πλήθος προσομοιώσεων που υιοθετούν πειραματικά σενάρια.

ABSTRACT

The increased adoption of the Internet of Things (IoT) for the delivery of intelligent applications over huge volumes of data opens new opportunities to draw knowledge from data and support efficient decision making. In recent years, a large set of applications has been developed for data collection and processing. The majority of them have been developed aligned with the requirements of the vast infrastructure of IoT. However, one of the biggest problems occurring at real-time applications is that they are prone to missing values. Missing values can negatively affect the outcomes of any processing activity, thus, they can limit the performance of any IoT application. In this thesis, we depart from the relevant literature and propose a data imputation model that is based on the correlation of data reported by different IoT devices. Our aim is to support data imputation using the 'knowledge' of a team of IoT devices over their reports for various phenomena. Our scheme adopts a continuous correlation detection methodology applied at real time reports of the involved devices. Hence, any missing value can be replaced by the aggregated outcome of data reported by correlated devices. We describe our approach and evaluate our model through a high number of simulations adopting various experimental scenarios.

Table of Contents

ΚΕΦΑΛΑΙΟ 1 Εισαγωγή

1.1 Περιγραφή Προβλήματος

Στις μέρες μας η ραγδαία ανάπτυξη του πλήθους των συσκευών που δραστηριοποιούνται στο Internet of Things (IoT) όπως αισθητήρες, κάμερες, smartwatches κ.α., οι οποίες παράγουν κατανεμημένες ροές δεδομένων, έχει δημιουργήσει αυξημένες απαιτήσεις για γρήγορες και ακριβείς ανταποκρίσεις απέναντι στις δραστηριότητες που επιθυμούν να εκτελέσουν. Το παραπάνω γεγονός γίνεται ακόμη πιο έντονο και εμφανές σε περιπτώσεις εφαρμογών οι οποίες παρουσιάζουν ευαισθησία στο χρόνο με αποτέλεσμα να απαιτούν ανταπόκριση με τη μικρότερη δυνατή καθυστέρηση. Σημαντικό ρόλο για την κάλυψη των παραπάνω απαιτήσεων μπορεί να επιτελέσει η λειτουργία του edge computing. Η χρήση της λειτουργίας του edge computing μπορεί να προσφέρει πολλά πλεονεκτήματα ειδικότερα στην περίπτωση που εφαρμόζεται σε ένα σύνολο κόμβων οι οποίοι αποθηκεύουν τα δεδομένα που λαμβάνουν από τις IoT συσκευές και ταυτόχρονα αποτελούν και τους κόμβους όπου θα εκτελεστούν οι διεργασίες για την επεξεργασία των δεδομένων. Οι κόμβοι αυτοί ονομάζονται edge κόμβοι. Με βάση τα όσα αναφέρθηκαν παραπάνω για τη διαδικασία του edge computing γίνεται εύκολα αντιληπτό το συμπέρασμα ότι η μείωση της απόστασης μεταξύ των πηγών των δεδομένων και των κόμβων διεξαγωγής της επεξεργασίας τους έχει ως αποτέλεσμα την μείωση ή ακόμη και την ελαχιστοποίηση του χρόνου απόκρισης για την παροχή απαντήσεων στα αιτήματα που διατυπώθηκαν από τους χρήστες ή τις εφαρμογές. Έτσι λοιπόν με τη μείωση της απόστασης του υπολογισμού και επεξεργασίας των δεδομένων ώστε αυτή να είναι κοντά στις IoT συσκευές μας δίνεται η δυνατότητα να εξάγουμε συμπεράσματα χρησιμοποιώντας κατανεμημένες ροές δεδομένων. Οι κατανεμημένες αυτές ροές δεδομένων παράγονται από τις IoT συσκευές, οι οποίες κατανέμονται γεωγραφικά γύρω από τους edge κόμβους, και διαμορφώνονται με τέτοιο τρόπο προκειμένου να αποτελέσουν αντικείμενο επεξεργασίας για διάφορες διεργασίες από τους edge κόμβους.

Οι ροές δεδομένων που προωθούνται για επεξεργασία στους edge κόμβους πολλές φορές εμφανίζουν για διάφορους λόγους 'ελαττώματα' όπως 'ανωμαλίες' (outliers), ελλιπείς τιμές (missing values) κ.α. που αποτελούν σοβαρό εμπόδιο και καθιστούν δύσκολη την επεξεργασία των δεδομένων ενώ προκαλούν ανακρίβεια στο τελικό αποτέλεσμα κάνοντας τα συμπεράσματα και τις αναλύσεις που εξάγονται από τα δεδομένα αναξιόπιστα. Η παρούσα εργασία ασχολείται με το πρόβλημα των ελλιπών τιμών και πώς αυτό μπορεί να αντιμετωπιστεί. Απέναντι σε αυτό το πρόβλημα έχει αναπτυχθεί πληθώρα μεθόδων και αλγορίθμων όπως ο αποκλεισμός δεδομένων(data exclusion), η συμπλήρωση των ελλιπών τιμών με τη χρήση μεθόδων όπως αυτή της μέσης τιμής(mean substitution), της τυχαίας αντικατάστασης(replacement at random) κ.α. Οι παραπάνω μέθοδοι συνήθως βασίζονται στην στατιστική πλευρά του προβλήματος προκειμένου να αντικαταστήσουν την τιμή που λείπει. Γενικότερα όμως ανεξαρτήτως της μεθόδου η οποία χρησιμοποιείται για τον υπολογισμό της τιμής που θα αντικαταστήσει την ελλείπουσα τιμή ο στόχος παραμένει ο ίδιος. Η εξάλειψη του σφάλματος ανάμεσα

στην τιμή που υπολογίσαμε να αντικαταστήσει την ελλείπουσα τιμή και την πραγματική άγνωστη τιμή η οποία λείπει.

1.2 Σύντομη Περιγραφή Προτεινόμενου Αλγορίθμου

Στην παρούσα μελέτη παρουσιάζουμε ένα μηχανισμό για την αντιμετώπιση και αντικατάσταση των ελλιπών τιμών. Βασισμένοι από άλλες μεθόδους που αναπτύχθηκαν σε αντίστοιχες μελέτες, οι οποίες αναφέρονται στη βιβλιογραφία, αναπτύξαμε και προτείνουμε ένα νέο μοντέλο που λαμβάνει υπόψιν του τις τιμές των δεδομένων που στάλθηκαν από τις IoT συσκευές με τη μορφή ροών(streams) με βάση την ομοιότητα που αυτές παρουσιάζουν σε σχέση με την ροή που εμφανίζει ελλιπές τιμές. Ο στόχος της μελέτης αυτής είναι η προσέγγιση του προβλήματος η οποία θα βασίζεται στην ομαδοποίηση και τη συνεχή ενημέρωση των δεδομένων με τέτοιο τρόπο ώστε ένα σύνολο ροών δεδομένων (streams) να συνεισφέρει στον υπολογισμό και την αντικατάσταση της τιμής που λείπει. Για το λόγο αυτό στον προτεινόμενο αλγόριθμο για την αντικατάσταση των ελλιπών τιμών, το οποίο θα αναλυθεί εκτενέστερα στα επόμενα κεφάλαια, εισάγουμε έναν μηχανισμό παρακολούθησης και συνεχούς ανίχνευσης της συσχέτισης μεταξύ των ροών που παράγονται από τις IoT συσκευές. Ο μηχανισμός αυτός με την λειτουργία που επιτελεί μας δίνει την δυνατότητα να υποστηρίξουμε μία χωροχρονική προσέγγιση απέναντι στις IoT συσκευές ώστε να εντοπίσουμε αυτές που είναι κατάλληλες για να συμμετέχουν στον υπολογισμό της τιμής που θα αντικαταστήσει την τιμή που λείπει. Προκειμένου να υλοποιηθεί ο παραπάνω μηχανισμός εφαρμόζουμε ευρέως χρησιμοποιούμενες μεθόδους μέτρησης της συσχέτισης με σκοπό να προσφέρουμε ένα συνολικό σχήμα το οποίο θα πάρει μέρος στον τελικό υπολογισμό. Ειδικότερα για την εύρεση της συσχέτισης μεταξύ των IoT συσκευών θα χρησιμοποιηθούν δύο διαφορετικές τεχνικές. Οι τεχνικές αυτές είναι οι Mahalanobis distance [45] και Cosine Similarity [46] τις οποίες θα συνδυάσουμε για να εξάγουμε το μοντέλο εύρεσης για το συσχετισμό μεταξύ των δεδομένων που παράγονται από τις IoT συσκευές.

Πιο συγκεκριμένα η Mahalanobis distance εφαρμόζεται σε ροές δεδομένων που ήδη έχουν σταλεί από τις IoT συσκευές και αποτελεί συντελεστή βαρύτητας ο οποίος εφαρμόζεται στα αποτελέσματα τα οποία προκύπτουν από την εφαρμογή της τεχνικής του Cosine Similarity. Η συσχέτιση μεταξύ των πιο πρόσφατων ροών που στέλνονται από τις IoT συσκευές εκφράζεται μέσω της τιμής του Cosine Similarity. Με αυτόν τον τρόπο μας δίνεται η δυνατότητα να ανιχνεύουμε την ομοιότητα μεταξύ των ροών που στέλνουν οι συσκευές βασιζόμενοι τόσο στην ομοιότητα που παρουσιάζουν την στιγμή που εντοπίζεται μια ελλιπής τιμή όσο και στην ομοιότητα που αυτές παρουσιάζουν ανάμεσα στις ροές που έστειλαν στο παρελθόν.

Στόχος μας είναι να βρούμε συσκευές που να παρουσιάζουν παρόμοιες τιμές ώστε να αντικαθιστούμε την τιμή που λείπει μέσω της συνεργασίας των συσκευών αυτών. Γίνεται προφανές ότι η στρατηγική μας είναι να δίνουμε σημασία και να λαμβάνουμε υπόψιν τις ελλιπές τιμές στους υπολογισμούς μας αφότου όμως τις έχουμε αντικαταστήσει με την τιμή που προκύπτει από τη συνεργασία των συσκευών που παρουσιάζουν παρόμοια συμπεριφορά.

ΚΕΦΑΛΑΙΟ 2 Βιβλιογραφική Επισκόπηση

Στην παράγραφο αυτή θα παρουσιαστούν μερικές μελέτες και δημοσιεύσεις οι οποίες αναφέρονται σε πεδία σχετικά με την αντιμετώπιση ελλιπών τιμών και τη διαχείριση δεδομένων τόσο σε επίπεδο Edge όσο και σε επίπεδο Cloud.

Στον τομέα της διαχείρισης των δεδομένων συναντάμε μεγάλο πλήθος αλγορίθμων οι οποίοι υιοθετούν διαφορετικές προσεγγίσεις και μοντέλα. Στη μελέτη [59] οι συγγραφείς προτείνουν ένα μοντέλο και ένα σχήμα λήψης απόφασης για την αποθήκευση δεδομένων στο επίπεδο του Cloud. Στην [60] η μελέτη αναφέρεται στη διαχείριση δομημένων και αδόμητων δεδομένων που προκύπτουν από το συνδυασμό πολλών βάσεων δεδομένων με τη χρήση της εφαρμογής Hadoop προκειμένου να ικανοποιηθούν οι απαιτήσεις για την αποθήκευσή τους. Εκτός από την σωστή αποθήκευση των δεδομένων με βάση τις απαιτήσεις τους σημαντική είναι και η διασφάλιση τους προκειμένου να αποφευχθούν διάφορες δυσλειτουργίες. Σε αυτήν την κατεύθυνση μπορεί να βοηθήσει η τεχνολογία blockchain που αναφέρεται στην [61]. Στην ίδια κατεύθυνση είναι και η δημοσίευση [62] όπου παρουσιάζεται ένα μοντέλο για την ασφάλεια των δεδομένων σε ένα IoT σύστημα αποθήκευσης. Στο αντίστοιχο κεφάλαιο που αναπτύσσεται παρακάτω παρουσιάζονται και άλλοι αλγόριθμοι διαχείρισης των δεδομένων.

Η επεξεργασία των δεδομένων αποτελεί σημαντικό τομέα για τον οποίο αναπτύσσονται συνεχώς μοντέλα και αλγόριθμοι. Σημαντική διεργασία στον τομέα αυτόν αποτελεί η αντιμετώπιση των ελλιπών τιμών για την οποία έχουν αναπτυχθεί αρκετοί αλγόριθμοι. Στην [63] οι ερευνητές προτείνουν τη χρήση μιας υβριδικής μεθόδου η οποία συνδυάζει τον αλγόριθμο Fuzzy C-Means με ένα μοντέλο Particle Swarm Optimization και μία μηχανή υποστήριξης διανυσμάτων (Support Vector Machine). Ακόμη για την αντιμετώπιση του προβλήματος αυτού άλλα μοντέλα υιοθετούν Perceptron πολλών επιπέδων (Multi-layer Perceptrons) [64], Self-Organizing Maps [65] και πολλά ακόμη μοντέλα που ανήκουν στον τομέα της ασαφούς λογικής (Fuzzy Logic). Στη συνέχεια στο κεφάλαιο για τη διαχείριση των ελλιπών δεδομένων αναφέρονται μερικοί ακόμη αλγόριθμοι για την αντιμετώπιση του φαινομένου αυτού.

ΚΕΦΑΛΑΙΟ 3 Διαδίκτυο των Πραγμάτων (Internet of Things)

3.1 Ορισμός

Ο όρος IoT ή Διαδίκτυο των Πραγμάτων δημιουργήθηκε στα τέλη της δεκαετίας του 1990 από τον Kevin Ashton ο οποίος ήταν μέρος μιας ομάδας ερευνητών όπου μπόρεσαν να βρούνε τρόπο επικοινωνίας μεταξύ αντικειμένων μέσω της χρήσης του διαδικτύου. Με τον όρο IoT αναφερόμαστε γενικότερα σε ένα δίκτυο στο οποίο συνδέονται ολοένα και περισσότερες συσκευές όπως αισθητήρες, κάμερες, smartwatches κ.α. μέσω του οποίου επικοινωνούν και ανταλλάσσουν δεδομένα. Το IoT έχει γνωρίσει τα τελευταία χρόνια μεγάλη ανάπτυξη λόγω της ραγδαίας αύξησης συσκευών που ενσωματώνονται σε αυτό όσο και της δημιουργίας ολοένα και περισσότερων εφαρμογών οι οποίες βασίζονται πάνω σε αυτό. Ωστόσο παρά τη μεγάλη ανάπτυξη που έχει γνωρίσει δεν υπάρχει μοναδικός ορισμός για την διατύπωση του όρου Internet of Things. Ορισμένοι από αυτούς παρατίθενται στη συνέχεια:

- Σύμφωνα με τους Weil και Souissi [1], ορίζεται ως: «Η επέκταση του σημερινού Internet σε όλα τα αντικείμενα που είναι σε θέση να επικοινωνούν, άμεσα ή έμμεσα, με ηλεκτρονικές συσκευές που συνδέονται στο διαδίκτυο».
- Σε μια ειδική έκθεση για το IoT η οποία δημοσιεύθηκε τον Μάρτιο του 2014 (IEEE, “Internet of Things”, 2014), η IEEE περιγράφει τον όρο “Internet of Things” ως: «Ένα δίκτυο αντικειμένων – το καθένα ενσωματωμένο με αισθητήρες – τα οποία είναι συνδεδεμένα στο Internet.[2]
- Η ITU-T Study Group 13 που καθοδηγεί τις μελέτες της ITU σχετικά με τα πρότυπα για τα δίκτυα επόμενης γενιάς (NGN) και μελλοντικά δίκτυα(ITU, SERIES Y, 2005). Έχει ορίσει το IoT ως εξής: «Μια παγκόσμια υποδομή για την κοινωνία της πληροφορίας, η οποία επιτρέπει προηγμένες υπηρεσίες μέσω διασύνδεσης (φυσικών και εικονικών) αντικειμένων που βασίζονται σε υπάρχουσες και εξελισσόμενες διαλειτουργικές τεχνολογίες πληροφοριών και επικοινωνιών.

Ένα ακόμη αποτέλεσμα της ευρείας χρήσης του IoT από δισεκατομμύρια συσκευές που επικοινωνούν και αλληλοεπιδρούν μεταξύ τους είναι η δημιουργία ενός τεράστιου όγκου πληροφοριών. Ωστόσο οι συσκευές που αποτελούν αλλά και εκείνες που, ολοένα και με μεγαλύτερο ρυθμό, θα περιλαμβάνονται στο IoT παρουσιάζουν μεγάλες διαφορές μεταξύ τους τόσο στο κομμάτι που αφορά στο υλικό (Hardware) όσο και στο κομμάτι που αφορά στο λογισμικό (Software).

3.2 Αρχιτεκτονική του IoT

Στο παρόν υποκεφάλαιο παρουσιάζεται η δομή του IoT η οποία αποτελείται από τέσσερα βασικά επίπεδα. Στη συνέχεια δίνεται μία σύντομη περιγραφή των επίπεδων αυτών.

- Επίπεδο IoT συσκευών (Things Layer): Στο πρώτο επίπεδο του IoT βρίσκονται οι αισθητήρες, οι ενεργοποιητές και όλες εκείνες οι συσκευές οι οποίες καταγράφουν δεδομένα και συλλέγουν πληροφορίες.
- Επίπεδο 2 (Gateway and data acquisition): Το επίπεδο αυτό είναι επιφορτισμένο με τη λειτουργία να μετατρέπει τις πληροφορίες που δέχεται από ένα σύνολο διαφορετικών IoT συσκευών σε κατάλληλη μορφή για επεξεργασία. Ακόμη είναι υπεύθυνο για τη μεταφορά των πληροφοριών και των δεδομένων που καταγράφηκαν από τις IoT συσκευές στο επόμενο επίπεδο.
- Επίπεδο 3 (Edge Analytics): Στο επίπεδο αυτό διεξάγεται η επεξεργασία και διαχείριση των δεδομένων τα οποία έφτασαν από τις IoT συσκευές μέσω του προηγούμενου επιπέδου. Ειδικότερα στο επίπεδο αυτό επεξεργάζονται τα δεδομένα προκειμένου να δοθεί απόκριση ώστε να εκτελέσουν οι IoT εφαρμογές την λειτουργία τους. Τέλος το επίπεδο αυτό μεταφέρει στο Cloud τα δεδομένα που προορίζονται για περεταίρω επεξεργασία.
- Επίπεδο 4 (Cloud Analytics): Στο επίπεδο αυτό μεταφέρονται και επεξεργάζονται τα δεδομένα από το τρίτο επίπεδο που απαιτούν περεταίρω επεξεργασία για εξαγωγή γνώσης και αποτελεσμάτων.

3.3 IoT Hardware

Στην παρούσα ενότητα θα παρουσιάσουμε βασικές πλατφόρμες υλικού που χρησιμοποιούνται σήμερα όπως το Arduino και το Raspberry καθώς και άλλες μικρότερες συσκευές όπως αισθητήρες(sensors) και ενεργοποιητές (actuators) κ.α. που αποτελούν και αυτές μέρος του IoT hardware.

3.3.α IoT Συσκευές

Το υλικό που χρησιμοποιείται σε IoT συστήματα ποικίλει περιλαμβάνοντας πλήθος διαφόρων συσκευών όπως controllers, κάμερες κ.α. προκειμένου να επιτελούνται διάφορες λειτουργίες όπως εκείνη της επικοινωνίας και της ασφάλειας για την επίτευξη συγκεκριμένων ενεργειών.

1. <u>Αισθητήρες (Sensors)</u>

Με τον όρο αισθητήρας περιγράφουμε μία συσκευή η οποία είναι ικανή να ανιχνεύσει, να μετρήσει και να δηλώσει μια μεταβολή σε ένα περιβάλλον όπως θερμοκρασία, υγρασία, ακτινοβολία κ.α. ακόμη ένας αισθητήρας μπορεί να αποτελείται και από modules ενέργειας, διαχείρισης ισχύος και ραδιοσυχνότητας. Οι αισθητήρες υπάρχουν πολύ πριν τη δημιουργία του όρου IoT και αποτελούν ένα από τα πιο σημαντικά συστατικά στοιχεία για την ανάπτυξη IoT εφαρμογών και κατέχουν ένα μεγάλο μέρος από τις IoT συσκευές.

Μερικά παραδείγματα αισθητήρων είναι οι εξής:

- Αισθητήρες θερμοκρασίας

- Αισθητήρες ήχου
- Αισθητήρες πίεσης
- Αισθητήρες φωτός
- Αισθητήρες κίνησης

2. Ενεργοποιητές (Actuators)

Οι ενεργοποιητές όπως και οι αισθητήρες χρησιμοποιούνταν πολύ πριν εγκαθιδρυθεί ο όρος του IoT. Ενεργοποιητή χαρακτηρίζουμε μία συσκευή η οποία λαμβάνει ένα σήμα και θέτει σε κίνηση το αντικείμενο που πρέπει να εκτελέσει μία ενέργεια. Οι ενεργοποιητές εκτελούν μία ενέργεια ανάλογα με τα δεδομένα τα οποία συλλέγονται από τους αισθητήρες. Μερικές κατηγορίες ενεργοποιητών παρουσιάζονται παρακάτω:

- Υδραυλικοί ενεργοποιητές όπως υδραυλικοί κινητήρες.
- Ηλεκτρικοί ενεργοποιητές όπως φωνητικά πηνία

3. Συσκευές που μπορούν να φορεθούν (Wearables)

Μια κατηγορία IoT συσκευών που έχει γνωρίσει μεγάλη αύξηση τα τελευταία χρόνια είναι οι συσκευές που μπορούν να φορεθούν. Οι συσκευές αυτές είναι μικρές σε μέγεθος και μπορούν να φορεθούν σε πολλά μέρη του σώματος προκειμένου να δώσουν τις απαραίτητες πληροφορίες. Τα smartwatches που μπορούν να δίνουν πληροφορίες για τόσο για αυτόν που τα φορά όσο και για το περιβάλλον, τα smart glasses για να απολαμβάνουμε περισσότερο μία δραστηριότητα, είδη ρουχισμού κ.α.

4. Συσκευές ευρείας χρήσης

Στις συσκευές IoT συγκαταλέγονται και οι ηλεκτρονικοί υπολογιστές όπως desktop και laptop καθώς και άλλες μικρότερες συσκευές καθημερινής χρήσης όπως tablet και κινητά τηλέφωνα αλλά και περιφερειακές συσκευές όπως routers και switchers.

3.3.β IoT Πλακέτες

Στο υλικό που χρησιμοποιείται για τις IoT εφαρμογές εκτός από τις συσκευές που αναφέρθηκαν στην προηγούμενη παράγραφο συγκαταλέγονται και οι πλατφόρμες στις οποίες αναπτύσσονται IoT εφαρμογές. Στην παράγραφο αυτήν περιγράφονται μερικές από τις πιο γνωστές και ευρέως διαδεδομένες IoT πλακέτες που χρησιμοποιούνται στην ανάπτυξη IoT εφαρμογών.

1. Arduino Uno [4],[5],[11],[12]

Το Arduino δημιουργήθηκε το 2005 από δύο Ιταλούς μηχανικούς για εκπαιδευτικούς σκοπούς και είναι μια πλακέτα με ενσωματωμένο μικροελεγκτή που έχει σχεδιαστεί με τέτοιο τρόπο ώστε να μπορεί να αντιλαμβάνεται και να αλληλοεπιδρά με το περιβάλλον σε συνεργασία με άλλους μικροελεγκτές (microcontrollers), αισθητήρες και ενεργοποιητές όπως αυτούς που περιγράψαμε παραπάνω. Ένα από τα πιο δημοφιλή μοντέλα που έχουν αναπτυχθεί είναι το Arduino Uno εκτός από αυτό όμως υπάρχουν και άλλα μοντέλα όπως το Arduino Zero, Arduino Leonardo, Arduino 101, Arduino Nano κ.α..

Το Arduino Uno βασίζεται σε ένα chip τύπου ATmega328. Ακόμη περιέχει δεκατέσσερα ψηφιακά pins για είσοδο και έξοδο με τα έξι από αυτά να μπορούν να χρησιμοποιηθούν και ως έξοδοι τύπου PWM, έναν κεραμικό συντονιστή(resonator) συχνότητας 16 MHz, μία κεφαλή τύπου ICSP, μια σύνδεση USB, έξι αναλογικές εισόδους, θύρα τροφοδοσίας και κουμπί επαναφοράς. Επίσης το Arduino Uno περιέχει ένα προεγκατεστημένο λογισμικό που ονομάζεται Arduino IDE το οποίο χρησιμοποιείται προκειμένου να γράψουμε και να αποθηκεύσουμε τον κώδικα μας από τον υπολογιστή στην πλατφόρμα του Arduino. Εκτός από τα παραπάνω χαρακτηριστικά το Arduino Uno έχει ακόμη και τις εξής δυνατότητες:

- Μπορεί να διαβάσει αναλογικά και ψηφιακά σήματα από διαφορετικούς αισθητήρες προκειμένου να τα επεξεργαστεί και να εκτελέσει μία ενέργεια όπως την περιστροφή ενός σερβοκινητήρα, το σβήσιμο ενός LED λαμπτήρα κ.α.
- Διαθέτει 32 KB flash μνήμης για την αποθήκευση του κώδικα.
- Ο έλεγχος των λειτουργιών της πλατφόρμας μέσω ενός συνόλου εντολών κάνοντας χρήση του Arduino IDE.
- Σε αντίθεση με άλλες πλατφόρμες ο κώδικας μπορεί να αποθηκευτεί σε αυτό συνδέοντας το με τον υπολογιστή μέσω μόνο ενός USB καλωδίου.
- Χρησιμοποιεί μια απλοποιημένη έκδοση της C++ κάνοντας έτσι ευκολότερο τον προγραμματισμό του.

Στην Εικόνα 1 που ακολουθεί φαίνεται η έκδοση Arduino Uno.

Εικόνα 1: Arduino Uno

2. Raspberry Pi Board [13],[14]

Το Raspberry Pi δημιουργήθηκε το 2012 από την Raspberry pi foundation στο Ηνωμένο Βασίλειο ως ένας μικροϋπολογιστής χαμηλού κόστους για μαθητές και παιδιά. Αποτελεί μία πλακέτα μικρού μεγέθους που μπορεί να χρησιμοποιηθεί για πολλούς τύπους διεργασιών που εκτελούνται και σε έναν υπολογιστή όπως η επεξεργασία κειμένου, προβολή βίντεο κ.α.. Οι πρώτες πλακέτες Raspberry Pi περιλάμβαναν μνήμη RAM, επεξεργαστή (CPU), κάρτα γραφικών (GPU), θύρα Ethernet, GPIO pins, υποδοχή τύπου XBee για ασύρματες επικοινωνίες, θύρα τύπου UART που είναι μια σειριακή θύρα εισόδου/εξόδου, θύρα τροφοδοσίας και διάφορες θύρες για σύνδεση με εξωτερικές συσκευές. Για αποθηκευτικό χώρο απαιτούνταν μια κάρτα αποθήκευσης τύπου SD στην οποία γινόταν η εγκατάσταση

του λογισμικού, το οποίο ήταν συνήθως κάποια διανομή Linux. Η τελευταία έκδοση που κυκλοφόρησε στις 24 Ιουνίου 2019 προσφέρει ακόμη πιο πολλές δυνατότητες. Στη συνέχεια παρουσιάζονται τα χαρακτηριστικά της τελευταίας έκδοσης Raspberry Pi 4 τα οποία είναι:

- Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
- 1GB, 2GB ή 4GB LPDDR4-3200 SDRAM αναλόγως το μοντέλο.
- 2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0, BLE
- Θύρα Gigabit Ethernet
- 2 θύρες USB 3.0
- 2 θύρες USB 2.0
- Raspberry Pi standard 40 pin GPIO header
- 2 θύρες τύπου micro-HDMI
- Θύρα 2-lane MIPI DSI display
- Θύρα 2-lane MIPI CSI camera
- Θύρα 4-pole stereo audio and composite video
- H.265 (4kp60 decode), H264 (1080p60 decode, 1080p30 encode)
- OpenGL ES 3.0 κάρτα γραφικών
- Θύρα για κάρτα αποθήκευσης τύπου Micro-SD για την εγκατάσταση του λογισμικού και για χώρο αποθήκευσης

Στην Εικόνα 2 απεικονίζεται η έκδοση Raspberry Pi 4.

Εικόνα 2: Raspberry Pi 4

3. Intel Edison [15],[16]

Μία ακόμη ισχυρή πλακέτα για την ανάπτυξη απαιτητικών IoT εφαρμογών είναι η Intel Edison. Η πλακέτα έχει δύο εκδόσεις, μία συμβατή με το Arduino και μία έκδοση σε μικρότερο μέγεθος για την εύκολη ανάπτυξη δοκιμαστικών εφαρμογών. Η πλακέτα είναι εφοδιασμένη με ένα διπύρηνο επεξεργαστή Intel Atom χρονισμένος στα 500 Mhz και ένα μικροελεγκτή χρονισμένο στα 100 Mhz. Ακόμη χρησιμοποιεί μνήμη 1 GB RAM, 4 GB αποθηκευτικού χώρου ενώ έχει ακόμη δυνατότητες σύνδεσης WiFi και Bluetooth 4.0. Τέλος έχει 40 GPIO pins για σύνδεση με άλλες IoT συσκευές. Η πλακέτα Intel Edison έχει εγκατεστημένο ένα λειτουργικό σύστημα που ονομάζεται Yocto και είναι μία διανομή Linux. Τέλος είναι από τις λίγες πλακέτες που έχουν πιστοποιηθεί από την Microsoft, τις AWS (Amazon Web Services) και την IBM για την διασύνδεση στο cloud.

Στην Εικόνα 3 παρουσιάζεται η πλακέτα Intel Edison.

Εικόνα 3: Intel Edison

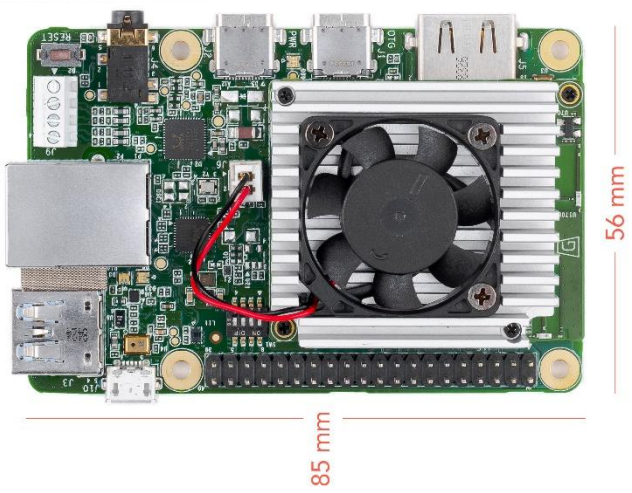
4. Coral TPUs

Μία ακόμη πλακέτα που υποστηρίζει την ανάπτυξη τέτοιου είδους εφαρμογών είναι το Coral Dev Board. Πρόκειται για ένα μικροϋπολογιστή ο οποίος μπορεί να υποστηρίξει τη γρήγορη εκμάθηση μηχανικής μάθησης (machine learning) σε μικρότερους τομείς. Ειδικότερα το Coral επιτρέπει τη δημιουργία γρήγορων, αποτελεσματικών και ισχυρών τοπικά κατανεμημένων εφαρμογών διασφαλίζοντας με αυτόν τον τρόπο την ιδιοτικότητα των δεδομένων. Μερικά παραδείγματα εφαρμογών που μπορούν να δημιουργηθούν με τη χρήση της πλακέτας αυτής και αλγορίθμων μηχανικής μάθησης είναι ο εντοπισμός ή η αναγνώριση αντικειμένων σε μία εικόνα, η εκτίμηση της θέσης ενός ατόμου προσδιορίζοντας διάφορες αρθρώσεις του σώματος κ.α.. Ακόμη μπορεί να χρησιμοποιηθεί τόσο στη βιομηχανία προκειμένου να κάνει πιο αποτελεσματική μια εργασία ή και στην δημιουργία «έξυπνων» πόλεων (smart cities) δίνοντας λύσεις σε διάφορα προβλήματα [47]. Στη συνέχεια παρουσιάζονται ορισμένα χαρακτηριστικά της πλακέτας όπως αυτά αναφέρονται στην ιστοσελίδα του[48]:

- Edge TPU System-on-Module (SoM)
- NXP i.MX 8M SoC (Quad-core Arm Cortex-A53, plus Cortex-M4F)
- Google Edge TPU ML accelerator coprocessor
- Cryptographic coprocessor
- Wi-Fi 2x2 MIMO (802.11b/g/n/ac 2.4/5 GHz)
- Bluetooth 4.2
- 8 GB eMMC
- 1 GB LPDDR4
- USB connections
- USB Type-C power port (5 V DC)
- USB 3.0 Type-C OTG port
- USB 3.0 Type-A host port
- USB 2.0 Micro-B serial console port
- Audio connections
- 3.5 mm audio jack (CTIA compliant)
- Digital PDM microphone (x2)
- 2.54 mm 4-pin terminal for stereo speakers
- Video connections
- HDMI 2.0a (full size)
- 39-pin FFC connector for MIPI DSI display (4-lane)
- 24-pin FFC connector for MIPI CSI-2 camera (4-lane)
- MicroSD card slot
- Gigabit Ethernet port
- 40-pin GPIO expansion header

- Supports Mendel Linux (derivative of Debian)

Η Εικόνα που ακολουθεί παρουσιάζει την πλακέτα Coral TPUs.



Εικόνα 4: Coral TPUs

3.3 IoT Software

Η παρούσα παράγραφος περιέχει μία σύντομη περιγραφή στα κύρια λειτουργικά συστήματα που χρησιμοποιούνται στις IoT συσκευές. Τα λειτουργικά συστήματα που αναφέρονται παρακάτω έχουν σχεδιαστεί να λειτουργούν σύμφωνα με τις δυνατότητες των IoT συσκευών [9], [10].

1. Contiki

Το Contiki δημιουργήθηκε το 2002 και αποτελεί ένα από τα κύρια και δημοφιλέστερα λειτουργικά συστήματα για IoT συσκευές όπως μικροελεγκτές χαμηλής ισχύος που μπορούν να λειτουργούν αποτελεσματικά με τη χρήση πρωτόκολλων όπως το IPv4 και το IPv6. Μερικά βασικά χαρακτηριστικά αυτού του λειτουργικού συστήματος είναι:

- Δυνατότητα Multitasking περιλαμβάνοντας μια ενσωματωμένη σουίτα διαδικτυακού πρωτοκόλλου.
- Απαιτούνται μόνο 10 KB μνήμης RAM και 30 KB ROM για την εκτέλεση του λειτουργικού συστήματος.
- Η βασική γλώσσας του λειτουργικού είναι η γλώσσα C.
- Πριν την ανάπτυξη σε πραγματικό χρόνο των IoT εφαρμογών δοκιμάζονται χωριστά σε έναν προσομοιωτή που ονομάζεται Cooja.
- Μικρή απαίτηση ενεργειακής ισχύος.
- Η χρήση ενός μηχανισμού που αποκαλείται protothreads προκειμένου να εξοικονομηθεί μνήμη
- Υποστήριξη των πρωτοκόλλων IPv4 και IPv6 που αποτελούνται από τα πρωτόκολλα TCP, UDP και HTTP.

2. Raspbian

Όπως αναφέρθηκε και παραπάνω μια από τις πιο διαδεδομένες και πολυχρησιμοποιούμενες πλακέτες στον χώρο των IoT συσκευών είναι το Raspberry το οποίο έχει και το δικό του λειτουργικό σύστημα που ονομάζεται Raspbian. Η ανάπτυξη του είναι βασισμένη στο λειτουργικό σύστημα Debian Linux κάτι το οποίο διαφαίνεται από τις ομοιότητες που παρουσιάζει στον πυρήνα (kernel) του λειτουργικού. Ακόμη το Raspbian χρησιμοποιεί το PIXEL PI ως περιβάλλον εργασίας και περιέχει προεγκατεστημένες πλήθος εφαρμογών όπως το "Mathematica" κ.α.. Το Raspbian κυκλοφορεί σε δύο εκδόσεις το Raspbian Buster και το Raspbian Stretch. Το λειτουργικό αυτό αναπτύσσεται συνεχώς καθώς η ζήτηση για αυτό το λειτουργικό σύστημα συνεχώς αυξάνεται.

3. Windows 10 IoT

Ένα άλλο λειτουργικό σύστημα που έχει αναπτυχθεί τελευταία για τις συσκευές IoT είναι τα Windows 10 IoT που αναπτύχθηκε από τη Microsoft. Τα Windows 10 IoT είναι μέρος της οικογένειας των εκδόσεων του λειτουργικού συστήματος Windows 10 το οποίο αφορά όμως τον τομέα των IoT συσκευών και δεν αποτελεί λειτουργικό σύστημα ανοιχτού κώδικα. Το λειτουργικό σύστημα αυτό χωρίζεται σε δύο εκδόσεις. Οι εκδόσεις αυτές είναι οι Windows 10 IoT core η οποία έχει σχεδιαστεί για μικρές συσκευές IoT και ενσωματωμένες συσκευές και Windows 10 IoT Enterprise το οποίο έχει σχεδιαστεί για εφαρμογή στον βιομηχανικό τομέα. Οι εκδόσεις αυτές εκμεταλλεύονται τη συνδεσιμότητα που προσφέρει το IoT και το cloud ώστε να παρέχουν δυνατότητα σύνδεσης με IoT συσκευές. Ακολουθούν κάποια χαρακτηριστικά των δύο εκδόσεων του λειτουργικού συστήματος Windows 10 IoT.

- Η έκδοση Windows 10 IoT Enterprise λειτουργεί σε επεξεργαστή τύπου ARM.
- Η έκδοση Windows 10 IoT core δεν υποστηρίζει τις λειτουργίες Cortana και FileOpenPicker οι οποίες είναι διαθέσιμες στα Windows 10.
- Η έκδοση Windows 10 IoT core παρέχει διαχειρισιμότητα όπως το λειτουργικό σύστημα Windows 10 παρόλο που λειτουργεί ως εφαρμογή.

4. RIOT

Το RIOT ανήκει στην κατηγορία των δωρεάν λειτουργικών συστημάτων ανοικτού κώδικα που έχουν σχεδιαστεί για IoT συσκευές. Το λειτουργικό σύστημα RIOT έχει μια μεγάλη κοινότητα ανάπτυξης και κυκλοφόρησε με άδεια τύπου Lesser General Public License (LGPL). Επιτρέπει τη δημιουργία εφαρμογών με χρήση των γλωσσών προγραμματισμού C και C++ και παρέχει πλήρης δυνατότητες πολυνημάτωσης (multi-threading), πρωτόκολλά δικτύωσης TCP, UDP και CoAp και βιβλιοθήκες τύπου SSL/TSL. Το λειτουργικό σύστημα RIOT συνήθως εγκαθίσταται σε επεξεργαστές μεγέθους 8-bit (AVR Atmega), 16-bit (TI MSP430) και 32 bit (ARM Cortex). Ακόμη μια θύρα του λειτουργικού συστήματος επιτρέπει την εκτέλεση του ως μία διεργασία σε περιβάλλον Linux ή MacOS. Τέλος επιτρέπει την χρήση εργαλείων ανάπτυξης εφαρμογών και αποσφαλμάτωσης όπως GNU Compiler Collection, GNU Debugger, Wireshark κ.α..

5. TinyOS

Το λειτουργικό σύστημα Tiny OS είναι ένα δωρεάν, βασισμένο σε components, ανοικτού κώδικα λειτουργικό σύστημα σχεδιασμένο για ασύρματες IoT συσκευές χαμηλής ισχύος και δίκτυα αισθητήρων. Η ομάδα που το δημιούργησε ονομάζεται TinyOS Alliance. Η γλώσσα στην οποία έχει δημιουργηθεί ονομάζεται nesC (Network Embedded Systems C) η οποία είναι μία έκδοση της C σχεδιασμένη με τέτοιον τρόπο ώστε να υποστηρίζει διάφορα components και τη λειτουργία του ταυτοχρονισμού. Η σχεδίαση του επιτρέπει την υποστήριξη εντατικών ταυτόχρονων λειτουργιών, που απαιτούνται από τα δίκτυα αισθητήρων, με τις μικρότερες απαιτήσεις σε υλικό. Το λειτουργικό σύστημα TinyOs χρησιμοποιείται ευρέως συμβάλλοντας με αυτόν τον τρόπο στην προσομοίωση αλγορίθμων και πρωτοκόλλων.

6.Amazon FreeRTOS

Το Amazon FreeRTOS αποτελεί ένα λειτουργικό σύστημα ανοικτού κώδικα για μικροελεγκτές για την ανάπτυξη IoT εφαρμογών που δημιουργήθηκε από την Amazon. Οι βιβλιοθήκες που διαθέτει το λογισμικό αυτό διευκολύνει τη σύνδεση με μικρές συσκευές IoT και την ασφαλή μεταφορά δεδομένων μεταξύ των συσκευών αυτών. Για τη διασύνδεση μεταξύ των συσκευών γίνεται χρήση βιβλιοθηκών όπως οι MQTT, HTTP,WiFi Mgmt, Bluetooth Low Energy Mgmt κ.α. ενώ για την ασφάλεια των δεδομένων χρησιμοποιεί πρωτόκολλα κρυπτογράφησης όπως το TLS και το PKCS#11. Επίσης χρησιμοποιεί την υπηρεσία cloud της Amazon Web Service που ονομάζεται AWS IoT Core προκειμένου να εκτελεί IoT εφαρμογές. Τα τελευταία χρόνια έχει γίνει ένα από τα κύρια λειτουργικά συστήματα που χρησιμοποιούνται από μικροελεγκτές.

ΚΕΦΑΛΑΙΟ 4 Διαχείριση Δεδομένων στις Παρυφές του Δικτύου (Data Management at the Edge)

4.1 Edge Computing

4.1.α Ορισμός Edge Computing

Η μεγάλη ανάπτυξη του IoT και των συσκευών που συμμετέχουν σε αυτό έχει ως αποτέλεσμα την ανάγκη για δημιουργία τεχνολογιών που να βοηθήσουν αποτελεσματικά στη διαχείριση και επεξεργασία του τεράστιου όγκου δεδομένων που δημιουργούνται από τις συσκευές αυτές. Μία τέτοια τεχνολογία είναι και το edge computing. Με τον όρο edge computing αναφερόμαστε στην προσέγγιση κατά την οποία οι διαδικασίες επεξεργασίας των δεδομένων καθώς και η αποθήκευση των απαραίτητων πληροφοριών και δεδομένων γίνονται όσο το δυνατότερο πιο κοντά στην πηγή των δεδομένων προκειμένου να ελαττωθεί η καθυστέρηση στην επιστροφή των αποτελεσμάτων και πληροφοριών που έχουν ζητηθεί από διάφορες συσκευές ή εφαρμογές. Ειδικότερα το λεξικό Gartner [17] ορίζει το edge computing ως « Το Edge Computing είναι μέρος μιας κατανεμημένης υπολογιστικής τοπολογίας όπου η επεξεργασία των πληροφοριών τοποθετείται κοντά στις παρυφές του δικτύου (edge), εκεί που αντικείμενα και άνθρωποι παράγουν ή καταναλώνουν εκείνες τις πληροφορίες ».

4.1.β Πλεονεκτήματα Edge computing

Η δημιουργία ολοένα και περισσότερων εφαρμογών με τη χρήση IoT συσκευών προκάλεσε τη δημιουργία μιας σειράς απαιτήσεων για καλύτερες αποδόσεις των εφαρμογών και την εξασφάλιση της καλύτερης λειτουργίας τους. Η λύση σε πολλές από τις παραπάνω απαιτήσεις προσφέρθηκε με την χρήση του edge computing μέσω των πλεονεκτημάτων που διαθέτει. Στη συνέχεια παρουσιάζονται μερικά από τα πλεονεκτήματα τα οποία εδραίωσαν το edge computing ως την πιο ευρέως χρησιμοποιούμενη τεχνολογία στο χώρο των IoT εφαρμογών.

1. **Ελαχιστοποίηση του χρόνου απόκρισης**: Οι υπηρεσίες για την επεξεργασία των δεδομένων τοποθετούνται πιο κοντά στις πηγές των πληροφοριών μειώνοντας την απόσταση που πρέπει να ταξιδέψουν τα δεδομένα. Έτσι οι απαντήσεις ή τα αποτελέσματα που περιμένουν οι χρήστες είναι πιο γρήγορα διαθέσιμες.

2.**Μείωση του κόστους**: Το edge computing επιτρέπει τη μείωση των δεδομένων που θα στέλνονται και θα αποθηκεύονται στο Cloud. Με αυτόν τον τρόπο μειώνεται το κόστος για τη δημιουργία IoT εφαρμογών μεταφέροντας σε αντίθεση με το Cloud Computing την επεξεργασία των δεδομένων ένα επίπεδο χαμηλότερα.

3.**Επεκτασιμότητα**: Ένα μεγάλο πλεονέκτημα που προσφέρει η χρήση του edge computing είναι η επεκτασιμότητα. Ειδικότερα η αύξηση των IoT συσκευών σε ένα σύστημα που κάνει χρήση της τεχνολογίας του edge computing δεν απαιτεί αλλαγές στις δυνατότητες του Cloud όπως για παράδειγμα η αύξηση του εύρους

ζώνης (bandwidth) καθώς το μεγαλύτερο μέρος της επεξεργασίας και αποθήκευσης των πληροφοριών γίνεται στο επίπεδο του Edge.

4.1.γ Διεργασίες που εκτελούνται στις παρυφές του δικτύου

Στο επίπεδο του edge εφαρμόζονται ένα πλήθος σημαντικών διεργασιών που αφορά τόσο την αποθήκευση όσο και την επεξεργασία των δεδομένων που καταγράφονται από IoT συσκευές. Γενικότερα το επίπεδο του edge ενσωματώνει όλες εκείνες τις τεχνολογίες και μπορεί να εκτελεστεί σε αυτό κάθε μορφής διεργασία που εκτελείται και στο Cloud. Αυτό είναι δυνατό γιατί η κύρια διαφορά μεταξύ των δύο αυτών είναι η απόσταση στην οποία βρίσκονται από τις πηγές των δεδομένων. Παρακάτω θα αναφερθούν ορισμένες από τις πιο γνωστές και δημοφιλείς διεργασίες που εκτελούνται πάνω στα δεδομένα στο επίπεδο του edge.

- **Ταξινόμηση (Classification)**: Η διαδικασία της ταξινόμησης των δεδομένων είναι μία από τις πιο διαδεδομένες τεχνικές για την εξόρυξη γνώσης από δεδομένα. Αποτελείται από δύο στάδια τη διαδικασία της εκμάθησης (learning) και από τη διαδικασία της δοκιμής (testing). Τόσο στο πρώτο όσο και στο δεύτερο στάδιο χρησιμοποιούνται δεδομένα προκειμένου να εκπαιδευτεί και να δοκιμαστεί ο αλγόριθμος ταξινόμησης που χρησιμοποιούμε. Μερικοί τύποι μοντέλων που χρησιμοποιούνται για ταξινόμηση των δεδομένων είναι:
    - **Δένδρα απόφασης (Decision trees)**
    - **Νευρωνικά Δίκτυα (Neural Networks)**
    - **Bayesian Ταξινόμηση**
    - **Μηχανές διανυσμάτων υποστήριξης (Support Vector Machines)**
- **Ομαδοποίηση (Clustering):** Η διαδικασία αυτή προσφέρει την δυνατότητα να ομαδοποιούμε τα δεδομένα βάση κάποιων κοινών χαρακτηριστικών και να εξάγουμε πιθανές συσχετίσεις μεταξύ αυτών των δεδομένων που ανήκουν στην ίδια ομάδα. Ορισμένοι αλγόριθμοι ομαδοποίησης δεδομένων είναι:
    - **CHAMELEON**
    - **DBSCAN**
    - **K-MEANS**
    - **FUZZY-K-MEANS**
- **Εντοπισμός ανώμαλων τιμών (Outlier detection):** Η διαδικασία αυτή όπως και η εξάλειψη των ελλιπών τιμών από ένα σύνολο δεδομένων αποτελούν δύο από τις σημαντικότερες διεργασίες προκειμένου να μπορούμε να εξάγουμε ακριβή συμπεράσματα και αποτελέσματα από την επεξεργασία των δεδομένων. Για την επίτευξη αυτού του στόχου έχουν αναπτυχθεί πολλοί αλγόριθμοι και τεχνικές μερικές από τις οποίες είναι:
    - **Κρυμμένα Μοντέλα Μαρκόφ (Hidden Markov Models)**
    - **Τοπικός παράγοντας ανωμαλίας (Local Outlier Factor)**
    - **Δάσος Απομόνωσης (Isolation Forest)**
    - **Κ-Πλησιέστεροι Γείτονες (k-Nearest Neighbors)**

4.2 Διαχείριση δεδομένων: μελέτες και μοντέλα.

Η διαχείριση του όγκου δεδομένων που δημιουργούνται από τις IoT συσκευές αποτελεί ένα συνεχώς αναπτυσσόμενο ερευνητικό πεδίο καθώς ολοένα και περισσότερες προσεγγίσεις αναπτύσσονται για λειτουργία αυτή. Στην συνέχεια θα γίνουν αναφορές σε διάφορες τέτοιες προσεγγίσεις που αναπτύχθηκαν από την ευρύτερη ερευνητική κοινότητα. Οι παρακάτω αναφορές αφορούν κυρίως μεθόδους διαχείρισης δεδομένων στο edge επίπεδο αλλά και σε fog και cloud επίπεδο.

Στην δημοσίευση [31] περιγράφεται η ανάπτυξη ενός μοντέλου διαχείρισης των δεδομένων για υπολογιστικά περιβάλλοντα στο edge επίπεδο. Ειδικότερα περιγράφεται ένας πολυεπίπεδος μηχανισμός χρονοπρογραμματισμού που αφού πρώτα ταξινομήσει τα δεδομένα που είναι προς επεξεργασία σε κατάλληλους διοχετευτές πληροφοριών τοποθετεί τις διεργασίες σύμφωνα με τα δεδομένα που χρειάζονται για την εκτέλεση τους και την κατανομή των δεδομένων που έχει γίνει. Η επόμενη μελέτη [32] έχει να κάνει με την διαμόρφωση μιας αρχιτεκτονικής για τον συντονισμό των εργασιών εξόρυξης δεδομένων μεταξύ edge και cloud αποθηκευτικών δομών. Η επίτευξη της αρχιτεκτονικής αυτής βασίζεται στην δημιουργία μοντέλων που εφαρμόζονται κεντρικά αλλά και κατανεμημένα σε edge συσκευές. Τα δεδομένα που καταγράφονται αποθηκεύονται και επεξεργάζονται σε πρώτη φάση τοπικά και σε περίπτωση περαιτέρω επεξεργασίας μπορούν να αναλυθούν στο edge επίπεδο. Η [33] αναφέρεται στη δημιουργία ενός μηχανισμού τριών επιπέδων για την διαχείριση των δεδομένων σε επίπεδο edge, fog και cloud αρχιτεκτονικής για ετερογενείς, γεωγραφικά διασκορπισμένων IoMT συσκευών καταγραφής ροών δεδομένων. Στην μελέτη [49] οι ερευνητές παρουσιάζουν κατάλληλες δομές δεδομένων και αλγορίθμους που είναι ικανοί να επιλύσουν το πρόβλημα της διαλογής (filtering) δεδομένων από μεγάλες βάσεις δεδομένων. Ειδικότερα ασχολείται, όπως αναφέρεται, με την επίλυση του προβλήματος της ομοιότητας ενός αρχείου d με ένα πλήθος q αρχείων από σε μία βάση δεδομένων. Η ομαδοποίηση (clustering) των δεδομένων αποτελεί μία από τις πιο σημαντικές διεργασίες πάνω σε ένα σύνολο δεδομένων που μπορεί να υλοποιηθεί στο επίπεδο του edge computing. Στην [50] μελετώνται ανά κατηγορίες αλγόριθμοι ομαδοποίησης όπως οι k-means, k-modes, PAM, CLARA, CLARANS, Fuzzy-C-means, DBSCAN κ.α. σημαντικοί αλγόριθμοι που χρησιμοποιούνται στην ομαδοποίηση των δεδομένων. Η δημοσίευση [51] παρουσιάζει αλγορίθμους που χρησιμοποιούνται στον εντοπισμό ανώμαλων τιμών (outliers). Ακόμη στην [52] παραθέτουν μία μελέτη για τις διαφορετικές προσεγγίσεις των μεθόδων που χρησιμοποιούνται στον εντοπισμό των ανώμαλων τιμών. Τέλος στην [53] γίνεται χρήση του αλγόριθμου Random Forest και των μηχανών διανυσμάτων υποστήριξης (Support Vector Machines) και προτείνεται ένας νέος αλγόριθμος για την εξόρυξη γνώσης από ιατρικά δεδομένα.

Όλες οι παραπάνω μελέτες και δημοσιεύσεις αποτελούν μόνο ένα μικρό δείγμα από το τεράστιο πλήθος μελετών που υπάρχουν για τον τομέα της διαχείρισης των δεδομένων τόσο στο επίπεδο του edge όσο και γενικότερα.

ΚΕΦΑΛΑΙΟ 5 Διαχείριση Ελλιπών Τιμών (Missing Values Imputation)

Η εμφάνισή ελλιπών τιμών αποτελεί όπως αναφέρθηκε και στην εισαγωγή ένα από τα προβλήματα που είναι δυνατόν να εμφανιστούν σε ένα σύνολο δεδομένων. Αυτό έχει ως αποτέλεσμα να εξάγουμε συμπεράσματα τα οποία είναι ανακριβή και λανθασμένα ή ακόμη να μην μπορούμε να εκτελέσουμε καμία ενέργεια επεξεργασίας αν οι τιμές που λείπουν καλύπτουν μεγάλο ποσοστό του συνόλου δεδομένων. Για τον λόγο αυτό καλούμαστε να αντιμετωπίσουμε το πρόβλημα αυτό με διάφορες τεχνικές. Μερικές από αυτές τις τεχνικές θα παρουσιαστούν στο παρόν κεφάλαιο. Οι μέθοδοι αυτοί αποτελούνται τόσο από μαθηματικά μοντέλα όσο και από τεχνικές που βασίζονται σε διάφορα είδη αλγορίθμων όπως αλγόριθμοι ομαδοποίησης.

5.1 Μέθοδοι διαγραφής ελλιπών Τιμών

Οι προσεγγίσεις αυτές αντιμετωπίζουν το πρόβλημα των ελλιπών τιμών διαγράφοντάς τις ελλείπουσες τιμές και επικεντρώνοντας στα υπόλοιπα δεδομένα προκειμένου να εξάγουν συμπεράσματα. Παρακάτω παρουσιάζονται μερικές μέθοδοι διαγραφής των δεδομένων [21].

5.1.α Μέθοδος διαγραφής List-wise

Η μέθοδος αυτή αφού εντοπίσει τις ελλείπουσες τιμές και τις διαστάσεις στις οποίες εμφανίζονται σε ένα σύνολο δεδομένων διαγράφει εκείνες τις καταγραφές εξολοκλήρου και όχι μόνο τη διάσταση στην οποία εμφανίστηκε η ελλείπουσα τιμή επικεντρώνοντας για τις ενέργειες που θέλει να εκτελέσει πάνω στα υπόλοιπα δεδομένα. Στο ακόλουθο παράδειγμα εφαρμόζεται η μέθοδος διαγραφής list-wise. Αν υποθέσουμε ότι διαθέτουμε το παρακάτω σύνολο δεδομένων από καταγραφές που αναφέρονται ως $(x) = \langle x_1, x_2, \dots, x_M \rangle$.

ID καταγραφής	Υγρασία	Θερμοκρασία	Ταχύτητα Ανέμου
1	53%	17°C	8 km/h
2	58%	16°C	6 km/h
3	40%	15°C	5 km/h
4	N/A	13°C	5 km/h
5	26%	16°C	4 km/h
6	33%	22°C	3 km/h
7	N/A	N/A	4 km/h
8	53%	17°C	6 km/h
9	46%	20°C	7 km/h
10	36%	N/A	9 km/h

Η μέθοδος List-wise εντοπίζει ελλείπουσες τιμές στη διάσταση που αναφέρεται στην Υγρασία για την τέταρτη και έβδομη καταγραφή και στη διάσταση που αναφέρεται στη Θερμοκρασία για την έβδομη και δέκατη καταγραφή. Μετά τον εντοπισμό τους η μέθοδος διαγράφη εξ' ολοκλήρου τις καταγραφές αυτές. Ύστερα από την εφαρμογή της μεθόδου το παραπάνω σύνολο γίνεται

ID καταγραφής	Υγρασία	Θερμοκρασία	Ταχύτητα Ανέμου
1	53%	17°C	8 km/h
2	58%	16°C	6 km/h
3	40%	15°C	5 km/h
5	26%	16°C	4 km/h
6	33%	22°C	3 km/h
8	53%	17°C	6 km/h
9	46%	20°C	7 km/h

5.1.β Μέθοδος διαγραφής Pairwise

Η μέθοδος διαγραφής pairwise διαγράφει μόνο τις διαστάσεις των καταγραφών στις οποίες εντοπίστηκε η ελλείπουσα τιμή. Έτσι η μέθοδος αυτή διατηρεί τη δυνατότητα επεξεργασίας των τιμών για τις διαστάσεις των καταγραφών που δεν παρουσίασαν ελλείπουσες τιμές.

ID καταγραφής	Υγρασία	Θερμοκρασία	Ταχύτητα Ανέμου
1	53%	17°C	8 km/h
2	58%	16°C	6 km/h
3	40%	15°C	5 km/h
4	N/A	13°C	5 km/h
5	26%	16°C	4 km/h
6	33%	22°C	3 km/h
7	N/A	N/A	4 km/h
8	53%	17°C	6 km/h
9	46%	20°C	7 km/h
10	36%	N/A	9 km/h

Η μέθοδος Pairwise εντοπίζει ελλείπουσες τιμές στη διάσταση που αναφέρεται στην Υγρασία για την τέταρτη και έβδομη καταγραφή και στην διάσταση που αναφέρεται στη Θερμοκρασία για την έβδομη και δέκατη καταγραφή. Η εφαρμογή της μεθόδου δίνει το εξής αποτέλεσμα αφού διαγραφούν οι διαστάσεις των καταγραφών που εντοπίστηκε η ελλείπουσα τιμή.

ID καταγραφής	Υγρασία	Θερμοκρασία	Ταχύτητα Ανέμου
1	53%	17°C	8 km/h
2	58%	16°C	6 km/h
3	40%	15°C	5 km/h
4		13°C	5 km/h
5	26%	16°C	4 km/h
6	33%	22°C	3 km/h
7			4 km/h
8	53%	17°C	6 km/h
9	46%	20°C	7 km/h
10	36%		9 km/h

5.1.γ Απόρριψη μεταβλητών (Dropping Variables)

Σε περίπτωση που υπάρχουν πολλές ελλείπουσες τιμές σε ένα σύνολο δεδομένων τότε μπορεί να γίνει διαγραφή ολόκληρης της διάστασης αυτής για το σύνολο δεδομένων που εξετάζουμε. Ωστόσο η επιλογή αυτής της μεθόδου θα πρέπει να είναι η τελευταία επιλογή καθώς η διαγραφή μιας ολόκληρης μεταβλητής για την οποία καταγράφουμε τιμές αλλάζει το σύνολο των δεδομένων.

ID καταγραφής	Υγρασία	Θερμοκρασία	Ταχύτητα Ανέμου
1	53%	17°C	8 km/h
2	N/A	16°C	6 km/h
3	40%	15°C	5 km/h
4	N/A	13°C	5 km/h
5	N/A	16°C	4 km/h
6	33%	22°C	3 km/h
7	N/A	20°C	4 km/h
8	53%	17°C	6 km/h
9	N/A	20°C	7 km/h
10	N/A	19°C	9 km/h

Η διάσταση που εμπεριέχει τις περισσότερες ελλείπουσες τιμές στο παράδειγμα αυτό είναι η μεταβλητή της Υγρασίας οπότε η μέθοδος διαγράφει από το υπο επεξεργασία σύνολο δεδομένων την στήλη της μεταβλητής αυτής. Έτσι η εφαρμογή της μεθόδου αυτής στο παραπάνω σύνολο δεδομένων δίνει το εξής νέο υποσύνολο.

ID καταγραφής	Θερμοκρασία	Ταχύτητα Ανέμου
1	17°C	8 km/h
2	16°C	6 km/h
3	15°C	5 km/h
4	13°C	5 km/h
5	16°C	4 km/h
6	22°C	3 km/h
7	20°C	4 km/h
8	17°C	6 km/h
9	20°C	7 km/h
10	19°C	9 km/h

5.2 Μέθοδοι αντιμετώπισης ελλιπών τιμών: Διατηρώντας όλα τα δεδομένα

Οι μέθοδοι που παρουσιάζονται στη συνέχεια αποτελούν ευρέως χρησιμοποιούμενες μεθόδους από τον τομέα της στατιστικής για την αντικατάσταση των ελλιπών τιμών. Ακόμη παρουσιάζεται και ο αλγόριθμος ομαδοποίησης k-Nearest Neighbors (k-NN).

5.2.α Μέθοδος Mean

Η μέθοδος αυτή αντικαθιστά τις ελλείπουσες τιμές με το μέσο όρο των τιμών της διάστασης στην οποία παρατηρούνται οι ελλείπουσες τιμές. Η μέθοδος αυτή χρησιμοποιείται κυρίως σε αριθμητικά δεδομένα.

ID καταγραφής	Υγρασία	Θερμοκρασία	Ταχύτητα Ανέμου
1	53%	17°C	8 km/h
2	58%	16°C	6 km/h
3	40%	15°C	5 km/h
4	33%	13°C	5 km/h
5	26%	16°C	4 km/h
6	33%	22°C	3 km/h
7	40%	N/A	4 km/h
8	53%	17°C	6 km/h
9	46%	20°C	7 km/h
10	36%	N/A	9 km/h

Η τιμή με την οποία αντικαθιστούμε τις ελλείπουσες τιμές στην στήλη με τις θερμοκρασίες υπολογίζεται από το άθροισμα των υπόλοιπων τιμών και την διαίρεση του με το πλήθος των μη ελλιπών τιμών στη στήλη των θερμοκρασιών. Ειδικότερα ο μέσος όρος είναι mean= (17+16+15+13+16+22+17+20)/8 =17. Άρα το παραπάνω σύνολο δεδομένων γίνεται:

ID καταγραφής	Υγρασία	Θερμοκρασία	Ταχύτητα Ανέμου
1	53%	17°C	8 km/h
2	58%	16°C	6 km/h
3	40%	15°C	5 km/h
4	33%	13°C	5 km/h
5	26%	16°C	4 km/h
6	33%	22°C	3 km/h
7	40%	17°C	4 km/h
8	53%	17°C	6 km/h
9	46%	20°C	7 km/h
10	36%	17°C	9 km/h

5.2.β Μέθοδος Median

Η μέθοδος αυτή χρησιμοποιεί τη μεσαία τιμή ανάμεσα σε ένα εύρος τιμών. Προκειμένου να βρούμε τη μεσαία τιμή σε ένα σύνολο δεδομένων ταξινομούμε τα δεδομένα αυτά σε αύξουσα σειρά. Η μεσαία τιμή για ένα πλήθος δεδομένων με άρτιο αριθμό είναι ο μέσος όρος των δύο μεσαίων τιμών που βρίσκονται στις θέσεις n/2 και (n+2)/2. Αντίθετα για περιττό αριθμό δεδομένων η μεσαία τιμή είναι μία και βρίσκεται στη θέση (n+1)/2. Στο επόμενο παράδειγμα αφού ταξινομήσουμε τις τιμές των δεδομένων σε αύξουσα σειρά η μεσαία τιμή είναι ο μέσος όρος των τιμών 16 και 18 αφού έχουμε άρτιο πλήθος δεδομένων. Άρα συμπληρώνουμε τις ελλείπουσες τιμές με την τιμή (16+16)/2=16.

ID καταγραφής	Υγρασία	Θερμοκρασία	Ταχύτητα Ανέμου
1	53%	15°C	8 km/h
2	58%	16°C	6 km/h
3	40%	15°C	5 km/h
4	33%	13°C	5 km/h
5	26%	16°C	4 km/h
6	33%	22°C	3 km/h
7	40%	N/A	4 km/h
8	53%	17°C	6 km/h
9	46%	20°C	7 km/h
10	36%	N/A	9 km/h

Το σύνολο των δεδομένων μετά την συμπλήρωση των ελλιπών τιμών είναι:

ID καταγραφής	Υγρασία	Θερμοκρασία	Ταχύτητα Ανέμου
1	53%	15°C	8 km/h
2	58%	16°C	6 km/h
3	40%	15°C	5 km/h
4	33%	13°C	5 km/h
5	26%	16°C	4 km/h
6	33%	22°C	3 km/h
7	40%	16°C	4 km/h
8	53%	17°C	6 km/h
9	46%	20°C	7 km/h
10	36%	16°C	9 km/h

5.2.γ Μέθοδος Mode

Η μέθοδος Mode συμπληρώνει τις ελλείπουσες τιμές που εντοπίζονται σε ένα σύνολο δεδομένων με την τιμή που εμφανίζεται συχνότερα στο σύνολο δεδομένων που εξετάζουμε για τη μεταβλητή στην οποία εμφανίζονται ελλείπουσες τιμές.

ID καταγραφής	Υγρασία	Θερμοκρασία	Ταχύτητα Ανέμου
1	53%	15°C	8 km/h
2	58%	16°C	6 km/h
3	40%	15°C	5 km/h
4	33%	15°C	5 km/h
5	26%	16°C	4 km/h
6	33%	22°C	3 km/h
7	40%	N/A	4 km/h
8	53%	17°C	6 km/h
9	46%	15°C	7 km/h
10	36%	N/A	9 km/h

Στην περίπτωση του παραδείγματος που παρουσιάζεται η πιο συχνά εμφανίσιμη τιμή είναι η 15°C οπότε αντικαθιστούμε τις ελλείπουσες τιμές με αυτήν την τιμή.

ID καταγραφής	Υγρασία	Θερμοκρασία	Ταχύτητα Ανέμου
1	53%	15°C	8 km/h
2	58%	16°C	6 km/h
3	40%	15°C	5 km/h
4	33%	15°C	5 km/h
5	26%	16°C	4 km/h
6	33%	22°C	3 km/h
7	40%	15°C	4 km/h
8	53%	17°C	6 km/h
9	46%	15°C	7 km/h
10	36%	15°C	9 km/h

5.2.δ Μέθοδος Next Observation Carried Backward (NOCB)

Η μέθοδος αυτή λειτουργεί αντίθετα από την προηγούμενη μέθοδο που περιγράψαμε μιας και αυτή η μέθοδος αντικαθιστά τις ελλείπουσες τιμές με την πρώτη τιμή που κατέγραψαν μετά από την ελλείπουσα τιμή. Ειδικότερα στο παράδειγμα που ακολουθεί φαίνεται η προσέγγιση που ακολουθεί η μέθοδος αυτή.

ID καταγραφής	Υγρασία	Θερμοκρασία	Ταχύτητα Ανέμου
1	53%	15°C	8 km/h
2	58%	16°C	6 km/h
3	40%	15°C	5 km/h
4	33%	N/A	5 km/h
5	26%	16°C	4 km/h
6	33%	22°C	3 km/h
7	40%	N/A	4 km/h
8	53%	17°C	6 km/h
9	46%	20°C	7 km/h
10	36%	20°C	9 km/h

Η εφαρμογή της μεθόδου συμπληρώνει την ελλείπουσα τιμή της τέταρτης καταγραφής με την τιμή 16°C και την ελλείπουσα τιμή της έβδομης καταγραφής με την τιμή 17°C όπως απεικονίζεται στον παρακάτω πίνακα.

ID καταγραφής	Υγρασία	Θερμοκρασία	Ταχύτητα Ανέμου
1	53%	15°C	8 km/h
2	58%	16°C	6 km/h
3	40%	15°C	5 km/h
4	33%	16°C	5 km/h
5	26%	16°C	4 km/h
6	33%	22°C	3 km/h
7	40%	17°C	4 km/h
8	53%	17°C	6 km/h
9	46%	20°C	7 km/h
10	36%	20°C	9 km/h

5.2.ε Μέθοδος Linear Interpolation

Η μέθοδος Linear Interpolation λειτουργεί αντικαθιστώντας τα δεδομένα όπως και η μέθοδος Mean με το μέσο όρο. Σε αντίθεση όμως με τη μέθοδο Mean η μέθοδος Linear Interpolation επικεντρώνεται μόνο σε δύο τιμές, την τιμή που καταγράφηκε πριν την ελλείπουσα τιμή και την τιμή που καταγράφηκε μετά την ελλείπουσα τιμή. Αναλυτικότερα η μέθοδος αυτή φαίνεται στο ακόλουθο αντιπροσωπευτικό παράδειγμα.

ID καταγραφής	Υγρασία	Θερμοκρασία	Ταχύτητα Ανέμου
1	53%	15°C	8 km/h
2	58%	16°C	6 km/h
3	40%	15°C	5 km/h
4	33%	N/A	5 km/h
5	26%	17°C	4 km/h
6	33%	22°C	3 km/h
7	40%	N/A	4 km/h
8	53%	18°C	6 km/h
9	46%	20°C	7 km/h
10	36%	20°C	9 km/h

Η αντικατάσταση των χαμένων τιμών αντικαθιστά με 16°C και 20°C τις χαμένες τιμές στην τέταρτη και έβδομη καταγραφή αντίστοιχα.

ID καταγραφής	Υγρασία	Θερμοκρασία	Ταχύτητα Ανέμου
1	53%	15°C	8 km/h
2	58%	16°C	6 km/h
3	40%	15°C	5 km/h
4	33%	16°C	5 km/h
5	26%	17°C	4 km/h
6	33%	22°C	3 km/h
7	40%	20°C	4 km/h
8	53%	18°C	6 km/h
9	46%	20°C	7 km/h
10	36%	20°C	9 km/h

5.2.στ Αλγόριθμος k-NN (k-nearest neighbors)

Εκτός από τις υπολογιστικές μεθόδους που αναφέρθηκαν παραπάνω στην αντιμετώπιση του προβλήματος των ελλιπών τιμών χρησιμοποιούνται αλγόριθμοι clustering, machine learning κ.α.. Ένας αλγόριθμος που ανήκει σε αυτήν την κατηγορία και χρησιμοποιείται συχνά στον υπολογισμό των ελλιπών τιμών είναι ο αλγόριθμος k-NN. Ο αλγόριθμος αυτός ομαδοποιεί τα δεδομένα του συνόλου χρησιμοποιώντας μετρικές όπως η απόσταση mahalanobis, ευκλείδεια απόσταση κ.α. και αντικαθιστά τις χαμένες τιμές με τη μέση τιμή των τιμών που κατέγραψαν οι k πλησιέστεροι γείτονες για τη διάσταση στην οποία εντοπίστηκε η ελλείπουσα τιμή. Ένα από τα πλεονεκτήματα του αλγορίθμου αυτού που προκαλούν την ευρεία χρήση του είναι ότι μπορεί να εφαρμοστεί σε δεδομένα που είναι συνεχή, διακριτά, σε δεδομένα που εκφράζουν κατηγορίες κ.α..[22]

5.3 Μελέτες πάνω σε αλγόριθμους υπολογισμού χαμένων τιμών από την ερευνητική κοινότητα

Στο υποκεφάλαιο αυτό γίνεται αναφορά σε δημοσιεύσεις που έχουν γίνει από την ερευνητική κοινότητα σχετικά με το φαινόμενο των ελλιπών τιμών. Στη συνέχεια θα αναφερθούν ορισμένες τέτοιες δημοσιεύσεις που έχουν να κάνουν με σύγκριση της απόδοσης αλγορίθμων, προτάσεις νέων αλγορίθμων κ.α..

Πιο συγκεκριμένα στη δημοσίευση [41] προτείνεται μία μέθοδος υπολογισμού ελλιπών τιμών που είναι ικανή να χειριστεί διάφορους τύπους δεδομένων. Η μέθοδος που προτείνεται είναι βασισμένη στην μέθοδο Random Forest που αποτελεί μια μέθοδο υπολογισμού ελλιπών τιμών. Στην επόμενη μελέτη [40] παρουσιάζεται μία σύγκριση ανάμεσα σε τέσσερεις από τους πιο δημοφιλείς αλγορίθμους υπολογισμού ελλιπών τιμών. Ειδικότερα συγκρίνονται οι αλγόριθμοι Regularized Expectation-Maximization (EM), Multiple Imputation (MI), k-NN Imputation (kNNI) και Mean Imputation (MI). Εκτός από στατιστικές μεθόδους για τον υπολογισμό των ελλιπών τιμών χρησιμοποιούνται και αλγόριθμοι από άλλα επιστημονικά πεδία όπως αυτό του Machine Learning. Στη δημοσίευση [39]

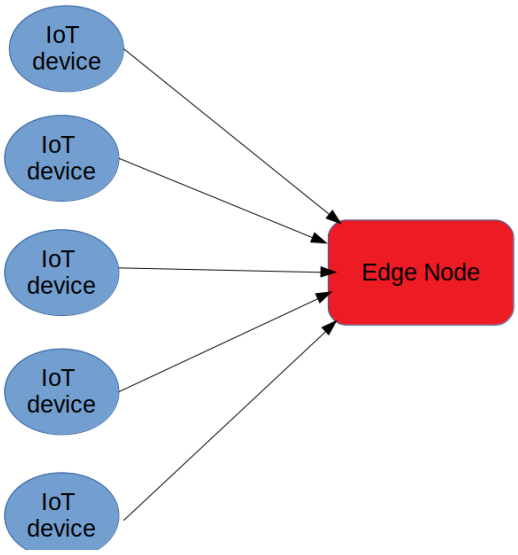
γίνεται μία σύγκριση ανάμεσα σε τρείς αλγορίθμους ταξινόμησης από την περιοχή του Machine Learning. Οι αλγόριθμοι που συγκρίνονται είναι οι k-NN, δένδρα απόφασης (Decision Tree) και Bayesian Networks. Στη μελέτη [42] γίνεται περιγραφή και σύγκριση της απόδοσης μεταξύ μηχανισμών υπολογισμού ελλιπών τιμών προκειμένου να διαπιστωθεί ποιος είναι ο ποιο κατάλληλος αλγόριθμος για την αντιμετώπιση των ελλιπών τιμών σε οικονομικά στατιστικά στοιχεία. Τέλος στη δημοσίευση [38] παρουσιάζεται σύγκριση ανάμεσα σε τεχνικές υπολογισμού ελλιπών τιμών και ειδικότερα των Mean\Mode, K-NN, Hot-Deck, Expectation Maximization and C5.0. Ένας ακόμη τομέας που μπορεί να προσφέρει αρκετά στην επίλυση του προβλήματος των ελλιπών τιμών είναι οι τεχνητή νοημοσύνη. Στην [54] προτείνεται ένας αλγόριθμος υπολογισμού των ελλιπών τιμών ο οποίος χρησιμοποιεί γενετικό αλγόριθμο. Τα αποτελέσματα του προτεινόμενου μοντέλου στο [54] συγκρίνονται με διάφορες μεθόδους μερικές εκ των οποίων περιεγράφηκαν παραπάνω όπως η μέθοδος mean και ο k-NN. Στην [57] προτείνεται ακόμη ένας γενετικός αλγόριθμος για την αντιμετώπιση του φαινομένου των ελλιπών τιμών. Στη συνέχεια η [55] προσφέρει γενικές οδηγίες για την επιλογή του κατάλληλης οικογένειας αλγορίθμων μηχανικής μάθησης (Machine Learning) ανάλογα με τα χαρακτηριστικά του συνόλου των δεδομένων στα οποία εντοπίζονται οι ελλείπουσες τιμές. Αντίθετα στην [56] προτείνεται μια νέα προσέγγιση της μεθόδου Mean που περιεγράφηκε προηγουμένως. Τέλος στην [58] προτείνεται ένας νέος αλγόριθμος υπολογισμού των ελλιπών τιμών βασισμένος στον αλγόριθμο K-Means.

ΚΕΦΑΛΑΙΟ 6 Προτεινόμενος Αλγόριθμος

Στο κεφάλαιο αυτό παρουσιάζεται το προτεινόμενο μοντέλο για τον υπολογισμό και την αντικατάσταση των ελλιπών τιμών. Ειδικότερα θα δοθεί περιγραφή για το δίκτυο και τον τρόπο λειτουργίας του πάνω στο οποίο εφαρμόζεται ο προτεινόμενος αλγόριθμος, τις μεθόδους συσχέτισης που χρησιμοποιούνται, το τελικό σχήμα συσχέτισης καθώς και το σχήμα αντικατάστασης που προτείνεται.

6.1 Δίκτυο και τρόπος λειτουργίας του

Η κατηγορία των δικτύων πάνω στα οποία εφαρμόζεται ο προτεινόμενος αλγόριθμος αποτελούνται από ένα πλήθος IoT συσκευών και από ένα σύνολο edge κόμβων. Οι IoT συσκευές συλλέγουν τιμές από το περιβάλλον τους για διάφορες παραμέτρους ή εξάγουν γνώση μετά από την εκτέλεση ενεργειών ή επεξεργασίας πάνω σε διάφορα δεδομένα. Στην συνέχεια οι IoT συσκευές στέλνουν τα δεδομένα τους στους edge κόμβους προκειμένου να υποστούν περαιτέρω επεξεργασία. Αυτό έχει ως αποτέλεσμα οι edge κόμβοι να λειτουργούν ως συλλέκτες πληροφορίας αποθηκεύοντας πληροφορίες οι οποίες παρέχονται από τις IoT συσκευές. Στην παρούσα εργασία το δίκτυο αποτελείται από έναν αριθμό N IoT συσκευών και έναν edge κόμβο.



Εικόνα 5:Παράδειγμα δομής δικτύου

Τα δεδομένα που καταφτάνουν στον edge κόμβο αναφέρονται με τη μορφή πινάκων πολλαπλών μεταβλητών για παράδειγμα $(x) = \langle x_1, x_2, \dots, x_M \rangle$, όπου M είναι ο αριθμός των μεταβλητών που καταγράφουν οι IoT συσκευές. Ο edge κόμβος λαμβάνοντας τα δεδομένα με τη μορφή που αναφέρθηκε παραπάνω τα αποθηκεύει με τέτοιον τρόπο ώστε να είναι διαθέσιμα για την επεξεργασία που θα χρειαστεί να υποστούν. Ειδικότερα κάθε πίνακας που καταφτάνει από μια από τις IoT συσκευές του δικτύου αποθηκεύεται με τη χρήση δύο δεικτών του δείκτη j και t. Ο δείκτης j χρησιμοποιείται για την αναφορά στην IoT συσκευή που στέλνει τα δεδομένα και ο δείκτης t χρησιμοποιείται για την αναφορά στη χρονική στιγμή που στέλνεται από την j-οστή IoT συσκευή ο πίνακας με τις καταγεγραμμένες τιμές των μεταβλητών.

Έτσι ο edge κόμβος αποθηκεύει τα δεδομένα που δέχεται από κάθε IoT συσκευή με τη μορφή $(x)^j = \langle x_1^j[t], x_2^j[t], \dots, x_M^j[t] \rangle$.

Ο edge κόμβος κάθε χρονική στιγμή αφού λάβει τα δεδομένα από τις IoT συσκευές με μορφή πινάκων όπως περιεγράφηκε παραπάνω ελέγχει για την ύπαρξη σε αυτούς ελλιπών τιμών. Οι ελλείπουσες τιμές μπορεί να βρεθούν σε κάποια συγκεκριμένη διάσταση του πίνακα ή σε όλο τον πίνακα. Σε περίπτωση που ανιχνευτεί κάποια ελλείπουσα τιμή ο edge κόμβος εφαρμόζει τον αλγόριθμο που προτείνεται προκειμένου να υπολογίσει την τιμή που θα αντικαταστήσει την ελλείπουσα τιμή ώστε να διασφαλιστεί η ποιότητα των δεδομένων που συλλέχθηκαν.

Στον προτεινόμενο αλγόριθμο της παρούσας εργασίας εκτός από τη συλλογή των δεδομένων, την αποθήκευσή τους και την επεξεργασία του συνόλου των τιμών που δημιουργήθηκε από τις IoT συσκευές τη χρονική στιγμή που εντοπίστηκε ένα πλήθος ελλιπών τιμών ο edge κόμβος είναι επιφορτισμένος και με την επεξεργασία των W πιο πρόσφατων τιμών που έλαβε από τις IoT συσκευές. Για τον λόγο αυτό ο edge κόμβος κρατά μόνο τα W πιο πρόσφατα δεδομένα για κάθε IoT συσκευή δημιουργώντας έτσι για κάθε μια ένα δισδιάστατο πίνακα που ανανεώνεται συνεχώς με το πέρασμα του χρόνου. Ο δισδιάστατος αυτός πίνακας συνεισφέρει σημαντικά στον υπολογισμό της τιμής η οποία θα αντικαταστήσει την ελλείπουσα τιμή και το προτεινόμενο μοντέλο εστιάζει στις W πιο πρόσφατες τιμές που καταγράφθηκαν. Στον Πίνακα 1 παρουσιάζεται το πρότυπο του δισδιάστατου πίνακα στον οποίο αποθηκεύονται τα δεδομένα της j-οστής IoT συσκευής για τις W πιο πρόσφατες τιμές που κατέγραψαν.

Πίνακας 1:Πίνακας για την αποθήκευση των W πιο πρόσφατων καταγεγραμμένων τιμών για μία IoT συσκευή.

	1η διάσταση	2η διάσταση	…	M-οστή διάσταση
t=1	$x_1^j[1]$	$x_2^j[1]$	…	$x_M^j[1]$
t=2	$x_1^j[2]$	$x_2^j[2]$	…	$x_M^j[2]$
…	…	…	…	
t=W	$x_1^j[W]$	$x_2^j[W]$	…	$x_M^j[W]$

Ο προτεινόμενος αλγόριθμος που περιγράφεται προκειμένου να υπολογίσει τις τιμές που θα αντικαταστήσουν τις ελλείπουσες τιμές που εντοπίστηκαν βασίζεται στα δεδομένα τα οποία έστειλαν οι IoT συσκευές με τη μεγαλύτερη συσχέτιση με την IoT συσκευή, τις οποίας τα δεδομένα που έστειλε στον edge κόμβο περιέχουν ελλείπουσες τιμές,. Πιο συγκεκριμένα για την επιλογή των κόμβων στους οποίους θα βασιστεί ο edge κόμβος για τον υπολογισμό των τιμών ο προτεινόμενος αλγόριθμος χρησιμοποιεί ένα συνολικό σχήμα για τον υπολογισμό της συσχέτισης μεταξύ των IoT συσκευών συνδυάζοντας τόσο τα δεδομένα που έστειλαν οι IoT συσκευές τη χρονική στιγμή που εντοπίστηκε μία ελλείπουσα τιμή όσο και τα δεδομένα που έστειλαν για τις προηγούμενες χρονικές στιγμές που βρίσκονται μέσα στο πίνακα με τις W πιο πρόσφατες χρονικές στιγμές.

Ο προτεινόμενος αλγόριθμος μπορούμε να θεωρήσουμε ότι χωρίζεται σε τρία μέρη.

- Το πρώτο μέρος αποτελείται από τον υπολογισμό της συσχέτισης μεταξύ των τιμών που καταγράφουν οι IoT συσκευές τόσο για τη χρονική στιγμή που εντοπίστηκε η ελλείπουσα τιμή όσο και για τη συσχέτιση που εμφανίζουν μεταξύ τους οι W πιο πρόσφατες χρονικές στιγμές.
- Το δεύτερο μέρος αποτελείται από τον υπολογισμό της τελικής συσχέτισης μεταξύ των IoT συσκευών συνδυάζοντας σε ένα συνολικό σχήμα τα αποτελέσματα του υπολογισμού των συσχετίσεων του πρώτου μέρους και την επιλογή με βάση την τελική συσχέτιση των IoT συσκευών που θα συμμετέχουν στον υπολογισμό της τιμής που θα αντικαταστήσει την τιμή που λείπει.
- Το τρίτο και τελευταίο κομμάτι είναι εκείνο στο οποίο χρησιμοποιούμε μία ευρετική μέθοδο αντικατάστασης βασισμένη στις συσχετίσεις.

Η παραπάνω διαδικασία μας δείχνει την πρόθεση της προτεινόμενης μεθόδου να βασιστεί μόνο σε δεδομένα των top-k κόμβων που παρουσιάζουν ομοιότητα στις τιμές που στέλνουν στον edge κόμβο με την προοπτική να επηρεάσουν θετικά τον υπολογισμό της τιμής που θα αντικαταστήσει την ελλείπουσα τιμή. Τα παραπάνω στάδια περιγράφονται αναλυτικότερα παρακάτω.

6.2 Περιγραφή σταδίων προτεινόμενου μηχανισμού

6.2.α Πρώτο στάδιο – Υπολογισμός των συσχετίσεων μεταξύ των IoT συσκευών.

Στο πρώτο στάδιο όπως προαναφέρθηκε υπολογίζονται οι συσχετίσεις μεταξύ των IoT συσκευών τόσο για τη στιγμή που ανιχνεύτηκε η ελλείπουσα τιμή όσο και για τις W πιο πρόσφατες τιμές που κατέγραψαν οι IoT συσκευές. Για τον υπολογισμό των συσχετίσεων αυτών χρησιμοποιούνται δύο μέθοδοι υπολογισμού συσχέτισης. Η μία μέθοδος είναι η απόσταση Mahalanobis η οποία χρησιμοποιείται για τη συσχέτιση που παρουσιάζουν οι W πιο πρόσφατες τιμές που καταγράφηκαν από τις IoT συσκευές. Από την άλλη πλευρά για την εύρεση της συσχέτισης μεταξύ των IoT συσκευών τη χρονική στιγμή που εντοπίστηκε η ελλείπουσα τιμή χρησιμοποιούμε τη μέθοδο Cosine Similarity. Στη συνέχεια παρουσιάζονται οι δύο μέθοδοι για τον υπολογισμό της συσχέτισης έτσι όπως χρησιμοποιούνται στον προτεινόμενο αλγόριθμο.

<u>1. Απόσταση Mahalanobis (Mahalanobis Distance)</u>

Η απόσταση Mahalanobis μετράει τη συσχέτιση ως την απόσταση μεταξύ ενός σημείου P , που μπορεί να αποτελείται από την καταγραφή μίας τιμής ή ενός συνόλου τιμών, και ενός πλήθους D από καταγεγραμμένα σύνολα δεδομένων. Στην περίπτωση αυτή ο τύπος που δίνει την απόσταση mahalanobis είναι ο ακόλουθος:

$$MD(\vec{x}) = \sqrt{(\vec{x} - \vec{\mu})^T S^{-1} (\vec{x} - \vec{\mu})} \qquad (1)$$

Στον παραπάνω τύπο το $\vec{x}$ αποτελείται από τον πίνακα των τιμών των μεταβλητών που καταγράφηκαν σε ένα σημείο P και το $\vec{\mu}$ είναι ο πίνακας που περιέχει το μέσο όρο των τιμών των μεταβλητών που έχουν καταγραφεί στο σύνολο δεδομένων D. Επίσης η απόσταση Mahalanobis εφαρμόζεται και στην ανίχνευση της συσχέτισης μεταξύ δύο πολυδιάστατων πινάκων πολλαπλών μεταβλητών που ανήκουν στο ίδιο σύνολο δεδομένων.

Στον αλγόριθμο που παρουσιάζουμε χρησιμοποιούμε την απόσταση Mahalanobis για την εύρεση της συσχέτισης μεταξύ των δισδιάστατων πινάκων που σχηματίζουν οι τιμές που καταγράφηκαν από τις IoT συσκευές για τις W πιο πρόσφατες χρονικές στιγμές. Έτσι αν θεωρήσουμε ότι τα διανύσματα $\vec{x}$ και $\vec{y}$ είναι δύο δισδιάστατοι πίνακες που περιέχουν τα δεδομένα που κατέγραψαν δύο διαφορετικές IoT συσκευές για τις W πιο πρόσφατες χρονικές στιγμές η απόσταση Mahalanobis δίνεται από την παρακάτω εξίσωση.

$$MD(\vec{x}, \vec{y}) = \sqrt{(\vec{x} - \vec{y})^T S^{-1} (\vec{x} - \vec{y})} \qquad (2)$$

Τόσο στον πρώτο τύπο όσο και στον δεύτερο το S εκφράζει τον πίνακα covariance. Όπως αναφέρθηκε και προηγουμένως η απόσταση Mahalanobis εφαρμόζεται πάνω σε συνεχώς ανανεώσιμους πίνακες καθώς όσο ο χρόνος αυξάνει αλλάζουν και οι W πιο πρόσφατες καταγραφές.

2.Cosine Similarity

Η μέθοδος συσχέτισης Cosine Similarity μετράει την ομοιότητα μεταξύ δύο μη μηδενικών πινάκων λαμβάνοντας υπόψιν το συνημίτονο της μεταξύ τους γωνίας. Η γωνία μεταξύ των δύο διανυσμάτων καθορίζει και την τιμή του συνημίτονου καθώς όσο αυτή αυξάνεται η τιμή του συνημίτονου μειώνεται, επομένως μειώνεται και η ομοιότητα που παρουσιάζουν οι δύο αυτοί πίνακες μεταξύ τους. Η παρακάτω μαθηματική συνάρτηση δίνει την τιμή για το Cosine Similarity.

$$CS\big((x)^i[t], (x)^j[t]\big) = \frac{(x)^i[t] \cdot (x)^j[t]}{\|(x)^i[t]\| \cdot \|(x)^j[t]\|} = \frac{\sum_{l=1}^{M} (x)_l^i[t] \cdot (x)_l^j[t]}{\sqrt{\sum_{l=1}^{M} ((x)_l^i[t])^2} \cdot \sqrt{\sum_{l=1}^{M} ((x)_l^j[t])^2}} \quad (3)$$

Στην εξίσωση που παρατέθηκε παραπάνω οι δείκτες i, j συμβολίζουν τη μεταβλητή των πινάκων για τις αντίστοιχες IoT συσκευές για τις οποίες εξετάζουμε την ομοιότητα μεταξύ των τιμών που κατέγραψαν τη χρονική στιγμή που εντοπίστηκε η ελλείπουσα τιμή. Για τον λόγο αυτό προκειμένου να καταστεί δυνατή η εφαρμογή της μεθόδου Cosine Similarity τη χρονική στιγμή που υπάρχει τουλάχιστον μία ελλείπουσα τιμή εφαρμόζουμε την ακόλουθη στρατηγική. Ειδικότερα αγνοούμε από τους υπολογισμούς της μεθόδου τις διαστάσεις εκείνες, και από τους δύο πίνακες, που εντοπίζονται οι ελλείπουσες τιμές βασιζόμενοι έτσι στις υπόλοιπες τιμές που περιέχονται στις υπόλοιπες

διαστάσεις των πινάκων. Η εφαρμογή της μεθόδου γίνεται σε κάθε ζεύγος IoT συσκευών με τον τρόπο που περιεγράφηκε προηγουμένως.

6.2.β Δεύτερο στάδιο – Περιγραφή του συνολικού μοντέλου για την τελική συσχέτιση μεταξύ των IoT συσκευών.

Για την τελική συσχέτιση μεταξύ των IoT συσκευών συνδυάζουμε τις δύο μεθόδους που περιεγράφηκαν στην προηγούμενη παράγραφο σε ένα συνολικό σχήμα. Ειδικότερα στο συνολικό μοντέλο που υιοθετούμε για τον υπολογισμό της τελικής συσχέτισης χρησιμοποιούμε την απόσταση Mahalanobis προκειμένου να «βαθμονομήσουμε» τα αποτελέσματα της μεθόδου Cosine Similarity. Έτσι δημιουργούμε για κάθε αποτέλεσμα της μεθόδου Cosine Similarity, που προκύπτει από κάθε ζεύγος IoT συσκευών, ένα βάρος w ως εξής:

$$w = \frac{1}{MD} \qquad (4)$$

Με αυτόν τον τρόπο δημιουργούμε μία σύνδεση της συσχέτισης, ανάμεσα σε δύο IoT συσκευές, που παρουσιάζουν οι τιμές των δεδομένων για τις W πιο πρόσφατες καταγεγραμμένες τιμές και την συσχέτιση/ομοιότητα που έχουν οι τιμές που καταγράφηκαν την χρονική στιγμή του εντοπισμού των ελλιπών τιμών. Η εξίσωση που ακολουθεί παρουσιάζει τον τρόπο διασύνδεσης και υπολογισμού των μεθόδων Mahalanobis distance και Cosine Similarity στο συνολικό μοντέλο συσχέτισης.

$$F_C = w \cdot CS\big((x)^i[t], (x)^j[t]\big), \forall i, j, i \neq j \qquad (5)$$

Όσο μεγαλύτερη είναι η τιμή της απόστασης Mahalanobis τόσο μικρότερη είναι η τιμή του βάρους w. Αυτό σημαίνει ότι σε περίπτωση που δύο IoT συσκευές παρουσιάζουν μικρή συσχέτιση στις W πιο πρόσφατες τιμές, δηλαδή η απόσταση Mahalanobis έχει μεγάλη τιμή, τότε η τελική συσχέτιση μειώνεται ανεξαρτήτως της τιμής που επιστρέφει η μέθοδος Cosine Similarity. Αντιθέτως σε περίπτωση που η συσχέτιση μεταξύ των W πιο πρόσφατων τιμών είναι μεγάλη, δηλαδή η τιμή της Mahalanobis distance είναι χαμηλή, τότε ενισχύουμε το αποτέλεσμα της μεθόδου Cosine Similarity προκειμένου να επιβεβαιώσουμε τη μεγάλη συσχέτιση μεταξύ των δύο IoT συσκευών του ζεύγους που αφορά η τελική συσχέτιση.

Στη συνέχεια επιλέγουμε τα top-k αποτελέσματα που αντιστοιχούνε στις IoT συσκευές, με τη μεγαλύτερη τελική συσχέτιση (F_C) , στις οποίες θα βασιστούμε για τον υπολογισμό των τιμών που θα αντικαταστήσουν τις ελλείπουσες τιμές. Ο προτεινόμενος αλγόριθμος δίνει ιδιαίτερη σημασία στα αποτελέσματα της μεθόδου Cosine Similarity καθώς θα διατελέσει σημαντικό ρόλο στην διαδικασία της αντικατάστασης των ελλιπών τιμών που θα αναλυθεί στην συνέχεια.

6.2.γ Τρίτο Στάδιο – Περιγραφή της μεθόδου αντικατάστασης.

Για τον υπολογισμό των τιμών που θα αντικαταστήσουν τις ελλείπουσες τιμές απαιτείται η διεκπεραίωση των υπολογισμών που περιεγράφηκαν στις προηγούμενες παραγράφους. Έτσι για κάθε μια από τις ελλείπουσες τιμές που εντοπίστηκαν το προτεινόμενο σχήμα βασίζεται στις τιμές που κατέγραψαν οι top-k, σύμφωνα με την τελική συσχέτιση, IoT συσκευές στην διάσταση που εντοπίστηκε η ελλείπουσα τιμή.

Για να γίνει πιο κατανοητή η διαδικασία της αντικατάστασης μίας χαμένης τιμής ας θεωρήσουμε το ακόλουθο σενάριο. Υποθέτοντας ότι με $MN_i, i \in \{1,2,\dots,k\}$ συμβολίζονται τα top-k αποτελέσματα της μεθόδου Cosine Similarity από τις top-k συσχετισμένες IoT συσκευές, οι οποίες επιλέχθηκαν με βάση την τελική συσχέτιση (F_C), ταξινομημένα κατά αύξουσα σειρά και d η διάσταση που παρατηρήθηκε η ελλείπουσα τιμή. Έτσι η τιμή η οποία θα πάρει την θέση της ελλείπουσας τιμής προκύπτει από τον ακόλουθο τύπο.

$$PD = \frac{\sum_{i=1}^{k} MN_i \cdot x_d}{\sum_{l=1}^{k} CS_l} \quad (6)$$

Όπου CS_l είναι η ομοιότητα που προκύπτει από την εφαρμογή της μεθόδου Cosine Similarity ανάμεσα στην IoT συσκευή στην οποία εντοπίστηκε η χαμένη τιμή και σε κάθε μία από τις k IoT συσκευές που ανήκουν στην λίστα με τις IoT συσκευές με την μεγαλύτερη τελική συσχέτιση. Ειδικότερα ο προτεινόμενος τρόπος υπολογισμού της τιμής που θα αντικαταστήσει την ελλείπουσα τιμή λειτουργεί με έναν μοντέλο «επιβράβευσης» καθώς η IoT συσκευή που είναι ανάμεσα στις top-k IoT συσκευές και βρίσκεται στην πρώτη θέση της λίστας $MN_i, i \in \{1,2,\dots,k\}$ θα συνεισφέρει περισσότερο στην τελική τιμή σε αντίθεση με την IoT συσκευή που βρίσκεται στην θέση k.

ΚΕΦΑΛΑΙΟ 7 Πειραματική Αποτίμηση

Στο παρόν κεφάλαιο παρουσιάζονται οι μετρικές και οι συνδυασμοί των παραμέτρων που εφαρμόσαμε για να διαπιστώσουμε την απόδοση και την αποτελεσματικότητα του προτεινόμενου αλγορίθμου. Ακόμη γίνεται αναφορά στα σύνολο δεδομένων που βασιστήκαμε προκειμένου να δοκιμάσουμε τον προτεινόμενο αλγόριθμο. Τέλος στα πειραματικά διαγράμματα παρουσιάζεται και η σύγκριση της απόδοσης του προτεινόμενου αλγορίθμου με έναν μηχανισμό που τον ονομάσαμε Averaging Model. Ο μηχανισμός αυτός αντικαθιστά την ελλείπουσα τιμή χρησιμοποιώντας το μέσο όρο των τιμών που κατέγραψαν οι IoT συσκευές που ανήκουν στη δομή $MN_i, i \in \{1,2,\dots,k\}$.

7.1 Σύνολα δεδομένων που βασιστήκαμε

Το σύνολο δεδομένων που χρησιμοποιήσαμε βασίζονται στα ακόλουθα δύο σύνολα δεδομένων. Το πρώτο σύνολο στο οποίο βασιστήκαμε είναι το GNFUV Unmanned Surface Vehicles Sensor Data Set [44] και το δεύτερο σύνολο είναι το Intel Berkeley Research Lab dataset[43]. Το σύνολο δεδομένων GNFUV εμπεριέχει τιμές καταγεγραμμένες από κινητούς αισθητήρες μη επανδρωμένων οχημάτων (Unmanned Surface Vehicles (USV)) οι οποίοι καταγράφουν τις τιμές της υγρασίας και της θερμοκρασίας. Τα οχήματα αυτά κινούνται με βάση ένα GPS που ακολουθεί μια προκαθορισμένη διαδρομή. Το σύνολο δεδομένων που παρέχεται από την Intel εμπεριέχει εκατομμύρια μετρήσεις θερμοκρασίας υγρασίας και φωτός που παρέχονται από 54 αισθητήρες σε ένα εργαστήριο. Από αυτό το σύνολο δεδομένων λαμβάνουμε 15,000 μετρήσεις καθώς κάθε ένας από τους δεκαπέντε παράγει 1,000 μετρήσεις.

7.2. Μετρικές υπολογισμού σφάλματος

Προκειμένου να διαπιστώσουμε την αποτελεσματικότητα του αλγορίθμου εστιάζουμε στο προβλεπόμενο σφάλμα. Το προβλεπόμενο σφάλμα υπολογίζεται χρησιμοποιώντας την τιμή που υπολόγισε ο αλγόριθμος που προτείναμε και περιγράψαμε στα προηγούμενα κεφάλαια και στην πραγματική τιμή που περιέχεται στα σύνολα δεδομένων που εξετάζουμε. Οι μετρικές τις χρησιμοποιούμε προκειμένου να ελέγξουμε την απόδοση του προτεινόμενου αλγορίθμου είναι δύο ευρέως χρησιμοποιούμενες μετρικές υπολογισμού του σφάλματος. Ειδικότερα οι μετρικές αυτές είναι το Μέσο Απόλυτο Σφάλμα (Mean Absolute Error (MAE)) και το Μέσο Τετραγωνικό Σφάλμα (Root Mean Square Error (RMSE)). Οι τύποι των παραπάνω μετρικών παρουσιάζονται παρακάτω.

$$MAE = \frac{\sum_{l=1}^{n} |PD_l - a_l|}{n}$$

$$RMSE = \frac{\sqrt{\sum_{l=1}^{n} (PD_l - a_l)^2}}{n}$$

Στους παραπάνω τύπους των μετρικών το n είναι ο αριθμός των ελλιπών τιμών που εντοπίστηκαν στο σύνολο δεδομένων που εξετάζει ο προτεινόμενος αλγόριθμος και a_l είναι η πραγματική τιμή που βρίσκεται στο αντίστοιχο σύνολο δεδομένων.

7.3. Παράμετροι Πειραμάτων και επιλογή των τιμών τους.

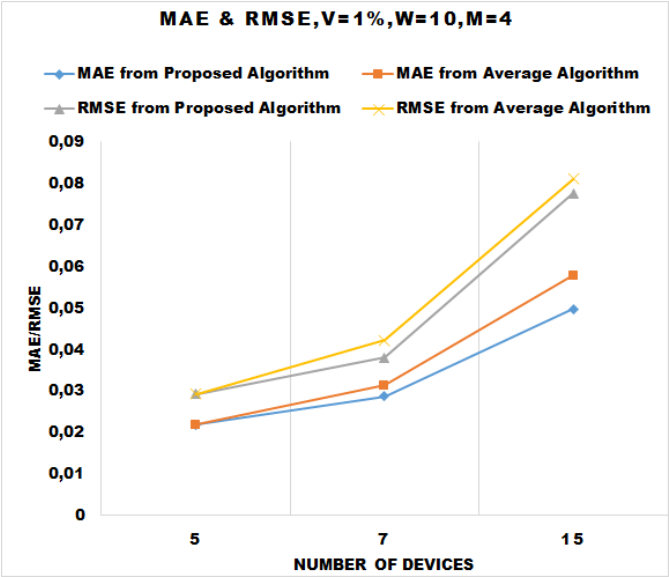
Η διεξαγωγή των πειραμάτων απαιτεί των καθορισμό πέντε παραμέτρων. Οι παράμετροι αυτοί είναι το ποσοστό **V** των ελλιπών τιμών που θα δημιουργήσουμε στο σύνολο δεδομένων που εξετάζουμε, ο αριθμός των **k** κορυφαίων κόμβων με τη μεγαλύτερη τελική συσχέτιση, τον αριθμό των IoT συσκευών (**NUMBER OF DEVICES**) από τις οποίες θα αποτελείται το δίκτυο στο οποίο εργαζόμαστε, το πλήθος **M** των μεταβλητών για τις οποίες οι IoT συσκευές θα καταγράφουν τις τιμές τους και το πλήθος **W** των πιο πρόσφατων χρονικών στιγμών που θα λαμβάνουμε υπόψιν στον υπολογισμούς που διεξάγουμε για τον υπολογισμό των συσχετίσεων. Οι τιμές που επιλέγουμε για τις μεταβλητές αυτές παρουσιάζονται στην συνέχεια.

- $V \in \{1,5,10\}$
- $k = 4$
- $NUMBER\ OF\ DEVICES\ \in \{5,7,15\}$
- $M \in \{4,9\}$
- $W \in \{10,30\}$

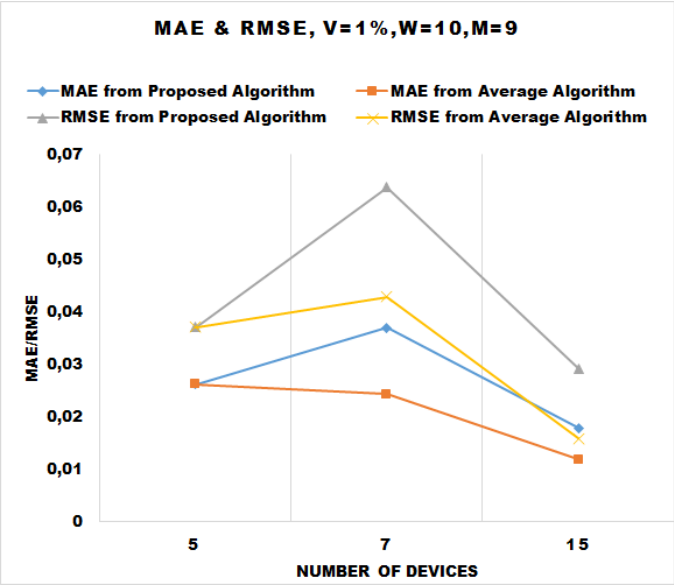
7.4. Αποτελέσματα και αξιολόγηση Πειραμάτων.

Στα παρακάτω διαγράμματα θα παρουσιαστούν τα αποτελέσματα των πειραμάτων που διεξήγαμε συνοδευόμενη από μια μικρή αξιολόγηση τους. Τα πειράματα χωρίζονται σε δύο υποπαραγράφους στα πειράματα όπου η μεταβλητή **W** είναι ίση με 10 χρονικές στιγμές και σε αυτά όπου η μεταβλητή **W** είναι ίση με 30 χρονικές στιγμές.

7.4.α Πειράματα με W=10.

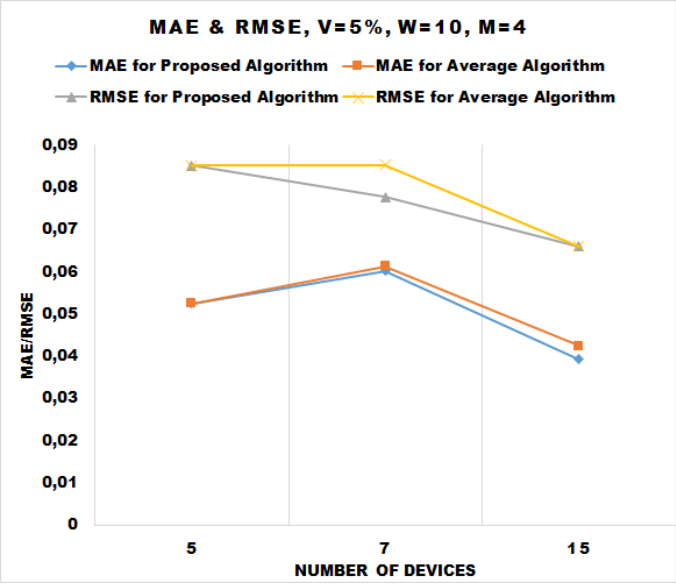


Εικόνα 6: MAE και RMSE για V=1% και M=4

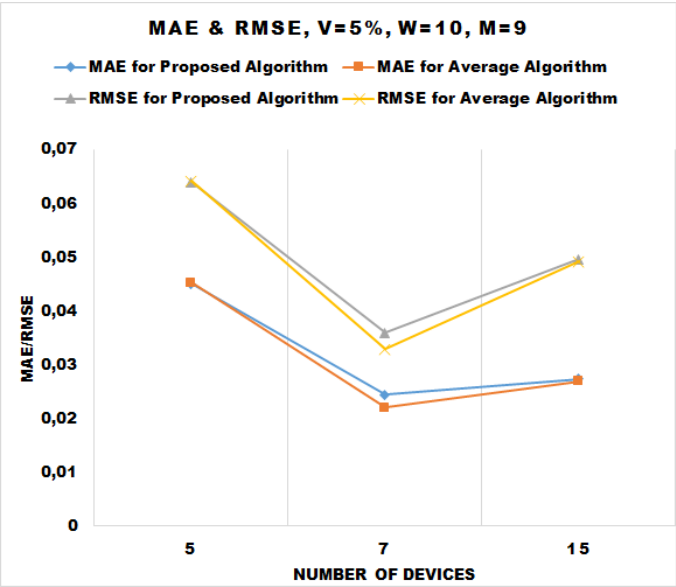


Εικόνα 7:MAE και RMSE για V=1% και M=9

Στις Εικόνες 6 και 7 παρουσιάζονται τα αποτελέσματα του προτεινόμενου αλγορίθμου σε σύγκριση με τον AM μηχανισμό για $M = 4$ και $M = 9$ αντίστοιχα και με ποσοστό ελλιπών τιμών $V = 1\%$. Παρατηρώντας τα αποτελέσματα βλέπουμε ότι ο αριθμός των IoT συσκευών που συμμετέχουν στο δίκτυο επηρεάζει την απόδοση του αλγορίθμου. Ειδικότερα όταν το $M = 4$ τόσο η μετρική του MAE όσο και η μετρική του $RMSE$ σφάλματος μεγαλώνει όσο αυξάνεται ο αριθμός των IoT συσκευών, παρουσιάζοντας τη μεγαλύτερη τιμή σφάλματος για 15 IoT συσκευές. Το αντίθετο συμβαίνει για $M = 9$. Έτσι καταλήγουμε στο συμπέρασμα ότι ένας μεγάλος αριθμός μεταβλητών δεν επηρεάζει αρνητικά την απόδοση του αλγορίθμου που προτείνεται και μπορεί να αντικαταστήσει αποτελεσματικά τις ελλείπουσες τιμές. Σε σύγκριση με τον AM μηχανισμό, ο αλγόριθμος που προτείνεται υπερτερεί για μικρό αριθμό μεταβλητών.

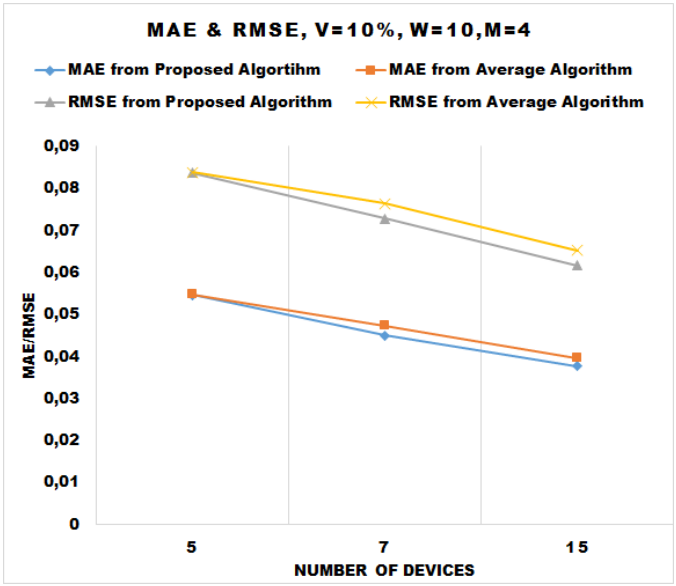


Εικόνα 8: MAE και RMSE για V=5% και M=4

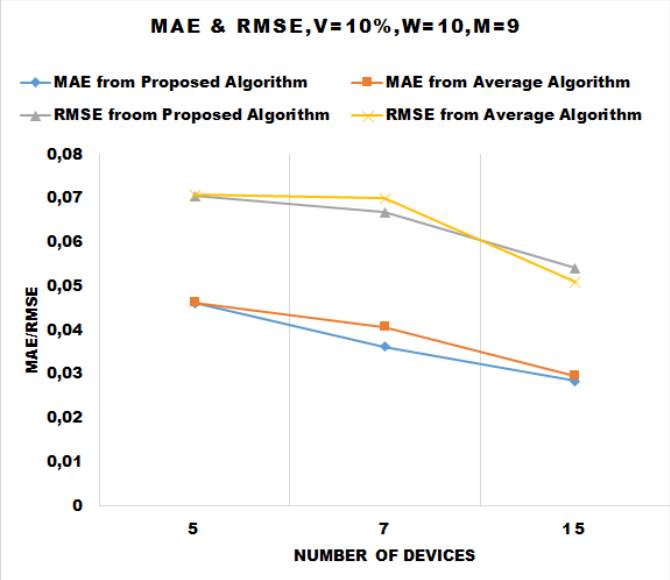


Εικόνα 9: MAE και RMSE για V=5% και M=9

Στις Εικόνες 8 και 9 παρουσιάζονται τα αποτελέσματα του προτεινόμενου αλγορίθμου σε σύγκριση με τον AM μηχανισμό για $M = 4$ και $M = 9$ αντίστοιχα και με ποσοστό ελλιπών τιμών $V = 5\%$. Μία προσεκτική ματιά στα αποτελέσματα είναι ικανή ώστε να μας επιβεβαιώσει τα αποτελέσματα του προηγούμενου πειραματικού σεναρίου μιας και στο παρών πειραματικό σενάριο ο αλγόριθμος μας παρουσιάζει καλύτερα αποτελέσματα για $M = 4$ και χειρότερα αποτελέσματα για $M = 9$. Ο αριθμός των IoT συσκευών επηρεάζει φανερά και τα δύο σενάρια σε αντίθετη ωστόσο κατεύθυνση. Τέλος για το παρόν σενάριο η τιμή του MAE κυμαίνεται κοντά στο 0.05 για $M = 4$ και 0.03 για $M = 9$.

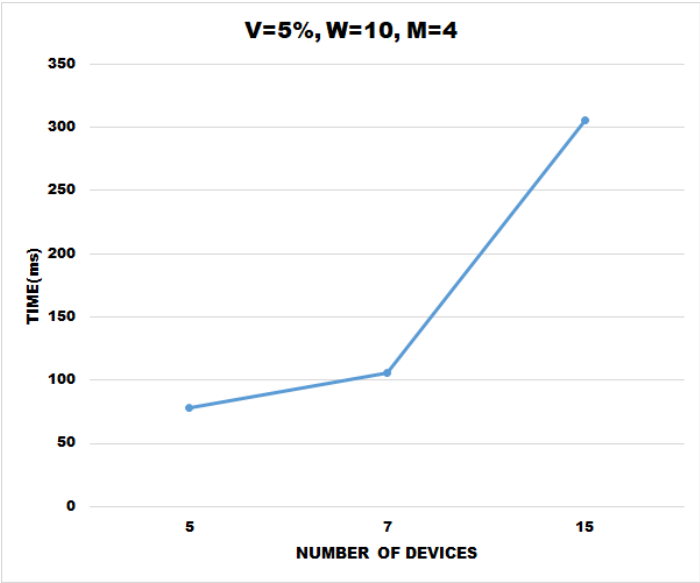


Εικόνα 10:MAE και RMSE για V=10% και M=4

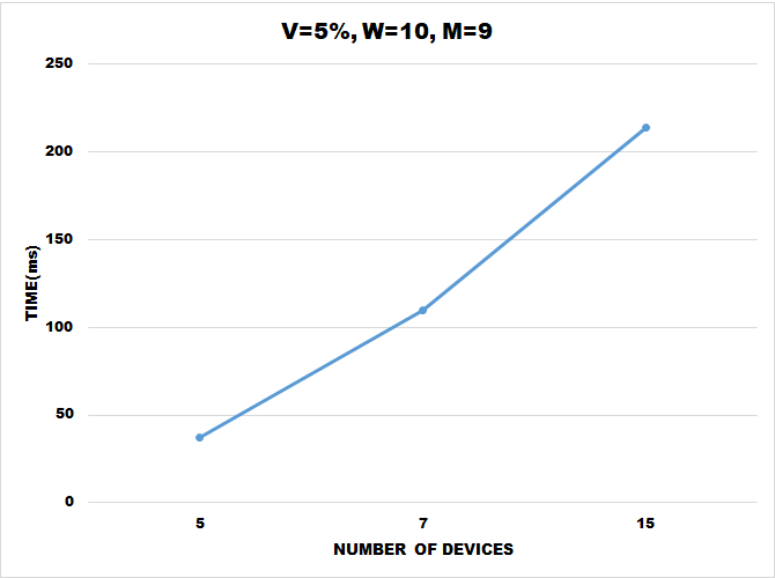


Εικόνα 11:MAE και RMSE για V=10 και M=9

Στις Εικόνες 10 και 11 παρουσιάζονται τα αποτελέσματα του προτεινόμενου αλγορίθμου σε σύγκριση με τον AM μηχανισμό για $M = 4$ και $M = 9$ αντίστοιχα και με ποσοστό ελλιπών τιμών $V = 10\%$. Η αύξηση του ποσοστού των ελλιπών τιμών οδηγεί πάλι σε παρόμοια συμπεράσματα με τα προηγούμενα δύο πειραματικά σενάρια. Στο ζεύγος αυτό των πειραματικών σεναρίων παρατηρούμε ότι η αύξηση του αριθμού των κόμβων οδηγεί σε μειωμένη τιμή των μετρικών MAE και RMSE. Αυτό σημαίνει ότι ο αλγόριθμος που προτείνεται καταφέρνει να επωφεληθεί από την αύξηση του αριθμού των ελλιπών τιμών που απαιτούν αντικατάσταση με τις τιμές του MAE να κυμαίνονται περίπου στο 0.065 για $M = 4$ και στο 0.075 $M = 9$.

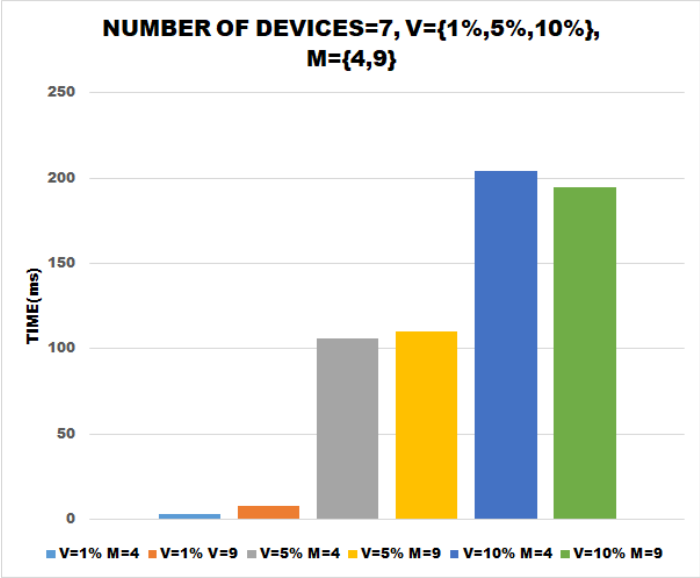


Εικόνα 12:Χρόνος αντικατάστασης για V=5% και M=4



Εικόνα 13:Χρόνος αντικατάστασης για V=5% και M=9

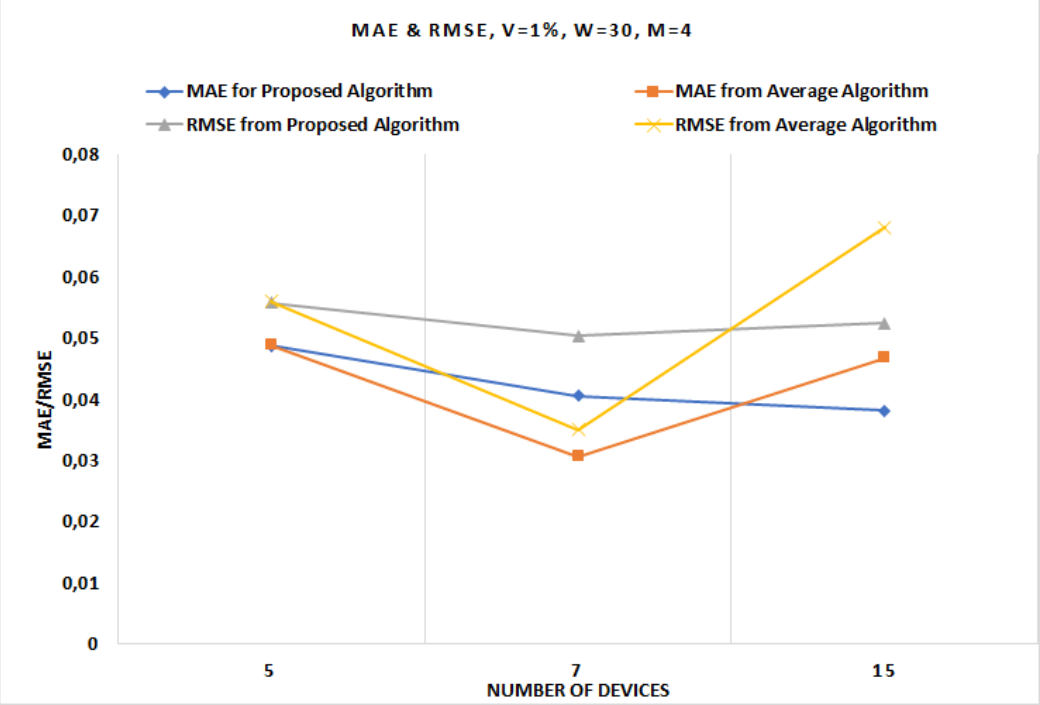
Ένα ακόμη ζεύγος πειραμάτων που διεξήγαμε προκειμένου να ελέγξουμε την απόδοση του προτεινόμενου αλγορίθμου είναι για το χρόνο που απαιτείται για την αντικατάσταση μιας ελλείπουσας τιμής από τον προτεινόμενο αλγόριθμο. Στις Εικόνες 12 και 13 παρουσιάζουμε τα αποτελέσματα για $M \in \{4,9\}$ και $V = 5\%$. Παρατηρώντας τα αποτελέσματα των πειραμάτων στις Εικόνες 11 και 12 βλέπουμε ότι ο χρόνος αντικατάστασης μιας ελλείπουσας τιμής αυξάνεται όσο αυξάνεται και ο αριθμός των IoT συσκευών που συμμετέχουν στο δίκτυο. Ακόμη παρά το γεγονός της αύξησης των μεταβλητών που καταγράφουν οι IoT συσκευές ο χρόνος αντικατάστασης επηρεάζεται θετικά π.χ. $M = 9$. Σε κάθε περίπτωση ο χρόνος αντικατάστασης είναι μικρότερος των 350 ms που σημαίνει ότι γίνονται περίπου τρείς αντικαταστάσεις ελλιπών τιμών ανά δευτερόλεπτο.



Εικόνα 14:Χρόνος αντικατάστασης για διαφορετικές τιμές των V και M

Στην Εικόνα 14 παρουσιάζουμε τα αποτελέσματα του προτεινόμενου αλγορίθμου για διαφορετικές τιμές των V και M κρατώντας σταθερό των αριθμό των IoT συσκευών που συμμετέχουν στο δίκτυο. Η αύξηση των V και M οδηγεί σε αύξηση του χρόνου που χρειάζεται για την ολοκλήρωση του προτεινόμενου αλγορίθμου αντικατάστασης των ελλιπών τιμών. Ο λόγος που προκαλεί αυτήν την αύξηση στο χρόνο βρίσκεται στην πολυπλοκότητα των υπολογισμών που πρέπει να γίνουν καθώς η αύξηση των μεταβλητών που καταγράφονται αναγκάζει τον αλγόριθμο σε άντληση πληροφοριών από μεγαλύτερες δομές. Ωστόσο ο χρόνος που απαιτείται παραμένει χαμηλός, μικρότερος περίπου των 200 ms που απεικονίζει την απόδοση και την ικανότητα του προτεινόμενου αλγορίθμου να υποστηρίζει εφαρμογές σε πραγματικό χρόνο. Ο μηχανισμός παρακολούθησης είναι ικανός να εντοπίζει γρήγορα τις ελλείπουσες τιμές και να παρέχει τις κατάλληλες τιμές για να τις αντικαταστήσουν.

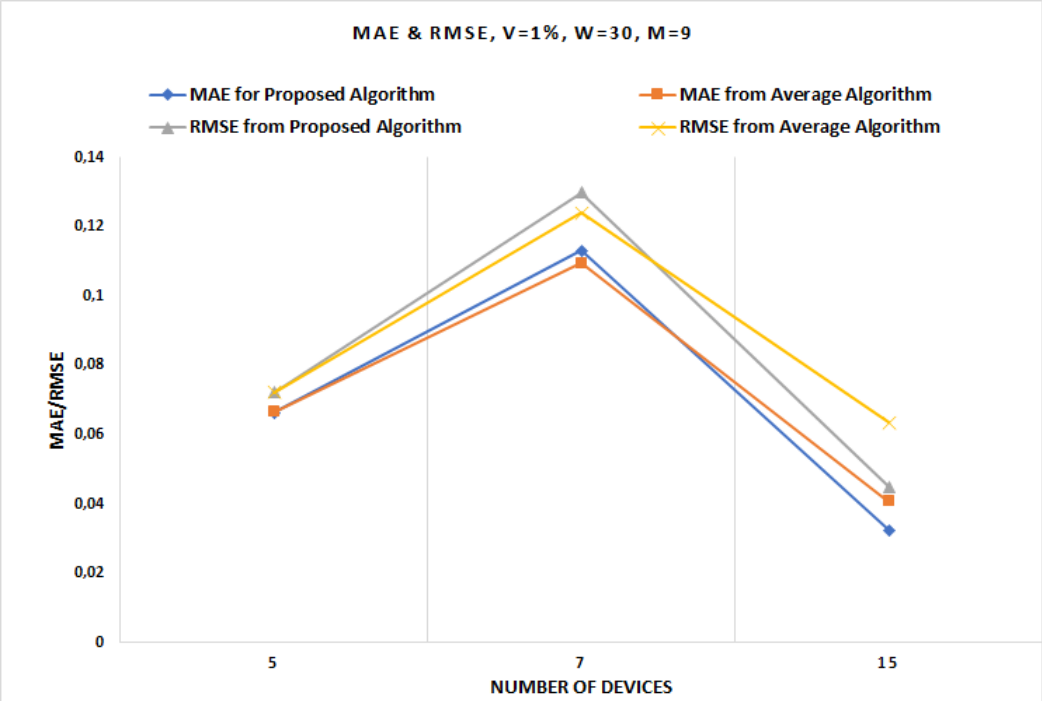
7.4.β Πειράματα με W=30.



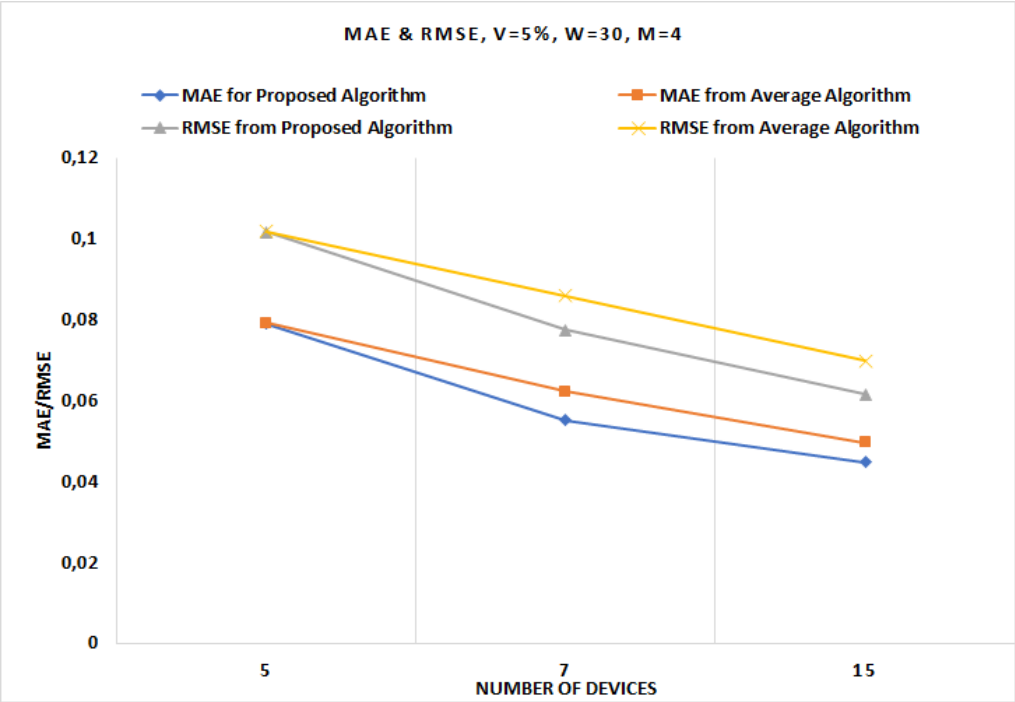
Εικόνα 15:MAE & RMSE για V=1%, W=30 και M=4

Στις εικόνες 15 και 16 παρουσιάζονται τα αποτελέσματα του προτεινόμενου αλγορίθμου σε σύγκριση με τον AM μηχανισμό για $M = 4$ και $M = 9$ αντίστοιχα και με ποσοστό ελλιπών τιμών $V = 1\%$. Κοιτάζοντας προσεκτικά τα αποτελέσματα που αντικατοπτρίζονται στις γραφικές παραστάσεις μπορεί κανείς εύκολα να καταλάβει ότι ο αλγόριθμος που προτείνεται υπερτερεί έναντι του AM μηχανισμού για τις ακραίες τιμές του πλήθους των συσκευών ενώ υστερεί για 7 IoT συσκευές. Ειδικότερα ο προτεινόμενος αλγόριθμος για $M = 4$ παρουσιάζει σχεδόν σταθερές τις τιμές του σφάλματος με μεταβολή της τάξεως των 0.01 και 0.005 μονάδων για MAE και RMSE αντίστοιχα. Τέλος για 7 IoT συσκευές τα διαγράμματα παρουσιάζουν ακριβώς αντίθετη συμπεριφορά καθώς για $M = 4$ οι μετρικές του σφάλματος παρουσιάζουν τη μικρότερη τιμή ενώ για $M = 9$ τη μεγαλύτερη τιμή τους. Συμπεραίνουμε λοιπόν ο μεγάλος αριθμός πλήθους IoT συσκευών και

μεταβλητών δεν επηρεάζει αρνητικά την απόδοση του προτεινόμενου αλγορίθμου που μπορεί να αντικαταστήσει αποτελεσματικά τις ελλείπουσες τιμές.



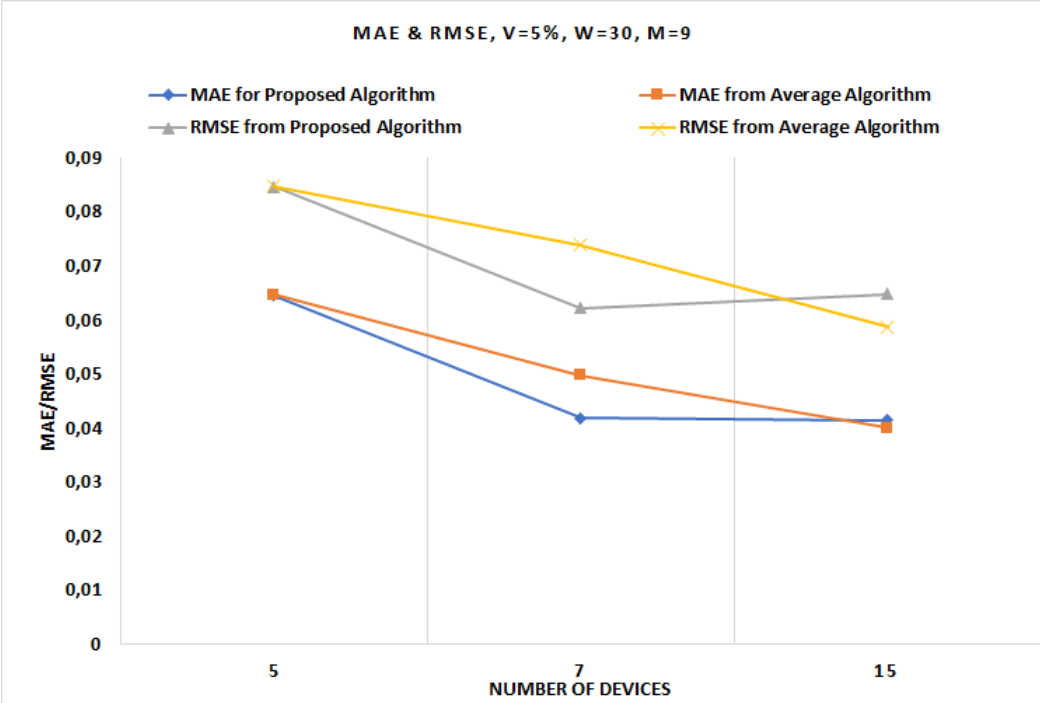
Εικόνα 16:MAE & RMSE για V=1%, W=30, M=9



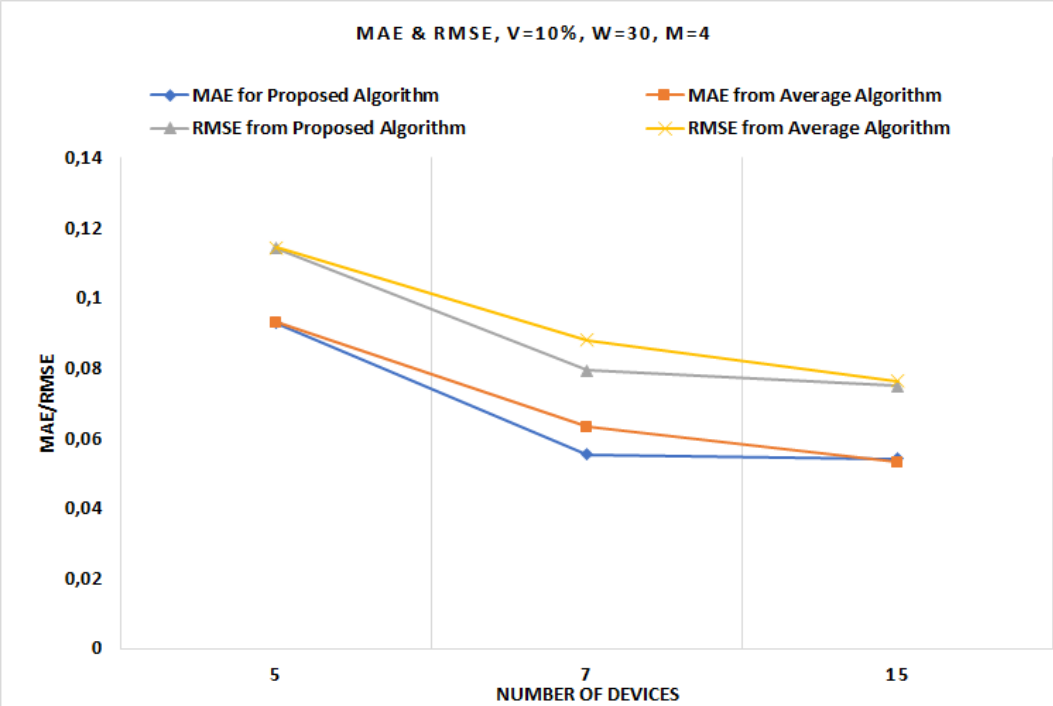
Εικόνα 17:MAE & RMSE για V=5%, W=30 και M=4

Στις εικόνες 17 και 18 παρουσιάζονται τα αποτελέσματα του προτεινόμενου αλγορίθμου σε σύγκριση με τον AM μηχανισμό για $M = 4$ και $M = 9$ αντίστοιχα και με ποσοστό ελλιπών τιμών $V = 5\%$. Παρατηρούμε λοιπόν ότι η αύξηση των συσκευών και στα δύο διαγράμματα επιφέρει μείωση των μετρικών του σφάλματος. Συγκρίνοντας τον προτεινόμενο αλγόριθμο με τον AM μηχανισμό παρατηρούμε ότι ο προτεινόμενος αλγόριθμος παρουσιάζει σαφώς καλύτερα αποτελέσματα για $M = 4$

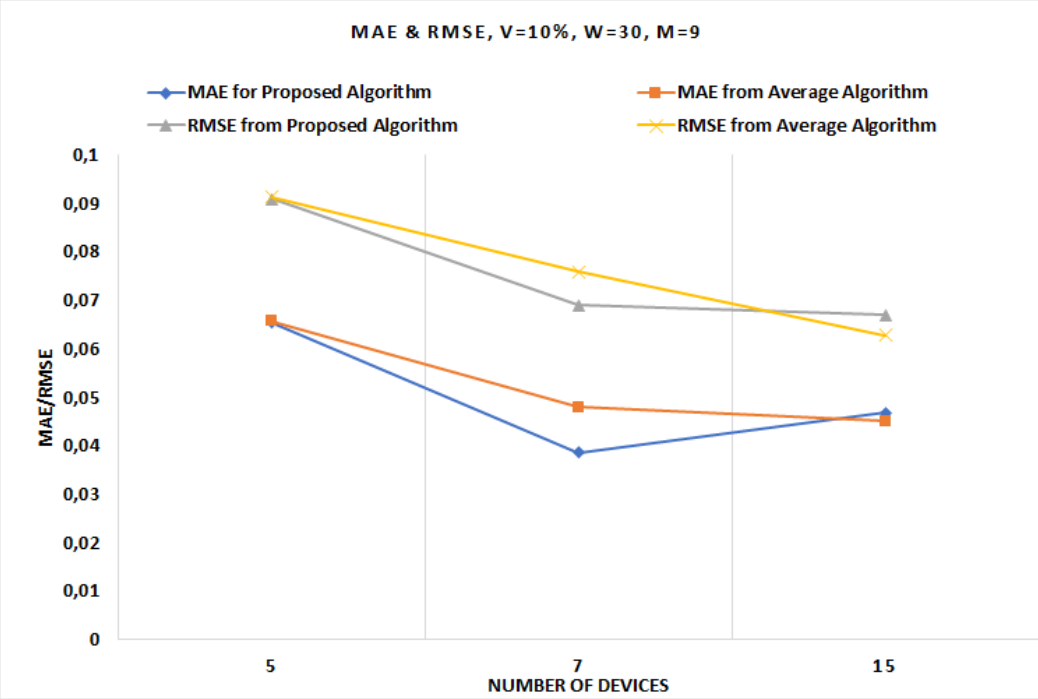
ενώ για $M = 9$ ο AM μηχανισμός υπερτερεί μόνο για 15 IoT συσκευές όπου παρουσιάζει και μικρότερο σφάλμα από τον προτεινόμενο αλγόριθμο. Επομένως μπορούμε να συμπεράνουμε ότι η αύξηση των IoT συσκευών όπως και στο προηγούμενο ζεύγος πειραμάτων δεν επηρεάζει αρνητικά τον αλγόριθμο αλλά παρουσιάζει καλύτερα αποτελέσματα σε σχέση με την απόδοση του για 4 IoT συσκευές.



Εικόνα 18:MAE & RMSE για V=5%, W=30, M=9

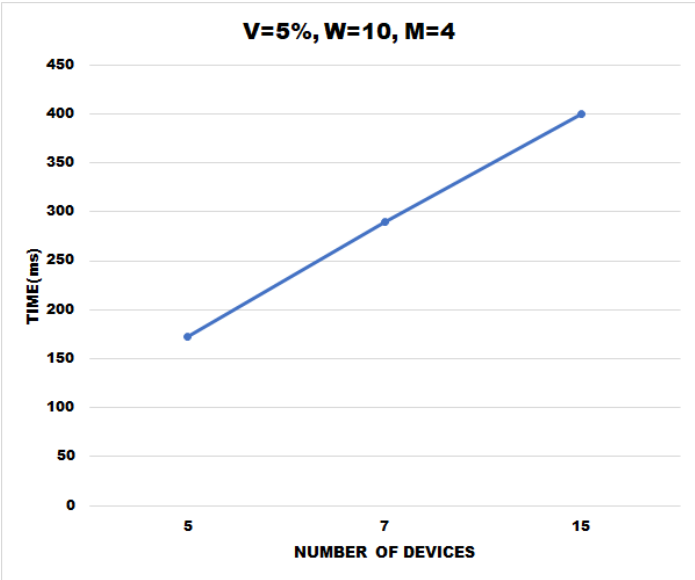


Εικόνα 19: MAE & RMSE για V=10%, W=30, M=4

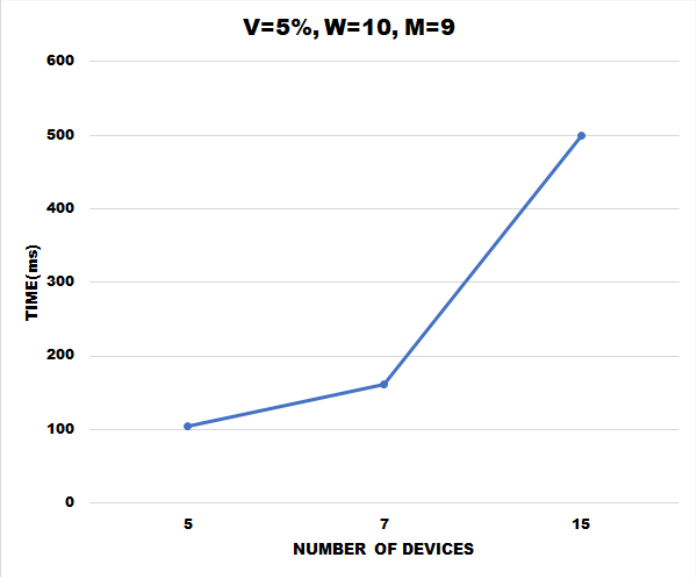


Εικόνα 20: MAE & RMSE για V=10%, W=30, M=9

Στις εικόνες 19 και 20 παρουσιάζονται τα αποτελέσματα του προτεινόμενου αλγορίθμου σε σύγκριση με τον AM μηχανισμό για $M = 4$ και $M = 9$ αντίστοιχα και με ποσοστό ελλιπών τιμών $V = 10\%$. Σε αυτό το ζεύγος των πειραμάτων είναι εύκολο κανείς να παρατηρήσει ότι ο προτεινόμενος αλγόριθμος παρουσιάζει καλύτερα αποτελέσματα από τον AM μηχανισμό για 5 και 7 IoT συσκευές ενώ εμφανίζει μεγαλύτερη τιμή για τη μετρική του MAE για 15 IoT συσκευές τόσο για $M = 4$ όσο και $M = 9$. Ακόμη συγκρίνοντας τις τιμές των μετρικών για τον υπολογισμό σφαλμάτων παρατηρούμε ότι αυτές είναι χαμηλότερες για $M = 9$. Έτσι μπορούμε να καταλήξουμε στο συμπέρασμα ότι η απόδοση του αλγορίθμου για ποσοστό ελλιπών τιμών $V = 10\%$ η απόδοση επηρεάζεται τόσο από το πλήθος των συσκευών όσο και από το πλήθος των μεταβλητών.

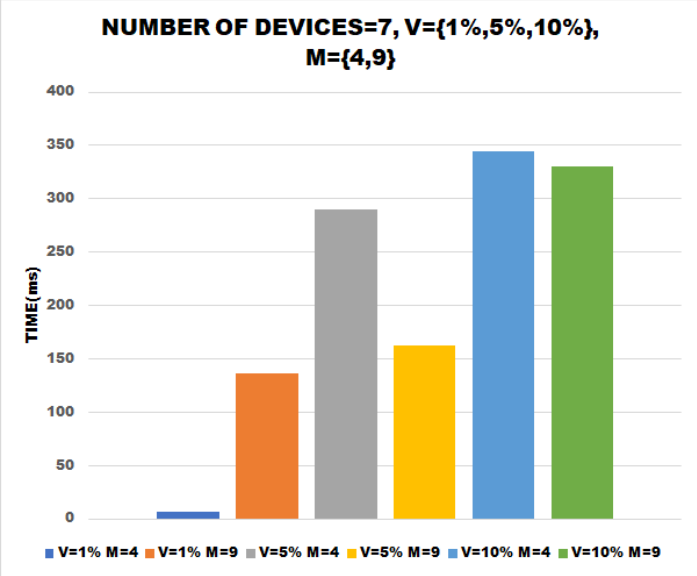


Εικόνα 21:Χρόνος αντικατάστασης για V=5% και M=4



Εικόνα 22:Χρόνος αντικατάστασης για V=5% και M=9

Στις Εικόνες 21 και 22 παρουσιάζουμε τα αποτελέσματα για $M \in \{4,9\}$ και $V = 5\%$ για τον χρόνο που απαιτείται για την αντικατάσταση μιας ελλείπουσας τιμής από τον προτεινόμενο αλγόριθμο. Παρατηρώντας τα αποτελέσματα των πειραμάτων στις Εικόνες 21 και 22 βλέπουμε ότι ο χρόνος αντικατάστασης μιας ελλείπουσας τιμής αυξάνεται όσο αυξάνεται και ο αριθμός των IoT συσκευών που συμμετέχουν στο δίκτυο. Ωστόσο το γεγονός της αύξησης των μεταβλητών που καταγράφουν οι IoT συσκευές επηρεάζει το χρόνο αντικατάστασης θετικά για 5 και 7 IoT συσκευές ενώ αρνητικά για 15 IoT συσκευές. Σε κάθε περίπτωση ο χρόνος αντικατάστασης είναι μικρότερος ή ίσος των 500 ms που σημαίνει ότι γίνονται τουλάχιστον δύο αντικαταστάσεις ελλιπών τιμών ανά δευτερόλεπτο.



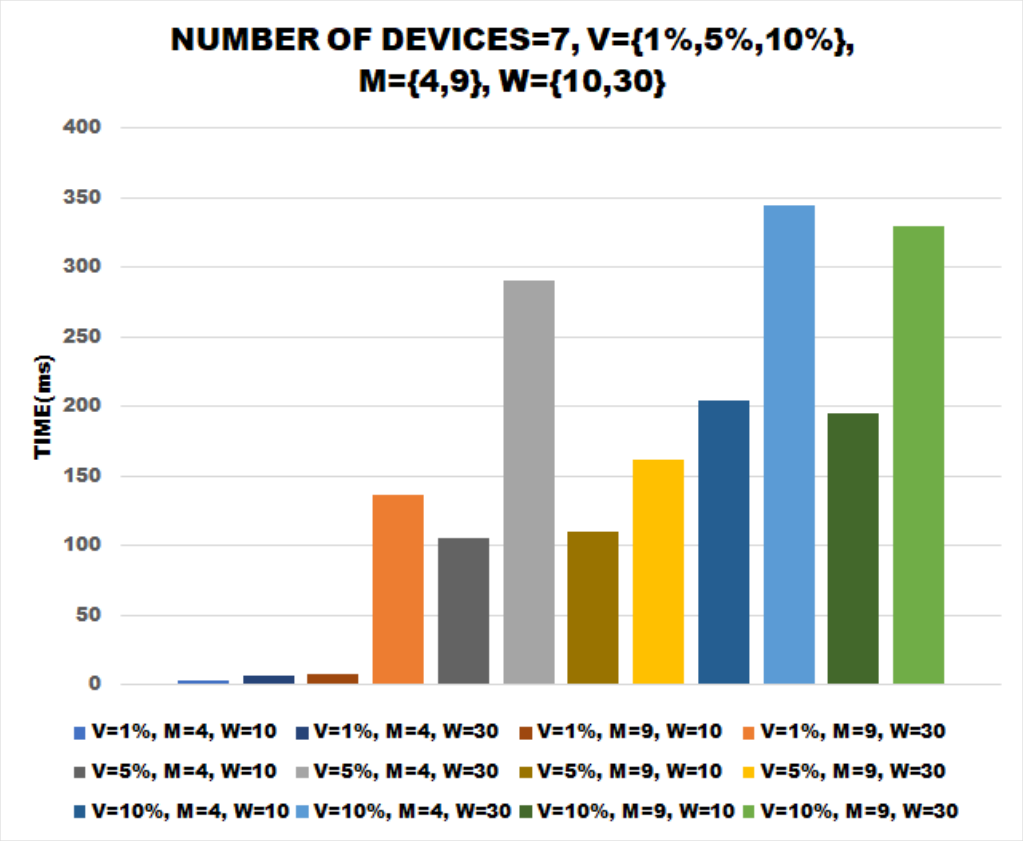
Εικόνα 23:Χρόνος αντικατάστασης για διαφορετικές τιμές των V και M και W=30

Στην Εικόνα 23 παρουσιάζουμε τα αποτελέσματα του προτεινόμενου αλγορίθμου για διαφορετικές τιμές των V και M κρατώντας σταθερό των αριθμό των IoT συσκευών που συμμετέχουν στο δίκτυο. Η αύξηση των V και M οδηγεί σε

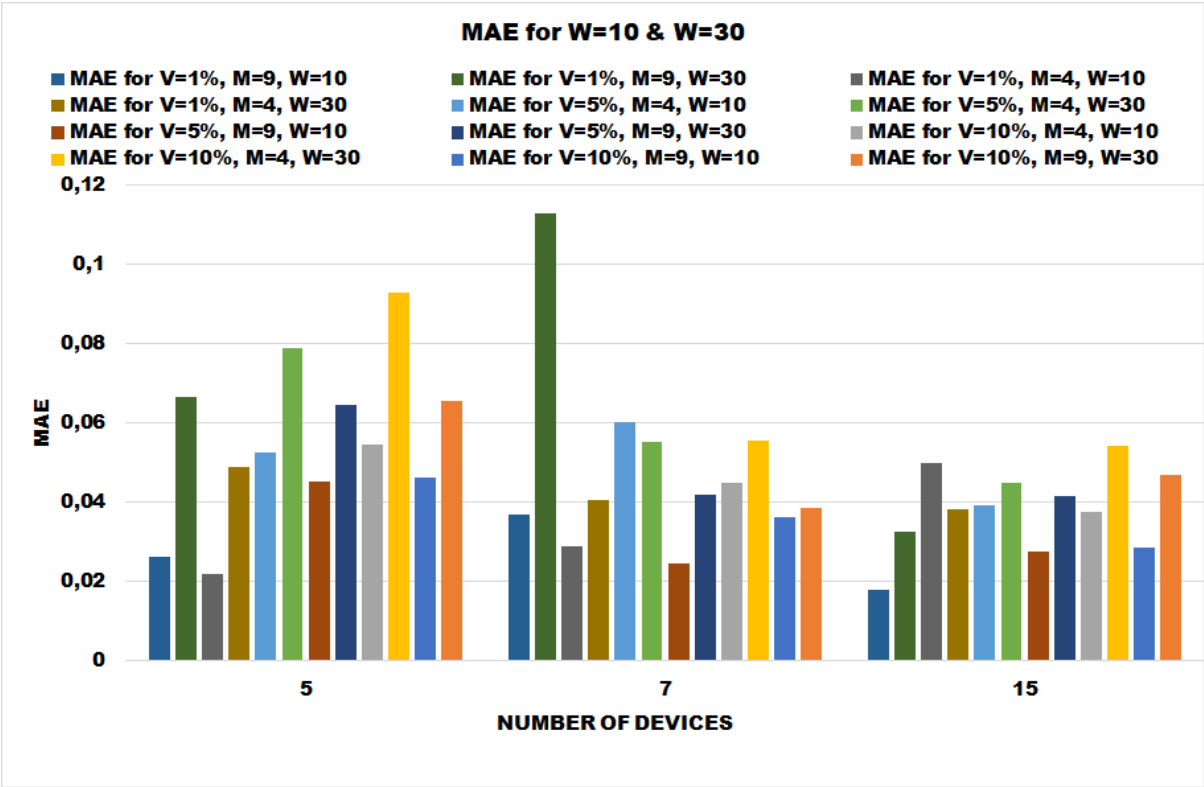
αύξηση του χρόνου που χρειάζεται για την ολοκλήρωση του προτεινόμενου αλγορίθμου αντικατάστασης των ελλιπών τιμών. Ο λόγος που προκαλεί αυτήν την αύξηση στο χρόνο βρίσκεται στην πολυπλοκότητα των υπολογισμών που πρέπει να γίνουν καθώς η αύξηση των μεταβλητών που καταγράφονται αναγκάζει τον αλγόριθμο σε άντληση πληροφοριών από μεγαλύτερες δομές. Ωστόσο ο χρόνος που απαιτείται παραμένει χαμηλός, λιγότερο από 350 ms που απεικονίζει την απόδοση και την ικανότητα του προτεινόμενου αλγορίθμου να υποστηρίζει εφαρμογές σε πραγματικό χρόνο. Ο μηχανισμός παρακολούθησης είναι ικανός να εντοπίζει γρήγορα τις ελλείπουσες τιμές και να παρέχει τις κατάλληλες τιμές για να τις αντικαταστήσουν.

7.4.γ Σύγκριση απόδοσης του προτεινόμενο αλγορίθμου για διαφορετικά W.

Στην παρούσα παράγραφο παρουσιάζεται η απόδοση του αλγορίθμου ανάμεσα στις δύο τιμές της μεταβλητής W που εκφράζει τον αριθμό των πιο πρόσφατων χρονικών τιμών που λάβαμε υπόψιν στους υπολογισμούς του αλγορίθμου. Για την σύγκριση αυτή λάβαμε υπόψιν τις τιμές που παρουσιάζει η μετρική MAE για διαφορετικό πλήθος IoT συσκευών και μεταβλητών και διαφορετικό ποσοστό χαμένων τιμών. Τέλος λάβαμε υπόψιν και το χρόνο που χρειάζεται κατά μέσο όρο η αντικατάσταση μιας χαμένης τιμής.



Εικόνα 24: Χρόνος αντικατάστασης για διαφορετικά V, M, W και 7 IoT συσκευές



Εικόνα 25: MAE για V = {1%, 5%, 10%}, M = {4, 9}, W = {10, 30}

Στις Εικόνες 24 και 25 φαίνονται τα αποτελέσματα σχετικά με τον χρόνο που χρειάζεται ο αλγόριθμος για την αντικατάσταση μιας ελλείπουσας τιμής και οι τιμές της μετρικής MAE για διαφορετικό πλήθος IoT συσκευών και μεταβλητών και διαφορετικό ποσοστό ελλιπών τιμών αντίστοιχα. Παρατηρώντας λοιπόν τα δύο αυτά διαγράμματα μπορεί κανείς να δει ότι τόσο ο χρόνος αντικατάστασης για μία ελλείπουσα τιμή όσο και η τιμή του MAE αυξάνονται για W=30. Το γεγονός αυτό μας οδηγεί στο συμπέρασμα ότι ο προτεινόμενος αλγόριθμος παρουσιάζει καλύτερη απόδοση και μπορεί να υποστηρίξει πιο αποτελεσματικά εφαρμογές πραγματικού χρόνου λαμβάνοντας υπόψιν μικρότερο αριθμό πρόσφατων καταγεγραμμένων τιμών. Αυτό το χαρακτηριστικό του αλγορίθμου δημιουργεί ένα μεγάλο πλεονέκτημα καθώς μειώνει τον αριθμό των δεδομένων που πρέπει να κρατηθεί στο επίπεδο του edge για να αντιμετωπιστούν αποτελεσματικά οι ελλείπουσες τιμές σε ένα σύνολο δεδομένων.

ΚΕΦΑΛΑΙΟ 8 Συμπεράσματα

Η διαδικασία συμπλήρωσης των ελλιπών τιμών αποτελεί μία από τις πιο σημαντικές διεργασίες που πρέπει να γίνονται πάνω στα δεδομένα τα οποία διαθέτουμε προκειμένου τα αποτελέσματα από την ανάλυση των δεδομένων να είναι ακριβή για την αποτελεσματική λήψη αποφάσεων. Στην περίπτωση που επικεντρωνόμαστε σε εφαρμογές με IoT υποδομή η υιοθέτηση αποτελεσματικών τεχνικών, για τη συμπλήρωση των δεδομένων που λείπουν, που είναι ικανές να παρέχουν το τελικό αποτέλεσμα στον ελάχιστο χρόνο με τη μέγιστη δυνατή απόδοση είναι μία από τις σημαντικότερες αποφάσεις που καλούμαστε να πάρουμε. Η χρήση του IoT μας προσφέρει πολυάριθμες τεχνολογίες που μπορούμε να χρησιμοποιήσουμε ώστε να μπορούν να συλλεχθούν και να επεξεργαστούν τεράστιες ποσότητες δεδομένων από πολυάριθμες συσκευές. Η λειτουργία του edge computing μπορεί να υλοποιηθεί με τη χρήση ενός συνόλου κόμβων που θα λαμβάνουν τα δεδομένα και θα εκτελούν μια ελαφριά επεξεργασία πάνω σε αυτά περιορίζοντας έτσι την καθυστέρηση στην εξαγωγή αποτελεσμάτων.

Σε αυτήν την εργασία προτάθηκε ένας αλγόριθμος που εφαρμόζεται πάνω σε ένα δίκτυο αποτελούμενο από IoT συσκευές και ένα σύνολο edge κόμβων. Το συνολικό μοντέλο που προτάθηκε περιέχει ένα μηχανισμό που είναι υπεύθυνος για την ανίχνευση των ελλιπών τιμών στις εισερχόμενες ροές δεδομένων στους edge κόμβους. Σε περίπτωση που εντοπιστεί μία ροή που περιέχει μία ή περισσότερες ελλείπουσες τιμές γίνεται εφαρμογή του προτεινόμενου αλγορίθμου αντικατάστασης των τιμών αυτών. Ο προτεινόμενος αλγόριθμος αντικατάστασης των ελλιπών τιμών αποτελείται από ένα συνολικό μοντέλο εύρεσης της συσχέτισης και από ένα ευρετικό μηχανισμό υπολογισμού της τιμής που θα αντικαταστήσει την ελλείπουσα τιμή. Ο μηχανισμός εύρεσης της συσχέτισης βασίζεται σε δύο ευρέως χρησιμοποιούμενες μεθόδους, την απόσταση Mahalanobis και τη μέθοδο Cosine Similarity. Από την άλλη ο προτεινόμενος μηχανισμός αντικατάστασης των ελλιπών τιμών προσπαθεί να λάβει υπόψιν τις τιμές που καταγράφει ένα σύνολο IoT συσκευών με υψηλή συσχέτιση με την συσκευή που η καταγραφή της περιέχει την ελλείπουσα τιμή. Όλα τα παραπάνω προσομοιώθηκαν σε ένα σύνολο πειραμάτων που παρουσιάστηκε παραπάνω προβάλλοντας τα δυνατά σημεία του αλγορίθμου.

Στο μέλλον θα γίνει προσπάθεια να υλοποιηθεί μια πιο σύνθετη μεθοδολογία για την αντικατάσταση των ελλιπών τιμών, λαμβάνοντας υπόψιν την αβεβαιότητα που υπάρχει πίσω από την υιοθέτηση συγκεκριμένων ομότιμων συσκευών στην επεξεργασία που απαιτείται πάνω στα δεδομένα.

ΠΑΡΑΡΤΗΜΑ ΚΩΔΙΚΕΣ

Το παράρτημα αυτό περιέχει ορισμένα από τα πιο σημαντικά κομμάτια κώδικα που χρησιμοποιήθηκαν.

1. Υπολογισμός του Cosine Similarity

```
public ArrayList<ArrayList<Double>> cosineSimilarity(ArrayList<ArrayList<ArrayList<Double>>> t2DList) {
    double dotProduct = 0.0;
    double normA = 0.0;
    double normB = 0.0;
    double cosineSimilarity;
    int firstDs = t2DList.size();//number of nodes
    int secDs = t2DList.get(0).size();//number of times
    int time = secDs-1;
    int thirdDs = t2DList.get(0).get(0).size();//number of variables
    Double Ai,Bi;
    for(int q=0;q<firstDs;q++){
        CosSimil.add(q,new ArrayList<Double>());
    }
    for(int i=0;i<firstDs;i++){
        for(int j=0;j<firstDs;j++){
            dotProduct = 0.0;
            normA=0.0;
            normB=0.0;
            for(int k=0;k<thirdDs;k++){
                if((t2DList.get(i).get(time).get(k)!= -1)&&(t2DList.get(j).get(time).get(k)!= -1)){//remove missing values from calculation
                    Ai=t2DList.get(i).get(time).get(k);
                    Bi=t2DList.get(j).get(time).get(k);
                    dotProduct =dotProduct + Ai * Bi;
                    normA =normA + Ai*Ai;
                    normB =normB + Bi*Bi;
                }
            }
        cosineSimilarity=(double)dotProduct/(Math.sqrt(normA)*Math.sqrt(normB));
        //System.out.println("time: "+time+"i: "+i+" j: "+j+" cosineSim: "+cosineSimilarity);
        CosSimil.get(i).add(j,cosineSimilarity);
        }
        System.out.println(CosSimil.get(i)+" "+i);
    }
    return CosSimil;
}
```

2. Υπολογισμός της Απόστασης Mahalanobis

```
public ArrayList<ArrayList<Double>> mahalanobisNtoN(ArrayList<ArrayList<ArrayList<Double>>> t2DList){//node to node
    ///Mahalanobis distance is calculated to compare the correlation of the past values.
    int TrialP=10;
    int firstDs = t2DList.size();//number of nodes
    int secDs = t2DList.get(0).size();//number of times
    int thirdDs = t2DList.get(0).get(0).size();//number of variables
    int PastTimes=secDs-1;
    Integer a;
    Integer b;
    double[][] aA= new double[TrialP][thirdDs];
    double[][] bA =new double[TrialP][thirdDs];
    for(int n=0;n<firstDs;n++){
        mahDs.add(n,new ArrayList<Double>());
    }
    int past=PastTimes-TrialP;
    System.out.println("period: "+past);
    for(int q=0;q<firstDs;q++){
        for(int i=0; i<firstDs;i++){
            if(q!=i){
                for(int k=past; k<PastTimes; k++){
                    int cn=0;
                    for(int j=0; j<thirdDs; j++){
                        aA[k-past][j]=(double)t2DList.get(q).get(k).get(j);
                        bA[k-past][j]=(double)t2DList.get(i).get(k).get(j);
                    }
                }
                mahDs.get(q).add(i,distance(aA,bA));
            }
            if(q==i){
                mahDs.get(q).add(i,0.0);
            }
        }
        System.out.println(mahDs.get(q)+" "+q);
    }
    return mahDs;
}
```

3. Υπολογισμός της Τελικής Συσχέτισης

```
public void FinalCorrel(ArrayList<ArrayList<ArrayList<Double>>> t2DList){//make final correlation
    double fcorrel;
    int firstDs = t2DList.size();//number of nodes
    int secDs = t2DList.get(0).size();//number of times
    int thirdDs = t2DList.get(0).get(0).size();//number of variables
    Times tl = new Times();


    for(int j=0;j<firstDs;j++){
        tl.AddArrayListN();
        for(int k=0;k<firstDs;k++){
            if(mahDs.get(j).get(k)!= 0){
                fcorrel=(1/mahDs.get(j).get(k))*CosSimil.get(j).get(k);
                tl.timeVal=secDs-1;
                tl.InsertCorr(j, k, fcorrel);
            }else{
                fcorrel=0;
                tl.timeVal=secDs-1;
                tl.InsertCorr(j, k, fcorrel);

            }

        }

    }

    FinalC.add(tl);//adding an arraylist for each time

}
```

4. Υπολογισμός της χαμένης τιμής από τον προτεινόμενο αλγόριθμο

```
public double ComputationMissingVal(int clock,ArrayList<ArrayList<ArrayList<Double>>> t2DList,ArrayList<MissNodesChars> MissN,int counter){
    double missnumber=0;
    int MNodeID=-1;
    int MVarID=-1;
    double sumCosine = 0 ;
    int numOfTopN;
    double cosSimProdVal=0.0;
    int MaxNodeId;
    numOfTopN=0;
    MNodeID= MissN.get(counter).MNodeiD;
    MVarID=MissN.get(counter).MVarId;
    System.out.println("MNodeID: "+MNodeID+" "+"clock: "+MissN.get(counter).timeofM);
    for(int r=0;r<FinalC.size();r++){
        if(FinalC.get(r).timeVal==clock){
            for(int g=0;g<FinalC.get(r).MCorrN.size();g++){
                if(g==MNodeID){
                    numOfTopN=0;
                    for(int v=0;v<FinalC.get(r).MCorrN.get(g).size();v++){
                        if(numOfTopN<4){
                            MaxNodeId=FinalC.get(r).MCorrN.get(g).get(v).Nodeid;
                            if(t2DList.get(MaxNodeId).get(clock).get(MVarID)!=(-1)){
                                numOfTopN++;
                                sumCosine=sumCosine+CosSimil.get(MNodeID).get(MaxNodeId);
                                cosSimProdVal=cosSimProdVal+(CosSimil.get(MNodeID).get(MaxNodeId)*t2DList.get(MaxNodeId).get(clock).get(MVarID));
                            }
                        }
                    }
                    missnumber=cosSimProdVal/sumCosine;
                }
            }
        }
    }
    return missnumber;
}
```

5. Υπολογισμός της χαμένης τιμής από τον AM μηχανισμό

```
public double CalculateAverageVal(int clock,ArrayList<ArrayList<ArrayList<Double>>> t2DList,ArrayList<MissNodesChars> MissN,int counter)
throws FileNotFoundException, IOException{
    double missnumber=0.0;
    int MNodeID=-1;
    int MVarID=-1;
    int MaxNodeId;
    int nOfNodes=0;
    double predictV=0.0;
    MNodeID= MissN.get(counter).MNodeiD;
    MVarID=MissN.get(counter).MVarId;
    System.out.println("MNodeID: "+MNodeID+" "+"clock: "+MissN.get(counter).timeofM);
    for(int r=0;r<FinalC.size();r++){
        if(FinalC.get(r).timeVal==clock){
            for(int g=0;g<FinalC.get(r).MCorrN.size();g++){
                if(g==MNodeID){
                    for(int v=0;v<FinalC.get(r).MCorrN.get(g).size();v++){
                        MaxNodeId=FinalC.get(r).MCorrN.get(g).get(v).Nodeid;
                        if(t2DList.get(MaxNodeId).get(clock).get(MVarID)!=(-1)){
                            nOfNodes++;
                            missnumber = missnumber + t2DList.get(MaxNodeId).get(clock).get(MVarID);
                        }
                    }
                }
            }
        }
    }
    predictV=missnumber/nOfNodes;
    return predictV;
}
```

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] WEIL M., SOUISSI M., "L'Internet des objets: concept ou réalité?", Annales des Mines – Réalités industrielles, no. 4, pp. 90–96, 2010.

[2] IEEE_IoT_Towards_Definition_Internet_of_Things_Revision1_27MAY15.pdf

[3] https://dzone.com/articles/introduction-to-iot-sensors

[4] https://www.tutorialspoint.com/internet_of_things/internet_of_things_hardware.htm

[5] https://data-flair.training/blogs/iot-hardware/

[6] https://engineering.eckovation.com/iot-stack/

[7] https://www.i-scoop.eu/internet-of-things-guide/iot-technology-stack-devices-gateways-platforms/

[8] https://internetofthingsagenda.techtarget.com/definition/actuator

[9] https://www.ubuntupit.com/best-iot-operating-system-for-your-iot-devices/

[10] https://www.postscapes.com/internet-of-things-software-guide/#platforms

[11] https://www.watelectronics.com/arduino-uno-board-tutorial-and-its-applications/

[12] https://el.wikipedia.org/wiki/Arduino

[13] https://www.watelectronics.com/know-all-about-raspberry-pi-board-technology/

[14] https://www.raspberrypi.org/products/raspberry-pi-4-model-b/specifications/

[15] https://thenewstack.io/10-diy-development-boards-iot-prototyping/

[16] https://maker.pro/intel-edison/tutorial/get-started-with-intel-edison-open-source-development-board

[17] https://www.gartner.com/en/information-technology/glossary/edge-computing

[18] https://www.cloudflare.com/learning/serverless/glossary/what-is-edge-computing/

[19] https://www.networkworld.com/article/3224893/what-is-edge-computing-and-how-it-s-changing-the-network.html

[20] https://www.theverge.com/circuitbreaker/2018/5/7/17327584/edge-computing-cloud-google-microsoft-apple-amazon

[21] https://towardsdatascience.com/all-about-missing-data-handling-b94b8b5d2184

[22] https://towardsdatascience.com/the-use-of-knn-for-missing-values-cf33d935c637

[23] https://it.toolbox.com/blogs/carmashoemaker/the-advantages-risks-and-inevitability-of-edge-computing-121218

[24] https://towardsdatascience.com/you-need-to-move-from-cloud-computing-to-edge-computing-now-e8759eb9690f

[25] https://www.stackpath.com/edge-academy/edge-computing/

[26] https://electricalfundablog.com/iot-edge-computing-types-architecture-advantages-applications/#2_Consistent_Operations_with_Sporadic_Connectivity

[27] https://www.ionos.com/digitalguide/server/know-how/edge-computing/

[28] https://medium.com/datadriveninvestor/4-stages-of-iot-architecture-explained-in-simple-words-b2ea8b4f777f

[29] https://www.avsystem.com/blog/what-is-iot-architecture/

[30] https://www.edureka.co/blog/what-is-iot/

[31] Martin Breitbach, Dominik Schäfer, Janick Edinger, Christian Becker "Context-Aware Data and Task Placement in Edge Computing Environments", 2019 IEEE International Conference on Pervasive Computing and Communications (PerCom)

[32] "Dynamic Collaboration of Centralized & Edge Processing for Coordinated Data Management in an IoT Paradigm", Roger Young, Sheila Fallon, Paul Jacob Software Research Institute, Athlone Institute of Technology, Athlone, Co Westmeath

[33] Implementing an Edge-Fog-Cloud architecture for stream data management at https://www.researchgate.net/publication/322962512

[34] Dolui, K. and Datta, K. S., 'Comparison of edge computing implementations: Fog computing, Cloudlet and mobile edge computing', IEEE GIoTS, 2017.

[35] Rethinking the Internet of Things, A Scalable Approach to Connecting Everything, Francis DaCosta

[36] Internet of Things: Evolutions and Innovations (Computer Engineering: Digital Tools and Uses) by Nasreddine Bouhaï (Editor), Imad Saleh (Editor)

[37] Internet of Things Architecture, Protocols and Standards by Simone Cirani, Gianluigi Ferrari, Marco Picone

[38] Proper imputation techniques for missing values in data sets at https://www.researchgate.net/publication/312568863

[39] Performance Analysis of Machine Learning Algorithms for Missing Value Imputation at https://thesai.org/Downloads/Volume9No6/Paper_60-Performance_Analysis_of_Machine_Learning_Algorithms.pdf

[40] Comparison of the Most Influential Missing Data Imputation Algorithms for Healthcare at http://www.jaist.ac.jp/~razvan/publications/comparison_imputation_healthcare.pdf

[41] A ROBUST MISSING VALUE IMPUTATION METHOD MIFOIMPUTE FOR INCOMPLETE MOLECULAR DESCRIPTOR DATA AND COMPARATIVE ANALYSIS WITH OTHER MISSING VALUE IMPUTATION METHODS at https://arxiv.org/ftp/arxiv/papers/1312/1312.2859.pdf

[42] Multiple Imputation of Missing Values in Economic Surveys: Comparison of Competing Algorithms at https://www.nstac.go.jp/services/society_paper/25_04_02_Paper.pdf

[43] Intel Lab Data, http://db.csail.mit.edu/labdata/labdata.html
[44] Harth, N., Anagnostopoulos, C., 'Edge-centric Efficient Regression Analytics', IEEE EDGE, 2018.
[45] https://en.wikipedia.org/wiki/Mahalanobis_distance
[46] https://en.wikipedia.org/wiki/Cosine_similarity
[47] https://coral.ai/
[48] https://coral.ai/docs/dev-board/datasheet/
[49] Filtering Algorithms for Information Retrieval Models with Named Attributes and Proximity Operatorshttp://cgi.di.uoa.gr/~koubarak/publications/sigir04-filtering.pdf
[50] Review based on data clustering algorithms at https://www.researchgate.net/publication/261355115
[51] Outlier Detection Algorithms in Data Mining Systems at https://www.researchgate.net/publication/227252790_Outlier_Detection_Algorithms_in_Data_Mining_Systems
[52] A SURVEY OF OUTLIER DETECTION IN DATA MINING at https://www.researchgate.net/publication/280829045
[53] An Algorithm for Predictive Data Mining Approach in Medical Diagnosis at https://www.researchgate.net/publication/323539223
[54] Missing Data Imputation using Genetic Algorithm for Supervised Learning at https://thesai.org/Downloads/Volume8No3/Paper_60-Missing_Data_Imputation_using_Genetic_Algorithm.pdf
[55] Experimental analysis of methods for imputation of missing values in databases at https://sci2s.ugr.es/keel/pdf/specific/congreso/farhangfar_kurgan_pedrycz04.pdf
[56] Missing Value Imputation using Refined Mean Substitution at https://www.ijcsi.org/papers/IJCSI-9-4-3-306-313.pdf
[57] Data imputing using genetic algorithms (GA) at https://www.diva-portal.org/smash/get/diva2:1183382/FULLTEXT01.pdf
[58] Missing Value Imputation Method Based on Clustering and Nearest Neighbours at http://www.ijfcc.org/papers/54-T40006.pdf

[59] Ruiz-Alvarez, A. and Humphrey, M. (2012). A Model and Decision Procedure for Data Storage in Cloud Computing. 12th IEEE/ACM CCGrid, 2012.
[60] Jiang, L., et al., 'An IoT-Oriented Data Storage Framework in Cloud Computing Platform', IEEE TII, 2015, 10(2), 1443–1451.
[61] Shafagh, H., et al., 'Towards Blockchain-based Auditable Storage and Sharing of IoT Data', 9th ACM CCS Workshop, 2017.
[62] Fu, J.-S., et al., 'Secure Data Storage and Searching for Industrial IoT by Integrating Fog Computing and Cloud Computing', IEEE TII, 2018.
[63] Shang, B., et al., 'An Imputation Method for Missing Traffic Data Based on FCM Optimized by PSO-SVR', JAT, 2018.
[64] Reznik, L., et al., 'Signal change detection in sensor networks with artificial neural network structure', IEEE ICCIHSPS, 2005, 44–51.