ΠΑΝΕΠΙΣΤΗΜΙΟ
ΙΩΑΝΝΙΝΩΝ

Τμήμα Πληροφορικής και Τηλεπικοινωνιών

ΚΑΤΑΝΕΜΗΜΕΝΕΣ ΤΕΧΝΙΚΕΣ ΜΗΧΑΝΙΚΗΣ ΜΑΘΗΣΗΣ ΚΑΙ ΕΦΑΡΜΟΓΕΣ ΤΟΥΣ

Επιμέλεια: Περσεφόνη Λάππα

Επιβλέπων καθηγητής: Χρυσόστομος Στύλιος

ΑΡΤΑ 2022

Περίληψη

Ο αριθμός των συνδεμένων συσκευών στο διαδίκτυο αυξάνεται συνεχώς και έχει ήδη ξεπεράσει τον αριθμό των ανθρώπων που κατοικούν στον πλανήτη. Οι συσκευές αυτές παράγουν δεδομένα τα οποία διακινούνται μέσω του δικτύου με προορισμό κάποια ισχυρή υπολογιστική δομή η οποία θα τα επεξεργαστεί και ενδεχομένως να τα αποθηκεύσει για μελλοντική χρήση. Η πρακτική αυτή έχει αυξήσει ιδιαίτερα τον όγκο των δεδομένων που διακινούνται μέσω του δικτύου και παρά τις προόδους για την αύξηση της χωρητικότητας του δικτύου, η αύξηση του όγκου των δεδομένων είναι δυσανάλογα μεγαλύτερη από την αύξηση της χωρητικότητας του δικτύου με αποτέλεσμα να δημιουργούνται αρκετά προβλήματα. Επιπλέον αρκετές συσκευές που παράγουν δεδομένα βρίσκονται σε σημεία που δεν επιτρέπουν τη χρήση των παραδοσιακών μεθόδων διασύνδεσης στο δίκτυο δεδομένων και τροφοδοσίας ηλεκτρικής ενέργειας με αποτέλεσμα να πρέπει να χρησιμοποιηθούν άλλες μέθοδοι που ανεβάζουν αρκετά το κόστος μεταφοράς των δεδομένων.

Μια προσέγγιση προς την αντιμετώπιση του ζητήματος αυτού επιτυγχάνεται με τη χρήση τεχνικών μηχανικής μάθησης μέσω των οποίων οι συσκευές θα εκπαιδευτούν ώστε να μπορούν να επεξεργάζονται τα δεδομένα και συνεπώς να αποστέλλουν στην ισχυρή υπολογιστική δομή μόνο τα απαραίτητα στοιχεία αντί του συνόλου των δεδομένων. Πράγματι η εξέλιξη της τεχνολογίας επιτρέπει πλέον την παραγωγή αρκετά έξυπνων συσκευών σε διαχειρίσιμο κόστος οι οποίες καταφέρνουν να συγκεντρώσουν σε μικρές διαστάσεις ένα ικανοποιητικό σύνολο υπολογιστικών πόρων και να λειτουργούν καταναλώνοντας μικρές ποσότητες ενέργειας, όπως για παράδειγμα το Raspberry Pi. Συνεπώς οι συσκευές αυτές θα μπορούσαν να εκπαιδευτούν ώστε να είναι σε θέση να επεξεργαστούν τα δεδομένα στην πηγή και συνεπώς να μειώσουν σημαντικά τον όγκο των δεδομένων που μεταδίδονται.

Η εκπαίδευση όμως μιας συσκευής και πιο συγκεκριμένα η παραγωγή ενός μοντέλου μηχανικής μάθησης είναι μια διαδικασία ιδιαίτερα απαιτητική σε υπολογιστικούς πόρους αλλά και σε ενέργεια και συνεπώς δεν είναι δυνατόν να πραγματοποιηθεί από συσκευές τύπου Raspberry αλλά μόνο από ισχυρές υπολογιστικές δομές που διαθέτουν εύκολα και οικονομικά πρόσβαση τόσο σε ενέργεια όσο και στο διαδίκτυο. Συνεπώς η πλέον αποδοτική αρχιτεκτονική αποδεικνύεται εκείνη σύμφωνα με την οποία το μοντέλο μηχανικής μάθησης παράγεται σε μια ισχυρή υπολογιστική δομή και κατόπιν αποστέλλεται στους επιμέρους κόμβους οι οποίοι θα το χρησιμοποιήσουν για να μελετήσουν τα δεδομένα που διαθέτουν και να εξάγουν τα απαραίτητα συμπεράσματα τα οποία στη συνέχεια θα αποσταλούν στον οποιοδήποτε ενδιαφερόμενο.

Στόχος της παρούσας πτυχιακής εργασίας είναι η διερεύνηση των μεθόδων κατανεμημένης μηχανικής μάθησης καθώς και διαφόρων εφαρμογών της, ώστε να αποκτήσει ο αναγνώστης μια όσο το δυνατόν πληρέστερη εικόνα της κατάστασης όπως έχει δημιουργηθεί έως το διάστημα συγγραφής της παρούσας εργασίας. Παράλληλα παρουσιάζεται και η ανάπτυξη μιας εφαρμογής αναγνώρισης χειρόγραφων χαρακτήρων, σε γλώσσα προγραμματισμού Python, η οποία

περιλαμβάνει την δημιουργία ενός μοντέλου σε ένα υπολογιστή και ακολούθως αποστολή του σε μία συσκευή Raspberry προκειμένου να το χρησιμοποιήσει. Τα αποτελέσματα αποδεικνύουν την χρησιμότητα της αρχιτεκτονικής.

Λέξεις Κλειδιά: Κατανεμημένη μηχανική μάθηση, Διαδίκτυο των πραγμάτων, Edge Computing, Raspberry Pi, Python

Abstract

The number of devices connected to the internet is constantly increasing and has already exceeded the number of people living on the planet. These devices generate data that travels over the network to a powerful computing structure that will process it and possibly store it for future use. This practice has greatly increased the volume of data flowing through the network and despite advances in increasing network capacity, the increase in data volume is disproportionately greater than the increase in network capacity resulting in several problems. In addition, several data generators are located in areas that do not allow the use of traditional methods of interconnection in the data network and electricity supply, so other methods must be used that significantly increase the cost of data transfer.

An approach to tackling this issue is achieved through the use of machine learning techniques through which devices will be trained to be able to process data and therefore send to the high computational ability computing structure only the necessary data instead of the total data. Indeed, the advancement of technology now allows the production of quite smart devices at manageable costs which manage to accumulate, in a small size, a sufficient set of computing resources and operate by consuming small amounts of energy, such as, for example, the Raspberry Pi. These devices could therefore be trained to be able to process data at source and thus significantly reduce the amount of data transmitted.

However, the training of a device and more specifically the production of a machine learning model is a very demanding process in terms of computing resources but also in terms of energy and therefore it cannot be done by Raspberry type devices but only by powerful computing structures that has easy and economically access to energy as well as to the internet. Therefore the most efficient architecture proves to be the one according to which the machine learning model is produced in a strong computational structure and then sent to the individual nodes which will use it to study the data they have and draw the necessary conclusions which will then be sent to any interested party.

The aim of this dissertation is to explore the methods of distributed machine learning as well as its various applications, so that the reader can obtain as complete a picture of the situation as has been created up to the time of writing this dissertation. At the same time, the development of a

handwriting recognition application in the Python programming language is presented, which includes creating a model on a computer and then sending it to a Raspberry device in order to use it. The results prove the usefulness of the architecture.

Key Words: Distributed Machine Learning, IoT, Edge Computing, Raspberry Pi, Python

Πίνακας Περιεχομένων

Πίνακας Εικόνων

Κεφάλαιο 1: Κατανεμημένη Μηχανική Μάθηση

1. Εισαγωγή

Ο όρος κατανεμημένη μηχανική μάθηση αναφέρεται σε μια σειρά αλγορίθμων και συστημάτων μηχανικής μάθησης πολλαπλών κόμβων που έχουν σχεδιαστεί με στόχο τη βελτίωση της απόδοσης και την αύξηση της ακρίβειας στις περιπτώσεις που ο όγκος των δεδομένων εισόδου είναι αρκετά μεγάλος. Η αύξηση του μεγέθους των δεδομένων εισόδου για πολλούς αλγόριθμους μπορεί να μειώσει σημαντικά τα σφάλματα και μπορεί συχνά να είναι πιο αποτελεσματικό ακόμα και από τη χρήση πιο σύνθετων μεθόδων [Halevy et al, 2009]. Η κατανεμημένη μηχανική μάθηση επιτρέπει στους ερευνητές να λαμβάνουν τεκμηριωμένες αποφάσεις και να βγάζουν ουσιαστικά συμπεράσματα από μεγάλες ποσότητες δεδομένων.

Υπάρχουν πολλά συστήματα που χρησιμοποιούνται για την εκτέλεση εργασιών μηχανικής μάθησης σε κατανεμημένο περιβάλλον. Αυτά τα συστήματα μπορούν να χωριστούν σε τρεις βασικές κατηγορίες: βάσεις δεδομένων, γενικά και ειδικά σχεδιασμένα συστήματα. Κάθε τύπος συστήματος έχει ξεχωριστά πλεονεκτήματα και μειονεκτήματα, αλλά όλα χρησιμοποιούνται ανάλογα με τις απαιτήσεις απόδοσης και τα μεγέθη δεδομένων εισόδου.

Η εκτέλεση αλγορίθμων μηχανικής μάθησης σε ενσωματωμένα συστήματα είναι ένας μεγάλος τομέας μελέτης και παραμένει δύσκολο να βρεθούν κατάλληλες λύσεις για κάθε περίπτωση χρήσης. Υπάρχουν πολλές επιλογές για το υλικό, το λογισμικό και τη σύνδεση μεταξύ τους. Οι κύριοι περιορισμοί για τη λειτουργία μοντέλων μηχανικής μάθησης σε ενσωματωμένα συστήματα είναι οι εξής [Arm, 2019]:

- Απόδοση: Το ενσωματωμένο μοντέλο μηχανικής μάθησης πρέπει να είναι όσο το δυνατόν γρηγορότερο, να απαντά με σχεδόν μηδενική καθυστέρηση και με την υψηλότερη δυνατή ακρίβεια. Οι περισσότερες ενσωματωμένες συσκευές δεν έχουν πρόσβαση στο δίκτυο τροφοδοσίας, επομένως λειτουργούν από την μπαταρία.

Επομένως, είναι σημαντικό για την εφαρμογή να χρησιμοποιεί την ελάχιστη δυνατή ισχύ για την εξασφάλιση μιας μακροχρόνιας απόδοσης.

- Κόστος: περιλαμβάνει το κόστος κατασκευής μιας μονάδας, το κόστος της ενέργειας, το κόστος της συνδεσιμότητας του δικτύου και το κόστος του νέφους (cloud) και περιορίζεται από τα προσβάσιμα μέσα.

Για να ξεπεραστούν οι προαναφερθέντες περιορισμοί, υπάρχουν πράγματα που μπορούν να βελτιώσουν την απόδοση του ενσωματωμένου λογισμικού μηχανικής μάθησης, ανεξάρτητα από τις επιλογές του υλικού ή τον αλγόριθμο που χρησιμοποιείται για τον υπολογισμό. Στη συνέχεια παρέχεται μια περιγραφή της μηχανικής μάθησης, καθώς και μια σύντομη συζήτηση σχετικά με το edge computing σε σύγκριση με το cloud computing στην περίπτωση ενσωματωμένων συστημάτων που εκτελούν αλγόριθμους μηχανικής μάθησης.

2. Ιστορικό υπόβαθρο

Η ψηφιακή επανάσταση είναι γεγονός. Σήμερα, όλα είναι ψηφιακά και συνδέονται στο Διαδίκτυο [Evans, 2011]. Το έτος 2009, ο αριθμός των συσκευών που συνδέονταν στο Διαδίκτυο ξεπέρασε τον παγκόσμιο πληθυσμό. Ως εκ τούτου, ο αρχικός σχεδιασμός δεν μπορεί πλέον να καλύψει τις ανάγκες και το Διαδίκτυο των πραγμάτων (IoT) γεννήθηκε [Evans, 2011]. Ο αριθμός των συνδεδεμένων στο Διαδίκτυο συσκευών συνεχίζει να αυξάνεται και έως το 2021 ο αριθμός των συσκευών θα ξεπεράσει τα 20 δισεκατομμύρια.

Λόγω όλων αυτών των συσκευών, η κίνηση στο Διαδίκτυο θα φτάσει τα 20,6 Zettabytes. Τα περισσότερα από τα δεδομένα που παράγονται θα είναι άχρηστα ή δεν θα αποθηκευτούν. Επιπλέον, μόνο ένα μικρό κλάσμα θα διατηρηθεί σε βάσεις δεδομένων. Ωστόσο, ακόμη και αυτό το κλάσμα αντιπροσωπεύει περισσότερα από τα δεδομένα που αποθηκεύονται από την Google, το Amazon και το Facebook [Mitchell]. Επειδή οι άνθρωποι δεν μπορούν να παρακολουθήσουν ένα τόσο μεγάλο όγκο δεδομένων, η λύση ήταν να πραγματοποιηθούν αλληλεπιδράσεις μεταξύ μηχανών (M2M). Αυτές οι αλληλεπιδράσεις, ωστόσο, έπρεπε να είναι έξυπνες, για να έχουν υψηλά επίπεδα ανεξαρτησίας.

Για να επιτευχθεί ο στόχος της παροχής μηχανών με ευφυΐα και ανεξαρτησία, αναπτύχθηκε ο τομέας της τεχνητής νοημοσύνης (AI). Μέσα στο πεδίο της τεχνητής νοημοσύνης, τα πεδία της μηχανικής μάθησης (ML) και της βαθιάς μάθησης (DL) έχουν σημειώσει περισσότερες εξελίξεις τις τελευταίες δεκαετίες. Οι αλγόριθμοι μηχανικής μάθησης χρονολογούνται από το 1950 [Buchanan, 2006]. Ωστόσο, το πεδίο της τεχνητής νοημοσύνης άργησε να αναπτυχθεί επειδή οι υπολογιστές δεν ήταν αρκετά ισχυροί για την εκτέλεση των περισσότερων αλγορίθμων ML [Newquist,1994].

Μόλις οι υπολογιστές άρχισαν να έχουν αρκετή μνήμη, ισχυρές κεντρικές μονάδες επεξεργασίας (CPU) και κάρτες γραφικών (GPU) [Janakiram], ξεκίνησε η επανάσταση της AI. Οι υπολογιστές μπόρεσαν να μειώσουν το χρόνο δημιουργίας και εκτέλεσης μοντέλων ML και να χρησιμοποιούν αυτούς τους αλγόριθμους για την επίλυση πολύπλοκων προβλημάτων. Επιπλέον, το cloud computing (Mell, 2011) ενθάρρυνε τη δημιουργία πιο ευέλικτων μοντέλων που παρέχουν όλους τους απαραίτητους πόρους. Το Cloud παρέχει απεριόριστους πόρους με τη χρήση ενός δικτύου υπολογιστών ικανών να διαιρούν τις υπολογιστικές εξαρτήσεις μεταξύ τους, κάνοντας αυτούς τους υπολογιστές να λειτουργούν ως ένας [Mirashe, 2010]. Αυτή η τεχνική κατέστησε δυνατή τη δημιουργία μεγάλων βάσεων δεδομένων και τον χειρισμό μεγάλων συνόλων δεδομένων.

Ωστόσο, για τη δημιουργία ενός καλού μοντέλου ML δεν μετράει τόσο η δύναμη του υπολογιστή, αλλά τα δεδομένα που χρησιμοποιήθηκαν για την κατασκευή του [Domingos, 2012].

Η πρώτη προσέγγιση, για τη δημιουργία ευφυών δικτύων, ήταν η συλλογή τελικών συσκευών με τη χρήση αισθητήρων. Επειδή αυτές οι συσκευές δεν διαθέτουν πόρους, δεν μπορούν να αποθηκεύσουν τα δεδομένα. Μεταδίδουν τα δεδομένα στην συσκευή αιχμής (edge device). Στη συνέχεια, η edge συσκευή θα προωθήσει τα δεδομένα στο Cloud μέσω σύνδεσης στο Διαδίκτυο. Το Cloud θα λάβει τα δεδομένα και θα τα επεξεργαστεί και είτε θα τα αποθηκεύσει είτε θα τα χρησιμοποιήσει για να τροφοδοτήσει ένα μοντέλο ML.

Το κύριο πρόβλημα με ένα σύστημα που εξαρτάται από το cloud είναι ότι βασίζεται σε μια σύνδεση στο Διαδίκτυο για τη μεταφορά δεδομένων. Όταν η σύνδεση καθυστερήσει ή διακοπεί, το σύστημα δεν θα λειτουργεί σωστά. Υπάρχουν επίσης ανησυχίες σχετικά με το απόρρητο και

την ασφάλεια [Chen et al., 2012]. Για να είναι προσβάσιμο το Cloud, πρέπει να έχει μια στατική και δημόσια διεύθυνση IP. Επομένως, οποιαδήποτε συσκευή με σύνδεση στο Διαδίκτυο μπορεί να φτάσει σε αυτήν.

Αξίζει να αναφερθεί ότι πλαίσια προγραμματισμού όπως το MapReduce [Dean J et al, 2004] απλοποίησαν πολύ τη διαδικασία του κατανεμημένου υπολογισμού και επέτρεψαν στους χρήστες να εφαρμόσουν τους αλγόριθμους σε μεγαλύτερα σύνολα δεδομένων. Αυτά τα πλαίσια παρέχουν ένα κατανεμημένο περιβάλλον χρόνου εκτέλεσης και ένα σύστημα αρχείων, επιτρέποντας στους χρήστες να επικεντρωθούν στην εφαρμογή των αλγορίθμων και όχι στη διαχείριση των λεπτομερειών σε χαμηλό επίπεδο.

3. Παράλληλοι αλγόριθμοι

Η εκτέλεση των αλγορίθμων μηχανικής μάθησης σε ένα κατανεμημένο περιβάλλον συνεπάγεται πρώτα την μετατροπή των αλγορίθμων σε παράλληλους αλγόριθμους. Αυτό το βήμα μπορεί συχνά να είναι το πιο δύσκολο και απαιτεί ο χρήστης να έχει κατανοήσει σε βάθος τον αλγόριθμο. Το δεύτερο βήμα περιλαμβάνει την εφαρμογή των παράλληλων αλγορίθμων.

Οι αλγόριθμοι μηχανικής μάθησης μπορούν να χωριστούν σε δύο κατηγορίες: εποπτευόμενοι και μη εποπτευόμενοι. Στην εποπτευόμενη μάθηση στα δεδομένα εισόδου δίνονται κάποιες ετικέτες (π.χ. ένα σύνολο μηνυμάτων ηλεκτρονικού ταχυδρομείου με ετικέτα ανεπιθύμητης αλληλογραφίας / όχι spam) και δημιουργείται ένα μοντέλο που μπορεί να χρησιμοποιηθεί για την πρόβλεψη μελλοντικών δεδομένων χωρίς ετικέτα. Η μη εποπτευόμενη μάθηση έχει ως στόχο την ανακάλυψη ιδιοτήτων σχετικά με τα δεδομένα (π.χ., ομαδοποίηση πελατών σε κατηγορίες για ανάλυση αγοράς).

4. Διαχείριση του φόρτου εργασίας

Για την αντιμετώπιση του διογκωμένου φόρτου εργασίας που προκαλείται από την ανάγκη επεξεργασίας του μεγάλου όγκου δεδομένων, πέρα από την ταυτόχρονη εκτέλεση των διαφόρων εργασιών, μπορούν να εφαρμοστούν δύο διαφορετικές προσεγγίσεις. Η μία λύση είναι η αύξηση

των υπολογιστικών πόρων που διαθέτει ο υπολογιστής που επεξεργάζεται τα δεδομένα και η άλλη είναι η αύξηση των υπολογιστών που εμπλέκονται στη διαδικασία.

1.4.1 Αύξηση των διαθέσιμων πόρων

Καθώς η μηχανική μάθηση βασίζεται σε πολλαπλασιασμούς πινάκων, για την βελτίωση της απόδοσης του συστήματος είναι απαραίτητη η αύξηση του αριθμού των πράξεων που μπορεί να εκτελέσει ο επεξεργαστής. Αυτό μπορεί να επιτευχθεί με την αύξηση των διαθέσιμων επεξεργαστών ή με την αντικατάσταση τους με πιο συγχρόνους, που διαθέτουν μεγαλύτερο αριθμό πυρήνων οι οποίοι έχουν τη δυνατότητα να έχουν πρόσβαση σε κοινά μπλοκ μνήμης ώστε να επεξεργάζονται τα ίδια δεδομένα, μειώνοντας σημαντικά με τον τρόπο αυτό, το κόστος επικοινωνίας μεταξύ τους. Επιπλέον και το διαθέσιμο σύνολο εντολών του επεξεργαστή μπορεί να αυξηθεί ώστε να πραγματοποιεί τις πράξεις αποδοτικότερα.

Πέρα από τους επεξεργαστές (CPU), οι επεξεργαστές γραφικών (GPU) μπορούν να είναι αρκετά πιο γρήγοροι σε εφαρμογές μηχανικής μάθησης. Πιο συγκεκριμένα οι επεξεργαστές της εταιρίας Nvidia Titan V και Tesla V100 διαθέτουν 5,120 πυρήνες που τους καθιστούν έως και 47 φορές γρηγορότερους από τον επεξεργαστή της Intel, τον Xeon E5. Πέρα από τον αριθμό των πυρήνων όμως σημαντικές αποδεικνύονται και οι βελτιώσεις που απολαμβάνουν από πλευράς αρχιτεκτονικής. Πιο συγκεκριμένα, αρχικά οι πυρήνες των επεξεργαστών αυτών ήταν περιορισμένοι ώστε τη κάθε χρονική στιγμή να εκτελούν το ίδιο κώδικα μεταξύ τους πράγμα που περιορίζει σημαντικά τις δυνατότητες εκτέλεσης παράλληλων διεργασιών. Καθώς όμως η τεχνολογία βελτιωνόταν, οι επεξεργαστές αυτοί απέκτησαν περισσότερες δυνατότητες, μεταξύ αυτών και να μπορούν οι πυρήνες τους να εκτελούν διαφορετικό κώδικα.

Σημαντικός είναι και ο ρόλος των ολοκληρωμένων κυκλωμάτων ειδικού σκοπού (ASICs) τα οποία υλοποιούν συγκεκριμένη λειτουργικότητα μέσω του ειδικού σχεδιασμού τους. Λόγο ακριβώς του ειδικού σχεδιασμού τους και ποιο συγκεκριμένα ότι είναι κατασκευασμένα να εκτελούν συγκεκριμένες λειτουργίες αποδεικνύονται ιδιαίτερα αποδοτικά σε σχέση με τα

κυκλώματα γενικού σκοπού, τόσο σε θέμα ταχύτητας όσο και κατανάλωσης ηλεκτρικής ενέργειας. Το γεγονός ότι οι αλγόριθμοι μηχανικής μάθησης έχουν περιορισμένες λειτουργίες και βασίζονται σε μεγάλο βαθμό σε πολλαπλασιασμούς πινάκων επιτρέπει το σχεδιασμό ενός κυκλώματος που θα έχει περιορισμένες δυνατότητες, αφού η βασική του εργασία είναι ο πολλαπλασιασμός πινάκων. Συνεπώς θα διαθέτει όλους του τους πόρους σε αυτή και μόνο τη διεργασία καθιστώντας το με τον τρόπο αυτό πιο αποδοτικό. Παράδειγμα ενός τέτοιου ολοκληρωμένου κυκλώματος αποτελεί το Tensor Processing Unit [Kaz et all, 2017] της εταιρίας Google το οποίο εκτελεί την βιβλιοθήκη της ίδιας εταιρίας Tensorflow [Martin et all, 2015]. Το συγκεκριμένο ολοκληρωμένο είναι αρχιτεκτονικής MIMD (Multiple Instructions, Multiple Data) που του επιτρέπει να εκτελούν οι πυρήνες του διαφορετικό κώδικα. Επιπλέον επικοινωνεί με τον εξυπηρετητή μέσω PCI Express πράγμα που του επιτρέπει ρυθμό μετάδοσης δεδομένων της τάξης των 63 GB/s. Τα χαρακτηριστικά αυτά του επιτρέπουν να είναι έως και 200 φορές γρηγορότερο από τα παραδοσιακά συστήματα.

1.4.2 Αύξηση των διαθέσιμων κόμβων

Στην αντίπερα όχθη, η αύξηση του αριθμού των κόμβων που επεξεργάζονται τα δεδομένα θα οδηγήσει σε μια αισθητή αύξηση της απόδοσης του συστήματος. Ο συνδυασμός όμως των δύο διαφορετικών πρακτικών τείνει να κερδίζει έδαφος καθώς παρουσιάζει περισσότερα πλεονεκτήματα όπως χαμηλότερο κόστος τόσο από πλευράς αρχικής επένδυσης αλλά όσο και εγκατάστασης. Παράλληλα με το κόστος η προσέγγιση πλεονεκτεί και σε θέματα αξιοπιστίας και διαθεσιμότητας καθώς η διαδικασία θα συνεχιστεί ακόμα και αν κάποιος κόμβος τεθεί εκτός λειτουργίας, αφού οι υπόλοιποι θα συνεχίσουν αδιάκοπα την επεξεργασία.

5. Αρχιτεκτονική κατανεμημένου περιβάλλοντος

Η διαδικασία της μηχανικής μάθησης ολοκληρώνεται σε δύο φάσεις, όπου η πρώτη περιλαμβάνει τις διαδικασίες δημιουργίας του μοντέλου και η δεύτερη την χρήση του μοντέλου αυτού για την παραγωγή των αποτελεσμάτων.

Για τη δημιουργία του μοντέλου εφαρμόζονται οι αλγόριθμοι μηχανικής μάθησης που παρουσιάζονται στην επόμενη παράγραφο πάνω σε μεγάλου όγκου δεδομένα. Όσο μεγαλύτερος ο όγκος των δεδομένων που θα χρησιμοποιηθούν για την εκπαίδευση του μοντέλου, τόσο ακριβέστερο θα είναι το μοντέλο που θα παραχθεί. Παράλληλα, ο κάθε αλγόριθμος χαρακτηρίζεται από μια ομάδα παραμέτρων που θα πρέπει να ορισθούν κατάλληλα πράγμα που επηρεάζει σημαντικά την απόδοση του μοντέλου.

Όταν ολοκληρωθεί η φάση της δημιουργίας του μοντέλου αυτό πλέον είναι έτοιμο για χρήση. Το μοντέλο θα πρέπει να τροφοδοτηθεί με νέα δεδομένα και θα προβεί σε αποφάσεις με βάση τη γνώση που έχει αποκτήσει στη φάση της εκπαίδευσής του.

Σε ένα κατανεμημένο περιβάλλον εγείρονται διάφορα ζητήματα μεταξύ αυτών και η θέση που θα πρέπει να τοποθετηθούν τα δεδομένα ή το μοντέλο ώστε να μπορέσει να λειτουργήσει η υποδομή. Πιο συγκεκριμένα τα δεδομένα, επειδή καταλαμβάνουν μεγάλο όγκο και παράλληλα θα πρέπει να είναι διαθέσιμα στο σύνολο της υποδομής, προκειμένου να δημιουργηθεί το μοντέλο, θα πρέπει να ληφθεί ιδιαίτερη μέριμνα για τη διάθεση τους καθώς το κόστος διατήρησης τους στους κόμβους ή διάθεσης αυτών μέσω διαδικτύου, καθίσταται αξιοσημείωτο.

Μία από τις προτάσεις αρχιτεκτονικής [Joost et all, 2020] όσο αφορά την κατανομή των δεδομένων, διακρίνει δύο διαφορετικές αρχιτεκτονικές όπου στη πρώτη τα δεδομένα μοιράζονται στους κόμβους ενώ στη δεύτερη όλοι οι κόμβοι αποκτούν το σύνολο των δεδομένων.

Πιο συγκεκριμένα, στη πρώτη φάση τα δεδομένα διαχωρίζονται και μοιράζονται στους κόμβους με αποτέλεσμα κάθε ένας από αυτούς να διαθέτει ένα διαφορετικό υποσύνολο των αρχικών

δεδομένων. Κατά συνέπεια, κάθε κόμβος δουλεύει σε διαφορετικά δεδομένα και δημιουργεί το δικό του ξεχωριστό μοντέλο το οποίο συγκεντρώνεται σε ένα κεντρικό σημείο ή μεταδίδεται στους υπόλοιπους κόμβους ώστε να καταλήγουν όλοι να δουλεύουν στο ίδιο μοντέλο. Η προσέγγιση αυτή κρίνεται ιδιαίτερα απαραίτητη σε περιπτώσεις όπου το σύνολο των δεδομένων δεν είναι δυνατόν να διατηρούνται στους επιμέρους κόμβους.

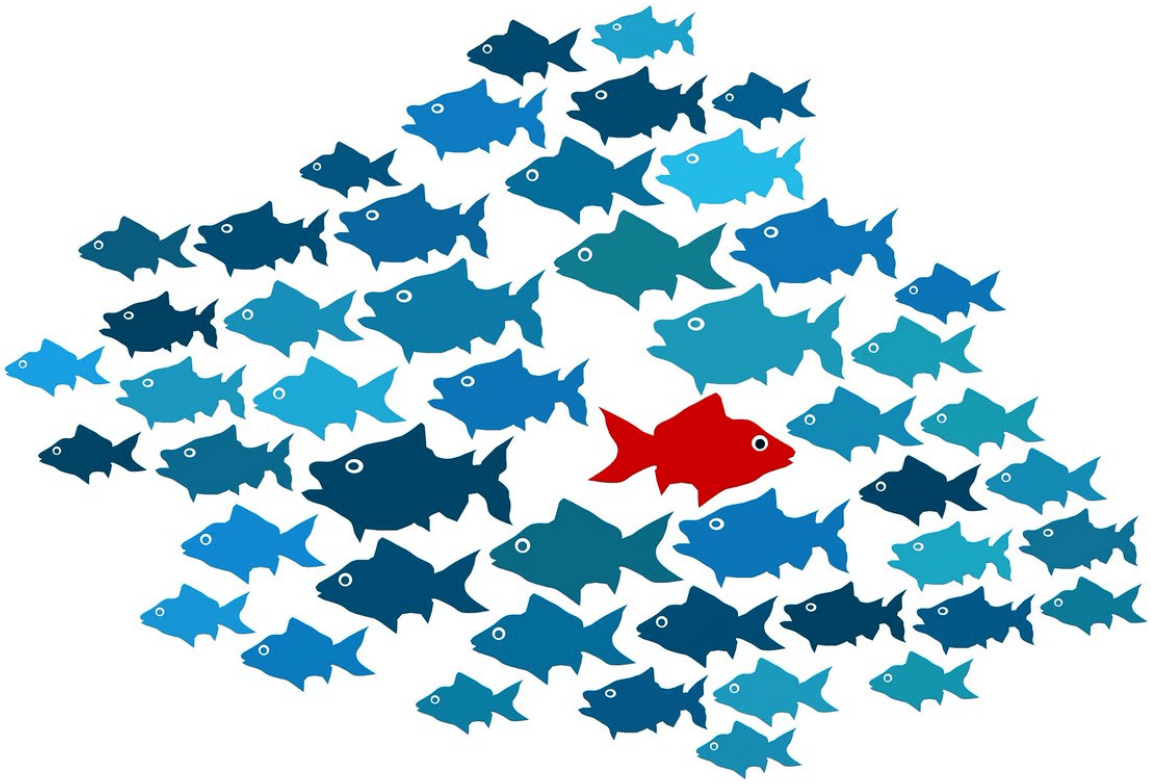
Στη περίπτωση που οι επιμέρους κόμβοι μπορούν να φιλοξενήσουν το σύνολο των δεδομένων δύναται να εφαρμοστεί η δεύτερη από τις προτεινόμενες αρχιτεκτονικές σύμφωνα με την οποία οι κόμβοι, έχοντας πλήρη πρόσβαση στα δεδομένα δουλεύουν σε διαφορετικά τμήματα του ίδιου μοντέλου και στη συνέχεια λειτουργούν συλλογικά για την παραγωγή του ενιαίου μοντέλου. Η αρχιτεκτονική αυτή είναι δυσκολότερο να εφαρμοστεί καθώς τα χαρακτηριστικά του μοντέλου δεν είναι εύκολο να διαμοιραστούν στους κόμβους. Υπάρχουν όμως περιπτώσεις που η αρχιτεκτονική είναι μονόδρομος μιας και για την παραγωγή αξιόπιστων αποτελεσμάτων απαιτούνται περισσότερα του ενός μοντέλα. Πιο συγκεκριμένα στην αρχιτεκτονική αυτή μπορούν να δημιουργηθούν περισσότερα του ενός μοντέλα, ένα από κάθε διαφορετικό κόμβο και ακολούθως να συνδυαστούν με διαφόρους τρόπους για την παραγωγή του ενιαίου μοντέλου. Μεταξύ των μεθόδων αυτών συνδυασμού των μοντέλων αξίζει να αναφερθεί η διαδικασία δημιουργίας των διαφορετικών μοντέλων και ακολούθως η επιλογή αυτού που παράγει τα καλύτερα αποτελέσματα. Άλλη μέθοδος συνίσταται στη δημιουργία νέων μοντέλων που έχουν σαν στόχο να κατηγοριοποιήσουν τα δεδομένα που τα ήδη υπάρχοντα μοντέλα απέτυχαν να κατηγοριοποιήσουν.

6. Οι αλγόριθμοι μηχανικής μάθησης ML

Οι αλγόριθμοι μηχανικής μάθησης μπορούν να χρησιμοποιηθούν σε ένα πλήθος εργασιών [Donghwoon Kwon et al. , 2017]:

• Η ανίχνευση ανωμαλιών (Anomaly detection)

Ως ανωμαλίες μπορούν να οριστούν τα μοτίβα που δεν συνάδουν με την κανονική συμπεριφορά. Η ανίχνευση ανωμαλιών αναφέρεται στο πρόβλημα εύρεσης τέτοιων μοτίβων σε δεδομένα [Chandola et al., 2007]. Η ανίχνευση της ανώμαλης συμπεριφοράς μπορεί να θεωρηθεί ως ένα δυαδικό πρόβλημα ταξινόμησης, όπου ένας ταξινομητής αποφασίζει εάν ένα σημείο είναι φυσιολογικό ή μη φυσιολογικό [Lane and Brodly, 1997]. Η καταλληλότερη τεχνική ανίχνευσης, εξαρτάται από την περιοχή της έρευνας, τα χαρακτηριστικά του προβλήματος και του τομέα εφαρμογής [Chandola et al., 2007].

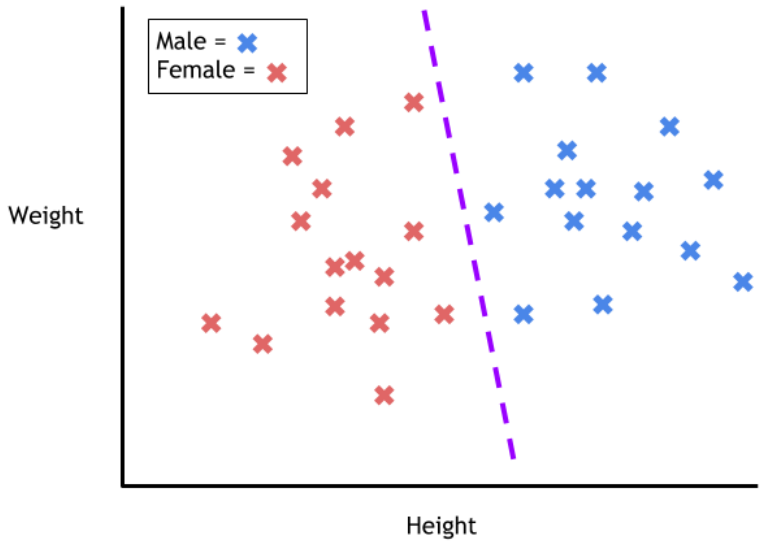


Εικόνα 1: Anomaly Detection. Πηγή: https://www.goalprofit.com/resources/tpost/1ptttuulf1-anomaly-detection

• Η ταξινόμηση (Classification)

Η ταξινόμηση είναι μια προσέγγιση μηχανικής μάθησης που χρησιμοποιείται για την πρόβλεψη της ομάδας στην οποία μπορεί ανήκει το κάθε δεδομένο [Kesavaraj G et al., 2013]. Η ταξινόμηση χαρακτηρίζεται ως ένα από τα μεγαλύτερα προβλήματα που μελετήθηκαν από τους ερευνητές των πεδίων της μηχανικής μάθησης και της εξόρυξης δεδομένων [Baradwaj BK,

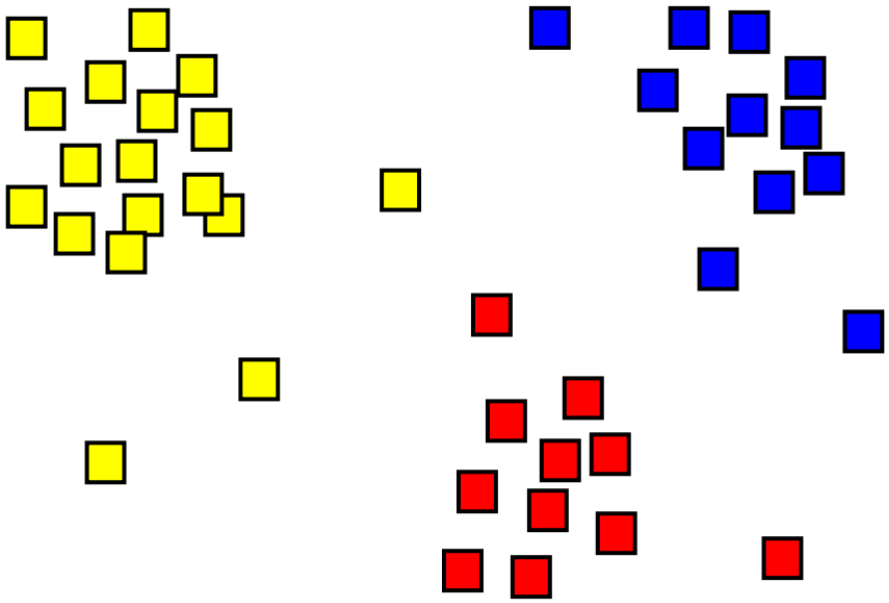
2012]. Αν και η ταξινόμηση είναι γνωστή τεχνική στη μηχανική μάθηση, υπολείπεται στην περίπτωση που λείπουν κάποια δεδομένα. Τότε μπορεί να προκληθεί πρόβλημα τόσο κατά τη διάρκεια της φάσης της εκπαίδευσης όσο και κατά την ταξινόμηση.



Εικόνα 2: Classification. Πηγή: https://towardsdatascience.com/a-brief-introduction-to-supervised-learning-54a3e3932590

• Clustering

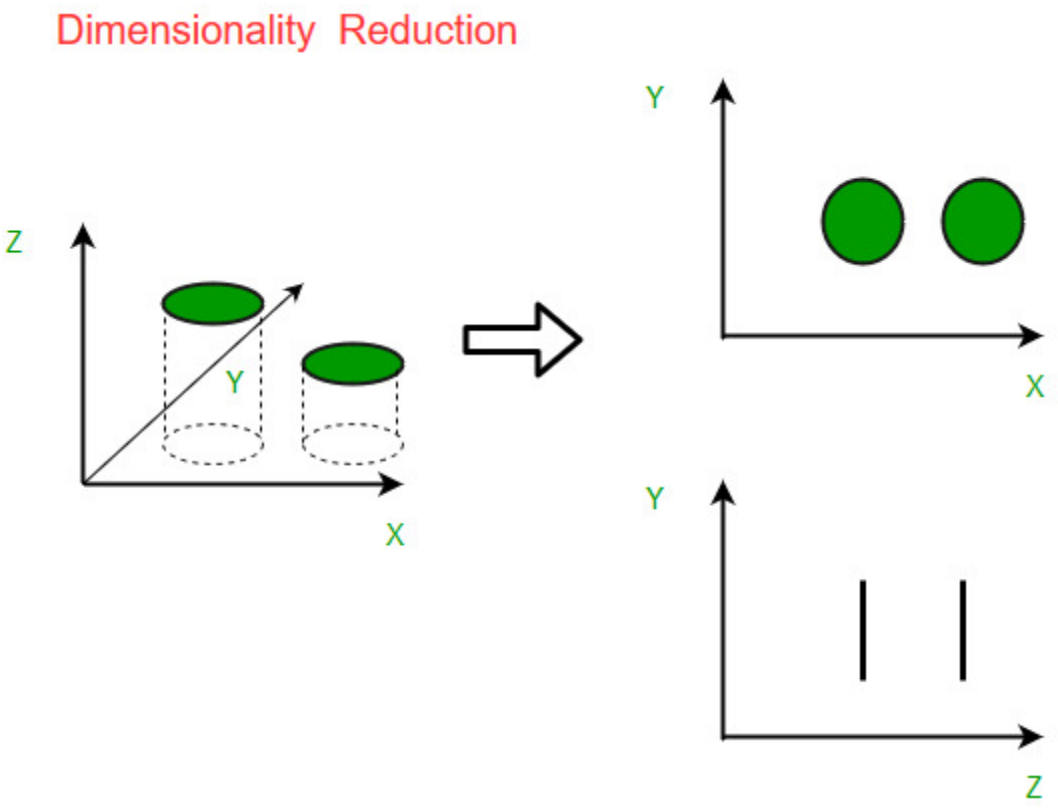
Η ομαδοποίηση των δεδομένων είναι η διαδικασία ταυτοποίησης ομάδων με βάση κάποιο μέτρο ομοιότητας (π.χ. Ευκλείδια απόσταση) [Jain et al. 1999, Jain et al. 2000]. Είναι μια σημαντική διαδικασία αναγνώρισης προτύπων και μηχανικής μάθησης [Hamerly and Elkan 2002]. Οι αλγόριθμοι clustering χρησιμοποιούνται σε πολλές εφαρμογές, όπως η τμηματοποίηση μιας εικόνας [Jain και Dubes 1988], η εξόρυξη δεδομένων [Judd et al. 1998], η συμπίεση [Abbas and Fahmy 1994] κλπ. Η ομαδοποίηση των δεδομένων είναι ένα δύσκολο πρόβλημα στην αναγνώριση προτύπων χωρίς επίβλεψη, καθώς οι συστάδες των δεδομένων ενδέχεται να έχουν διαφορετικά σχήματα και μεγέθη [Jain et al. 2000].



Εικόνα 3: Clustering. Πηγή: https://en.wikipedia.org/wiki/Cluster_analysis

• Μείωση διαστάσεων:

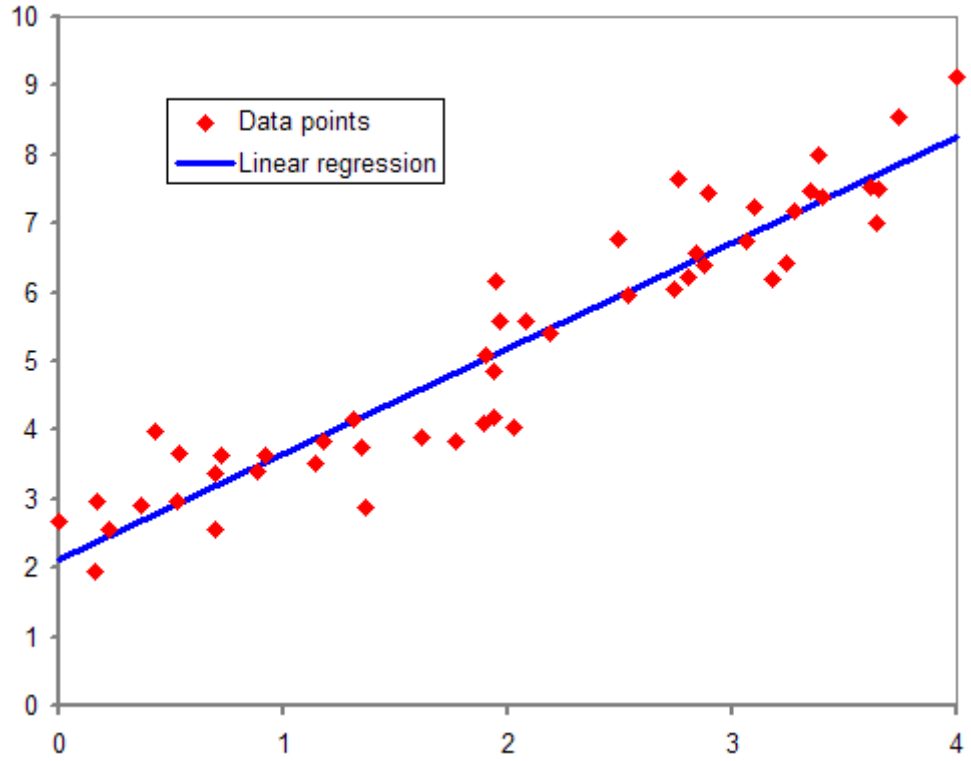
Όταν τα δεδομένα περιγράφονται από μεγάλο αριθμό χαρακτηριστικών δηλαδή όταν τα δεδομένα έχουν υψηλή διάσταση, είναι συχνά ωφέλιμο να μειωθεί η διάσταση των δεδομένων. Αυτό μπορεί να επιτευχθεί επιλέγοντας μόνο τις σχετικές μεταβλητές (επιλογή χαρακτηριστικών) ή δημιουργώντας νέες μεταβλητές που αντιπροσωπεύουν πολλές άλλες (εξαγωγή χαρακτηριστικών). Η μείωση των διαστάσεων μπορεί να είναι επωφελής, όχι μόνο για λόγους υπολογιστικής αποτελεσματικότητας αλλά και επειδή μπορεί να βελτιώσει την ακρίβεια της ανάλυσης.



Εικόνα 4: Dimensionality Reduction. Πηγή: https://www.geeksforgeeks.org/dimensionality-reduction/

• Η παλινδρόμηση

Είναι το πρόβλημα της εκτίμησης του τρόπου με τον οποίο μια λεγόμενη εξαρτημένη μεταβλητή αλλάζει αξία όταν άλλες μεταβλητές αλλάζουν με ένα συγκεκριμένο ποσό. Η ανάλυση παλινδρόμησης είναι μια στατιστική μέθοδος που μας βοηθά να αναλύσουμε και να κατανοήσουμε τη σχέση μεταξύ δύο ή περισσότερων μεταβλητών. Η διαδικασία της παλινδρόμησης βοηθά να κατανοηθεί ποιοι παράγοντες είναι σημαντικοί, ποιοι παράγοντες μπορούν να αγνοηθούν και πώς επηρεάζουν ο ένας τον άλλο.



Εικόνα 5: Regression. Πηγή: https://en.wikipedia.org/wiki/Regression_analysis

Η διαδικασία μάθησης σε ένα απλό μοντέλο μηχανικής μάθησης χωρίζεται σε δύο στάδια: Τη διαδικασία της εκπαίδευσης και τη διαδικασία της δοκιμής. Κατά τη διαδικασία της εκπαίδευσης, ένα μέρος των δεδομένων δίνεται ως είσοδος και σύμφωνα με αυτά ο αλγόριθμος μαθαίνει και δημιουργεί το μοντέλο. Στη διαδικασία της δοκιμής, το μοντέλο μάθησης χρησιμοποιεί τη μηχανή εκτέλεσης για να κάνει την πρόβλεψη στα υπόλοιπα δεδομένα.

1.6.1 Supervised learning

Στην εποπτευόμενη μάθηση, ο εκπαιδευόμενος που συνήθως είναι ένα πρόγραμμα υπολογιστή, διαθέτει δύο σύνολα δεδομένων, ένα σύνολο που προορίζεται για την εκπαίδευση και ένα σύνολο που προορίζεται για τη δοκιμή του μοντέλου. Η ιδέα βασίζεται στη λογική ότι ο μαθητής «μαθαίνει» μέσα από ένα σύνολο παραδειγμάτων που βρίσκονται στο σετ των δεδομένων εκπαίδευσης έτσι ώστε να μπορεί να προβλέψει αντίστοιχα παραδείγματα από το σύνολο

δοκιμών με την υψηλότερη δυνατή ακρίβεια. Δηλαδή, ο αλγόριθμος ταξινομεί νέα παραδείγματα (στο σύνολο δοκιμών) αναλύοντας παραδείγματα που έχουν δοθεί και έχουν ήδη μια ετικέτα. Για παράδειγμα, ένα εκπαιδευτικό σύνολο δεδομένων μπορεί να αποτελείται από εικόνες διαφορετικών τύπων φρούτων (ροδάκινα και μήλα), όπου η ταυτότητα του φρούτου σε κάθε εικόνα δίνεται στον αλγόριθμο ως είσοδος. Τα δεδομένα δοκιμής αποτελούνται από άγνωστα κομμάτια αυτών των φρούτων και ο αλγόριθμος καλείται να κατατάξει τα φρούτα στη σωστή κατηγορία.

1.6.2 Unsupervised Learning

Η μη εποπτευόμενη μάθηση αναφέρεται στην εκπαίδευση μηχανών που χρησιμοποιούν πληροφορίες που δεν είναι ταξινομημένες. Εδώ το καθήκον της μηχανής είναι να ομαδοποιήσει τις μη ταξινομημένες πληροφορίες σύμφωνα με ομοιότητες, μοτίβα και διαφορές χωρίς προηγούμενη εκπαίδευση. Σε αντίθεση με την εποπτευόμενη μάθηση, δεν παρέχεται κανένας δάσκαλος στον υπολογιστή. Επομένως, η μηχανή περιορίζεται να βρει την κρυφή δομή σε δεδομένα χωρίς να έχουν δοθεί ετικέτες από πριν. Για παράδειγμα, έστω ότι δίνεται ένα σύνολο εικόνων με σκύλους και γάτες που δεν τις έχει δει ποτέ η μηχανή. Έτσι, ο υπολογιστής δεν έχει ιδέα για τα χαρακτηριστικά των σκύλων και των γατών, έτσι δεν μπορεί να το κατηγοριοποιήσει σε σκύλους και γάτες. Όμως μπορεί να τις κατηγοριοποιήσει σύμφωνα με τις ομοιότητες, τα μοτίβα και τις διαφορές τους, δηλαδή, μπορεί εύκολα να κατηγοριοποιήσει τις εικόνες σε δύο ομάδες. Η πρώτη μπορεί να περιέχει όλες τις φωτογραφίες που έχουν σκύλους και η δεύτερη μπορεί να περιέχει όλες τις φωτογραφίες που έχουν γάτες. Η μηχανή δεν έμαθε τίποτα από το παρελθόν, δεν υπάρχουν δεδομένα εκπαίδευσης ή παραδείγματα. Το μοντέλο λειτουργεί από μόνο του ώστε να ανακαλύψει μοτίβα και πληροφορίες σε μη επισημασμένα δεδομένα.

1.6.3 Reinforcement learning

Η ενισχυμένη μάθηση είναι ένας τομέας της μηχανικής μάθησης. Πρόκειται για τη λήψη κατάλληλων ενεργειών ώστε να επιτευχθεί η μεγιστοποίηση της ανταμοιβής σε μια

συγκεκριμένη κατάσταση. Χρησιμοποιείται από διάφορα λογισμικά και μηχανήματα για να βρει την καλύτερη δυνατή συμπεριφορά ή διαδρομή που πρέπει να ακολουθηθεί σε μια συγκεκριμένη κατάσταση. Διαφέρει από την εποπτευόμενη μάθηση, διότι στην εποπτευόμενη μάθηση τα δεδομένα εκπαίδευσης έχουν μαζί τους και την ετικέτα, οπότε το μοντέλο εκπαιδεύεται με την ίδια τη σωστή απάντηση, ενώ στην ενισχυμένη μάθηση, δεν υπάρχει καμία απάντηση, αλλά ο υπολογιστής αποφασίζει τι να κάνει. Ελλείψει εκπαιδευτικού συνόλου δεδομένων, η μηχανή είναι υποχρεωμένη να μάθει από την εμπειρία της.

1.7 Περιβάλλον Κατανεμημένης Μηχανικής Μάθησης

Η επεξεργασία μεγάλου όγκου δεδομένων που αποτελεί ένα βασικό τμήμα της μηχανικής μάθησης, είναι ένας τομέας που έχει μελετηθεί αρκετά στη διεθνή κοινότητα με αποτέλεσμα να έχουν προκύψει διάφορες προσεγγίσεις. Οι προσεγγίσεις αυτές μπορούν να κατηγοριοποιηθούν σε αυτές που αφορούν την υλοποίηση συστημάτων τόσο γενικού όσο και ειδικού σκοπού που ενώ αρχικά είχαν σχεδιαστεί ώστε να λειτουργούν σε μεμονωμένα μηχανήματα εξελίχθησαν ώστε να μπορούν να λειτουργήσουν και σε ένα κατανεμημένο περιβάλλον. Στην αντίπερα όχθη εντοπίζονται αυτές που έξ αρχής είχαν σχεδιαστεί ώστε να λειτουργούν σε ένα κατανεμημένο περιβάλλον και να υλοποιούν συγκεκριμένους αλγόριθμους.

1.7.1 Κατανεμημένα συστήματα Γενικού Σκοπού

Η επεξεργασία των δεδομένων στα κατανεμημένα συστήματα βασίζεται σε υπολογιστές χαμηλού κόστους και όχι σε ακριβούς εξυπηρετητές καθώς η εμπειρία έχει δείξει ότι η μέθοδος αυτή είναι αποδοτικότερη. Την προσέγγιση αυτή την εφάρμοσε αρχικά η Google [Luiz et all, 2003] και ακολούθως υιοθετήθηκε και από τις υπόλοιπες εταιρίες του χώρου. Πέρα από τα οφέλη στον οικονομικό τομέα η προσέγγιση κερδίσει έδαφός και λόγο των επεξεργαστικών δυνατοτήτων που διαθέτει μιας και μια αύξηση του αριθμού των μηχανημάτων επιφέρει μεγαλύτερη αύξηση της συνολικής επεξεργαστικής ικανότητας σε σχέση με τη χρήση ακριβών εξυπηρετητών. Το γεγονός αυτό οφείλεται στο ότι λόγο του χαμηλού κόστους των μηχανημάτων

μπορούν να προστεθούν περισσότερα, πράγμα που οδηγεί στη βελτίωση της απόδοσης της υποδομής σε σχέση με την προσθήκη μικρότερου αριθμού αλλά ισχυρότερων μηχανημάτων.

Για τη διαχείριση των αρχείων, μεταξύ των διαθέσιμων βιβλιοθηκών ξεχωρίζει και πάλι η λύση που έχει εφαρμόσει η Google και συνίσταται στον διαχωρισμό των δεδομένων σε ομάδες και κατόπιν διαμοιρασμό των ομάδων αυτών σε ξεχωριστούς υπολογιστές. Με τον τρόπο αυτό επιτυγχάνεται και η διασφάλιση των δεδομένων σε περίπτωση αστοχίας υλικού αφού τα δεδομένα διατηρούνται σε περισσότερες τις μιας τοποθεσίες τη κάθε στιγμή. Οι υπολογιστές αυτοί οργανώνονται από ένα κεντρικό ο οποίος αναλαμβάνει να εξυπηρετήσει τις αιτήσεις των αρχείων αφού γνωρίζει σε ποιον υπολογιστή της ομάδας του βρίσκεται. Μία από τις διασημότερες βιβλιοθήκες, το Hadoop [Jyoti et all, 2013], το οποίο αρχικά αναπτύχθηκε από τη Yahoo!, βασίζει τη λειτουργία του στο σύστημα αυτό.

Πέρα από τη διαχείριση αρχείων σε ένα κατανεμημένο περιβάλλον, αρκετές βιβλιοθήκες έχουν δημιουργηθεί για τον προγραμματισμό και τη διανομή των έργων επεξεργασίας. Κατά την επεξεργασία των μεγάλων δεδομένων πέρα από τα δεδομένα, στους επιμέρους κόμβους ανατίθενται έργα επεξεργασίας τα οποία αποτελούνται από διεργασίες οι οποίες θα πρέπει να διεκπαιραιωθούν από τους κόμβους που τις έχουν αναλάβει. Οι βιβλιοθήκες αυτές αναλαμβάνουν να μοιράσουν τις διεργασίες και να οργανώσουν τους κόμβους κατά την διεκπαιρέωση τους. Μεταξύ των βιβλιοθηκών αυτών ξεχωρίζει η MapReduce, η οποία δεν αποτελεί απλά μια βιβλιοθήκη αλλά και μία αρχιτεκτονική και αναπτύχθηκε από τη Google [Jeffrey et all, 2004] .

Για την επεξεργασία των δεδομένων η βιβλιοθήκη ακολουθεί τρείς φάσεις όπου στη πρώτη διασπά τα δεδομένα σε ομάδες. Η διαδικασία αυτή γίνεται παράλληλα από πολλούς κόμβους καθώς τα δεδομένα δεν περιορίζονται από εξαρτήσεις. Στη δεύτερη φάση τα δεδομένα ανταλλάσσονται μεταξύ των κόμβων ώστε να εξασφαλιστεί ότι το κάθε δεδομένο θα εμπεριέχεται στη σωστή ομάδα και στο σωστό κόμβο. Τέλος στη τρίτη φάση κάθε ομάδα αποκτά το δικό της ξεχωριστό κλειδί χαρακτηρισμού των δεδομένων. Συνεπώς, η δυνατότητα ταυτόχρονης εκτέλεσης διεργασιών αλλά και το γεγονός ότι οι διεργασίες μπορούν να

εκτελεστούν από τους κόμβους που διαθέτουν τα απαραίτητα δεδομένα καθιστά τη μέθοδό ιδιαίτερα αποδοτική τόσο σε επεξεργαστικές δυνατότητες όσο και στη διαχείριση του εύρους ζώνης, αφού τα δεδομένα που καταλήγουν να διακινούνται είναι το ίδιο το πρόγραμμα και όχι τα δεδομένα του.

Πέρα από το MapReduce, μεταξύ των βιβλιοθηκών για κατανεμημένη επεξεργασία ξεχωρίζει και το Apache Spark, το οποίο σε αντίθεση με το MapReduce δεν χρησιμοποιεί το σύστημα αρχείων του Handoop το οποίο διανέμει το σύνολο των δεδομένων σε κόμβους αλλά τα διατηρεί στη μνήμη του συστήματος. Η προσέγγιση αυτή κρίνεται απαραίτητη σε περιπτώσεις όπου τα δεδομένα θα πρέπει να επεξεργάζονται επαναλαμβανόμενα, πράγμα που καθιστά τη τοποθεσία τους κρίσιμο παράγοντα για την απόδοση του συστήματος. Ειδικότερα σε περιπτώσεις μηχανικής μάθησης, η οποία βασίζεται σε πολλαπλασιασμούς πινάκων, η διατήρηση των δεδομένων στη μνήμη του συστήματος κρίνεται αναγκαία αφού τα δεδομένα θα πρέπει να είναι άμεσα και διαρκώς διαθέσιμα και η διατήρηση τους σε απομακρυσμένες και διαφορετικές τοποθεσίες θα έκανε την ανάκτηση του χρονοβόρα αλλά και κοστοβόρα. Παράλληλα η βιβλιοθήκη περιλαμβάνει επιπλέον βιβλιοθήκες μηχανικής μάθησης για την διεκπεραίωση αρκετών διαδικασιών όπως ομαδοποίησης ή ταξινόμησης, παραμετροποίησης παραμέτρων συστημάτων και άλλες.

1.7.2 Κατανεμημένα συστήματα Ειδικού Σκοπού

Πέρα από τα συστήματα γενικού σκοπού, τα τελευταία χρόνια έχουν αναπτυχθεί βιβλιοθήκες σχεδιασμένες για συγκεκριμένες εργασίες. Στη παράγραφο αυτή θα γίνει μια προσπάθεια να παρουσιαστούν οι πλέον καταξιωμένες εφαρμογές στο χώρο ώστε να σχηματίσει ο αναγνώστης μια όσο το δυνατόν πληρέστερη εικόνα του τοπίου που έχει δημιουργηθεί.

Στη διεθνή βιβλιογραφία θα εντοπίσει κανείς αρκετές προτάσεις που βασίζονται στα ήδη υπάρχοντα συστήματα γενικού σκοπού τα οποία εξελίσσονται ώστε να μπορούν να λειτουργήσουν και σαν ειδικού σκοπού. Ποιο συγκεκριμένα, συστήματα τα οποία έχει

αποδειχθεί ότι αποδίδουν ικανοποιητικά σε ένα μη κατανεμημένο περιβάλλον, μπορούν να τροποποιηθούν, ώστε να μπορούν να λειτουργήσουν και σε ένα κατανεμημένο αυξάνοντας με τον τρόπο αυτό την αποδοτικότητα τους. Μια μέθοδος για να επιτευχθεί αυτό είναι να διεκπεραιωθούν τα διαφορετικά στάδια της μάθησης από διαφορετικούς κόμβους. Ποιο συγκεκριμένα η διαδικασία της εκπαίδευσης θα μπορούσε να πραγματοποιηθεί από διαφορετικούς κόμβους που διαθέτουν τμήματα των δεδομένων και κάθε κόμβος να παράξει το δικό του μοντέλο. Στη συνέχεια κάθε κόμβος να εφαρμόσει την γνώση που απόκτησε στα δεδομένα που χρήζουν επεξεργασίας και ακολούθως οι διαφορετικές προτάσεις να συνδυαστούν χρησιμοποιώντας τις ήδη γνωστές μεθόδους συνδυασμού των αποτελεσμάτων αυτών. Καθίσταται προφανές στον αναγνώστη ότι η διαδικασία αυτή μπορεί να βασιστεί στις ήδη υπάρχουσες και δοκιμασμένες βιβλιοθήκες όπως η MapReduce. Στα μειονεκτήματα της μεθόδου θα εντοπίσει κανείς το γεγονός ότι ο διαμερισμός των δεδομένων στους κόμβους διαδραματίζει σημαντικό ρόλο στο αποτέλεσμα και για το λόγο αυτό θα πρέπει να ληφθεί ειδική μέριμνα ώστε όλοι οι κόμβοι να έχουν ένα αρκετά αντιπροσωπευτικό δείγμα του συνόλου ώστε να μπορούν να παράγουν ένα όσο το δυνατόν ακριβέστερο μοντέλο. Παραδείγματα τέτοιων βιβλιοθηκών είναι η Tensorflow [Martin et all, 2015], η MXNet [Tianqi et all, 2015] και η PyTorch [Adam Paszke et all, 2017].

Άλλες προσεγγίσεις μπορούν να ομαδοποιηθούν σε διάφορες κατηγορίες από τις οποίες ξεχωρίζουν αυτές που διακρίνουν την επικοινωνία μεταξύ των συστημάτων σε σύγχρονη ή ασύγχρονη. Στη πρώτη κατηγορία ανήκουν βιβλιοθήκες που χρησιμοποιούν γνωστές βιβλιοθήκες επικοινωνίας, όπως η MPI [William et all, 1999] για την επικοινωνία των συστημάτων που οργανώνονται σε μία δενδροειδής μορφή. Ποιο συγκεκριμένα ο κάθε κόμβος υπολογίζει τα επι μέρους βάρη του και κατόπιν ενημερώνει το γονικό του κόμβο. Τα βάρη αυτά μεταφέρονται σταδιακά στο κόμβο ρίζα ο οποίος ακολούθως ενημερώνει τους κόμβους φύλλα για το άθροισμα των βαρών. Βιβλιοθήκες αυτής της ομάδας είναι οι Baidu [Andrew et all, 2017], η Horovod [Alexander et all, 2018] και άλλες. Από την άλλη πλευρά, η ομάδα των ασύγχρονων περιλαμβάνει βιβλιοθήκες που είναι πιο δύσκολο να υλοποιηθούν καθώς και να εντοπισθούν πιθανές αστοχίες ακριβώς λόγο της φύσης της ασύγχρονης επικοινωνίας. Οι βιβλιοθήκες αυτές

όμως χαρακτηρίζονται από την υψηλή απόδοση που προσφέρει η ασύγχρονη επικοινωνία. Στην κατηγορία αυτή συμπεριλαμβάνονται βιβλιοθήκες όπως η DistBelief [Jeffrey et all, 2012] και η μεταγενέστερη εξέλιξη της Tensorflow, η DIANNE [Elias et all, 2018] καθώς και άλλες.

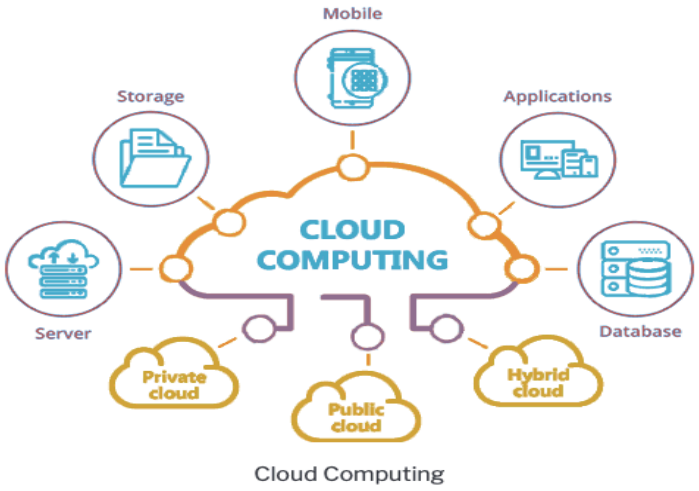
Κεφάλαιο 2: Υπολογιστική Παρυφών και Εφαρμογές

2.1. Υπολογιστική Παρυφών

Το cloud computing αποτελούσε μια τυπική πλατφόρμα τεχνολογίας πληροφοριών (IT) για περισσότερο από 10 χρόνια. Είναι μια υπηρεσία που επιτρέπει στους χρήστες που έχουν πρόσβαση στο Διαδίκτυο να έχουν πρόσβαση σε υπολογιστικούς πόρους και σε αποθηκευτικό χώρο [Dikaiakos et al, 2009]. Το cloud computing μπορεί να χρησιμοποιηθεί τόσο για την αποτελεσματική διαχείριση των πόρων που είναι αποθηκευμένοι σε έναν κεντρικό διακομιστή cloud όσο και για την αξιοποίηση αυτών των πόρων χωρίς περιορισμούς χρόνου και χώρου. Το σύστημα αρχείων της Google [Ghemawat et al., 2003], το MapReduce [Dean et al., 2008], το Apache Hadoop [Shvachko et al, 2010] και το Apache Spark [Zaharia et al., 2010] υποστηρίζουν υπηρεσίες cloud. Ωστόσο, καθώς η τεχνολογία αναπτύσσεται, αποκαλύπτονται τα μειονεκτήματα του cloud computing.

Με την πρόσφατη αύξηση της χρήσης του Internet of Things, ο αριθμός των συσκευών που συνδέονται στο Διαδίκτυο αυξάνεται καθημερινά. Το 1992, ο αριθμός των συνδεδεμένων συσκευών έφτανε το 1 εκατομμύριο και το 2003, οι χρήστες φορητού υπολογιστή αυξήθηκαν σε περισσότερους από 500 εκατομμύρια. Το 2012 λόγω των φορητών συσκευών ο αριθμός αυτός ανήλθε σε 8,7 δισεκατομμύρια. Ακολούθως το 2013 αυτός ο αριθμός έγινε 11,2 δισεκατομμύρια λόγω συνδεδεμένων οικιακών συσκευών και το 2014, αυξήθηκε ακόμα περισσότερο σε 14,4 δισεκατομμύρια λόγω της χρήσης έξυπνων συσκευών. Το 2020 ο αριθμός των συσκευών που συνδέονται στο Διαδίκτυο φαίνεται ότι έχει ξεπεράσει τα 50 δισεκατομμύρια [NCTA, 2015] [Evans, 2011].

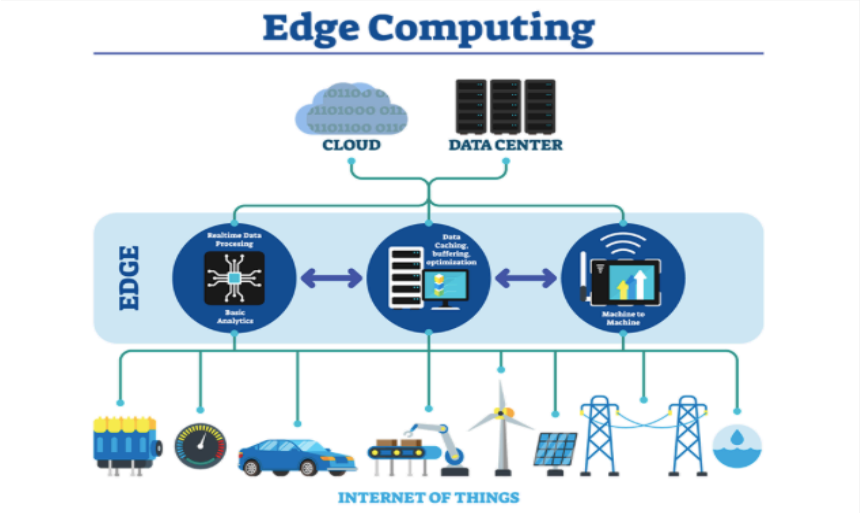
Αυτό καθιστά αδύνατη την επεξεργασία δεδομένων από αυτόν τον μεγάλο αριθμό συσκευών Internet of Things, ανεξάρτητα από το πόσο μεγάλος είναι ο διακομιστής cloud [Gezer, 2017]. Καθώς αυξάνεται η χρήση του cloud, ο χρόνος που απαιτείται για την επεξεργασία των δεδομένων αυξάνεται επίσης, με αποτέλεσμα μεγαλύτερη καθυστέρηση για τους χρήστες, γεγονός που με τη σειρά του αυξάνει το φορτίο στο διακομιστή και το δίκτυο. Επιπλέον, το cloud computing είναι ευάλωτο σε περιβάλλοντα ασφάλειας και δικτύου [Volkan et al., 2018].



Εικόνα 6: Cloud Computing. Πηγή: https://medium.com/featurepreneur/what-is-cloud-computing-290520365d48

Ως λύση σε αυτά τα προβλήματα, το edge computing προσελκύει την προσοχή. Πρόκειται για μια νέα τεχνολογία υπολογιστών που αναλύει και επεξεργάζεται άμεσα τα δεδομένα στο δίκτυο όπου συλλέγονται. Πρόκειται μια τεχνολογία όπου τα δεδομένα αναλύονται και υποβάλλονται σε επεξεργασία σε μια τερματική συσκευή και όχι σε ένα απομακρυσμένο κέντρο δεδομένων. Με αυτόν τον τρόπο, τα δεδομένα υποβάλλονται σε επεξεργασία σε πραγματικό χρόνο. Αυτό μειώνει σημαντικά την καθυστέρηση των δεδομένων και επιτρέπει στους χρήστες να έχουν πρόσβαση σε μια γρήγορη υπηρεσία [Premsankar et al., 2018].

Το Edge computing είναι μια έννοια παρόμοια με το κατανεμημένο cloud computing, όμως υπάρχουν διαφορές μεταξύ αυτών των δύο μεθόδων, όπως η φυσική απόσταση από την οποία μεταδίδονται τα δεδομένα. Εάν θεωρηθεί ότι το cloud computing είναι μια μέθοδος που χρησιμοποιείται για άμεση επικοινωνία με ένα κέντρο δεδομένων όπου αποθηκεύονται οι πόροι των υπολογιστών, στο Edge computing, ο υπολογιστής επικοινωνεί κυρίως με το λεγόμενο κέντρο δεδομένων (edge server) που επεξεργάζεται δευτερεύουσες εργασίες (λειτουργίες αποθήκευσης κ.λπ.) στον διακομιστή cloud.



Εικόνα 7: Edge Computing. Πηγή: https://forum.huawei.com/enterprise/en/edge-computing-a-cutting-edge-technology/thread/734895-893?page=1

Η κύρια διαφορά μεταξύ cloud computing και edge computing είναι ότι το πρώτο είναι ένα κεντρικό υπολογιστικό σύστημα ενώ το δεύτερο είναι ένα κατανεμημένο υπολογιστικό σύστημα. Το cloud computing επιτρέπει στο χρήστη να εκτελέσει την εργασία του χρησιμοποιώντας το κέντρο δεδομένων μέσω του διαδικτύου και το edge computing επιτρέπει στον χρήστη να χρησιμοποιήσει μια κοντινή τοπική συσκευή δικτύου (edge) [Buhalis et al., 2013]. Παρόλο που το σύστημα υπολογιστικού νέφους καταναλώνει περισσότερη ισχύ και το φορτίο των δεδομένων του μπορεί να είναι βαρύ, καθώς όλα τα δεδομένα πρέπει να υποβληθούν σε επεξεργασία στο κέντρο των δεδομένων, είναι σχετικά πιο εύκολο να επεκταθεί.

Όσον αφορά την ασφάλεια, το edge σύστημα είναι πιο ασφαλές από το σύστημα υπολογιστικού νέφους όπου συγκεντρώνονται όλα τα δεδομένα σε ένα μέρος. Έτσι, η αύξηση της ασφάλειας του συστήματος είναι δυνατή με την αποστολή των δεδομένων στο κέντρο μετά την κρυπτογράφηση τους. Επιπρόσθετα, η σταθερότητα της υποδομής του edge συστήματος αυξάνεται περισσότερο καθώς τα δεδομένα μπορούν να υποβληθούν σε επεξεργασία χωρίς διακοπή. Το σύστημα υπολογιστικού νέφους δεν μπορεί να το επιτύχει αυτό, καθώς όλοι οι εξοπλισμοί IoT που είναι συνδεδεμένοι στο κέντρο δεδομένων θα σταματήσουν να λειτουργούν όταν τερματιστεί για κάποιο λόγο.

Η σύγκριση μεταξύ των cloud computing και edge computing, δείχνει ότι η χρήση του edge computing βελτιώνει το δίκτυο ως προς τις ακόλουθες πτυχές:

• Απόδοση: μια edge συσκευή εκμεταλλεύεται πλήρως τους διαθέσιμους πόρους εκχωρώντας λειτουργίες αποθήκευσης, υπολογισμού και ελέγχου σε διαθέσιμους πόρους σε οποιοδήποτε μέρος μεταξύ του τελικού χρήστη και του cloud [Shi et al., 2016]. Αυτό επιτρέπει στις συσκευές IoT να χρησιμοποιούν αποτελεσματικά τους κοινόχρηστους υπολογιστικούς πόρους.

• Γνώση: μια edge συσκευή γνωρίζει τις απαιτήσεις των πελατών [Mao et al., 2017]. Για παράδειγμα, σε ένα σύστημα ηλεκτρονικής υγείας, ειδικά σε καταστάσεις έκτακτης ανάγκης, η φυσική υγεία των ασθενών παρακολουθείται μέσω IoT συσκευών και η κατανομή των υπολογιστικών πόρων προσαρμόζεται με βάση τον βαθμό κινδύνου για την υγεία των χρηστών [Chen et al., 2018].

• Ευελιξία: είναι πιο γρήγορο και φθηνό ώστε να μπορεί κανείς να πειραματιστεί με edge συσκευές και πελάτες, επειδή η επεξεργασία και η αποθήκευση δεδομένων γίνονται κοντά στον τελικό χρήστη [Yi et al., 2015]

• Καθυστέρηση: το edge computing υποστηρίζει κρίσιμες χρονικές εφαρμογές, επιτρέποντας την ανάλυση δεδομένων και την επεξεργασία δεδομένων κοντά στον τελικό χρήστη, η οποία παρέχει στις IoT εφαρμογές τη δυνατότητα να λαμβάνουν αποφάσεις γρηγορότερα και καλύτερα [Shi et al., 2016]

2.2. Νοημοσύνη αιχμής (Edge Intelligence)

Οι συμβατικοί υπολογιστές έχουν χαμηλές δυνατότητες νοημοσύνης. Αυτές οι συσκευές είναι υπεύθυνες για την τοπική επεξεργασία δεδομένων, όπως η εξαγωγή χαρακτηριστικών [Catarinucci et al., 2015] και τη μετάδοση δεδομένων σε διακομιστές cloud. Ωστόσο, αν εξοπλιστούν οι edge συσκευές με δυνατότητες μηχανικής μάθησης, τότε γίνονται πιο έξυπνες αυξάνοντας την ικανότητά τους να αναλύουν δεδομένα και να λαμβάνουν αποφάσεις χωρίς να χρειάζεται να συνδεθούν στο cloud [El-Sayed et al., 2017] Έτσι μειώνονται οι χρόνοι καθυστέρησης και οι χρόνοι λειτουργίας. Η μετακίνηση της νοημοσύνης από τη συσκευή cloud σε μια edge συσκευή είναι απαραίτητη για πολλούς λόγους:

• Ασφάλεια: όταν τα IoT δεδομένα μεταδίδονται σε διακομιστές cloud για ανάλυση, αυτό μπορεί να προκαλέσει ζητήματα ασφάλειας και απορρήτου κατά τη χρήση δημόσιων και ιδιωτικών υποδομών. Η επεξεργασία δεδομένων κοντά στον τελικό χρήστη προστατεύει το απόρρητο των χρηστών [Hassan et al., 2018].

• Απόδοση: σε κάποιες IoT εφαρμογές, παρατηρούνται μικρές καθυστερήσεις [Bonomi et al., 2014]. Επομένως, η επεξεργασία μέσω edge μεθόδων είναι επιθυμητή.

• Εύρος ζώνης: ο αυξανόμενος αριθμός συσκευών που δημιουργούν συνεχώς δεδομένα καταναλώνει εύρος ζώνης, όπως για παράδειγμα οι κάμερες που στέλνουν συνεχώς βίντεο και φωτογραφίες [Alrawais et al., 2017]. Η επεξεργασία δεδομένων στις edge συσκευές μειώνει την κατανάλωση εύρους ζώνης στο Διαδίκτυο.

• Ακεραιότητα δεδομένων: η μετάδοση δεδομένων σε edge συσκευές δεν απαιτεί συμπίεση ή τροποποίηση της μορφής δεδομένων. Επίσης, τα δεδομένα δεν εκτίθενται σε θόρυβο κατά τη διαδικασία μετάδοσης [Alrawais et al., 2017].

Η εφαρμογή αλγορίθμων μηχανικής μάθησης (ML) σε συσκευές υπολογιστών αιχμής παρουσιάζει πολλές προκλήσεις:

• Πολυπλοκότητα: Οι αλγόριθμοι μηχανικής μάθησης συνήθως για να εκτελεστούν χρειάζονται ισχυρές συσκευές σε πόρους, όπως μεγάλη υπολογιστική ισχύς και μνήμη. Από την άλλη πλευρά, οι edge συσκευές ενδέχεται να διαφέρουν ως προς τους πόρους. Επομένως, απαιτούνται τεχνικές μηχανικής μάθησης χαμηλής πολυπλοκότητας [Plastiras et al., 2018].

• Περιορισμοί μνήμης: Οι τεχνικές τεχνητής νοημοσύνης (AI), όπως τα νευρωνικά δίκτυα, απαιτούν πολύ χώρο μνήμης. Επομένως, υπάρχει ανάγκη σχεδιασμού τεχνικών τεχνητής νοημοσύνης που να μπορούν να λειτουργούν σε συσκευές περιορισμένης χρήσης πόρων [Plastiras et al., 2018].

Ένα παράδειγμα τεχνικών μηχανικής μάθησης που χρησιμοποιούνται στον υπολογιστικό άξονα είναι η βαθιά μάθηση (deep learning). Οι τεχνικές βαθιάς μάθησης είναι νευρωνικά δίκτυα πολλαπλών επιπέδων, με κάθε επίπεδο να είναι υπεύθυνο για την εξαγωγή συγκεκριμένων χαρακτηριστικών από το σύνολο δεδομένων εισόδου. Η βαθιά μάθηση είναι κατάλληλη για edge υπολογιστές [Li, et al., 2018] .

2.3. Εφαρμογές και τεχνολογίες αιχμής

Η τεχνολογία του Edge Computing βρίσκει αρκετές εφαρμογές στη σημερινή εποχή.

- Αυτο-οδηγούμενο όχημα

Τα αυτόνομα οχήματα είναι τα αυτοκίνητα που αναγνωρίζουν το περιβάλλον οδήγησης χωρίς ανθρώπινη παρέμβαση, κρίνουν αν υπάρχει κίνδυνος και σχεδιάζουν τη διαδρομή. Περιλαμβάνουν συστήματα ανίχνευσης εμποδίων, κεντρικές συσκευές ελέγχου και απαιτούν τη χρήση προηγμένων τεχνολογιών όπως η ρομποτική και η μηχανική υπολογιστών, το GPS, οι αισθητήρες και ο ηλεκτρονικός έλεγχος [Alcala et al., 2016]. Οι κύριες λειτουργίες στα αυτόνομα οχήματα είναι: η λειτουργία της ανίχνευσης του κινδύνου και η λειτουργία ελέγχου του οχήματος. Η δυνατότητα ανίχνευσης του κινδύνου περιλαμβάνει τις λειτουργίες της προειδοποίησης παραβίασης της λωρίδας κυκλοφορίας, την ένδειξη του ορίου ταχύτητας, την προειδοποίηση ενδεχόμενης σύγκρουσης, την αυτόματη πέδηση σε περίπτωση έκτακτης ανάγκης, τον έλεγχο της ταχύτητας, κ.α.
Η ανάπτυξη μιας τέτοιας τεχνολογίας προχωρά αρκετά γρήγορα, ωστόσο, η τρέχουσα αξιοπιστία και ακρίβεια αυτής της τεχνολογίας είναι ανεπαρκής για τα αυτόνομα αυτοκίνητα. Αυτό συμβαίνει επειδή ο αισθητήρας που χρησιμοποιείται δεν μπορεί να συλλέξει και να επεξεργαστεί δεδομένα σε επίπεδο ίδιο με εκείνο του ανθρώπου [Eliot, 2018].

- Έξυπνο εργοστάσιο

Ένα έξυπνο εργοστάσιο είναι ένα εργοστάσιο στο οποίο οι αισθητήρες (που χρησιμοποιούνται για IOT) υλοποιούνται στις εργοστασιακές εγκαταστάσεις και τα μηχανήματα αλλά και τα δεδομένα μπορούν να συλλεχθούν και να αναλυθούν σε πραγματικό χρόνο.
Καθώς η ύφεση της μεταποιητικής βιομηχανίας είχε μεγάλο αντίκτυπο στην οικονομία λόγω της παγκόσμιας χρηματοπιστωτικής κρίσης, ο κόσμος δίνει προσοχή στη σημασία της κατασκευής και της παραγωγής εναλλακτικών λύσεων και ιδιαίτερα στην καινοτομία. Έτσι η οικοδόμηση

ενός έξυπνου εργοστασίου, θα αποτελούσε μια ολοκληρωμένη λύση για τη δημιουργία νέας αξίας πέρα από την παραγωγή.

Οι λόγοι που οι εταιρείες ξεκινούν ή επεκτείνονται στα έξυπνα εργοστάσια ποικίλουν. Ωστόσο, θα μπορούσαν να αναφερθούν ως τέτοιοι λόγοι η αποδοτικότητα, η ποιότητα, το κόστος, η ασφάλεια και η βιωσιμότητα. Αυτές οι κατηγορίες, μεταξύ άλλων, μπορούν να αποδώσουν οφέλη που τελικά θα οδηγήσουν σε αυξημένη κερδοφορία, ποιοτικότερα προϊόντα και σε σταθερότητα του εργατικού δυναμικού.

Κάθε πτυχή του έξυπνου εργοστασίου δημιουργεί δεσμίδες δεδομένων που, μέσω της συνεχούς ανάλυσης, αποκαλύπτουν θέματα επιδόσεων που απαιτούν κάποιο είδος διορθωτικής βελτιστοποίησης σε σχέση με την αποδοτικότητα των περιουσιακών στοιχείων. Πράγματι, μια τέτοια αυτοδιάθεση είναι αυτό που διακρίνει το έξυπνο εργοστάσιο από μια παραδοσιακή διαδικασία αυτοματοποίησης [Lee et al., 2015].

Η αυτο-βελτιστοποίηση που είναι χαρακτηριστική του έξυπνου εργοστασίου μπορεί να προβλέψει και να ανιχνεύσει τις τάσεις της ποιότητας των μηχανημάτων πιο γρήγορα και να βοηθήσει στον εντοπισμό των αιτιών της κακής ποιότητας μηχανημάτων. Έτσι, θα μπορούσαν να μειωθούν τα ποσοστά των απορριμμάτων και οι χρόνοι ακεραιότητας των μηχανημάτων και να αυξηθεί η απόδοση. Μια πιο βελτιστοποιημένη διαδικασία ποιότητας θα μπορούσε να οδηγήσει σε μια καλύτερη ποιότητα προϊόντων με λιγότερα ελαττώματα και συνεπώς με λιγότερες ανακλήσεις προϊόντων [ROI, 2017].

Οι βελτιστοποιημένες διαδικασίες παραδοσιακά οδηγούν σε πιο οικονομικά αποδοτικές διαδικασίες, σε αποτελεσματικότερες αποφάσεις για τη μίσθωση και τη στελέχωση, καθώς και στη μειωμένη μεταβλητότητα των επιχειρήσεων. Και επειδή μια διαδικασία καλύτερης ποιότητας μπορεί επίσης να σημαίνει ένα προϊόν καλύτερης ποιότητας, θα μπορούσε επίσης να σημαίνει χαμηλότερο κόστος εγγύησης και συντήρησης [Bartodziej, 2017].

- Smart City

Η έξυπνη πόλη είναι μια αστική περιοχή στην οποία οι διάφοροι τύποι αισθητήρων ηλεκτρονικών δεδομένων χρησιμοποιούνται για την παροχή των απαραίτητων πληροφοριών, για

την αποτελεσματική διαχείριση των περιουσιακών στοιχείων και των πόρων [Gezer et al., 2017]. Περιλαμβάνει τα δεδομένα που συλλέγονται από τους πολίτες, τις συσκευές και τα περιουσιακά στοιχεία, τα οποία στη συνέχεια επεξεργάζονται ή αναλύονται για να χρησιμοποιηθούν στην παρακολούθηση και τη διαχείριση συστημάτων μεταφοράς, των σταθμών ηλεκτροπαραγωγής, τα δίκτυα ύδρευσης, της διαχείρισης των αποβλήτων, της επιβολής του νόμου, των σχολείων, των βιβλιοθηκών, των νοσοκομείων, των σχολείων, των σχολών, των βιβλιοθηκών, των νοσοκομείων κ.α. Η έννοια της έξυπνης πόλης περιλαμβάνει την ενσωμάτωση των τεχνολογιών του Διαδικτύου, των πληροφοριών και της επικοινωνίας, οι οποίες περιλαμβάνουν διάφορες φυσικές συσκευές που συνδέονται με το δίκτυο για τη βελτιστοποίηση της αποτελεσματικότητας των επιχειρήσεων και των υπηρεσιών της πόλης.

Στην περίπτωση μιας έξυπνης πόλης, είναι απαραίτητο να συλλεχθούν, να αναλυθούν και να επεξεργαστούν μεγάλες ποσότητες δεδομένων για την εκτέλεση όλων των προαναφερθέντων λειτουργιών, οι οποίες προκαλούν πλημμύρες δεδομένων. Οι πλημμύρες δεδομένων αποτελούν μια πρόκληση που απαιτείται να αντιμετωπιστεί προκειμένου να υλοποιηθούν οι βιώσιμες έξυπνες πόλεις, οι οποίες είναι δύσκολες και δαπανηρές ως προς τη διαχείριση. Η ανάλυση αιχμής (edge analysis) επεξεργάζεται τα δεδομένα που συλλέγονται από τους αισθητήρες, τους διακόπτες δικτύου και τις άλλες συσκευές χρησιμοποιώντας τον αλγόριθμο ανάλυσης αιχμής για να προσδιοριστεί η τιμή των πληροφοριών και η αποστολή ή η απόρριψη πληροφοριών στο νέφος. Έτσι, μειώνεται ο χρόνος λήψης αποφάσεων της συνδεδεμένης συσκευής. Η ανάλυση αιχμής απαιτεί πλατφόρμες υλικού και λογισμικού για την αποθήκευση, την προετοιμασία και την εκπαίδευση των αλγορίθμων καθώς και την επεξεργασία και την αποθήκευση των δεδομένων.

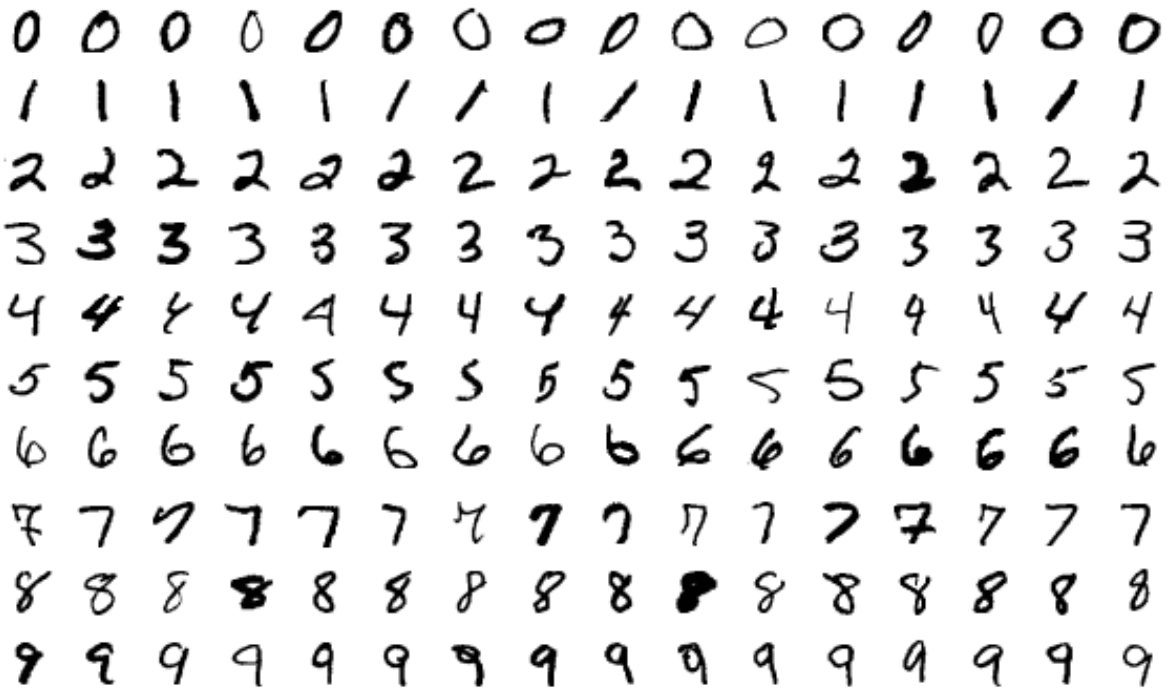
Κεφάλαιο 3: Εφαρμογές κατανεμημένης μηχανικής μάθησης

Στη διεθνή βιβλιογραφία θα εντοπίσει κανείς πληθώρα εφαρμογών κατανεμημένης μηχανικής μάθησης. Στην ενότητα αυτή θα παρουσιαστούν οι αντιπροσωπευτικότερες από αυτές ώστε να σχηματίσει ο αναγνώστης μια σφαιρική εικόνα του θέματος.

3.1 Κατανεμημένο σύστημα αναγνώρισης χαρακτήρων

Η παρούσα εργασία [Tiffany et all, 2018] αφορά την υλοποίηση ενός συστήματος μηχανικής μάθησης το οποίο εκπαιδεύει ένα μοντέλο ώστε να είναι σε θέση να ταξινομήσει δεδομένα. Το σύστημα είναι κατανεμημένο σε πέντε συνολικά κόμβους εκ των οποίων οι δύο είναι φορητοί υπολογιστές (laptop) και οι υπόλοιποι τρείς αποτελούνται από τα Raspberry Pi τα οποία αποτελούν ολοκληρωμένους ηλεκτρονικούς υπολογιστές σε μέγεθος πιστωτικής κάρτας. Οι κόμβοι επικοινωνούν μεταξύ τους μέσω ασύρματου δικτύου WiFi.

Οι ερευνητές χρησιμοποίησαν τα δεδομένα του MNIST προκειμένου να εκπαιδεύσουν ένα μοντέλο να ταξινομεί χειρόγραφους χαρακτήρες. Η βάση δεδομένων αποτελείται από 70.000 εικόνες που εμφανίζουν χειρόγραφα ψηφία. Από αυτά τα 60.000 προορίζονται για εκπαίδευση και τα υπόλοιπα 10.000 για τη δοκιμή του μοντέλου.



Εικόνα 8: Δείγμα της βάσης δεδομένων MNIST. Πηγή: https://en.wikipedia.org/wiki/MNIST_database

Η βάση δεδομένων μοιράζεται στους επιμέρους κόμβους και με βάση αυτά ο κάθε κόμβος ξεχωριστά παράγει το δικό του μοντέλο τύπου μηχανής διανυσμάτων υποστήριξης (Support Vector Machine). Ακολούθως ενημερώνει το φορητό υπολογιστή ο οποίος εκτελεί το ρόλο του ενορχηστρωτή. Αυτός συγκεντρώνει τα επιμέρους βάρη από τους υπόλοιπους κόμβους και κατόπιν τους ενημερώνει εκ νέου για τις επικρατέστερες τιμές ώστε να ενημερώσουν το μοντέλο τους. Η όλη επικοινωνία πραγματοποιείται μέσω ενός πρωτοκόλλου που οι ερευνητές έχουν αναπτύξει και που επιτρέπει στην όλη διαδικασία να προσαρμόζεται σύμφωνα με τις ανάγκες του κάθε μοντέλου ξεχωριστά. Ποιο συγκεκριμένα το μοντέλο επιτρέπει στους κόμβους να διαπραγματεύονται τη συχνότητα των ενημερώσεων. Η συχνότητα αυτή εξαρτάται από διάφορους παράγοντες εκ των οποίων η σημαντικότερη είναι η απόκλιση μεταξύ των τοπικών βαρών και των γενικών. Όσο μεγαλύτερη είναι η απόκλιση αυτή, τόσο ποιο σύντομα θα πρέπει να γίνονται οι ενημερώσεις. Στην αντίπερα όχθη, οι διαθέσιμοι πόροι του συστήματος περιορίζουν τη συχνότητα των ενημερώσεων. Ο φορητός υπολογιστής που εκτελεί το ρόλο του

ενορχηστρωτή, όταν παρατηρεί ότι οι διαθέσιμοι πόροι εξαντλούνται μειώνει τον ρυθμό των ενημερώσεων ή και σταματά τη διαδικασία της εκπαίδευσης.

Για την κατανομή των δεδομένων οι ερευνητές εφάρμοσαν τέσσερα διαφορετικά σενάρια με σκοπό να μελετήσουν τις διαφορετικές συμπεριφορές του συστήματος. Στο πρώτο σενάριο μοίρασαν τα δεδομένα με τυχαίο τρόπο στους τρείς κόμβους. Στο δεύτερο σενάριο σε κάθε κόμβο έδωσαν διαφορετικά δεδομένα και ποιο συγκεκριμένα δύο ψηφία ανά κόμβο. Στην τρίτη περίπτωση διέθεσαν το σύνολο των δεδομένων σε κάθε κόμβο. Τέλος στη τελευταία περίπτωση συνδύασαν το πρώτο σενάριο με το δεύτερο δίνοντας στους πρώτους τρείς κόμβους τα ψηφία μηδέν έως και τέσσερα και τα υπόλοιπα μοιράστηκαν στους υπόλοιπους.

Για την παρουσίαση των αποτελεσμάτων οι ερευνητές ανέπτυξαν μια παραθυρική εφαρμογή που επιτρέπει στο χρήστη να επιλέξει τον τρόπο με τον οποίο θα κατανέμονται τα δεδομένα στους επιμέρους κόμβους και ποιο συγκεκριμένα πιο από τα σενάρια που περιγράφηκαν θα εφαρμοστεί. Επιπλέον ο χρήστης έχει τη δυνατότητα να εισάγει στο σύστημα μια καθυστέρηση στην επικοινωνία μεταξύ των κόμβων. Ακολούθως η εφαρμογή παρουσιάζει μετρήσεις στο χρήστη κατά το χρόνο εκτέλεσης που περιλαμβάνουν διάφορες τιμές όπως την ακρίβεια του μοντέλου, τις απώλειες, τα διαστήματα ενημερώσεων κλπ.

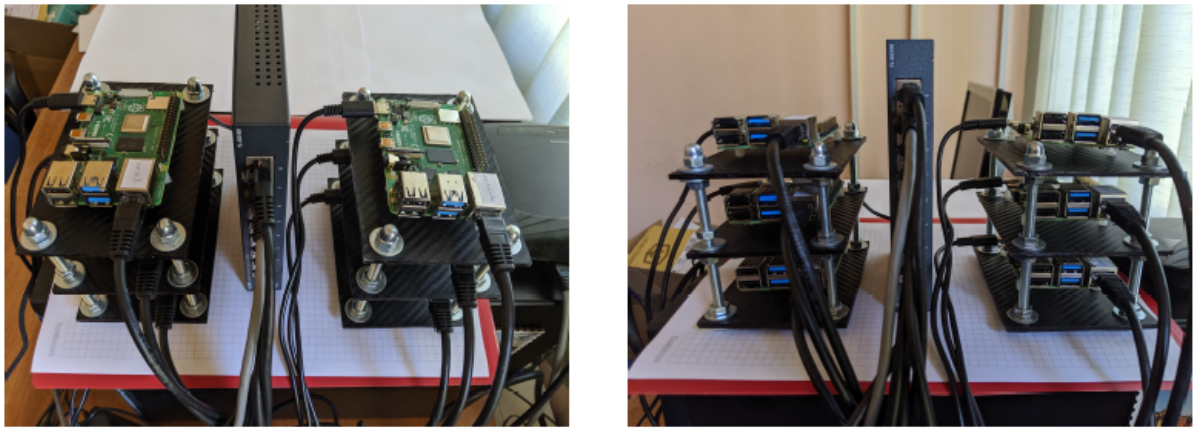
3.2 Κατανεμημένο σύστημα δημιουργίας συστημάτων προτάσεων

Ενδιαφέρον παρουσιάζει και η δουλειά των Κομνινός, Σίμου, Γκοργκόλης και Γκαροφαλάκης [Komninos et all, 2019]. Οι ερευνητές επισημαίνουν την βασική χρήση της μηχανικής μάθησης η οποία δεν είναι άλλη από την επεξεργασία μεγάλου όγκου δεδομένων με στόχο την παραγωγή διαφόρων αναλύσεων, την παραγωγή προτάσεων σε καταναλωτές καθώς και την παραγωγή διαφόρων ειδών προβλέψεων. Η επεξεργασία αυτή των δεδομένων μπορεί να γίνεται είτε σε πραγματικό χρόνο, είτε σε διαφορετική χρονική στιγμή και κατόπιν να παρουσιάζονται τα αποτελέσματα. Τα δεδομένα προκύπτουν από τους διάφορους κόμβους οι οποίοι αφού τα παράγουν τα αποστέλλουν προς επεξεργασία σε ένα κεντρικό κόμβο ο οποίος είτε τα αποθηκεύει

ώστε να τα επεξεργαστεί σε μελλοντικό χρόνο, είτε τα επεξεργάζεται άμεσα και ακολούθως τους επιστρέφει τα αποτελέσματα που παρήγαγε το μοντέλο μηχανικής μάθησης που εφαρμόζει. Ο κεντρικός κόμβος αυτός θα μπορούσε να βρίσκεται κοντά στους κόμβους που παράγουν τα δεδομένα ή σε κάποια υποδομή τύπου υπολογιστικό νέφος. Η δεύτερη επιλογή ενώ απολαμβάνει το σύνολο των πλεονεκτημάτων που έχει να προσφέρει η υπολογιστική νέφους παρουσιάζει σημαντικά μειονεκτήματα όπως τους περιορισμούς που επιφέρει το δίκτυο καθώς και θέματα αξιοπιστίας μιας και ένας μόνο κεντρικός υπολογιστής θα θέσει το σύνολο της υποδομής εκτός λειτουργίας στη περίπτωση που πάψει να λειτουργεί.

Για την αντιμετώπιση των προβλημάτων αυτών οι ερευνητές αναφέρουν το Edge Computing το οποίο θα μοιράσει την αποθήκευση αλλά και την επεξεργασία των δεδομένων στους διάφορους κόμβους, φέρνοντας έτσι τα αποτελέσματα πιο κοντά στο χρήστη. Με τον τρόπο αυτό το σύστημα γίνεται πιο γρήγορο και πιο ανθεκτικό με την παραδοχή όμως ότι οι επιμέρους κόμβοι δεν μπορούν να διαθέτουν τους ίδιους πόρους με το κεντρικό σύστημα και συνεπώς δεν είναι σε θέση να παράξουν ένα εξίσου αποτελεσματικό μοντέλο. Για την αντιμετώπιση των αδυναμιών αυτών οι ερευνητές προτείνουν ένα καταμερισμό ρόλων όπου οι κόμβοι θα επεξεργάζονται τα δεδομένα, ώστε να περιορίζεται ο όγκος που διακινείται και ακολούθως αυτά να αποστέλλονται στον κεντρικό εξυπηρετητή ο οποίος θα παράγει το μοντέλο και ακολούθως θα το αποστέλλει στους κόμβους.

Σε μια προσπάθεια να αυξήσουν τις δυνατότητες των επιμέρους κόμβων, οι ερευνητές προτείνουν τη δημιουργία ομάδων συστημάτων (Clusters) με τη σύνδεση πολλαπλών συσκευών Raspberry Pi. Με τον τρόπο αυτό οι ερευνητές καταφέρνουν την μεταφορά πλεονεκτημάτων της υπολογιστικής νέφους στους επιμέρους κόμβους του Edge Computing. Ποιο συγκεκριμένα η διασύνδεση περισσότερων της μιας συσκευής Raspberry Pi θα δημιουργήσει ένα σύστημα επεκτάσιμο που θα μπορεί να κλιμακωθεί ώστε να καλύψει τις επιπλέων υπολογιστικές ανάγκες που θα προκύψουν, διατηρώντας το κόστος σε χαμηλά επίπεδα, ενώ παράλληλα με τη χρήση έτοιμων βιβλιοθηκών θα είναι σε θέση να ανταποκριθεί σε αρκετά απαιτητικές διεργασίες.



Εικόνα 9: Υπολογιστική Νέφους από Raspberry Pi 4. Πηγή: Performance of Raspberry Pi microclusters for Edge Machine Learning in Tourism, Computer Science.

Πιο συγκεκριμένα, οι ερευνητές χρησιμοποίησαν έξι Raspberry Pi 4, όπου το κάθε ένα ήταν εφοδιασμένο με 4 GB μνήμης. Για την τοπική αποθήκευση των δεδομένων κάθε συσκευή έφερε μία κάρτα μνήμης τύπου MicroSD χωρητικότητας 64GB, ενώ η ταχύτητα εγγραφής των δεδομένων άγγιζε τα 30Mb/s. Για τη διασύνδεση των συσκευών χρησιμοποιήθηκε ένα δίκτυο τύπου Gigabit ενώ από πλευράς βιβλιοθηκών χρησιμοποιήθηκαν οι Hadoop για το κατανεμημένο σύστημα αρχείων και η Apache Spark για την επεξεργασία των δεδομένων.

Για την εκτέλεση των διεργασιών της ομάδας, η μία συσκευή επιτελεί το ρόλο του πελάτη, μία συσκευή το ρόλο του διαχειριστή των εφαρμογών και οι υπόλοιπες τέσσερις εκτελούν τις διάφορες εργασίες που τους αναθέτει ο διαχειριστής. Παράλληλα, σε τοπικό επίπεδο, από την κάθε συσκευή διατέθηκαν 2 Gb από τα 4 Gb της μνήμης στο YARN Container, 1 Gb στο διαχειριστή και τέλος το υπόλοιπο 1 για να λειτουργήσει το λειτουργικό σύστημα της συσκευής.

Για την εξαγωγή συμπερασμάτων οι ερευνητές χρησιμοποίησαν δεδομένα από την Kaggle, μια θυγατρική εταιρία της Google η οποία επιτρέπει στις εταιρίες να δημοσιεύσουν τα προβλήματα τους και κατόπιν οι διάφοροι χρήστες συναγωνίζονται στο να δημιουργήσουν τον πλέον αποδοτικότερο αλγόριθμο αντιμετώπισης του προβλήματος, λαμβάνοντας ως ανταμοιβή χρηματικά έπαθλα. Η εταιρία διαθέτει και δεδομένα τα οποία μπορούν να χρησιμοποιήσουν οι χρήστες για να εκπαιδεύσουν τα μοντέλα τους. Από τα δεδομένα αυτά οι ερευνητές επέλεξαν δεδομένα μεταχειρισμένων αυτοκινήτων και δεδομένα σημείων που επισκέπτονται οι τουρίστες

σε συνδυασμό με τα δεδομένα καιρού της Ελλάδας. Ποιο συγκεκριμένα, από τα δεδομένα των αυτοκινήτων συνδύασαν την ηλικία του οχήματος με τον αριθμό των χιλιομέτρων που έχει διανύσει με στόχο να προβλέψουν την τιμή πώλησης του. Τα δεδομένα επισκεψιμότητας από την άλλη πλευρά σε συνδυασμό με αυτά του καιρού χρησιμοποιήθηκαν για να προβλέψουν το είδος των αξιοθέατων που ένας χρήστης ενδέχεται να επιλέξει να επισκεφτεί.

Οι ερευνητές αρχικά σύγκριναν την απόδοση των γλωσσών προγραμματισμού Python και Scala όπου κατέληξαν ότι η Scala είναι αποδοτικότερη όσο αυξάνει ο αριθμός των κόμβων που έχουν αναλάβει το ρόλο του διεκπεραιωτή των διεργασιών. Ακολούθως εφάρμοσαν δύο αλγόριθμους μηχανικής μάθησης (Linear Regression και Δέντρα Απόφασης) στα δύο διαφορετικά σετ δεδομένων με στόχο να μετρήσουν τις διαφορές στην απόδοση που θα προκαλέσει η διαφοροποίηση του αριθμού των κόμβων που διεκπεραιώνουν τις διεργασίες καθώς και οι διαφορές στους πυρήνες που διαθέτει ο κάθε κόμβος. Στα συμπεράσματα τους αναφέρουν ότι η αύξηση του αριθμού των κόμβων που διαπεραιώνουν τις διεργασίες αυξάνει την αποδοτικότητα της ομάδας με μοναδική εξαίρεση αυτή της φάσης της εκκαθάρισης των δεδομένων. Στη φάση αυτή επειδή απαιτείται οι κόμβοι να ανταλλάσσουν πολλά δεδομένα η καθυστέρηση που μπορεί να φέρει το δίκτυο μειώνει την απόδοση του συστήματος και για το λόγο αυτό ο μεγαλύτερος αριθμός κόμβων έχει σαν συνέπεια τη μείωση της απόδοσης.

3.3 Σύγκριση απόδοσης Αλγορίθμων

Σχετικά με τις επιδόσεις της συσκευής κατά την εφαρμογή των συστημάτων μηχανικής μάθησης στο Raspberry Pi θα εντοπίσει κανείς αρκετές μελέτες στη διεθνή βιβλιογραφία. Μία εξ αυτών [Mahmut et all, 2018] παρουσιάζει την εφαρμογή των αλγόριθμων Random Forest, Support Vector Machine και Multi-Layer Perception σε δέκα διαφορετικά είδη δεδομένων τόσο στη διαδικασία της εκπαίδευσης όσο και εφαρμογής των παραγόμενων μοντέλων και παρουσιάζουν τις αποδόσεις της συσκευής στο τομέα της ταχύτητας, της ακρίβειας του μοντέλου αλλά και της κατανάλωσης ενέργειας.

Οι ερευνητές επισημαίνουν, μεταξύ άλλων, τη σημασία της χρήσης συσκευών Internet Of Things στο Edge Computing σε μια προσπάθεια εξοικονόμησης πόρων. Η εξοικονόμηση αυτή επιτυγχάνεται καθώς οι συσκευές έχουν τη δυνατότητα να εκπαιδευτούν και να χρησιμοποιούν τους διαθέσιμους πόρους της υποδομής με πλέον αποδοτικότερο τρόπο. Σαν παράδειγμα αναφέρουν την κίνηση της Google να εφαρμόσει συστήματα μηχανικής μάθησης τα οποία εκπαιδεύτηκαν στις ανάγκες ψύξεις των συστημάτων της και πλέον αποφασίζουν μόνα τους πώς θα λειτουργήσουν οδηγώντας με τον τρόπο αυτό σε μια εξοικονόμηση ενέργειας της τάξης του 15%, ποσοστό το οποίο αντιστοιχεί σε εκατοντάδες εκατομμύρια δολάρια.

Στο εμπόριο κυκλοφορούν διάφορες συσκευές οι οποίες θα μπορούσαν να χρησιμοποιηθούν για να φιλοξενήσουν το μοντέλο και η επιλογή του κατάλληλου εξαρτάται σε μεγάλο βαθμό από το είδος της εφαρμογής. Οι ερευνητές, μεταξύ των διαθέσιμων αυτών επιλογών επέλεξαν το Raspberry Pi λόγο της χαμηλής τιμής του, της δυνατότητας του να εκτελεί το λειτουργικό σύστημα Linux και τέλος επειδή μπορεί εύκολα να συνδεθεί με πληθώρα αισθητήρων που ενδεχομένως θα χρησιμοποιήσει το μοντέλο είτε για να εκπαιδευτεί, είτε για να ελέγξει. Ακολούθως εφάρμοσαν τους αλγόριθμους που αναφέρθηκαν ώστε να μελετήσουν το κατά πόσο μπορεί η συσκευή να παράξει ένα ακριβές μοντέλο καθώς επίσης και τις ποσότητες ενέργειας που θα απαιτηθούν, δεδομένου ότι το σύστημα μπορεί να βρίσκεται σε ένα σημείο που να μην επιτρέπει τη διασύνδεση του με το δίκτυο ηλεκτρικού ρεύματος και κατά συνέπεια να αντλεί την απαιτούμενη ενέργεια από συσσωρευτή.

Για τη μελέτη της απόδοσης του συστήματος οι ερευνητές επέλεξαν μια μεγάλη ποικιλία που περιλαμβάνει δεδομένα που περιγράφουν την ποιότητα του αέρα, την πρόβλεψη δασικών πυρκαγιών, περιπτώσεων αυτισμού αλλά και καρκίνο του στήθους. Καθίσταται προφανές στον αναγνώστη ότι οποιοδήποτε σύστημα για να είναι αποδεκτό θα πρέπει να μπορεί να ανταποκριθεί στην ικανοποιητική αξιοποίηση των δεδομένων αυτών, εντός αποδεκτού χρονικού διαστήματος και με όσο το δυνατόν ποιο περιορισμένη κατανάλωση ενέργειας. Δεδομένων των περιορισμένων πόρων που διαθέτει το Raspberry οι ερευνητές επισημαίνουν ότι οι εξαιρετικά απαιτητικές διεργασίες θα μπορούσαν να ανατεθούν προς εκτέλεση σε πιο δυνατά συστήματα,

όπως για παράδειγμα σε μια δομή στο νέφος και κατόπιν τα αποτελέσματα να αποσταλούν πίσω στο Raspberry ώστε να τα εφαρμόσει.

Μεταξύ των διαθέσιμων αλγόριθμων οι ερευνητές εστίασαν στους αλγόριθμους που περιγράφηκαν για το λόγο ότι είναι οι πλέον κατάλληλοι για να δοκιμάσουν το Raspberry Pi. Ποιο συγκεκριμένα οι Multi-Layer Perception, επειδή ακριβώς αποτελούνται από πολλαπλά επίπεδα, οι ερευνητές μπορούν να ξεκινήσουν αρχικά με ένα ρηχό σχήμα (με μικρό αριθμό επιπέδων) και ακολούθως να προσθέτουν επίπεδα μελετώντας την απόδοση του συστήματος καθώς επίσης και το χρόνο εκπαίδευσης, την κατανάλωση ενέργειας αλλά και τις δυνατότητες αποθήκευσης δεδομένων. Από την άλλη πλευρά οι Random Forest επιλέχθηκαν λόγω του μεγάλου εύρους εφαρμογών που μπορούν να καλύψουν αλλά και του γεγονότος ότι είναι ιδιαίτερα απλοί στη χρήση και τέλος οι Support Vector Machine λόγο της υψηλής απόδοσης τους σε πολυδιάστατα δεδομένα.

Για την εκτέλεση των πειραμάτων οι ερευνητές χρησιμοποίησαν ένα συνηθισμένο υπολογιστή εφοδιασμένο με τον επεξεργαστή Intel Core i7-6700HQ ενώ για τη συσκευή IoT επέλεξαν το Raspberry Pi 3 το οποίο διαθέτει επεξεργαστή τεσσάρων πύρινων ταχύτητας 1.2 GHz, μνήμη 1 GB, συσκευή ασύρματου δικτύου και Bluetooth. Από πλευράς εφαρμογών χρησιμοποίησαν την σουίτα Anaconda η οποία είναι εφοδιασμένη με τις πλέον κατάλληλες για την περίπτωση εφαρμογές.
Η απόδοση του συστήματος μετρήθηκε σε ακρίβεια με τη βοήθεια της βιβλιοθήκης scikit-learn. Επιπλέον μετρήθηκε το χρονικό διάστημα που απαιτούνταν προκειμένου ο κάθε αλγόριθμος να ολοκληρωθεί. Η μέτρηση αυτή πραγματοποιήθηκε με τη χρήση της βιβλιοθήκης χρόνου της Python. Τέλος οι ερευνητές σύγκριναν και την ηλεκτρική ενέργεια που απαιτούσε ο κάθε αλγόριθμος. Για την μέτρηση αυτή χρησιμοποίησαν τη συσκευή Muker USB μέσω της οποίας σύγκριναν την ενέργεια που κατανάλωνε η συσκευή καθώς εκτελούσε τον κάθε αλγόριθμο με αυτή που κατανάλωνε σε κατάσταση ηρεμίας. Έτσι κατέληξαν στη πόση ενέργεια απαιτούσε ο κάθε αλγόριθμος.

Εικόνα 10: Muker USB. Πηγή: https://www.researchgate.net/figure/The-Muker-USB-multimeter-which-was-used-to-measure-the-energy-consumption-of-running-MLs_fig4_327406558.

Μελετώντας τα αποτελέσματα οι ερευνητές καταλήγουν ότι η εκπαίδευση του αλγόριθμου θα πρέπει να προτιμάται να μην γίνεται από συσκευές τύπου IoT καθώς είναι απαιτητική τόσο σε χρόνο - υπολογιστική ισχύ όσο και σε ηλεκτρική ενέργεια. Η εφαρμογή όμως του μοντέλου είναι δυνατόν να υλοποιηθεί από τις συσκευές αυτές καθώς η κατανάλωση ηλεκτρικής ενέργειας για την εφαρμογή των μοντέλων είναι μικρότερη από αυτή που απαιτείται για συνηθισμένη χρήση της συσκευής όπως περιήγηση στο διαδίκτυο ή η παρακολούθηση βίντεο.

Από την πλευρά των αλγορίθμων ο Random Forest αποδείχθηκε βέλτιστος χωρίς όμως να έχει ιδιαίτερα μεγάλη διαφορά από τους υπόλοιπους. Μάλιστα, οι συγγραφείς σημειώνουν ότι σε προβλήματα ταξινόμησης ο Support Vector Machine παρουσίασε υψηλότερη ακρίβεια.

3.4 Εντοπισμός μήλων

Οι περιορισμένες δυνατότητες του Raspberry στο τομέα της μηχανικής μάθησης προέτρεψαν αρκετούς ερευνητές στη διερεύνηση νέων μεθόδων βελτίωσης της απόδοσης. Η συσκευή μπορεί να εφαρμόσει ένα μοντέλο που έχει ήδη δημιουργηθεί από κάποια ισχυρή υπολογιστική δομή σε

κάποιο νέφος και να εξάγει αποτελέσματα, η περιορισμένη ισχύς της περιορίζει τις δυνατότητες της. Για παράδειγμα η συσκευή θα μπορούσε να εφαρμόσει το μοντέλο που θα της επιτρέπει να εντοπίζει αντικείμενα σε εικόνες όμως οι περιορισμένες δυνατότητες της δεν τις επιτρέπουν να επεξεργάζεται υψηλό αριθμό εικόνων ανά δευτερόλεπτο πράγμα που περιορίζει σημαντικά του τομείς που θα μπορούσε να χρησιμοποιηθεί η υποδομή.

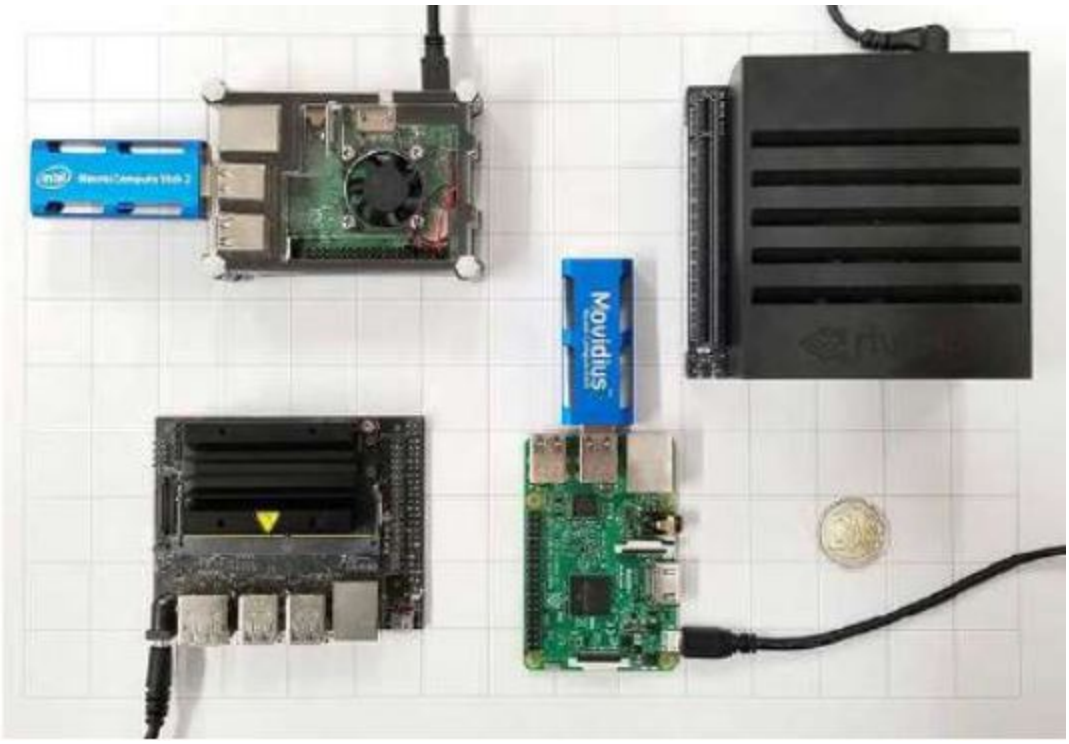
Μεταξύ των εργασιών που εστιάζουν στο θέμα, στη παρούσα εργασία θα παρουσιαστεί μία [Vittorio et all, 2019] η οποία μελετά την υλοποίηση ενός συστήματος εντοπισμού μήλων μέσα σε ένα περιβόλι. Σκοπός της εργασίας είναι να δημιουργηθεί ένα σύστημα το οποίο να μπορεί να ενσωματωθεί σε ένα αυτοκινούμενο όχημα το οποίο θα κινείται εντός των ορίων ενός περιβολιού και θα εκτιμά την ποσότητα των ώριμων φρούτων προς συγκομιδή παρέχοντας στον αγρότη τα δεδομένα εκείνα που θα τον βοηθήσουν να αποφασίσει τις κατάλληλες παραμέτρους για την εργασία του όπως τον αριθμό των εργατών που θα χρειαστεί για να μαζέψουν τους καρπούς ή τον αριθμό των οχημάτων που θα απαιτηθούν, κ.α. Η εργασία αυτή, έως τώρα, διεκπεραιώνεται από εξειδικευμένο προσωπικό, είναι ιδιαίτερα απαιτητική σε χρόνο και αυξάνει κατά πολύ το συνολικό κόστος. Επιπλέον, στη περίπτωση που δεν επιτευχθεί η απαιτούμενη ακρίβεια το κόστος θα είναι αρκετά υψηλό. Παράλληλα, παραδοσιακές τεχνικές μηχανικής μάθησης που προϋποθέτουν ισχυρά υπολογιστικά συστήματα δεν μπορούν να χρησιμοποιηθούν στο περιβόλι μιας και είναι ιδιαίτερα ευαίσθητα συστήματα, βαριά και ενεργοβόρα. Για τους λόγους αυτούς οι ερευνητές προτείνουν την χρήση οικονομικών συσκευών, όπως το Raspberry για την λήψη των εικόνων, ακολούθως την αποστολή τους σε μια ισχυρή υπολογιστική δομή για την παραγωγή του μοντέλου και ακολούθως εφαρμογή του μοντέλου από τις συσκευές αυτές στο περιβόλι για την εξαγωγή των συμπερασμάτων.

Το όχημα θα κινείται εντός των ορίων του περιβολιού και η συσκευή θα λαμβάνει εικόνες οι οποίες θα αποτελέσουν τα δεδομένα από τα οποία θα προκύψει το μοντέλο. Ακολούθως, οι εικόνες αυτές αποστέλλονται σε μια ισχυρή υπολογιστική δομή στο νέφος για την εκπαίδευση και παραγωγή του μοντέλου. Για την δοκιμή του μοντέλου χρησιμοποιούνται εικόνες από την Google. Ακολούθως το μοντέλο αποστέλλεται πίσω στη συσκευή στο όχημα η οποία πρέπει από

τις νέες εικόνες που λαμβάνει να εξάγει απευθείας συμπεράσματα για την ποσότητα των ώριμων φρούτων που βρίσκονται πάνω στα δέντρα. Συνεπώς η συσκευή θα πρέπει να μπορεί να ξεχωρίσει τους καρπούς ανεξάρτητα από τις συνθήκες φωτισμού ενώ δεν θα πρέπει να παραλείπει αυτούς που δεν είναι ευδιάκριτοι λόγω του ότι μπορεί να βρίσκονται πίσω από κλαδιά, φύλλα ή άλλους καρπούς. Επιπλέον θα πρέπει να είναι σε θέση να επεξεργάζεται τις εικόνες με έναν ικανοποιητικό ρυθμό ώστε η όλη διαδικασία να μπορεί να ολοκληρωθεί σε ένα λογικό χρονικό διάστημα δίχως να εξαντλεί τα αποθέματα ηλεκτρικής ενέργειας που θα διαθέτει το όχημα.

Για την αντιμετώπιση των προκλήσεων αυτών οι ερευνητές μελέτησαν την απόδοση του, βελτιωμένου από τους ίδιους, αλγόριθμου YOLOv3-tiny για τον εντοπισμό των μήλων σε τέσσερις συσκευές IoT και έδειξαν ότι μπορούν να πετύχουν ρυθμό επεξεργασίας εικόνων της τάξης των τριάντα εικόνων ανά δευτερόλεπτο. Ποιο συγκεκριμένα ο YOLO [J. Redmon et all, 2016] είναι ένας αλγόριθμος που χαρακτηρίζεται τόσο από ταχεία όσο και ακριβή εντοπισμό αντικειμένων ενώ παράλληλα το σχετικά μικρό του μέγεθος τον καθιστά ιδανικό για χρήση σε ενσωματωμένα συστήματα. Επιπλέον, με την πάροδο των ετών νέες εκδόσεις αναπτύχθηκαν, μεταξύ αυτών και η tiny η οποία είναι η πλέον κατάλληλη για χρήση σε συσκευές IoT. Στη βιβλιοθήκη αυτή εφάρμοσαν ένα διαφορετικό τρόπο καθορισμού των ορίων του τετραγώνου που ορίζει το αντικείμενο που ενδιαφέρει με αποτέλεσμα να επιτύχουν μια αξιοσημείωτη βελτίωση στον αριθμό των αντικείμενων που εντοπίζει.

Πέρα από το λογισμικό, για να καταφέρουν έναν ικανοποιητικό ρυθμό επεξεργασίας εικόνων οι ερευνητές ενίσχυσαν το Raspberry Pi 3 B+ με συσκευές τύπου Neural Compute Sticks, Οι συσκευές αυτές έχουν την μορφή των usb sticks και επιτελούν ρόλο επιταχυντή υλικού (hardware accelerators) με εξειδίκευση στις εφαρμογές τεχνικής νοημοσύνης. Οι συσκευές που χρησιμοποίησαν είναι οι Myriad 2 VPU και η Myriad X VPU. Με την πρώτη το Raspberry κατάφερε να επεξεργάζεται τέσσερις εικόνες ανά δευτερόλεπτο καταναλώνοντας έξι Watt ενώ με το δεύτερο πέντε εικόνες ανά δευτερόλεπτο καταναλώνοντας 5.6 Watt. Πέρα από το Raspberry οι ερευνητές πραγματοποίησαν μετρήσεις και σε άλλες δυο συσκευές την εταιρίας Nvidia, την Jetson AGX Xavier και την Jetson Nano.



Εικόνα 11: Οι συσκευές IOT που χρησιμοποίησαν οι ερευνητές. Πάνω αριστερά διακρίνεται το Raspberry Pi 3 B+ με το Myriad 2 VPU, ενώ κάτω δεξιά το αντίστοιχο με το Myriad X VPU. Αντίστοιχα, πάνω δεξιά το Jetson AGX Xavier ενώ κάτω αριστερά το NANO. Πηγή: Real-Time

Συμπερασματικά οι ερευνητές καταλήγουν ότι είναι εφικτή η δημιουργία ενός συστήματος εκτίμησης εικόνων πραγματικού χρόνου και από συσκευές τύπου Raspberry Pi 3 B+ ο ρυθμός επεξεργασίας όμως αποτελεί ένα παράγοντα που θα πρέπει να ληφθεί υπόψιν. Στη περίπτωση που απαιτείται ρυθμός που δεν θα υπερβαίνει τις πέντε εικόνες ανά δευτερόλεπτο τότε η συσκευή θα πρέπει να ενισχυθεί με τις USB συσκευές που αναφέρθηκαν. Αν όμως η εφαρμογή απαιτεί υψηλότερο ρυθμό τότε θα πρέπει να επιλεχθεί κάποια από τις άλλες προτάσεις που κυκλοφορούν.

Κεφάλαιο 4: Εφαρμογή κατανεμημένης μάθησης

4.1. Εισαγωγή

Στο κεφάλαιο αυτό θα υλοποιηθεί μια εφαρμογή κατανεμημένης μάθησης σε γλώσσα προγραμματισμού Python και με τη βοήθεια διαφόρων βιβλιοθηκών. Η εφαρμογή θα διαχωριστεί σε δύο μέρη όπου το πρώτο περιλαμβάνει την παραγωγή του μοντέλου το οποίο θα γίνει σε ένα τυπικό υπολογιστή γραφείου και ακολούθως το μοντέλο θα αποσταλεί στη συσκευή IoT το ρόλο της οποίας θα εκτελεί ένα Raspberry Pi 3 B+. Η συσκευή IoT ακολούθως θα επεξεργάζεται τα νέα δεδομένα και θα παρουσιάζει τα αποτελέσματα.

4.2. Το σύνολο των δεδομένων

Στόχος της εφαρμογής είναι ο χαρακτηρισμός – ταξινόμηση εικόνων που παρουσιάζουν χειρόγραφους χαρακτήρες αριθμών από το 0 έως και το 9. Για την εκπαίδευση του μοντέλου θα χρησιμοποιηθούν τα δεδομένα αριθμητικών χειρόγραφων χαρακτήρων από τον ιστότοπο MNIST που παρουσιάστηκαν στη παράγραφο 3.1 Τα δεδομένα αυτά είναι διαθέσιμα από τον ιστότοπο http://yann.lecun.com/exdb/mnist/

THE MNIST DATABASE

of handwritten digits

Yann LeCun, Courant Institute, NYU
Corinna Cortes, Google Labs, New York
Christopher J.C. Burges, Microsoft Research, Redmond

Please refrain from accessing these files from automated scripts with high frequency. Make copies!`

The MNIST database of handwritten digits, available from this page, has a training set of 60,000 examples, and a test set of 10,000 examples. It is a subset of a larger set available from NIST. The digits have been size-normalized and centered in a fixed-size image.

It is a good database for people who want to try learning techniques and pattern recognition methods on real-world data while spending minimal efforts on preprocessing and formatting.

Four files are available on this site:

```
train-images-idx3-ubyte.gz:  training set images (9912422 bytes)
train-labels-idx1-ubyte.gz:  training set labels (28881 bytes)
t10k-images-idx3-ubyte.gz:   test set images (1648877 bytes)
t10k-labels-idx1-ubyte.gz:   test set labels (4542 bytes)
```

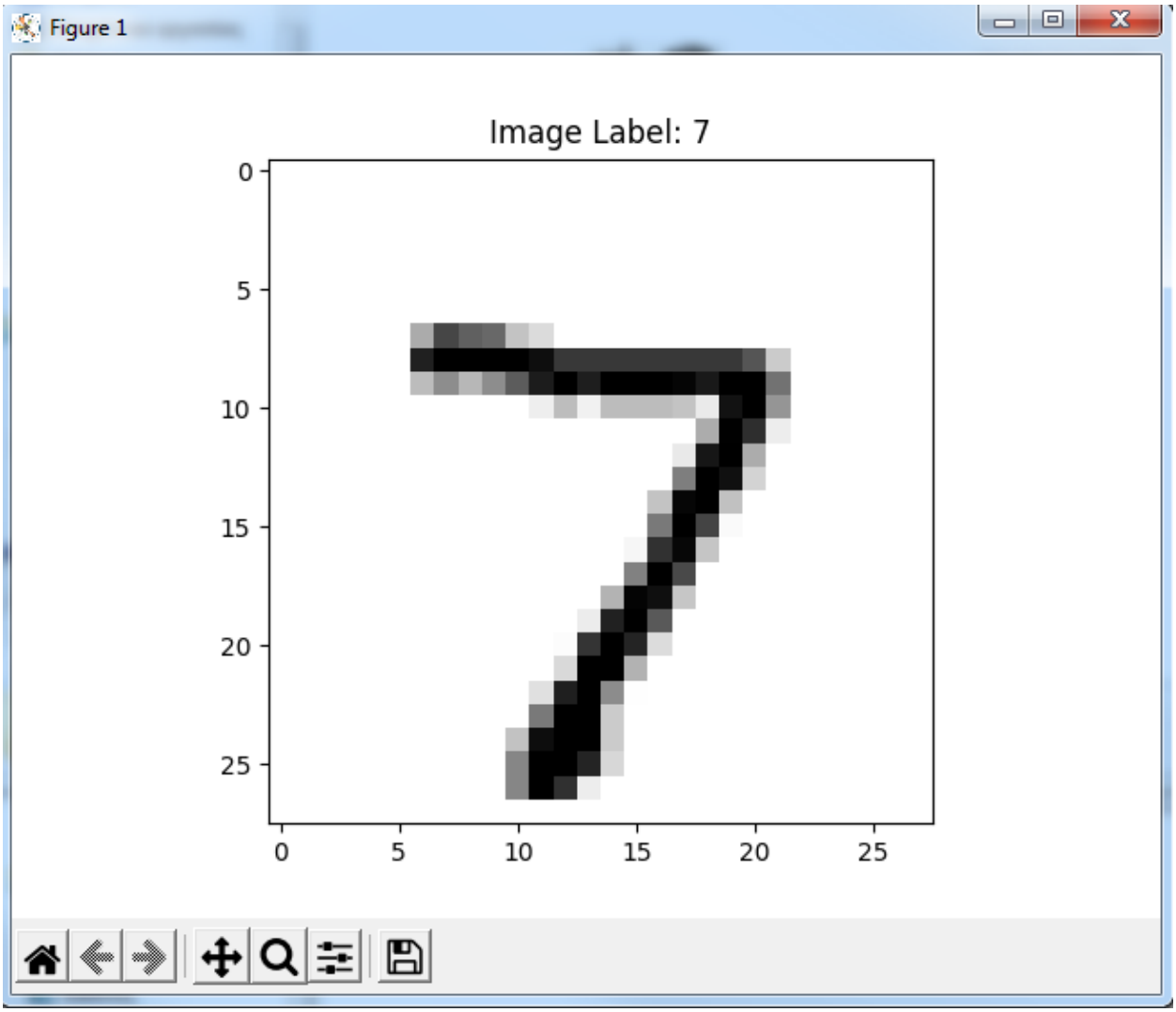
Εικόνα 12: Τα δεδομένα που θα χρησιμοποιηθούν για να εκπαιδεύσουν το μοντέλο. Πηγή: http://yann.lecun.com/exdb/mnist/

Τα δεδομένα αποτελούντα από τέσσερα συνολικά αρχεία τα οποία έρχονται συμπιεσμένα σε μορφή gz στο υπολογιστή του χρήστη. Πιο συγκεκριμένα το αρχείο train-images.idx3-ubyte περιέχει τις εικόνες προς εκπαίδευση ενώ το train-labels.idx1-ubyte περιέχει τα ονόματα των εικόνων. Όμοια το t10k-images.idx3-ubyte περιέχει 10.000 εικόνες για τη δοκιμή του μοντέλου και το t10k-labels.idx1-ubyte περιέχει τα ονόματα των εικόνων αυτών. Εκτυπώνοντας την πρώτη εικόνα σε ένα αρχείο ο χρήστης θα παρατηρήσει ότι αποτελείται από ένα πίνακα διαστάσεων 28 επί 28 που περιέχει τιμές από 0 έως το 255.

```
[[  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0  84 185 159 151  60  36   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0 222 254 254 254 254 241 198 198 198 198 198 198 198 198 170  52   0   0   0   0   0   0]
 [  0   0   0   0   0   0  67 114  72 114 163 227 254 225 254 254 254 250 229 254 254 140   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0  17  66  14  67  67  67  59  21 236 254 106   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  83 253 209  18   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  22 233 255  83   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0 129 254 238  44   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  59 249 254  62   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0 133 254 187   5   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   9 205 248  58   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0 126 254 182   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0  75 251 240  57   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0  19 221 254 166   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   3 203 254 219  35   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0  38 254 254  77   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0  31 224 254 115   1   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0 133 254 254  52   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0  61 242 254 254  52   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0 121 254 254 219  40   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0 121 254 207  18   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]]
```

Εικόνα 13: Η πρώτη εικόνα του δείγματος από το αρχείο train-images.idx3-ubyte. Πηγή : Συγγραφέας

Με τη βοήθεια της βιβλιοθήκης matplotlib μπορεί να εκτυπωθεί η εικόνα μαζί με την αντίστοιχη τιμή του αρχείου train-labels.idx1-ubyte στον τίτλο.



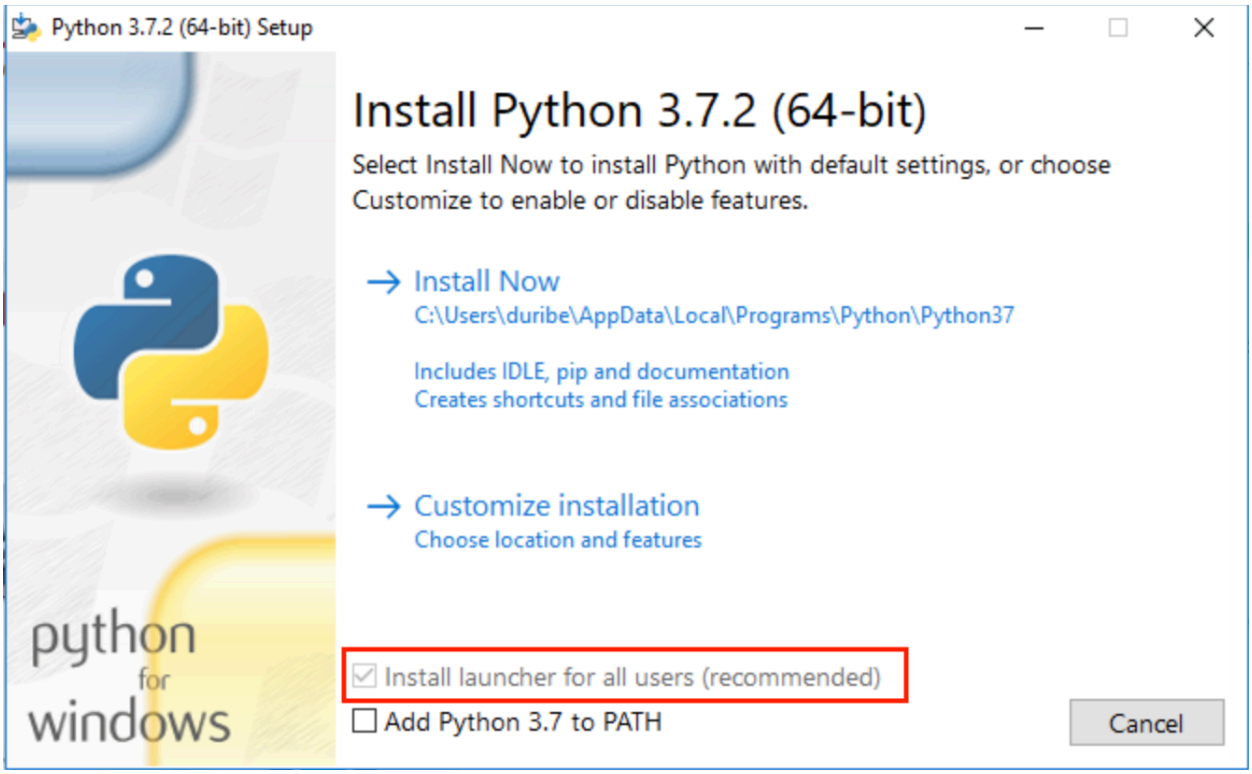
Εικόνα 14: Η εικόνα μαζί με τον τίτλο της. Πηγή: Συγγραφέας

4.3. Παραγωγή του μοντέλου

4.3.1 Διαμόρφωση του περιβάλλοντος

Για την εκπαίδευση του μοντέλου θα χρησιμοποιηθεί ένας υπολογιστής εφοδιασμένος με επεξεργαστή Intel Core i3 στα 2.93Gz, λειτουργικό σύστημα Windows 7 αρχιτεκτονικής 64bit και 16GB μνήμη. Η έκδοση της python είναι η 3.7 και για την εγκατάσταση της ο χρήστης λαμβάνει το εκτελέσιμο αρχείο από τον επίσημο ισότοπο της εταιρίας https://www.python.org/

downloads/release/python-374/. Μετά τη λήψη του αρχείου ο χρήστης το εγκαθιστά στον υπολογιστή του προσθέτοντας την python στη PATH του υπολογιστή του ώστε να είναι προσβάσιμη από οπουδήποτε.



Εικόνα 15: Εγκατάσταση Python. Πηγή: https://i.stack.imgur.com/5JCIE.png

Το πρόγραμμα εκτελείται σε ένα εικονικό περιβάλλον virtualenv ώστε να είναι απομονωμένο από τα υπόλοιπα προγράμματα του υπολογιστή. Για τη δημιουργία του περιβάλλοντος αρκεί η εντολή μέσω του παραθύρου εντολών Command Prompt:

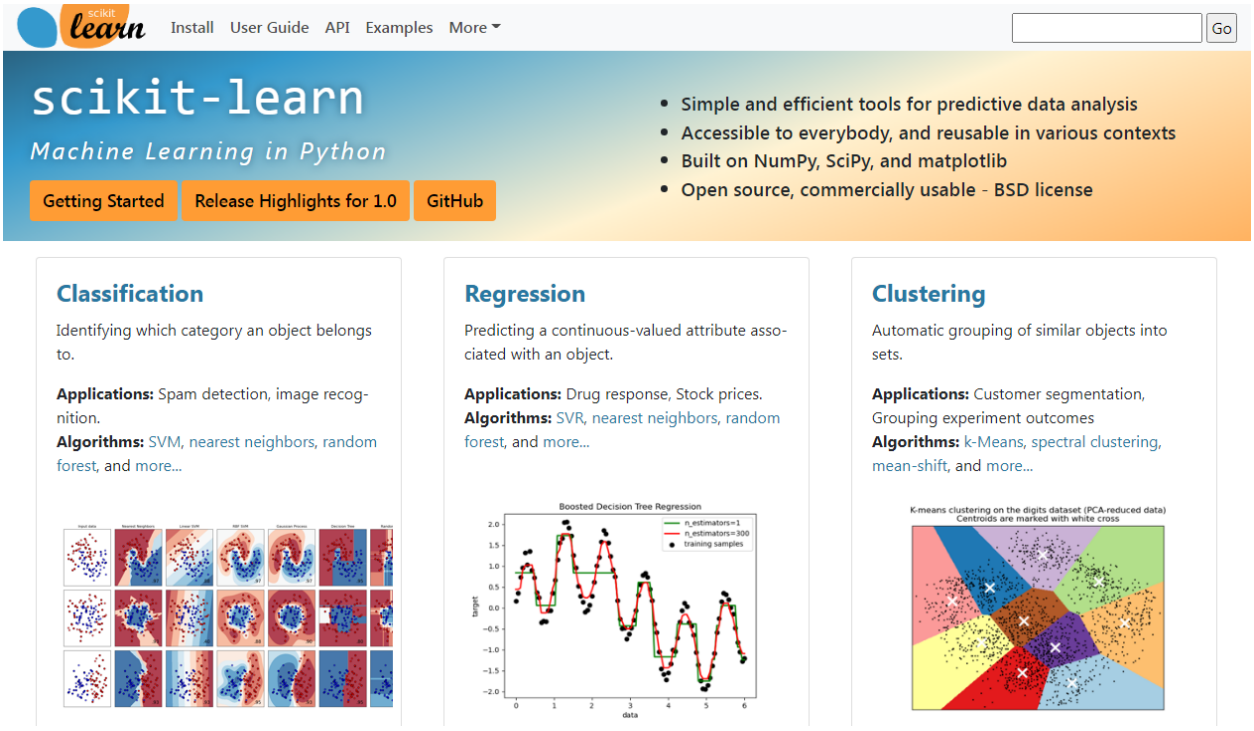
```
python -m virtualenv onomaFakeloy
```

η οποία θα δημιουργήσει ένα νέο φάκελο με όνομα onomaFakeloy. Μέσα στο φάκελο αυτό υπάρχουν διάφορα αρχεία μεταξύ αυτών και το activate το οποίο θα ενεργοποιήσει το εικονικό περιβάλλον. Η ενεργοποίηση επιτυγχάνεται με την εκτέλεση του αρχείου activate που βρίσκεται στο φάκελο Scripts:

onomaFakeloy\Scripts\activate

Το εικονικό περιβάλλον είναι πλέον έτοιμο και ο χρήστης θα πρέπει να εγκαταστήσει τις απαραίτητες βιβλιοθήκες. Αρχικά η scikit-learn, μία από τις πλέον καταξιωμένες βιβλιοθήκες μηχανικής μάθησης ανοιχτού κώδικά, από την οποία θα χρησιμοποιηθεί ο αλγόριθμος Support Vector Machine για την παραγωγή του μοντέλου. Αυτός είναι ο πλέον κατάλληλος για την περίπτωση αφού αποδίδει καλύτερα σε πολυδιάστατα δεδομένα, σύμφωνα και με τη σύγκριση των αλγορίθμων της παραγράφου 3.2. Πέρα από τον SVM θα χρησιμοποιηθεί και η συνάρτηση preprocessing για την κανονικοποίηση των δεδομένων αλλά και η metrics για τον έλεγχο. Η εγκατάσταση της θα γίνει μέσω του συστήματος διαχείρισης πακέτων PIP μέσω της εντολής:

```
pip install scikit-image
```

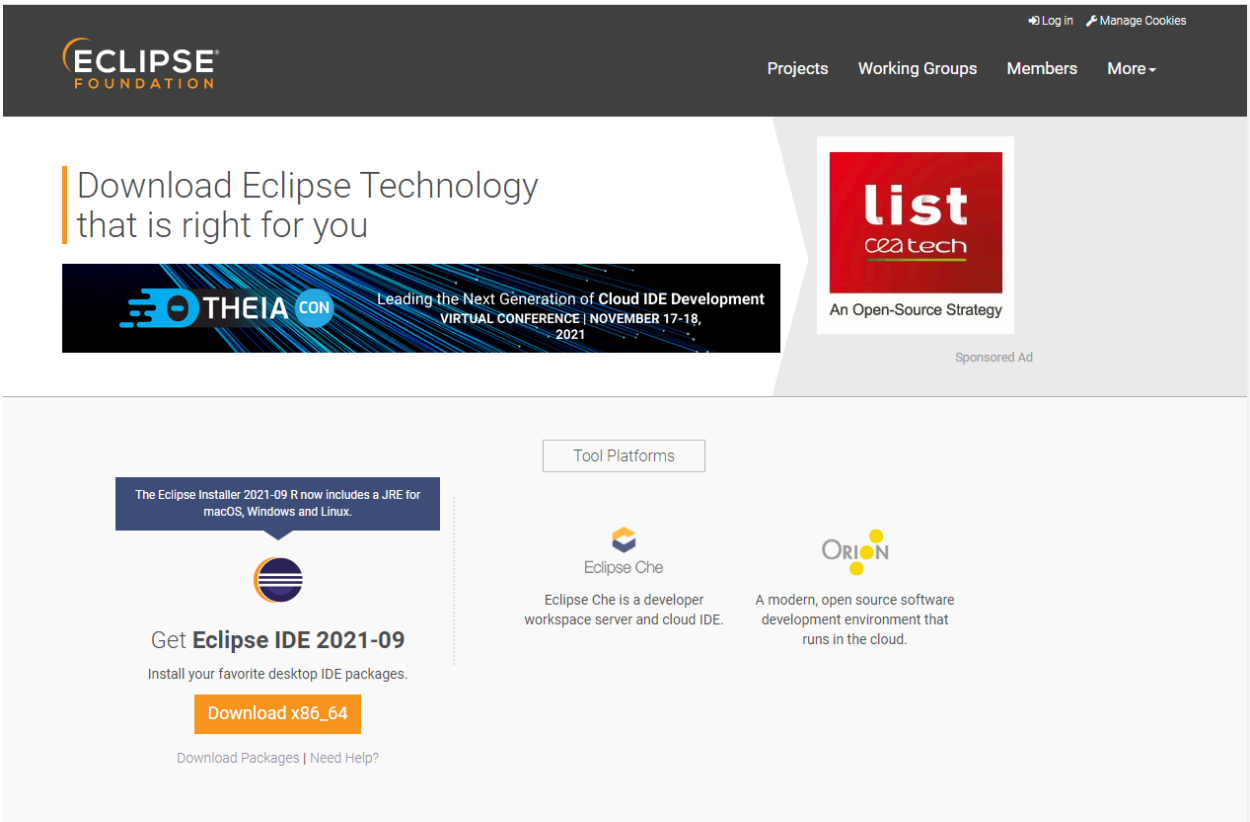


Εικόνα 16: Η βιβλιοθήκη scikit – learn. Πηγή: https://scikit-learn.org/stable/

Πέρα από την scikit-learn θα χρησιμοποιηθεί η idx2numpy για ανάγνωση των αρχείων των εικόνων του MNIST και η Numpy, αφού τα αρχεία θα φορτωθούν σε πίνακες αυτού του τύπου. Τέλος, για την αποθήκευση του μοντέλου ως αντικειμένου Python σε αρχείο ώστε να μπορεί να μεταφερθεί για εκτέλεση θα χρησιμοποιηθεί η Pickle. Συνεπώς για την εισαγωγή των βιβλιοθηκών που αναφέρθηκαν θα εισαχθούν οι εντολές :

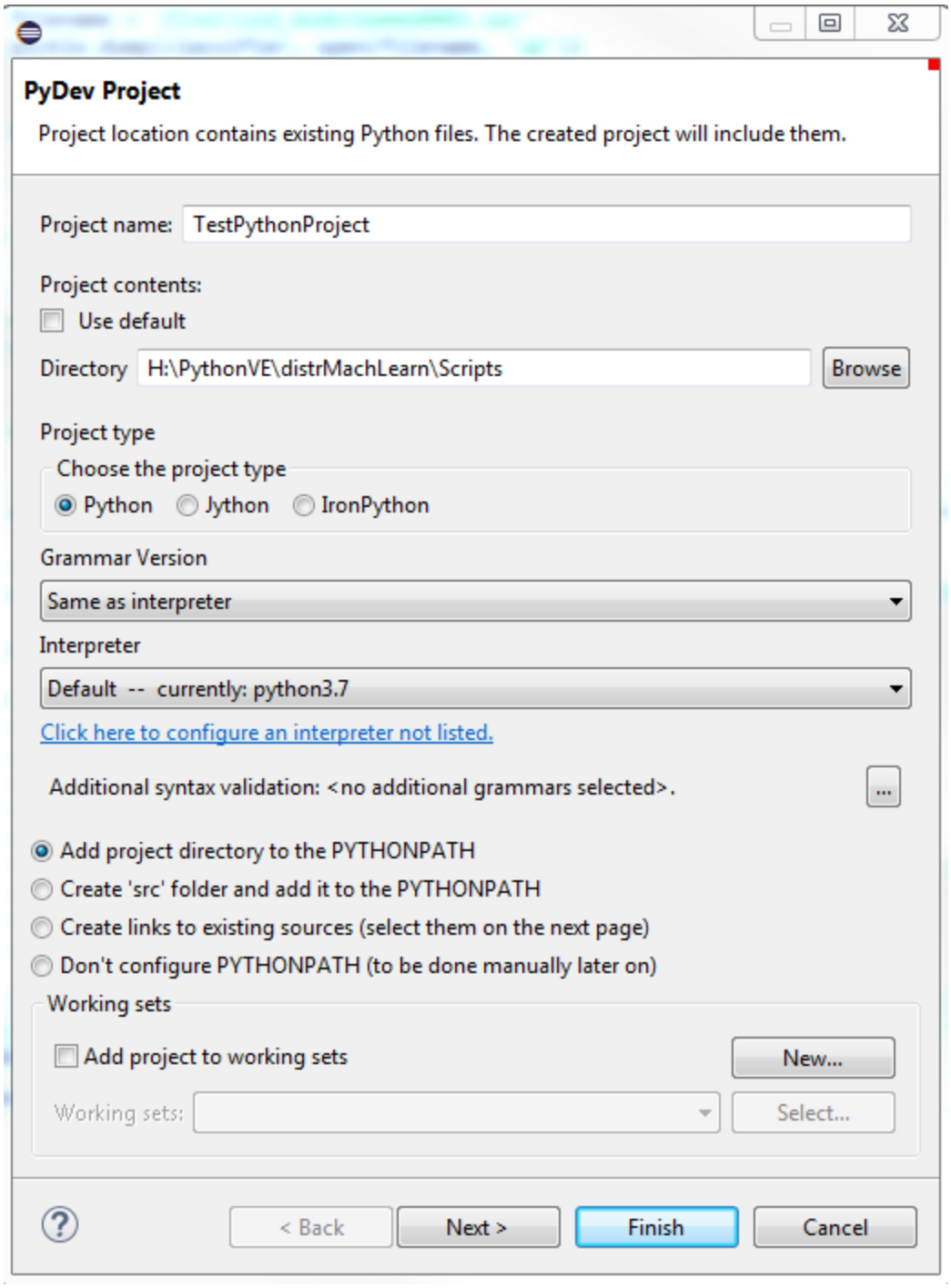
```
pip install idx2numpy
pip install numpy
pip install pickle-mixin
```

Επόμενο βήμα το ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) ώστε να αρχίσει τη συγγραφή του κώδικά. Στη παρούσα παρουσίαση θα χρησιμοποιηθεί το eclipse το οποίο είναι δωρεάν και λαμβάνεται από τον ιστότοπο https://www.eclipse.org/downloads/.



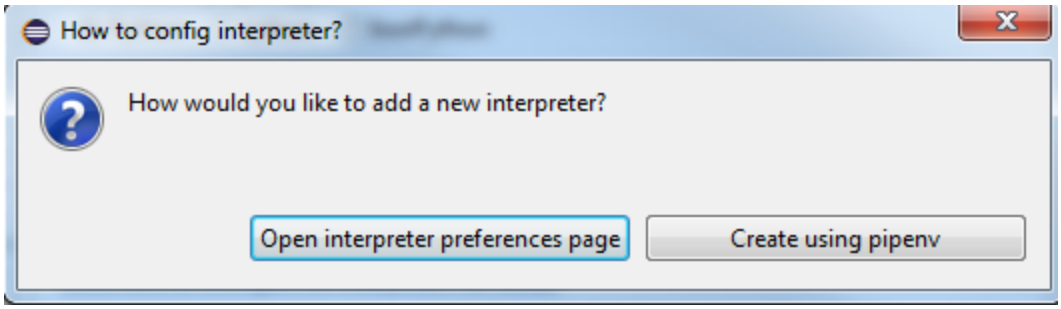
Εικόνα 17: Το ολοκληρωμένο περιβάλλον ανάπτυξης λογισμικού Eclipse. Πηγή: https://www.eclipse.org/downloads/

Μετά την εγκατάσταση του ο χρήστης θα πρέπει να εισάγει τον μεταφραστή που δημιούργησε στο εικονικό περιβάλλον στο eclipse. Για να το πετύχει αυτό θα πρέπει να επιλέξει να δημιουργήσει ένα νέο Python Project και αφού δώσει το κατάλληλο όνομα να επιλέξει τη δημιουργία ενός νέου μεταφραστή (Click Here to configure an interpreter not listed).



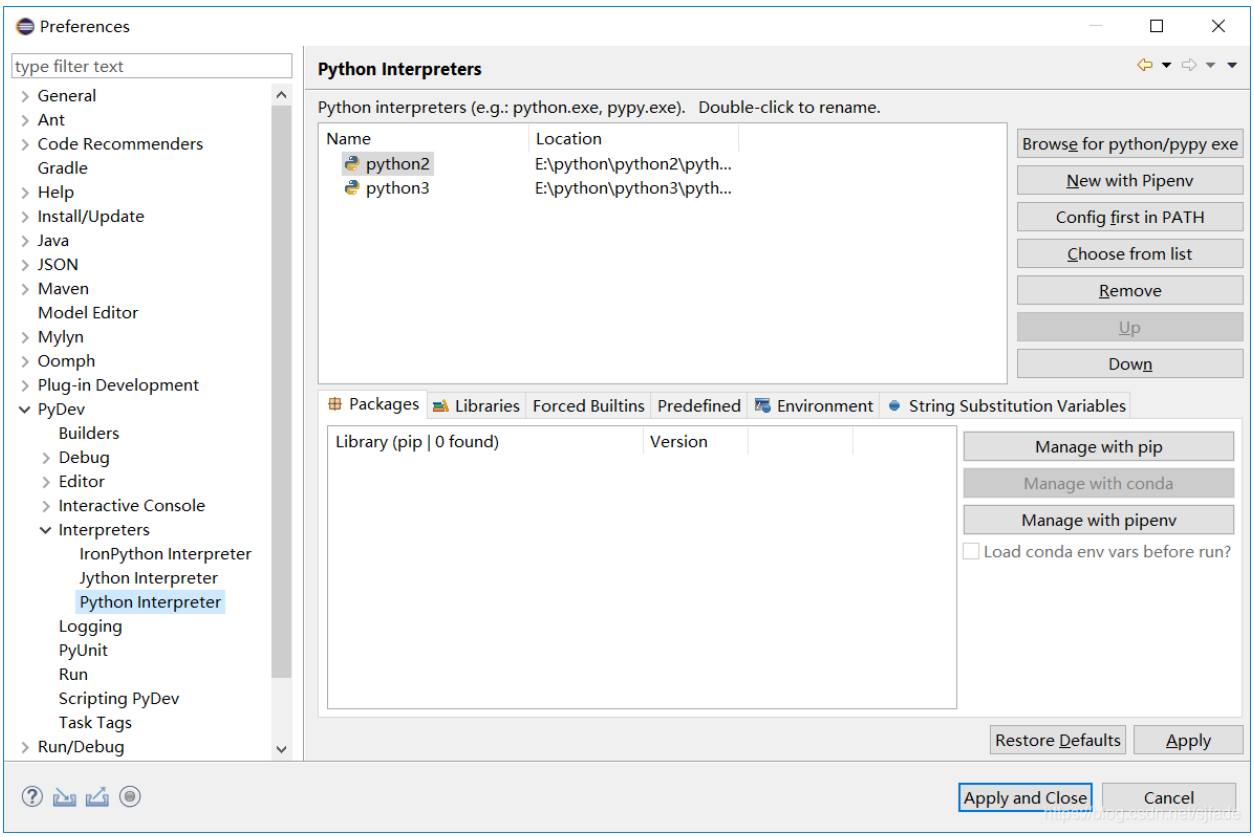
Εικόνα 18: Παραμετροποίηση Eclipse IDE. Πηγή: Συγγραφέας

Ακολούθως το πρόγραμμα του ζητά να επιλέξει μεταφραστή ανοίγοντας της σελίδα των μεταφραστών αφού έχει ήδη δημιουργήσει ένα στο εικονικό περιβάλλον.



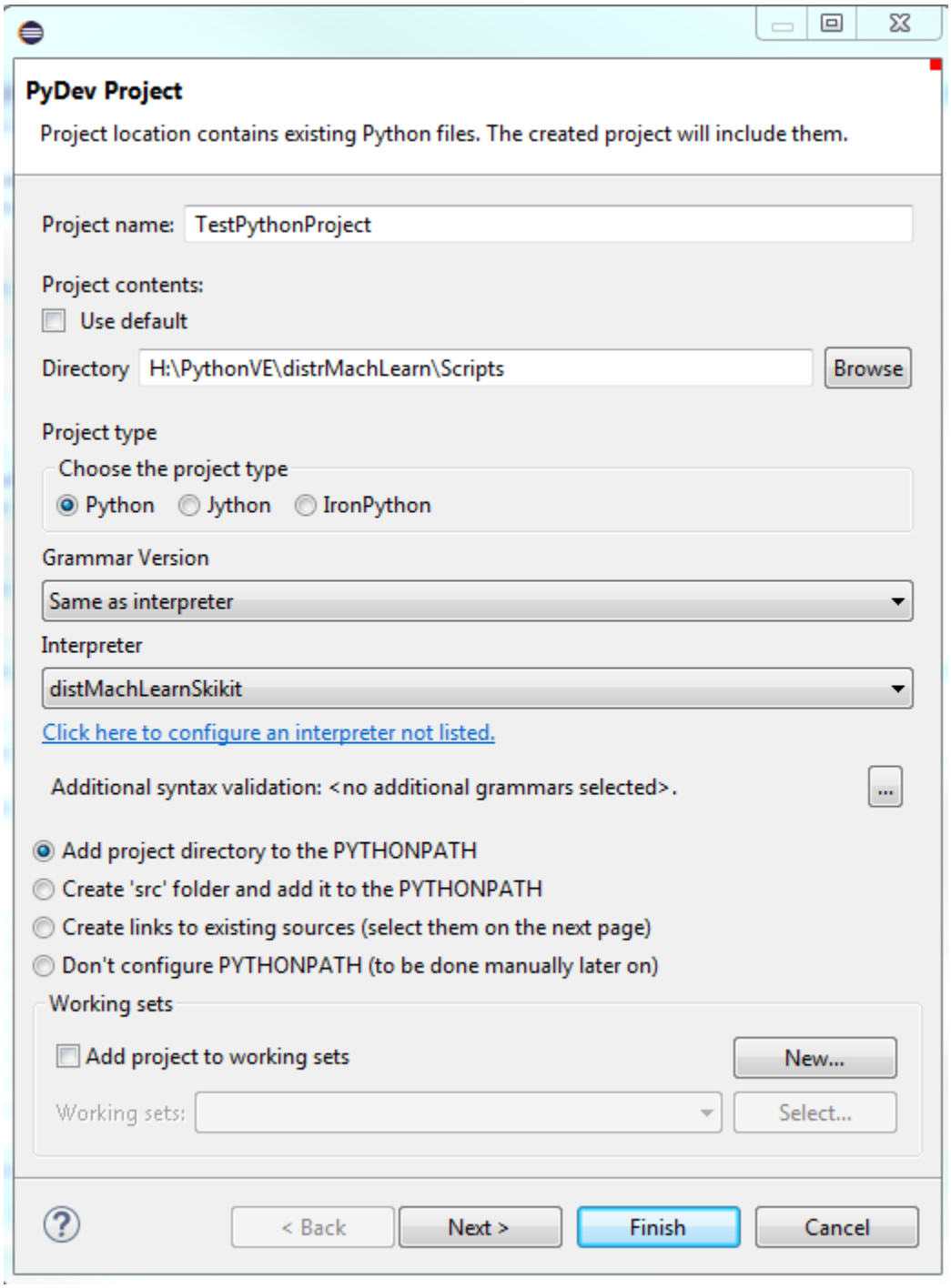
Εικόνα 19: Επιλογή της σελίδας των μεταφραστών. Πηγή: Συγγραφέας

Από το παράθυρο θα επιλέξει το κουμπί Browse for python/pypy.exe και θα επιλέξει το εκτελέσιμο python.exe από το φάκελο Scripts του εικονικού περιβάλλοντος που έχει δημιουργήσει στο προηγούμενο βήμα.



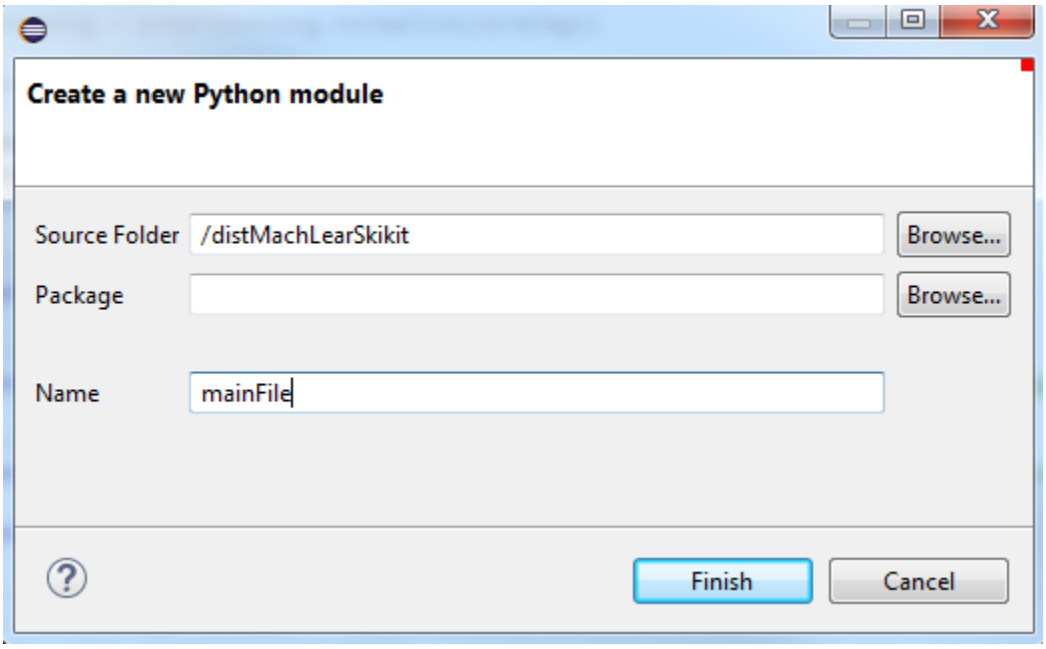
Εικόνα 20: Επιλογή του μεταφραστή. Πηγή: Συγγραφέας

Στο επόμενο παράθυρο ορίζει το όνομα του μεταφραστή και τον επιλέγει για το νέο του project. Η τελική εικόνα των ιδιοτήτων του project θα πρέπει να έχει τη μορφή της ακόλουθης εικόνας με τον Interpreter που δημιουργήθηκε επιλεγμένο.



Εικόνα 21: Ολοκλήρωση παραμετροποίησης του IDE. Πηγή: Συγγραφέας

Ακολούθως επιλέγεται Finish και η διαδικασία έχει ολοκληρωθεί. Ο χρήστης μπορεί πλέον να επιλέξει το νέο project που δημιούργησε και να αρχίσει να δημιουργεί τα επιμέρους αρχεία επιλέγοντας New PyDev Model.



Εικόνα 22: Δημιουργία νέου αρχείου. Πηγή: Συγγραφέας

4.3.2 Συγγραφή του προγράμματος

Πρώτο βήμα η κωδικοποίηση του αρχείου και ακολούθως η εισαγωγή των απαραίτητων βιβλιοθηκών. Παράλληλα με αυτές που προαναφέρθηκαν εισάγεται και η βιβλιοθήκη συστήματος time για τη μέτρηση του χρόνου εκτέλεσης

```
1  # coding=utf8
2
3  import idx2numpy
4  from sklearn import svm
5  from sklearn import preprocessing
6  from sklearn import metrics
7  import pickle
8  import time
9
```

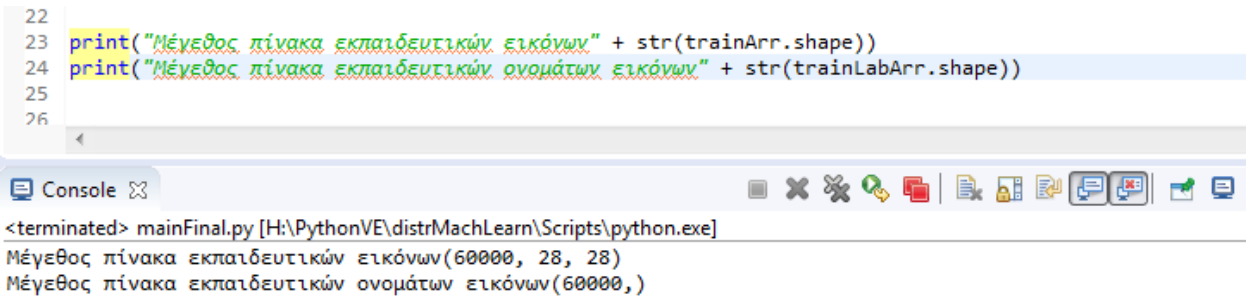
Εικόνα 23: Η εισαγωγή των απαραίτητων βιβλιοθηκών. Πηγή: Συγγραφέας

Επόμενο βήμα το διάβασμα των αρχείων του MNIST. Τα αρχεία αποσυμπιέζονται και τοποθετούνται σε ένα φάκελο με όνομα Mnist μέσα στο φάκελο του Project του Eclipse. Η εισαγωγή τους σε πίνακες τύπου Numpy γίνεται με τη χρήση της συνάρτησης convert_from_file της idx2numpy βιβλιοθήκης.

```
11 trainFile = 'Mnist/train-images.idx3-ubyte'
12 trainArr = idx2numpy.convert_from_file(trainFile)
13
14 trainLabFile = 'Mnist/train-labels.idx1-ubyte'
15 trainLabArr = idx2numpy.convert_from_file(trainLabFile)
16
17 testFile = 'Mnist/t10k-images.idx3-ubyte'
18 testArr = idx2numpy.convert_from_file(testFile)
19
20 testLabFile = 'Mnist/t10k-labels.idx1-ubyte'
21 testLabArr = idx2numpy.convert_from_file(testLabFile)
```

Εικόνα 24: Τα δεδομένα φορτώνονται σε πίνακες τύπου Numpy. Πηγή: Συγγραφέας

Πλέον οι εικόνες προς εκπαίδευση είναι στο πίνακα trainArr ενώ τα ονόματα των εικόνων αυτών είναι στο πίνακα trainLabArr. Αντίστοιχα και οι εικόνες ελέγχου είναι στο πίνακα testArr ενώ τα ονόματα στον testLabArr. Με τη βοήθεια της συνάρτησης shape του αντικειμένου Numpy μπορούμε να δούμε το μέγεθος του κάθε πίνακα.



Εικόνα 25: Επισκόπηση των δεδομένων. Ο πίνακας trainArr περιέχει 60 χιλιάδες εικόνας μεγέθους 28 επι 28 ενώ ο πίνακας trainLabArr περιέχει 60 χιλιάδες ονόματα. Πηγή: Συγγραφέας

Εκτυπώνοντας το τελευταίο στοιχείο του πίνακα trainArr ο χρήστης θα πάρει το αποτέλεσμα της επόμενης εικόνας.

```
[[   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  38  48  48  22   0   0   0   0   0   0   0   0]
 [   0   0   0   0   0   0   0   0   0   0   0   0   0  62  97 198 243 254 254 212  27   0   0   0   0   0   0   0]
 [   0   0   0   0   0   0   0   0   0   0   0   0  67 172 254 254 225 218 218 237 248  40   0  21 164 187   0   0]
 [   0   0   0   0   0   0   0   0   0   0   0  89 219 254  97  67  14   0   0  92 231 122  23 203 236  59   0   0]
 [   0   0   0   0   0   0   0   0   0   0  25 217 242  92   4   0   0   0   0   4 147 253 240 232  92   0   0   0]
 [   0   0   0   0   0   0   0   0   0   0 101 255  92   0   0   0   0   0   0 105 254 254 177  11   0   0   0   0]
 [   0   0   0   0   0   0   0   0   0   0 167 244  41   0   0   0   7  76 199 238 239  94  10   0   0   0   0   0]
 [   0   0   0   0   0   0   0   0   0   0 192 121   0   0   2  63 180 254 233 126   0   0   0   0   0   0   0   0]
 [   0   0   0   0   0   0   0   0   0   0 190 196  14   2  97 254 252 146  52   0   0   0   0   0   0   0   0   0]
 [   0   0   0   0   0   0   0   0   0   0 130 225  71 180 232 181  60   0   0   0   0   0   0   0   0   0   0   0]
 [   0   0   0   0   0   0   0   0   0   0 130 254 254 230  46   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [   0   0   0   0   0   0   0   0   6  77 244 254 162   4   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [   0   0   0   0   0   0   0   0 110 254 218 254 116   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [   0   0   0   0   0   0   0 131 254 154  28 213  86   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [   0   0   0   0   0   0  66 209 153  19  19 233  60   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [   0   0   0   0   0   0 142 254 165   0  14 216 167   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [   0   0   0   0   0   0  90 254 175   0  18 229  92   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [   0   0   0   0   0   0  26 229 249 176 222 244  44   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [   0   0   0   0   0   0   0  73 193 197 134   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]]
```

Εικόνα 26: Τα περιεχόμενα του πίνακα στη θέση 59999. Πηγή: Συγγραφέας

Ενώ ο πίνακας trainLabArr στην ίδια θέση έχει τη τιμή 8.

```
26 print("Περιεχόμενο του πίνακα τελευταία θέση: "+ str(trainLabArr[59999]))
27
```

Console

```
<terminated> mainFinal.py [H:\PythonVE\distrMachLearn\Scripts\python.exe]
Περιεχόμενο του πίνακα τελευταία θέση: 8
```

Εικόνα 27: Το περιεχόμενο του πίνακα trainLabArr στη τελευταία θέση. Πηγή: Συγγραφέας

Στη συνέχεια θα πρέπει ο πίνακας των δεδομένων να μετατραπεί σε μονής διάστασης καθώς το μοντέλο προϋποθέτει τα δεδομένα να είναι σε αυτή τη μορφή. Κατά συνέπεια ο πίνακας θα μετατραπεί σε ένα πίνακα που θα περιέχει την κάθε εικόνα σε μία θέση. Οπότε θα πρέπει αρχικά να ανακτηθεί η πρώτη διάσταση του πίνακα που αντιστοιχεί στο σύνολο των εικόνων και ακολούθως να χρησιμοποιηθεί η συνάρτηση reshape του αντικειμένου numpy. Στη συνάρτηση αυτή θα πρέπει να δοθεί το μέγεθος των διαστάσεων που θα μετατραπεί ο πίνακας οπότε δίνεται σαν πρώτο όρισμα ο αριθμός των εικόνων. Σαν δεύτερο δίνεται η τιμή -1 ώστε η numpy να υπολογίσει αυτόματα την τιμή η οποία θα είναι 28 επί 28. Η όλη διαδικασία καταλήγει να παράξει ένα πίνακα διαστάσεων 60.000 επί 784.

```
29 n_samples = len(trainArr)
30 flatImgArr = trainArr.reshape((n_samples, -1))
31
32 print("Μέγεθος πίνακα εκπαιδευτικών εικόνων" + str(flatImgArr.shape))
33
34
```

Console

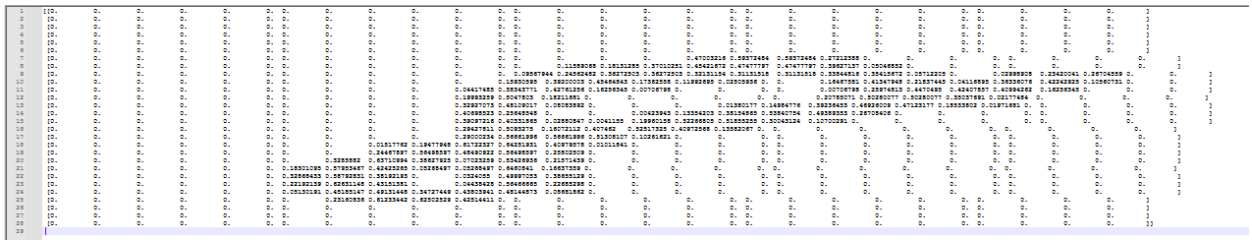
```
<terminated> mainFinal.py [H:\PythonVE\distrMachLearn\Scripts\python.exe]
Μέγεθος πίνακα εκπαιδευτικών εικόνων(60000, 784)
```

Εικόνα 28: Μετατροπή διαστάσεων πίνακα. Πηγή: Συγγραφέας

Επόμενο βήμα η κανονικοποίηση των δεδομένων. Τα δεδομένα, όπως παρουσιάστηκε σε προηγούμενη παράγραφο, αποτελούνται από θετικές ακέραιες τιμές από το 0 έως το 255. Προκειμένου να έχουν μια πιο ενιαία μορφή, η οποία θα βελτιώσει την απόδοση του μοντέλου, θα χρησιμοποιηθεί η συνάρτηση normalize του αντικειμένου preprocessing.

```
36
37
38 normImg = preprocessing.normalize(trainArr[59999])
39
40 print(normImg)
41
42
```

Εικόνα 29: Κανονικοποίηση του τελευταίου στοιχείου του πίνακα. Πηγή: Συγγραφέας



Εικόνα 30: Το αποτέλεσμα της κανονικοποίησης του τελευταίου στοιχείου του πίνακα που αναπαριστά το 8. Οι τιμές άνω του 0 αναπαρίστανται ως δεκαδικοί με τιμή μικρότερη της μονάδας. Πηγή: Συγγραφέας

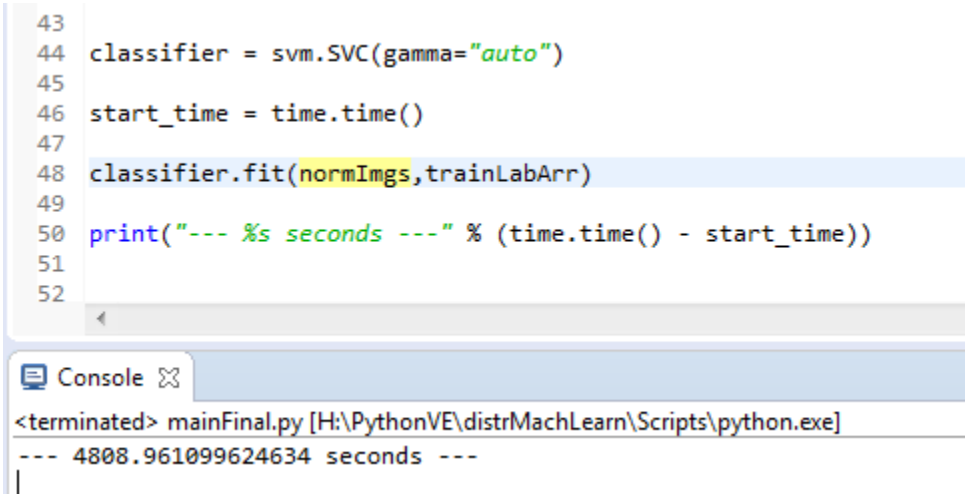
Επόμενο βήμα η δημιουργία του αντικειμένου SVM δίνοντας τις κατάλληλες παραμέτρους στον constructor. Ο constructor δέχεται πληθώρα παραμέτρων οι οποίες προκύπτουν κατόπιν επαναλαμβανόμενων δοκιμών ή και μέσα από αυτόματες διαδικασίες και καθορίζουν την αποτελεσματικότητα του μοντέλου. Στα πλαίσια της παρούσας παρουσίασης θα εφαρμοστούν οι βασικές ρυθμίσεις και πιο συγκεκριμένα θα χρησιμοποιηθεί ο πυρήνας RBF (Radial Basis Function) με τιμή gamma ίση με “auto”.

```
43
44 classifier = svm.SVC(gamma="auto")
45
46
```

Εικόνα 31: Αρχικοποίηση του αντικειμένου SVM. Πηγή: Συγγραφέας

Στη συνέχεια θα πραγματοποιηθεί η εκπαίδευση του μοντέλου. Η διαδικασία πραγματοποιείται μέσω της συνάρτησης fit του αντικειμένου classifier που δημιουργήθηκε στο προηγούμενο βήμα. Σαν όρισμα παίρνει τον πίνακα με τις εικόνες και τον αντίστοιχο με τα ονόματα. Η πρώτη

διάσταση και των δύο πινάκων πρέπει να είναι προφανώς ίδια ώστε η κάθε εικόνα να μπορεί να αντιστοιχηθεί στο αντίστοιχο όνομα της. Προκειμένου να μετρηθεί ο χρόνος που απαιτείται για την ολοκλήρωση της διαδικασίας αποθηκεύεται η χρονική στιγμή έναρξης της εκπαίδευσης και ακολούθως μετά την ολοκλήρωση εκτυπώνεται η διαφορά της αρχικής τιμής και της τρέχουσας.



Εικόνα 32: Εκπαίδευση του μοντέλου. Πηγή: Συγγραφέας

Η άνω διαδικασία, στον υπολογιστή που περιγράφηκε στη προηγούμενη παράγραφο ολοκληρώθηκε σε χρονικό διάστημα των 4808 δευτερολέπτων. Χρονικό διάστημα για την εκπαίδευση ενός μοντέλου που θα ταξινομεί αρκετά μικρές εικόνες μεγέθους 28 pixel επί 28 όπου κάθε pixel έχει μία τιμή, δηλαδή το μοντέλο επεξεργάζεται 784 χαρακτηριστικά για κάθε ένα αντικείμενο που πρέπει να ταξινομήσει. Δεδομένου ότι ένα κινητό τηλέφωνο που διαθέτει κάμερα 12 MegaPixel έχει την δυνατότητα να παράγει εικόνες 4000 επί 3000 pixel όπου το κάθε pixel περιέχει μια τιμή RGB η οποία συνεπάγεται με τρείς επιπλέον διαστάσεις (μία τιμή για κάθε χρώμα RGB) καταλήγει σε ένα σύνολο 36000 χαρακτηριστικών ανά εικόνα. Καθίσταται προφανές στον αναγνώστη ο χρόνος και η υπολογιστική ισχύς που απαιτείται για την παραγωγή μοντέλων που να είναι σε θέση να επεξεργαστούν τις εικόνες που παράγονται στη πραγματικότητα από τις σύγχρονες φωτογραφικές μηχανές..

Το τελευταίο βήμα για την ολοκλήρωση της διαδικασίας είναι η αποθήκευση του μοντέλου σε ένα αρχείο το οποίο θα μεταφερθεί στη συνέχεια στη συσκευή IoT και πιο συγκεκριμένα στο Raspberry το οποίο θα το χρησιμοποιήσει προκειμένου να εκτιμήσει τις νέες εικόνες που θα λάβει ως είσοδο. Η διαδικασία επιτυγχάνεται με τη χρήση τη βιβλιοθήκης pickle η οποία θα αποθηκεύσει το μοντέλο ως αντικείμενο της Python σε ένα αρχείο.

```
46
47 filename = 'model.sav'
48 pickle.dump(classifier, open(filename, 'wb'))
49
```

Εικόνα 33: Αποθήκευση του μοντέλου. Πηγή: Συγγραφέας

Το αρχείο που παράγεται με την διαδικασία αυτή είναι αρκετά μεγάλο και ποιο συγκεκριμένα έχει έκταση που αγγίζει τα 352MB πράγμα που το καθιστά δύσκολο στη μεταφορά μέσω δικτύου και σχεδόν αδύνατο να το αξιοποιήσει το Raspberry Pi 3 B+ που θα χρησιμοποιηθεί στη παρούσα παρουσίαση. Το Raspberry διαθέτει 1GB μνήμης RAM και στη περίπτωση που διαθέτει λειτουργικό σύστημα με γραφικό περιβάλλον, διαθέτει περίπου 300 από αυτά για το λειτουργικό και στον αριθμό αυτό προστίθενται οι απαιτήσεις των υπολοίπων εφαρμογών που πιθανόν να εκτελεί ο χρήστης. Συνεπώς δεν είναι ασφαλές να χρησιμοποιήσει το μοντέλο αυτής της έκτασης παράλληλα με το λειτουργικό σύστημα που διαθέτει γραφικό περιβάλλον. Για το λόγο αυτό δημιουργήθηκε ένα δεύτερο μοντέλο με τη χρήση μόνο των πρώτον 10 χιλιάδων εικόνων. Το νέο μοντέλο που παράγεται είναι της έκτασης των 60MB.

```
53 smallData = normImgs[:10000]
54 smallLabel = trainLabArr[:10000]
55
56
57
58 start_time = time.time()
59 classifier.fit(smallData,smallLabel)
60
61 print("--- %s seconds ---" % (time.time() - start_time))
62 # save the model to disk
63 filename = 'finalized_modelL10k.sav'
64 pickle.dump(classifier, open(filename, 'wb'))
```

Εικόνα 34: Παραγωγή μοντέλου μόνο με τις πρώτες 10 χιλιάδες εικόνες. Πηγή: Συγγραφέας

Η βιβλιοθήκη μας παρέχει και ένα αριθμό εργαλείων προς εκτίμηση του παραγόμενου μοντέλου. Ένα εξ αυτών μετρά την ακρίβεια του μοντέλου συγκρίνοντας τις πραγματικές τιμές των δεδομένων με αυτές που το μοντέλο παρήγαγε. Συνεπώς αρχικά θα πρέπει να ζητηθεί από το μοντέλο να εκτιμήσει ένα σύνολο εικόνων και ακολούθως να ελεγχθούν τα αποτελέσματα. Για το σκοπό αυτό θα χρησιμοποιηθούν οι εικόνες δοκιμής που παρέχονται από το MNIST.

Προκειμένου να ολοκληρωθεί η διαδικασία θα πρέπει τα δεδομένα να προετοιμαστούν σύμφωνα με την διαδικασία που παρουσιάστηκε. Ποιο συγκεκριμένα, τα δεδομένα αφού διαβαστούν από το αρχείο θα φορτωθούν σε πίνακες τύπου numpy. Ακολούθως θα πρέπει να μετατραπούν σε μονής διάστασης και να κανονικοποιηθούν. Τέλος θα φορτωθεί το μοντέλο που έχει δημιουργηθεί στο προηγούμενο στάδιο και θα του ζητηθεί να εκτιμήσει τα δεδομένα μέσω της συνάρτησης predict. Η συνάρτηση λαμβάνει ως όρισμα τον πίνακα με τις εικόνες και επιστρέφει ένα πίνακα διαστάσεων όσες είναι και οι εικόνες που περιέχει τα ονόματα που τους έδωσε. Η ακρίβεια του μοντέλου υπολογίζεται με την accuracy_score του αντικειμένου metrics το οποίο λαμβάνει ως όρισμα τον πίνακα ονομάτων που παρήγαγε το μοντέλο και τον πίνακα των πραγματικών ονομάτων, τους συγκρίνει και ακολούθως εκτυπώνει το ποσοστό ακρίβειας.

```
12 testFile = 'Mnist/t10k-images.idx3-ubyte'
13 testArr = idx2numpy.convert_from_file(testFile)
14
15 testLabFile = 'Mnist/t10k-labels.idx1-ubyte'
16 testLabArr = idx2numpy.convert_from_file(testLabFile)
17
18 n_samples = len(testArr)
19 flatImgArr = testArr.reshape((n_samples, -1))
20 normImgs = preprocessing.normalize(flatImgArr)
21
22
23 filename = 'finalized_modelAutoGamma.sav'
24
25 loaded_model = pickle.load(open(filename, 'rb'))
26
27 start_time = time.time()
28 mypred = loaded_model.predict(normImgs)
29 print("--- %s Predict seconds ---" % (time.time() - start_time))
30
31 print("Accuracy:",metrics.accuracy_score(testLabArr, mypred))
```

Console

```
<terminated> newMetric.py [H:\PythonVE\distrMachLearn\Scripts\python.exe]
--- 933.3621349334717 Predict seconds ---
Accuracy: 0.8626
```

Εικόνα 35: Εκτίμηση της ακρίβειας του μοντέλου. Πηγή: Συγγραφέας

Το μοντέλο επιτυγχάνει ακρίβεια της τάξης του 86%, συνεπώς θα πρέπει να αναγνωρίζει με επιτυχία σχεδόν 9 στις 10 εικόνες. Για την ταξινόμηση των 10 χιλιάδων εικόνων ο υπολογιστής χρειάστηκε 933 δευτερόλεπτα. Παράλληλα το μικρότερο μοντέλο (των 10 χιλιάδων εικόνων αντί των 60) παρουσιάζει ακρίβεια της τάξης του 41%.

```
12 testFile = 'Mnist/t10k-images.idx3-ubyte'
13 testArr = idx2numpy.convert_from_file(testFile)
14
15 testLabFile = 'Mnist/t10k-labels.idx1-ubyte'
16 testLabArr = idx2numpy.convert_from_file(testLabFile)
17
18 n_samples = len(testArr)
19 flatImgArr = testArr.reshape((n_samples, -1))
20 normImgs = preprocessing.normalize(flatImgArr)
21
22
23 #filename = 'finalized_modelAutoGamma.sav'
24 filename = 'finalized_modelL10k.sav'
25
26 loaded_model = pickle.load(open(filename, 'rb'))
27
28 start_time = time.time()
29 mypred = loaded_model.predict(normImgs)
30 print("--- %s Predict seconds ---" % (time.time() - start_time))
31
32 print("Accuracy:",metrics.accuracy_score(testLabArr, mypred))
```

Console

<terminated> newMetric.py [H:\PythonVE\distrMachLearn\Scripts\python.exe]

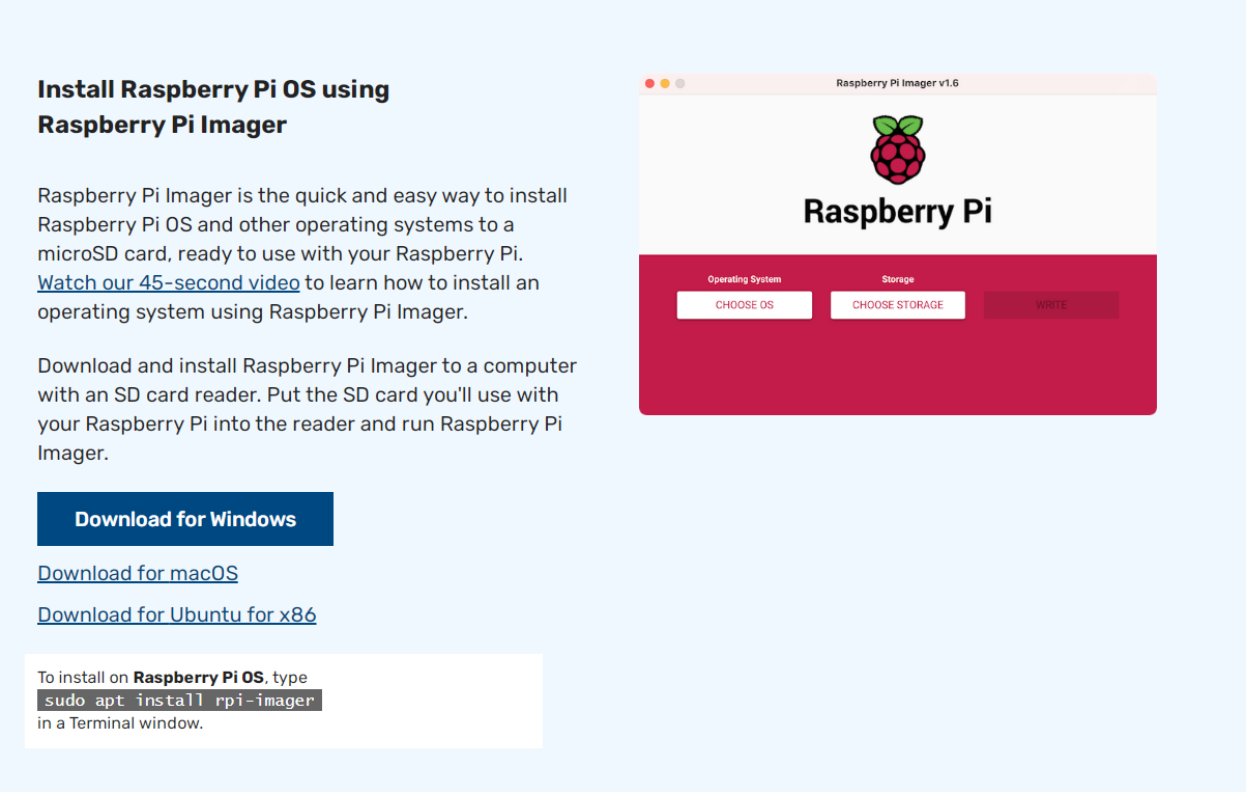
```
--- 153.1291196346283 Predict seconds ---
Accuracy: 0.4104
```

Εικόνα 36: Εκτίμηση ακρίβειας μοντέλου των 10 χιλιάδων δειγμάτων. Πηγή: Συγγραφέας

4.3.3 Ταξινόμηση εικόνων από το Raspberry

Προκειμένου η διαδικασία να επιτευχθεί από το Raspberry θα πρέπει αρχικά να δημιουργηθεί το κατάλληλο περιβάλλον που θα εκτελείται η εφαρμογή, ακολούθως να μεταφερθεί το μοντέλο και τέλος να δοθεί στη συσκευή το σύνολο των εικόνων που θα επεξεργαστεί. Για το πρώτο βήμα η διαδικασία δεν απέχει ιδιαίτερα από τη διαδικασία που περιγράφηκε και ακολουθήθηκε στον υπολογιστή που παρήγαγε το μοντέλο δεδομένου ότι η συσκευή διαθέτει το λειτουργικό σύστημα που συνιστά η εταιρία.

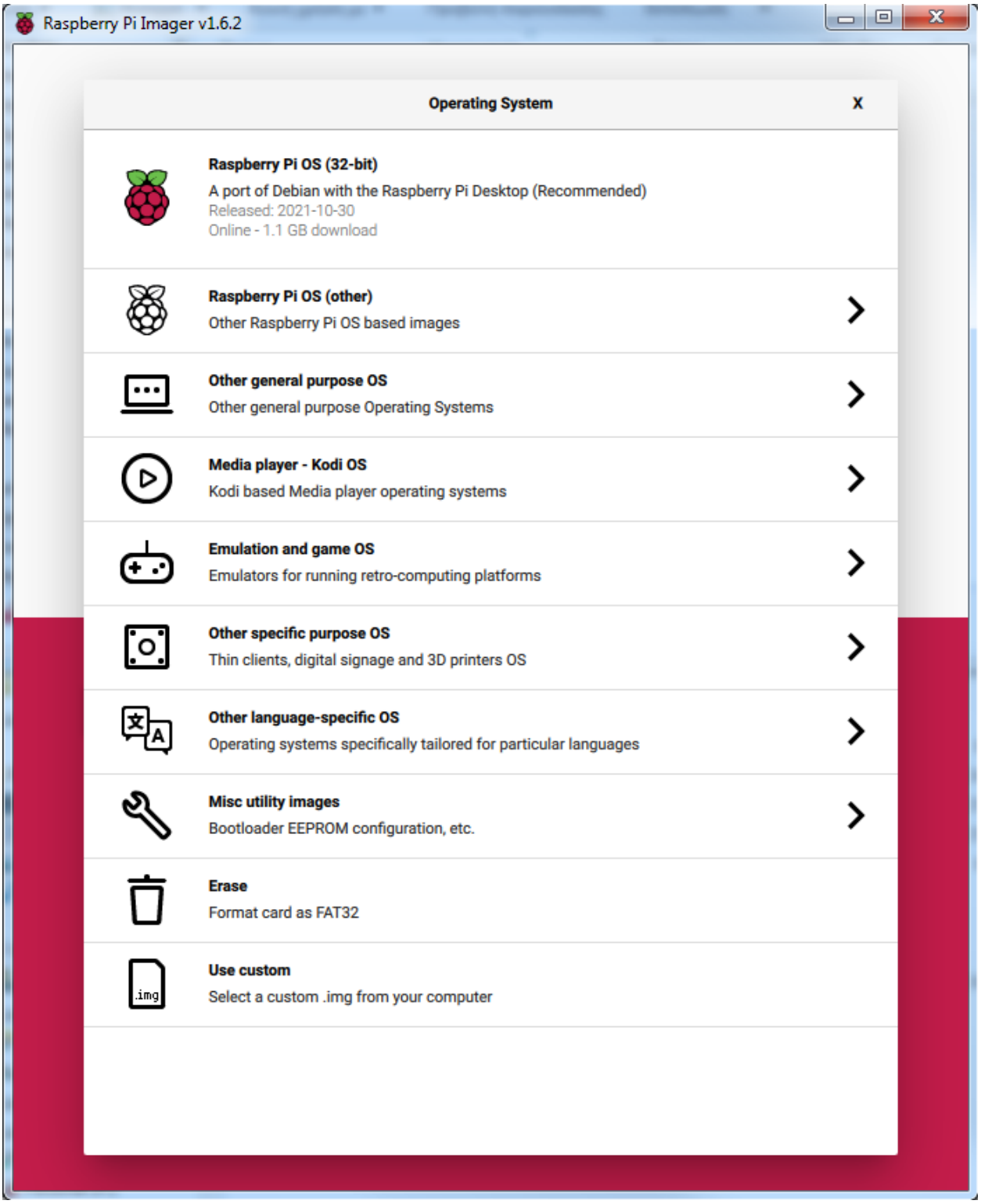
Για την εγκατάσταση του λειτουργικού συστήματος στη συσκευή απαιτείται η χρήση μιας κάρτας τύπου MicroSD χωρητικότητας άνω των 16GB. Η κάρτα αυτή θα πρέπει να εισαχθεί στον Η/Υ μέσω κατάλληλου μετατροπέα (adaptor) ώστε να συνδέεται στη θύρα USB του υπολογιστή και ακολούθως να γραφτεί σε αυτή το επιθυμητό λειτουργικό σύστημα μέσω της εφαρμογής Raspberry Pi Imager. Η εφαρμογή είναι διαθέσιμη από τον επίσημο ισότοπο της εταιρίας https://www.raspberrypi.com/software/.



Εικόνα 37: Λήψη της εφαρμογής Raspberry Pi Imager. Πηγή: https://www.raspberrypi.com/software/

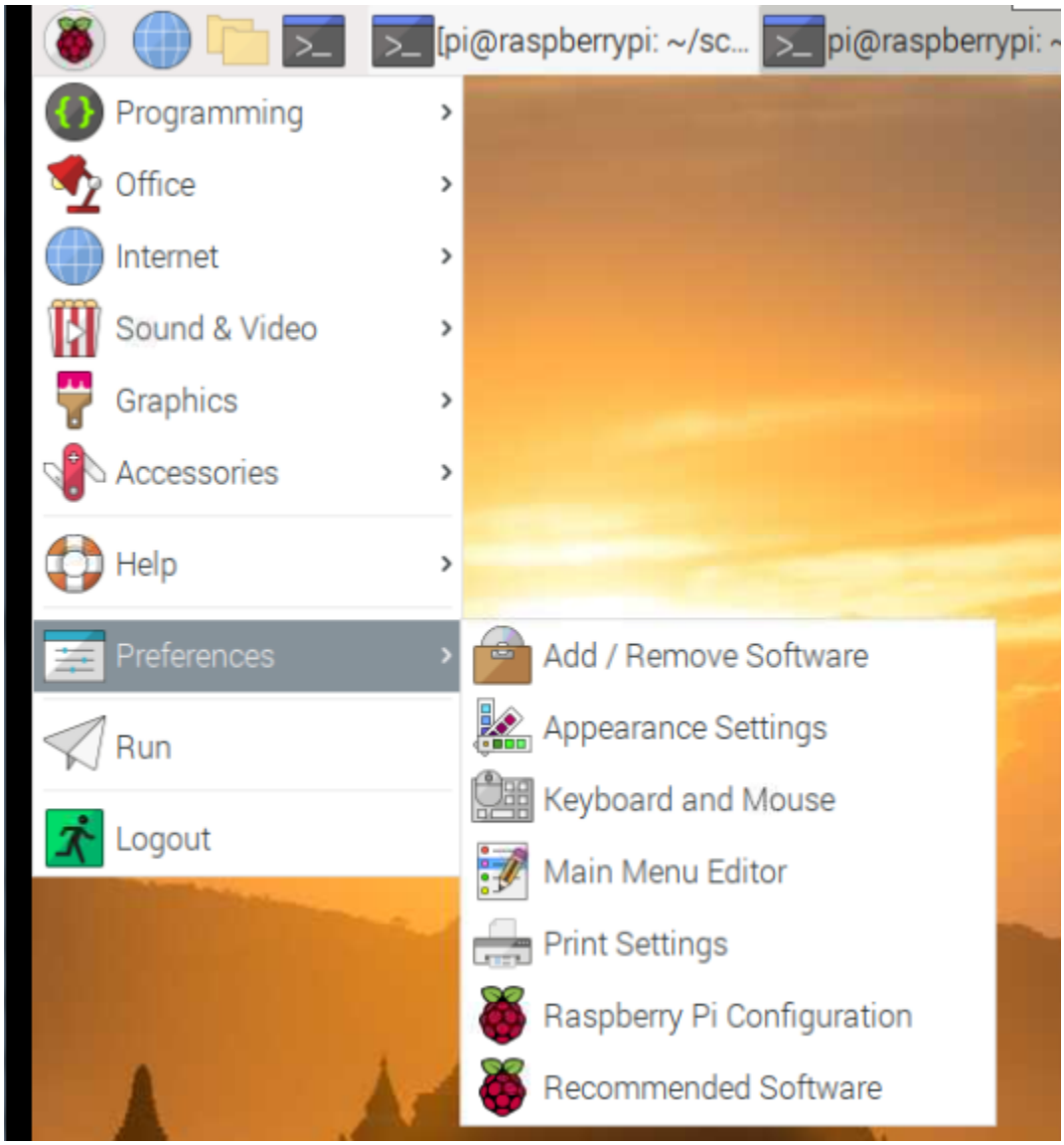
Μετά τη λήψη της εφαρμογής και την εγκατάσταση στον υπολογιστή ο χρήστης θα επιλέξει το λειτουργικό σύστημα που η εφαρμογή θα κατεβάσει στον υπολογιστή και ακολούθως θα εισάγει στην κάρτα MicroSD. Διατίθεται ικανοποιητικός αριθμός επιλογών με την εταιρία να συνιστά το Raspberry Pi OS 32 για το Raspberry Pi 3 B+. Μετά την επιλογή του λειτουργικού και του μέσου αποθήκευσης η εφαρμογή θα κατεβάσει το λειτουργικό και στη συνέχεια θα το περάσει στη κάρτα μνήμης αφού πρώτα ενημερώσει το χρήστη ότι θα διαγράψει το σύνολο των

δεδομένων που ενδεχόμενος να υπάρχουν στη κάρτα. Το λειτουργικό είναι σε μορφή εικόνας (img) και συνεπώς δεν απαιτείται κάποια προηγούμενη προετοιμασία της κάρτας αφού κατά την εισαγωγή του αρχείου εικόνας η κάρτα αποκτά την απαιτούμενη παραμετροποίηση.

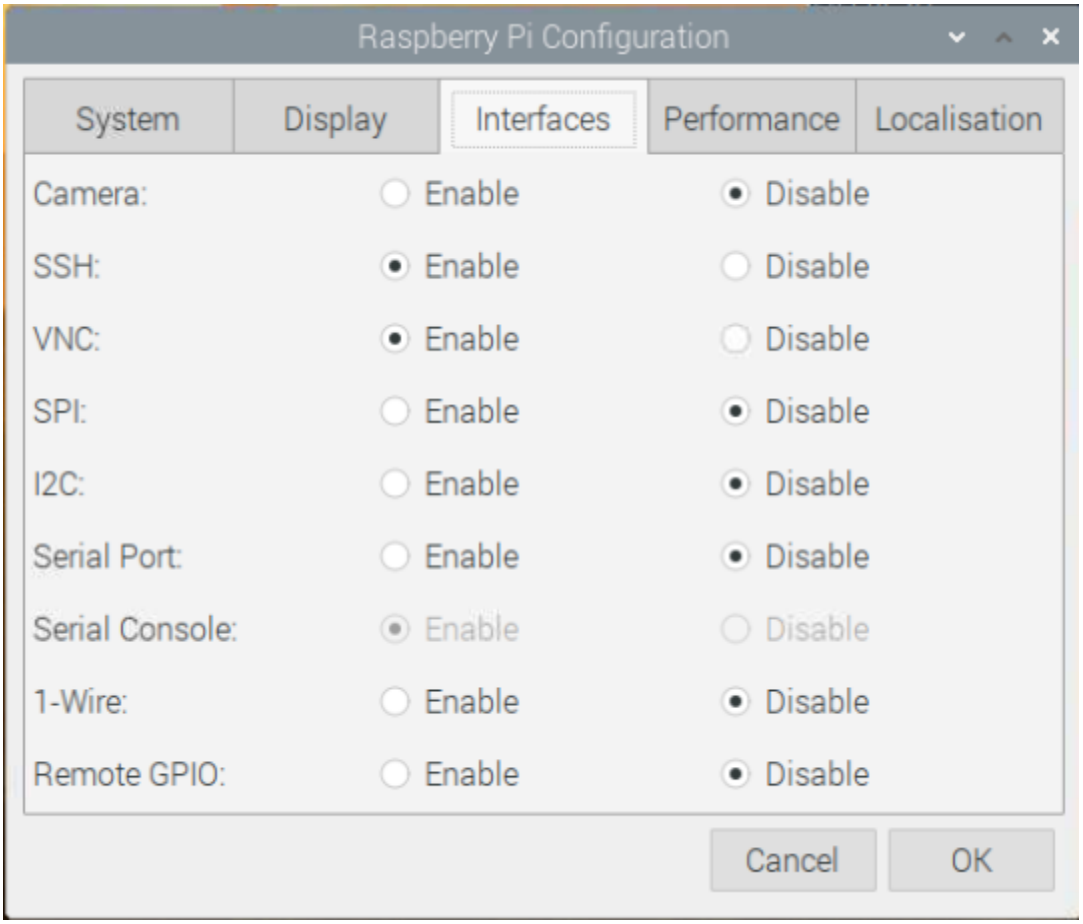


Εικόνα 38: Επιλογή λειτουργικού συστήματος για το Raspberry Pi. Πηγή: Συγγραφέας

Στη συνέχεια η κάρτα εισάγεται στη συσκευή και ακολούθως αυτή τίθεται σε λειτουργία, αφού πρώτα συνδεθεί με τις υπόλοιπες περιφερικές συσκευές, όπως το πληκτρολόγιο, το ποντίκι, την οθόνη και ενδεχόμενος το καλώδιο δικτύου, στη περίπτωση που η συσκευή θα συνδέεται με τον τρόπο αυτό στο δίκτυο. Ακολούθως ο χρήστης θα πρέπει να εισάγει διάφορα στοιχεία στο Raspberry όπως τη γλώσσα, τον τύπο του πληκτρολογίου, τις παραμέτρους του δικτύου καθώς και άλλα και στη συνέχεια να επανεκκινήσει τη συσκευή ώστε να φορτωθούν οι νέες ρυθμίσεις. Για να μπορεί να συνδέεται μέσω ssh ή VNC θα πρέπει η υπηρεσία να ενεργοποιηθεί μέσω του μενού.

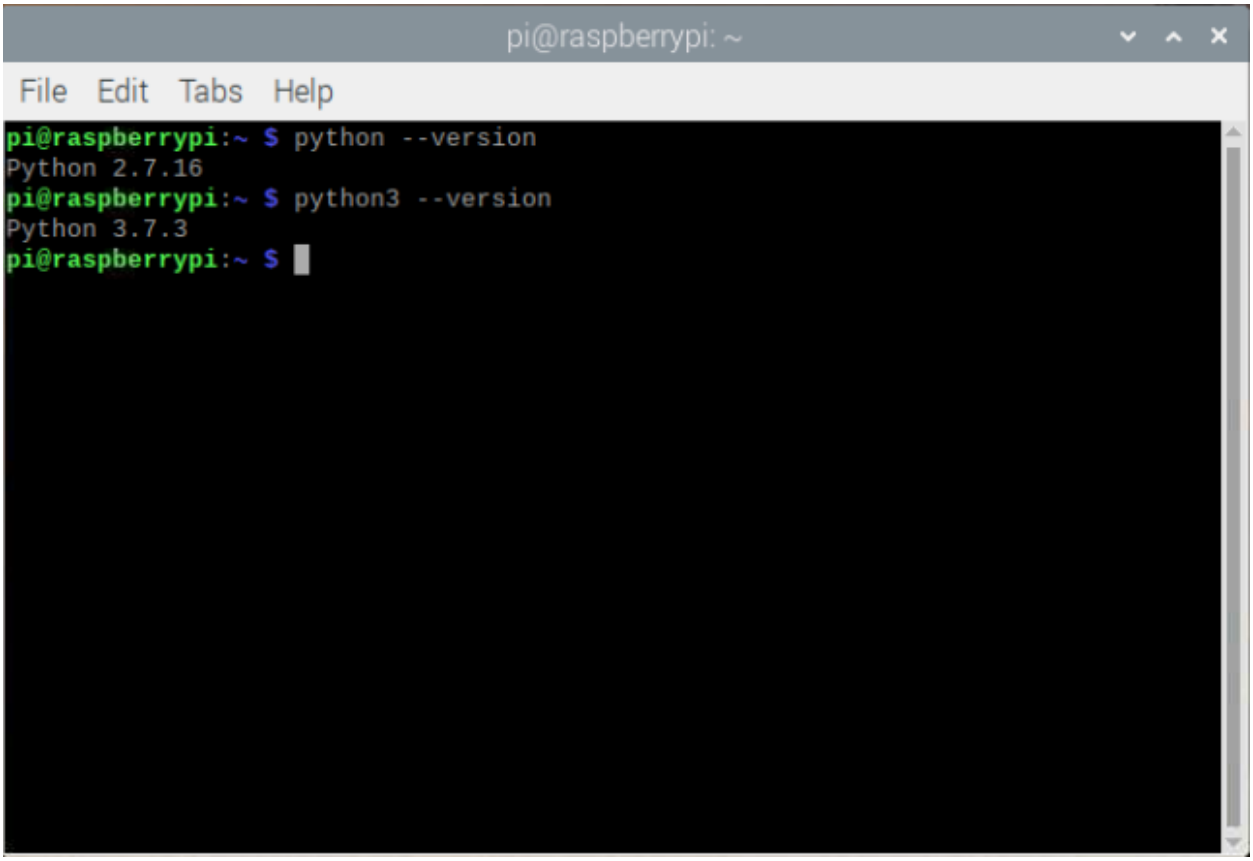


Εικόνα 39: Ενεργοποίηση της εφαρμογής διαχείρισης του Raspberry. Πηγή: Συγγραφέας



Εικόνα 40: Παραμετροποίηση της συσκευής. Πηγή: Συγγραφέας

Το λειτουργικό σύστημα είναι εφοδιασμένο με δύο εκδόσεις της python, την 2.7 και την 3.7. Συνεπώς η διαδικασία για την προετοιμασία του περιβάλλοντος στο οποίο θα εκτελεστεί η εφαρμογή δεν απέχει από την διαδικασία που ακολουθήθηκε για τον υπολογιστή.



Εικόνα 41: Έκδοση της Python στη συσκευή. Πηγή: Συγγραφέας

Το Raspberry θα χρησιμοποιήσει το ήδη παραχθέν μοντέλο και συνεπώς δεν απαιτούνται όλες οι βιβλιοθήκες που εγκαταστάθηκαν στον υπολογιστή. Και εδώ, για λόγους διευκόλυνσης της παρουσίασης θα δημιουργηθεί ένα εικονικό περιβάλλον που θα φιλοξενήσει την εφαρμογή, σε πραγματικές συνθήκες όμως αυτό μάλλον δεν είναι απαραίτητο καθώς στη συσκευή πιθανότατα να μην εγκατασταθεί άλλη εφαρμογή. Επιπλέον, σε πραγματικές συνθήκες το λειτουργικό της συσκευής πιθανόν να μην διαθέτει γραφικό περιβάλλον για λόγους αποδοτικότερης διαχείρισης των διαθέσιμων υπολογιστικών πόρων. Το σύνολο των απαραίτητων βιβλιοθηκών θα εγκατασταθούν μέσω της PIP όπως ακριβώς και στον υπολογιστή. Οπότε αρχικά θα δημιουργηθεί το εικονικό περιβάλλον μέσω του module venv της python 3.7 και στη συνέχεια θα ενεργοποιηθεί:

```
python3 -m venv scikit
```

```
source /scikit/bin/activate
```

Στη συνέχεια θα εγκατασταθούν οι βιβλιοθήκες. Αρχικά η scikit ώστε να χρησιμοποιηθεί η predict η οποία θα ταξινομήσει τις εικόνες, αλλά και η normalize η οποία θα τις κανονικοποιήσει.

```
pip install scikit-learn
```

Ακολούθως η Numpy για τη χρήση των πινάκων

```
pip install numpy
```

Η Pickle για το διάβασμα του μοντέλου

```
pip install pickle-mixin
```

Η Glob για το διάβασμα των αρχείων εικόνας

```
pip install glob2
```

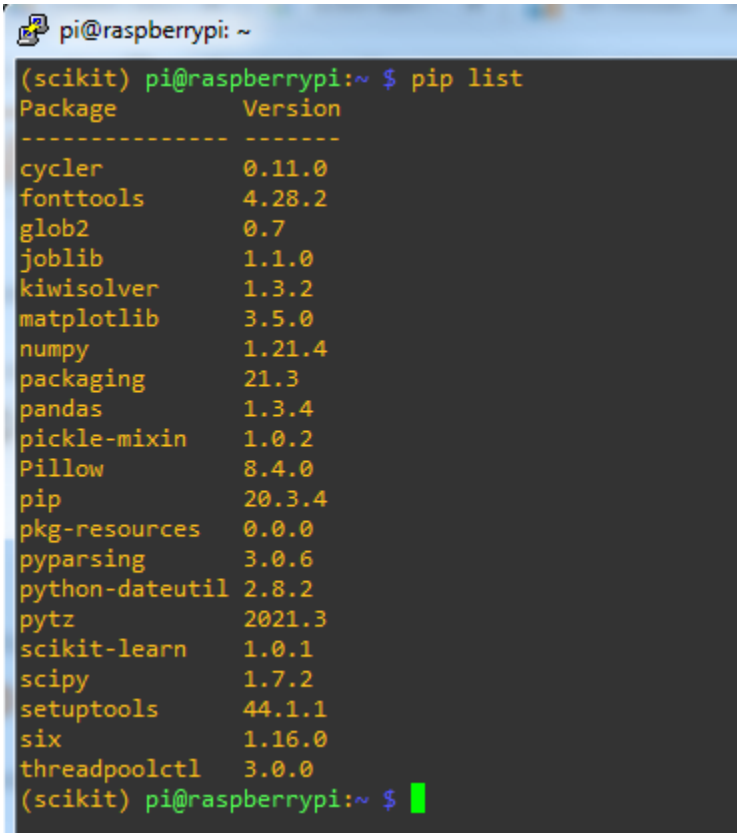
Η matplotlib για διάβασμα και προβολή των εικόνων

```
pip install matplotlib
```

Τέλος η pandas για τα dataframes

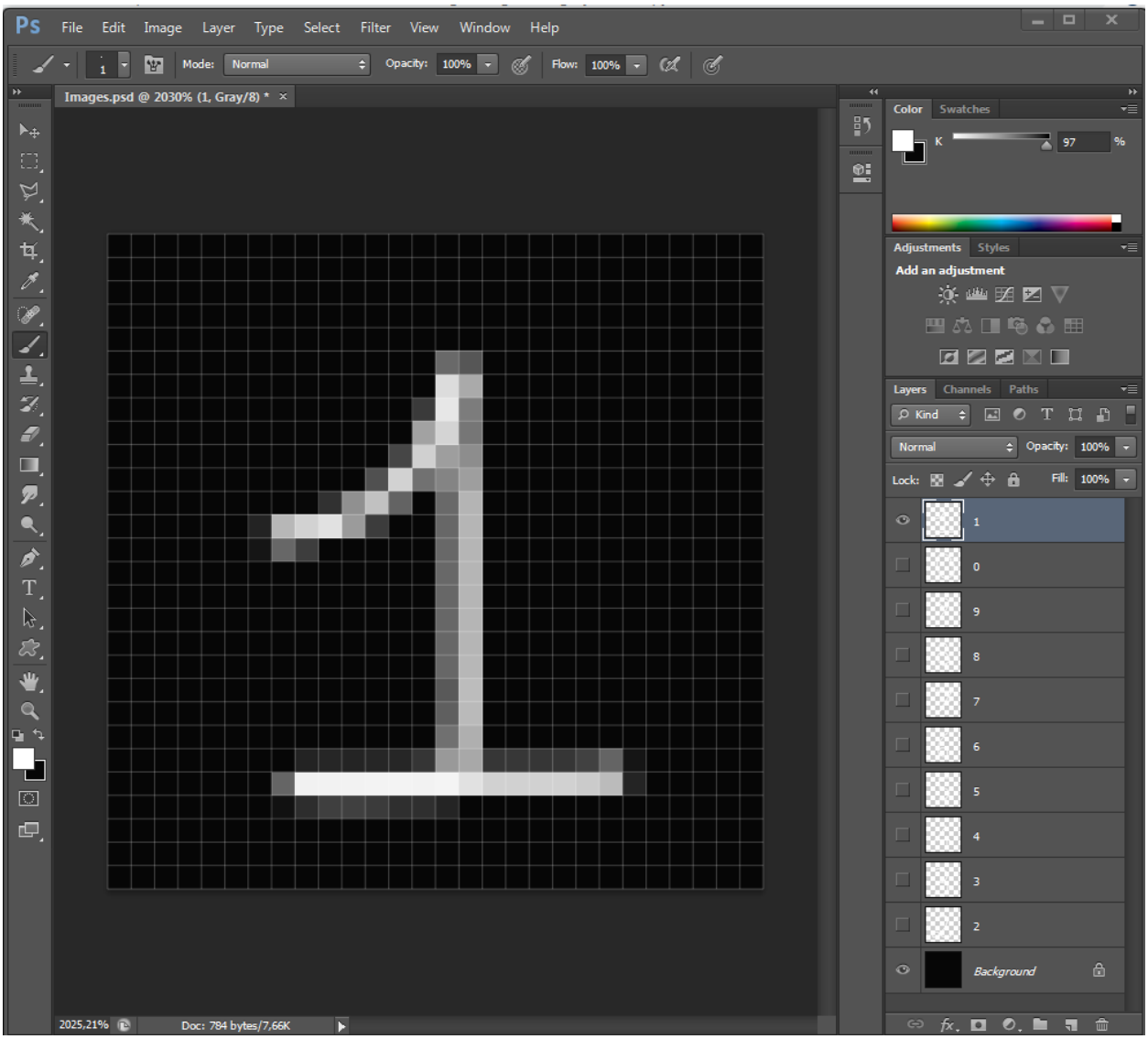
```
pip install pandas
```

Εκτελώντας την εντολή pip list το Raspberry δίνει μια συγκεντρωτική κατάσταση των πακέτων που έχουν εγκατασταθεί στο συγκεκριμένο περιβάλλον μαζί με τον αριθμό της έκδοσης τους.

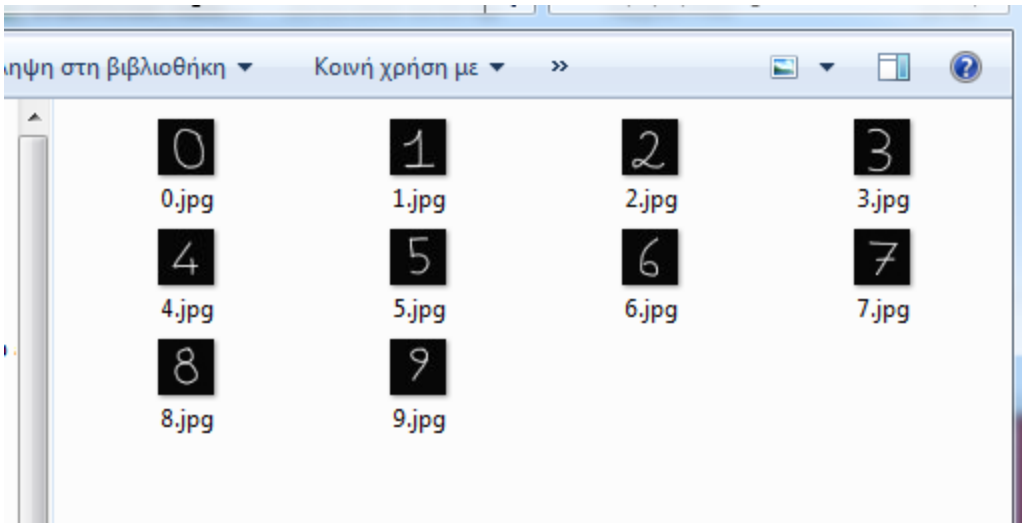


Εικόνα 42: Το σύνολο των πακέτων που εγκαταστάθηκαν στο Raspberry. Πηγή: Συγγραφέας

Για τις εικόνες ελέγχου χρησιμοποιήθηκε η εφαρμογή Photoshop (www. Adobe.com) για την παραγωγή δέκα εικόνων διαστάσεων 28 επι 28 οι οποίες εμφανίζουν χειρόγραφους τους αριθμούς του δεκαδικού συστήματος. Οι αριθμοί σχεδιάστηκαν με άσπρο χρώμα πάνω σε μαύρο φόντο ώστε οι εικόνες να έχουν κοντά στο 0 για το φόντο και τιμές κοντά στο 255 για το λευκό.

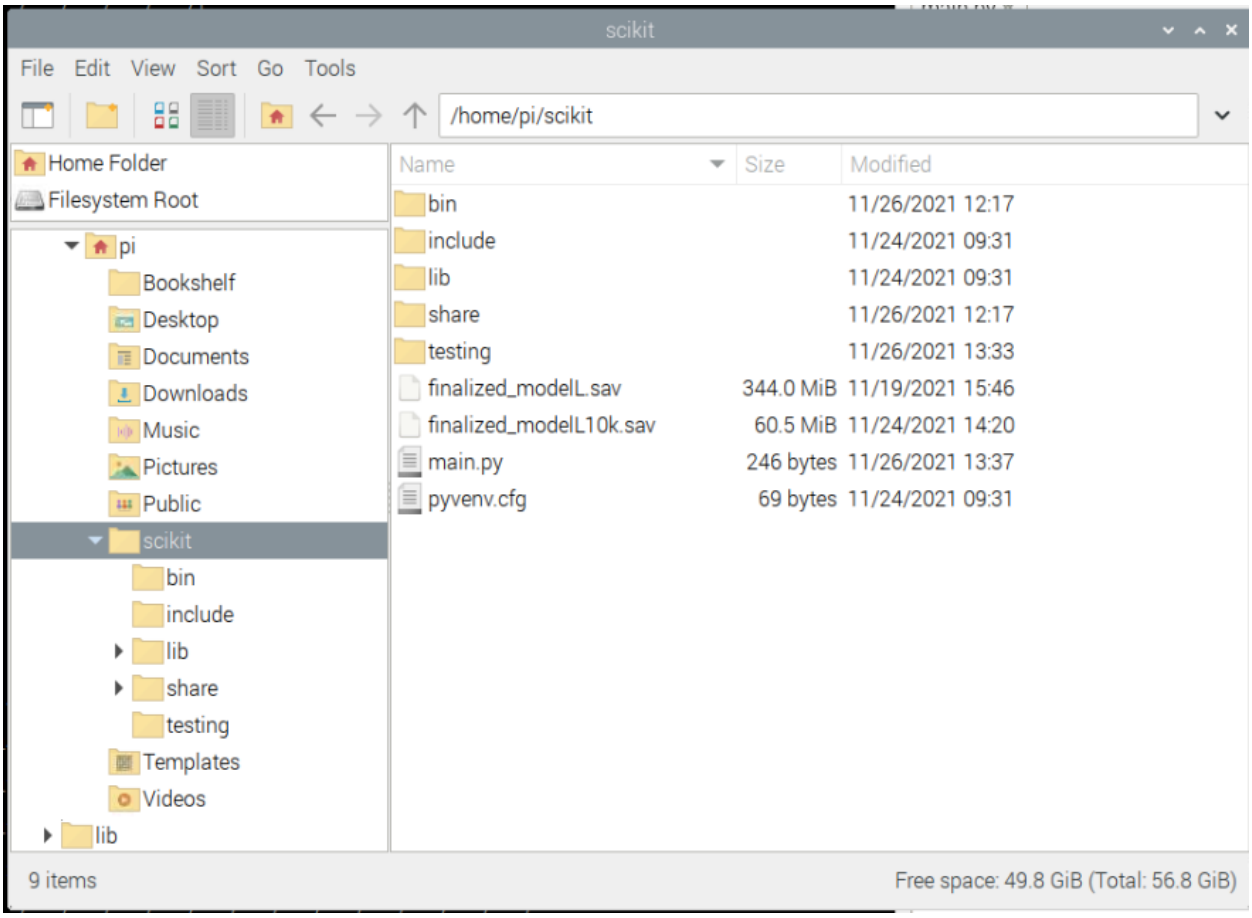


Εικόνα 43: Παραγωγή εικόνων προς ταξινόμηση στην εφαρμογή Photoshop. Πηγή: Συγγραφέας



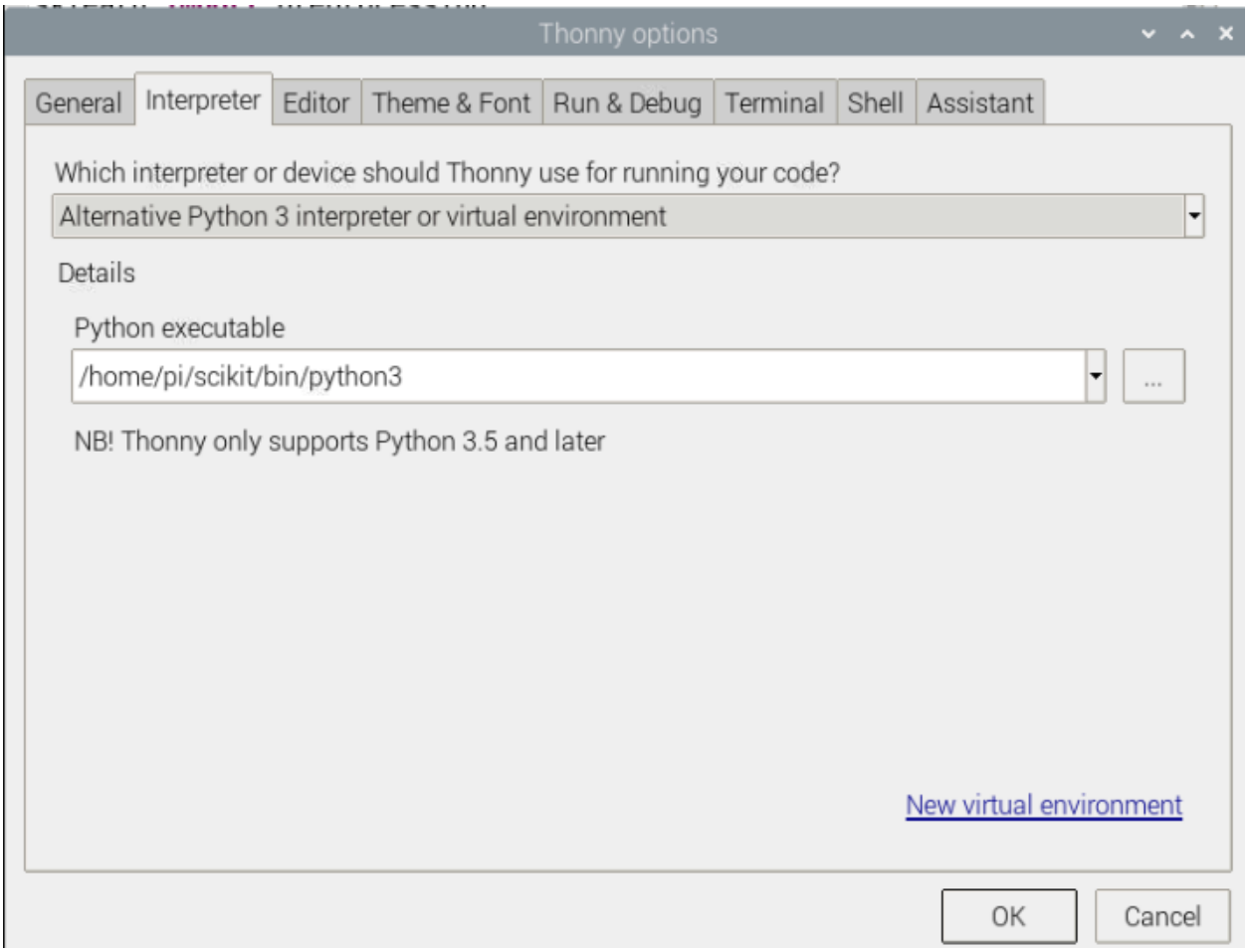
Εικόνα 44: Το dataset που θα ταξινομηθεί από το Raspberry. Πηγή: Συγγραφέας

Οι εικόνες θα μεταφερθούν στο φάκελο του εικονικού περιβάλλοντος, μέσα σε ένα φάκελο με την ονομασία ‘testing’. Παράλληλα, στην ίδια τοποθεσία θα μεταφερθούν και τα 2 μοντέλα που έχουν παραχθεί. Η μεταφορά μπορεί να γίνει με πολλούς τρόπους, στην παρούσα παρουσίαση επιλέχθηκε το πρωτόκολλο SFTP.



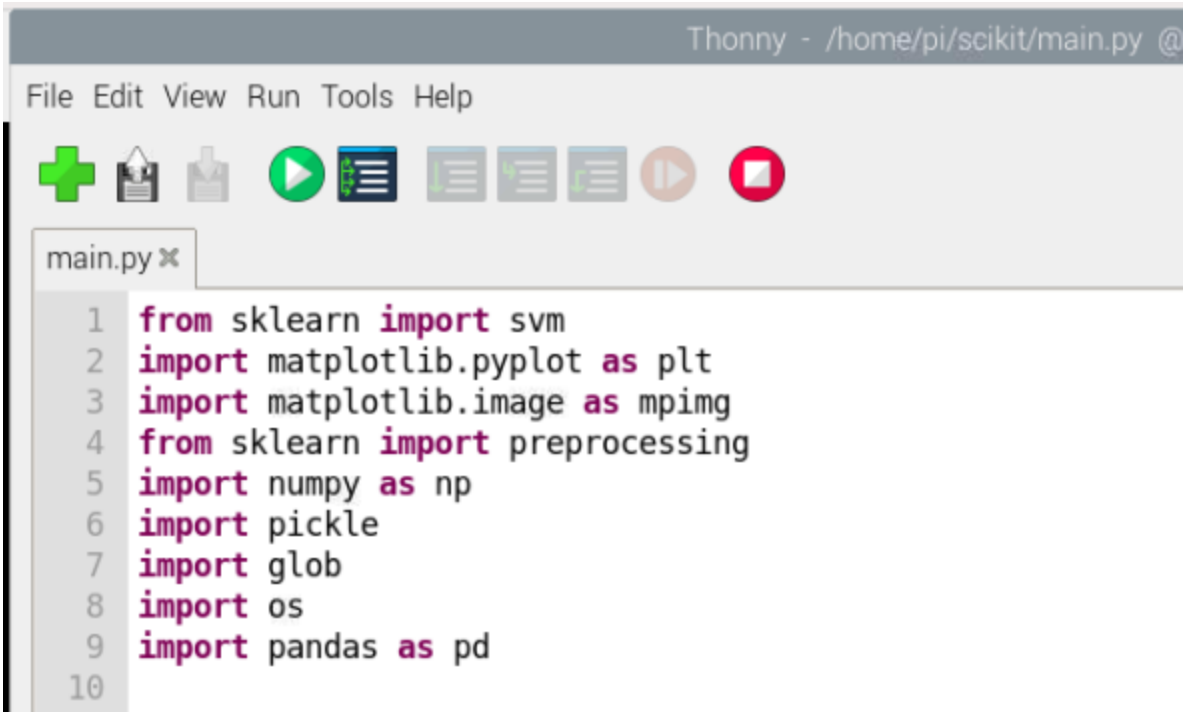
Εικόνα 45: Η δομή των φακέλων. Πηγή: Συγγραφέας

Ο κώδικάς της εφαρμογής θα συγγραφεί στο Raspberry και πιο συγκεκριμένα στο ολοκληρωμένο περιβάλλον ανάπτυξης Thonny Python. Σε αυτό θα δηλωθεί ως μεταφραστής αυτός που έχει δημιουργηθεί στο εικονικό περιβάλλον.



Εικόνα 46: Παραμετροποίηση Thonny Python IDE. Πηγή: Συγγραφέας

Το περιβάλλον είναι πλέον έτοιμο να δεχθεί τον κώδικα. Συνεπώς αρχικά γίνεται η απαραίτητη εισαγωγή των βιβλιοθηκών.

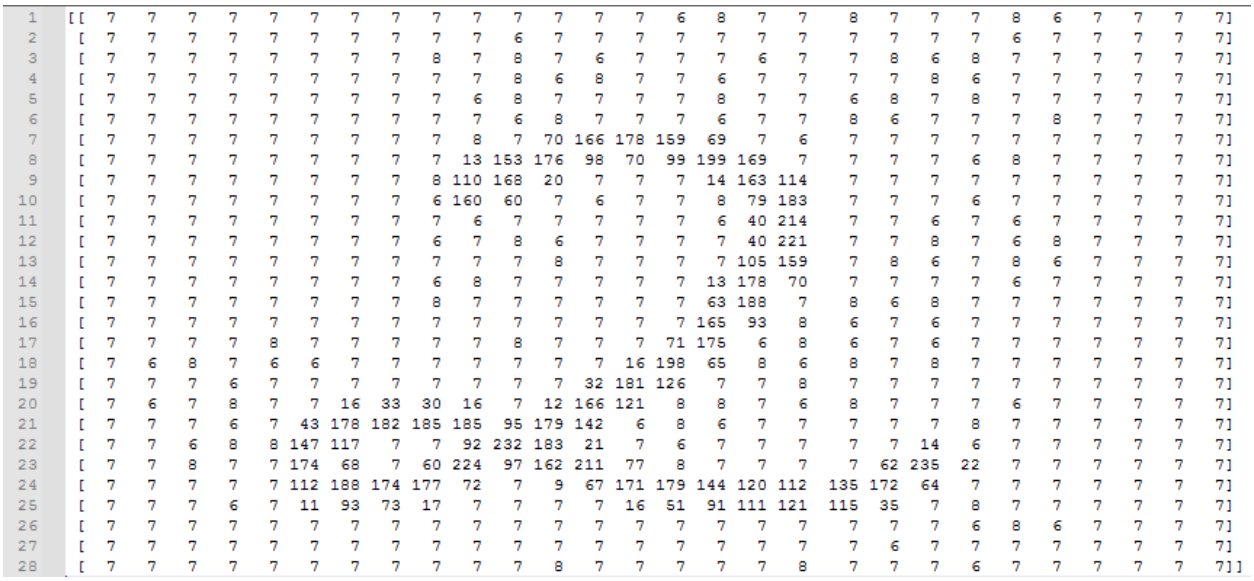


Εικόνα 47: Εισαγωγή των απαραίτητων βιβλιοθηκών. Πηγή: Συγγραφέας

Ακολούθως διαβάζονται οι εικόνες. Για λόγους διευκόλυνσης της παρουσίασης παρουσιάζεται η 2η εικόνα που αναπαριστά τον αριθμό 2.

```
11
12 img = mpimg.imread('testing/2.jpg')
13 print(img)
```

Εικόνα 48: Διάβασμα και προβολή του αρχείου 2.jpg. Πηγή: Συγγραφέας



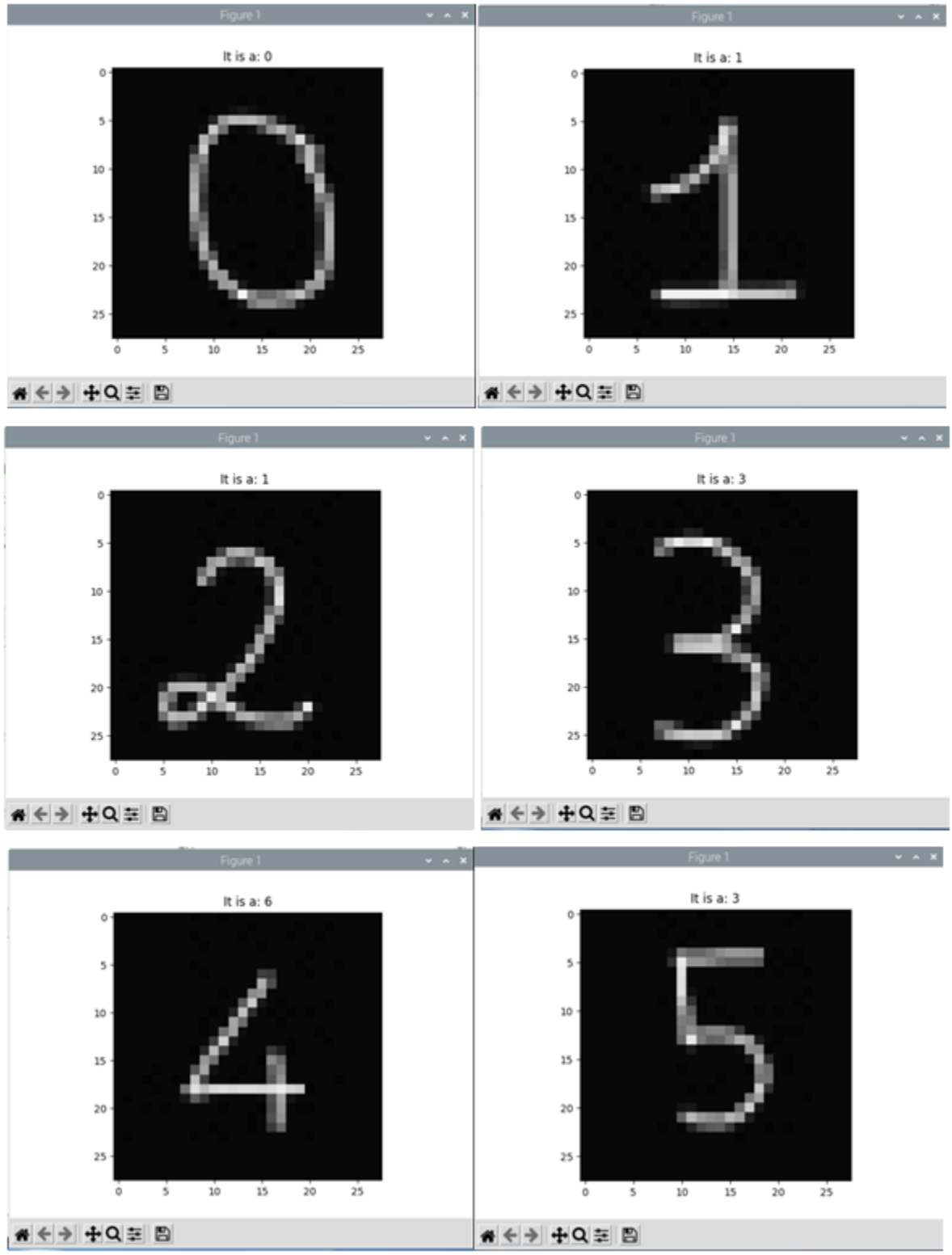
Εικόνα 49: Η εικόνα που παράχθηκε με το Photoshop. Πηγή: Συγγραφέας

Ο πίνακας είναι διαστάσεων 28x28 και περιέχει θετικούς ακέραιους από το 6 έως το 235. Συνεπώς τα δεδομένα θα πρέπει να κανονικοποιηθούν ώστε να είναι κατάλληλα για το μοντέλο. Ακολούθως ο πίνακας θα πρέπει να γίνει μονής διάστασης και να δοθεί σαν όρισμα στη συνάρτηση predict του μοντέλου που έχει δημιουργηθεί από το μικρό αρχείο του μοντέλου. Η συνάρτηση επιστρέφει ένα πίνακα διαστάσεων ίσων με τον αριθμό τον εικόνων που περιέχει ο πίνακας όρισμα. Στη συνέχεια ο κώδικας εμφανίζει την εικόνα, μέσω της βιβλιοθήκης matplotlib χρησιμοποιώντας σαν χρώματα το χρωματικό χάρτη gray. Το σύνολο του κώδικα, εκτός από την δημιουργία του μοντέλου εισάγεται σε ένα βρόχο επανάληψης ο οποίο εκτελείται τόσες φορές όσος και ο αριθμός των αρχείων που περιέχονται στο φάκελο testing.

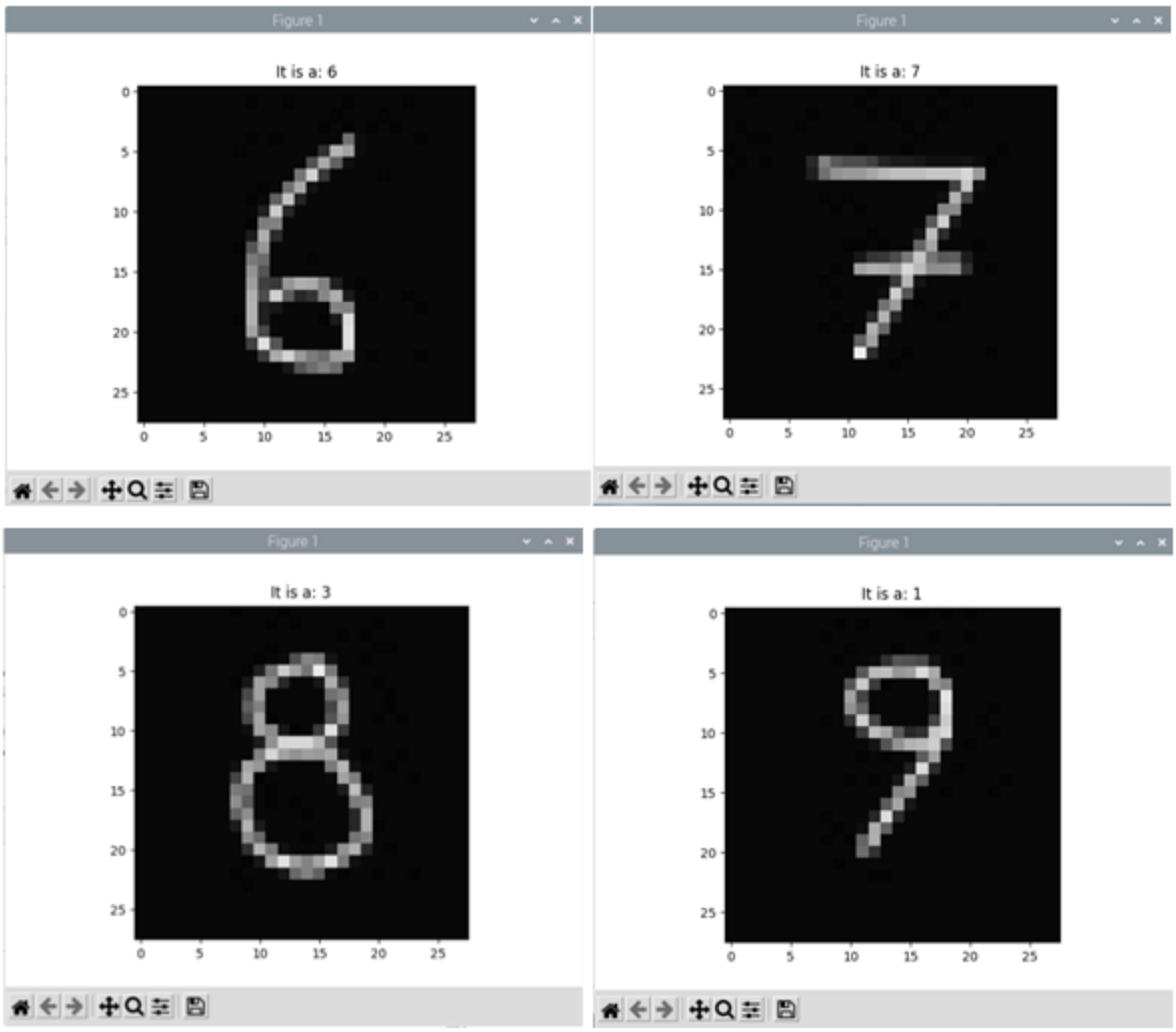
```
model = 'finalized_modelL10k.sav'
loaded_model = pickle.load(open(model, 'rb'))

for filename in glob.glob('testing/*.jpg'):
    img = mpimg.imread(filename)
    normImg = preprocessing.normalize(img)
    norSing = normImg.reshape(1,28*28)
    mypred = loaded_model.predict(norSing)
    plt.imshow(img,cmap='gray',vmin=0,vmax=255)
    plt.title("It is a: " + str(mypred[0]))
    plt.show()
```

Εικόνα 50: Ο κώδικας επεξεργασίας των εικόνων. Πηγή: Συγγραφέας



Εικόνα 51: Τα έξι πρώτα παράθυρα που παρουσιάζει η εφαρμογή. Πηγή: Συγγραφέας



Εικόνα 52: Τα επόμενα τέσσερα παράθυρα που εμφανίζει η εφαρμογή: Πηγή: Συγγραφέας

Η εφαρμογή εντοπίζει σωστά τις έξι από τις δέκα εικόνες. Ποιο συγκεκριμένα επιτυγχάνει να αναγνωρίσει το μηδέν, το ένα, το δύο, το τρία, το έξι και το επτά. Πετυχαίνει συνεπώς ακρίβεια τις τάξης του 60 % ενώ η μέτρηση της ακρίβειας του μοντέλου είχε δώσει 86%. Το ποσοστό αυτό παραμένει σταθερό ανεξάρτητα από τον αριθμό των ελέγχων. Ποιο συγκεκριμένα οι εικόνες της παρούσας παρουσίασης ταξινομήθηκαν δέκα φορές από το μοντέλο αυτό και πάντα τα αποτελέσματα ταυτίζονταν.

Προκειμένου να αποκτήσει ο αναγνώστης μια εικόνα των επεξεργαστικών δυνατοτήτων της συσκευής, τροποποιείται ο κώδικάς ώστε να μην εμφανίζει τις εικόνες και να εμφανίζει τα αποτελέσματα της ταξινόμησης αλλά και το χρόνο που απαιτήθηκε για την επεξεργασία τους. Το Raspberry χρειάστηκε 1.3 δευτερόλεπτα για την επεξεργασία των δέκα εικόνων διάστασης 28 επί 28.

```
15 start_time = time.time()
16
17 for filename in glob.glob('testing/*.jpg'):
18     img = mpimg.imread(filename)
19     normImg = preprocessing.normalize(img)
20     norSing = normImg.reshape(1,28*28)
21     mypred = loaded_model.predict(norSing)
22     print("File: " + filename + " is a :" + str(mypred[0]))
23
24 print("--- %s seconds ---" % (time.time() - start_time))
25
```

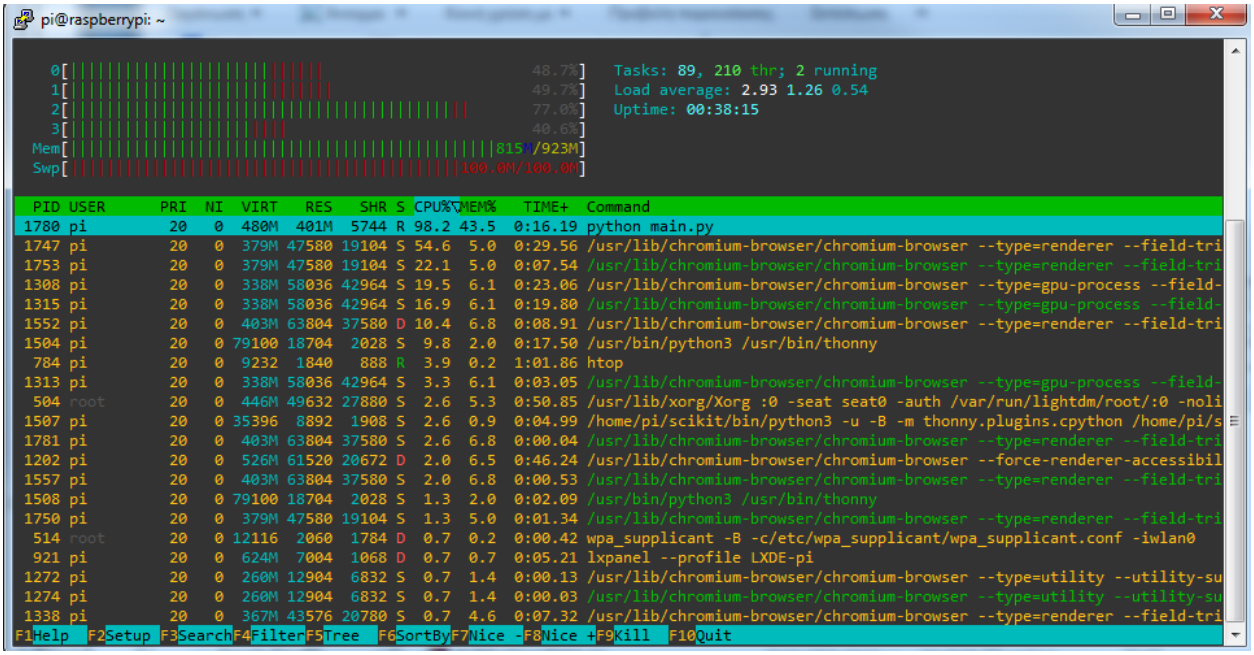
```
Shell
>>> %Run main.py

  File: testing/6.jpg is a :6
  File: testing/2.jpg is a :1
  File: testing/9.jpg is a :1
  File: testing/4.jpg is a :6
  File: testing/1.jpg is a :1
  File: testing/3.jpg is a :3
  File: testing/7.jpg is a :7
  File: testing/8.jpg is a :3
  File: testing/0.jpg is a :0
  File: testing/5.jpg is a :3
  --- 1.360154628753662 seconds ---
```

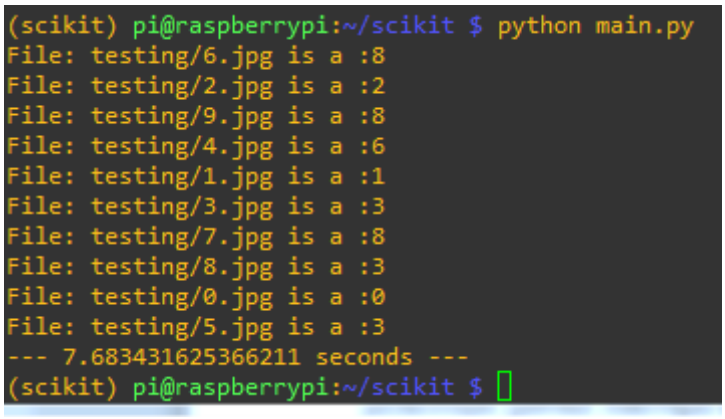
Εικόνα 53: Χρονομέτρηση της ταξινόμησης. Πηγή: Συγγραφέας

Το μοντέλο που παράχθηκε με το σύνολο των εικόνων και είναι μεγέθους 352Mb φέρνει το Raspberry στα όρια του καθώς η συσκευή διαθέτει 1GB μνήμη RAM και απαιτούνται περισσότερα από 300 για το λειτουργικό σύστημα και τις διάφορες εφαρμογές που πιθανόν να εκτελεί ο χρήστης παράλληλα με την επεξεργασία των εικόνων. Στη περίπτωση που η διαθέσιμη

μνήμη εξαντλείται, προς αποφυγή ‘παγώματος’ της συσκευής το λειτουργικό επεμβαίνει και τερματίζει την εκτέλεση της εφαρμογής.



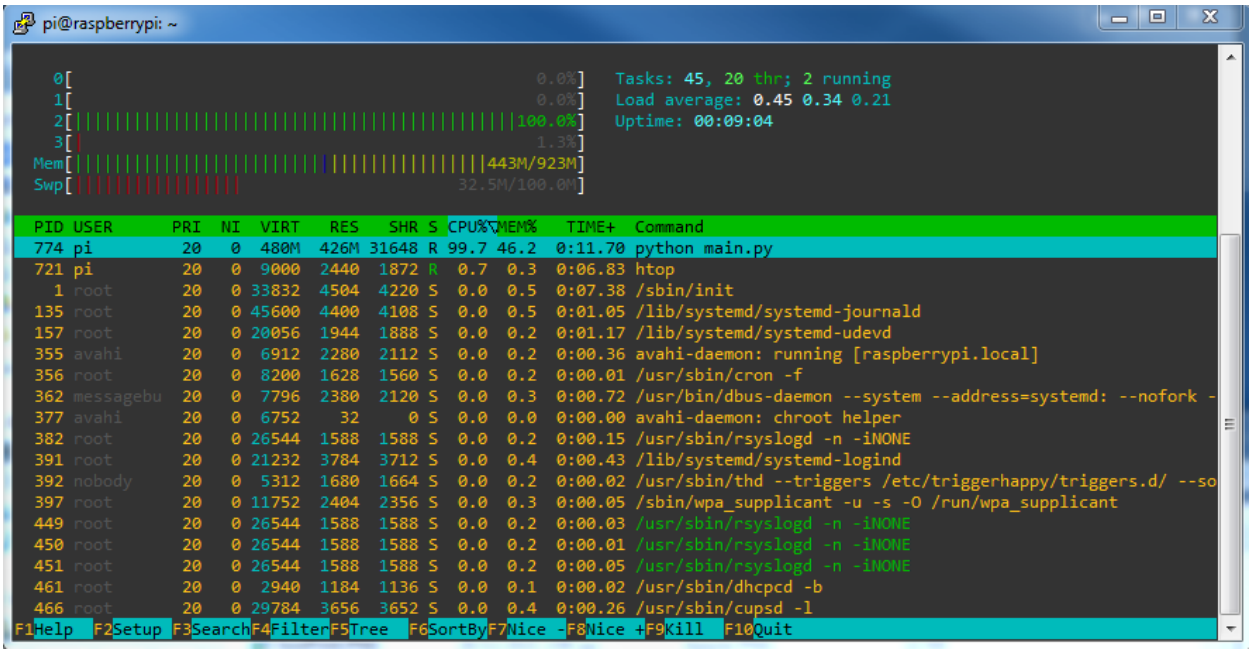
Εικόνα 54: Εκτέλεση της εφαρμογής με το γραφικό περιβάλλον ενεργοποιημένο. Πηγή: Συγγραφέας



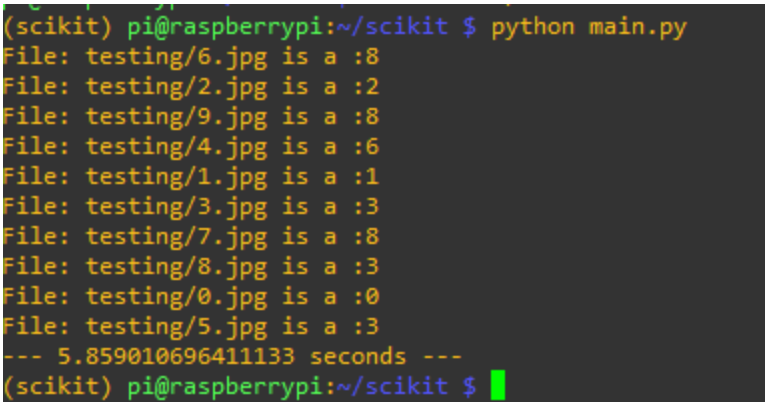
Εικόνα 55: Χρονική επιβάρυνση του μεγαλύτερου μοντέλου. Πηγή: Συγγραφέας

Κατά τη διάρκεια των εκτελέσεων η εφαρμογή ξεπέρασε σε απαιτήσεις τα 800MB και συχνά το λειτουργικό δεν της επέτρεπε να ολοκληρώσει. Στις περιπτώσεις που ολοκλήρωνε απαιτούσε μεγαλύτερο χρονικό διάστημα απ’ ότι χωρίς το γραφικό περιβάλλον (περίπου 2 δευτερόλεπτα).

Στη περίπτωση απουσίας του γραφικού περιβάλλοντος του λειτουργικού η εκτέλεση είναι αρκετά πιο ασφαλής καθώς οι απαιτήσεις σε μνήμη παραμένουν σε χαμηλά επίπεδα και πιο συγκεκριμένα κάτω των 500 MB (70MB για το λειτουργικό και 350MB για το μοντέλο).



Εικόνα 56: Εκτέλεση απουσία γραφικού περιβάλλοντος. Πηγή: Συγγραφέας



Εικόνα 57: Χρονική καθυστέρηση γραφικού περιβάλλοντος. Πηγή: Συγγραφέας

Από πλευράς αποτελεσμάτων, το πληρέστερο μοντέλο παρουσιάζει αισθητά χειρότερη απόδοση μιας και εντοπίζει τέσσερις εικόνες στις δέκα αντί των πέντε στις δέκα που εντοπίζει το

μικρότερο. Ποιο συγκεκριμένα εντοπίζει την εικόνα με τον αριθμό μηδέν, ένα, δύο και τρία ενώ το μικρότερο μοντέλο δεν εντόπιζε επιτυχώς το χαρακτήρα δύο καθώς τον ταξινομούσε σε ένα και επιπλέον ταξινομούσε επιτυχώς το χαρακτήρα έξι και επτά. Συνεπώς μόνο το μεγαλύτερο δείγμα δεν εξασφαλίζει καλύτερα αποτελέσματα και αυτό χρήζει περαιτέρω διερεύνηση ώστε να διευκρινιστεί ο λόγος για τον οποίο συμβαίνει.

Κεφάλαιο 5: Σύνοψη

5.1 Μέθοδος

Στόχος της παρούσας πτυχιακής εργασίας ήταν η μελέτη των κατανεμημένων μηχανικών μάθησης και οι εφαρμογές τους. Στα πλαίσια της μελέτης αυτής υλοποιήθηκε ένα σύστημα κατανεμημένης μηχανικής μάθησης σε υπολογιστές τύπου Raspberry PI, υποστηριζόμενους από έναν υπολογιστή. Η ιδέα πίσω από την υλοποίηση αυτή προέρχεται από την εργασία των Tiffany [Tiffany et all, 2018] οι οποίοι εφάρμοσαν την τεχνική κατανεμημένης μηχανικής μάθησης σε υπολογιστές Raspberry οι οποίοι όμως κατευθύνονταν από ένα φορητό υπολογιστή. Οι συσκευές Raspberry προσπαθούσαν να δημιουργήσουν το δικό τους μοντέλο μηχανικής μάθησης λαμβάνοντας πληροφορίες και από τις άλλες συσκευές Raspberry μέσω του φορητού υπολογιστή.

Στη παρούσα προσέγγιση το σκεπτικό είναι ότι οι συσκευές Raspberry, λόγο των περιορισμένων πόρων που διαθέτουν τόσο σε υπολογιστική ισχύ, όσο και σε ενέργεια, καθώς είναι πιθανόν να είναι εγκατεστημένες σε κάποια απομακρυσμένη περιοχή και να μην έχουν πρόσβαση στο δίκτυο ηλεκτρικής ενέργειας, δεν είναι σε θέση να παράξουν το μοντέλο μηχανικής μάθησης. Για το λόγο αυτό το μοντέλο το παράγει μια ισχυρή υπολογιστική δομή που δεν διέπεται από τους ανωτέρω περιορισμούς και ακολούθως το αποστέλλει στις επιμέρους συσκευές IOT για να το εκτελέσουν.

5.2 Υλοποίηση

Το παραγόμενο μοντέλο αφορά ένα σύστημα αναγνώρισης χειρόγραφων ψηφίων από το μηδέν έως και το εννέα τα οποία προέρχονται από το MNIST. Η βάση δεδομένων αποτελείται από

70.000 εικόνες οι οποίες εμφανίζουν τα προαναφερόμενα χειρόγραφα ψηφία. Οι 60.000 εξ αυτών είναι κατηγοριοποιημένες και χρησιμοποιούνται για εκπαίδευση ενώ οι 10.000 χρησιμοποιούνται για τη δοκιμή του μοντέλου που δημιουργείται.

Για τη δημιουργία του μοντέλου χρησιμοποιήθηκε ένας υπολογιστής γραφείου ο οποίος διέθετε επεξεργαστή Intel Core i3 ταχύτητας 2.93Gz, λειτουργικό σύστημα Windows 7 και μνήμη 16GB. Η γλώσσα προγραμματισμού που χρησιμοποιήθηκε ήταν η Python έκδοσης 3.7 πλαισιωμένη με τη βιβλιοθήκη scikit-learn για την υλοποίηση του SVM, ο οποίος είναι ο πλέον κατάλληλος για την περίπτωση, όπως αναφέρθηκε και σε προηγούμενη παράγραφο. Άλλες βιβλιοθήκες που χρησιμοποιήθηκαν είναι η Numpy, η matplotlib και η Pickle. Για τη δημιουργία του μοντέλου από το δείγμα των 60.000 εικόνων ο υπολογιστής χρειάστηκε 4808 δευτερόλεπτα και το μοντέλο που παράχθηκε ήταν μεγέθους 352MB.

Το συγκεκριμένο μοντέλο δεν είναι ασφαλές να αξιοποιηθεί από το Raspberry Pi 3 που χρησιμοποιήθηκε στη παρούσα εργασία και για το λόγο αυτό παράχθηκε από τον υπολογιστή ένα δεύτερο μοντέλο χρησιμοποιώντας μόνο 10.000 εικόνες. Το νέο αυτό μοντέλο είναι μεγέθους 60MB. Στη συνέχεια πραγματοποιήθηκε και μια εκτίμηση της απόδοσης των παραγόμενων μοντέλων μέσω της accuracy score.

Μέγεθος Μοντέλου (Δείγματα)	**Χρόνος Εκτέλεσης Μέτρησης (Δευτερόλεπτα)**	**Αποτέλεσμα**
60.000	933	86%
10.000	153	41%

Πίνακας 5.1: Εκτίμηση μοντέλου μέσω της accuracy score

5.3 Αποτελέσματα

Ο ακόλουθος πίνακας εμφανίζει τα αποτελέσματα των μοντέλων στο Raspberry Pi. Πιο συγκεκριμένα, χρησιμοποιώντας το μοντέλο των 10.000 εικόνων το Raspberry χρειάστηκε 1.3 δευτερόλεπτα για να ταξινομήσει τις εικόνες που του δοθήκαν και κατάφερε να αναγνωρίσει σωστά τις έξι από τις δέκα. Αντίθετα για το μοντέλο που παράχθηκε από το δείγμα των 60.000 χρειάστηκε 7.6 δευτερόλεπτα και κατάφερε να αναγνωρίσει 4 στις 10.

Μέγεθος Μοντέλου (Δείγματα)	Χρόνος Εκτέλεσης Ταξινόμησης (Δευτερόλεπτα)	Αποτέλεσμα
10.000	1.3	60%
60.000	7.6	40%

Πίνακας 5.2: Πραγματική απόδοση του μοντέλου

Κεφάλαιο 6: Επίλογος – Συμπεράσματα

6.1 Στόχος

Στόχος της παρούσας πτυχιακής εργασίας ήταν η μελέτη των κατανεμημένων τεχνικών μηχανικής μάθησης και οι εφαρμογές τους. Για το λόγο αυτό η παρούσα εργασία ξεκινά με μια παρουσίαση της τεχνικής μάθησης τόσο στη παρούσα της μορφή όσο και την εξέλιξης της στο χρόνο. Ακολούθως παρουσιάζονται ζητήματα βελτίωσης της απόδοσης των συστημάτων καθώς και οι σημαντικότεροι αλγόριθμοι στους οποίους βασίζει τη λειτουργία της. Το πρώτο και εισαγωγικό κεφάλαιο κλείνει με μια παρουσίαση των ειδών μηχανικής μάθησης καθώς και με τα περιβάλλοντα της κατανεμημένης της μορφής.

Στο επόμενο κεφάλαιο αναλύονται οι όροι Cloud Computing, Edge Computing, Edge Intelligence καθώς και κάποιες ενδεικτικές εφαρμογές τους στη καθημερινότητα της σύγχρονης ανθρωπότητας. Ακολούθως, στο τρίτο κεφάλαιο παρουσιάζονται αντιπροσωπευτικές εφαρμογές συστημάτων κατανεμημένης μηχανικής μάθησης, μεταξύ αυτών και μία εφαρμογή ανίχνευσης χαρακτήρων σε μια κατανεμημένη αρχιτεκτονική η οποία αποτέλεσε και έμπνευση για την ανάπτυξη μιας αντίστοιχης εφαρμογής στο επόμενο κεφάλαιο της εργασίας.

Στο επόμενο κεφάλαιο παρουσιάζεται η υλοποίηση μιας εφαρμογής μηχανικής μάθησης η οποία περιλαμβάνει την εκπαίδευση δύο μοντέλων μηχανικής μάθησης από μια ισχυρή υπολογιστική δομή και ακολούθως μεταφορά των μοντέλων αυτών στις επιμέρους IOT συσκευές προκειμένου να τα χρησιμοποιήσουν. Ποιο συγκεκριμένα ο υπολογιστής, μέσω της γλώσσας

προγραμματισμού Python και την βιβλιοθήκη scikit-learn, θα εκπαιδεύσει δύο μοντέλα μηχανικής μάθησης να αναγνωρίζουν χειρόγραφους χαρακτήρες που αναπαριστούν τα δεκαδικά ψηφία, χρησιμοποιώντας τα δεδομένα του MNIST. Θα δημιουργηθούν δύο μοντέλα, ένα που θα χρησιμοποιήσει το σύνολο των εικόνων του MNIST και ένα που θα χρησιμοποιήσει μονό τις πρώτες 10.000 εικόνες. Ο λόγος που πραγματοποιείται ο διαχωρισμός αυτός είναι διότι το μοντέλο που χρησιμοποιεί το σύνολο των εικόνων είναι αρκετά μεγάλο και συνεπώς περιορίζει σημαντικά τον αριθμό των διαθέσιμων συσκευών IOT στις οποίες μπορεί να εκτελεστεί. Τέλος τα δύο μοντέλα θα χρησιμοποιηθούν από μια συσκευή Raspberry Pi 3 B+, το οποίο μέσω της Python διαβάζει δέκα εικόνες που εμφανίζουν χειρόγραφους χαρακτήρες και χρησιμοποιώντας το μικρό μοντέλο ταξινομεί επιτυχώς τους έξι ενώ με το μεγάλο τους τέσσερις.

6.2 Συμπεράσματα

Συμπερασματικά θα μπορούσε να καταλήξει ο αναγνώστης ότι το αντικείμενο είναι αρκετά εκτενές και επιπλέον μελέτη απαιτείται για την εξαγωγή ασφαλών συμπερασμάτων. Πέρα από το μεγάλο αριθμό των συσκευών IOT αλλά και των πρόσθετων που θα αναλάβουν την επεξεργασία των δεδομένων ώστε να μπορούν να ανταποκριθούν οι συσκευές, διατίθεται επιπλέον και αρκετός αριθμός βιβλιοθηκών μηχανικής μάθησης για διάφορες πλατφόρμες και γλώσσες προγραμματισμού, είτε δωρεάν, είτε επι πληρωμής. Το σίγουρο είναι ότι το μοντέλο που επικρατούσε το διάστημα συγγραφής της παρούσας πτυχιακής, σύμφωνα με το οποίο οι διάφορες περιφερειακές συσκευές παρήγαγαν δεδομένα και τα απέστελλαν στο σύνολο τους σε μια ισχυρή υπολογιστική δομή σε κάποιο νέφος για επεξεργασία, φαίνεται ότι πλησιάζει προς αντικατάσταση με ένα νέο μοντέλο όπου τα δεδομένα θα επεξεργάζονται στην πηγή και στην ισχυρή δομή θα αποστέλλεται μόνο πληροφορίες που αφορούν τα δεδομένα.

Βιβλιογραφία

A. Jain, M. Murty and P. Flynn. Data Clustering: A Review. ACM Computing Surveys, vol. 31, no. 3, pp. 264-323,1999

A. Jain, R. Dubes. Algorithms for Clustering Data. Prentice Hall, New Jersey, USA, 1988.

A. Jain, R. Duin and J. Mao. Statistical Pattern Recognition: A Review. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no.1, pp. 4-37, 2000.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban
Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch, 2017.

Alexander Sergeev and Mike Del Balso. 2018. Horovod: Fast and easy distributed deep learning in TensorFlow. Retrieved from https://arxiv.org/abs/1802.05799.

Alrawais, A.; Alhothaily, A.; Hu, C.; Cheng, X. Fog computing for the internet of things: Security and privacy issues. IEEE Internet Comput. 2017

Andrew Gibiansky. 2017. Bringing HPC Techniques to Deep Learning. Retrieved from http://research.baidu.com/bringing-hpc-techniques-deep-learning/.

Arm. 2019. Embedded Machine Learning Design for Dummies. John Wiley & Sons, Inc.

Baradwaj BK, Pal S. Mining educational data to analyze students' performance. arXiv preprint arXiv:1201.3417, 2012.

Bonomi, F.; Milito, R.; Natarajan, P.; Zhu, J. Fog computing: A platform for internet of things and analytics. In Big Data and Internet of Things: A Roadmap for Smart Environments; Springer: Berlin/Heidelberg, Germany, 2014; pp. 169–186

Buchanan, B.G. A (Very) Brief History of Artificial Intelligence. AI Mag. 2006, 26, 53–60.

Buhalis, D.; Amaranggana, A. Smart tourism destinations. In Information and Communication Technologies in Tourism 2014; Springer: Berlin/Heidelberg, Germany, 2013; pp. 553–564.

Catarinucci, L.; De Donno, D.; Mainetti, L.; Palano, L.; Patrono, L.; Stefanizzi, M.L.; Tarricone, L. An IoT-aware architecture for smart healthcare systems. IEEE Internet Things J. 2015, 2, 515–526.

CHANDOLA, V., BANERJEE, A. & KUMAR, V. (2009) Anomaly detection: A survey", ACM Computing Surveys (CSUR), vol. 41, no. 3, pp. 1-58.

Chen, D.; Zhao, H. Data security and privacy protection issues in cloud computing. In Proceedings of the 2012 International Conference on Computer Science and Electronics Engineering, ICCSEE 2012, Hangzhou, China, 23–25 March 2012; Volume 1, pp. 647–651

Chen, M.; Li, W.; Hao, Y.; Qian, Y.; Humar, I. Edge cognitive computing based smart healthcare system. Future Gener. Comput. Syst. 2018, 86, 403–411.

Christoph Jan Bartodziej, The Concept Industry 4.0: An Empirical Analysis of Technologies and Applications in Produc- tion Logistics (Springer Fachmedien Wiesbaden GmbH, 2017), DOI: 10.1007/978-3-658-16502-4.

D. Evans, “The Internet of Things: How the next evolution of the Internet is changing everything,” Apr. 2011. [Online]. Available:

https://www.cisco.com/c/dam/en_us/about/ac79/docs/innov/IoT_IBSG_0411FINAL

D. Judd, P. Mckinley and A. Jain. Large-scale Parallel Data Clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 20, no. 8, pp. 871-876, 1998

Dean J, Ghemawat S. Mapreduce: simplified data processing on large clusters. In: Proceedings of the 6th Conference on Symposium on Operating Systems Design and Implementation (OSDI'04). USENIX As- sociation; 2004.

Domingos, P.M. A Few Useful Things to Know About Machine Learning. Commun. ACM 2012, 55, 78–87.

Donghwoon Kwon, Hyunjoo Kim, Jinoh Kim, Sang C. Suh, Ikkyun Kim, and Kuinam J. Kim. 2017. A survey of deep learning-based network anomaly detection. Cluster Comput. (27 Sep. 2017).

E. Alcala, L.Sellart, V. Puig, J. Quevedo, J.Saludes, D.Vazquez, and A.Lopez "Comparison of two non-linear model-based control strategies for autonomous vehicles", in Proc. 2016 MED, Athens, Greece, 2016, pp. 846-851.

El-Sayed, H.; Sankar, S.; Prasad, M.; Puthal, D.; Gupta, A.; Mohanty, M.; Lin, C.T. Edge of things: The big picture on the integration of edge, IoT and the cloud in a distributed computing environment. IEEE Access 2017, 6, 1706–1717.

Elias De Coninck, Steven Bohez, Sam Leroux, Tim Verbelen, Bert Vankeirsbilck, Pieter Simoens, and Bart Dhoedt.
2018. DIANNE: A modular framework for designing, training and deploying deep neural networks on heterogeneous
distributed infrastructure. J. Syst. Softw. 141, 2018, 52–65.

Evans, D. The Internet of Things How the Next Evolution of the Internet Is Changing Everything. CISCO White Pap, 2011.

Garofalakis, G., Komninos, A., Simou, I., Gkorgkolis, N. Performance of Raspberry Pi microclusters for Edge Machine Learning in Tourism. Computer Science, Aml 2019.

G. Premsankar, M. D. Francesco, and Tarik Taleb, "Edge Computing for the Internet of Things: A Case Study," IEEE Internet Things J., vol. 5, no. 2, Apr. 2018

G. Volkan, U. Jumyung, R. Martin "An introduction to Edge Computing and A Real-Time Capable Server Architecture," International Journal on Advances in Intelligent Systems, vol. 11, no. 1&2, pp. 105-115, 2018.

H. Abbas and M. Fahmy. Neural Networks for Maximum Likelihood Clustering.
Signal Processing, vol. 36, no.1, pp. 111-126, 1994

Halevy A, Norvig P, Pereira F. The unreasonable effectiveness of data. IEEE Intell Syst. 2009;24(2): 8–12.

Hamerly and C. Elkan. Alternatives to the K-means Algorithm that Find Better Clusterings. In Proceedings of the ACM Conference on Information and Knowledge Management (CIKM-2002), pp. 600-607, 2002

Hassan, N.; Gillani, S.; Ahmed, E.; Yaqoob, I.; Imran, M. The role of edge computing in internet of things. IEEE Commun. Mag. 2018, 56, 110–115.

J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," Commun. ACM, vol. 51, no. 1, Jan. 2008

J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, ‘‘You only look once: Unified, real-time object detection,’’ in Proc. IEEE Conf. Comput. Vis.Pattern Recognit., Jun. 2016, pp. 779–788.

Janakiram MSV. In the Era of Artificial Intelligence, GPUs Are the New CPUs. Available online: https://www.forbes.com/sites/janakirammsv/2017/08/07/in-the-era-of-artificial-intelligence-gpus-are-the-new-cpus/#78e36b4c5d16.

Jay Lee, Behrad Bagheri, and Hung-An Kao, “A cyber-physical systems architecture for Industry 4.0-based manufacturing systems,” Manufacturing Letters, January 2015, http:// www.sciencedirect.com/science/article/pii/ S221384631400025X.

Jeffrey Dean and Sanjay Ghemawat. 2004. MapReduce: Simplified data processing on large clusters. In Proceedings of the 6th Conference on Operating Systems Design & Implementation, Vol. 6 (OSDI’04). USENIX Association, 10–10.

Jeffrey Dean, Greg S. Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Quoc V. Le, Mark Z. Mao, Marc’Aurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, and Andrew Y. Ng. 2012. Large scale distributed deep networks. In Proceedings of the 25th International Conference on Neural Information Processing Systems, Vol. 1 (NIPS’12). Curran
Associates Inc., 1223–1231. Retrieved from http://dl.acm.org/citation.cfm?id=2999134.2999271.

Joost Verbraeken, Matthijs Wolting, Jonathan Katzy, Jeroen Kloppenburg, Tim Verblelen, Jan S. Pellermeyer, “A Survey on Distributed Machin Learning”. ACM, vol 53, no. 2. March 2020.

Jyoti Nandimath, Ekata Banerjee, Ankur Patil, Pratima Kakade, Saumitra Vaidya, and Divyansh Chaturvedi. 2013. Big data analysis using Apache Hadoop. In Proceedings of the IEEE 14th International Conference on Information Reuse& Integration (IRI’13). IEEE, 700–703.

K. Shvachko, H. Kuang, S. Radia, and R. Chansler, "The hadoop distributed file system," in Proc. MSST, Incline Village, NV, USA, 2010.

Kaggle. Available: www.kaggle.com

Kaz Sato, Cliff Young, and David Patterson. 2017. An in-depth look at Google's first Tensor Processing Unit (TPU). Google Cloud Big Data Mach. Learn. Blog 12, 2017.

Kesavaraj G, Sukumaran S. A study on classification techniques in data mining. in Computing, Communications and Networking Technologies (ICCCNT), 2013 Fourth International Conference on, 2013; pp. 1-7.

L. Eliot, "Edge Computing for AI Self-Driving Cars," Jan. 2018. [Online]. Available: https:// www.aitrends.com/ai- insider/edge-computing-ai-self-driving-cars/

LANE, T. and BRODLY, C. (1997) An Application of Machine Learning to Anomaly Detection. Purdue University, West Lafayette.

Li, H.; Ota, K.; Dong, M. Learning IoT in edge: Deep learning for the Internet of Things with edge computing. IEEE Netw. 2018, 32, 96–101

Luiz André Barroso, Jeffrey Dean, and Urs Hölzle. 2003. Web search for a planet: The Google cluster architecture. IEEE Micro 2 (2003), 22–28.

M. D. Dikaiakos, D. Katsaros, P. Mehra, G. Pallis, and A. Vakali, "Cloud computing: Distributed Internet Computing for IT and Scientific Research," IEEE Internet Comput., vol. 13, no. 5, pp. 11-13, Sep. 2009.

M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: Cluster computing with working sets," in Proc. HotCloud'10, Boston, MA, USA, 2010.

Mahmut Taha Yazici, Shadi Basurra, Mohamed Medhat Gaber. Edge Machine Learning: Enabling Smart Internet of Things Applications. MDPI, Sep. 2018.

Mao, Y.; You, C.; Zhang, J.; Huang, K.; Letaief, K. Mobile edge computing: Survey and research outlook. arXiv 2017, arXiv:1701.01090.

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. Retrieved from https://www.tensorflow.org/.

Mell, P.M.; Grance, T. SP 800-145. The NIST Definition of Cloud Computing; Technical Report; NIST: Gaithersburg, MD, USA, 2011.

Mitchell, G. How Much Data Is on the Internet? Available online: https://www.sciencefocus.com/future-technology/how-much-data-is-on-the-internet/

Mirashe, S.P.; Kalyankar, N.V. Cloud Computing. arXiv 2010, arXiv:1003.4074.

MNIST. http://yann.lecun.com/exdb/mnist/

Newquist, H. The Brain Makers, 1st ed.; Sams: Camel, IN, USA, 1994.

NCTA, “Behind The Numbers: Growth in the Internet of Things, Aug. 2015. [Online]. Available: https://www.ncta.com/whats-new/behind-the-numbers-growth-in-the-internet-of-things-2

NVIDIA Corporation. 2017. Nvidia Tesla V100. Retrieved from https://www.nvidia.com/en-us/data-center/ tesla-v100/

Plastiras, G.; Terzi, M.; Kyrkou, C.; Theocharidcs, T. Edge Intelligence: Challenges and Opportunities of Near-Sensor Machine Learning Applications. In Proceedings of the 2018 IEEE 29th International Conference on Application-Specific Systems, Architectures and Processors (ASAP), Milano, Italy, 10–12 July 2018; pp. 1–7

ROI, Inside the smart factory: How the Internet of Things is transforming manufacturing, http://www.roi-international.com/fileadmin/ROI_DIALOG/ab_DIALOG_44/EN-ROI-DIALOG-49_web.pdf, accessed August 18, 2017.

S. Ghemawat, H. Gobioff, and S.-T. Leung, “The Google file system,” ACM SIGOPS, Oper. Syst. Rev., vol. 37, no. 5, pp. 30–42, 2003.

Shi, W.; Cao, J.; Zhang, Q.; Li, Y.; Xu, L. Edge computing: Vision and challenges. IEEE Internet Things J. 2016, 3, 637–646.

Shi, W.; Dustdar, S. The promise of edge computing. Computer 2016, 49, 78–81.

Tianqi Chen, Mu Li, Yutian Li, Min Lin, Naiyan Wang, Minjie Wang, Tianjun Xiao, Bing Xu, Chiyuan Zhang, and Zheng Zhang. 2015. MXNet: A flexible and efficient machine learning

library for heterogeneous distributed systems. CoRR abs/1512.01274 (2015). arxiv:1512.01274 http://arxiv.org/abs/1512.01274

Tiffany Tuor, Shiqiang Wang, Theodoros Salonidis, Bong Jun Ko, Kin K. Leung. 2018. Demo Abstract: Distributed Machine Learning at Resource-Limited Edge Nodes. IEEE Conference on Computer Communications Poster and Demo. INFOCOM 2018.

V. Gezer, J. Um, and M. Ruskowski, "An Extensible Edge Computing Architecture: Definition, Requirements and Enablers," in Proc. UBICOMM-2017, Barcelona, Spain, 2017.

Vittorio, M., Allem, K., Francesco, S., Marcello, C. 2019. Real-Time Apple Detection System Using Embedded Systems With Hardware Accelerators: An Edge AI Application. IEEE, Jan 2020.

William D. Gropp, William Gropp, Ewing Lusk, and Anthony Skjellum. 1999. Using MPI: Portable Parallel Program- ming with the Message-passing Interface. Vol. 1. The MIT Press.

Yi, S.; Qin, Z.; Li, Q. Security and privacy issues of fog computing: A survey. In Proceedings of the International Conference on Wireless Algorithms, Systems, and Applications, Qufu, China, 10–12 August 2015; pp. 685–695.