
# ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

## ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

### ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

# Εφαρμογές IoT και Βελτιστοποίηση Διαχείρισης Πόρων στον τομέα της Ευφυούς Γεωργίας

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

**ΜΙΧΑΗΛΙΔΗΣ ΜΑΡΙΟΣ**

**Επιβλέπουσα:** Ιωάννα Ρουσσάκη
Επικ. Καθηγήτρια Ε.Μ.Π.

**Αθήνα 2020**


# ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

## ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

### ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

## Εφαρμογές IoT και Βελτιστοποίηση Διαχείρισης Πόρων στον τομέα της Ευφυούς Γεωργίας

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

**ΜΙΧΑΗΛΙΔΗΣ ΜΑΡΙΟΣ**

**Επιβλέπουσα:** Ιωάννα Ρουσσάκη

Επικ. Καθηγήτρια Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την

| | | |
|---|---|---|
| ..... | ..... | ..... |
| Ιωάννα Ρουσσάκη | Μιλτιάδης Αναγνώστου | Συμεών Παπαβασιλείου |
| Επικ.Καθηγήτρια Ε.Μ.Π. | Καθηγητής Ε.Μ.Π. | Καθηγητής Ε.Μ.Π. |

**Αθήνα 2020**

……..

ΜΙΧΑΗΛΙΔΗΣ ΜΑΡΙΟΣ
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών

# Περίληψη

Σκοπός της διπλωματικής αυτής είναι η ανάπτυξη δύο εφαρμογών που άπτονται στο Διαδίκτυο των Αντικειμένων με σκοπό την βελτιστοποίηση της διαχείρισης των πόρων στην γεωργία και την διευκόλυνση του τελικού χρήστη στην λήψη ορθότερων αποφάσεων, παρέχοντας του τις απαραίτητες πληροφορίες και δεδομένα. Ο στόχος είναι να βελτιωθούν τα εργαλεία που έχει στην διάθεση του ο άνθρωπος για την παρακολούθηση σε πραγματικό χρόνο αλλά και την επίλυση προβλημάτων στον τομέα της γεωργίας.

Πιο συγκεκριμένα, η διπλωματική βασίζεται σε δύο κεντρικούς άξονες. Ο πρώτος είναι η ανάπτυξη μια εφαρμογή επεξεργασίας και κατηγοριοποίησης εικόνων με την βοήθεια της μηχανικής μάθησης και των συνελικτικών νευρωνικών δικτύων για ασθένειες φύλλων, όπου κατατάσσονται σε 33 διαφορετικές κατηγορίες. Αυτό που εξετάζεται κατά βάση είναι η ρεαλιστικότερη εκπαίδευση του μοντέλου σε πραγματικές εικόνες, παρά το μικρό πλήθος αρχικών φωτογραφιών και η εκτέλεση της διαδικασίας πρόβλεψης μιας νέας εικόνας με την βοήθεια διαφόρων IoT συσκευών, όπως το Raspberry Pi. Ο δεύτερος άξονας της παρούσας εργασίας αναφέρεται στην δημιουργία και υλοποίηση ενός οικοσυστήματος Διαδικτύου των Αντικειμένων, όπου ο επικεφαλής διαχειριστής του συστήματος (Arduino) ελέγχει τους αισθητήρες που μας παρέχουν μετρήσεις σχετικά με την θερμοκρασία, την υγρασία αέρα και εδάφους καθώς και την ακτινοβολία του ήλιου, ώστε να επιτύχουμε την απομακρυσμένη παρακολούθηση και επεξεργασία των μετρικών σε πραγματικό χρόνο, προτείνοντας ταυτόχρονα μια καινοτόμα ιδέα, όπου για την βελτιστοποίηση της διαχείρισης του νερού κατά την άρδευση.

Το Διαδίκτυο των Αντικειμένων αποτελεί ένα καινοτόμο ερευνητικό πεδίο, στο οποίο συντελείται διαρκώς σημαντική τεχνολογική εξέλιξη και ενθαρρύνεται ο συνδυασμός του με την τεχνητή νοημοσύνη και την μηχανική μάθηση, ώστε να προκύπτουν ολοένα και αποδοτικότερα αποτελέσματα. Στην παρούσα διπλωματική παρουσιάζονται και οι δύο πλευρές του παραπάνω συνδυασμού, με στόχο την παροχή στον τελικό χρήστη πληροφοριών που ο άνθρωπος θα χρειαζόταν πολύ περισσότερο χρόνο να συλλέξει μόνος του, και την οπτικοποίηση αυτών, ώστε η παρακολούθηση σε πραγματικό χρόνο να αποτελεί πλέον πραγματικότητα και να μπορεί να συνδυαστεί με την μετέπειτα αναλυτικότερη επεξεργασία των δεδομένων.

**Λέξεις-Κλειδιά:** Διαδίκτυο των Αντικειμένων (IoT), Μηχανική Μάθηση, Επεξεργασία Εικόνας και Κατηγοριοποίηση, «Έξυπνη» Άρδευση, Διαχείριση Πόρων

# Abstract

The purpose of this diploma thesis constitutes the creation of two applications pertained to Internet of Things, in order to optimize the resource management in Agriculture and the facilitation of end user to make right decisions, while providing him with all the necessary information and data. The objective is the improvement of available tools, that users have for real-time monitoring and solving problems in Agriculture field.

In specific, this diploma thesis focuses on two main directions. The former is the development of an image processing and classification application for leaf disease classification into 33 distinct classes by deploying machine learning and convolutional neural networks. What we mainly focus here is the realistic training of the model from real images, despite the fact that we have a small dataset of images and the prediction of new images by running the inference process in IoT devices, like Raspberry Pi. The latter is the creation and implementation of an IoT ecosystem, where head administrator of the system (Arduino) controls the sensors that give us metrics such as temperature, humidity, soil moisture and UV, in order to achieve remote real-time monitoring and processing of the incoming data. Simultaneously, we propose an innovative idea for smart irrigation in order to optimize the management of water.

Internet of things constitutes a pioneering scientific field leading to constant technological breakthroughs and scientists are encouraged to combine IoT with Artificial Intelligence and Machine Learning, in order to get the finest results. In this thesis, we present both sides of the prior referred combination and our goal is to provide the end user with various data, which a single man will need ages to collect, and the visualization of these data, in order to enable real-time monitoring to be combined with afterwards thorough processing of the collected information.

**Keywords:** Internet of Things (IoT), Machine Learning, Image Processing and Classification, Smart Irrigation, Resource Management

# Ευχαριστίες

Θα ήθελα να ευχαριστήσω ιδιαίτερα την Επικ.Καθηγήτρια κ. Ιωάννα Ρουσσάκη που μου ανέθεσε την παρούσα διπλωματική εργασία δίνοντας μου παράλληλα την ευκαιρία να ασχοληθώ με ενδιαφέροντες τομείς της σύγχρονος τεχνολογικής εξέλιξης όπως το Διαδίκτυο των Αντικειμένων και τη μηχανική μάθηση. Επιπρόσθετα, με ενέπνευσε να επιλέξω τις μεταπτυχιακές μου σπουδές με βάση την μηχανική μάθηση ώστε να ασχοληθώ επαγγελματικά και επιστημονικά στο μέλλον καθώς και την πεποίθηση να προσπαθώ να εξελίσσομαι συνεχώς και να διευρύνω τις γνώσεις μου. Εν συνεχεία, θα ήθελα να ευχαριστήσω τον Καθηγητή κ. Συμεών Παπαβασιλείου και τον Καθηγητή κ. Μιλτιάδη Αναγνώστου για την τιμή που μου έκαναν να συμμετάσχουν στην τριμελή εξεταστική επιτροπή της εργασίας.

Επίσης, θα ήθελα να ευχαριστήσω ιδιαιτέρως τον ερευνητή κ. Γεώργιο Ρούτη που ήταν αρωγός καθ' όλη την διάρκεια διεξαγωγής της συγκεκριμένης διπλωματικής με τις πολύτιμες συμβουλές και υποδείξεις του.

Αφιερώνω την διπλωματική εργασία στους γονείς μου, Μιχαηλίδη Παναγιώτη και Αλεξάνδρα Γαβραλίδου, για την υποστήριξη και την αγάπη τους όλα τα χρόνια. Τέλος, θα ήθελα να ευχαριστήσω τους φίλους και συμφοιτητές μου για την συνεχή υποστήριξη και όλα αυτά τα υπέροχα φοιτητικά χρόνια.

# Πίνακας περιεχομένων

# ΚΕΦΑΛΑΙΟ 1 – Εισαγωγή

## 1.1 Διαδίκτυο των Αντικειμένων (IoT)

Για το Διαδίκτυο των Αντικειμένων (Internet of Things) [1] έχουν δοθεί πολλοί και ποικίλοι ορισμοί, ο βασικότερος εκ των οποίων είναι ότι αποτελεί το δίκτυο επικοινωνίας πληθώρας συσκευών, οικιακών συσκευών, αυτοκινήτων καθώς και κάθε «αντικειμένου» που ενσωματώνει ηλεκτρονικά μέσα, λογισμικό, αισθητήρες και συνδεσιμότητα σε δίκτυο ώστε να επιτρέπεται η σύνδεση και ανταλλαγή δεδομένων. Ως τεχνολογικός κλάδος εξελίσσεται ραγδαία τα τελευταία χρόνια και αναμένεται να εισέρχεται στην ζωή των ανθρώπων όλο και περισσότερο μέσα από πληθώρα εφαρμογών, από εφαρμογές εξυπηρέτησης πελατών μέχρι εφαρμογές λήψης και επεξεργασίας δεδομένων από αισθητήρες [2]. Ο βασικός στόχος του IoT είναι να διευκολύνει την καθημερινότητα των ανθρώπων με την χρήση χαμηλού κόστους συσκευών και με την κατανάλωση των βέλτιστων πόρων, κάτι απόλυτα απαραίτητο αν αναλογιστούμε το μέγεθος του παγκόσμιου δικτύου από συνδεδεμένες συσκευές που αυξάνεται με ταχύτατο ρυθμό. Αρκεί να αναλογιστούμε ότι τα συνδεδεμένα «αντικείμενα» έχουν ξεπεράσει κατά πολύ τον αριθμό των κατοίκων της Γης και για να αντιληφθούμε ακόμη καλύτερα τον όγκο των συσκευών υπολογίζεται ότι μέχρι το 2025 θα χρησιμοποιούνται περίπου 41.6 δισεκατομμύρια IoT συνδεδεμένες συσκευές. Ακόμη, αποτελεί ένα πολύ προσοδοφόρο δίκτυο επικοινωνίας για πάρα πολλές εταιρίες, καθώς με την χρήση IoT συσκευών μειώνεται αισθητά το κόστος κατανομής, η παρακολούθηση των υλικών και πρώτων υλών και γενικότερα διευκολύνεται η γραμμή παραγωγής των εκάστοτε προϊόντων. Οι απλούστερες συσκευές θα μπορούσαμε να πούμε ότι είναι POS συσκευές, smartwatches, IP κάμερες μέχρι συστήματα από συνδεδεμένα αντικείμενα για την Γεωργία, για την αυτοκινητοβιομηχανία, για την παροχή ιατρικών υπηρεσιών και πολλών άλλων, που όλα αυτά συγκλίνουν στην εξοικονόμηση πόρων και την σαφή βελτίωση του βιοτικού επιπέδου ζωής.

## 1.2 Διαδίκτυο των Αντικειμένων στην Γεωργία

Για τον σκοπό αυτή της διπλωματικής εργασίας, χρησιμοποιήθηκαν ορισμένες IoT συσκευές όπως Arduino, Raspberry Pi, Jetson Nano, Google Coral, XBee και αισθητήρες υγρασίας (αέρα και εδάφους), θερμοκρασίας και ακτινοβολίας UV. για τις οποίες θα γίνει αναλυτικότερη αναφορά παρακάτω, ώστε να γίνει η υλοποίηση και των 2 εφαρμογών που θα παρουσιαστούν τόσο για την επεξεργασία εικόνας και ταξινόμηση με βάση τις ασθένειες των φύλλων (image processing and classification) όσο και του συστήματος αισθητήρων που κατασκευάσαμε για την παρακολούθηση διάφορων περιβαλλοντικών μετρικών για την διαδικασία της άρδευσης.

Ο τομεάς της γεωργίας φυσικά είναι άρρηκτα συνδεδεμένος με IoT συσκευές καθότι χρησιμοποιούνται ρομποτικά και μηχανικά μέρη  για την αυτοματοποίηση

διαφόρων και συνεχώς εξελισσόμενων και προηγμένων εργασιών με απώτερο στόχο την αύξηση της παραγωγής [3]. Επιπρόσθετα, η γεωργία είναι μια από τις κύριες βιομηχανίες που ενσωματώνουν την χρήση των drones, όπου και εμείς ελαφρώς ενσωματώσαμε για την λήψη εικόνων μέσω την κάμερας του drone. Γενικότερα, τα drones χρησιμοποιούνται τόσο για την λήψη εικόνων όσο και για την χαρτογράφηση και παρακολούθηση των αγροκτημάτων. Ένα ακόμη κομμάτι που συνδέει την γεωργία (agriculture) με τις IoT συσκευές και αυτό που θα αναλυθεί στα πλαίσια αυτής της διπλωματικής εργασίας είναι το remote sensing, δηλαδή η απομακρυσμένη λήψη μετρήσεων από διάφορους αισθητήρες που μεταδίδονται ασύρματα σε έναν υπολογιστή ή μια IoT συσκευή όπως το Raspberry Pi για περαιτέρω ανάλυση. Συνεπώς, δίνεται η δυνατότητα στον αγρότη να παρακολουθεί τις καλλιέργειες με πολύ αποτελεσματικότερο τρόπο και να έχει μια γενικότερη εικόνα και γνώση του τι συμβαίνει ώστε να παίρνει και καταλληλότερες αποφάσεις αλλά και φυσικά να βελτιστοποιεί την κατανάλωση των πόρων. Τέλος, η γεωργία συνδέεται με τις IoT συσκευές μέσω του Computer Imaging, ουσιαστικά δηλαδή την λήψη φωτογραφιών μέσω drones ή και την προσομοίωση εικόνων σε συνδυασμό με την κατάλληλη ψηφιακή επεξεργασία εικόνας, όπου βασίζεται η έτερη εφαρμογή για ασθένειες φύλλων που θα παρουσιαστεί. Η ψηφιακή επεξεργασία εικόνας υλοποιεί την βασική έννοια της επεξεργασίας μιας εισερχόμενης εικόνας με την χρήση αλγορίθμων, μηχανικής μάθησης και νευρωνικών δικτύων.

## 1.3 Οργάνωση Κειμένου

Στο 2ο κεφάλαιο θα παρουσιαστεί το σχετικό υπόβαθρο που απαιτείται για την υλοποίηση της συγκεκριμένης διπλωματικής. Πιο συγκεκριμένα, θα αναλυθεί η ιδέα του image processing και classification και πως αυτά υλοποιούνται με νευρωνικά δίκτυα και μάλιστα στην συγκεκριμένη περίπτωση με CNN (Convolutional Neural Networks), ώστε να επιτυγχάνουμε τα καλύτερα δυνατά αποτελέσματα στην κατηγοριοποίηση των ασθενειών των φύλλων αλλά και την ορθότερη πρόβλεψη με βάση νέες φωτογραφίες. Ακολούθως, θα γίνει μια αναλυτικότερη εισαγωγή στο Arduino και πως αυτό αποτελεί τον εγκέφαλο του συστήματος αισθητήρων, που κατασκευάσαμε με στόχο την αποτελεσματικότερη άρδευση, ώστε να μας παρέχει ασύρματα τις μετρήσεις προς επεξεργασία και παρουσίαση. Στο τέλος του 2ου κεφαλαίου, παρουσιάζονται οι λοιπές IoT συσκευές και πως αυτές ενεπλάκησαν στις δύο εφαρμογές μας. Στο 3ο κεφάλαιο υπάρχει μια σύντομη παρουσίαση της πιο σύγχρονης σχετικής βιβλιογραφίας. Στο 4ο κεφάλαιο αναφερόμαστε στο κίνητρο και το βασικό πρόβλημα που μας οδήγησε στην εκπόνηση της συγκεκριμένης εργασίας. Στο 5ο κεφάλαιο παρουσιάζεται η «επίλυσης» του προβλήματος, όπου θα υπάρξει ο κώδικας και η αναλυτική υλοποίηση και των δυο εφαρμογών. Στο 6ο κεφάλαιο ακολουθεί η παρουσίαση σχετικών διαγραμμάτων και πινάκων, ώστε να υπάρξει ένα εποπτικό μέσο για την αξιολόγηση των εφαρμογών και των συσκευών που χρησιμοποιήθηκαν. Κλείνοντας στο 7ο κεφάλαιο διατυπώνονται συμπεράσματα, σκέψεις και προτάσεις για μελλοντική εργασία και έρευνα στο συγκεκριμένο αντικείμενο.

# Κεφάλαιο 2 – Σχετικό Υπόβαθρο

## 2.1 Επεξεργασία Εικόνας με Convolutional Neural Networks (CNNs)

Με την ολοένα και μεγαλύτερη αύξηση των μοντέρνων frameworks η μηχανική και η βαθιά μάθηση (machine and deep learning) με την χρήση Convolutional Neural Networks (CNNs) βρίσκει ολοένα και μεγαλύτερη επιτυχία σε εργασίες που σχετίζονται με την εικονική αναγνώριση [4], [5]. Τα CNN αποτελούν μια από τις πιο επιτυχημένες υλοποιήσεις για την κατηγοριοποίηση/ταξινόμηση εικόνων, ειδικότερα όταν συνδυάζονται με τεχνικές βαθιάς μάθησης όπως τις συναρτήσεις ενεργοποίησης ReLU, τα dropout επίπεδα, και την στοχευμένη αύξηση των δεδομένων (data augmentation) . Προφανώς, γίνεται αντιληπτό ότι όσο αναλυτικότερο και βαθύτερο είναι ένα τέτοιο δίκτυο τόσο περισσότερη υπολογιστική ισχύ απαιτεί και η διαδικασία της εκμάθησης γίνεται εκθετικά δυσκολότερη, ωστόσο όπως θα δούμε στο 5ο κεφάλαιο με τις κατάλληλες τροποποιήσεις στην υλοποίηση μπορούμε να βελτιστοποιήσουμε τους πόρους που χρησιμοποιούμε καθότι είναι απαραίτητο αυτό οποτεδήποτε το συνδυάζουμε με IoT εφαρμογές, διότι οι συσκευές μας δεν θέλουμε να είναι πανίσχυροι υπολογιστές, αλλά μικρές και οικονομικές συσκευές που θα μας δίνουν μεγάλη ευελιξία, καθώς λόγο του μικρού κόστους και κατανάλωσης τους μπορούν να τρέχουν αρκετές ταυτόχρονα και απομακρυσμένα.

### 2.1.1 Η πράξη της συνέλιξης

Ας δούμε την θεωρία λίγο πίσω από τα CNN ξεκινώντας από την συνέλιξη (convolution). Η συνέλιξη είναι μια πράξη που ορίζεται μεταξύ δύο συναρτήσεων και η τιμή της δείχνει τον βαθμό ομοιότητας μεταξύ των δύο αυτών συναρτήσεων ας πούμε f και g. Η συνέλιξη φυσικά ορίζεται και σε διακριτές τιμές και ορίζεται ως εξής:

$$\text{f}[n]\ (0 \le n \le N-1)\ \textit{και}\ g[j]\ (0 \le m \le M-1)$$

$$(f * g)[n] = \sum_{m=0}^{M-1} f[n+m]g[m] \quad (\textit{εξίσωση}\ 1)$$

Η πράξη της συνέλιξης μπορεί φυσικά μπορεί να επεκταθεί και στις 2 διαστάσεις αλλά και σε περισσότερες, ας δούμε όμως για τις 2 διαστάσεις που είναι και αυτό που μας απασχολεί στην επεξεργασία εικόνας.

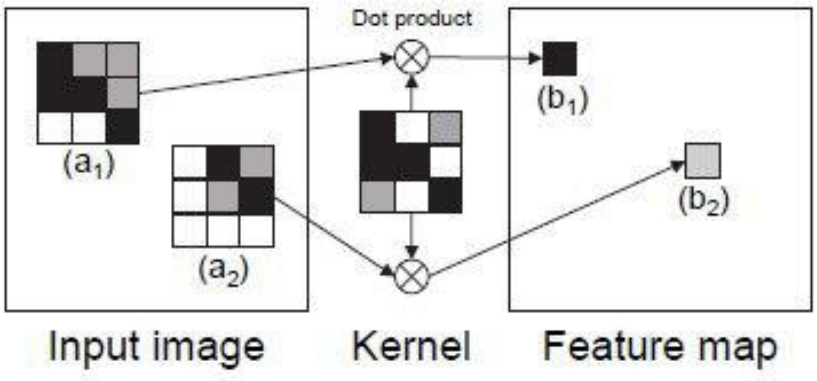
Η 2D- συνέλιξη μεταξύ μιας συνάρτησης φίλτρου (filter ή kernel, όπως αναφέρεται στην εικόνα 1) και μιας συνάρτησης δεδομένων ώστε να δούμε την ομοιότητα μεταξύ τους ορίζεται ως εξής:

Συνάρτηση φίλτρου: $\text{F}[r][s]\ (0 \le r \le R-1,\ 0 \le s \le S-1)$

Συνάρτηση δεδομένων: $\text{D[h][w]}\ \ (0 \le h \le H-1,\ 0 \le w \le W-1)$

2D – συνέλιξη:

(D * F)[h][w] = $\sum_{r=0}^{R-1} \sum_{s=0}^{S-1} D[h+r][w+s]\, F[r][s]$ ( $\varepsilon\xi\acute{\iota}\sigma\omega\sigma\eta$ 2)



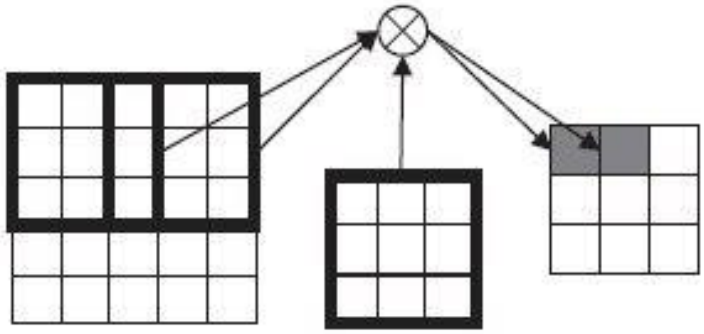

*Εικόνα 1: 2D – συνέλιξη (Πηγή: [4])*

Αυτή η δισδιάστατη συνέλιξη χρησιμοποιείται κατά κόρον σε εφαρμογές όπου απαιτείται επεξεργασία εικόνας και ονομάζεται δευτερευόντως image convolution, δηλαδή συνέλιξη εικόνας. Μια εικόνα ουσιαστικά αποτελεί την συνάρτηση δεδομένων D[h][w] και αντίστοιχα το φίλτρο (kernel) αποτελεί την συνάρτηση F[r][s]. Το αποτέλεσμα της της 2D – συνέλιξης δημιουργεί έναν πίνακα χαρακτηριστικών (feature map), ο οποίος ουσιαστικά είναι της που «μαθαίνει» σταδιακά στο πρόγραμμα μέσω του training τα βασικά χαρακτηριστικά κάθε εικόνας, ώστε να μπορέσει να μάθει όσο καλύτερα γίνεται ένα set εικόνων, ώστε στην συνέχεια να μπορεί το νευρωνικό δίκτυο να της δίνει σε ποια κατηγορία εικόνων από αυτές που έχει μάθει ταιριάζει περισσότερο μια καινούρια εικόνα που του δίνουμε ως είσοδο. Το μέγεθος των φίλτρων ( R × S) προφανώς είναι πολύ μικρότερα από το μέγεθος της εικόνας (H × W). Της, βλέπουμε και στην εικόνα 1, για δεδομένα μέρη της αρχικής εικόνας που δέχεται το νευρωνικό $a_1$ και $a_2$ όπου έχουν το ίδιο μέγεθος με την συνάρτηση φίλτρου πολλάσιάζονται point-wisely με το φίλτρο (kernel). Συνεπώς, της ο «χάρτης» χαρακτηριστικών αποτελείται από όλα τα αποτελέσματα (π.χ. $b_1$ και $b_2$) όλων αυτών των πολλαπλασιασμών. Υψηλή τιμή του αποτελέσματος της συνέλιξης σημαίνει ότι η αντίστοιχα επιλεγμένη περιοχή της εικόνας έχει υψηλό βαθμό ομοιότητας με το φίλτρα [4].

## 2.1.2 Convolutional Layer και Pooling Layer

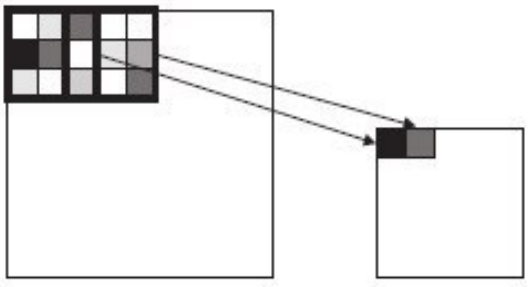
Της μπορεί να γίνει αντιληπτό, η πράξη της συνέλιξης είναι πολύ κοστοβόρα, ειδικότερα όσο αυξάνεται το μέγεθος μιας εικόνας, καθώς απαιτούνται πολλοί υπολογισμοί. Συνεπώς, θυσιάζοντας εν μέρει λίγη ακρίβεια στο κατά πόσο μαθαίνει το νευρωνικό επακριβώς τα χαρακτηριστικά μιας εικόνας, αλλά κερδίζοντας ταυτόχρονα πολύ περισσότερο σε υπολογιστικό χρόνο χρησιμοποιούμε δύο πολύ κλασσικές τεχνικές για να μειώσουμε τον τεράστιο όγκο πράξεων χωρίς να χάνουμε αρκετά από το αποτέλεσμα. Η πρώτη τεχνική που χρησιμοποιείται για αυτό τον σκοπό είναι η εφαρμογή της 2D-συνέλιξης στην εικόνα εισόδου με κάποιο βήμα (stride της θα αναφερθεί παρακάτω στον κώδικα και στην εικόνα 2). Μια συνέλιξη με βήμα

πετυχαίνει αυτό που ονομάζεται down-sampling, δηλαδή μείωση του δείγματος της εικόνας καθώς λαμβάνει βήματα κάθε s pixels σε κάθε διεύθυνση της εικόνας, όπου s είναι προφανώς η τιμή του βήματος. Με το down-sampling αυτό που επιτυγχάνεται είναι η μείωση των παραμέτρων εισόδου της εικόνας χωρίς να χάνεται ωστόσο ουσιαστική πληροφορία. Εκεί που καταλήγουμε μέσω της της διαδικασίας λοιπόν είναι η συμπύκνωση της πληροφορίας στο feature map που προκύπτει ώστε να είναι μικρότερου μεγέθους από την αρχική εικόνα.



*Εικόνα 2: Συνέλιξη με βήμα s = 2 (Πηγή: [4])*

Το πρώτο επίπεδο του CNN συνεπώς της γίνεται αντιληπτό είναι αυτό το επίπεδο που υλοποιείται η πράξη της συνέλιξης για να βγουν τα χαρακτηριστικά μιας εικόνας με βάση κάποιο φίλτρο ώστε να τα μάθει το νευρωνικό δίκτυο κατά την διαδικασία του training. Το αμέσως επόμενο επίπεδο υλοποιεί την διαδικασία του pooling. Το pooling αποτελεί της μια διαδικασία down-sampling, ώστε να μειωθεί ο υπολογιστικός χρόνος και συνάμα να εξάγουμε τα χρήσιμα δεδομένα που μαθαίνει το νευρωνικό. Σε αυτό το επίπεδο, αυτό που ουσιαστικά γίνεται είναι ότι θεωρούμε ένα pixel ως το αντιπροσωπευτικό για μια μικρή περιοχή από pixels ώστε να μικρύνουμε το μέγεθος της εισόδου. Για παράδειγμα, η εικόνα 3 δείχνει την διαδικασία max pooling με βήμα s = 2. Δηλαδή, διαιρείται η εικόνα σε περιοχές 3x3 με βήμα 2 (σε κάθε διεύθυνση) και για κάθε περιοχή 3x3 επιλέγεται το αντιπροσωπευτικό pixel με την μέγιστη τιμή (max pooling).



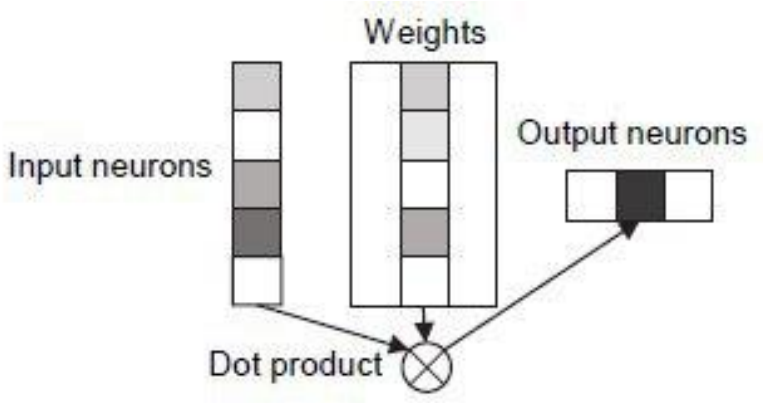
*Εικόνα 3: 3x3 max pooling με βήμα s = 2 (Πηγή: [4])*

Ως εδώ λοιπόν, έχουμε ότι τα CNN χρησιμοποιούν φίλτρα συνέλιξης (συνήθως με κάποιο βήμα s) για να εξάγουν χαρακτηριστικά από την εικόνα εισόδου. Αυτά τα

επίπεδα ονομάζονται convolutional layers και ενδιάμεσα παρεμβάλλονται pooling επίπεδα ώστε να γίνεται down-sampling και να μειώνεται ο υπολογιστικός όγκος εξάγοντας ταυτόχρονα χρήσιμα χαρακτηριστικά για την εικόνα. Συνεπώς, ένα convolutional layer που περιέχει N φίλτρα εξάγει πολλαπλά χαρακτηριστικά από την εικόνα εισόδου, καταλήγοντας σε N feature maps. Το στάδιο της εκμάθησης του CNN του μαθαίνει ουσιαστικά ένα φίλτρο για κάθε ένα από τα χαρακτηριστικά, ώστε να μπορεί να το αναγνωρίσει μετέπειτα και σε άλλες εικόνες [4].

### 2.1.3 Fully Connected Layer

Αφού λάβουμε τα αποτελέσματα των παραπάνω επιπέδων (convolutional layers και pooling layers) ακολουθεί το πλήρως συνδεδεμένο επίπεδο, το οποίο συνδυάζει τα αποτελέσματα των παραπάνω επιπέδων. Όπως φαίνεται από την εικόνα 4, αποτελείται από τους νευρώνες εισόδου (input neurons), τους νευρώνες εξόδου (output neurons) και βάρη (weights), τα οποία απεικονίζουν την σχέση μεταξύ εισόδου και εξόδου. Πραγματοποιεί τον πολλαπλασιασμό μεταξύ του πίνακα των βαρών και του διανύσματος της εισόδου παράγοντας έτσι την έξοδο. Ο σκοπός των fully connected layers είναι να κατατάξουν την αρχική εικόνα με βάση κάποιο label (ετικέτα) ή κλάση. Για τον υπολογισμό των weights το πλήρως συνδεδεμένο επίπεδο του CNN δικτύου περνάει μέσα από την δική του διαδικασία backpropagation ώστε να καθορίσει τα πιο ακριβή βάρη, κάτι το οποίο δεν θα αναλυθεί καθώς ξεφεύγει από τον σκοπό της παρούσας διπλωματικής, καθώς είναι κάτι που υλοποιεί το CNN από μόνο του. Κάθε νευρώνας επομένως, λαμβάνει βάρη τα οποία βάζουν σε σειρά προτεραιότητας την πιο κατάλληλη κλάση ή ετικέτα (label) όπου κατατάσσεται η εικόνα εισόδου.




*Εικόνα 4: Υπολογισμός στο πλήρως συνδεδεμένο επίπεδο (Πηγή: [4])*

Συνοψίζοντας λοιπόν ως εδώ, τα convolutional layers λειτουργούν ως feature extractors ενώ τα fully connected layers ως classifiers. Τα convolutional layers εξάγουν τα feature maps που είναι τα χαρακτηριστικά ανά κάποια περιοχή της εικόνας από την εικόνα εισόδου [6]. Κάθε τέτοιο επίπεδο δημιουργεί έναν 3D – τένσορα (ανάλογα και το βήμα) και τον τροφοδοτεί στο επόμενο επίπεδο. Κατόπιν, μετά το πέρας αυτών των επιπέδων τα πλήρως συνδεδεμένα επίπεδα παίρνουν τους τελευταίους πίνακες χαρακτηριστικών

και τους κάνουν flatten, δηλαδή τους μετατρέπουν σε μονοδιάστατα διανύσματα και δημιουργούν ένα διάνυσμα εξόδου με L τιμές όπου L είναι ο αριθμός των labels στα οποία θέλουμε να ταξινομήσουμε και να κατατάξουμε τις φωτογραφίες. Στο τέλος, εφαρμόζουμε μια μέθοδο κανονικοποίησης με το επίπεδο softmax (δεδομένου ότι L > 2) και έτσι κάθε διάσταση στο κανονικοποιημένο πλέον διάνυσμα της εξόδου εκφράζει την πιθανότητα η αρχική εικόνα να ανήκει στην αντίστοιχη κλάση εικόνας (label).

### 2.1.4 Οι συναρτήσεις ReLU και softmax

Εν συντομία, χρήσιμο αναφοράς είναι η συνάρτηση ReLU, η οποία είναι η συνάρτηση ενεργοποίησης του εκάστοτε convolutional layer, η οποία ορίζεται ως y = max(0, x). Ουσιαστικά για να χρησιμοποιηθεί η τεχνική SGD (stochastic gradient descent) μέσω του backpropagation και να γίνουν train τα νευρωνικά δίκτυα, απαιτείται μια συνάρτηση ενεργοποίησης, η οποία να επιτρέπει να μαθαίνονται οι διάφορες σχέσεις μεταξύ των δεδομένων [7].
Από την άλλη, η συνάρτηση softmax λαμβάνει ως είσοδο ένα διάνυσμα με K αριθμούς και το κανονικοποιεί σε μια πιθανοτική κατανομή που αποτελείται από K πιθανότητες ανάλογες των εκθετικών των αρχικών αριθμών. Δηλαδή, μετά την εφαρμογή της softmax όλες οι τιμές βρίσκοντα στο πεδίο (0,1) και το άθροισμα όλων των τιμών είναι 1 και συνεπώς μπορούν να ερμηνευθούν ως πιθανότητες. Συνεπώς, μεγαλύτεροι είσοδο αντιστοιχούν σε μεγαλύτερες πιθανότητες. Η βασική συνάρτηση softmax που χρησιμοποιείται ορίζεται ως εξής:

$$\sigma : \mathbb{R}^K \to \mathbb{R}^K$$

$$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \text{ for } i = 1, \ldots, K \text{ and } \mathbf{z} = (z_1, \ldots, z_K) \in \mathbb{R}^K$$

### 2.1.5 Η εκπαίδευση ενός CNN

### 2.1.5.1 Batch processing

Η εκμάθηση τώρα του CNN αποτελεί μια επιτηρούμενη/εποπτευόμενη διαδικασία μάθησης χρησιμοποιώντας ένα σετ εικόνων για training και τις αντίστοιχες σωστές ετικέτες (labels) στις οποίες ανήκουν οι εικόνες. Αυτές οι εικόνες εισόδου φορτώνονται στο CNN ανά batch, δηλαδή ανά ομάδα εικόνων (π.χ. ανά 8), κάτι το οποίο αποτελεί τον κλασσικό τρόπο φόρτωσης εικόνων σε ένα CNN, ωστόσο όπως θα αναλυθεί στο 5ο κεφάλαιο υπάρχει τρόπος βελτιστοποίησης αυτής της διαδικασίας χωρίς να εξαντλούμε την μνήμη καθώς απαιτούνται πολλές εικόνες να επεξεργαστούν. Το CNN επεξεργάζεται ένα batch την φορά επειδή η τεχνική του SGD (stochastic gradient descent) [8] που τυπικά χρησιμοποιείται σε τέτοιες εφαρμογές μπορεί εύκολα να παραλληλοποιηθεί μεταξύ των εικόνων του ίδιου batch.

### 2.1.5.2 Forward Computation και Backward Computation

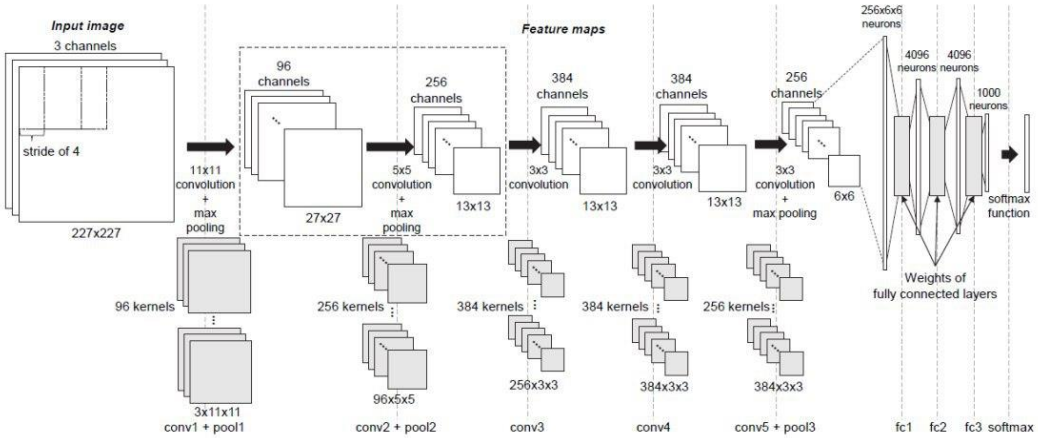
Αναλύοντας λίγο παραπάνω την διαδικασία του training γίνεται εύκολα αντιληπτό ότι αποτελεί την διαδικασία που απαιτεί τον περισσότερο υπολογιστικό χρόνο, συνεπώς για αυτό πρέπει να έχουμε την καλύτερη δυνατή υλοποίηση του CNN ώστε να βελτιστοποιηθεί όσο γίνεται ο χρόνος. Εξού, όπως θα δείξω και στο 5$^{ο}$ κεφάλαιο, δημιούργησα μόνος μου τα επίπεδα του CNN που χρησιμοποίησα ώστε να είναι βέλτιστα μετά από δοκιμές για τον όγκο των δεδομένων του προβλήματος. Η εκμάθηση του CNN χωρίζεται σε 2 στάδια: το forward computation και το backward computation. Για μια δεδομένη εικόνα εκμάθησης κατά την είσοδο το στάδιο forward computation περνάει μέσα από τα διάφορα επίπεδα του CNN από το πρώτο ως το τελευταίο και επιστρέφει την έξοδο από το τελευταίο επίπεδο. Δηλαδή, καθορίζει την κλάση στην οποία ανήκει η training εικόνα. Μετά τον υπολογισμό της διαφοράς μεταξύ της εξόδου και της σωστής κλάσης (ετικέτας) ξεκινά το στάδιο του backward computation, το οποίο λαμβάνει τις απαραίτητες τιμές/παραμέτρους που πρέπει να προστεθούν στις παραμέτρους του δικτύου (π.χ. βάρη) του τελευταίου πλήρως συνδεδεμένου επιπέδου υπολογίζοντας τα gradients. Ύστερα από την ενημέρωση αυτών των παραμέτρων στο τελευταίο layer, αυτές οι τιμές μεταφέρονται προς τα προηγούμενα layers (backpropagation) ώστε να προσαρμοστούν ή να διορθωθούν οι παράμετροι στο στάδιο backward computation. Οι κλίσεις (gradients) υπολογίζονται με την λογική της ελαχιστοποίησης (minimizing) της διαφοράς μεταξύ της εξόδου κάθε επιπέδου και της σωστής απάντησης κάθε επιπέδου (που είναι γνωστή κατά το training) [9].

## 2.1.6 Σύνοψη

Συνοψίζοντας σε σημεία την ανάλυση περί image processing και classification έχουμε:

- Αρχικά έχουμε μια εικόνα εισόδου
- Εφαρμογή διαφόρων φίλτρων για την δημιουργία feature maps
- Εφαρμογή της συνάρτησης ReLU για να ενεργοποιήσουμε τα convolutional layers
- Εφαρμογή pooling layer σε κάθε feature map ή channel για την λήψη του πιο αντιπροσωπευτικού pixel από μια περιοχή εικόνας
- Flatten των εικόνων μετά το τελευταίο pooling σε ένα μονοδιάστατο διάνυσμα
- Είσοδος του διανύσματος αυτού σε ένα πλήρες συνδεδεμένο επίπεδο του νευρωνικού δικτύου
- Επεξεργασία των χαρακτηριστικών μέσα από το δίκτυο, όπου το τελευταίο fully connected layer και με την εφαρμογή κανονικοποίησης παρέχει την πιθανοτική κατανομή για τις κλάσεις που έχουμε
- Εκπαίδευση (training) μέσω του forward propagation και του back propagation για πολλές εποχές, έως ότου έχουμε ένα καλώς ορισμένο νευρωνικό δίκτυο με

καλά εκπαιδευμένα βάρη και feature detectors («αναγνωριστές» χαρακτηριστικών)




*Εικόνα 5: Η αναλυτική οργάνωση ενός γνωστού νευρωνικού δικτύου (AlexNet) (Πηγή: [4])*

## 2.2 IoT συσκευές και αισθητήρες

### 2.2.1 Arduino

Το Arduino χρησιμοποιήθηκε αποκλειστικά στην εφαρμογή μας με το IoT οικοσύστημα ( Arduino + Xbee +αισθητήρες + raspberry pi 4), όπου και αποτέλεσε το «μυαλό» του όλου συστήματος με την συλλογή, την επεξεργασία και την αποστολή των δεδομένων. Ας κάνουμε αρχικά λοιπόν μια εισαγωγή στο τι είναι το Arduino και γιατί το επιλέξαμε για την υλοποίηση μιας εφαρμογής του Internet of Things, όπου οι χρησιμοποιούμενοι πόροι πρέπει να είναι όσο το δυνατόν περισσότερο βελτιστοποιημένοι δίνοντας μας ταυτόχρονα τόσο απαιτούμενη ευελιξία όσο και την απαραίτητη σταθερότητα κατά την υλοποίηση και την «εκτέλεση» της εφαρμογής παράγοντας αξιόπιστα αποτελέσματα.

#### 2.2.1.1 Τι είναι το Arduino

Το Arduino είναι μια ηλεκτρονική πλατφόρμα (electronics platform) ανοιχτού λογισμικού που βασίζεται σε εύκολο στην χρήση υλικό (hardware) και λογισμικό (software). Οι πλακέτες (boards) του Arduino είναι ικανές να διαβάσουν ποικιλοτρόπως εισόδους, όπως για παράδειγμα φως σε έναν αισθητήρα, έναν διακόπτη, ακόμη κι ένα μήνυμα από το Twitter κλπ., και να τις μετατρέψουν σε εξόδους διαφόρων μορφών, όπως την ενεργοποίηση ενός κινητήρα, το άναμμα ενός LED, την online δημοσίευση ενός κειμένου κλπ.. Ο χρήστης ορίζοντας ένα σύνολο από οδηγίες που στέλνει / «φορτώνει» στον μικροεπεξεργαστή της πλακέτας κάθε φορά μπορεί να καθορίσει τον χειρισμό της πληροφορίας και το τι ακριβώς θα κάνει το Arduino. Για να γίνει φυσικά αυτό απαιτείται η γλώσσα προγραμματισμού Arduino, η οποία

βασίζεται στο Wiring, το οποίο είναι ένα προγραμματιστικό framework ανοιχτού λογισμικού για μικροεπεξεργαστές, και το λογισμικό του Arduino (IDE), το οποίο βασίζεται στο Processing που είναι ένα ευέλικτο μπλοκ προγραμματισμού (software sketchbook). Λόγω της ποικιλίας σε δυνατότητες που μας δίνει για την είσοδο και την έξοδο το Arduino αποτελεί τον εγκέφαλο αναρίθμητων projects, από καθημερινές απλές εφαρμογές μέχρις σύνθετα επιστημονικά πειράματα. Το γεγονός ότι αποτελεί πλατφόρμα ανοιχτού λογισμικού είναι κάτι που μας ενθάρρυνε ιδιαίτερα στην επιλογή του, διότι μπορεί να χρησιμοποιηθεί από τον πιο αρχάριο μέχρι τον πιο εξειδικευμένο, καθότι η συλλογή της προσβάσιμης γνώσης γύρω από την συγκεκριμένη πλατφόρμα ανοιχτού λογισμικού είναι τεράστια για κάθε επίπεδο εφαρμογών.

Το Arduino «γεννήθηκε» στο Ivrea Interaction Design Institute ως ένα εύκολο εργαλείο για γρήγορη προτυποποίηση, με στόχο την χρήση από σπουδαστές χωρίς υπόβαθρο στα ηλεκτρονικά και τον προγραμματισμό. Γρήγορα όμως απέκτησε πολύ ευρύ κοινό και άρχισε να προσαρμόζεται στις νέες ανάγκες και προκλήσεις, διαφοροποιώντας την προσφορά από έναν απλό 8-bit processor σε ένα προϊόν για IoT εφαρμογές, wearables, 3D printers, και ενσωματωμένα συστήματα [10].

### 2.2.1.2 Γιατί Arduino

Χάρη στην απλή και εύκολα προσβάσιμη εμπειρία χρήστη, χρησιμοποιείται σε διάφορα projects και εφαρμογές. Το λογισμικό του είναι εύκολο στην χρήση για κάποιον αρχάριο, ενώ παράλληλα δίνει αρκετή ευελιξία στον έμπειρο χρήστη. Αυτό επιτυγχάνεται διότι έχει μαζέψει την «ακαταστασία» που επικρατεί σε έναν μικροεπεξεργαστή και την έχει «τυλίξει» σε ένα εύκολο πακέτο προς χρήση απλοποιώντας σε μεγάλο βαθμό την εργασία με μικροεπεξεργαστές. Συνεπώς, τα πλεονεκτήματα του που μας οδήγησαν να το επιλέξουμε χωρίς δεύτερη σκέψη είναι τα εξής:

- **Οικονομικό**

  Οι πλακέτες του Arduino είναι σχετικά φθηνότερες σε σύγκριση με τους αντίστοιχους μικροεπεξεργαστές της αγοράς. Οι φθηνότερες εκδόσεις απαιτούν συναρμολόγηση με το χέρι, αλλά ακόμη και οι προ-συναρμολογημένες εκδόσεις κοστίζουν λιγότερο από 50 ευρώ (€).

- **Cross – platform**

  Το λογισμικό του Arduino (IDE) τρέχει σε Windows, Macintosh OSX και Linux, που αποτελούν το 99% όλων των λειτουργικών συστημάτων, εν αντιθέσει με τα περισσότερα αντίστοιχα συστήματα μικροεπεξεργαστών που είναι περιορισμένα στα Windows.

- **Απλό και καθαρό προγραμματιστικό περιβάλλον**

  Το Arduino IDE είναι πολύ εύκολο στην χρήση απαιτώντας πολύ λίγο χρόνο για την εξοικείωση με αυτό, δίνοντας ταυτόχρονα την απαραίτητη ελευθερία και ευελιξία στον χρήστη.

- **Open-source και επεκτάσιμο λογισμικό (software)**

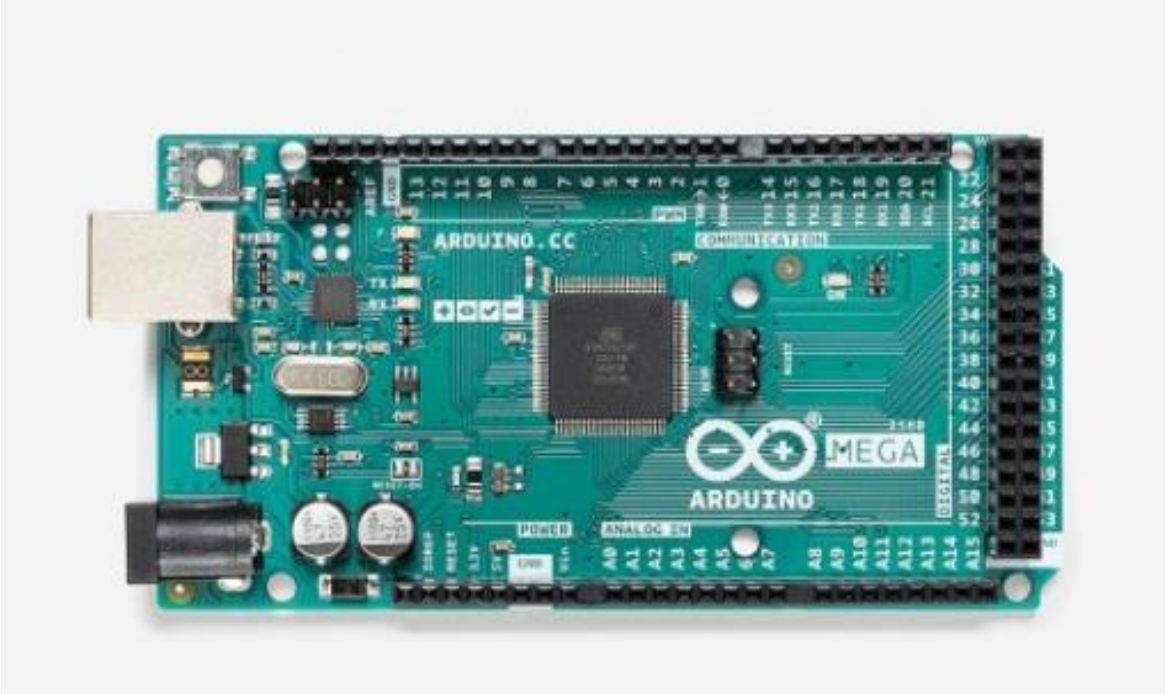
  Το λογισμικό του Arduino έχει εκδοθεί ως εργαλείο ανοιχτού λογισμικού, διαθέσιμο για επέκταση από έμπειρους προγραμματιστές. Η γλώσσα μπορεί να διευρυνθεί μέσω βιβλιοθηκών της C++, καθώς επίσης για όσους κατανοούν τις τεχνικές λεπτομέρειες των μικροεπεξεργαστών μπορούν να προσθέσουν και απευθείας κομμάτια AVR-C κώδικα στα Arduino προγράμματα τους, για να ενισχύσουν τις λειτουργίες τους.
- **Open-source και επεκτάσιμο υλικό (hardware)**

  Η σχεδίαση των πλακετών Arduino έχει εκδοθεί υπό μια άδεια που ονομάζεται Creative Commons και επιτρέπει σε έμπειρους σχεδιαστές κυκλωμάτων να φτιάξουν τις δικές τους εκδόσεις διαφόρων modules, είτε της πλακέτας είτε επέκτασης αυτής (π.χ. Wi-Fi module), να τις διευρύνουν και να τις βελτιώσουν. Ακόμη και σχετικά άπειροι χρήστες μπορούν να χτίσουν τα δικά τους μικρότερα modules με την χρήση ενός απλού breadboard, κάτι το οποίο εξοικονομεί ακόμη περισσότερα χρήματα.
- **Εύκολος χειρισμός των δεδομένων και Cloud**

  Το Arduino δίνει την δυνατότητα ασύρματης μεταφοράς των αρχείων με εύκολο τρόπο με την χρήση των αντίστοιχων modules (π.χ. Xbee) αλλά και μέσω του Arduino IoT Cloud για την αποθήκευση των δεδομένων και τον απομακρυσμένο έλεγχο του εξοπλισμού. Συνεπώς, καθιστά ευκολότερο το χτίσιμο του δικτύου των αισθητήρων, την παρακολούθηση σε πραγματικό χρόνο και την συμβατότητα με Wi-Fi και LoRa

### 2.2.1.3 Το Arduino που χρησιμοποιήσαμε (Arduino MEGA 2560 REV 3)

Για την δική μας εφαρμογή και τον έλεγχο του IoT οικοσυστήματος χρησιμοποιήσαμε την πλακέτα Arduino MEGA 2560 REV 3, η οποία έχει κόστος περίπου 35 ευρώ (€) κάτι που χρήζει αναφοράς δεδομένου ότι οι IoT εφαρμογές οφείλουν να κρατούν το κόστος χαμηλά όσο γίνεται τόσο στην αγορά του εξοπλισμού όσο και στην κατανάλωση πόρων (π.χ. ρεύμα, ισχύς).




*Εικόνα 6: Arduino MEGA 2560 REV 3 (Πηγή: [10])*


*Εικόνα 7: Arduino MEGA 2560 REV 3 (Πηγή: [10])*

Το Arduino Mega 2560 είναι μια πλακέτα μικροελεγκτή βασισμένα στο ATmega2560 και έχει 54 ψηφιακές θύρες εισόδου/εξόδου (από τις οποίες οι 15 μπορούν να χρησιμοποιηθούν ως PWM έξοδοι), 16 αναλογικές θύρες εισόδου, 4 UARTs (hardware serial ports), έναν κρυσταλλικό ταλαντωτή 16 MHz, μια θύρα

σύνδεσης USB, μια θύρα jack, έναν ICSP header και το reset button για την επαναφορά της πλακέτας. Από την κατασκευή της περιέχει οτιδήποτε χρειάζεται για την υποστήριξη του μικροελεγκτή. Με τον υπολογιστή συνδέεται απλώς με ένα USB καλώδιο και η τροφοδοσία παρέχεται είτε μέσω ενός AC-to-DC adapter ή μέσω μπαταρίας. Τα τεχνικά χαρακτηριστικά του φαίνονται στην επόμενη εικόνα.

| | |
|---|---|
| Microcontroller | ATmega2560 |
| Operating Voltage | 5V |
| Input Voltage (recommended) | 7-12V |
| Input Voltage (limit) | 6-20V |
| Digital I/O Pins | 54 (of which 15 provide PWM output) |
| Analog Input Pins | 16 |
| DC Current per I/O Pin | 20 mA |
| DC Current for 3.3V Pin | 50 mA |
| Flash Memory | 256 KB of which 8 KB used by bootloader |
| SRAM | 8 KB |
| EEPROM | 4 KB |
| Clock Speed | 16 MHz |
| LED_BUILTIN | 13 |
| Length | 101.52 mm |
| Width | 53.3 mm |
| Weight | 37 g |

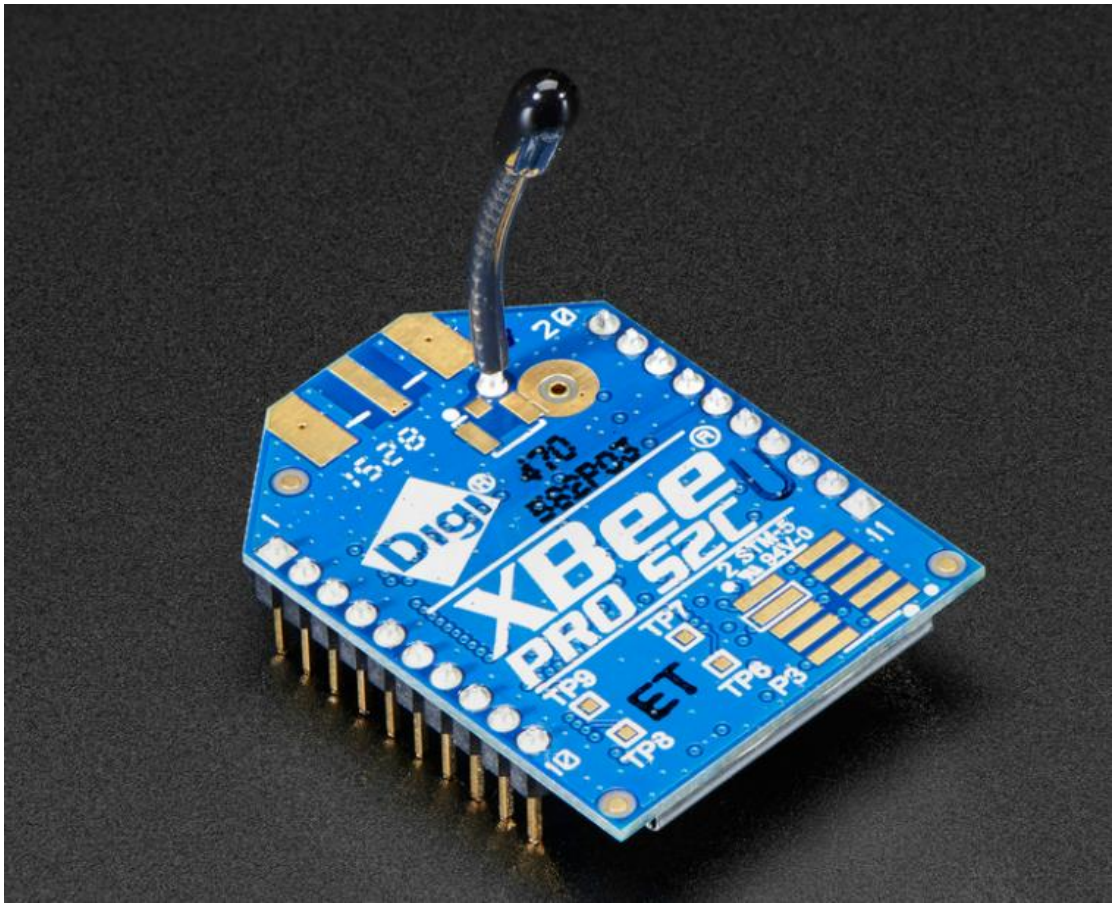
*Εικόνα 8: Τεχνικά χαρακτηριστικά*

### 2.2.2 XBee module για ασύρματη μετάδοση

Όπως είπαμε παραπάνω το Arduino είναι πλήρως επεκτάσιμο τόσο στο λογισμικό όσο και στο υλικό. Για να υπάρξει η δυνατότητα ασύρματης μεταφοράς δεδομένων χρησιμοποιήθηκαν τα XBees [11], τα οποία είναι ασύρματοι πομποδέκτες. Ο λόγος που επιλέξαμε αυτό το module για την επέκταση του Arduino μας ώστε να μπορούμε να μεταφέρουμε ασύρματα τα δεδομένα από το Arduino σε ένα Raspberry Pi 4, όπως θα αναλυθεί και παρακάτω, είναι η αξιοπιστία τους, το χαμηλό τους κόστος και η αμφίδρομη επικοινωνία μεταξύ τους σε ικανοποιητικές ταχύτητες. Ακόμη, τα XBees είναι ιδιαίτερα ευέλικτα καθώς μπορούν να στέλνουν και λαμβάνουν δεδομένα απλά μέσω μιας σειριακής θύρας, κάτι που τα καθιστά συμβατά με τους υπολογιστές,

με τους μικροελεγκτές όπως το Arduino και με άλλες IoT συσκευές που λειτουργούν ως υπολογιστές (RPi). Ακόμη είναι πολύ διαμορφώσιμα και παραμετροποιήσιμα, καθώς μπορούν να συνεργάζονται με πολλά άλλα XBees ή απλα και σε δυάδες, και μπορούν να ελεγχθούν απομακρυσμένα για την παρακολούθηση δεδομένων σε πραγματικό χρόνο. Ωστόσο, ένα ζεύγος από XBee από μόνο του δεν είναι τόσο χρήσιμο, διότι χρειάζεται ένα ενδιάμεσο module που ονομάζεται XBee Explorer, το οποίο είναι μια μικρή σχετικά πλακέτα στην οποία «κουμπώνει» πάνω ο πομποδέκτης και δρα ως διεπαφή μεταξύ του υπολογιστή και του XBee, οπότε αποτελεί το module που διαμορφώνει το XBee ώστε να μεταφέρει δεδομένα από και προς έναν υπολογιστή.

Το βασικό module πομποδέκτη που εμείς χρησιμοποιήσαμε στην παρούσα εργασία είναι το XBee Pro Module – ZB Series 2SC – 63mW with Wire Antenna – XBP24CZ7WIT-004 [12].




*Εικόνα 9: XBee πομποδέκτης (Πηγή: [12])*

Τα τεχνικά χαρακτηριστικά του συγκεκριμένου είναι τα εξής:

- Μέγιστο ρεύμα πομπού: 205 mA
- Μέγιστο ρεύμα δέκτη: 47 mA
- Τάση: 3.3 V
- Indoor/Urban: μέχρι 90 μέτρα
- Outdoor line-of-sight: μέχρι 3200 μέτρα
- Ισχύς μετάδοσης: 63 mW (18 dBm)

- Ευαισθησία δέκτη: -102 dBm
- Διαστάσεις: 24mm x 33mm x 9mm
- Βάρος: 3.91g

Το module XBee explorer που χρησιμοποιήθηκε για να έχουμε άμεση πρόσβαση στους XBee πομποδέκτες μέσω της σειριακής ή των προγραμματιζόμενων θυρών είναι το SparkFun XBee Explorer USB, το οποίο είχε και 4 λυχνίες LED πάνω για πιο εύκολο debug κατά την λειτουργία των XBees (Rx, Tx, RSSI (signal-strength δείκτης), δείκτης τροφοδοσίας) [12].

*Εικόνα 10: SparkFun XBee Explorer USB (Πηγή: [11])*

## 2.2.3 Οι αισθητήρες που χρησιμοποιήθηκαν

Για την IoT εφαρμογή χρησιμοποιήθηκαν 3 τύποι αισθητήρων για Arduino:

1) Αισθητήρας θερμοκρασίας και υγρασίας αέρα
2) Χωρητικός (capacitive) αισθητήρας υγρασίας εδάφους (x2)
3) Αισθητήρας ανίχνευσης UV

## 2.2.3.1 Αισθητήρας θερμοκρασίας και υγρασίας αέρα

Ο πρώτος αισθητήρας που χρησιμοποιήσαμε για την εκτέλεση των πειραμάτων-εφαρμογής για το IoT οικοσύστημα μας είναι ένας αισθητήρας ο οποίος συνδέεται σε ένα pin του Arduino που αναφέρεται σε digital input-output. Η θερμοκρασία είναι μια απλή και κατανοητή μετρική για να κατανοήσουμε το περιβάλλον στο οποίο ανήκει το φυτό. Η υγρασία όμως και πιο συγκεκριμένα η σχετική υγρασία που μας δίνουν οι αισθητήρες και κατ’ επέκταση οι μετρήσεις είναι λίγο πιο

σύνθετη μέτρηση. Κατ' αρχάς είναι μια τιμή από το 0 μέχρι το 1 ή καλύτερα και συνηθέστερα όπως την βλέπουμε να εκφράζεται είναι ένα ποσοστό (%). Η σχετική υγρασία (RH – Relative Humidity) λοιπόν είναι το ποσό της υγρασίας στον αέρα σε μια συγκεκριμένη θερμοκρασία προς το μέγιστο ποσό υγρασίας που μπορεί να συγκρατήσει ο αέρας στην θερμοκρασία αυτή. Για να είμαστε λίγο πιο ξεκάθαροι σε αυτό το σημείο, ο αέρας δεν μπορεί να συγκρατήσει με την φυσική έννοια τους υδρατμούς, απλώς όταν μαζεύεται ένα ποσό υδρατμών στον αέρα αυτό συμπυκνώνεται ως αυτό που αποκαλούμε υγρασία κάνοντας την αίσθηση του αέρα νωπή. Η μέτρηση της απόλυτης υγρασίας αφενός θα ήταν πιο δύσκολη αφετέρου δεν θα ήταν χρήσιμη και ας εξηγήσουμε γιατί. Η απόλυτη υγρασία είναι η ακριβής ποσότητα υγρασίας στον αέρα, δηλαδή η μέτρηση της ποσότητας των υδρατμών (gas vapor) σε $g / m^3$ στον αέρα, ανεξαρτήτως θερμοκρασίας. Εν αντιθέσει με την σχετική υγρασία (RH) που όπως περιγράψαμε είναι η ποσότητα των υδρατμών σε μια συγκεκριμένη θερμοκρασία, σχετικά με την μέγιστη αντίστοιχη ποσότητα στον ίδιο όγκο αέρα.

Ο αισθητήρας που εμείς χρησιμοποιήσαμε για το πείραμα είναι ο DHT22/AM2302 που είναι ένα module για Arduino. Είναι ένας απλός και φθηνός αισθητήρας για ψηφιακές μετρήσεις. Χρησιμοποιεί έναν χωρητικό (capacitive) αισθητήρα υγρασίας και ένα θερμίστορ (thermistor) για την μέτρηση του περιβάλλοντος αέρα [13]. Στα τεχνικά χαρακτηριστικά του:

- Πολύ χαμηλό κόστος
- 3 – 5 V DC τροφοδοσία
- Μέγιστο ρεύμα 2.5 mA (πολύ χαμηλή κατανάλωση)
- Εύρος μετρήσεων : 0 – 100% RH με ακρίβεια ±2%
  -40 ~ 80 °C με ακρίβεια ±0.5°C
- 0.5 Hz sampling rate (μετρήσεις κάθε 2 sec)
- Εσωτερική αντίσταση 5.1 kΩ

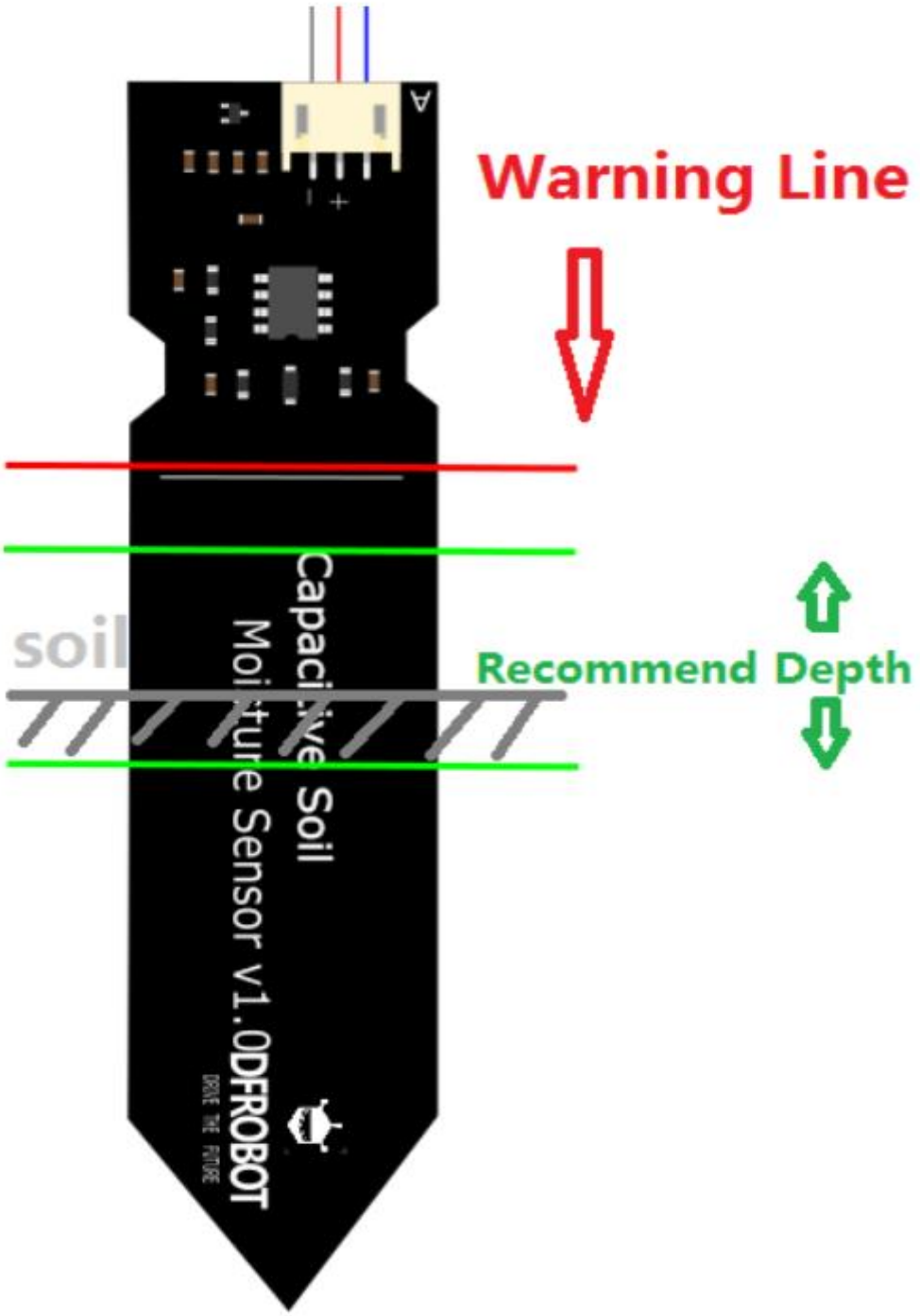
*Εικόνα 11: Αισθητήρας θερμοκρασίας και υγρασίας (Πηγή: [13])*

### 2.2.3.2 Χωρητικός (capacitive) αισθητήρας υγρασίας εδάφους

Εν συνεχεία, χρησιμοποιήθηκαν δύο χωρητικοί (capacitive) αισθητήρες για Arduino για την μέτρηση της υγρασίας εδάφους και της θα περιγράψουμε πιο συγκεκριμένα στα επόμενα κεφάλαια για την μέτρηση της υγρασίας του χώματος στην γλάστρα όπου διεξήχθησαν τα πειράματα. Η υγρασία εδάφους ουσιαστικά αποτελεί το ποσό του νερού που είναι αποθηκευμένο στο χώμα και της τρεις φύσεις του (στερεό, υγρό και αέριο). Εδώ δεν έχουμε κάποια ακριβής μέτρηση της της της αισθητήρες αλλά μια «καλιμπραρισμένη» μέτρηση με βάση πειράματα και παρατηρήσεις. Επειδή, οι αισθητήρες συνδέονται σε ψηφιακές θύρες του Arduino το output ουσιαστικά είναι μια 10-bit έξοδος, δηλαδή της αριθμός από το 0 έως το 1023, όπου οι ρεαλιστικές μετρήσεις φυσικά περιορίζονται σε ένα μικρό φάσμα του πλήρους εύρους. Ο αισθητήρας που χρησιμοποιήσαμε ονομάζεται Capacitive Soil Moisture Sensor v1.2 και μετράει συνεπώς την υγρασία εδάφους μέσω capacitive (χωρητικός – δηλαδή με χρήση πυκνωτών) sensing και όχι με resistive sensing (με την χρήση αντιστάσεων) της οι υπόλοιποι αισθητήρες. Είναι κατασκευασμένος από υλικό ανθεκτικό στην διάβρωση που τον καθιστά εξαιρετικό για μακροχρόνια χρήση. Η χρήση του είναι πολύ απλή,

απλώς τον εισάγουμε στο χώμα και της παράγει real-time δεδομένα εφόσον συνδεθεί με το Arduino. Στα τεχνικά του χαρακτηριστικά:

- Τάση Λειτουργίας: 3.3 – 5.5 V (DC)
- Τάση Εξόδου: 0 ~ 3.0 V (DC)
- Ρεύμα Λειτουργίας: 5 mA




*Εικόνα 12: Χωρητικός αισθητήρας υγρασίας εδάφους (Πηγή: [14])*

Οι τιμές εξόδου που δίνει ο συγκεκριμένος αισθητήρας έχουν εύρος από [260 – 520) και εξαρτώνται από το βάθος εισχώρησης του στο έδαφος και από το πόσο σφιχτά είναι

συμπιεσμένο το χώμα γύρω του [14]. Αυτές λοιπόν οι τιμές ανίχνευσης χωρίζονται σε τρεις κατηγορίες όπου τα όριο είναι κάπως σχετικά αλλά δεν υπάρχει και ακριβής υπολογισμός της της γίνεται αντιληπτό παρά μόνο μια όλο και καλύτερη προσέγγιση με βάση της συνθήκες.

1. Στεγνό (Dry): [430 – 520)
2. Υγρό (Wet): [350 – 430)
3. Νερό (Water): [260 – 350)

### 2.2.3.3 Αισθητήρας ανίχνευσης UV

Ο τελευταίος αισθητήρας που χρησιμοποιήθηκε στο IoT οικοσύστημα που κατασκευάσαμε ήταν ο αισθητήρας VEML6070 ο οποίος είναι της advanced αισθητήρας για την μέτρηση του υπεριώδους φωτός που χρησιμοποιεί το I2C πρωτόκολλο (για την επικοινωνία μεταξύ των συσκευών) και σχεδιάστηκε με CMOS. Ο χειρισμός του είναι πολύ εύκολος. Χρησιμοποιώντας το I2C πρωτόκολλο επικοινωνίας για την «ζεύξη» με το Arduino, το οποίο της επιτρέπει να λάβουμε ως έξοδο μια τιμή μεταξύ του 0 και του 12. Για εμάς αυτή η μέτρηση ουσιαστικά εκδηλώνει το πόσο φως δέχεται το φυτό ανάλογα με την ώρα της ημέρας, ώστε να κάνει της απαραίτητες διεργασίες του της η φωτοσύνθεση. Της θα δούμε στα αποτελέσματα στο τέλος κατά της βραδινές ώρες η τιμή εξόδου είναι 0 και κατά της μεσημεριανές ώρες που το φυτό βρίσκεται σε πλήρη έκθεση στον ήλιο αγγίζει την τιμή 12. Ο αισθητήρας VEML6070 ενσωματώνει μια φωτοδίοδο, ενισχυτές και analog/digital κυκλώματα ενσωματωμένα σε ένα και μόνο chip. Ακόμη, υιοθετεί την FiltronTM UV τεχνολογία που παρέχει την βέλτιστη φασματική ανίχνευση για την μέτρηση του UV spectrum. Επιπρόσθετα, η ευαισθησία των μετρήσεων είναι γραμμική όσον αφορά το φυσικό ηλιακό φως στην κλίμακα UV και μπορεί να ρυθμιστεί μέσω μιας εξωτερικής αντίστασης. Ο αισθητήρας έχει και shutdown mode το οποίο μπορεί να προγραμματιστεί, μειώνοντας έτσι την κατανάλωση ρεύματος ώστε να είναι κάτω από 1 μA [15]. Στα τεχνικά του χαρακτηριστικά έχουμε:

- Τάση Λειτουργίας: 2.7 – 5.5V (DC)
- Μετατροπή ηλιακού UV φωτός σε ψηφιακά δεδομένα
- Εξαιρετική απόδοση ακόμη και σε μακροχρόνια έκθεση σε ηλιακό φως
- Εσωτερική αντίσταση 270 kΩ
- Θερμοκρασίες Λειτουργίας: -40 ~ 85 °C

## 2.2.4 Single Board Computers ως IoT συσκευές

Κυρίως για το κομμάτι της image processing εφαρμογής χρησιμοποιήθηκαν διάφορες IoT συσκευές, ( Raspberry Pi 3B+ / 4B, Nvidia Jetson Nano και Google Coral dev board) στο κομμάτι της πρόβλεψης και παρουσίασης των αποτελεσμάτων. Στην δεύτερη IoT εφαρμογή χρησιμοποιήθηκε το Raspberry Pi 4B για την συλλογή, την παρουσίαση και την επεξεργασία των δεδομένων (μέσω της σειριακής θύρας) που παρείχαν οι αισθητήρες μέσω του Arduino.

### 2.2.4.1 Raspberry Pi 3B+ / 4B

Τα Raspberry Pi είναι υπολογιστές χαμηλού κόστους σε μέγεθος πιστωτικής κάρτας που συνδέονται απλά σε μια οθόνη υπολογιστή ή τηλεόραση και χρησιμοποιούν κανονικά πληκτρολόγιο και ποντίκι. Είναι μικρές αλλά πολύ ικανές συσκευές που επιτρέπουν από την εκμάθηση προγραμματισμού μέχρι την υλοποίηση πραγματικών εφαρμογών για την αγορά εργασίας. Συνεπώς είναι ικανές να υλοποιήσουν οτιδήποτε θα περιμέναμε να μπορεί να υλοποιήσει κι ένας σταθερός υπολογιστής (desktop), από απλό καθημερινό browsing μέχρι gaming ακόμη και την δημιουργία μικρών clusters. Ο κύριος λόγος που χρησιμοποιούνται τόσο ευρέως σε εφαρμογές είναι οι ικανότητες τους να επικοινωνούν με τον έξω κόσμο για την μετάδοση, διαχείριση και επεξεργασία διαφόρων δεδομένων καθώς και η ευκολία προσαρμογής στην χρήση τους που τα καθιστούν εξαιρετικά για χρήση από μικρά παιδιά μέχρι μεγάλους και έμπειρους προγραμματιστές. Το λειτουργικό που χρησιμοποιούν ονομάζεται Raspberry Pi OS από τον Αύγουστο του 2020, παλιότερα γνωστό ως Raspbian, και είναι σαφώς το προτεινόμενο λειτουργικό σύστημα για φυσιολογική χρήση ενός Raspberry Pi. Το Raspberry Pi OS είναι ένα ελεύθερο/δωρεάν λειτουργικό σύστημα βασισμένο στο Debian, βελτιστοποιημένο για το hardware του RPi. Το λειτουργικό αυτό σύστημα έρχεται με πάνω από 35000 πακέτα, προμεταγλωττισμένου και πακεταρισμένου λογισμικού με καλό format για εύκολη εγκατάσταση στο RPi. Το Raspberry Pi OS ως ελεύθερο λογισμικό βρίσκεται υπό συνεχή βελτίωση από το community, με έμφαση στην βελτίωση της σταθερότητας και της απόδοσης όσων περισσότερων Debian πακέτων γίνεται. Οι επεξεργαστές των Raspberry Pi βασίζονται στην τεχνολογία ARM. Ας δούμε εν συντομία τα χαρακτηριστικά των δύο RPi που χρησιμοποιήθηκαν ώστε να γίνει κατανοητό ότι πρόκειται για 2 μικρούς υπολογιστές με σαφέστατα πολύ χαμηλό κόστος αγοράς. Αξίζει να τονισθεί ότι οι δυνατότητες του Raspberry Pi 4B ωστόσο είναι πολύ μεγάλες για την δεδομένη τιμή, καθώς μπορεί να καλύψει ένα τεράστιο εύρος καθημερινών εφαρμογών έως και large-scale projects [16].

**Raspberry Pi 3 B+ (κόστος ~ 42 €)**

- Quad-Core Processor Cortex-A53 (ARMv8) 64-bit SoC @ 1.4 GHz
- 1 GB LPDDR2 SDRAM
- 128 GB microSD (storage)
- 5 V / 3 A DC power input (micro USB cable)
- 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE
- Gigabit Ethernet over USB 2.0 (maximum throughput 300 Mbps)
- Full-size HDMI
- 4 USB 2.0 ports
- Extended 40-pin GPIO (General Purpose Input Output) header


*Εικόνα 13:Raspberry Pi 3B+ (Πηγή: [16])*

**<u>Raspberry Pi 4B (κόστος ~ 64 €)</u>**

- Quad-Core Processor Cortex-A72 (ARMv8) 64-bit SoC @ 1.5 GHz
- 4 GB LPDDR4-3200 (MHz) SDRAM
- 128 GB microSD (storage)
- 5 V / 3 A DC (USB-C cable)
- 2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0, BLE
- Gigabit Ethernet
- 2 × micro-HDMI ports (up to 4kp60 supported)
- 2 USB 3.0 ports, 2 USB 2.0 ports
- Raspberry Pi standard 40 pin GPIO header (fully backwards compatible with previous boards)
- Power over Ethernet (PoE) enabled
- Operating temperature: 0 – 50 degrees C ambient

*Εικόνα 14: Raspberry Pi 4B (Πηγή: [16])*

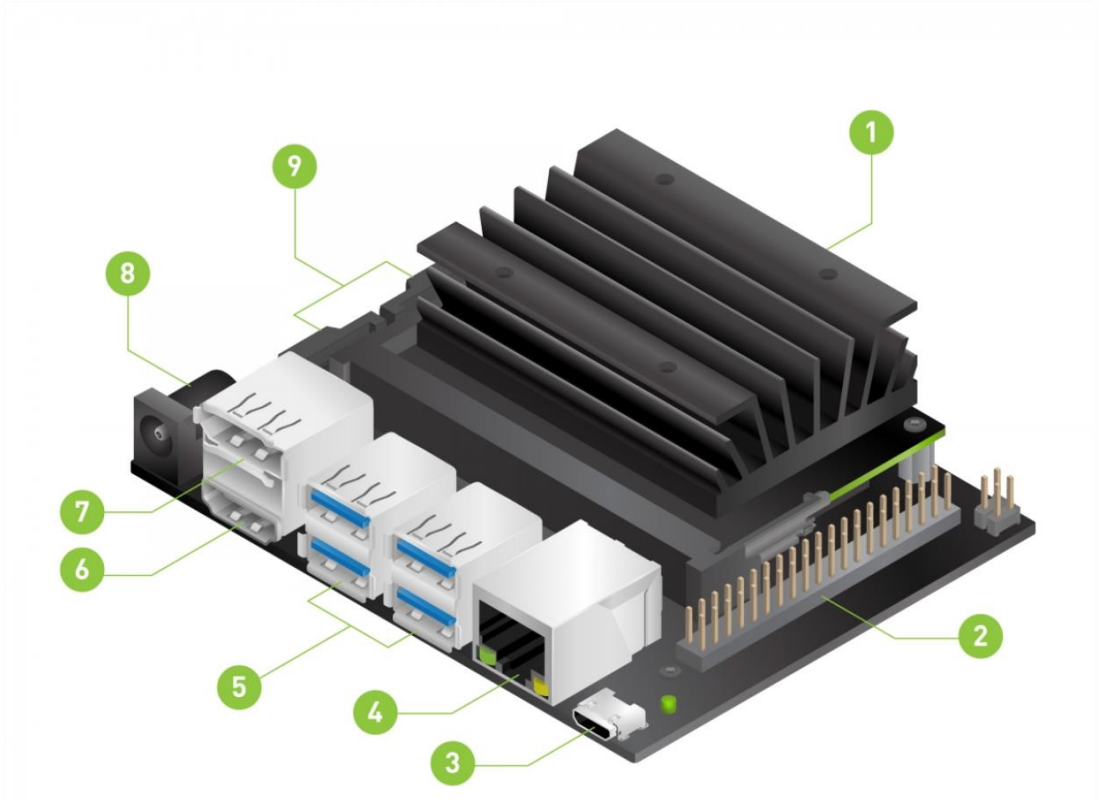
Συνεπώς, παρατηρώντας τα χαρακτηριστικά και μόνο παρατηρούμε ότι πρόκειται για ένα πολύ ισχυρό μηχάνημα δεδομένου του κόστους σε μέγεθος τσέπης, κάτι που καθιστά εξαιρετικό για IoT εφαρμογές.

### 2.2.4.2 Nvidia Jetson Nano

Το Nvidia® Jetson Nano ™ αποτελεί έναν μικρό αλλά πανίσχυρο υπολογιστή που επιτρέπει την παράλληλη εκτέλεση πολλαπλών νευρωνικών δικτύων για εφαρμογές όπως image classification, object detection, segmentation και speech processing. Παρέχει και αυτό συνεπώς μια εύκολη στην χρήση πλατφόρμα που τρέχει με λιγότερο από 5 Watts (χαμηλή κατανάλωση) και την καθιστά μια εξαιρετική IoT συσκευή. Το λειτουργικό που χρησιμοποιεί είναι Linux και πιο συγκεκριμένα το επίσημο λογισμικό του ονομάζεται Linux4Tegra, που είναι στην πραγματικότητα μία έκδοση του Ubuntu 18.04 σχεδιασμένη για να τρέχει στο hardware της Nvidia®. Αυτό που το κάνει να ξεχωρίζει από τα Raspberry Pi που αναλύσαμε παραπάνω είναι η ύπαρξη GPU (Graphics Processor Unit), η οποία επεκτείνει τις δυνατότητες του όσον αφορά την παράλληλη επεξεργασία και την επιτάχυνση σε εφαρμογές με νευρωνικά δίκτυα. Ως μέγεθος είναι κάπως μεγαλύτερο από τα Raspberry Pi, ωστόσο παραμένει ξεκάθαρα σε IoT πλαίσια με διαστάσεις 69mm x 45 mm ενώ ταυτόχρονα ενσωματώνει 260-pin edge connectors [17]. Ας δούμε τα τεχνικά χαρακτηριστικά του καθότι πρόκειται για ένα τόσο ισχυρό μηχάνημα που θα μπορούσε ίσως να μπει δίπλα και με κανονικούς υπολογιστές παρότι κοστίζει περίπου 110 – 120 €.

- CPU: Quad-Core ARM A57 @1.43 GHz
- GPU: 128-cores Maxwell

- Memory: 4 GB 64-bit LPDDR4 @ 25.6 GB/s
- Storage: microSD 128 GB
- Connectivity:  Gigabit Ethernet, M.2 Key E
- Display: HDMI and display port
- USB: 4x USB 3.0, USB 2.0 Micro-B
- Power: 5 V / 3 A (micro USB cable)
- Others: GPIO, I2C, I2S, SPI, UART




*Εικόνα 15: Nvidia® Jetson Nano ™ (Πηγή: [17])*

1) microSD card slot for main storage
2) 40-pin expansion header
3) Micro-USB port for 5V power input, or for Device Mode
4) Gigabit Ethernet port
5) USB 3.0 ports (x4)
6) HDMI output port
7) DisplayPort connector
8) DC Barrel jack for 5V power input
9) MIPI CSI-2 camera connector

### 2.2.4.3 Google Coral dev board

Το Google Coral EDGE TPU αποτελεί ένα ολοκληρωμένο κύκλωμα για συγκεκριμένη εφαρμογή, γνωστότερο και ως ASIC (Application-Specific-Integrated-Circuit) που κατασκευάστηκε από την Google για Machine Learning και εκτελεί πολύ

γρήγορα το inference σε TensorFlow lite μοντέλα με χαμηλή κατανάλωση ισχύος. Την έννοια του inference time θα την εισάγουμε και παρακάτω, αλλά ας αναφέρουμε εισαγωγικά ότι αποτελεί τον χρόνο για την ολοκλήρωση της διαδικασίας πρόβλεψης χρησιμοποιώντας ένα εκπαιδευμένο νευρωνικό δίκτυο / αλγόριθμο μηχανικής μάθησης. Το Google Coral είναι μια πλατφόρμα γενικού σκοπού για machine learning εφαρμογές και βασίζεται στο λογισμικό Mendel Linux, που είναι ένα παράγωγο του Debian Linux από την Google [18].

*Εικόνα 16: Google Coral EDGE TPU (Πηγή: [18])*

Αποτελεί και αυτό έναν μικρό υπολογιστή, ιδανικό για γρήγορη εκτέλεση αλγορίθμων μηχανικής μάθησης για πρόβλεψη. Μπορεί να χρησιμοποιηθεί για την δημιουργία πρωτότυπων σε ενσωματωμένα συστήματα και κατόπιν αυτά να περάσουν στην παραγωγή με την βοήθεια του on-board Coral System-on-Module (SoM) το οποίο επιτρέπει στο Coral να συνδυαστεί με το εκάστοτε custom hardware. Το χαρακτηριστικό που το κάνει να ξεχωρίζει από τις άλλες συσκευές είναι ο ML accelerator που έχει που βασίζεται σε TPU (Tensor Processing Unit) co-processors που είναι ουσιαστικά και αυτοί που του επιτρέπουν να κάνει τόσο γρήγορα τις «πράξεις» μέσα σε ένα νευρωνικό. Είναι ικανοί να εκτελέσουν μέχρι 4 τρισεκατομμύρια πράξεις το δευτερόλεπτο (Trillion Operations Per Second – TOPS), καταναλώνοντας μόλις 0.5 Watt για κάθε TOPS (2 TOPS per Watt). Όπως αναφέραμε παραπάνω υποστηρίζει TensorFlow Lite, οπότε δεν χρειάζεται να χτιστεί ένα μοντέλο στο Google Coral, αλλά απλώς ένα trained μοντέλο μπορούμε να το μετατρέψουμε σε TF-lite μοντέλο καθώς η πραγματική του δύναμη είναι στο inference, δηλαδή στην πρόβλεψη π.χ. στο image classification. Το κόστος του ανέρχεται περίπου στα 110-120 € όπως και του Jetson Nano και αποτελεί μία πλατφόρμα που επιδέχεται σημαντικής βελτίωσης ακόμη αν και έχει ήδη δείξει πολύ καλά αποτελέσματα. Οι διαστάσεις του είναι στα IoT πλαίσια

εννοείται καθώς έχει μικρό μέγεθος (88 mm x 60 mm x 24 mm) αλλά με μεγάλε δυνατότητες. Ας δούμε και τα τεχνικά χαρακτηριστικά του:

- CPU: quad-core Cortex-A53 (ARM architecture)
- GPU: Integrated GC7000 Lite Graphics
- RAM:  1 GB LPDDR4
- Storage: 128 GB microSD
- Flash Memory: 8GB eMMC, MicroSD slot
- Wireless: Wi-Fi 2x2 MIMO (802.11b/g/n/ac 2.4/5GHz) and Bluetooth 4.2
- Lan: Gigabit Ethernet port
- USB: Type-C OTG; Type-C power; Type-A 3.0 host; Micro-B serial console
- Power: 5V DC (USB Type-C) / 3 A
- Display: HDMI 2.0a (full size); 39-pin FFC connector for MIPI-DSI display (4-lane); 24-pin FFC connector for MIPI-CSI2 camera (4-lan
- GPIO: 3.3V power rail; 40 - 255 ohms programmable impedance; ~82 mA max current

# Κεφάλαιο 3 – Συναφής Βιβλιογραφία

Στο Κεφάλαιο αυτό γίνεται παράθεση σχετικών άρθρων και δημοσιεύσεων που αφορούν κυρίως την χρήση των CNN σε εφαρμογές για την γεωργία μαζί με τις σχετικές μετρικές αξιολόγησης, κατ' αντιστοιχία με την δική μας εφαρμογή που χρησιμοποιεί τα Convolution Neural Networks για την πραγματοποίηση του Image Classification για ασθένειες φύλλων, καθώς και ορισμένων παραδειγμάτων που μελετήσαμε σχετικά με IoT εφαρμογές στο κομμάτι της γεωργίας που μας ενέπνευσε να υλοποιήσουμε πειραματικά το δικό μας IoT οικοσύστημα.

## 3.1 Η χρήση των Convolutional Neural Networks στην γεωργία

Η μηχανική μάθηση και ειδικότερα η βαθιά μάθηση (Deep Learning) αποτελεί μια πολύ μοντέρνα τεχνική για επεξεργασία εικόνας με πολλές δυνατότητες. Έχει εφαρμοστεί με μεγάλη επιτυχία σε πολλούς τομείς, ένας εκ των οποίων φυσικά είναι και η γεωργία. Πολύ σημαντικό ρόλο σε αυτό έχει παίξει η χρήση των CNN για αντιμετωπιστούν διάφορα challenges σχετικά με την γεωργική παραγωγή (π.χ. φαγητό). Το κοινό χαρακτηριστικό που αναφέρεται σε όλες τις δημοσιεύσεις είναι ότι η επιτυχία κάθε CNN μοντέλου έγκειται σε μεγάλο βαθμό στην ποιότητα των δεδομένων που χρησιμοποιούνται κάτι που καθιστά απαραίτητο το pre-processing των δεδομένων όπως θα επισημάνω και στην δική μου υλοποίηση. Οι μετρικές που εν γένει χρησιμοποιούνται σε τέτοιου είδους εφαρμογές είναι οι εξής:

- **Validation Accuracy**

  Το ποσοστό των σωστών προβλέψεων στο validation / test data set

- **Root Mean Square Error (RMSE)**

  Τυπική απόκλιση της διαφοράς μεταξύ των εκτιμώμενων τιμών και των πραγματικών τιμών

- **Precision, Recall, F1 Score**

  Για το precision και το recall θα αναφερθούμε αναλυτικότερα στο τέλος του κεφαλαίου, το F1 Score είναι ο αρμονικός μέσος των τιμών precision και recall, όπου για multi-classification προβλήματα υπολογίζεται ανάμεσα σε όλες τις κλάσεις

- **Quality Measure**

  Υπολογίζεται από τον πολλαπλασιασμό του sensitivity ( ποσοστό των pixels τα οποία ανιχνεύθηκαν σωστά) με το specificity (ποσοστό των ανιχνευμένων pixels που είναι πραγματικά σωστά) [19]

- **Ratio of total fruits counted (RFC)**

Η αναλογία του εκτιμώμενου πλήθους φρούτων μιας κλάσης όπως υπολογίζεται από το CNN μοντέλο προς την πραγματική μέτρηση που έχει προηγηθεί από τους συγγραφείς ή από ειδικούς [20], [21]

- **LifeCLEF metric (LC)**

  Η βαθμολογία που σχετίζεται με την θέση των σωστών ειδών στην λίστα των ανακτημένων ειδών κατά την διάρκεια του LifeCLEF 2015 Challenge [22]

Ακολούθως, παρουσιάζονται οι εφαρμογές που κρίνονται συναφέστερες, σημαντικότερες και πιο αντιπροσωπευτικές για την περιοχή και τα προβλήματα που μελετά η διπλωματική αυτή. Ας αρχίσουμε με το leaf disease detection με το οποίο και ασχολήθηκα. Η πρώτη δημοσίευση που μελετήθηκε των Sladojevic et al. [23] στόχευε στο classification 13 διαφορετικών τύπων ασθενειών σε φυτά, συν των υγιών φυτών. Το dataset περιείχε περίπου 4400 φωτογραφίες, πέτυχε validation accuracy 96.30% και υλοποιήθηκε με το Caffe Framework και το CaffeNet DL μοντέλο, που αποτελεί ένα έτοιμο νευρωνικό δίκτυο για μηχανική μάθηση. Μια ακόμη σχετική δημοσίευση σχετική με το classification ασθενειών φύλλων είναι των Renugambal και Senthilraja [24] όπου πρότειναν μια τεχνική τεχνητής νοημοσύνης για ασθένειες ζαχαροκάλαμων υλοποιώντας την λύση τους με Support Vector Machine (SVM) πετυχαίνοντας ακρίβεια της τάξης του 91%. Αντίστοιχα με τα φύλλα μελετώνται και οι ασθένειες των φυτών με την αξιοποίηση σχετικών εικόνων. Δυο σχετικές δημοσιεύσεις είναι πρώτον των Mohanty et al. [25] όπου αναγνώριζαν 14 είδη καλλιέργειας και 26 ασθένειες χρησιμοποιώντας τα έτοιμα μοντέλα AlexNet και GoogleNet μέσω του Caffe Framework και ένα μεγάλο data set της τάξης των 54000+ φωτογραφιών. Τα αποτελέσματα τους παρουσιάστηκαν με βάση την μετρική F1 score που υπολογίστηκε ίση με 0.9935. Δεύτερον, έχουμε την δημοσίευση των Amara et al. [26] όπου ταξινόμησαν (classify) τις ασθένειες της μπανάνας χρησιμοποιώντας το LeNet, ένα dataset και με 3700 φωτογραφίες και πετύχανε ακρίβεια 96% (validation accuracy).

Ξεφεύγοντας από το κομμάτι των ασθενειών μελετήθηκαν και κάπως γενικότερες δημοσιεύσεις για να δημιουργηθεί η κατάλληλη αντίληψη της χρήσης του Image Processing στην γεωργία εν γένει. Οι Kussul et al. [27] πρότειναν μια μέθοδο για την κατηγοριοποίηση της σοδειάς (σιτάρι, καλαμπόκι, σόγια, ζαχαρότευτλα) αξιοποιώντας δορυφορικές εικόνες και βίντεο πετυχαίνοντας validation accuracy 94.60% με την χρήση custom νευρωνικού δικτύου. Ακόμη μια δημοσίευση που χρησιμοποίησε custom νευρωνικό δίκτυο, όπως κι εγώ στην παρούσα διπλωματική, είναι των Grinblat et al. [28] που στόχευε στην αναγνώριση φυτών (white, soya and red beans) από τα μοτίβα στα νεύρα των φύλλων. Χρησιμοποιήθηκαν λίγες σχετικά εικόνες στο dataset (866 χωρισμένες σε 3 κλάσεις), ωστόσο επιτεύχθηκε validation accuracy της τάξης του 96.90%. Μια ακόμη δημοσίευση που θα αναφερθεί από τις πραγματικά πολλές υπάρχουσες είναι των Kuwata και Shibasaki [29], όπου κάνανε μια εκτίμηση του κέρδους της σοδειάς σε μια κομητεία των ΗΠΑ (Illinois) αξιοποιώντας δεδομένα που συλλέχθηκαν από το CRU (Climate Research Unit) από το 2001 μέχρι και το 2010, με ακρίβεια εκτίμησης RMSE = 6.298.

Ας αναφέρουμε δύο εφαρμογές και από τον τομέα της καταμέτρησης των φρούτων, όπου χρησιμοποιήθηκε η μετρική RFC που αναφέρθηκε παραπάνω. Οι συγγραφείς του [21] με την αξιοποίηση του TensorFlow και του ResNet μοντέλου, που είναι και το framework το οποίο και εγώ επέλεξα για την υλοποίηση του CNN, και την παραγωγή 24000 συνθετικών εικόνων δώσανε μια εκτίμηση του αριθμού των τοματών με RFC = 91% και 1.16 RMSE σε πραγματικές εικόνες, και RFC = 93% και 2.52 RMSE στις custom συνθετικές εικόνες τους. Οι συγγραφείς του [20] χρησιμοποιώντας το Caffe Framework «χαρτογράφησαν» την κατηγοριοποίηση μεταξύ μήλων και πορτοκαλιών από τις εικόνες εισόδου ως προς τον συνολικό πληθάριθμο των φρούτων. Χρησιμοποιήσαν 71 φωτογραφίες πορτοκαλιών (1280 x 960) και 21 φωτογραφίες μήλων (1920 x 1200) και πέτυχαν ακρίβεια 0.968 RFC στα πορτοκάλια και 0.913 στα μήλα. Αντίστοιχες δημοσιεύσεις σχετικά με την ανίχνευση φρούτων παρουσιάζονται και από τους Bargoti και Underwood [30] και Sa et al. [31] με F1 score 0.8-0.9.

Κάπου εδώ αξίζει να σημειωθεί ότι είναι τέτοια η φύση του προβλήματος όπως αντιλαμβανόμαστε που κάθε ερευνητής παρουσιάζει μια μοντελοποίηση ενός συγκεκριμένου προβλήματος, για παράδειγμα την αναγνώριση ασθενειών στα φύλλα, δίνοντας έτσι όχι τόσο την ανάλυση των εσωτερικών δομών και των αλγορίθμων, όσο κυρίως το κίνητρο ώστε ο καθένας να χρησιμοποιήσει το καλύτερο δυνατό dataset με το δικό του custom νευρωνικό δίκτυο ή ακόμη και κάποιο pre-trained με σκοπό να βελτιώσει τις μετρικές ακρίβειας, ανάλογα και με τις ανάγκες φυσικά του εκάστοτε ζητήματος.

## 3.2 IoT στην γεωργία

Όσον αφορά τις εφαρμογές του Internet of Things στον τομέα της γεωργίας θα δοθεί η γενικότερη εικόνα, η οποία έδωσε και σε εμάς το κίνητρο για την κατασκευή του δικού μας πειραματικού IoT οικοσυστήματος, ώστε να μπορέσουμε να δώσουμε την δική μας οπτική στην βελτίωση της γεωργίας, εν προκειμένω του ποτίσματος ενός φυτού. Η ιδέα του Smart farming (smart irrigation / monitoring / sensing networks) είναι απαραίτητη για την αντιμετώπιση των σύγχρονων προκλήσεων στην γεωργία σχετικά με την παραγωγικότητα, την περιβαλλοντική αλλαγή, την ασφάλεια των τροφίμων και την βιωσιμότητα [32]. Αυτό γίνεται εύκολα αντιληπτό διότι με την συνεχή αύξηση του πληθυσμού απαιτείται η αύξηση της παραγωγής τροφίμων, συνοδευόμενη από την προστασία των φυσικών οικοσυστημάτων χρησιμοποιώντας βιώσιμες αγροτικές/γεωργικές διαδικασίες. Ωστόσο, δεν αρκεί απλώς να παράγονται περισσότερα τρόφιμα, αλλά ταυτόχρονα να διασφαλίζεται η υψηλή θρεπτική τους αξία [3] σε συνδυασμό με την απαραίτητη ασφάλεια. Για να αντιμετωπισθούν όλες οι παραπάνω προκλήσεις δημιουργούνται πολυσύνθετα IoT γεωργικά οικοσυστήματα, ώστε να επιτευχθεί η παρακολούθηση, η μέτρηση και η ανάλυση διαφόρων φυσικών τιμών και φαινομένων αδιάλειπτα. Συνεπώς αναπτύσσονται νέες τεχνολογίες πληροφορίας και επικοινωνίας (ICT) για την διαχείριση μικρής κλίμακας σοδειάς (small scale crop/farm), όπου αν ενοποιηθούν δίνουν λύση στην διαχείριση οικοσυστημάτων μεγαλύτερης κλίμακας. Το κρίσιμο σημείο εδώ είναι ότι η λήψη των

αποφάσεων είναι απόλυτα εστιασμένη με βάση την εκάστοτε κατάσταση και την γνώση της τοποθεσίας, διότι κάθε small-scale εφαρμογή τρέχει τοπικά και βοηθάει στην παραλληλοποίηση της παρακολούθησης. Φυσικά, το Internet of Things διαπρέπει σε αυτό τον τομέα με χαρακτηριστικότερο παράδειγμα το remote sensing, το οποίο κατ' επέκταση συνδυάζεται με το cloud computing [33] και την ανάλυση μεγάλων δεδομένων [34]. Το remote sensing ουσιαστικά είναι κάθε λογής πληροφορία που λαμβάνεται από ένα απομακρυσμένο σημείο, ενώ ταυτόχρονα μπορούμε να την παρακολουθήσουμε, να την επεξεργαστούμε και να την αναλύσουμε. Αυτές οι πληροφορίες μπορεί να προέρχονται από απλούς αισθητήρες θερμοκρασίας, υγρασίας, καπνού κλπ. αλλά και από δορυφόρους ή UAVs όπως τα drones που παρέχουν large-scale εικόνες από ένα γεωργικό περιβάλλον [35]. Συνεπώς, όπως γίνεται αντιληπτό τα δεδομένα από αυτή την τηλεανίχνευση συμπεριλαμβάνουν περιοχές που πιθανώς να είναι δύσκολα ή και καθόλου προσβάσιμες στον άνθρωπο. Το IoT υπεισέρχεται λοιπόν σε αυτό το σημείο και με την προηγμένη τεχνολογία αισθητήρων λύνει τα χέρια του ανθρώπου για την μέτρηση διαφόρων παραμέτρων ενός αγρού και σε συνδυασμό με το cloud computing οι IoT συσκευές (π.χ. RPi) με χαμηλό κόστος κατανάλωσης συλλέγουν, αποθηκεύουν, επεξεργάζονται και μοντελοποιούν όλα αυτά τα πιθανώς ετερογενή δεδομένα σε πραγματικό χρόνο (real-time). Συνεπώς, συνδυάζοντας όλα τα παραπάνω με την αξιοποίηση των δυνατοτήτων του Διαδικτύου των Αντικειμένων δημιουργούνται καινοτόμες εφαρμογές και υπηρεσίες που βελτιώνουν την γεωργική παραγωγή και ασφάλεια, κατανοώντας για παράδειγμα τις διάφορες κλιματικές συνθήκες.

Ας δούμε περιληπτικά ορισμένες τέτοιες εφαρμογές. Αρχικά, οι συγγραφείς του [36] πρότειναν μια IoT τεχνολογία με την αξιοποίηση του cloud computing και του Li-Fi. Το Li-Fi είναι μια τεχνολογία ασύρματης επικοινωνίας που χρησιμοποιεί φως για την μετάδοση δεδομένων και την θέση μεταξύ των συσκευών και παρέχει ισχυρή κάλυψη στην ασύρματη διαχείριση των δεδομένων σε «πυκνούς» χώρους σε σχέση με το Wi-Fi. Συνεπώς έχει καλύτερο εύρος ζώνης, υψηλότερη αποδοτικότητα και διαθεσιμότητα καθώς και ισχυρότερη ασφάλεια συγκριτικά με το Wi-Fi. Η δημοσίευση τους αφορά αρχικά τον απομακρυσμένο έλεγχο για την εκτέλεση εργασιών όπως ψεκασμός, ξεβοτάνισμα, bird scaring, διατήρηση «επαγρύπνησης» και μέτρηση υγρασίας. Ακολούθως, περιείχε την υλοποίηση μιας «έξυπνης» αποθήκης όπου συμπεριλαμβανόταν η μέτρηση θερμοκρασίας, υγρασίας και η ανίχνευση κίνησης σε περίπτωση κλοπής. Τέλος προτείνανε ένα σύστημα απόφασης για «έξυπνη» άρδευση, όπως και οι συγγραφείς του [37], με «έξυπνο χειρισμό», όπου όλες οι διεργασίες και ο σχετικός χειρισμός γινόταν από απομακρυσμένες IoT συσκευές συνδεδεμένες στο Internet, που εκμεταλλευόντουσαν την διασύνδεση των καμερών, των αισθητήρων, του Li-Fi και των ZigBee modules με τις IoT συσκευές [38]. Ο Khelifi [39] σε μια πολύ σύγχρονη εφαρμογή που παρουσίασε φέτος (σ.σ. 2020) πρότεινε ένα σύστημα παρακολούθησης με την χρήση ενός ασύρματου δικτύου αισθητήρων για Precision Agriculture. Επεκτείνοντας την ιδέα της μέτρηση των διαφόρων ατμοσφαιρικών παραμέτρων πρότεινε ακολούθως έναν αλγόριθμο για την κατάλληλη ομαδοποίηση των κόμβων των αισθητήρων ώστε να παρέχουν την αποδοτικότερη κάλυψη στην γεωργική περιοχή. Οι Dong et al. [40] πρότειναν μια νέα μέθοδο για "precision

fertilization” συνδυάζοντας τις παραδοσιακές αναλυτικές μεθόδους καταγραφής των διαφόρων μετρήσεων με την μελέτη της πολυσύνθετης σχέσης μεταξύ των θρεπτικών στοιχείων του εδάφους, του fertilization και της επικερδούς σοδειάς με την χρήση νευρωνικών δικτύων. Έτσι, συνδυάσανε το IoT με την μηχανική μάθηση που οδήγησε στην βελτίωση την παραγωγικότητας, μειώνοντας ταυτόχρονα το κόστος παραγωγής και την ρύπανση, οδηγώντας σε «καθαρότερη» αγροτική παραγωγή.

# Κεφάλαιο 4 – Περιγραφή του προβλήματος

## 4.1 Ορισμοί και πλαίσιο του προβλήματος

Τα τελευταία χρόνια η επιρροή της τεχνητής νοημοσύνης είναι πολύ έντονη σε ένα μεγάλο εύρος εφαρμογών, οδηγώντας στην ανάγκη για όλο και περισσότερα δεδομένα και μεγαλύτερα επεξεργαστικά μοντέλα, κάτι που επέφερε την ανάγκη για αποδοτικότερη αξιοποίηση των διαθέσιμων πόρων (μνήμη, RAM κλπ.). Η τεχνητή νοημοσύνη συνεπώς είναι η γνώση που αποκτούν και παρουσιάζουν οι «μηχανές» (υπολογιστές), με βάση την φυσική γνώση που τους παρέχει ο άνθρωπος, με στόχο να μεγιστοποιηθεί η ικανότητα των υπολογιστών να μαθαίνουν και να επιλύουν διάφορα προβλήματα και ποικίλα tasks. Φυσικά η τεχνητή νοημοσύνη έχει πολλές μορφές και εκφάνσεις π.χ. επίλυση προβλημάτων, μηχανική μάθηση, επεξεργασία φυσικής γλώσσας, αντίληψη (όραση υπολογιστών) κ.ά.. Η μηχανική μάθηση (machine learning), η οποία αποτελεί θεμελιώδη ιδέα της έρευνας για την τεχνητή νοημοσύνη, είναι η μελέτη προγραμματιστικών αλγορίθμων που βελτιώνεται αυτόματα μέσω της εμπειρίας. Στην παρούσα εργασία μελετάται η περίπτωση της επιβλεπόμενης μάθησης, όπου ο άνθρωπος βάζει ετικέτες στα δεδομένα εισόδου, όπως θα δείξουμε και στο image classification πρόβλημα που θα περιγραφεί στην συνέχεια. Η κατηγοριοποίηση (Classification) χρησιμοποιείται για να καθορισθεί σε ποια κατηγορία ανήκει ένα δεδομένο εισόδου και προκύπτει σταδιακά καθώς το πρόγραμμα μας «βλέπει» μια σειρά παραδειγμάτων που ανήκουν στις συγκεκριμένες κατηγορίες, γνωρίζοντας φυσικά τις σχετικές ετικέτες. Με την παραπάνω διαδικασία συνεπώς εκπαιδεύεται το μηχάνημα ώστε να ξεχωρίζει χαρακτηριστικά (features) με βάση τα δεδομένα εισόδου (training dataset). Κατόπιν χρησιμοποιούμε ένα validation dataset, για το οποίο γνωρίζουμε τα δεδομένα εισόδου σε ποιες ετικέτες αντιστοιχούν και κατά την διαδικασία του training το χρησιμοποιούμε για να υπάρχει μια μετρική αξιολόγησης (accuracy) του πόσο καλά μαθαίνει το μηχάνημα μας στην εξέλιξη του χρόνου. Τέλος χρησιμοποιούμε το test set αφού ολοκληρωθεί πλήρως η εκπαίδευση, ώστε να αξιολογήσουμε πόσο καλά έμαθε το μοντέλο. Κατά την τελευταία διαδικασία «κρύβουμε» τις ετικέτες από τον υπολογιστή και αυτός με την γνώση που έχει αποκτήσει κατηγοριοποιεί τα δεδομένα εισόδου και εκ των υστέρων βλέπουμε πόσα πέτυχε σωστά και αντιλαμβανόμαστε συνεπώς και το πόσο αξιόπιστα έμαθε. Στο πρόβλημα μας τώρα θα χρησιμοποιήσουμε τα CNNs που αποτελούν μια κλάση των νευρωνικών δικτύων βαθιάς μάθησης ώστε να εκπαιδεύσουμε ένα μοντέλο που να επιλύει το image classification πρόβλημα. Η κατηγοριοποίηση αυτή είναι η διαδικασία φορτώματος των εικόνων στο μοντέλο και της εξόδου της κλάσης στην οποία τα κατατάσσει (ή εν γένει μια πιθανότητα ότι μια εικόνα ανήκει σε μια συγκεκριμένη κλάση). Η δομή του νευρωνικού δικτύου, οι διαδικασίες του training και του testing (evaluation) θα αναδειχθούν στο 5$^{ο}$ κεφάλαιο όπου θα υπάρχει η αναλυτικότερη υλοποίηση.

Το 2$^{ο}$ πρόβλημα το οποίο επιλέξαμε να αντιμετωπίσουμε είναι η δημιουργία ενός small-scale IoT οικοσυστήματος που θα παρέχει στον χρήστη τις απαραίτητες

πληροφορίες και δυνατότητες για τον έλεγχο του ποτίσματος μιας μικρής γλάστρας σε συνδυασμό με διάφορες ατμοσφαιρικές μετρήσεις όπου μπορεί να ελέγχει. Ας δούμε λίγο το γενικότερο πλαίσιο γύρω από τα IoT οικοσυστήματα. Οι IoT συσκευές δεν έχουν κάποια ιδιαίτερη σημασία από μόνες τους. Ένας μοναχικός αισθητήρας δεν είναι καλός για κάτι, ούτε καν μια ομάδα από αυτούς, εκτός αν είναι συνδεδεμένοι ο ένας με τον άλλον μέσω μιας πλατφόρμας που παράγει δεδομένα για περαιτέρω χρήση. Αυτό ουσιαστικά είναι ένα IoT οικοσύστημα, ένα ευρύ δίκτυο από συνδεδεμένες και αλληλεξαρτώμενες συσκευές και τεχνολογίες που εφαρμόζονται συνδυαστικά από τον άνθρωπο για την επίτευξη κάποιου στόχου, όπως για παράδειγμα η δημιουργία μιας έξυπνης πόλης. Αν αναλύσουμε τι συμβαίνει σε κάθε IoT οικοσύστημα καταλήγουμε πάντα σε ένα απλό σχήμα: μια **συσκευή** (device) συλλέγει δεδομένα και τα αποστέλλει μέσω του **δικτύου** (network) σε μια **πλατφόρμα** (platform) η οποία συγκεντρώνει τα δεδομένα για μελλοντική χρήση από τον **χρήστη** (agent).


*Εικόνα 17: IoT οικοσύστημα*

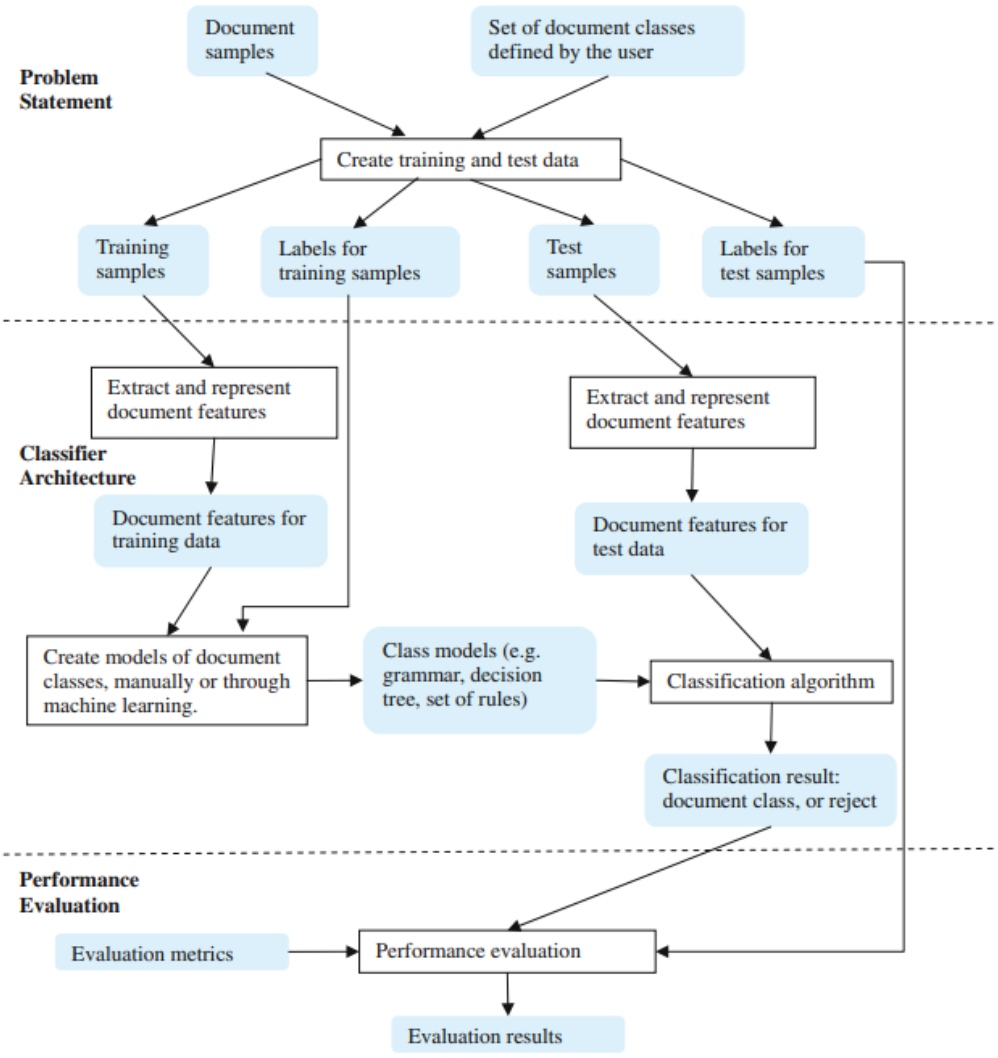
Συνεπώς αρχικά έχουμε τις συσκευές όπως οι αισθητήρες (π.χ. κίνησης, θερμοκρασίας) και ενεργοποιητές (π.χ. διακόπτης, ρότορας), όπου συνήθως σε μια εφαρμογή δεν υπάρχει μόνο ενός τύπου αισθητήρας ή ενεργοποιητής αλλά μια πληθώρα στοιχείων που μετράνε διάφορες παραμέτρους και δρουν ανάλογα. Για την επικοινωνία αυτών με τον «ανθρώπινο» κόσμο απαιτούνται τα κατάλληλα IoT gateways (π.χ. Xbee) όπου επιτρέπουν την επικοινωνία μεταξύ συσκευών και του δικτύου. Το Internet of Things απαιτεί την διασύνδεση συσκευών εν γένει μέσω ενός δικτύου, το οποίο δεν είναι απαραίτητα το IP (internet protocol) και εδώ κρύβεται η πραγματική δύναμη του IoT στην συνδεσιμότητα. Ανάλογα με τις ανάγκες του προβλήματος, υπάρχουν διάφορες επιλογές IoT συνδεσιμότητας από τις κλασσικές όπως το Wi-Fi, Bluetooth, μέχρι εξειδικευμένες τεχνολογίες όπως Zigbee, LPWAN (Low-Power Wide Area Networks) κ.ά.. Συνεπώς, ανάλογα με τις απαιτούμενες ανάγκες προσφέρονται τεχνολογίες που διαφέρουν στην απόσταση επικοινωνίας (range) και στην ταχύτητα μεταφοράς των

δεδομένων. Εν συνεχεία, οι IoT πλατφόρμες μπορεί να είναι cloud ή και φυσικές πλατφόρμες που αποτελούν τον ενωτικό κρίκο του οικοσυστήματος. Είναι οι «ήσυχοι» ενορχηστρωτές και διαχειριστές που φροντίζουν για την σωστή χρήση των συσκευών. Ακόμη, αποτελούν τον κόμβο οπού συλλέγει και συναθροίζει τα δεδομένα ώστε μετέπειτα ο χρήστης να μπορεί να τα αξιοποιήσει. Η επιλογή μιας IoT πλατφόρμας βασίζεται στην ασφάλεια, στην επεκτασιμότητα και στην διαλειτουργικότητα. Η σωστή IoT πλατφόρμα διαχείρισης οφείλει να είναι ευέλικτη και ευπροσάρμοστη, καθώς ο κόσμος του IoT είναι διαμοιρασμένος σε πολλά επιμέρους κομμάτια και συνεχώς μπορεί να αλλάζει, οπότε δεν είναι θεμιτό το βασικό στοιχείο του οικοσυστήματος να αποτελεί ασταθές κομμάτι της συνολικής υλοποίησης. Ακόμη, πρέπει να είναι επεκτάσιμη, ώστε το οικοσύστημα να μπορεί να εμπλουτιστεί και ασφαλής, για την εξασφάλιση της λειτουργίας χωρίς απειλές. Τέλος, έχουμε τους χρήστες που είναι όλοι οι άνθρωποι που χρησιμοποιούν και επηρεάζουν τα IoT οικοσυστήματα. Μπορεί συνεπώς να είναι οι μηχανικοί οι οποίοι τα κατασκευάζουν, οι διαχειριστές των πλατφορμών, και εν γένει ο κάθε ενδιαφερόμενος που θέλει να απολαύσει τις υπηρεσίες ενός IoT οικοσυστήματος. Όλα αυτά συνεπώς στοχεύουν ώστε να βελτιωθεί η αποδοτικότητα και η ποιότητα ζωής και είναι οι τελικοί χρήστες που ορίζουν πως θα χρησιμοποιήσουν όλες αυτές τις δυνατότητες για να επιτύχουν αποτελέσματα. Επομένως, λίγο πολύ αυτό είναι το πλαίσιο των IoT οικοσυστημάτων που βρίσκουν εφαρμογή σε διαφόρους τομείς ένας εκ των οποίων είναι και η γεωργία με σημαντική επιτυχία.

## 4.2 Image Processing and Classification

Το βασικό πρόβλημα που πραγματεύεται αυτή η εργασία όσον αφορά το image processing είναι η κατηγοριοποίηση φύλλων με βάση τις ασθένειες τους ή αν είναι υγιή. Σε μια προσπάθεια επέκτασης παλαιότερων κατηγοριοποιήσεων που επιλέγονταν 10-15 κλάσεις, αποφασίσαμε να υλοποιήσουμε τα πρόβλημα χρησιμοποιώντας 33 κλάσεις για τα φύλλα που αναλύθηκαν, προσπαθώντας παράλληλα να μην θυσιαστεί η ακρίβεια του μοντέλου. Το dataset που χρησιμοποιήθηκε είναι διαθέσιμο στο [41]. Το dataset δεν είναι balanced όσον αφορά το πλήθος των φωτογραφιών για αυτό απαιτείται μια διαδικασία pre-processing όπου αρχικά υπολογίσαμε το min(number of images in a class),το οποίο προέκυψε 152, και κατόπιν για κάθε κλάση κρατήσαμε το πολύ 3*min = 456 φωτογραφίες, καθώς υπήρχαν κλάσεις με 1000+ φωτογραφίες οπού θα χαλούσαν πάρα πολύ την ακρίβεια του μοντέλου μας, διότι η εκμάθηση θα βασιζόταν κυρίαρχα σε αυτές, ενώ πρέπει να υπάρχει μια ισορροπία. Η διαδικασία του pre-processing υλοποιήθηκε στο cloud με την βοήθεια του Google Drive. Ακολούθως, η διαδικασία του training έγινε με την βοήθεια του google colab και του εικονικού μηχανήματος (VM) που παρέχει για την ταχύτερη διεκπεραίωση της εκμάθησης, διότι αυτό που μας απασχολεί είναι το inference, δηλαδή το πόσο γρήγορα γίνεται το classification μιας καινούριας εικόνας και όχι τόσο η εκμάθηση δεδομένου ότι οι IoT συσκευές έχουν περιορισμένες δυνατότητες. Συνεπώς, στις IoT συσκευές μας φορτώναμε το trained μοντέλο για να υλοποιήσουμε το prediction/classification. Το σημαντικό κομμάτι όμως

που συσχετίζει την επεξεργασία εικόνας με το Διαδίκτυο των Αντικειμένων είναι η εκτέλεση του αλγορίθμου που αποφασίζει για την κατηγοριοποίηση νέων εικόνων στις IoT συσκευές μας (Raspberry Pi 3 / 4, Jetson Nano και Google Coral). Ο στόχος μας ήταν να υλοποιηθεί το μοντέλο με τρόπο τέτοιο ώστε να επαρκεί η μνήμη RAM και οι δυνατότητες των επεξεργαστών τους για την διεκπεραίωση με τις IoT συσκευές. Ταυτόχρονα, όπως θα υποδειχθεί στην υλοποίηση κατά την διαδικασία εκμάθησης, οι εικόνες που φορτώνονται κάθε φορά στο νευρωνικό δίκτυο περνάνε από ένα τυχαίο φιλτράρισμα, όπου επηρεάζει παραμέτρους όπως το range των χρωμάτων στην κλίμακα 0-255, την φωτεινότητα, το ζουμ και άλλα που θα φανούν με την ακριβής υλοποίηση. Η ιδέα είναι ότι σε μια ρεαλιστική IoT εφαρμογή οι εικόνες που θα φορτώνει ο χρήστης προς κατηγοριοποίηση δεν θα είναι τέλειες ούτε ολόιδιες με αυτές που έμαθε το μοντέλο μας, συνεπώς πρέπει να είναι μαθημένο να αντιμετωπίζει φωτογραφίες όπου υπάρχει περιστροφή ή οπού υπάρχει διαφορετικός φωτισμός κλπ.. Στις εφαρμογές του Διαδικτύου των Αντικειμένων εκτός από τα εξίσου αποδοτικά αποτελέσματα απαιτείται και η διαχείριση και η παρακολούθηση της κατανάλωσης των πόρων. Για τούτο τον λόγο αντί να τροφοδοτούμε τις IoT συσκευές μας απευθείας από την πρίζα, χρησιμοποιήσαμε ένα κουτί τροφοδοσίας που κατασκευάσθηκε από το εργαστήριο μας (εργαστήριο διάχυτης νοημοσύνης) και συγκεκριμένα από τον ερευνητή Γεώργιο Ρούτη, όπου μας έδινε μετρήσεις σχετικά με την τάση, το ρεύμα και την ισχύ με την βοήθεια της σειριακής θύρας του υπολογιστή μας. Δηλαδή, οι συσκευές ήταν συνδεδεμένες μέσω του κουτιού τροφοδοσίας στην παροχή και ταυτόχρονα το κουτί τροφοδοσίας έστελνε τα δεδομένα που μετρούσε μέσω της σειριακής θύρας σε έναν υπολογιστή. Στην Εικόνα: 17 βλέπουμε την συνολική διαδικασία που ακολουθήθηκε για την επίλυση του Image Classification [42], όπου στο 1$^{ο}$ κομμάτι έχουμε την προ-επεξεργασία των δεδομένων που θα χρειαστούν για την όλη διαδικασία. Το 2$^{ο}$ τμήμα αποτελεί την διαδικασία του training και του validation ενώ το 3$^{ο}$ υποδεικνύει την διαδικασία αξιολόγησης της απόδοσης του trained μοντέλου μέσω διαφόρων μετρικών που θα αναλυθούν κατά την εξήγηση της υλοποίησης.




*Εικόνα 18: Η υλοποίηση του image classification προβλήματος (Πηγή: [42])*

Ας δούμε λίγο και τον μαθηματικό φορμαλισμό. Το image classification πρόβλημα μπορεί να διατυπωθεί ως εξής:

*Δεδομένης μια άγνωστης συνάρτησης/απεικόνισης g: X → Y (the ground truth) η οποία αντιστοιχεί instances (στιγμιότυπα) εισόδου x ∈ X σε ετικέτες εξόδου y ∈ Y, καθώς τα δεδομένα προς εκμάθηση D = {($x_1$,$y_1$), … , ($x_n$,$y_n$)} αναπαριστούν τα ακριβή στιγμιότυπα αυτής της αντιστοίχισης, δημιουργούμε μια συνάρτηση/απεικόνιση h: X → Y, η οποία προσεγγίζει όσο πιο καλά γίνεται την σωστή αντιστοίχιση g.*

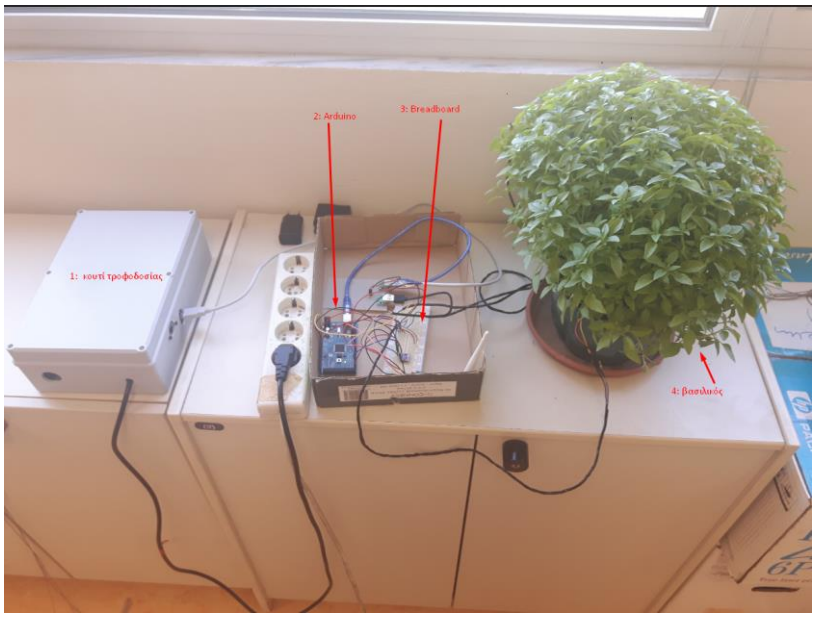
*Για να είναι ένα καλώς ορισμένο πρόβλημα πρέπει να οριστεί η φράση «προσεγγίζει όσο πιο καλά γίνεται» αυστηρά. Στην περίπτωση του Image Classification, η μετρική accuracy είναι συνηθώς αρκετή για το evaluation. Αυτή αντιστοιχεί απλά στην ανάθεση ενός 1 για κάθε ορθή κατηγοριοποίηση που υπονοεί ότι ο βέλτιστος classifier μεγιστοποιεί την ορθότητα σε ένα τυπικό τεστ δεδομένων και κατ' επέκταση σε ανεξάρτητα δεδομένα. Η ορθότητα/ακρίβεια*

*είναι το κλάσμα των στιγμιοτύπων (instances) που η συνάρτηση h: X → Y έθεσε την σωστή ετικέτα.*

## 4.3 Το δικό μας IoT οικοσύστημα

Οι εφαρμογές που άπτονται του Διαδικτύου των Αντικειμένων όπως παρατηρήσαμε και από την μελέτη διαφόρων δημοσιεύσεων, άρθρων και εργασιών βασίζονται σε μια ιδέα για την βελτίωση κάποιας εργασίας. Συνεπώς, κι εμείς στο εργαστήριο αποφασίσαμε να δημιουργήσουμε το δικό μας IoT οικοσύστημα και να το τεστάρουμε μέσα από πειράματα. Στόχος μας ήταν να παρέχουμε ένα λειτουργικό σύστημα που θα βοηθάει τον χρήστη στον έλεγχο του ποτίσματος ενός φυτού σε μια γλάστρα. Μπορεί να μην είναι ένα large-scale project διότι έγινε και στα πλαίσια ενός εργαστηρίου, ωστόσο είναι πολύ εύκολα επεκτάσιμο, αν αξιοποιήσουμε για παράδειγμα πολλές ομάδες των χρησιμοποιούμενων αισθητήρων για τον έλεγχο περισσότερων φυτών κλπ.. Το αρχικό πρόβλημα ήταν η απομακρυσμένη παρακολούθηση μετρικών όπως η θερμοκρασία, η υγρασία, η ακτινοβολία UV (η οποία για μας δείχνει ουσιαστικά την κατάσταση του ήλιου ως προς το φυτό μας) και την υγρασία εδάφους. Όλες αυτές οι μετρήσεις προφανώς για μας μπορεί να είναι λίγο πολύ τιμές και πειραματικά να αντιλαμβανόμαστε διάφορα πράγματα και να βγάζουμε συμπεράσματα, ωστόσο για έναν γεωργό μπορεί να σημαίνουν πολλά περισσότερα. Επομένως, προτείνουμε μια επέκταση της απλής παρακολούθησης των μετρικών, ένα irrigation mode, όπου ο χρήστης μπορεί ενεργοποιώντας το (θα δούμε πως στην υλοποίηση στο επόμενο κεφάλαιο) να λαμβάνει πολύ συχνότερες μετρήσεις από τους αισθητήρες την στιγμή που επιθυμεί να ποτίσει, ώστε να γνωρίζει πότε να σταματήσει καθώς ειδοποιείται με ένδειξη πότε οι πειραματικές μετρήσεις της υγρασίας του εδάφους πέσουν κάτω από το όριο που μετά από αρκετά πειράματα θέσαμε για το φυτό μας (βασιλικός). Η πραγματική δύναμη μιας IoT εφαρμογής φυσικά έγκειται στην δυνατότητα να ελέγχονται και να επεξεργάζονται απομακρυσμένα τα δεδομένα που λαμβάνονται. Συνεπώς, δημιουργήσαμε ένα IoT οικοσύστημα όπως φαίνεται και στις παρακάτω εικόνες, όπου οι αισθητήρες είναι συνδεδεμένοι στο Arduino μας, και ταυτόχρονα τοποθετημένοι στο φυτό για τις μετρήσεις θερμοκρασίας, υγρασίας και υγρασίας εδάφους (soil moisture) (ο ένας στην μια πλευρά του φυτού και ο άλλος στην άλλη), ενώ ο UV αισθητήρας βρίσκεται στο breadboard δίπλα στο φυτό. Το Arduino τροφοδοτείται μέσω του κουτιού τροφοδοσίας που κατασκευάσθηκε στο εργαστήριο μας (εργαστήριο διάχυτης νοημοσύνης) ώστε να ελέγχεται η κατανάλωση ρεύματος και ισχύος του Arduino. Τέλος, στο Arduino είναι συνδεδεμένο το Xbee module, το οποίο είναι υπεύθυνο για την ασύρματη μετάδοση των πληροφοριών που συλλέγουν οι αισθητήρες. Σε απομακρυσμένη θέση βρίσκεται το Raspberry Pi 4B, το οποίο λάμβανε τις μετρήσεις αυτές με την βοήθεια του Xbee module που λειτουργούσε ως δέκτης, τις αποθήκευε (σε μορφή CSV), τις παρουσίαζε live στην οθόνη μας και συνεπώς μας επέτρεπε και την μετέπειτα επεξεργασία των δεδομένων αλλά και την παρατήρησή τους σε πραγματικό χρόνο. Το κίνητρο μας λοιπόν είναι να παρουσιάσουμε την λειτουργία ενός small-scale IoT οικοσυστήματος για το πότισμα μιας γλάστρας, που

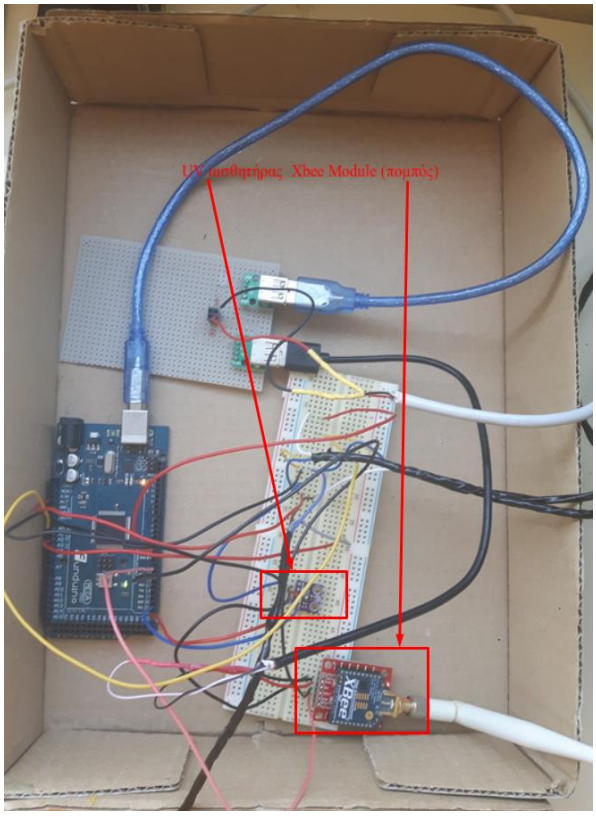
είναι εύκολα επεκτάσιμο για άρδευση περισσότερων φυτών, δίνει σημαντικές πληροφορίες στον χρήστη και ταυτόχρονα επιτρέπει την βελτιστοποίηση διαχείρισης πόρων. Το τελευταίο κομμάτι περί διαχείρισης πόρων μπορεί σε ένα small-scale πρόβλημα να φαντάζει λιγότερο σημαντικό, ωστόσο με την διεύρυνση και την επέκταση της εφαρμογής αντιλαμβανόμαστε ότι συντελεί κυρίαρχο ζήτημα. Η διαχείριση πόρων έγκειται στο irrigation mode όπως το αναφέραμε, όπου ο χρήστης λαμβάνει συχνότερες μετρήσεις οποτεδήποτε επιθυμεί να ποτίσει και ειδοποιείται μόλις οι μετρήσεις ξεπεράσουν το πειραματικό threshold όπου οι ανάγκες του φυτού σε νερό καλύφθηκαν. Προφανώς ο χρήστης ειδοποιείται με μήνυμα και στην αντίστροφη περίπτωση, όταν η ξηρασία του φυτού ξεπεράσει το άνω όριο, ενημερώνοντας τον ότι πρέπει να το ποτίσει. Να σημειωθεί εδώ ότι τα πειραματικά thresholds κατά την υλοποίηση εκτιμήθηκαν ύστερα από μια σειρά πειραμάτων με τις ορθότερες όσο γίνεται πρακτικές ακολουθώντας τις προτεινόμενες φορές ποτίσματος ανά ημέρα (1 για το φυτό μας (βασιλικός)) και τον σωστό όγκο νερού.



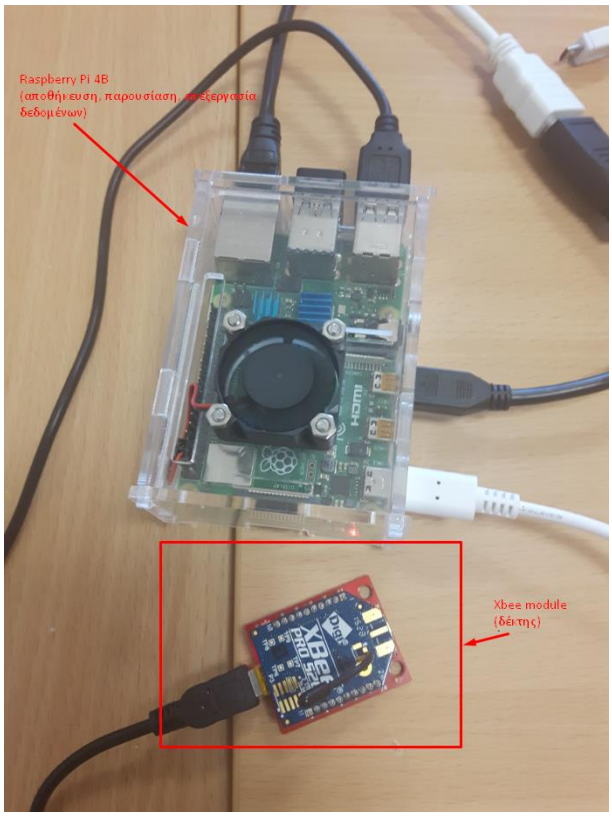

*Εικόνα 19: IoT οικοσύστημα*


*Εικόνα 20: Αισθητήρας θερμοκρασίας και υγρασίας*


*Εικόνα 21: Αισθητήρας υγρασίας εδάφους (x2)*




*Εικόνα 22: UV αισθητήρας και Xbee module (πομπός)*




*Εικόνα 23: Raspberry Pi 4B και Xbee module (δέκτης)*

# Κεφάλαιο 5 – Υλοποίηση

Στο κεφάλαιο αυτό γίνεται παρουσίαση των υλοποιήσεων μας καθώς και ανάλυση των διαφόρων βημάτων. Κώδικας θα προστίθεται όπου κρίνεται αναγκαίο και ίσως μη-τετριμμένο. Αρχικά, προηγείται η υλοποίηση για το Image Classification πρόβλημα και εν συνεχεία ακολουθεί η υλοποίηση της διασύνδεσης του IoT οικοσυστήματος.

## Κεφάλαιο 5.1 – Image Classification

Το πρώτο στάδιο για την υλοποίηση του Image Classification είναι το κομμάτι της προ-επεξεργασίας των δεδομένων του dataset, μια διαδικασία γνωστή ως data pre-processing. Αρχικά, στο dataset υπήρχαν 38 κλάσεις με ασθένειες φύλλων, ωστόσο ορισμένες είχανε πάρα πολύ λίγες φωτογραφίες ή ήταν μοναδικές για κάποιο φύλλο και συνεπώς τις βγάλαμε γιατί δεν θα παρείχαν ουσιαστική πληροφορία. Έτσι, κρατήσαμε 33 κλάσεις με ασθένειες φύλλων. Παρατηρήσαμε ότι η κλάση με τις λιγότερες φωτογραφίες περιείχε μόλις 152 εικόνες, ενώ υπήρχαν κλάσεις που περιείχαν παραπάνω από 2000 εικόνες. Αυτό θα οδηγούσε σε ιδιαίτερα εσφαλμένα αποτελέσματα αν δοκιμάζαμε να προχωρήσουμε στην διαδικασία του training έτσι διότι το dataset θα ήταν υπερβολικά unbalanced και δεν θα μπορούσε το μοντέλο μας να μάθει να κρίνει για κλάσεις με λίγες φωτογραφίες καθώς θα εστίαζε μόνο σε όσες είχαν πάρα πολλές φωτογραφίες συγκριτικά. Επομένως, για να προετοιμάσουμε καλύτερα το dataset ανεβάσαμε τα αρχεία στο google drive και με την βοήθεια ενός python script κρατήσαμε το πολύ τριπλάσιες φωτογραφίες για κάθε κλάση, με βάση τις λιγότερες φωτογραφίες που βρέθηκαν σε μια κλάση (152), άρα περιείχαν οι κατηγορίες το πολύ 3*152 = 456 εικόνες. Με την παρακάτω συνάρτηση (load_data) έγινε το διάβασμα όλων των δεδομένων και η διαδικασία επιλογής των εικόνων με βάση αυτό που αναλύθηκε παραπάνω.

```python
def load_data(data_directory):
    directories = []
    labels = []
    for directory in os.listdir(data_directory):
        if os.path.isdir(os.path.join(data_directory, directory)):
            directories.append(os.path.join(data_directory, directory))
            labels.append(directory)

    directories.sort()
    labels.sort()

    number = 3*min(len(os.listdir(directory)) for directory in directories)

    images = []
    final_labels = []
    for directory, label in tqdm(list(zip(directories, labels)), ncols=100):
        for filename in os.listdir(directory)[:number]:
            if filename.lower().endswith(".jpg"):
                name = os.path.join(directory, filename)
                images.append(name)
                final_labels.append(label)

    return images, final_labels
```

*Code Snippet 1: Προ-επεξεργασία Δεδομένων (Data Pre-Processing)*

Κατ' αυτόν τον τρόπο λοιπόν πετύχαμε την ισορροπία στο dataset μας, ώστε να είναι η βέλτιστη καθώς έγιναν δοκιμές ώστε να κρατήσουμε ίδιες εικόνες σε κάθε κατηγορία, διπλάσιες ή τετραπλάσιες. Ακολούθησε η διαδικασία του training, η οποία έγινε με την βοήθεια του google colab, εξού και η διαδικασία του preprocessing έγινε με τα δεδομένα στο google drive. Αρχικά, κατασκευάσαμε το μοντέλο του νευρωνικού δικτύου που χρησιμοποιήσαμε στο οποίο προφανώς υπήρξε fine-tuning ως προς τις παραμέτρους των layers. Στην ακόλουθη συνάρτηση (create_model) φαίνονται αναλυτικά τα layers του νευρωνικού. Τα επίπεδα dropout, τα οποία δεν έχουν αναλυθεί στην εισαγωγή, ουσιαστικά «πετούν» μέρος της γνώσης κατά την διαδικασία του training, ώστε να μειωθεί ο κίνδυνος του overfit, όπου το νευρωνικό δηλαδή μαθαίνει εξαιρετικά τα train δεδομένα, αλλά δεν μπορεί να διαχειριστεί νέες εικόνες και να είναι αποτελεσματικό.

```python
def create_model(shape, output):
    model = tf.keras.models.Sequential()
    model.add(tf.keras.layers.Conv2D(64, (3, 3), padding="same", strides=(2, 2), activation="relu", input_shape=shape))
    model.add(tf.keras.layers.MaxPooling2D((2, 2)))
    model.add(tf.keras.layers.Conv2D(128, (3, 3), padding="same", strides=(2, 2), activation="relu"))
    model.add(tf.keras.layers.MaxPooling2D((2, 2)))
    model.add(tf.keras.layers.Dropout(0.3))
    model.add(tf.keras.layers.Conv2D(256, (3, 3), padding="same", strides=(2, 2), activation="relu"))
    model.add(tf.keras.layers.MaxPooling2D((2, 2)))
    model.add(tf.keras.layers.Flatten())
    model.add(tf.keras.layers.Dense(1024))
    model.add(tf.keras.layers.Dropout(0.3))
    model.add(tf.keras.layers.Dense(256))
    model.add(tf.keras.layers.Dropout(0.3))
    model.add(tf.keras.layers.Dense(output, activation="softmax"))
    model.compile(optimizer='sgd', loss="categorical_crossentropy", metrics=["accuracy", tf.keras.metrics.Precision(), tf.keras.metrics.Recall()])
    model.summary()
    return model
```

*Code Snippet 2: Το CNN μοντέλο*

Η μετρική accuracy έχει αναλυθεί παραπάνω και είναι ουσιαστικά πόσες σωστές εκτιμήσεις έκανε ο αλγόριθμος για τα δεδομένα σε σχέση με όλες τις εκτιμήσεις. Οι

άλλες 2 μετρικές που χρησιμοποιήθηκαν είναι η Precision και η Recall, οι οποίες ορίζονται ως εξής και εν γένει επιθυμούμε να είναι όσο πιο κοντά στο 1 γίνεται:

Precision = $\frac{tp}{tp+fp}$ , όπου tp = true positive που είναι ισοδύναμο με hit και fp = false positive όπου είναι ισοδύναμο με false alarm

Recall = $\frac{tp}{tp+fn}$ , όπου fn = false negative ισοδυναμεί με miss

Η loss function που χρησιμοποιούμε είναι η κατηγορική εγκάρσια εντροπία (categorical_crossentropy) η οποία συνδυάζεται με την συνάρτηση softmax που επενεργεί στο τελευταίο dense layer. Χρησιμοποιώντας αυτή την loss function το εκπαιδευόμενο CNN δίνει στην έξοδο μια πιθανότητα για κάθε εικόνα να ανήκει σε καθεμιά από τις κατηγορίες (labels). Ωστόσο, όταν έχουμε πολλές κλάσεις να κατηγοριοποιήσουμε τότε τα labels είναι one-hot, δηλαδή μια εικόνα έχει 1 στο label που αντιστοιχεί και 0 σε όλα τα άλλα, και μόνο αυτή η τιμή χρησιμοποιείται για τον υπολογισμό του loss. Έτσι, ουσιαστικά εκφράζεται και η αυτοπεποίθηση που έχει ένα νευρωνικό ως προς την ακρίβεια, δεδομένου ότι το dataset είναι balanced.

Ακολούθως, επειδή είχαμε εν τέλει να κάνουμε με IoT συσκευές θέλαμε έναν τρόπο να περιορίσουμε την χρήση της μνήμης, καθώς απαιτούσε πάρα πολύ RAM αν φορτώναμε όλες τις εικόνες και αρχίζαμε την επεξεργασία τους. Για να αντιμετωπιστεί αυτό το πρόβλημα και ταυτόχρονα να δοθεί μια IoT πινελιά κατά την διαδικασία του training αξιοποιήσαμε τις δυνατότητες των generators και ειδικότερα του ImageDataGenerator. Για να προσδώσουμε αυτή την IoT αίσθηση κατά την διαδικασία του training προβήκαμε σε data augmentation «αυξάνοντας» τα δεδομένα που είχαμε, χωρίς να αυξήσουμε το πλήθος τους. Αυτό μπορεί να φαίνεται παράλογο αλλά δεν είναι. Σε κάθε εποχή του training μετά την πρώτη οι εικόνες που φορτώναμε δεχόντουσαν τυχαίους μετασχηματισμούς, αντίστοιχα όπως και μια πραγματική εικόνα μπορεί να είναι οριζόντια ή κάθετα γυρισμένη, να έχει κλίση, να διαφέρει ως προς την φωτεινότητα, την απόχρωση των χρωμάτων (channel_shift_range), τη μεγέθυνση κ.ά.. Δεν επιθυμούσαμε λοιπόν το μοντέλο μας να μάθει μονάχα τις καλοτραβηγμένες φωτογραφίες, διότι στην ρεαλιστική περίπτωση θα δυσκολευόταν, ειδικά δεδομένου ότι είχαμε λίγες εικόνες ανά κλάση. Συνεπώς, με αυτή την τεχνική αυξήσαμε τα δεδομένα μας αξιοποιώντας όσες εικόνες είχαμε.

```
train_datagen = tf.keras.preprocessing.image.ImageDataGenerator(
    rescale=1./255,
    horizontal_flip=True,
    vertical_flip=True,
    rotation_range=45,
    brightness_range=[0.9, 1.1],
    channel_shift_range=20,
    zoom_range=0.15,
    shear_range=0.1,
    fill_mode='nearest',
)
```

*Code Snippet 3: Αύξηση Δεδομένων (Data Augmentation) και κατασκευή generator*

Εν συνεχεία, αξιοποιήσαμε την ικανότητα των generators να μην χρησιμοποιούν μεγάλο μέρος της μνήμης καθώς φορτώνουν στο νευρωνικό τις φωτογραφίες σε batches, δηλαδή σε μικρότερες ομάδες φωτογραφιών (π.χ. ανά 8, 16, 32 κλπ.) και αφού γίνει η σχετική επεξεργασία των εικόνων να τις αποδεσμεύει από την μνήμη, έχοντας κρατήσει ωστόσο cache μνήμη. Αποτελούν συνεπώς, ένα πολύ ευέλικτο εργαλείο, διότι φορτώνοντας ολόκληρο το dataset παρατηρήσαμε ότι απαιτούνταν περισσότερα από 26GB RAM ενώ με την χρήση των generators λιγότερα από 2 GB RAM.

```
train_generator = train_datagen.flow_from_directory(
    directory="/content/drive/My Drive/leafnet_thesis/leaf_train_v3",
    target_size=(256, 256),
    color_mode="rgb",
    batch_size=batch_size,
    class_mode="categorical",
    shuffle=True,
    seed=42,
)
```

*Code Snippet 4: Η κλήση του generator*

Τέλος, για να κλείσει η διαδικασία του training κάναμε fit στο μοντέλο μας παρακολουθώντας το validation loss, δηλαδή το loss για το validation set, που όπως αναλύσαμε παραπάνω χρησιμοποιείται για την εκτίμηση της εκπαίδευσης του νευρωνικού δικτύου και του μοντέλου μας. Τα αποτελέσματα που λάβαμε σε βάθος 27 εποχών ήταν τα εξής:
val_loss: 0.3256 - val_accuracy: 0.9001 - val_precision: 0.9235 - val_recall: 0.8596,
τα οποία κρίναμε πολύ καλά δεδομένου ότι κληθήκαμε να κάνουμε κατηγοριοποίηση μεταξύ 33 κλάσεων που είναι πολλές.

Τέλος, για το κομμάτι του prediction χρησιμοποιήσαμε 2 τρόπους φόρτωσης των δεδομένων, είτε με την βοήθεια της βιβλιοθήκης Pillow που φορτώναμε μία-μία τις εικόνες για να ακολουθήσει η διαδικασία της πρόβλεψης από το trained μοντέλο μας είτε πάλι αξιοποιώντας τις δυνατότητες του ImageDataGenerator όπο φορτώναμε ανά batch τις εικόνες για να γίνει το classification. Η ανάλυση για τους καταναλωθέντες πόρους και τον χρόνο ανά περίπτωση, θα εξηγηθούν στο επόμενο κεφάλαιο, αν και είναι σαφές ότι η χρήση των generators επιτάχυναν αισθητά και την διαδικασία του prediction. Αξίζει να σημειωθεί εδώ, ότι το Google Coral δεν υποστηρίζει το TensorFlow, αλλά υποστηρίζει το TensorFlow lite, οπότε μετατρέψαμε το μοντέλο σε tflite και ακολούθησε η διαδικασία predict με βάση το trained μοντέλο που φυσικά φορτώσαμε σε όλες τις συσκευές που χρησιμοποιήσαμε και τα αποτελέσματα θα δειχθούν στο επόμενο κεφάλαιο.

## 5.3 Arduino και IoT οικοσύστημα

Αρχικά, η πρώτη εργασία μας κατά την υλοποίηση των πειραμάτων με το Arduino και τους αισθητήρες ήταν η σύνδεση τους με την βοήθεια του breadboard στα αντίστοιχα pins του Arduino. Ακολούθως, μέσω του Arduino IDE και με τις κατάλληλες εντολές επιτεύχθηκε η real-time λήψη των διαφόρων μετρήσεων μέσω της

σειριακής θύρας του υπολογιστή μας. Ωστόσο, ο σκοπός φυσικά ήταν η απομακρυσμένη παρακολούθηση των διαφόρων μετρικών. Συνδέοντας λοιπόν τα Xbee modules στο Arduino ως πομπός και στο Raspberry Pi 4B ως δέκτης και συγχρονίζοντας τα baud rate για να υπάρχει η δυνατότητα αυτής της ζεύξης μπορούσαμε πλέον να λάβουμε τα δεδομένα στο RPi 4B μέσω της σειριακής του θύρας απομακρυσμένα. Το script που χρησιμοποιήθηκε στην πλευρά του RPi για την υλοποίηση αυτής της εργασίας είναι το εξής:

```
import time
import serial

ser = serial.Serial(
        port='/dev/ttyUSB0',
        baudrate = 38400,
        parity=serial.PARITY_NONE,
        stopbits=serial.STOPBITS_ONE,
        bytesize=serial.EIGHTBITS,
        timeout=None
)

print("Soil_Moisture_1, Soil_Moisture_2, DHT_Temperature, DHT_Humidity,
 UV_light_level")

while 1:
    x=ser.readline()
    x=x.decode().strip()
    print(x)
```

*Code Snippet 5: Python script για την λήψη δεδομένων μέσω σειριακής στο RPi 4*

Συνεπώς, όσο έτρεχε το Arduino και παρήγαγε μετρήσεις μέσω των αισθητήρων εμείς τις καταγράφαμε και τις παρακολουθούσαμε μέσω του terminal του RPi.
Ο κώδικας του Arduino για την ανάγνωση των τιμών των αισθητήρων δεν θα παρουσιαστεί, ωστόσο θα δείξουμε την υλοποίηση του Irrigation Mode κατά το οποίο ο χρήστης στέλνοντας τον χαρακτήρα ‘t’ στην σειριακή του Arduino ενεργοποιεί αυτό το mode και πλέον οι μετρήσεις λαμβάνονται πολύ πιο γρήγορα ανάλογα με το delay που έχει επιλεχθεί π.χ. 1 δευτερόλεπτο, ώστε να πάει ο χρήστης να ποτίσει ή να θέσει το αυτόματο πότισμα που αναγνωρίζει τις τιμές του soil moisture (αυτό αποτελεί future work) και να ενημερωθεί μόλις η τιμή των soil moisture αισθητήρων πέσει κάτω από το σχετικό threshold όπου θα ενημερωθεί με μήνυμα “Irrigation Completed”. Αντίστοιχα, μήνυμα λαμβάνει και όταν η τιμή ξεπερνά το άνω όριο δεχόμενος το μήνυμα “water needed”. Τέλος, στέλνοντας τον χαρακτήρα ‘x’ στην σειριακή του Arduino ο χρήστης απενεργοποιεί το Irrigation Mode και οι μετρήσεις λαμβάνονται στον αρχικό τους ρυθμό ανά 15 λεπτά, ώστε να παρέχεται επαρκής πληροφορία μεν, χωρίς να είναι υπερβολική και χαοτική δε. Έτσι, ο εκάστοτε γεωργός μπορεί να ελέγχει τις περιβαλλοντικές τιμές ανά τακτά χρονικά διαστήματα και όποτε επιθυμεί να ποτίσει να έχει ένα μέσο που θα τον ενημερώνει ώστε να μην σπαταλά άσκοπα νερό ή να κάνει ζημιά στο φυτό. Κατ’ αυτόν τον τρόπο θεωρούμε ότι η εφαρμογή μας βοηθά στην βελτιστοποίηση της διαχείρισης των πόρων.

```
int avg = (soil_1 + soil_2) / 2;
if (avg > 390) {
  Serial.println("Water needed");
  Serial.flush()
}

if (Serial.available() > 0) {
  char r = Serial.read();
  if (r == 't') {
    delayMore = 0;
    Serial.println("Irrigation Mode ON");
    Serial.flush();
    if (avg < 320 && irrigated == 0) {
      Serial.println("Irrigation completed");
      Serial.flush();
      irrigated = 1;
    }
  }
  else if (r == 'x') {
    delayMore = 1;
    irrigated = 0;
    Serial.println("Irrigation Mode OFF");
    Serial.flush()
  }
}
```

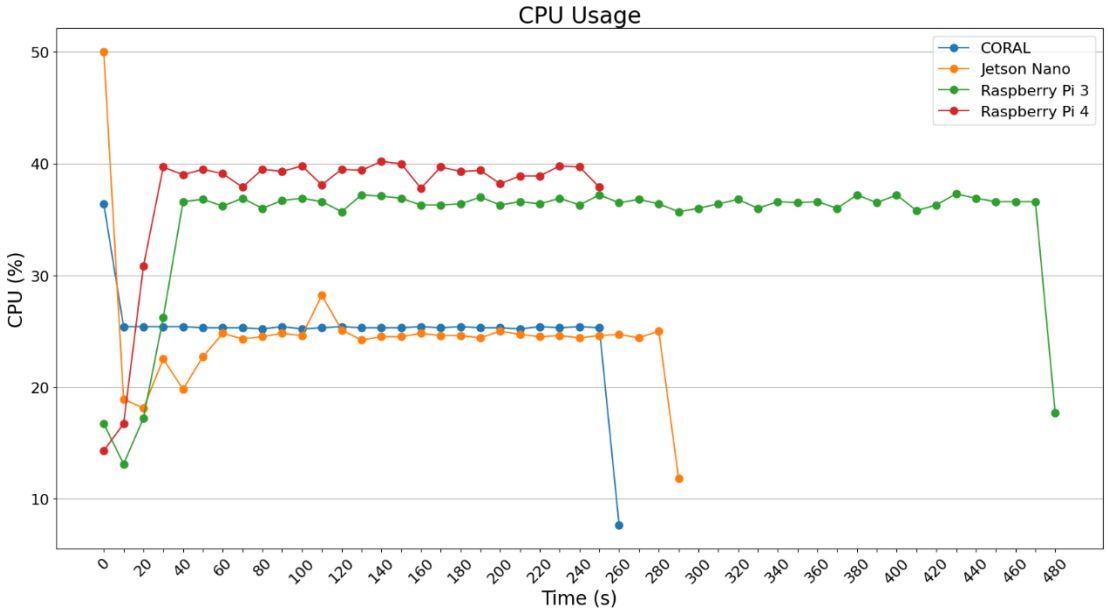
*Code Snippet 6: Arduino Script για τον έλεγχο του Irrigation Mode*

# Κεφάλαιο 6 – Πειραματική Αξιολόγηση

Στο κεφάλαιο αυτό θα παρουσιαστούν τα πειραματικά αποτελέσματα και των δύο εφαρμογών. Για το image classification κομμάτι θα δούμε σε γραφικές κυρίως IoT μετρικές που έχουν να κάνουν με την κατανάλωση ρεύματος σε κάθε συσκευή και την χρήση CPU, μνήμης και εσωτερικών θερμοκρασιών, παρατηρώντας ταυτόχρονα από τα διαγράμματα τον χρόνο εκτέλεσης για την εκάστοτε συσκευή και παραθέτοντας ορισμένα scores. Για το κομμάτι του Arduino και των αισθητήρων θα παρουσιαστούν 2 διαγράμματα, τα οποία αντιστοιχούν στα 2 σχετικά πειράματα που έγιναν και κρίναμε καλύτερα για παρουσίαση. Το πρώτο πείραμα αφορά το Irrigation Mode, όπου είχαμε μετρήσεις κάθε 1 λεπτό για σύνολο 40 λεπτών, και το δεύτερο πείραμα που διήρκεσε περίπου 26 ώρες με μετρήσεις ανά 15 λεπτά και παρουσιάζεται ο κύκλος νύχτας-ημέρας του φυτού μας.

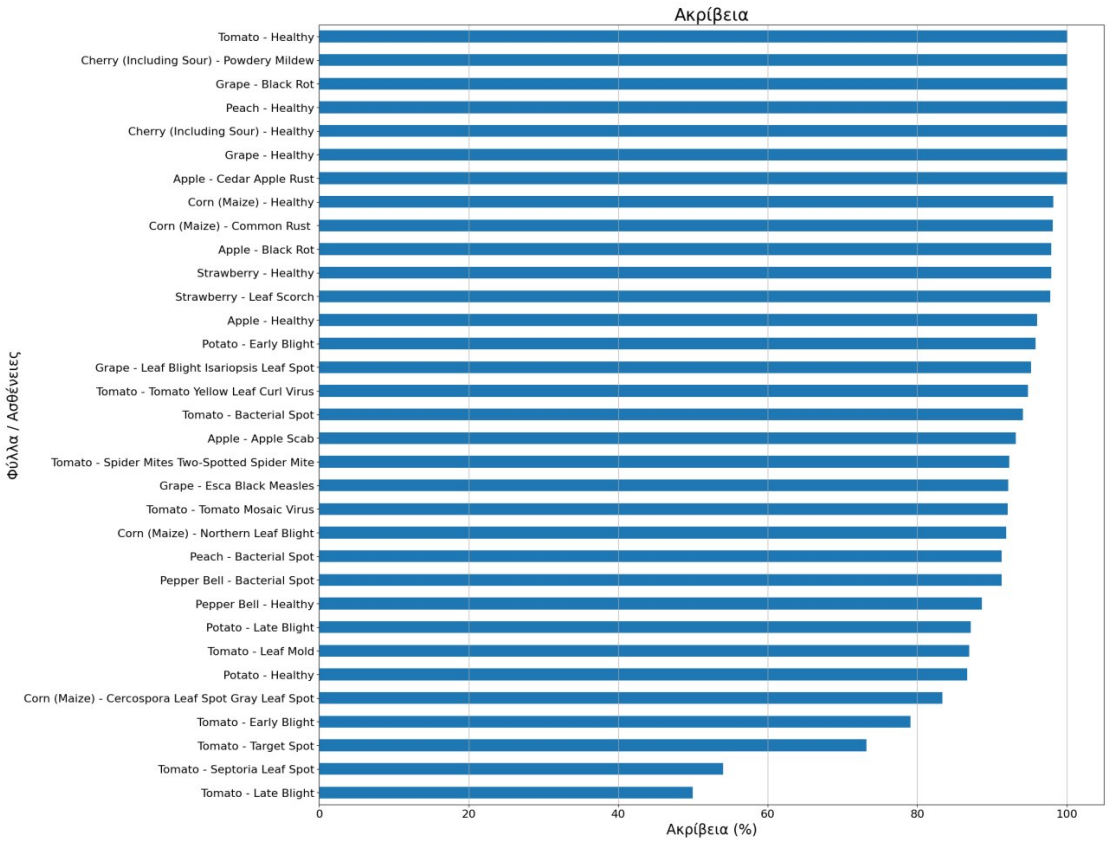
## 6.1 Image Classification

Όπως προείπαμε, για την διαδικασία του Image Classification στις IoT συσκευές μας χρησιμοποιήσαμε αρχικά την βιβλιοθήκη Pillow της python για να φορτώνουμε μία-μία τις φωτογραφίες εξοικονομώντας έτσι μνήμη RAM, αλλά καθιστώντας πιο χρονοβόρα την διαδικασία του inference, το οποίο αναφέρεται στην διαδικασία της χρήσης ενός εκπαιδευμένου αλγορίθμου μηχανικής μάθησης ώστε να κάνει μια πρόβλεψη. Οι βασικές μετρικές που θα δούμε εδώ για την εποπτική σύγκριση μεταξύ όλων των συσκευών μας είναι η χρήση της CPU, της μνήμης RAM ποσοστιαία και σε bytes καθώς το Raspberry Pi 4B και το Jetson Nano έχουν 4 GB μνήμης RAM, της μνήμης Swap και της εσωτερικής θερμοκρασίας τους. Αντίστοιχα, θα παρουσιάσουμε και το ρεύμα που κατανάλωνε η κάθε συσκευή. Οφείλουμε να αναφέρουμε σε αυτό το σημείο ότι όλες οι μετρήσεις για την επεξεργαστική δύναμη (CPU Usage), για την μνήμη RAM, για την μνήμη Swap και για την εσωτερική θερμοκρασία των συσκευών υλοποιήθηκαν με την βοήθεια ενός απλού python script που χρησιμοποιεί την βιβλιοθήκη την python psutil, η οποία μας παρέχει όλες αυτές τις μετρήσεις και τα δεδομένα σε επεξεργάσιμη και αποθηκεύσιμη μορφή όπως το csv. Για την μέτρηση των καταναλώσεων κάθε συσκευής (ρεύματος και ισχύος) όπου παρουσιάζεται η κατανάλωση ρεύματος και επεξηγούμε παρακάτω ότι λόγω της αναλογικής τους σχέσης, όποιο από τα 2 και να παρουσιαστεί προκύπτουν τα ίδια ακριβώς συμπεράσματα. Η συγκεκριμένη μέτρηση υλοποιήθηκε συνδέοντας το κουτί τροφοδοσίας που κατασκευάσθηκε στο εργαστήριο μας (Εργαστήριο Διάχυτης Νοημοσύνης) μέσω του οποίου μπορούμε να καταγράφουμε τις τιμές τάσεις, ρεύματος και ισχύος ανά τις χρονικές στιγμές που επιθυμούμε (εδώ 10 δευτερόλεπτα), ώστε να έχουμε μια εποπτική εικόνα των ενεργειακών πόρων που χρησιμοποιούνται, καθώς οι εφαρμογές του Διαδικτύου των Αντικειμένων βασίζονται στην χαμηλή αυτή κατανάλωση.




*Διάγραμμα 1: CPU Usage (Pillow)*

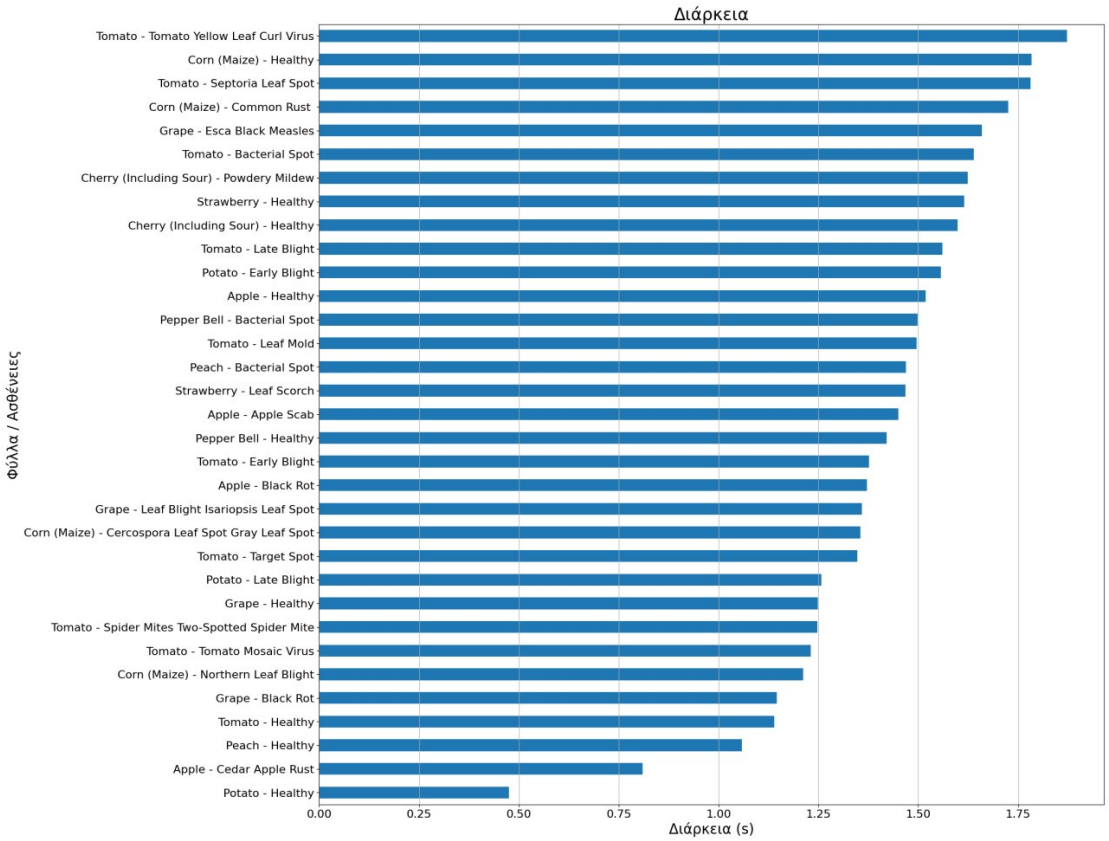
Από το παραπάνω διάγραμμα, παρατηρούμε το αναμενόμενο ότι όσο πιο δυνατή είναι μια συσκευή τόσο περισσότερο μπορεί να χρησιμοποιήσει και να αξιοποιήσει τις δυνατότητες της. Ενδιαφέρον σχολιασμού προκαλεί εδώ, ότι το Google Coral με τους ML accelerators κατάφερε να πετύχει τον ίδιο συνολικό χρόνο για το inference με το Raspberry Pi 4B που όπως φαίνεται ήταν και η ισχυρότερη και ταχύτερη συσκευή. Για το Jetson Nano παρατηρούμε ότι στην αρχή υπάρχει πολύ έντονη χρήση της επεξεργαστικής ισχύος που μετά μειώνεται, κάτι που είναι φυσιολογικό διότι το Jetson Nano φορτώνει δυναμικά την βιβλιοθήκη TensorFlow στην αρχή κάθε εκτέλεσης του αλγορίθμου και συνεπώς καταναλώνει λίγο παραπάνω χρόνο στην αρχή και αρκετή υπολογιστική δύναμη. Να σημειώσουμε σε αυτό το σημείο ότι η διαδικασία του classification/prediction υλοποιήθηκε με το ίδιο dataset σε όλες τις συσκευές το οποίο προέκυψε, κρατώντας το 20% του αρχικού πλήθους φωτογραφιών για κάθε κατηγορία και προφανώς τα αποτέλεσμα που προκύπταν ως τιμές ήταν σχεδόν ίδια. Ας παρουσιάσουμε τα αποτελέσματα με την βοήθεια του Google Colab σε αυτή την περίπτωση καθώς προφανώς και οι άλλες συσκευές εκτιμούν τις ίδιες τιμές, απλώς σε διαφορετικό χρόνο.




*Διάγραμμα 2: Ακρίβεια ανά κλάση*

Από το διάγραμμα 1 παρατηρούμε ότι το Raspberry Pi 4B έκανε περίπου 244 sec και πολύ κοντά χρονικά βρέθηκε το Google Coral με 253 sec και το Jetson Nano με 265 sec περίπου, ενώ το σαφώς πιο αδύναμο Rpi 3B+ χρειάστηκε λίγο παραπάνω από 453 sec. Αξιοσημείωτο είναι ότι παρότι το μοντέλο μας έχει ένα μέσο όρο ακρίβειας στο prediction της τάξης του 90% υπάρχουν δύο κλάσεις (Tomato Septoria Leaf Spot και Tomato Late Blight) που έχουν score περίπου 50%. Μελετώντας αναλυτικότερα τις φωτογραφίες μεταξύ των δύο αυτών κατηγοριών και αναλύοντας τα αποτελέσματα ανά φωτογραφία, παρατηρήσαμε ότι τα λάθη προκύπτουν διότι οι εικόνες είναι πολύ πανομοιότυπες και είναι δύσκολη η εξαγωγή ξεχωριστών χαρακτηριστικών καθώς και η κλάση Tomato Early Blight είχε παρόμοιες φωτογραφίες και γι’ αυτό τον λόγο είχε ακρίβεια 79% που είναι σαφώς χαμηλότερη από το 90% που είναι η μέση ακρίβεια. Δυστυχώς, παρά το data augmentation και τις σχετικές προσπάθειες μας, τέτοια ζητήματα δεν μπορούν εύκολα να αποφευχθούν έχοντας λίγες εικόνες για την

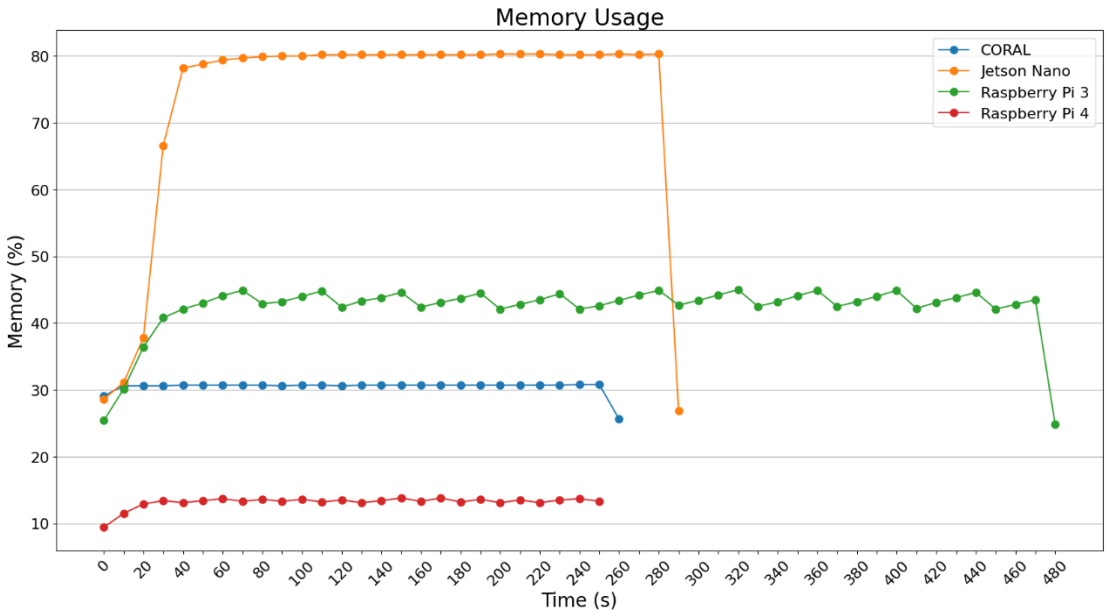
διαδικασία εκμάθησης και δεδομένου ότι κατηγοριοποιούσαμε μεταξύ 33 ετικετών, χωρίς να ενώσουμε παρόμοιες κατηγορίες.



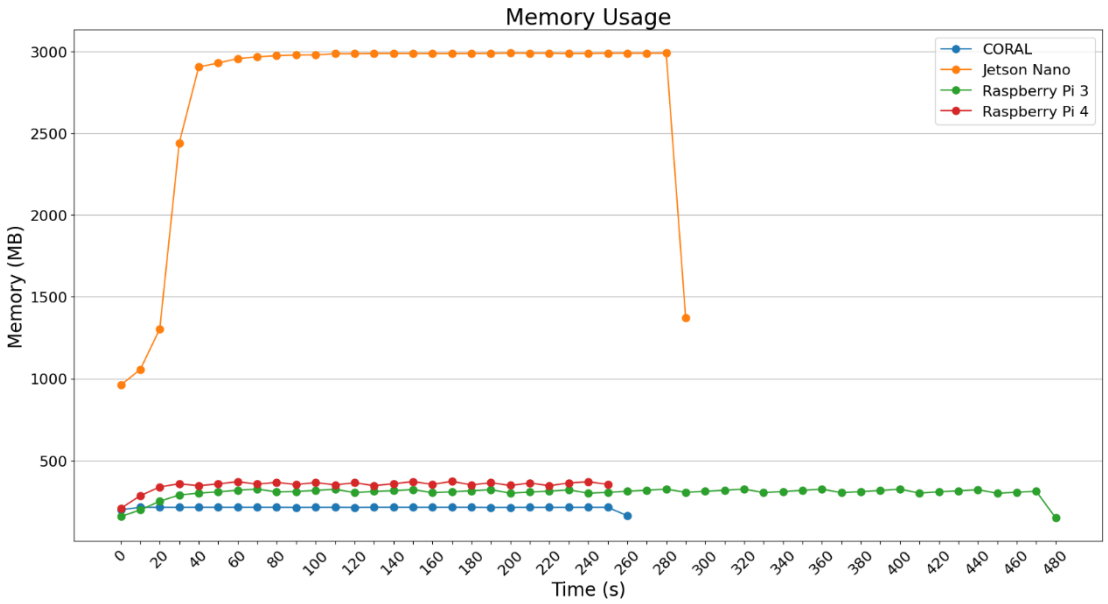

*Διάγραμμα 3: Χρόνος inference ανά κλάση με Google Colab*

Στο διάγραμμα 3 παρουσιάζεται ο χρόνος επεξεργασίας για κάθε κλάση μέσω του Google Colab. Πλήρως ανάλογοι χρόνοι προκύπτουν και με τις διάφορες συσκευές απλώς στα RPi 4, Jetson Nano και Google Colab ο χρόνος ανά κλάση είναι περίπου 5-πλάσιος, ενώ στο RPi3 μια τάξη μεγέθους πάνω σε σχέση με το Google Colab δηλαδή 10-πλάσιος. Αυτό που ωστόσο έχει πραγματικό νόημα είναι ο συνολικός χρόνος για την υλοποίηση της διαδικασίας, καθώς στην πραγματική χρήση της εφαρμογής/αλγορίθμου θα δοθεί ένα dataset εικόνων που δεν θα είναι κατηγοριοποιημένες. Οι μικρές χρονικές διαφορές ανά κλάση έχουν να κάνουν με το πλήθος των φωτογραφιών καθώς δεν ήταν ακριβώς ίσες αλλά ± 5 ανά κλάση, και το αποτέλεσμα παρουσιάζεται καθαρά για την σύγκριση του πόσο ταχύτερο είναι το Google Colab, το οποίο φυσικά δεν αποτελεί μια πραγματική συσκευή αλλά ένα πανίσχυρο VM (Virtual Machine).

Ακολουθούν τα διαγράμματα χρήσης της μνήμης RAM και των εσωτερικών θερμοκρασιών των συσκευών.



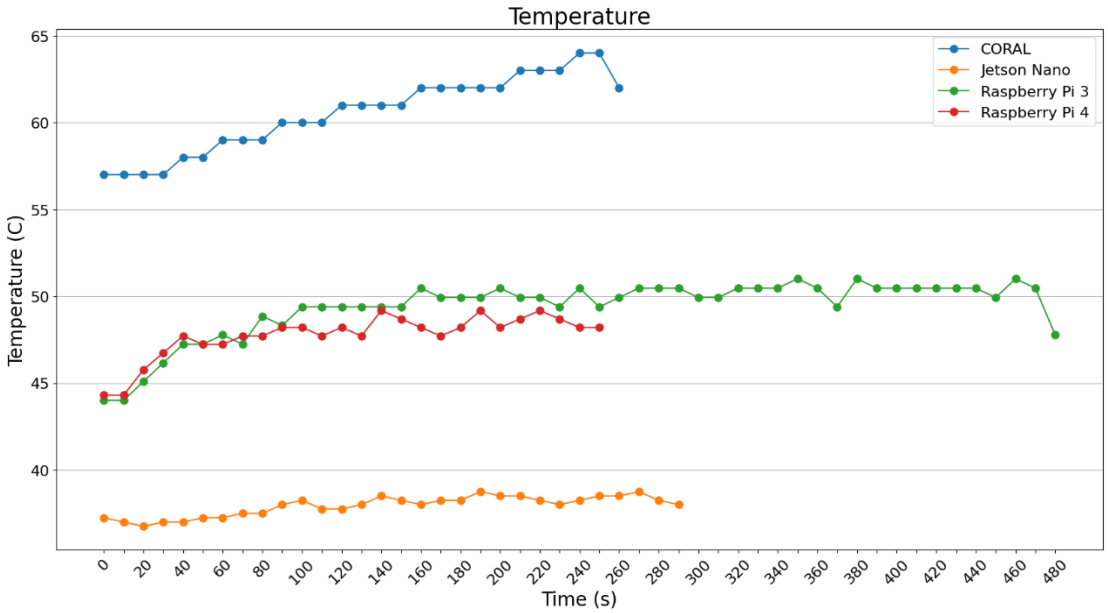

*Διάγραμμα 4: Memory Usage (Percentage) (Pillow)*




*Διάγραμμα 5: Memory Usage (MBytes) (Pillow)*

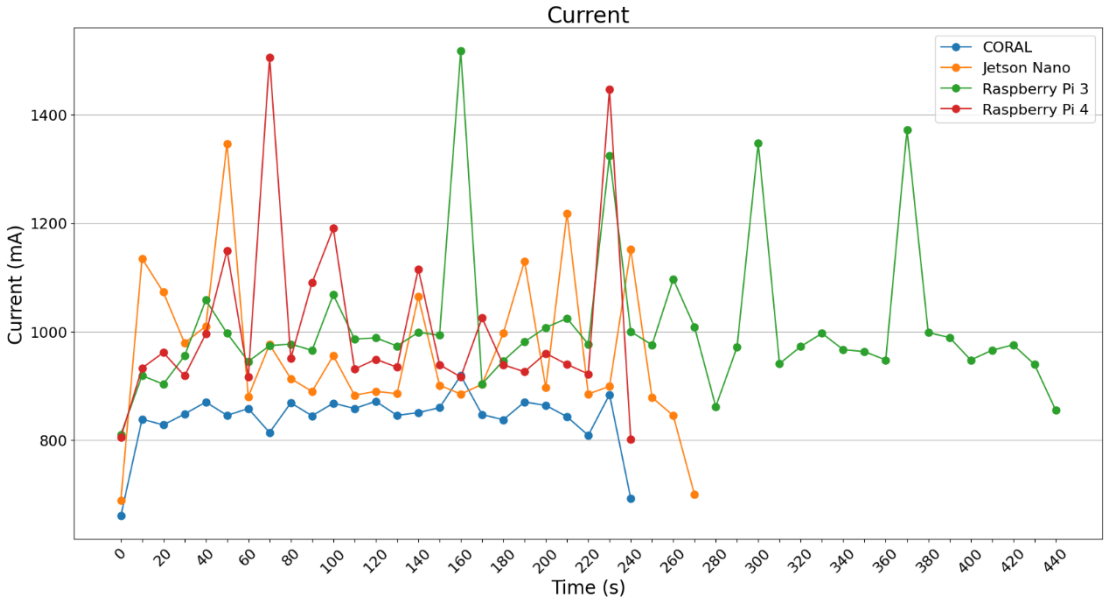
Αυτό που κρίνεται αξιοσημείωτο εδώ είναι η πολύ υψηλή χρήση του Jetson Nano σε μνήμη RAM, ενώ οι άλλες συσκευές βρίσκονται αρκετά χαμηλά κάτω από τα 500 MB και μην ξεπερνώντας το 50% για το Rpi3, στο 30% περίπου για το Google Coral ενώ το Raspberry Pi 4 έδειξε να μην ζορίζεται χρησιμοποιώντας μόλις το 15% της μνήμης RAM του. Το αποτέλεσμα που εκ πρώτης όψεως φαίνεται παράλογο, είναι απόλυτα φυσιολογικό, διότι το Jetson Nano χρησιμοποιεί την GPU του για την εκτέλεση των διαφόρων πράξεων κάτι που είναι πολύ περισσότερο ενεργοβόρο από άποψη μνήμης

καθώς η GPU χρησιμοποιεί ως buffer μνήμης την μνήμη RAM. Ας δούμε τώρα και την εσωτερική θερμοκρασία των συσκευών.




*Διάγραμμα 6: Εσωτερική Θερμοκρασία Συσκευής (Pillow)*

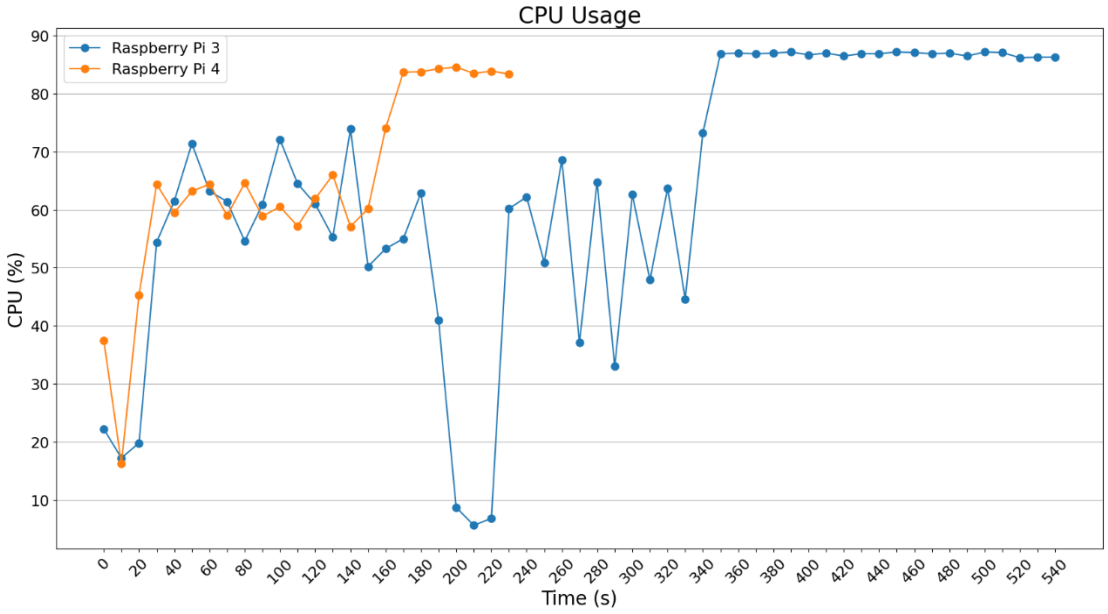
Το Jetson Nano και σε αυτή την περίπτωση χρίζει σχολιασμού μαζί με το Google Coral εν αντιθέσει με τα 2 Raspberry Pi που κινήθηκαν σε φυσιολογικές θερμοκρασίες. Το Jetson Nano παρουσίασε πολύ χαμηλές θερμοκρασίες κάτι που οφείλεται στον «τεράστιο» ανεμιστήρα που έχει από πάνω συγκριτικά με το μέγεθος του, ο οποίος ψύχει την πλακέτα και την GPU που έχει. Το Google Coral από την άλλη παρουσίασε ιδιαίτερα υψηλές φαινομενικά θερμοκρασίες, ωστόσο φαινόταν να το διαχειρίζεται καθώς ο ανεμιστήρας του δεν τέθηκε σε λειτουργία, διότι ενεργοποιείται όταν κρίνει ότι ξεπερνάει τις οριακές τιμές που του έχουν θέσει οι κατασκευαστές. Ακολούθως, παρακολουθήσαμε την κατανάλωση ρεύματος των συσκευών. Να σημειώσουμε ότι βρίσκονται υπό σταθερή τάση 5.1 V DC, οπότε παρότι λαμβάναμε μετρήσεις τάσης, ρεύματος και ισχύος, στα διαγράμματα θα παρουσιαστεί μόνο το ρεύμα, διότι προφανώς η ισχύς μπορεί να προκύψει πολύ εύκολα από την πράξη: $P(Power\ (mW)) = 5.1\ (Volt) * I\ (Current\ (mA))$




*Διάγραμμα 7: Μετρική Ρεύματος (mA) (Pillow)*

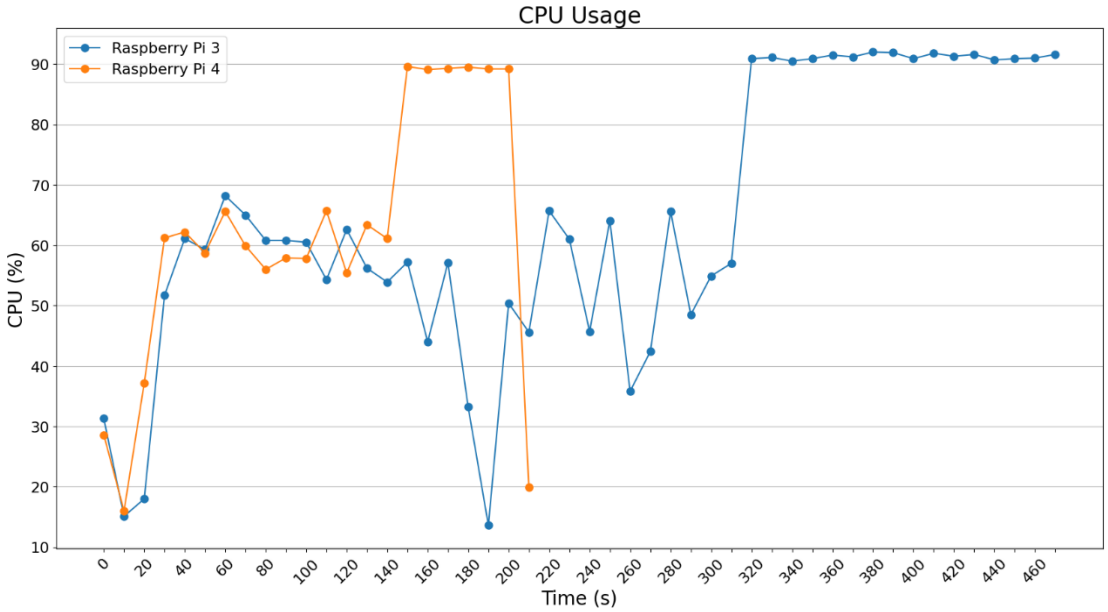
Εν γένει για όλες τις συσκευές πλην του Google Coral παρατηρούμε μια μέση τιμή γύρω στα 1000 mA με ορισμένες ακραίες τιμές πάνω από 1350 mA και τιμές ηρεμίας στα 800 mA και κάτω. Το Google Coral λόγω της αρχιτεκτονικής του επιτυγχάνει πολύ σταθερές τιμές κατανάλωσης ρεύματος και ισχύος, σαφώς χαμηλότερες από τις αντίστοιχες των άλλων συσκευών. Ας περάσουμε στο κομμάτι όπου θα δούμε και τα περισσότερα διαγράμματα και αποτελέσματα. Η χρήση των generators εκτός από την διαδικασία του training, επεκτείνεται φυσικά και στην διαδικασία του prediction όπου με την βοήθεια τους φορτώνονται οι εικόνες ανά batches στο μοντέλο και συνεπώς επεξεργάζονται σε ομάδες των 2, 4, 8, 16, 32 στην δική μας περίπτωση. Για την διενέργηση αυτού του πειράματος επιλέξαμε τα 2 Raspberry Pi, καθώς το Google Coral δεν υποστηρίζει το κανονικό TensorFlow και δεν ήταν δυνατό να χρησιμοποιηθεί ο ImageDataGenerator, ο οποίος ανήκει στο Keras. Το Keras είναι μια βιβλιοθήκη ανοιχτού κώδικα που παρέχει μια διεπαφή Python για νευρωνικά δίκτυα και λειτουργεί ως διεπαφή για την βιβλιοθήκη TensorFlow. Δυστυχώς, ούτε το Jetson Nano στην έκδοση λειτουργικού μπορούσε να υποστηρίξει αυτή την διαδικασία, επομένως κρατήσαμε μόνο τα 2 Raspberry Pi, με την βοήθεια των οποίων θα δούμε και τις αντίστοιχες μετρήσεις.

Τα πειράματα έγιναν με batch_size = 2, 4 ,8 ,16 και μόνο για το Raspberry Pi 4B+ δοκιμάστηκε και batch_size = 32 που επέφερε βελτίωση στον χρόνο εκτέλεσης. Εν γένει, αυτό που γίνεται αντιληπτό από τα επόμενα διαγράμματα είναι ότι όσο αυξάνεται το batch_size τόσο μειώνεται ο χρόνος εκτέλεσης ενώ ταυτόχρονα απαιτούνται περισσότεροι υπολογιστικοί πόροι, διότι ζορίζονται οι συσκευές. Αρχικά, παρουσιάζεται το διαγράμμα για batch size = 2.



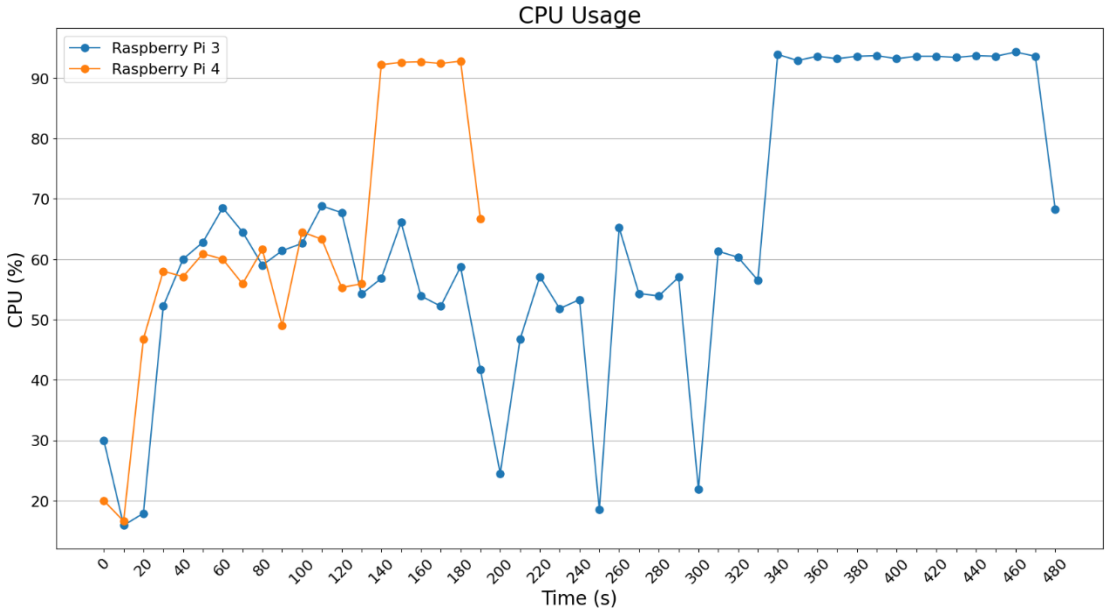

*Διάγραμμα 8: CPU Usage (ImageDataGenerator - Batch size = 2)*

Για την εκτέλεση με batch size = 2, δεν προκύπτει αρχικά κάποιο αξιοσημείωτο δεδομένο παρά μόνο ότι το RPi 4, μπορεί και υλοποιεί σαφώς πιο γρήγορο και σχετικά πιο ξεκούραστα. Ακολουθεί, το διάγραμμα χρήσης CPU με batch size = 4.



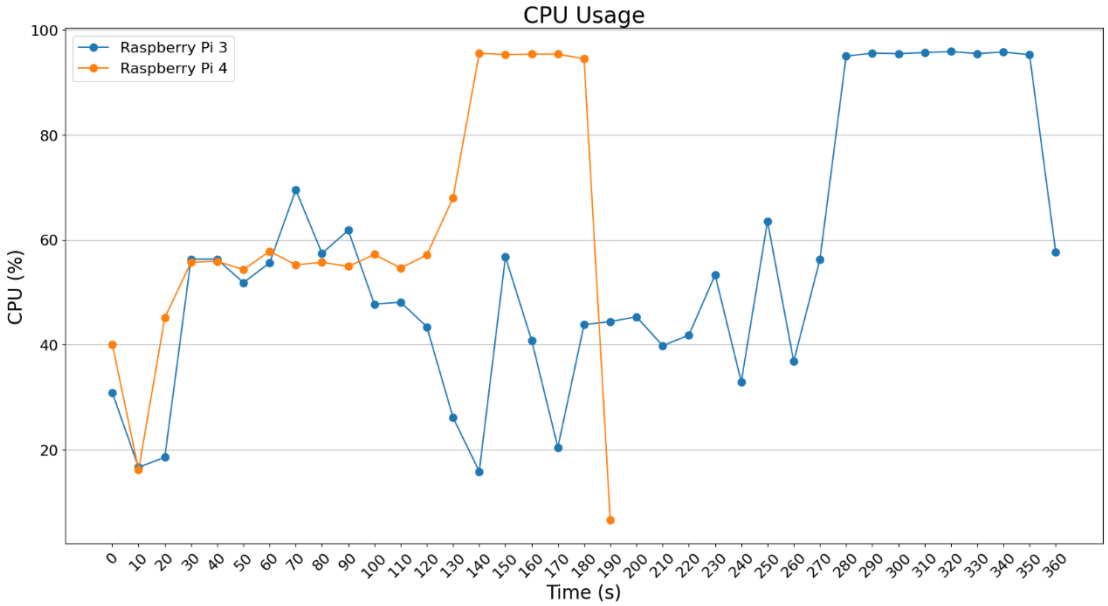

*Διάγραμμα 9: CPU Usage (ImageDataGenerator - Batch size = 4)*

Στο διάγραμμα 9 αυτό που αξίζει σχολιασμού είναι ότι το RPi 4 βελτίωσε λίγο τον χρόνο εκτέλεσης, εν αντιθέσει με το RPi 3 που βελτίωσε τον χρόνο 80 ολόκληρα δευτερόλεπτα κάτι που είναι πολύ σημαντικό αναλογικά του συνολικού χρόνου. Ακολουθούν τα διαγράμματα για batch size = 8 και batch size = 16.



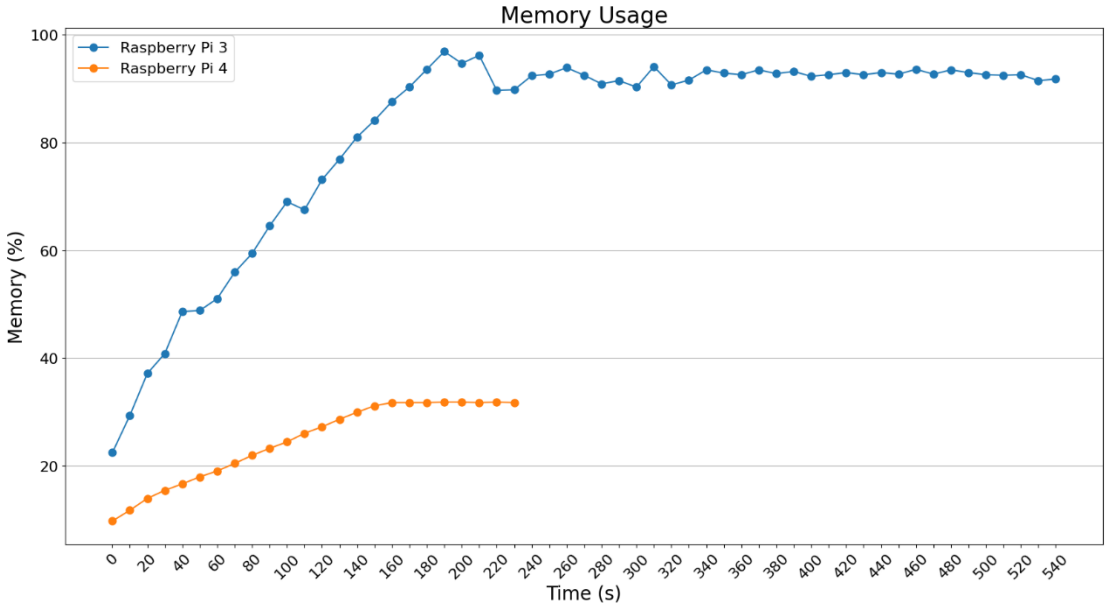

*Διάγραμμα 10: CPU Usage (ImageDataGenerator - Batch size = 8)*

Στο διαγράμμα 10 (batch size = 8) παρατηρείται μια μικρή χρονική βελτίωση ενώ παραμένει πολύ όμοιο με τα παραπάνω. Στο επόμενο διάγραμμα, το 11, ωστόσο παρατηρούμε σαφή βελτίωση στον χρόνο εκτέλεσης και για τα δύο RPi, και ο λόγος που επιτυγχάνεται αυτό είναι ότι με την κατάλληλη αύξηση των εικόνων για επεξεργασία ανά ομάδα (batch) μειώθηκαν οι επαναλήψεις όπου οι συσκευές λειτουργούν κοντά στην πλήρη επεξεργαστική τους δύναμη, καθώς την αξιοποιούσαν σαφώς αποτελεσματικότερα.



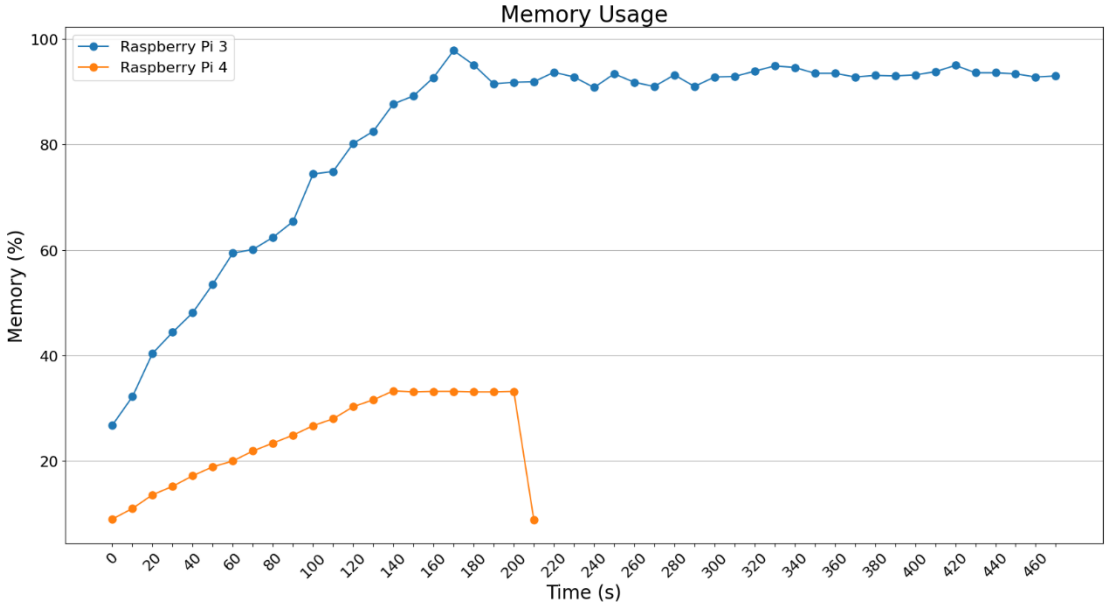

*Διάγραμμα 11: CPU Usage (ImageDataGenerator - Batch size = 16)*

Ακολουθούν τα διαγράμματα που παρουσιάζουν την χρήση της μνήμης για τα RPi 3 και RPi 4. Προηγούνται αυτά που καταδεικνύουν το ποσοστό μνήμης (διαγράμματα 12 – 15) που χρησιμοποιήθηκε και ακολουθούν τα αντίστοιχα που αναπαριστούν την χρησιμοποιούμενη μνήμη σε Mbytes (διαγράμματα 16 – 19).



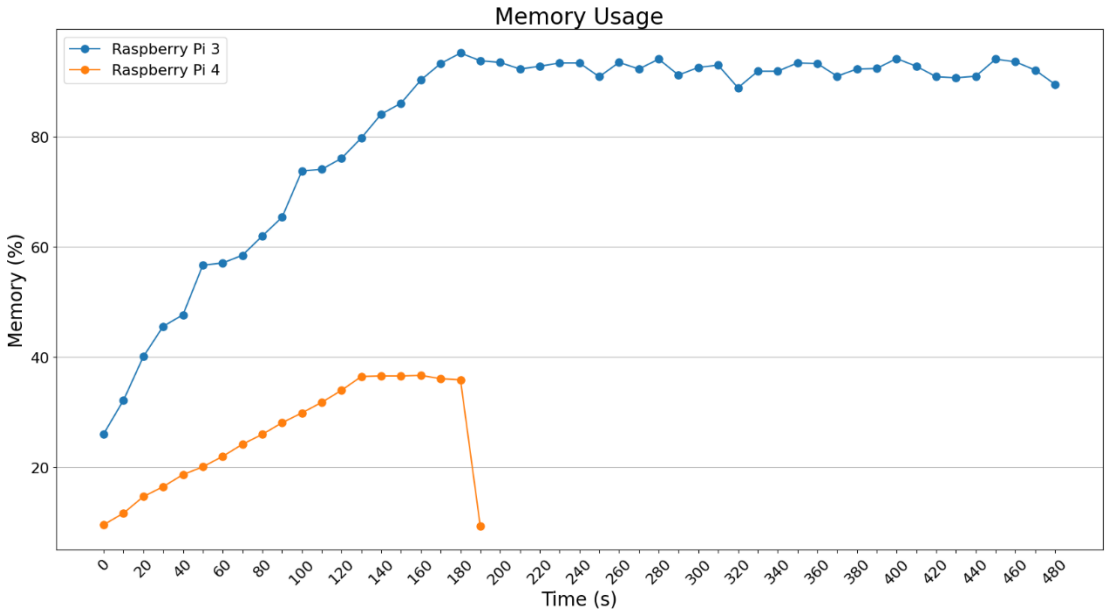

*Διάγραμμα 12: Memory Usage (Percentage) (ImageDataGenerator - Batch size = 2)*

Στα σχήματα 12 και 13 που αντιστοιχούν σε batch size = 2 και batch size = 4 παρατηρούμε αρκετά παρόμοιες τιμές τόσο για το RPi 3 όσο και για το RPi 4 συγκρινόμενα με τον εαυτό τους. Η διαφορά ωστόσο στην δυναμική των δύο μηχανημάτων είναι κάτι παραπάνω από προφανής καθώς το RPi 3 βρίσκεται γύρω στο 90% χρήσης της μνήμης RAM του ήδη, ενώ το RPi 4 βρίσκεται περίπου στο 30%.



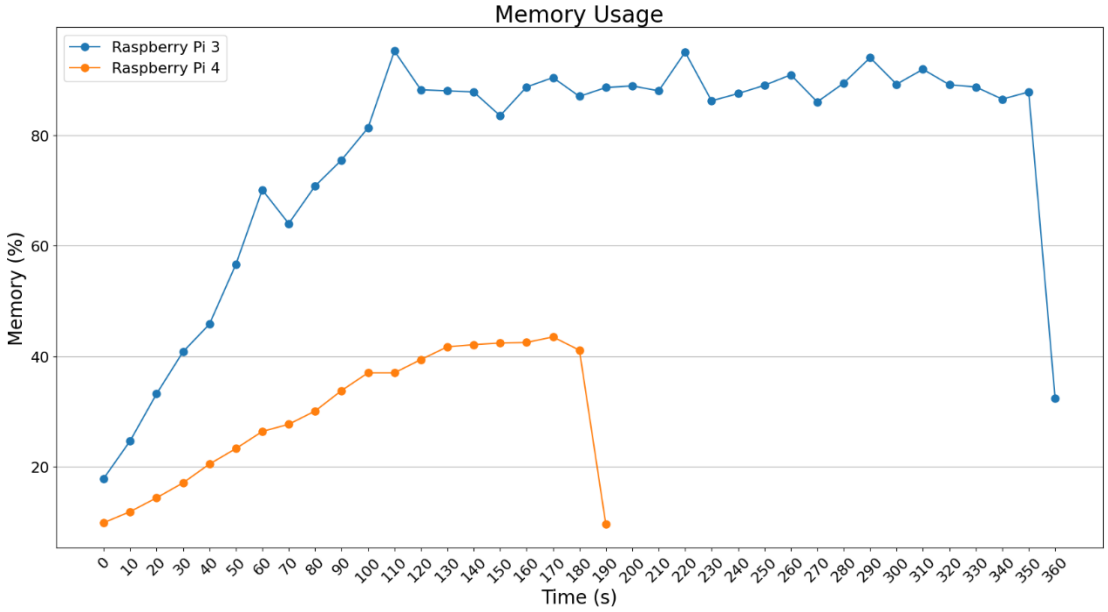

*Διάγραμμα 13: Memory Usage (Percentage) (ImageDataGenerator - Batch size = 4)*

Ακολουθούν τα σχήματα 14 και 15 που αντιστοιχούν σε batch size = 8 και batch size = 16.



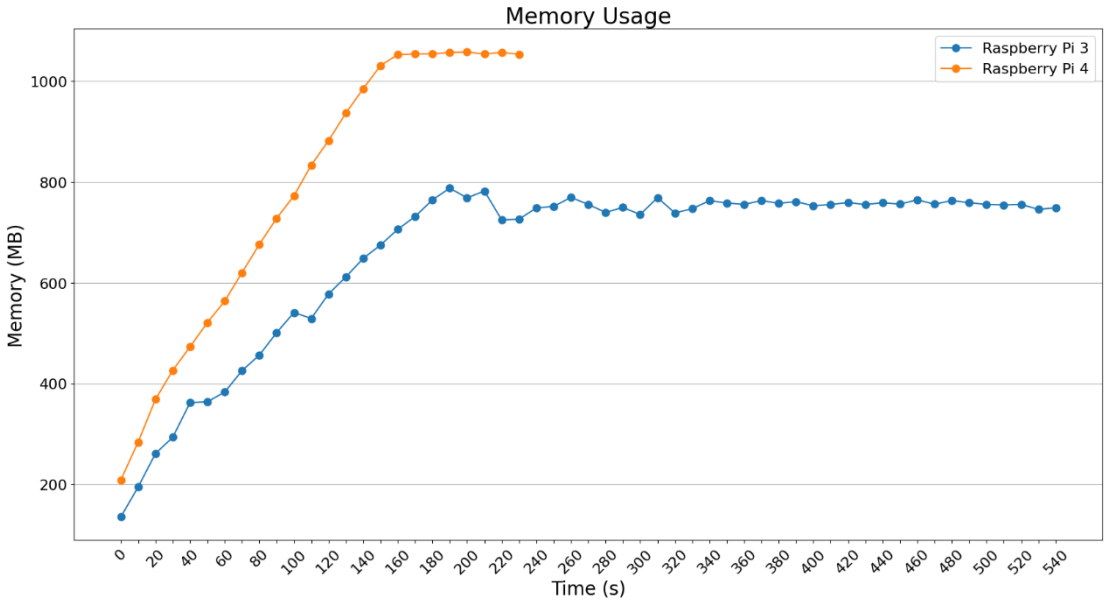

*Διάγραμμα 14: Memory Usage (Percentage) (ImageDataGenerator - Batch size = 8)*

Στα διαγράμματα 14 και 15 παρατηρούμε ότι η χρήσης RAM για το RPi 3 παραμένει υψηλά και περίπου 90%. Θεωρητικά αναμέναμε να μην μπορεί το RPi 3 να ολοκληρώσει αυτές τις διαδικασίες δεδομένου του έξτρα φόρτου, ωστόσο όπως θα δούμε παρακάτω παρά την οριακή χρήση της RAM, το «βοηθήσαμε» δίνοντας του την απαραίτητη extra μνήμη που είχε ανάγκη. Για το RPi 4, παρατηρούμε ότι δεν ζορίζεται επ' ουδενί κατ' αντιστοιχία με το RPi 3, ωστόσο από το 30% άγγιξε το 40% της χρήσης RAM του, ώστε να μπορέσει να επεξεργαστεί και να επιταχύνει τις διαδικασίες.



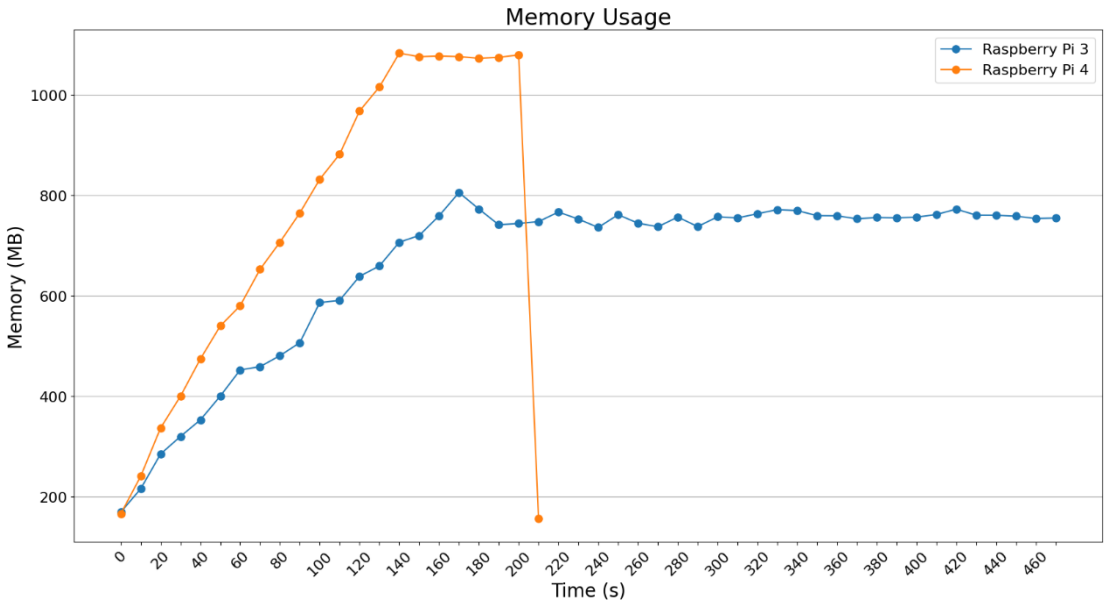

*Διάγραμμα 15: Memory Usage (Percentage) (ImageDataGenerator - Batch size = 16)*

Ας περάσουμε σε αυτό το σημείο στα αντίστοιχα διαγράμματα μνήμης με βάση την χωρητικότητα πλέον.



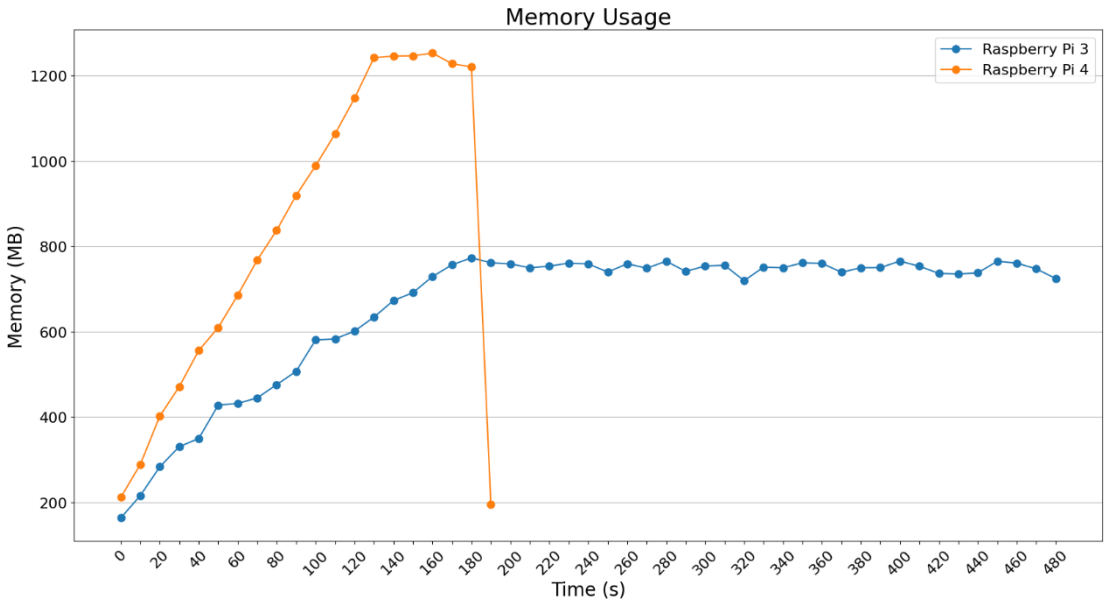

*Διάγραμμα 16: Memory Usage (MBytes) (ImageDataGenerator - Batch size = 2)*

Στα διαγράμματα 16 και 17 απεικονίζεται η χρήση της μνήμης σε Mbytes για την περίπτωση του batch size = 2 και batch size = 4 αντίστοιχα. Το RPi 3 χρησιμοποιεί 800 MB σχεδόν δηλαδή όλη την διαθέσιμη μνήμη του (875 MB η μέγιστη διαθέσιμη). Το RPi 4, χρησιμοποιεί περισσότερα MB μνήμης διότι η μέγιστη διαθέσιμη μνήμη του ανέρχεται στα 4 GB, κάτι που το καθιστά σαφώς ισχυρότερο μηχάνημα, όπως σταθερά αναδεικνύεται και από τον χρόνο διεκπεραίωσης των διεργασιών.



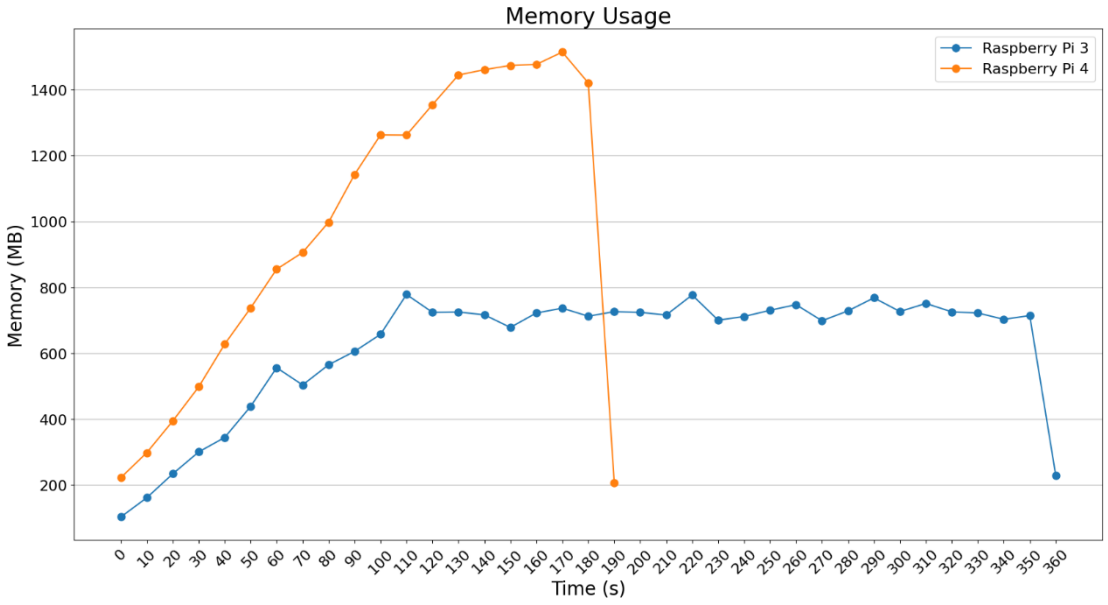

*Διάγραμμα 17: Memory Usage (MBytes) (ImageDataGenerator - Batch size = 4)*

Ακολουθούν στα διαγράμματα 18 και 19 η χρήση της μνήμης σε Mbytes στις 2 πιο βαριές περιπτώσεις που έχουμε παρουσιάσει ως τώρα, δηλαδή για batch size = 8 και batch size = 4.



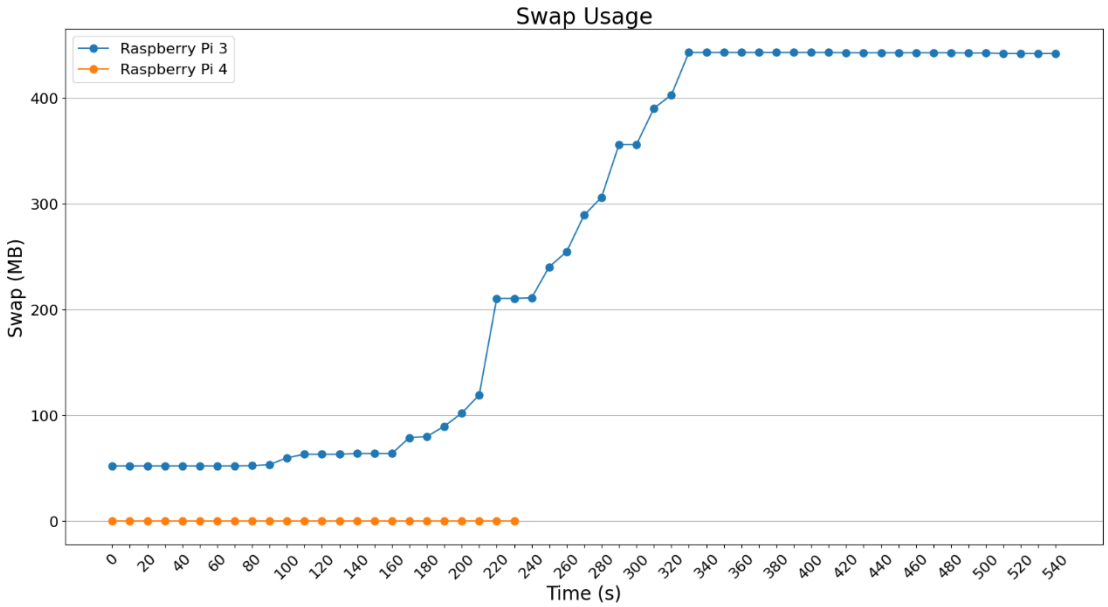

*Διάγραμμα 18: Memory Usage (MBytes) (ImageDataGenerator - Batch size = 8)*

Για τα σχήματα 18 και 19 αυτό που παρατηρούμε είναι ότι αρχίζει να δουλεύει λίγο παραπάνω το RPi 4 σε σχέση με τις προηγούμενες περιπτώσεις καθώς οι απαιτήσεις σε μνήμη αυξάνονται, δεδομένου ότι φορτώνει και επεξεργάζεται πλέον 8 και 16 φωτογραφίες, αγγίζοντας τα 1250 MB και 1500 MB μνήμης RAM. Το Rpi 3 παραμένει να βρίσκεται στα «κόκκινα» και παρά τις έντονες δυσκολίες που αντιμετωπίσαμε για την τελική εκτέλεση στο RPi 3 παρακάτω παρουσιάζεται πως επιλύθηκε το ζήτημα της μνήμης.

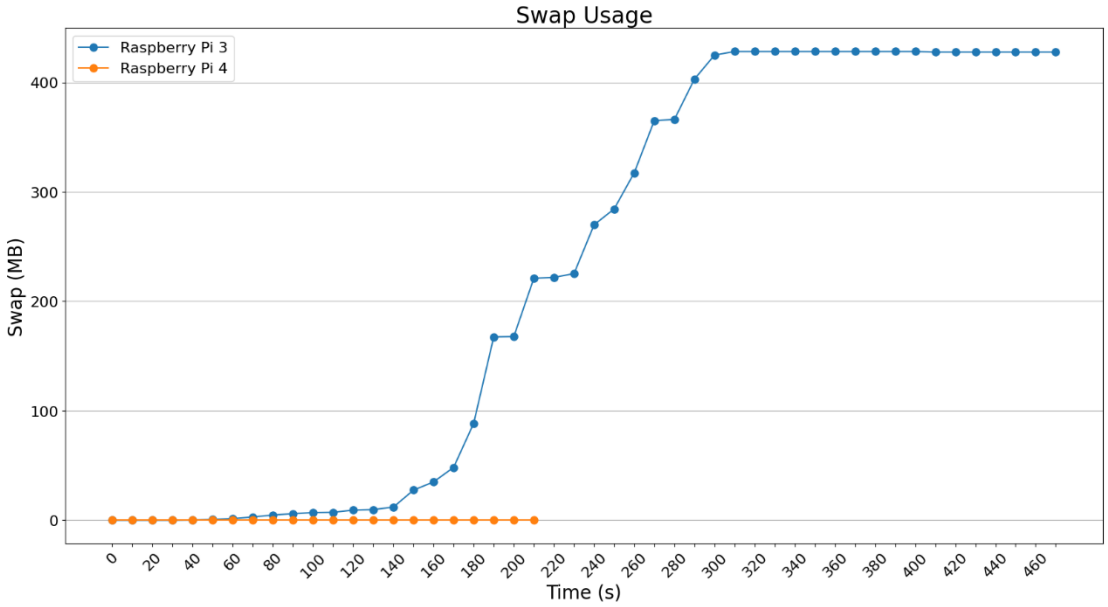



*Διάγραμμα 19: Memory Usage (MBytes) (ImageDataGenerator - Batch size = 16)*

Γίνεται εμφανές, ότι το Raspberry Pi 4 δεν φτάνει στα όρια του για την συγκεκριμένη εργασία εν αντιθέσει με το Raspberry Pi 3, όπου σχεδόν πάντα βρισκόταν πάνω από το 85% χρήσης της μνήμης RAM και CPU. Για να μπορέσει να ολοκληρώσει τις εργασίες αυτές, αρχικά «έσκαγε» από μνήμη, οπότε φροντίσαμε να κάνουμε SWAP ένα μέρος της αποθηκευτικής μνήμης, ώστε να την αξιοποιεί ως $2^{η}$ μνήμη RAM, θέτοντάς του ένα πολύ μεγάλο όριο της τάξης των 20 GB, ώστε να μπορεί να χρησιμοποιήσει όση χρειάζεται. Όπως θα γίνει σαφές από τα επόμενα διαγράμματα, το Rpi 3 ανάλογα με την δυσκολία του task που είχε να φέρει εις πέρας, καθώς αυξάναμε το batch_size, χρησιμοποιούσε όλο και περισσότερη από την μνήμη SWAP, σε αντίθεση με το RPi4, στο οποίο επίσης θέσαμε την ίδια SWAP μνήμη αλλά δεν την είχε ανάγκη. Ακολουθούν τα σχετικά διαγράμματα 20 – 23 που αναπαριστούν την μνήμη Swap για τα 2 μηχανήματα.

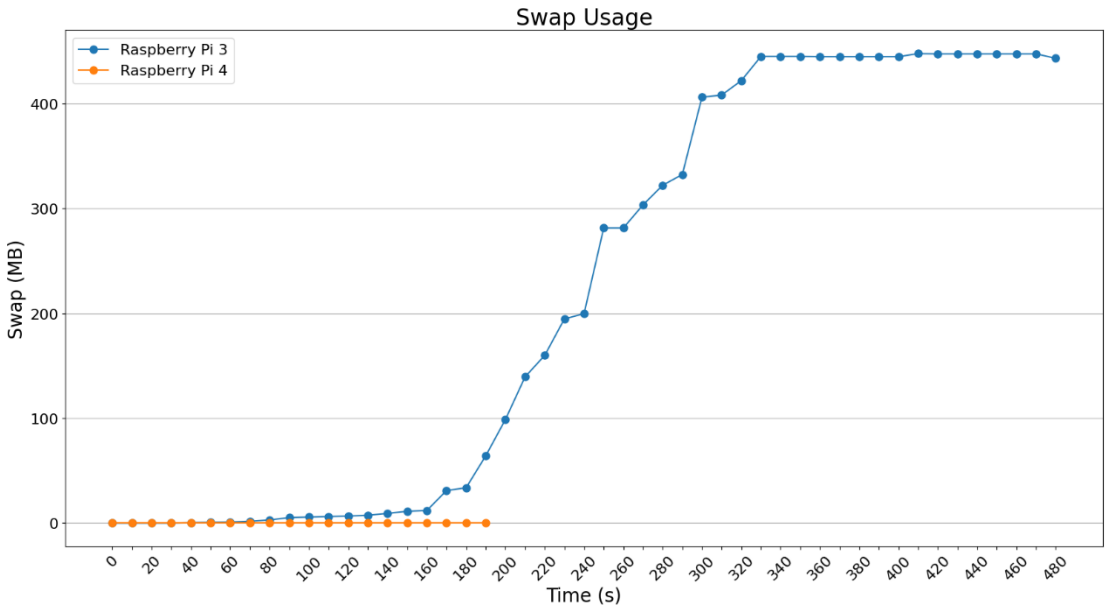



*Διάγραμμα 20: Swap Usage (MBytes) (ImageDataGenerator - Batch size = 2)*



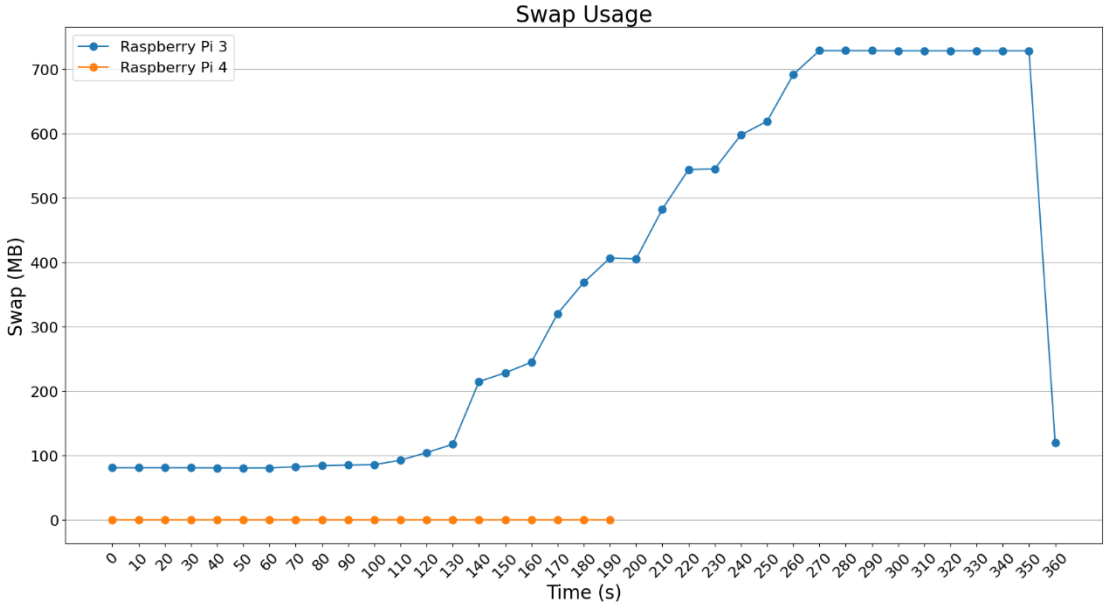

*Διάγραμμα 21: Swap Usage (MBytes) (ImageDataGenerator - Batch size = 4)*

Στις περιπτώσεις των batch size = 2, 4, 8 αυτό που αντιλαμβανόμαστε είναι ότι περίπου στην μέση της εκτέλεσης του inference (prediction process) «έσκαγε» από μνήμη και χρησιμοποιούσε περίπου 400 MB έξτρα μνήμης για να καλύψει τις ανάγκες του.




*Διάγραμμα 22: Swap Usage (MBytes) (ImageDataGenerator - Batch size = 8)*

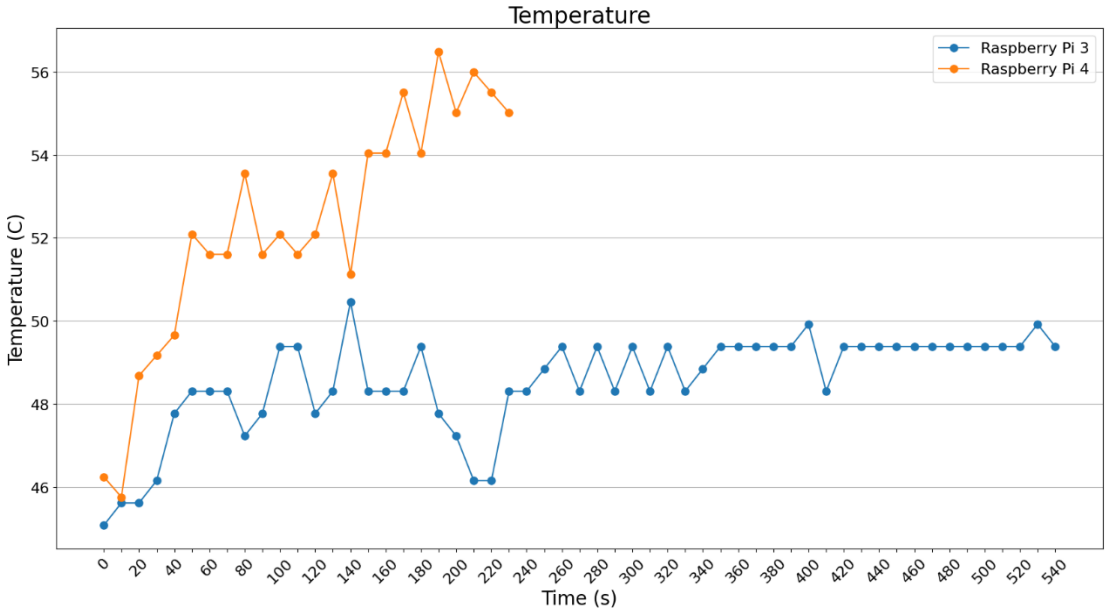
Ωστόσο, στην περίπτωση του batch size = 16 (διάγραμμα 23), αυτό αποδείχθηκε ιδιαίτερα πιο «επίπονο» για το RPi 3, όπου αναγκάστηκε για την φόρτωση των εικόνων να χρησιμοποιήσει περίπου 750 MB στο τέλος.



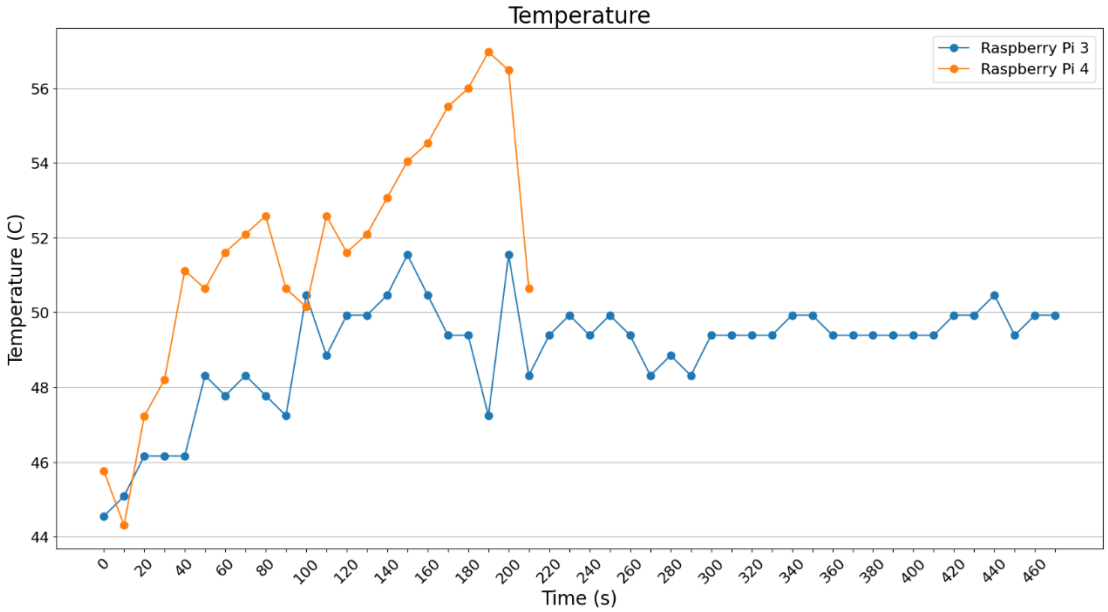

*Διάγραμμα 23: Swap Usage (MBytes) (ImageDataGenerator - Batch size = 16)*

Στις 3 πρώτες περιπτώσεις το RPi 3 αξιοποίησε περίπου 450 MB έξτρα μνήμης (50% επί της βασικής μνήμης RAM του), ενώ στην περίπτωση με batch_size = 16 αξιοποίησε σχεδόν 750 MB μνήμης, που είναι περίπου το 85% της αρχικής διαθέσιμης μνήμης

RAM του, κάτι που υποδεικνύει και την δυσκολία αυτού του task, πόσο μάλλον και του batch_size = 32, όπου θα το δούμε σε έναν συγκριτικό διάγραμμα με όλες τις μετρήσεις του RPi 4. Στον τομέα των θερμοκρασιών δεν παρατηρείται κάτι ιδιαίτερο, καθώς είναι σχετικά σταθερή η διακύμανσή τους.

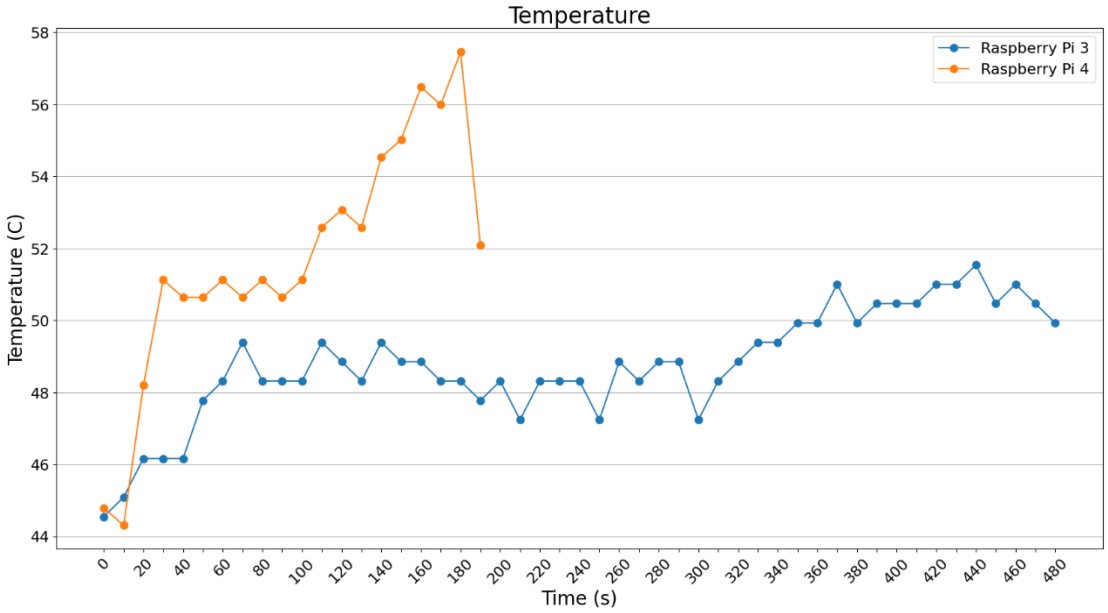



*Διάγραμμα 24: Θερμοκρασία Συσκευών (ImageDataGenerator - Batch size = 2)*



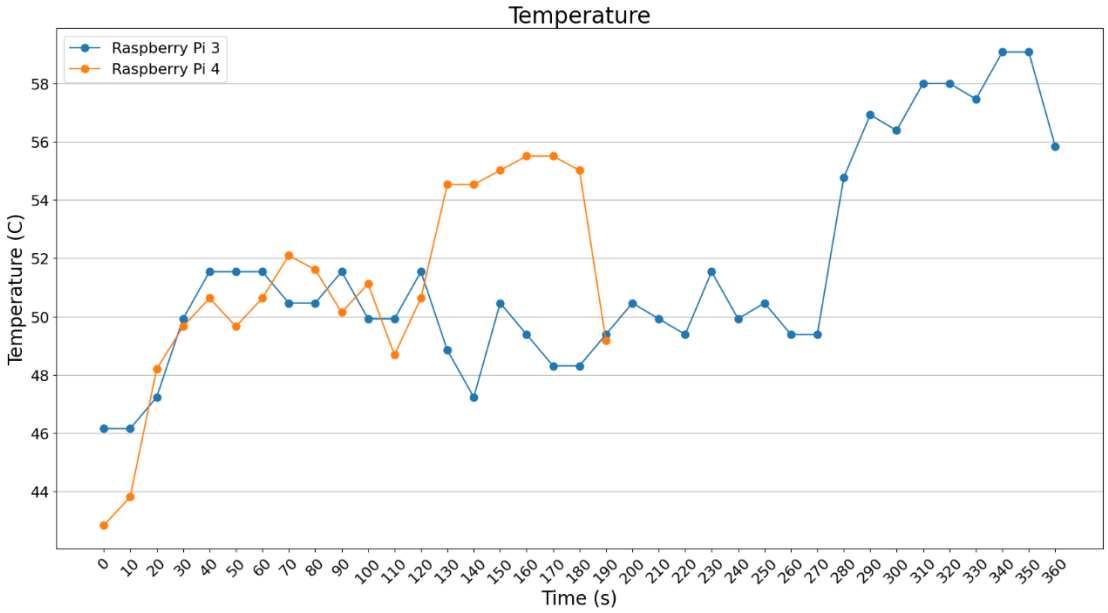

*Διάγραμμα 25: Θερμοκρασία Συσκευών (ImageDataGenerator - Batch size = 4)*

Για batch size = 2 και batch size = 4, παρατηρούμε ότι το RPi 3 δούλευε σταθερά περίπου στους 49$^{o}$ C, ενώ το RPi 4 είχε μια εντονότερη αύξηση από τους 45$^{o}$ C στους 55$^{o}$ C.




*Διάγραμμα 26: Θερμοκρασία Συσκευών (ImageDataGenerator - Batch size = 8)*

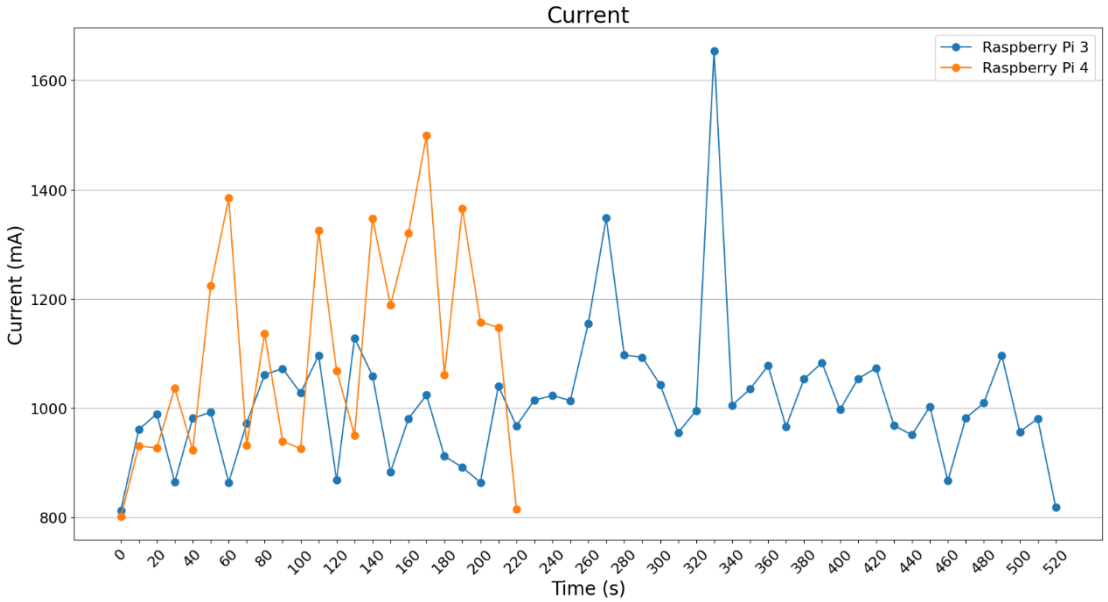
Στα σχήματα 26 και 27 όπου έχουμε αντίστοιχα τις περιπτώσεις για batch size = 8 και batch size = 16, παρατηρούμε ότι το RPi 3 παραμένει στους 49$^{o}$ C με ελαφριά spikes προς το τέλος, κάτι που είναι φυσιολογικό διότι όπως παρατηρήσαμε από τα σχετικά διαγράμματα με την επεξεργαστική χρήση και την μνήμη RAM, βρίσκεται από την αρχή μέχρι το τέλος στα «κόκκινα». Για το RPi 4 παρατηρούμε και πάλι ότι ξεκινάει από μια πιο χαλαρή κατάσταση, αλλά σταδιακά αυξάνει την θερμοκρασία του παρουσιάζοντας περισσότερες μέγιστες τιμές (spikes) όσο ζορίζει το πρόβλημα.




*Διάγραμμα 27: Θερμοκρασία Συσκευών (ImageDataGenerator - Batch size = 16)*

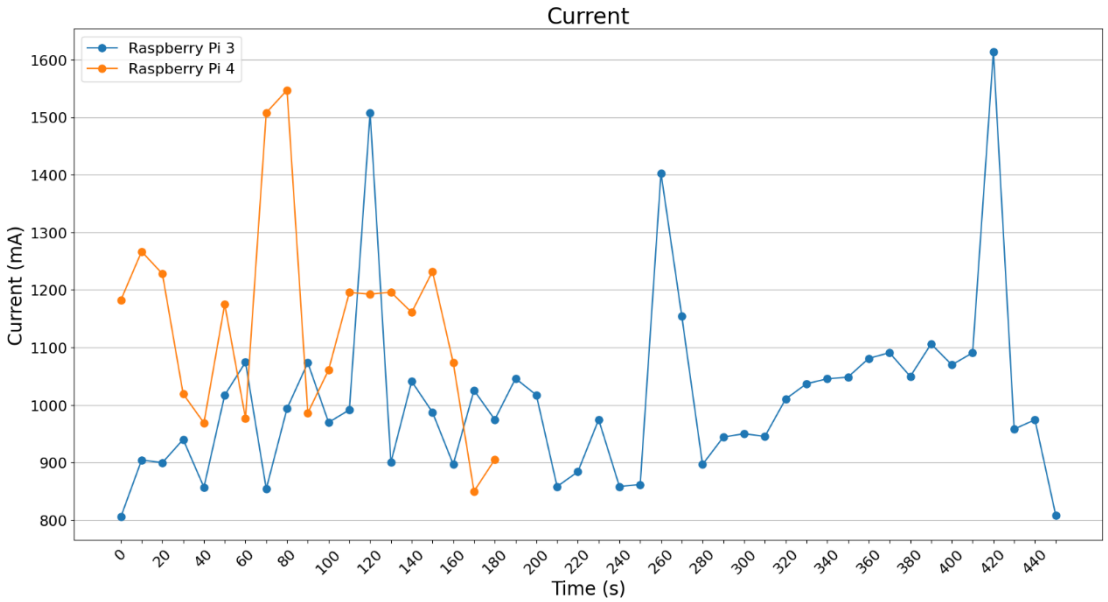
Όσον αφορά τώρα την κατανάλωση των ρευμάτων ανά batch size, αυτό που ίσχυε και παραπάνω ισχύει και σε αυτή την περίπτωση ότι όσο αυξάνεται το μέγεθος του batch,

τόσο περισσότερο ζορίζονται τα μηχανήματα και συνεπώς είχαμε και περισσότερη κατανάλωση ρεύματος και περισσότερα spikes. Ωστόσο, οι τιμές κυμάνθηκαν κατά μέσο όρο από 1050 mA στο batch size = 2, έως 1200 mA στο batch size = 16 για το RPi 4, ενώ το RPi 3 παρουσίασε εντονότερα ακραίες τιμές.




*Διάγραμμα 28: Μετρική Ρεύματος (mA) (ImageDataGenerator - Batch size = 2)*

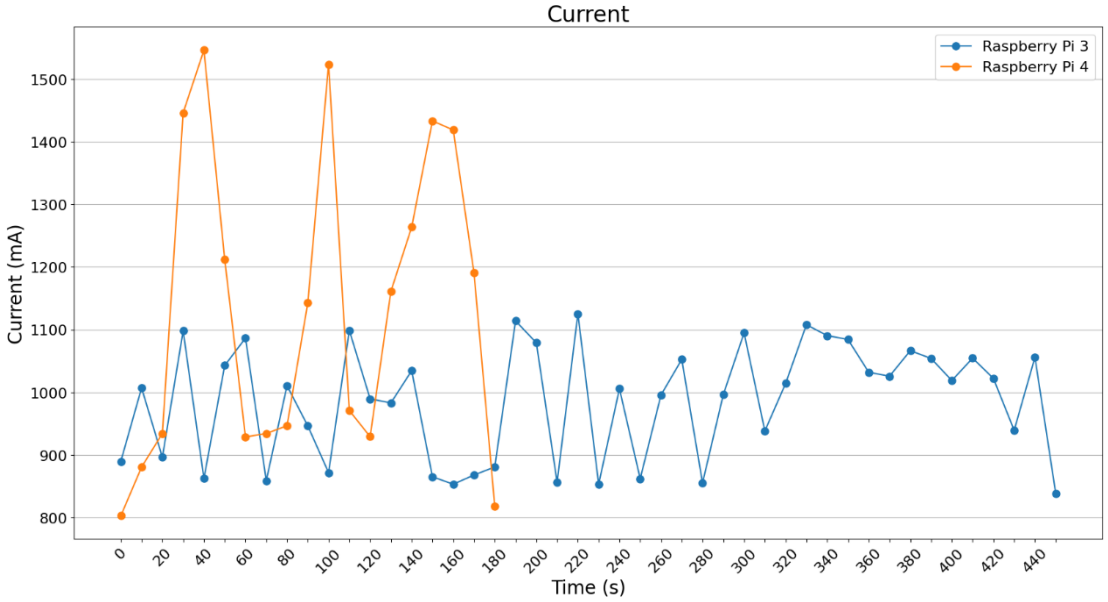
Στο παραπάνω διάγραμμα παρατηρούμε ότι δεν έχουμε πολλές ακραίες τιμές (spikes) και για τα δύο μηχανήματα, με περισσότερη σταθερότητα να φαίνεται αρχικά ότι παρουσιάζει το RPi 3, κάτι που όπως θα δούμε παρακάτω αλλάζει πολύ έντονά στο τέλος.




*Διάγραμμα 29: Μετρική Ρεύματος (mA) (ImageDataGenerator - Batch size = 4)*

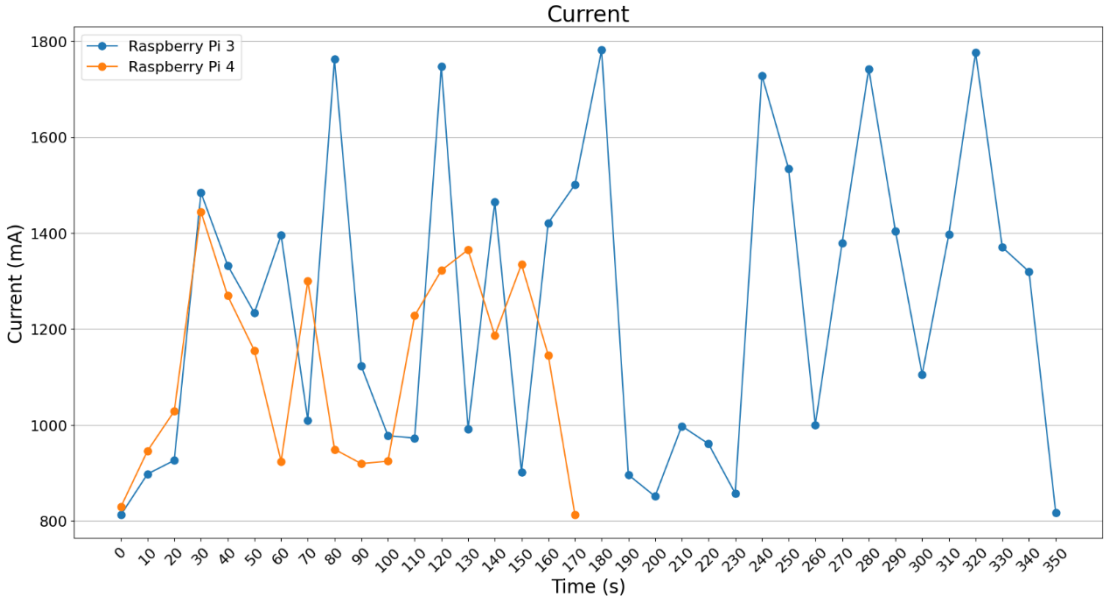
Στην περίπτωση του batch size = 4, παρατηρούμε μια ελαφρά αλλά σταθερή αύξηση για την χρήση ρεύματος στο RPi 4 κατά μέσο όρο γύρω στα 1200 mA, ενώ το RPi 3

αρχίζει να φανερώνει την δυσκολία του με ορισμένα έντονα spikes σε τιμές πάνω από 1500 mA. Ακολουθεί η περίπτωση του batch size = 8 στην οποία απλώς παρατηρούμε μια μικρή ρευματική αύξηση και στα δύο μηχανήματα, χωρίς να είναι κάτι το σπουδαίο.



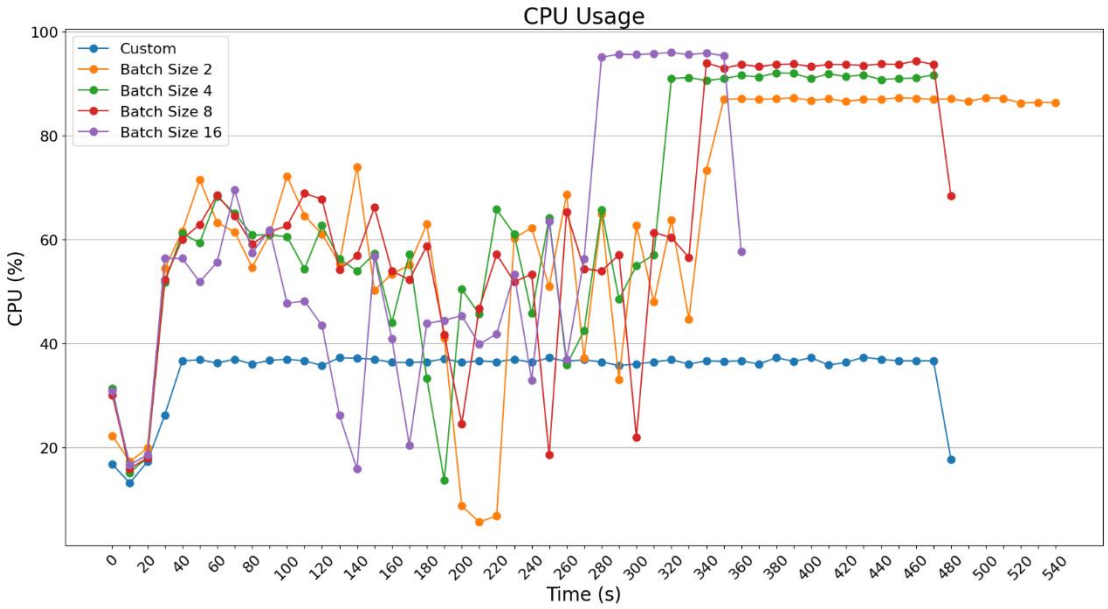

*Διάγραμμα 30: Μετρική Ρεύματος (mA) (ImageDataGenerator - Batch size = 8)*

Στο ακόλουθο σχήμα (διάγραμμα 31) είναι όπου φαίνεται η συντριπτική διαφορά μεταξύ του ισχυρού υπολογιστή RPi 4 και του σαφώς πιο αδύναμου RPi 3. Στην περίπτωση του batch size = 16, προφανώς παρατηρείται μια ελαφριά ρευματική αύξηση συγκριτικά με το batch size = 8. Ωστόσο, το RPi 3 έφτασε στα όρια του καταναλώνοντας ανά στιγμές πάνω από 1700 mA στις μισές σχεδόν τιμές που δειγματοληπτήσαμε, σε αντίθεση με το RPi 4, που δούλευε κατά μέσο όρο στα 1200 mA.



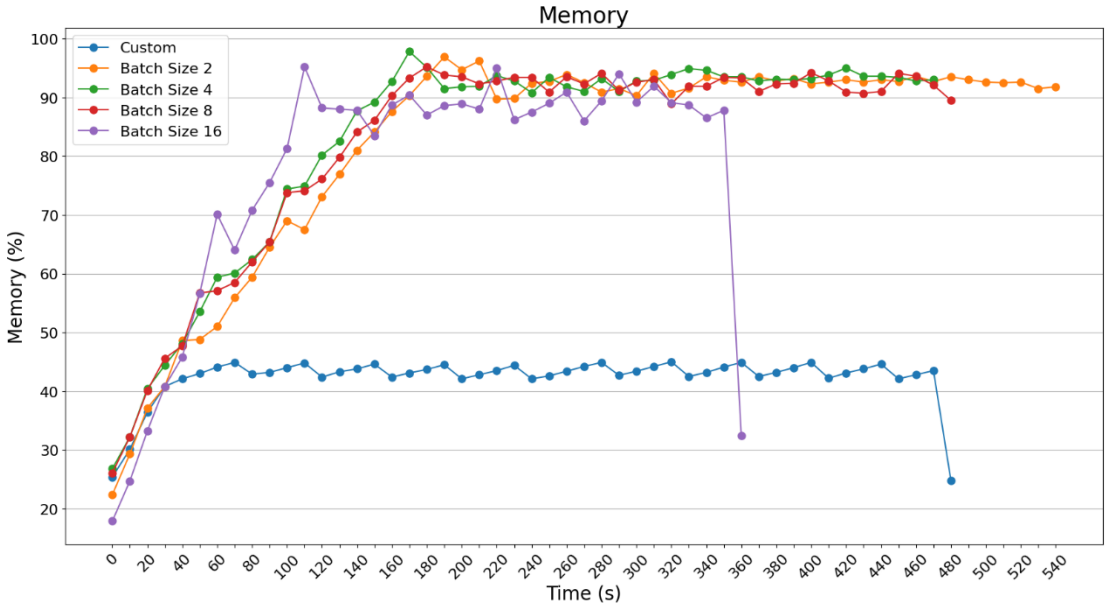

*Διάγραμμα 31: Μετρική Ρεύματος (mA) (ImageDataGenerator - Batch size = 16)*

Κατόπιν αυτών των μετρήσεων και διαγραμμάτων, θέλαμε να δούμε όχι μόνο πως συγκρίνονται μεταξύ τους τα RPi ανά batch_size, αλλά πως συγκρίνεται η κάθε συσκευή με τον εαυτό της δεδομένων όλων των εκτελέσεων συμπεριλαμβανομένης και της αρχικής τεχνικής με το Pillow. Ξεκινώντας με το RPi 3 λάβαμε τα εξής διαγράμματα στα οποία καθίσταται σαφές ότι για να κερδίσουμε σε χρόνο εκτέλεσης απαιτήθηκαν περισσότεροι πόροι και ειδικά στην περίπτωση του batch size = 16.



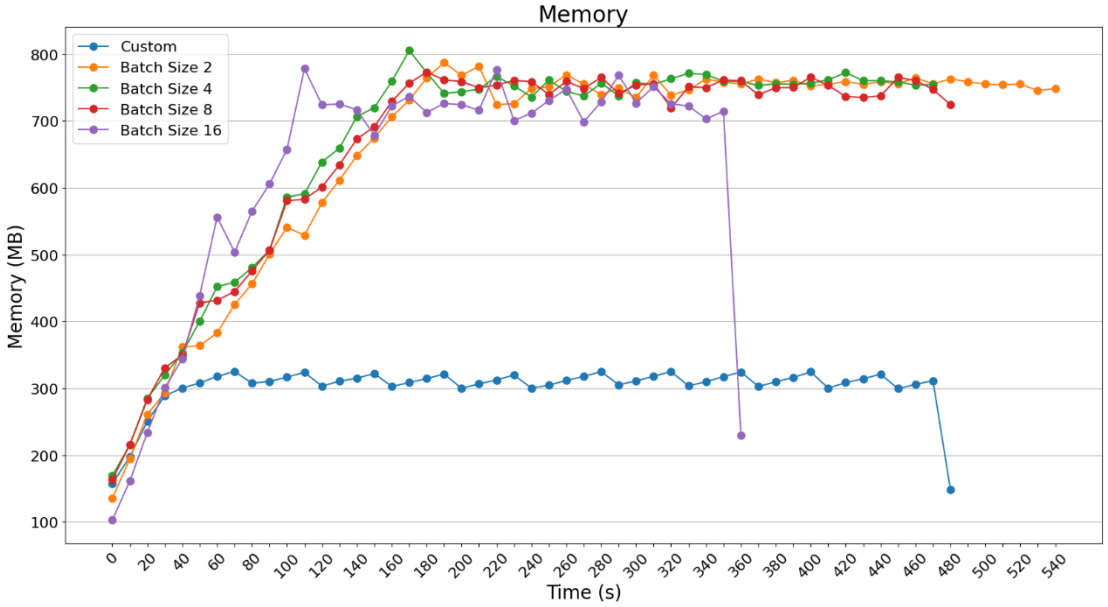

*Διάγραμμα 32: CPU Usage (RPi3)*

Στα διαγράμματα 32 και 33 παριστάνονται αντίστοιχα η χρήση του επεξεργαστή και της μνήμης RAM αντίστοιχα, για το RPi 3 βασισμένα σε όλες τις διαφορετικές εκτελέσεις. Αυτό που καθίσταται σαφές είναι ότι η χρήση ενός custom τρόπου φόρτωσης των εικόνων μία προς μία του επίτρέπει να μην χρησιμοποιεί πολλούς υπολογιστικούς πόρους φτάνοντας στο τελικό αποτέλεσμα. Ωστόσο, με την χρήση του ImageDataGenerator που μας επέτρεψε να αξιοποιήσουμε τις εικόνες ανά batch, δηλαδή ανά ομάδες επεξεργασίας μειώθηκε πολύ ο χρόνος εκτέλεσης.



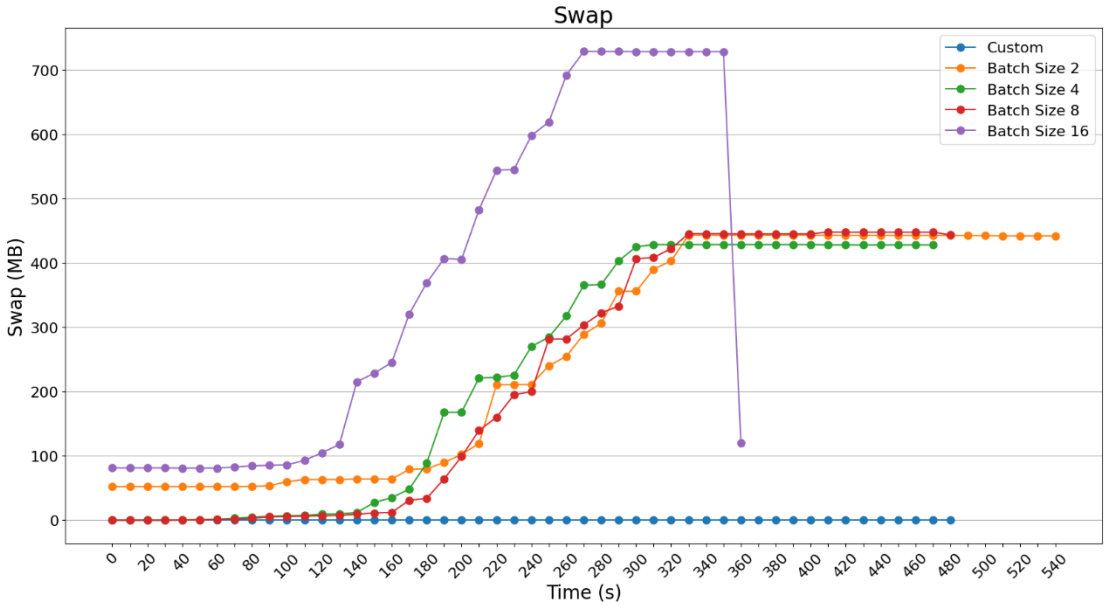

*Διάγραμμα 33: Memory Usage (Percentage) (RPi3)*

Στο ακόλουθο διάγραμμα απεικονίζεται η αντίστοιχη χρήση της μνήμης σε Mbytes που είναι εντελώς ανάλογη με το διάγραμμα του ποσοστού της χρησιμοποιούμενης μνήμης RAM.

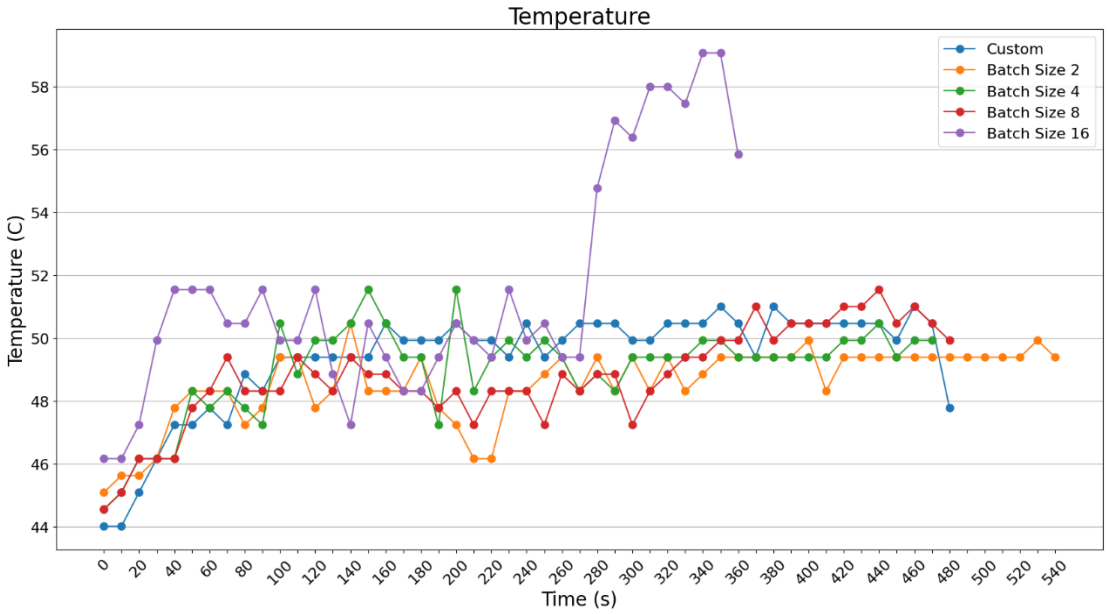



*Διάγραμμα 34: Memory Usage (MBytes) (RPi3)*

Το διάγραμμα που ακολουθεί, το 35, είναι αυτό που καταδεικνύει ουσιαστικά την επιπλέον απαιτητικότητα λόγω της μεγαλύτερης ομάδας φόρτωσης εικόνων, καθώς η μνήμη που απαιτείται είναι σαφώς παραπάνω. Όπως φαίνεται και παρακάτω το RPi 3 σχεδόν διπλασιάζει την μνήμη που χρησιμοποιεί προσθέτοντας 750 MB Swap, για να μπορέσει να ολοκληρώσει τις διεργασίες στην δυσκολότερη υπολογιστικά αλλά σαφώς ταχύτερη περίπτωση.



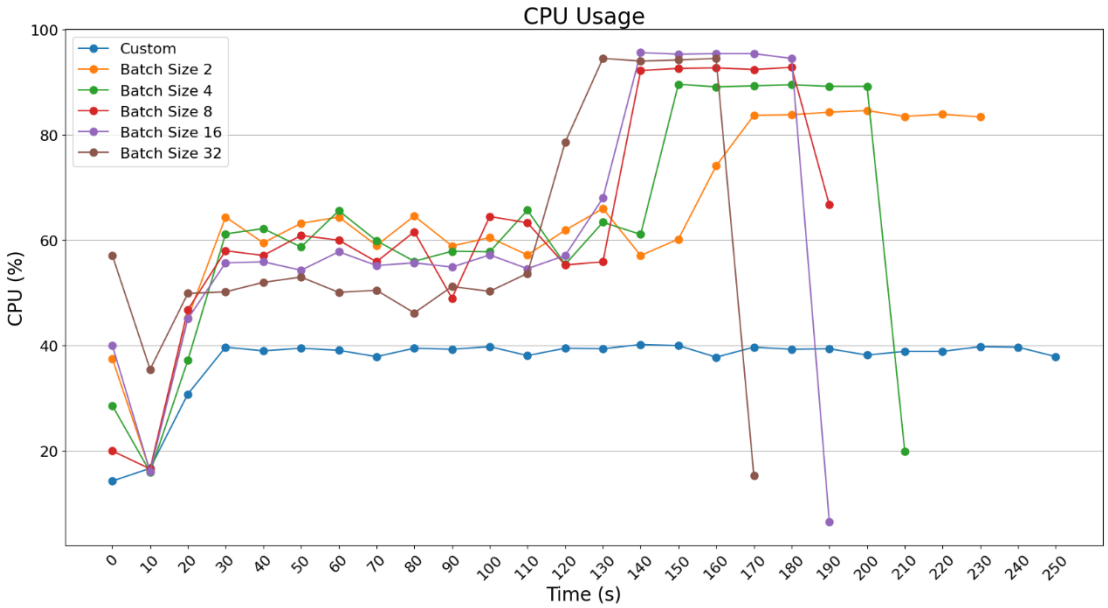

*Διάγραμμα 35: Swap Usage (MBytes) (Rpi3)*




*Διάγραμμα 36: Θερμοκρασία RPi3*

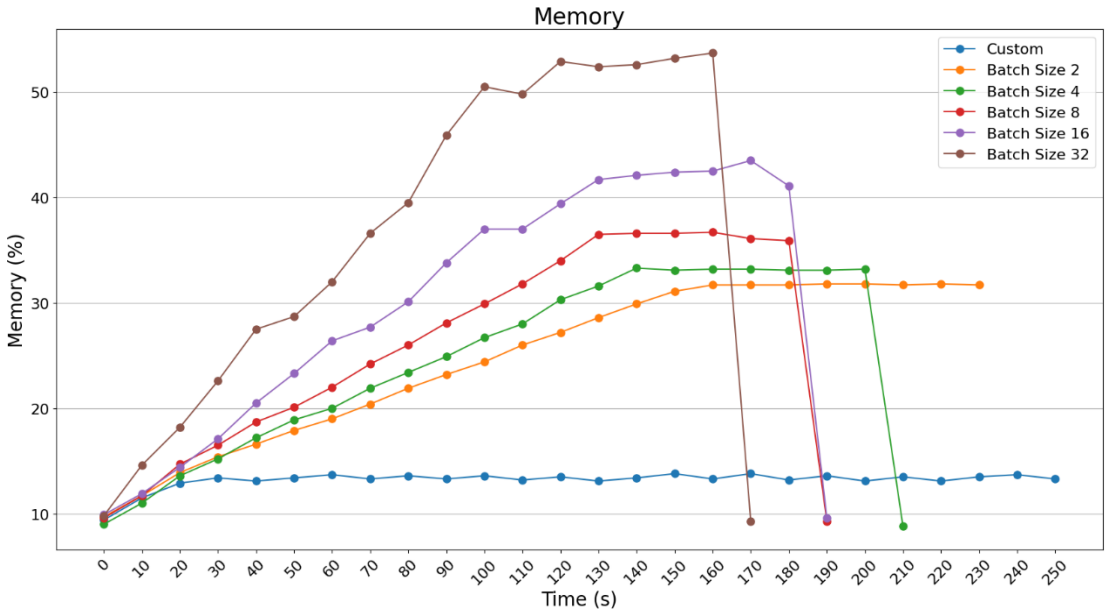
Είναι ξεκάθαρη η διαφορά στον χρόνο εκτέλεσης με batch size = 16, αλλά και η διαφορά στην ανάγκη υπολογιστικών πόρων, καθώς ακόμη και στην θερμοκρασία (διάγραμμα 36) που δεν είχαμε παρατηρήσει έντονες συμπεριφορές ως τώρα ξεχωρίζει έντονα το batch size = 16.

Για το RPi 4 δοκιμάσαμε και batch size = 32 όπως θα φανεί στα επόμενα διαγράμματα που φυσικά επέφερε ταχύτερα αποτελέσματα αξιοποιώντας πληρέστερα τους υπολογιστικούς του πόρους.




*Διάγραμμα 37: CPU Usage (RPi4)*

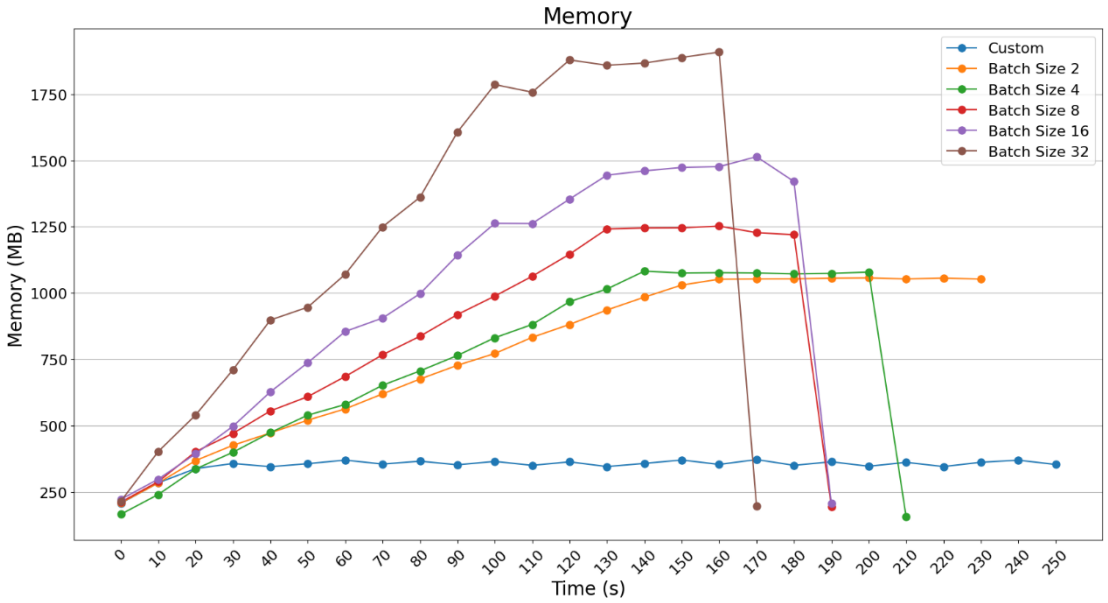
Όσον αφορά την χρήση επεξεργαστικής ισχύος δεν παρατηρήθηκε έντονη διαφορά μεταξύ των πειραμάτων με την χρήση του ImageDataGenerator, ωστόσο είναι σημαντική συγκριτικά με το custom prediction.




*Διάγραμμα 38: Memory Usage (Percentage) (Rpi4)*

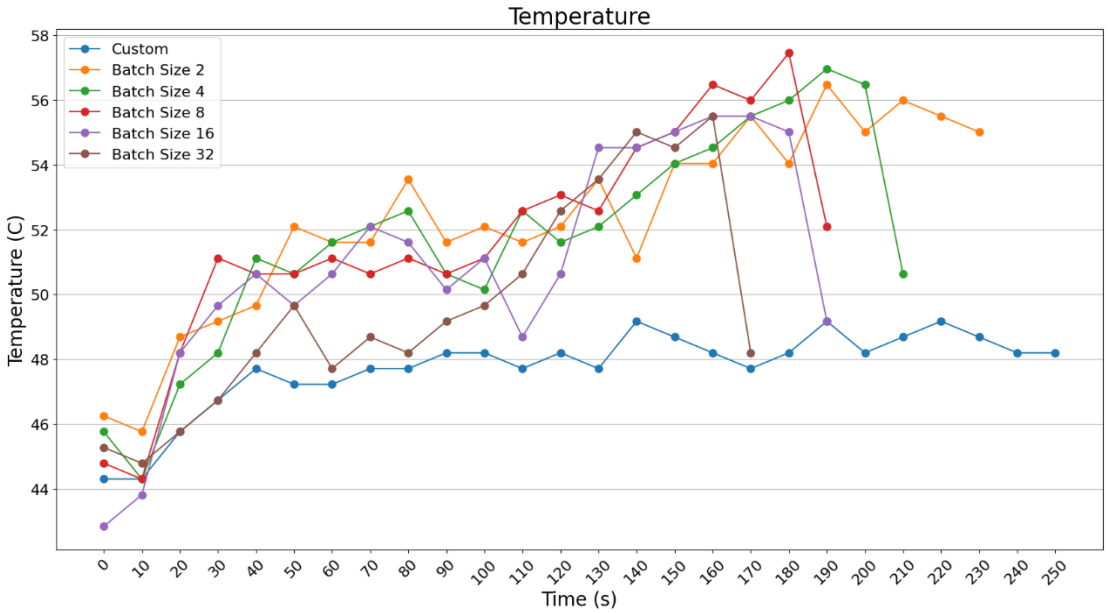
Στα διαγράμματα 38 λαμβάνουμε μια πρώτη εικόνα για την επιτάχυνση με την χρήση batch size = 32, περίπου 20 δευτερόλεπτα, ωστόσο για αυτό το κέρδος στον χρόνο εκτέλεσης με τον διπλασιασμό των φωτογραφιών προς επεξεργασία ανά ομάδα αυξήθηκε όπως παρατηρούμε το ποσοστό μνήμης σε πάνω από 50%, ενώ ως τότε

οριακά ξεπερνούσε το 40% και αντίστοιχα σε Mbytes πάνω από 1800 Mbytes, ενώ στις προηγούμενες περιπτώσεις δεν ξεπερνούσε τα 1500 Mbytes.




*Διάγραμμα 39: Memory Usage (MBytes) (RPi4)*

Στην περίπτωση της θερμοκρασίας (διάγραμμα 40) δεν προκύπτει κάποια ουσιαστική διαφορά, καθώς σε όλες τις περιπτώσεις διατηρεί την θερμοκρασία του του RPi 4 περί τους 52$^{o}$ C.

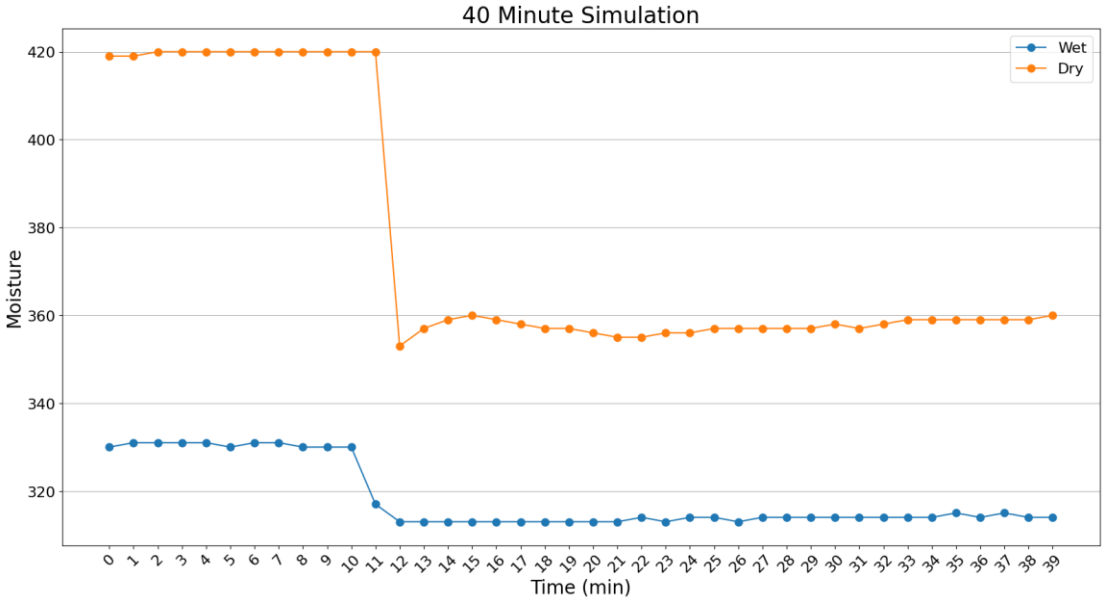



*Διάγραμμα 40: Θερμοκρασία RPi4*

Εκτός από την ξεκάθαρη βελτίωση στον χρόνο εκτέλεσης με batch size = 32 και την αναγκαία αύξηση κυρίως των υπολογιστικών πόρων μνήμης, αυτό που αξίζει σχολιασμού επίσης για το RPi 4 είναι το πόσο πιο άνετα υλοποιεί την διαδικασία του prediction με το Pillow που είναι ο custom τρόπος διαβάσματος των εικόνων μία προς μία συγκριτικά με την χρήση του ImageDataGenerator, αν και χρειάζεται περίπου 80 sec παραπάνω

## 6.2 Arduino και IoT πειράματα

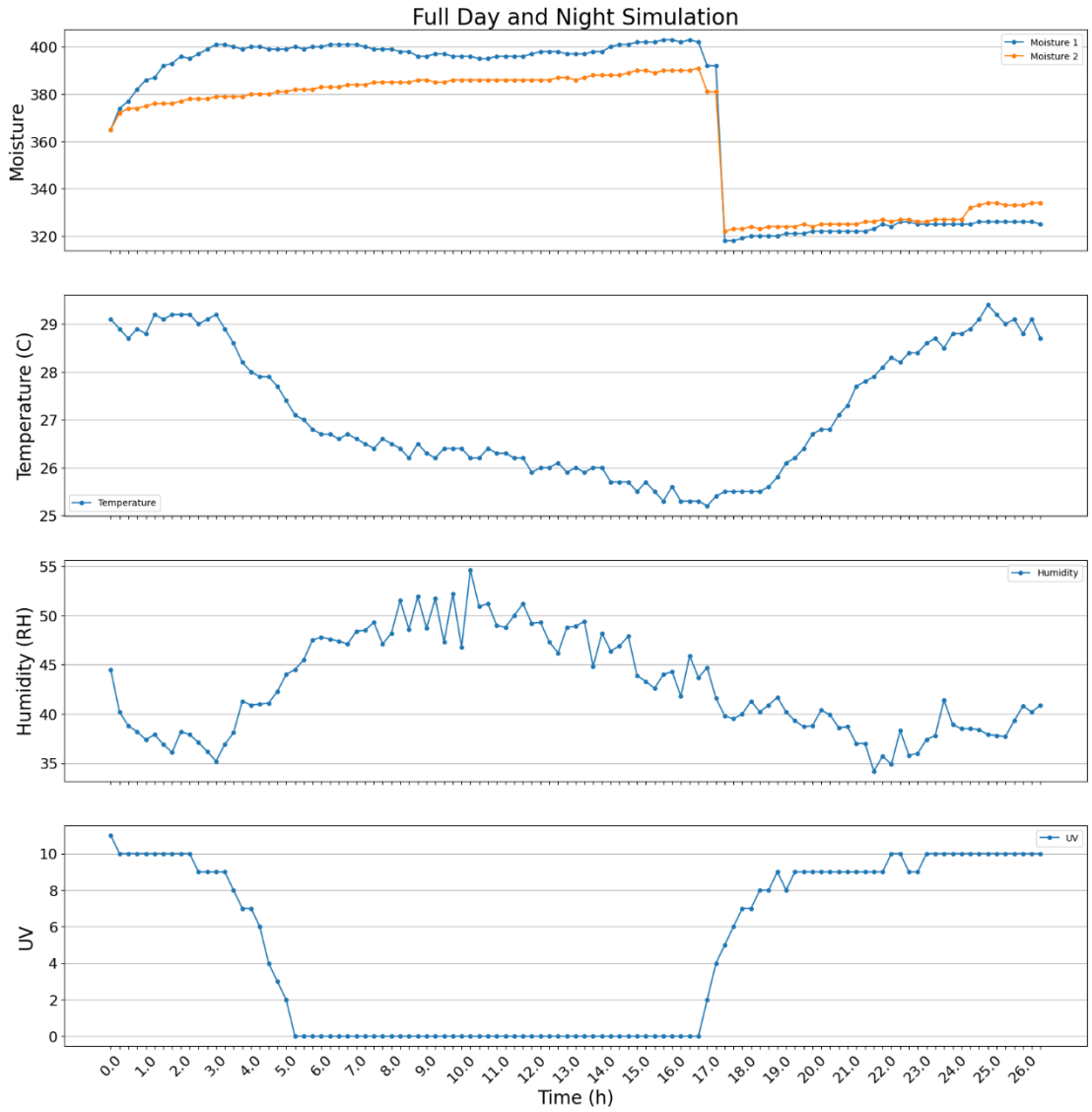
Για το IoT οικοσύστημα μας κάναμε διάφορα πειράματα για να υπολογίσουμε αρχικά τα όρια που τέθηκαν για την ανάγκη ποτίσματος και όταν το φυτό ήταν εντάξει από νερό βασιζόμενοι στις μετρήσεις των αισθητήρων υγρασίας εδάφους. Τα 2 πειράματα τα οποία θέλουμε να παρουσιάσουμε είναι τα ακόλουθα. Το 1ο πείραμα βασίστηκε στο Irrigation Mode όπως το περιγράψαμε όπου για 40 λεπτά λαμβάναμε μετρήσεις κάθε 1 λεπτό. Οι 2 αισθητήρες υγρασίας εδάφους βρίσκονται αντιδιαμετρικά στο φυτό και προετοιμάσαμε τις δύο πλευρές έστω η πρώτη (Soil_moisture 1) να είναι νωπή σχετικά, ενώ η δεύτερη (Soil_moisture 2) να είναι ξερή, ώστε να αντιληφθούμε το εύρος τιμών που προκύπτουν αμέσως μετά το πότισμα, δεδομένων 2 διαφορετικών συνθηκών στο χώμα. Το πείραμα αυτό έγινε μεσημέρι και ώρες 16:00 – 16:40, η θερμοκρασία ήταν 30.50 ºC , η υγρασία αέρα στο 34% ± 2% και ο δείκτης UV είχε την τιμή 11 (υπενθυμίζουμε max = 12), συνεπώς ήταν μια ηλιόλουστη μέρα.




*Διάγραμμα 41: IoT πείραμα 40 λεπτών*

Αυτό που παρατηρούμε από το διάγραμμα είναι ότι η ξηρή μεριά είχε αναμενόμενα την μεγαλύτερη μείωση στον αριθμό της μέτρησης της υγρασίας εδάφους, καθώς συντελέστηκε εντονότερη αλλαγή, ενώ η ήδη νωπή μεριά είχε σαφώς μικρότερη διακύμανση. Παρατηρούμε ότι η μέση τιμή μετά το πότισμα υπολογίζεται περίπου στο 330 (wet = 310, dry = 350) κάτι που είναι απαραίτητο για την υλοποίηση του κύκλου ημέρας-νύχτας του φυτού, διότι επιλέγοντας σωστά το όριο, επωφελείται ο χρήστης που γνωρίζει πλέον πότε είναι ποτισμένο το φυτό ώστε να λάβει το αντίστοιχο μήνυμα και να μην σπαταλήσει περισσότερο νερό. Ακόμη, από αρκετά αντίστοιχα πειράματα παρατηρήσαμε ότι η ανάγκη του φυτού για νερό προκύπτει όταν ο μέσος όρων των 2 αισθητήρων υγρασίας εδάφους ξεπεράσει το 390, κάτι το οποίο είναι προφανώς πειραματικό και διαφέρει από φυτό σε φυτό. Καθ’ όλη την διαδικασία η κατανάλωση ρεύματος του Arduino ήταν 134 ± 2 mA.

Τέλος, το τελευταίο και μεγαλύτερο πείραμα που υλοποιήσαμε ήταν ο κύκλος ημέρας-νύχτας του φυτού όπου ξεκίνησε περίπου στις 16:00 το μεσημέρι και ολοκληρώθηκε στις 18:30 της επόμενης ημέρας με ένα πότισμα γύρω στις 08:00 το πρωί, καθώς λαμβάναμε μετρήσεις κάθε 15 λεπτά. Ο δείκτης UV μας δείχνει έμμεσα την ώρα της ημέρας, καθώς γνωρίζουμε ότι λαμβάνει την τιμή 0, με την απουσία του φωτός. Τα χαρακτηριστικά και η πληροφορία που μπορούμε να εξάγουμε από το παρακάτω διάγραμμα προφανώς δεν περιορίζεται στην ώρα της ημέρας. Παρατηρούμε αρχικά ότι το απότιστο φυτό μας επιστρέφει μετρήσεις υγρασίας εδάφους με μέση τιμή περίπου 390, η μέση τιμή των αισθητήρων έπεσε στο 320 και σταδιακά άρχισε να ξανανεβαίνει. Πολύ σημαντικό εποπτικό στοιχείο είναι η διακύμανση της θερμοκρασίας και της υγρασίας αέρα, καθώς τις βραδινές ώρες υπήρξε χαμηλότερη θερμοκρασία και υψηλότερη υγρασία, κάτι που εκτός από γενική αίσθηση, εδώ παρουσιάζεται με ακριβείς τιμές. Συνεπώς, ο εκάστοτε χρήστης που θεωρούμε ότι έχει τις σχετικές γνώσεις να αξιοποιήσει αυτές τις μετρικές μπορεί να βελτιστοποιήσει την εξέλιξη των φυτών του, παρακολουθώντας τις διάφορες μετρικές και έχοντας εποπτικά διαγράμματα καθημερινά, ώστε να επιλέξει αφενός τις καταλληλότερες συνθήκες ανά φυτό, αφετέρου να βελτιστοποιήσει την διαχείριση των πόρων, στην συγκεκριμένη περίπτωση του νερού, χωρίς να το σπαταλάει ούτε άσκοπα ούτε υπερβολικά. Αξίζει να τονισθεί και εδώ ότι η κατανάλωση ρεύματος του Arduino ήταν πολύ χαμηλή μη ξεπερνώντας τα 138 mA. Το φυτό όπως γνωρίζουμε εκτελεί τις απαραίτητες λειτουργίες του, όπως η εξατμισοδιαπνοή. Αυτή είναι η απώλεια του νερού που επιτυγχάνεται με την εξάτμιση από την επιφάνεια του εδάφους και των φυτών με την συνδυασμένη απώλεια νερού μέσω της φυτοκόμης (διαπνοής). Προφανώς λοιπόν ένας γεωργός ή κάποιος χρήστης που έχει τις απαραίτητες σχετικές γνώσεις σαφώς περισσότερο από εμάς, μπορεί να λάβει καίριες αποφάσεις για την σχεδίαση συστημάτων άρδευσης, του προγραμματισμού της, ακόμη και να εξάγει υδρολογικές μελέτες. Η δύναμη των εφαρμογών του Διαδικτύου των Αντικειμένων καθίσταται πλέον ξεκάθαρη, καθώς παρέχει στον τελικό χρήστη πανίσχυρα εποπτικά εργαλεία για σύνθετες εργασίες με πολύ χαμηλό κόστος και με ελάχιστη ενεργειακή κατανάλωση. Ταυτόχρονα, αυτά τα συστήματα είναι απόλυτα επεκτάσιμα και μπορούν να χρησιμοποιηθούν για κάθε άλλο φυτό και ως μια συστάδα μπορούν να δημιουργήσουν ένα ισχυρό cluster που να δίνει στον τελικό χρήστη την εποπτεία όχι μόνο ενός φυτού, αλλά μιας καλλιέργειας.




*Διάγραμμα 42: IoT οικοσύστημα, πείραμα πλήρους κύκλου ημέρας-νύχτας*

Σε αυτό το σημείο, πριν κλείσουμε το 6ο κεφάλαιο της πειραματικής αξιολόγησης ας κάνουμε μια σύντομη σύνοψη των πειραματικών ευρημάτων μας για τις 2 εφαρμογές που παρουσιάσθηκαν. Ξεκινώντας από την εφαρμογή με την επεξεργασία εικόνας και κατηγοριοποίησης με βάση ασθένειες φύλλων χρησιμοποιήσαμε αρχικά με την βοήθεια της βιβλιοθήκης της python Pillow, έναν προσαρμοσμένο τρόπο φόρτωσης των εικόνων προς επεξεργασία μία προς μία, κάτι που επέφερε την χρήση λίγων υπολογιστικών πόρων ταυτόχρονα με την αργή εκτέλεση της διεργασίας σε όλα τις συσκευές μας (RPi 3, RPi 4, Jetson Nano και Google Coral). Ακολούθως, αξιοποιώντας τον ImageDataGenerator που μας επιτρέπει την διαχείριση των εικόνων ανά ομάδες (batch) πειραματιστήκαμε με διαφορικές τιμές πλήθους ανά ομάδα τόσο στο Raspberry Pi 3, όσο και στο Raspberry Pi 4. Οι τιμές που ελέγχθηκαν ήταν 2, 4, 8 και 16 φτάνοντας το RPi 3 στα όρια του. Για το RPi 4 επιδιώξαμε και την δοκιμή πλήθους ομάδας ίσο με 32, όπου καθώς φάνηκε και από τα διαγράμματα όσα αυξάναμε το πλήθος τόσο πιο απαιτητική γινόταν η διεργασία σε υπολογιστικούς και

ενεργειακούς πόρους, κερδίζοντας ωστόσο σε χρόνο εκτέλεσης, καθώς γινόντουσαν λιγότερες επαναλήψεις και εκμεταλλευόμασταν όλο και περισσότερο την υπολογιστική δύναμη των μηχανημάτων μας. Οι μετρικές που μας απασχόλησαν για την σύγκριση των διαφόρων πειραμάτων ήταν η χρήση της επεξεργαστικής ισχύος, η χρήση της μνήμης RAM και της μνήμης Swap που απαιτήθηκε στο RPi 3, η εσωτερική θερμοκρασία των μηχανημάτων και φυσικά η ενεργειακή τους κατανάλωση (ρεύμα). Για την δεύτερη εφαρμογή τώρα, παρουσιάσαμε το δικό μας λειτουργικό και εύκολα επεκτάσιμο IoT οικοσύστημα που συμβάλλει τόσο στην εποπτική διαχείριση όσο και την δυνατότητα ανθρώπινης παρέμβασης για την βελτιστοποίηση της διαχείρισης των πόρων κατά την διαδικασία του ποτίσματος. Το οικοσύστημα μας αποτελούνταν όπως είδαμε και παραπάνω από το Arduino που αποτέλεσε τον εγκέφαλο και τον διαχειριστή του όλου συστήματος, τους αισθητήρες που ήταν συνδεδεμένοι πάνω του, τα Xbee modules που κατέστησαν δυνατή την ασύρματη μετάδοση σε έναν απομακρυσμένο υπολογιστή (RPi 4) και τέλος το RPi 4 που συγκέντρωνε όλη την απαραίτητη πληροφορία. Τα δεδομένα συνεπώς δημιουργούνταν με την βοήθεια των αισθητήρων στο φυτό μας, ακολούθως «περνούσαν» από το Arduino που μέσω του Xbee τα μετέδιδε ασύρματα στο RPi 4 ή σε οποιονδήποτε άλλο υπολογιστή επιθυμούσαμε, για την συλλογή, αποθήκευση, διαχείριση και μετέπειτα επεξεργασία τους. Το πιο σημαντικό από όλα σε αυτή την IoT εφαρμογή είναι η δυνατότητα εποπτείας σε πραγματικό χρόνο και η επιλογή του αντίστοιχου mode που επιτρέπει συχνότερες μετρήσεις όποτε ο γεωργός ή ο εκάστοτε χρήστης το επιθυμεί για να έχει μια προσαρμοσμένη εμπειρία τόσο εποπτείας όσο και ενέργειας (εδώ το πότισμα του φυτού). Για τον λόγο αυτό, παρουσιάστηκαν δυο πειράματα, ένα σύντομων μετρήσεων για την λήψη οριακών τιμών που τις καθιστούν τιμές απόφασης και ένα πείραμα που καταδείκνυε τον κύκλο ημέρας-νύχτας του φυτού και συνεπώς δίνει στα χέρια του χρήστη ένα πανίσχυρο εργαλείο με πολύ εύκολη εποπτεία, χωρίς να απαιτείται η συνεχής παρακολούθηση του φυτού από κοντά, καθώς όλες οι απαραίτητες μετρήσεις εμφανίζονται στην οθόνη του.

# Κεφάλαιο 7 – Συμπεράσματα και Μελλοντική Μελέτη

## 7.1 Ανασκόπηση και Συμπεράσματα

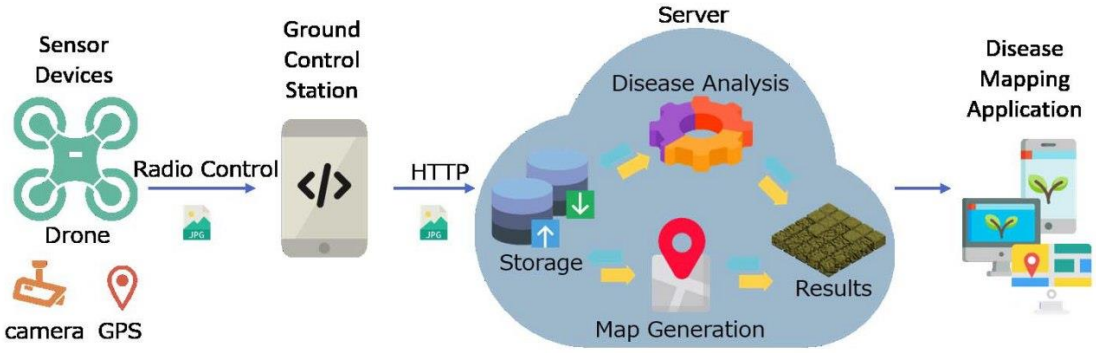
Κλείνοντας την παρούσα διπλωματική εργασία θεωρούμε ότι παρουσιάσαμε 2 αξιόλογες εφαρμογές στον τομέα της Γεωργίας που επιδιώκουν να βοηθήσουν τον άνθρωπο να βελτιώσει την διαχείριση των πόρων και την λήψη των αποφάσεων. Αρχικά, αφού αναλύθηκε το σχετικό πλαίσιο της μηχανικής μάθησης με νευρωνικά δίκτυα, παρουσιάσαμε μια εφαρμογή επεξεργασίας εικόνας και κατηγοριοποίησης ανάμεσα σε 33 κλάσεις για ασθένειες φύλλων με ακρίβεια περίπου 90%. Συγκρίνοντας με αποτελέσματα από σχετικές δημοσιεύσεις που μελετήσαμε το ποσοστό ήταν αρκετά καλό, αν και όχι το καλύτερο, ωστόσο επιδιώξαμε την κατηγοριοποίηση ανάμεσα σε πολλές ετικέτες, κάτι που καθιστά την εργασία μας ιδιαιτέρως δύσκολη, καθώς όσες μελέτες είχαν score ακρίβειας άνω του 90% χρησιμοποιούσαν πολύ λιγότερες κατηγορίες. Ακόμη, δείξαμε πως με την βοήθεια των generators και του data augmentation μπορούμε να δώσουμε μια IoT «γεύση» στην διαδικασία του training, επιδιώκοντας με το ίδιο πλήθος των λίγων φωτογραφιών που είχαμε στην διάθεση μας, να εκπαιδεύσουμε το μοντέλο μας σε ρεαλιστικότερες συνθήκες. Τέλος, για το κομμάτι της επεξεργασίας εικόνας παρουσιάσαμε διαφορετικά πειράματα με βάση τις IoT συσκευές μας, όπου συγκρίναμε τα αποτελέσματα τους, ενώ παράλληλα καταγράφαμε και τις αντίστοιχες καταναλώσεις ρεύματος και ισχύος, καθότι αν επρόκειτο να δημιουργηθεί μια μεγαλύτερη συστάδα από τέτοιες συσκευές θα υπάρχει φυσικά ο ενεργειακός περιορισμός των πόρων. Εν γένει, είδαμε ότι όσο θέλουμε να επιταχύνουμε τις διαδικασίες τόσο πιο κοστοβόρες γίνονται οι εφαρμογές, συνεπώς είναι και στο χέρι του χρήστη να επιλέξει τον κατάλληλο συνδυασμό χρόνου εκτέλεσης και κόστους πόρων. Ακολούθως, παρουσιάσαμε μια εφαρμογή όπου δημιουργήσαμε το δικό μας IoT οικοσύστημα, με το Arduino σε ρόλο διαχειριστή του συστήματος, τους διάφορους αισθητήρες που χρησιμοποιήθηκαν, το Xbee που επέτρεψε την επικοινωνία με έναν απομακρυσμένο υπολογιστή, τον ρόλο του οποίου έπαιξε το Raspberry Pi 4. Τα αποτελέσματα του IoT οικοσυστήματος είναι ιδιαίτερα ενθαρρυντικά καθώς αφού ολοκληρώθηκε το στήσιμο του όλου συστήματος, κατόπιν ήταν πολύ εύκολη η παρακολούθηση σε πραγματικό χρόνο των μετρήσεων και η μετέπειτα επεξεργασία τους, ώστε να λαμβάνει ο τελικός χρήστης εποπτικά αποτελέσματα που συμπυκνώνουν πολύ πληροφορία. Παράλληλα, προτείναμε και ένα Irrigation Mode, όπου ο χρήστης μπορεί να επιλέξει να εντείνει τις μετρήσεις, ώστε να έχει ακόμη περισσότερη ακρίβεια ως προς την πληροφορία που λαμβάνει, όταν επιλέξει να ποτίσει το φυτό.

Η συμβολή συνεπώς της παρούσας διπλωματικής σχετικά με την υφιστάμενη βιβλιογραφία είναι ουσιαστικά η επέκταση εφαρμογών που έχουν ως τώρα υλοποιηθεί πετυχαίνοντας εξίσου καλά αποτελέσματα. Στο κομμάτι του Image Processing οι εφαρμογές που παρατηρήσαμε ότι έχουν υλοποιηθεί ως τώρα αναφέρονται κυρίως σε ένα είδος φύλλου, όπου χωρίζεται σε βασικές κατηγορίες ασθενειών. Η συμβολή μας εδώ είναι ότι χρησιμοποιούμε συνολικά διαφορετικά φύλλα που χωρίζονται σαφώς σε

περισσότερες κατηγορίες, απ’ ότι να το αντιμετωπίζαμε ανά φύλλο, καθώς στην φωτογραφία μιας καλλιέργειας όπου υπάρχουν διαφορετικά είδη φυτών απαιτείται η γενικότερη αντιμετώπιση. Ταυτόχρονα υποδείξαμε την σημασία της IoT «γεύσης» στην επεξεργασία εικόνας με την επεξεργασία των εικόνων ώστε να προσομοιάζουν ένα ρεαλιστικό παράδειγμα και όχι απλά καλοστημένες εικόνες με τέλεια φωτεινότητα και λοιπά χαρακτηριστικά, ώστε το μοντέλο μας να είναι πολύ αποδοτικό στο πραγματικό prediction. Στο δεύτερο κομμάτι του IoT οικοσυστήματος, ουσιαστικά επεκτείναμε μια κλασσική εφαρμογή παρακολούθησης ατμοσφαιρικών δεδομένων με την χρήση αισθητήρων, καθώς όχι μόνο τα λαμβάναμε απομακρυσμένα και σε πραγματικό χρόνο, αλλά ταυτόχρονα προτείνουμε και μια κατάσταση λειτουργίας (Irrigation Mode) που μπορεί ο χρήστης να αξιοποιήσει προσαρμοσμένα στις ανάγκες του κάθε ξεχωριστού φυτού, με στόχο να βελτιστοποιήσει την διαχείριση των πόρων (στην περίπτωση μας του νερού) αλλά και την εποπτική παρακολούθηση λαμβάνοντας συμπυκνωμένα όλη την ουσιαστική πληροφορία στην οθόνη του.

## 7.2 Μελλοντική Μελέτη

Όσον αφορά την μελλοντική μελέτη θα θέλαμε να επιτύχουμε έναν συνδυασμό μεταξύ των δύο εφαρμογών στο ίδιο σύστημα, βοηθώντας ακόμη περισσότερο τον τελικό χρήστη την λήψη διαφόρων αποφάσεων και στην έγκαιρη παρακολούθηση των διάφορων γεωργικών διεργασιών. Αναλυτικότερα, επιθυμούμε να επεκτείνουμε την εφαρμογή του Image Classification, αξιοποιώντας την δυνατότητα ενός UAV Drone να αποστέλλει real-time εικόνες από την περιπολία του για παράδειγμα πάνω από μια αγροτική περιοχή, τις οποίες θα λαμβάνει η εκάστοτε απομακρυσμένη IoT συσκευή και θα χρησιμοποιεί τον αλγόριθμο κατηγοριοποίησης για να αποφανθεί για τις ασθένειες των φύλλων, κάτι που επιθυμούμε να συνδυαστεί με τις διάφορες ατμοσφαιρικές και γεωργικές μετρήσεις που προσφέρει το λοιπό IoT οικοσύστημα, δίνοντας στα χέρια του ανθρώπου πολλές συμπυκνωμένες αλλά επαρκείς πληροφορίες για να λάβει τις ορθότερες αποφάσεις. Όπως, έχουμε αναδείξει καθ’ όλη την παρούσα διπλωματική εργασία, το Διαδίκτυο των Αντικειμένων είναι ιδιαίτερα ευέλικτο και επεκτάσιμο. Συνεπώς, μια ακόμη επέκταση που σκεφτόμαστε να μελετήσουμε, είναι να εκπαιδεύσουμε τον αλγόριθμο του Image Classification με εικόνες που αναφέρονται στην άρδευση ενός γεωργικού χώρου και ταυτόχρονα να υλοποιήσουμε την κατασκευή του Arduino μαζί με τους αισθητήρες σε ένα κλειστό κουτί, ώστε να γίνει φορητό καθώς θα έχει πλέον εσωτερικά την καλωδίωση, αξιοποιώντας περισσότερους αισθητήρες και ένα σύστημα αυτόματου ελέγχου ποτίσματος, όπως μια ηλεκτροβάνα.




*Εικόνα 24: Η ενοποίηση των εφαρμογών και η μελλοντική μελέτη*

# Κεφάλαιο 8 – Βιβλιογραφία

[1] Lee In, Lee Kyoochun. The Internet of Things (IoT) : applications, investments, and challenges for enterprises, Business horizons. - Amsterdam : Elsevier, ISSN 0007-6813, ZDB-ID 222663-7. - Vol. 58, No. 4, p. 431-440, 2015.

[2] Brewster, C., Roussaki, I., Kalatzis, N., Doolin, K., & Ellis, K. (2017). IoT in Agriculture: Designing a Europe-Wide Large-Scale Pilot. IEEE Communications Magazine, 55(9), 26-33. doi:10.1109/mcom.2017.1600528

[3] Jaiganesh, S., Gunaseelan, K., & Ellappan, V. (2017). IOT agriculture to improve food and farming technology. 2017 Conference on Emerging Devices and Smart Systems (ICEDSS). doi:10.1109/icedss.2017.8073690

[4] Kim, Heehoon & Nam, Hyoungwook & Jung, Wookeun & Lee, Jaejin. (2017). Performance analysis of CNN frameworks for GPUs. 55-64. 10.1109/ISPASS.2017.7975270.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in Advances in neural information processing systems, 2012, pp. 1097–1105.

[6] O. R. et al., "Imagenet large scale visual recognition challenge," CoRR, vol. abs/1409.0575, 2014. [Online]. Available: http://arxiv.org/abs/1409.0575

[7] Hara, K., Saito, D., & Shouno, H. (2015). Analysis of function of rectified linear unit used in deep learning. 2015 International Joint Conference on Neural Networks (IJCNN). doi:10.1109/ijcnn.2015.7280578

[8] L. Bottou, "Online algorithms and stochastic approximations," in Online Learning and Neural Networks, D. Saad, Ed. Cambridge, UK: Cambridge University Press, 1998, revised, oct 2012. [Online]. Available: http://leon.bottou.org/papers/bottou-98x

[9] Munro P. (2011) Backpropagation. In: Sammut C., Webb G.I. (eds) Encyclopedia of Machine Learning. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-30164-8_51

[10] Arduino website available at https://www.arduino.cc/

[11] Xbee site available at https://www.digi.com/xbee

[12] Adafruit site for Xbee available at https://www.adafruit.com/product/967

[13] Adafruit site for temperature-humidity sensor available at https://www.adafruit.com/product/393

[14] Dfrobot site for soil moisture sensor available at https://wiki.dfrobot.com/Capacitive_Soil_Moisture_Sensor_SKU_SEN0193

[15] Adafruit site for UV light sensor available at https://learn.adafruit.com/adafruit-veml6070-uv-light-sensor-breakout

[16] Raspberry Pi site for our devices available at https://www.raspberrypi.org/products/

[17] Nvidia® Jetson Nano™ site available at https://developer.nvidia.com/embedded/jetson-nano-developer-kit

[18] Google Coral dev board site available at https://coral.ai/products/dev-board/

[19] Douarre C, Schielein R, Frindel C, Gerth S and Rousseau D(2016) Deeplearning based root-soil segmentation from X-ray tomography.bioRxiv,071662. doi:https://doi.org/10.1101/071662.

[20] Chen SW, Shivakumar SS, Dcunha S, Das J, Okon E, Qu C, Taylor CJ andKumar V(2017) Counting apples and oranges with deep learning: a data-driven approach.IEEE Robotics and Automation Letters2, 781–788.

[21] Rahnemoonfar M and Sheppard C(2017) Deep count: fruit counting basedon deep simulated learning.Sensors17, 905.

[22] Reyes AK, Caicedo JC and Camargo JE(2015) Fine-tuning deep convolutional networks for plant recognition. In Cappellato L, Ferro N,Jones GJF and San Juan E (eds),CLEF2015 Working Notes. WorkingNotes of CLEF 2015–Conference and Labs of the Evaluation Forum,Toulouse, France, September 8–11, 2015. Toulouse: CLEF. Available online at :http://ceur-ws.org/Vol-1391

[23] Sladojevic S, Arsenovic M, Anderla A, Culibrk D and Stefanovic D(2016)Deep neural networks based recognition of plant diseases by leaf image classification.Computational Intelligence and Neuroscience2016, 3289801.

[24] K, R., & B, S. (2015). Application of Image Processing Techniques in Plant Disease Recognition. *International Journal of Engineering Research and, V4*(03). doi:10.17577/ijertv4is030829

[25] Mohanty SP, Hughes DP and Salathé M(2016) Using deep learning forimage-based plant disease detection.Frontiers in Plant Science7, 1419.doi: 10.3389/fpls.2016.01419.

[26] Amara J, Bouaziz B and Algergawy A(2017) A deep learning-basedapproach for banana leaf diseases classification. In Mitschang B (ed.),Datenbanksysteme für Business, Technologie und Web (BTW 2017)–Workshopband. Lecture Notes in Informatics (LNI). Stuttgart, Germany: Gesellschaft für Informatik, pp. 79–88.

[27] Kussul N, Lavreniuk M, Skakun S and Shelestov A(2017) Deep learning classification of land cover and crop types using remote sensing data. IEEE Geoscience and Remote Sensing Letters14, 778–782.

[28] Grinblat GL, Uzal LC, Larese MG and Granitto PM(2016) Deep learning for plant identification using vein morphological patterns. Computers and Electronics in Agriculture127, 418–424.

[29] Kuwata K and Shibasaki R(2015) Estimating crop yields with deep learning and remotely sensed data. In IEEE International Geoscience and Remote Sensing Symposium (IGARSS). Milan, Italy: IEEE, pp. 858–861.

[30] Bargoti S and Underwood J(2017) Deep fruit detection in orchards. In Okamura A (ed.),2017 IEEE International Conference on Robotics and Automation (ICRA). Piscataway, NJ, USA: IEEE, pp. 3626–3633.

[31] Sa I, Ge Z, Dayoub F, Upcroft B, Perez T and McCool C(2016) Deepfruits: a fruit detection system using deep neural networks.Sensors16, E1222.

[32] Tyagi AC(2016) Towards a second green revolution.Irrigation and Drainage65, 388–389.

[33] Hashem IAT, Yaqoob I, Anuar NB, Mokhtar S, Gani A and Khan S(2015)The rise of big data on cloud computing: review and open research issues. Information Systems47,98–115

[34] Kamilaris A, Kartakoullis A and Prenafeta-Boldú FX(2017) A review on the practice of big data analysis in agriculture. Computers and Electronics inAgriculture143,23–37.

[35] Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., & Ayyash, M. (2015). Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications. *IEEE Communications Surveys & Tutorials, 17*(4), 2347-2376. doi:10.1109/comst.2015.2444095

[36] Mekala, M. S., & Viswanathan, P. (2017). A novel technology for smart agriculture based on IoT with cloud computing. *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*. doi:10.1109/i-smac.2017.8058280

[37] Y. Kim, R. Evans and W. Iversen, "Remote Sensing and Control of an Irrigation System Using a Distributed Wireless Sensor Network" ,IEEE Transactions on Instrumentation and Measurement, pp. 1379–1387, 200

[38] Lutfull Karim C. and Alagan Anpalagan., "sensor–based M2Magriculture monitoring system for developing countries: state and challenges," In Proceedings of the 2006 London Communications Symposium, Network Protocols and Algorithms ISSN 1943-35812013, Vol. 5, No.

[39] Khelifi, F. (2020). Monitoring System Based in Wireless Sensor Network for Precision Agriculture. *Internet of Things (IoT),* 461-472. doi:10.1007/978-3-030-37468-6_24

[40] Dong, Y., Fu, Z., Peng, Y., Zheng, Y., Yan, H., & Li, X. (2020). Precision fertilization method of field crops based on the Wavelet-BP neural network in China. *Journal of Cleaner Production, 246*, 118735. doi:10.1016/j.jclepro.2019.118735

[41] Dataset link available at https://github.com/spMohanty/PlantVillage-Dataset

[42] Chen, N., & Blostein, D. (2006). A survey of document image classification: Problem statement, classifier architecture and performance evaluation. *International Journal of Document Analysis and Recognition (IJDAR), 10*(1), 1-16. doi:10.1007/s10032-006-0020-2