ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ

Τμήμα Μηχανικών Βιομηχανικής Σχεδίασης & Παραγωγής

Αναγνώριση Δραστηριότητας με Βαθιά Μάθηση

Διπλωματική Εργασία

από:

Κοτσίκος Σπυρίδων

Επιβλέπων Καθηγητής:

Νικολάου Γρηγόρης

Αθήνα, Ιούνιος 2023

Επιτροπή Αξιολόγησης Διπλωματικής Εργασίας

Ονοματεπώνυμο	Βαθμίδα	Ψηφιακή Υπογραφή
Νικολάου Γρηγόρης	Λέκτορας	GRIGORIOS NIKOLAOU Digitally signed by GRIGORIOS NIKOLAOU Date: 2023.07.21 12:08:03 +03'00'
Βασιλειάδου Σουλτάνα	Επίκουρη Καθηγήτρια	Soultana Vasileiadou Digitally signed by Soultana Vasileiadou Date: 2023.07.25 08:56:19 +03'00'
Δρόσος Χρήστος	ΕΔΙΠ	Christos Drosos Digitally signed by Christos Drosos Date: 2023.07.23 15:36:27 +03'00'

Δήλωση Συγγραφέα Διπλωματικής Εργασίας

Ο κάτωθι υπογεγραμμένος Κοτσίκος Σπυρίδων του Κωνσταντίνου, με αριθμό μητρώου 18389039, φοιτητής του Πανεπιστημίου Δυτικής Αττικής της Σχολής Μηχανικών του Τμήματος Μηχανικών Βιομηχανικής σχεδίασης και παραγωγής, δηλώνω υπεύθυνα ότι:

«Είμαι συγγραφέας αυτής της διπλωματικής εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, οι όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε ακριβώς είτε παραφρασμένες, αναφέρονται στο σύνολό τους, με πλήρη αναφορά στους συγγραφείς, τον εκδοτικό οίκο ή το περιοδικό, συμπεριλαμβανομένων και των πηγών που ενδεχομένως χρησιμοποιήθηκαν από το διαδίκτυο.

Επίσης, βεβαιώνω ότι αυτή η εργασία έχει συγγραφεί από μένα αποκλειστικά και αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο δικής μου, όσο και του Ιδρύματος. Παράβαση της ανωτέρω ακαδημαϊκής μου ευθύνης αποτελεί ουσιώδη λόγο για την ανάκληση του πτυχίου μου».

Ο Δηλών:

Κοτσίκος Σπυρίδων

Περίληψη

Ο σκοπός αυτής της διπλωματικής εργασίας με τίτλο "Αναγνώριση Δραστηριότητας με Βαθιά Μάθηση" είναι να διερευνήσει και να αναπτύξει ένα σύστημα βασισμένο στη βαθιά μάθηση για την ακριβή ανίχνευση, ανάλυση και ταξινόμηση ανθρώπινων δραστηριοτήτων, χρησιμοποιώντας δεδομένα από διάφορους αισθητήρες. Η διατριβή, πιο συγκεκριμένα, στοχεύει να διερευνήσει τις δυνατότητες των αλγορίθμων βαθιάς μάθησης στην αντιμετώπιση των προκλήσεων της αναγνώρισης ανθρώπινης δραστηριότητας (Human Activity Recognition), καθώς και να παρουσιάσει ένα μοντέλο βαθιάς μάθησης, εφαρμοσμένο σε έναν μικροελεγκτή Arduino, το οποίο κατηγοριοποιεί συγκεκριμένες δραστηριότητες σε πραγματικό χρόνο.

Το πρώτο μέρος της διατριβής περιλαμβάνει μία σύντομη αναφορά στην έννοια και σημασία της τεχνητής νοημοσύνης και μηχανικής μάθησης, αλλά και έναν σαφή ορισμό της βαθιάς μάθησης, επισημαίνοντας τον τρόπο και τις τεχνικές για την εφαρμογή της, σε οποιοδήποτε σύστημα.

Το δεύτερο μέρος της διατριβής αποτελείται από την αναλυτική παρουσίαση των πλακετών Arduino. Πιο συγκεκριμένα, δίνεται έμφαση στον μικροελεγκτή που χρησιμοποιείται (Arduino Nano 33 BLE Sense) και τις δυνατότητές του για την ανάπτυξη μοντέλων βαθιάς μάθησης, με δεδομένα από τους αισθητήρες που αυτός φέρει.

Το τελευταίο μέρος της διατριβής επικεντρώνεται στην ανάπτυξη και βελτιστοποίηση ενός μοντέλου αναγνώρισης δραστηριότητας βασισμένου σε βαθιά μάθηση, το οποίο θα περιλαμβάνει την ακριβή μεθοδολογία της υλοποίησης του. Αυτό θα γίνει δίνοντας έμφαση στον τρόπο συλλογής και προεπεξεργασίας δεδομένων, την εξαγωγή χαρακτηριστικών και την εκπαίδευση μοντέλων, χρησιμοποιώντας ένα ικανοποιητικά μεγάλο σύνολο δεδομένων ανθρώπινων δραστηριοτήτων που συλλέγονται από έναν μικροελεγκτή Arduino, με την βοήθεια του επιταχυνσιομέτρου και του γυροσκοπίου της ίδιας της πλακέτας. Επίσης, περιλαμβάνεται το μοντέλο εφαρμοσμένο στην πλακέτα για την κατηγοριοποίηση των δεδομένων, σε πραγματικό χρόνο, στις αντίστοιχες δραστηριότητες.

Το προτεινόμενο μοντέλο HAR που βασίζεται σε βαθιά μάθηση αξιολογείται τελικά χρησιμοποιώντας πειράματα και μετρήσεις απόδοσης για την αξιολόγηση της ακρίβειας και της αποτελεσματικότητας του.

Τα αποτελέσματα αυτής της διπλωματικής μπορούν να έχουν πρακτικές εφαρμογές σε τομείς όπως η υγειονομική περίθαλψη, ο αθλητισμός και τα έξυπνα σπίτια, όπου η ακριβής και σε πραγματικό χρόνο αναγνώριση της ανθρώπινης δραστηριότητας είναι ζωτικής σημασίας για εξατομικευμένες εφαρμογές και υπηρεσίες.

Λέξεις κλειδιά:

Αναγνώριση Δραστηριότητας, Βαθιά Μάθηση, Νευρωνικά Δίκτυα, Python, Κατηγοριοποίηση, Arduino, Αισθητήρες (Επιταχυνσιόμετρο, Γυροσκόπιο)

Abstract

The purpose of this thesis entitled "Activity Recognition with Deep Learning" is to investigate and develop a deep learning-based system to accurately detect, analyze and classify human activities using data from various sensors. More specifically, the thesis aims to explore the potential of deep learning algorithms in addressing the challenges of Human Activity Recognition (HAR), and to present a deep learning model, implemented on an Arduino microcontroller, that categorizes specific activities in real-time.

The first part of the thesis includes a brief reference to the concept and importance of artificial intelligence and machine learning, as well as a clear definition of deep learning, pointing out the ways and the techniques to apply it, in any system.

The second part of the thesis consists of a detailed presentation of the Arduino boards. More specifically, emphasis is placed on the microcontroller used (Arduino Nano 33 BLE Sense) and its capabilities to develop deep learning models, with data from the sensors it carries.

The last part of the thesis focuses on the development and optimization of a deep learning-based activity recognition model, which will include the exact methodology of its implementation. This will be done by focusing on how to collect and pre-process data, extract features and train models using a sufficiently large human activity dataset collected by an Arduino microcontroller, with the help of the accelerometer and gyroscope of the board itself. The model implemented on the board to categorize the data, in real time, into the respective activities, is also included.

The proposed deep learning based HAR model is finally evaluated using experiments and performance measurements to assess its accuracy and effectiveness.

The results of this thesis may have practical applications in areas such as healthcare, sports and smart homes, where accurate and real-time recognition of human activity is crucial for personalized applications and services.

Key Words:

Activity Recognition, Deep Learning, Neural Networks, Python, Classification, Arduino, Sensors (Accelerometer, Gyroscope)

Ευχαριστίες

Η ολοκλήρωση της παρούσας διπλωματικής εργασίας δεν θα ήταν δυνατή χωρίς την πολύτιμη υποστήριξη και καθοδήγηση πολλών ατόμων, στους οποίους θα ήθελα να εκφράσω τις ευχαριστίες μου.

Πρώτων, θα ήθελα να εκφράσω τις ειλικρινείς ευχαριστίες μου στον επιβλέποντα καθηγητή μου, Γρηγόρη Νικολάου, για τη συνεχή καθοδήγηση και την τεχνογνωσία καθ' όλη τη διάρκεια της ερευνητικής διαδικασίας. Η υποστήριξή του, αλλά και οι πολύτιμες γνώσεις που μου προσέφερε συνέβαλαν καθοριστικά στη διαμόρφωση αυτής της διατριβής.

Επιπλέον, θα ήθελα να εκφράσω την εκτίμησή μου στην οικογένεια και τους φίλους μου για την ενθάρρυνση καθώς και την κατανόηση τους, όλο αυτό το διάστημα. Η συνεχής υποστήριξή τους και η πίστη τους στις ικανότητές μου αποτέλεσαν την κινητήρια δύναμη πίσω από την ολοκλήρωση των ακαδημαϊκών μου υποχρεώσεων.

Τέλος, θα ήθελα να ευχαριστήσω όλους τους συμμετέχοντες που με προθυμία προσέφεραν τον χρόνο τους για να συμμετάσχουν στη διαδικασία συλλογής δεδομένων για την παρούσα διπλωματική εργασία. Η συνεργασία και η συμμετοχή τους ήταν καθοριστικής σημασίας για την επίτευξη ουσιαστικών αποτελεσμάτων της.

Πίνακας περιεχομένων

Κατάλογος Εικόνων

Εισαγωγή

Η τεχνητή νοημοσύνη (AI) αναφέρεται στην ικανότητα των υπολογιστών να εκτελούν εργασίες που συνήθως απαιτούν ανθρώπινη νοημοσύνη, όπως μάθηση, συλλογισμός και λήψη αποφάσεων. Έχει γίνει μια επαναστατική τεχνολογία με ποικίλες εφαρμογές σε διάφορους τομείς, όπως η υγειονομική περίθαλψη, τα οικονομικά, οι μεταφορές και η ψυχαγωγία, μεταξύ άλλων. Με τις εξελίξεις στους αλγόριθμους μηχανικής μάθησης και την αυξημένη υπολογιστική ισχύ, έχει τη δυνατότητα να εξελίξει τις βιομηχανίες και να βελτιώσει την αποτελεσματικότητα με τρόπους που δεν ήταν εφικτοί πριν.

Μια σημαντική εφαρμογή της τεχνητής νοημοσύνης είναι η αναγνώριση ανθρώπινης δραστηριότητας (HAR), η οποία περιλαμβάνει τη χρήση αλγορίθμων AI για αυτόματη ανίχνευση, ανάλυση και ταξινόμηση ανθρώπινων δραστηριοτήτων με βάση δεδομένα από διάφορους αισθητήρες, όπως επιταχυνσιόμετρα, γυροσκόπια και κάμερες. Το HAR έχει κερδίσει ευρεία προσοχή σε τομείς όπως η υγειονομική περίθαλψη, ο αθλητισμός και τα έξυπνα σπίτια, όπου η κατανόηση των ανθρώπινων δραστηριοτήτων και συμπεριφορών μπορεί να προσφέρει πολύτιμες γνώσεις για τη βελτίωση των αποτελεσμάτων υγείας, τη βελτιστοποίηση της αθλητικής απόδοσης και τη βελτίωση των συστημάτων οικιακού αυτοματισμού. Τα συστήματα HAR με τεχνητή νοημοσύνη μπορούν να αναγνωρίσουν με ακρίβεια δραστηριότητες όπως περπάτημα, τρέξιμο, κάθισμα, ορθοστασία, ακόμη και περίπλοκες δραστηριότητες όπως αθλήματα, ανοίγοντας ευκαιρίες για εξατομικευμένες εφαρμογές και υπηρεσίες που καλύπτουν μεμονωμένες ανάγκες και προτιμήσεις.

Μια ειδική αναφορά στην αναγνώριση ανθρώπινης δραστηριότητας με χρήση τεχνητής νοημοσύνης μπορεί να βρεθεί στον κλάδο της υγειονομικής περίθαλψης, όπου έχει τεράστιες δυνατότητες να φέρει επανάσταση στην παρακολούθηση και τη φροντίδα των ασθενών. Για παράδειγμα, τα συστήματα HAR που λειτουργούν με τεχνητή νοημοσύνη μπορούν να χρησιμοποιηθούν για την παρακολούθηση των δραστηριοτήτων ηλικιωμένων ή χρονίως πασχόντων ασθενών στα σπίτια τους για την ανίχνευση πτώσεων, την παρακολούθηση της τήρησης των φαρμάκων και την παροχή έγκαιρων ειδοποιήσεων για πιθανούς κινδύνους για την υγεία. Το HAR μπορεί επίσης να χρησιμοποιηθεί σε περιβάλλοντα αποκατάστασης για να παρακολουθεί την πρόοδο των ασθενών που αναρρώνουν από τραυματισμούς ή

χειρουργικές επεμβάσεις και να παρέχει εξατομικευμένη ανατροφοδότηση για τη βελτιστοποίηση της διαδικασίας ανάρρωσής τους. Επιπλέον, στον αθλητισμό και τη φυσική κατάσταση, το HAR που λειτουργεί με AI μπορεί να χρησιμοποιηθεί για την παρακολούθηση της απόδοσης των αθλητών, την παρακολούθηση του προπονητικού φόρτου και την πρόληψη τραυματισμών αναλύοντας τις κινήσεις και τις δραστηριότητές τους.

Η εφαρμογή της τεχνητής νοημοσύνης στην αναγνώριση της ανθρώπινης δραστηριότητας έχει τη δυνατότητα να βελτιώσει σημαντικά την ποιότητα των υπηρεσιών υγειονομικής περίθαλψης, να βελτιώσει τα αποτελέσματα των ασθενών και να βοηθήσει τα άτομα να ζήσουν πιο υγιεινά.

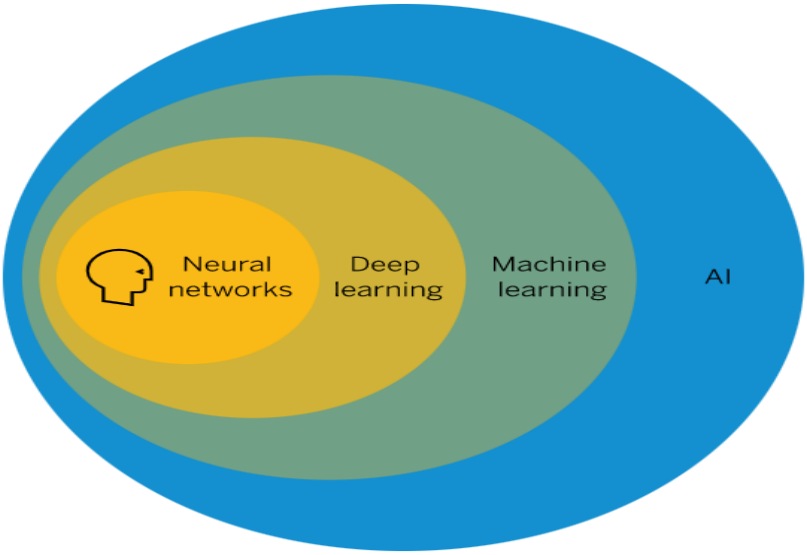
Κεφάλαιο 1: Τεχνητή Νοημοσύνη και Βαθιά Μάθηση

1.1 Εισαγωγή στην Τεχνητή Νοημοσύνη

Η τεχνητή νοημοσύνη (ΤΝ) αποτελεί μία επανάσταση στη σημερινή τεχνολογία. Ο όρος αναφέρεται στην ικανότητα των μηχανών να αναπαράγουν τις ανθρώπινες γνωστικές λειτουργίες, όπως η μάθηση, ο προγραμματισμός και η δημιουργικότητα. Η τεχνητή νοημοσύνη επιτρέπει στις μηχανές να ¨κατανοούν¨ το περιβάλλον τους, να επιλύουν προβλήματα και να ενεργούν προς την κατεύθυνση συγκεκριμένων στόχων.

Σε πιο δύσκολα προβλήματα περνάμε από τους απλούς αλγόριθμους σε μεγαλύτερους που μπορούν να ικανοποιήσουν τις παραπάνω απαιτήσεις. Ο υπολογιστής σε αυτές τις περιπτώσεις λαμβάνει δεδομένα, ήδη προετοιμασμένα ή συλλεγμένα μέσω αισθητήρων, τα επεξεργάζεται και ανταποκρίνεται, με βάση την ¨εμπειρία¨ του, για την πιο αποτελεσματική επίλυσή τους.

Ξεπερνώντας, λοιπόν, τους αρχικούς απλοϊκούς τρόπους, για την επίτευξης της τεχνητής νοημοσύνης η τεχνολογία εξετάζει πλέον νέες μεθόδους. Στις μεθόδους αυτές ανήκουν η μηχανική μάθηση και βαθιά μάθηση, η οποία τροφοδοτείται από τεχνητά νευρωνικά δίκτυα.



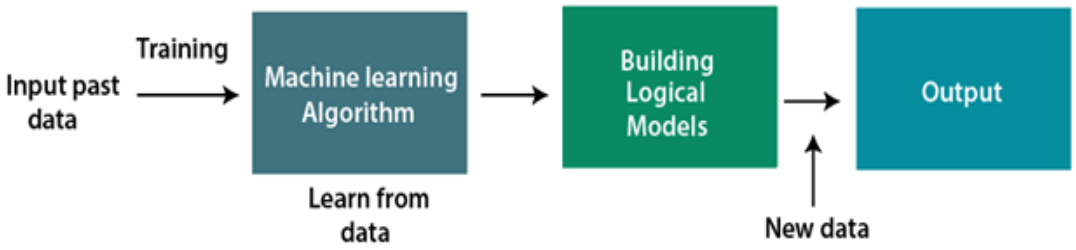
Εικόνα 1: Τεχνητή Νοημοσύνη και Υποσύνολα

1.2 Μηχανική Μάθηση

Η Μηχανική Μάθηση είναι ένα αναπτυσσόμενο πεδίο σπουδών στην επιστήμη των υπολογιστών που διερευνά το σχεδιασμό και την κατασκευή αλγορίθμων που μπορούν να μάθουν από δεδομένα και να κάνουν προβλέψεις για αυτά. Χρησιμοποιείται σε μια σειρά εργασιών όπου δεν είναι δυνατός ο ρητός προγραμματισμός αλγορίθμων. Οι αλγόριθμοι στη μηχανική μάθηση δεν λένε ακριβώς στον υπολογιστή τι να κάνει, αλλά βασίζονται σε μοντέλα που έχουν δημιουργηθεί από πειραματικά δεδομένα προκειμένου να κάνουν προβλέψεις ή να εξαγάγουν αποφάσεις.

Η Μηχανική Μάθηση ορίζεται ως ένα υποσύνολο της τεχνητής νοημοσύνης που ασχολείται κυρίως με την ανάπτυξη αλγορίθμων που επιτρέπουν στον υπολογιστή να μαθαίνει μόνος του από τα δεδομένα και τις προηγούμενες εμπειρίες. Με τη βοήθεια δειγμάτων ιστορικών δεδομένων, τα οποία είναι γνωστά ως δεδομένα εκπαίδευσης, οι αλγόριθμοι μηχανικής μάθησης κατασκευάζουν ένα μαθηματικό μοντέλο που βοηθά στη λήψη προβλέψεων ή αποφάσεων. Συνδυάζει άρα την επιστήμη των υπολογιστών και την στατιστική για τη δημιουργία προγνωστικών μοντέλων.

Η μηχανική μάθηση κατασκευάζει ή χρησιμοποιεί τους αλγόριθμους που μαθαίνουν από ιστορικά δεδομένα. Όσο περισσότερες πληροφορίες παρέχουμε, τόσο υψηλότερη θα είναι η απόδοση. Ένα μηχάνημα έχει τη δυνατότητα να μάθει εάν μπορεί να βελτιώσει την απόδοσή του λαμβάνοντας περισσότερα δεδομένα.



Εικόνα 2: Πώς λειτουργεί η Μηχανική Μάθηση

1.2.1 Μέθοδοι Μηχανικής Μάθησης

Το κάθε πρόβλημα χρειάζεται και διαφορετικό τρόπο επίλυσης. Για τον λόγο αυτό, υπάρχουν τρεις κύριοι τύποι μηχανικής μάθησης. Αυτές είναι η εποπτευόμενη μάθηση, η μάθηση χωρίς επίβλεψη και η ενισχυτική μάθηση.

Η εποπτευόμενη ή επιβλεπόμενη μάθηση περιλαμβάνει την εκπαίδευση ενός μοντέλου σε δεδομένα με ετικέτα, όπου ο αλγόριθμος παρέχεται με εισόδους και αντίστοιχες εξόδους. Σε αλγόριθμους εποπτευόμενης μάθησης, η μηχανή μαθαίνει με το παράδειγμα. Τα μοντέλα εποπτευόμενης μάθησης αποτελούνται από ζεύγη "εισόδου" και "εξόδου", όπου η έξοδος επισημαίνεται με την επιθυμητή τιμή. Μέσω ενός αλγορίθμου, το σύστημα συλλέγει όλα αυτά τα δεδομένα εκπαίδευσης με την πάροδο του χρόνου και αρχίζει να καθορίζει τις συσχετίσεις, τις διαφορές και άλλα σημεία της λογικής, μέχρι να μπορεί να προβλέψει τις απαντήσεις για τις αντίστοιχες ερωτήσεις.

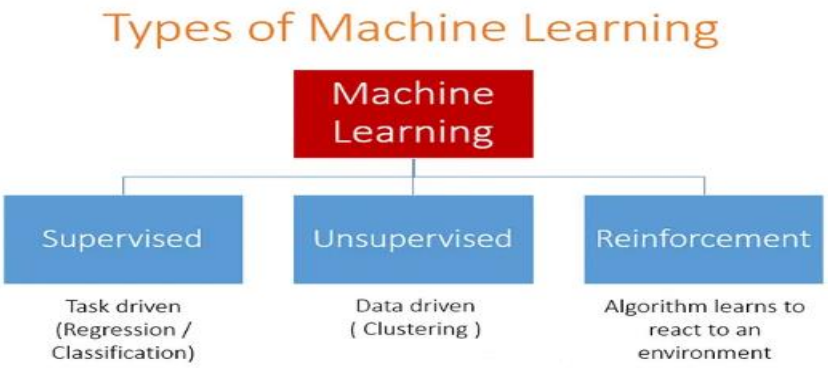
- Η επιβλεπόμενη μάθηση χρησιμοποιείται σε προβλήματα Ταξινόμησης (Classification), Πρόγνωσης (Prediction) και Διερμηνείας (Interpretation).

Αντίθετα, η μάθηση χωρίς επίβλεψη περιλαμβάνει την εύρεση προτύπων σε δεδομένα χωρίς ετικέτα, όπου ο αλγόριθμος προσδιορίζει τη δομή και τις σχέσεις μέσα στα δεδομένα. Η μάθηση χωρίς επίβλεψη είναι ένας τρόπος να μαθαίνεις πράγματα χωρίς να σου λένε τι να κάνεις. Ο υπολογιστής εξετάζει τα δεδομένα, τα οποία είναι συχνά χωρίς ετικέτα και χωρίς δομή, και αρχίζει να βλέπει μοτίβα και σχέσεις. Αυτό είναι παρόμοιο με το πώς μαθαίνουν οι άνθρωποι παρατηρώντας τον κόσμο γύρω τους, Χρησιμοποιώντας τη διαίσθησή και την εμπειρία για την ομαδοποίηση πραγμάτων.

- Η μη επιβλεπόμενη μάθηση χρησιμοποιείται σε προβλήματα Ανάλυσης Συσχετισμών (Association Analysis) και Ομαδοποίησης (Clustering).

Τέλος, η ενισχυτική μάθηση περιλαμβάνει την εκπαίδευση ενός μοντέλου για τη λήψη αποφάσεων με βάση τα σωστά ή τα σφάλματα που λαμβάνονται από ένα περιβάλλον, όπου ο αλγόριθμος μαθαίνει μέσω δοκιμής και λάθους. Το μοντέλο ενισχυτικής μάθησης βοηθά τις μηχανές να μάθουν παρατηρώντας πόσο επιτυχείς είναι στην επίτευξη συγκεκριμένων στόχων. Στις περισσότερες περιπτώσεις, αυτό γίνεται με παράδειγμα, δηλαδή βλέποντας ποιες ενέργειες λειτουργούν και ποιες όχι.

- Η ενισχυτική μάθηση χρησιμοποιείται σε προβλήματα Σχεδιασμού (Planning), όπως η βελτιστοποίηση εργασιών ή ο έλεγχος κίνησης ρομπότ, για παράδειγμα.



Εικόνα 3:Τύποι Μηχανικής Μάθησης

Η κατανόηση των διαφορών μεταξύ αυτών των τύπων μηχανικής μάθησης είναι ζωτικής σημασίας για την επιλογή της κατάλληλης προσέγγισης για ένα δεδομένο πρόβλημα.

1.3 Βαθιά Μάθηση

Η βαθιά μάθηση είναι ένα υποπεδίο της μηχανικής μάθησης που περιλαμβάνει τη δημιουργία τεχνητών νευρωνικών δικτύων με πολλαπλά επίπεδα για τη μοντελοποίηση και επίλυση σύνθετων προβλημάτων. Ο όρος "βαθιά" στη βαθιά μάθηση αναφέρεται στο γεγονός ότι αυτά τα νευρωνικά δίκτυα διαθέτουν πολλαπλά κρυφά στρώματα, τα οποία τους επιτρέπουν να μαθαίνουν όλο και πιο αφηρημένες και σύνθετες αναπαραστάσεις δεδομένων καθώς οι πληροφορίες ρέουν μέσω του δικτύου.

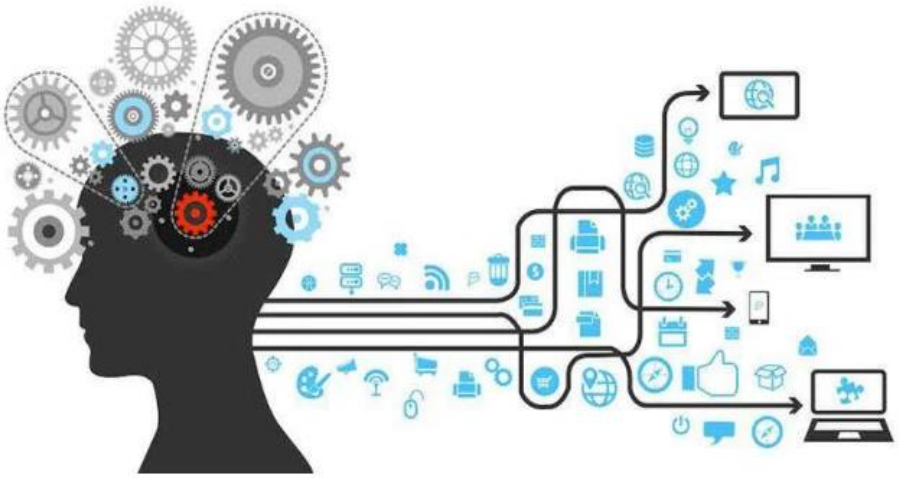
Η βαθιά μάθηση έχει γίνει όλο και πιο δημοφιλής τα τελευταία χρόνια, χάρη στην ανάπτυξη ισχυρών υπολογιστικών πόρων και στη διαθεσιμότητα μεγάλων ποσοτήτων δεδομένων. Με τον τρόπο αυτό, έχει επιτύχει αξιοσημείωτα αποτελέσματα σε ένα ευρύ φάσμα εφαρμογών, συμπεριλαμβανομένης της επεξεργασίας φυσικής γλώσσας, της αναγνώρισης εικόνας, ομιλίας, αλλά και δραστηριότητας.

Στον πυρήνα της, η βαθιά μάθηση περιλαμβάνει τη χρήση ενός συνόλου δεδομένων εκπαίδευσης για την εκπαίδευση ενός νευρωνικού δικτύου. Κατά τη διάρκεια της διαδικασίας εκπαίδευσης, το δίκτυο προσαρμόζει τα βάρη (weight) και τις προκαταλήψεις (bias) των νευρώνων του ως απάντηση στα δεδομένα εισόδου, προκειμένου να ελαχιστοποιήσει μια δεδομένη συνάρτηση απωλειών. Αυτή η διαδικασία προσαρμογής των βαρών και των προκαταλήψεων επιτυγχάνεται συχνά με τη χρήση ενός αλγορίθμου που ονομάζεται στοχαστική κάθοδος κλίσης.

Μόλις εκπαιδευτεί ένα μοντέλο βαθιάς μάθησης, μπορεί να χρησιμοποιηθεί για να κάνει προβλέψεις σε νέα δεδομένα. Η διαδικασία αυτή είναι γνωστή ως συμπερασματολογία. Κατά τη διάρκεια της εξαγωγής συμπερασμάτων, τα δεδομένα εισόδου περνούν από το νευρωνικό δίκτυο και η έξοδος του τελευταίου στρώματος χρησιμοποιείται για να γίνει μια πρόβλεψη.

Ένα από τα βασικά πλεονεκτήματα της βαθιάς μάθησης είναι η ικανότητά της να μαθαίνει ιεραρχικές αναπαραστάσεις δεδομένων. Χτίζοντας όλο και πιο σύνθετες αναπαραστάσεις των δεδομένων σε κάθε επίπεδο του νευρωνικού δικτύου, τα μοντέλα βαθιάς μάθησης είναι σε θέση να συλλάβουν και να μοντελοποιήσουν σύνθετα μοτίβα στα δεδομένα που θα ήταν δύσκολο να ανακαλυφθούν με τη χρήση παραδοσιακών τεχνικών μηχανικής μάθησης.

Συνοπτικά, αξιοποιώντας μεγάλες ποσότητες δεδομένων και ισχυρούς υπολογιστικούς πόρους, η βαθιά μάθηση έχει επιτύχει αξιοσημείωτα αποτελέσματα σε ένα ευρύ φάσμα εφαρμογών και συνεχίζει να αποτελεί ενεργό τομέα έρευνας και ανάπτυξης.



1.4 Σύγκριση Μηχανικής και Βαθιάς Μάθησης

Υπάρχουν πολλές διαφορές μεταξύ της Μηχανικής Μάθησης και της Βαθιάς Μάθησης. Μερικές από τις σημαντικότερες αποτελούν οι εξής:

1. Ανθρώπινη παρέμβαση

Η μηχανική μάθηση απαιτεί περισσότερη συνεχή ανθρώπινη παρέμβαση για την επίτευξη αποτελεσμάτων, ενώ η βαθιά μάθηση είναι πιο πολύπλοκη στη δημιουργία, αλλά απαιτεί ελάχιστη παρέμβαση στη συνέχεια.

2. Υλικό (Hardware)

Τα προγράμματα μηχανικής μάθησης τείνουν να είναι λιγότερο πολύπλοκα από τους αλγορίθμους βαθιάς μάθησης και μπορούν συχνά να εκτελούνται σε συμβατικούς υπολογιστές, αλλά τα συστήματα βαθιάς μάθησης απαιτούν πολύ πιο ισχυρό υλικό και πόρους. Αυτή η απαίτηση για ισχύ έχει οδηγήσει σε αυξημένη χρήση μονάδων γραφικής επεξεργασίας.

3. Χρόνος

Τα συστήματα μηχανικής μάθησης μπορούν να δημιουργηθούν και να λειτουργήσουν γρήγορα, αλλά μπορεί να είναι περιορισμένα ως προς την ισχύ των αποτελεσμάτων τους. Αντίθετα, τα συστήματα βαθιάς μάθησης χρειάζονται περισσότερο χρόνο για να δημιουργηθούν, αλλά μπορούν να παράγουν αποτελέσματα άμεσα (αν και η ποιότητα είναι πιθανό να βελτιωθεί με την πάροδο του χρόνου, καθώς θα διατίθενται περισσότερα δεδομένα).

4. Προσέγγιση

Η μηχανική μάθηση τείνει να απαιτεί δομημένα δεδομένα και χρησιμοποιεί παραδοσιακούς αλγορίθμους όπως η γραμμική παλινδρόμηση. Η βαθιά μάθηση

χρησιμοποιεί νευρωνικά δίκτυα και είναι κατασκευασμένη για να δέχεται μεγάλους όγκους μη δομημένων δεδομένων.

5. Εφαρμογές

Η μηχανική μάθηση χρησιμοποιείται ήδη στα εισερχόμενα του ηλεκτρονικού ταχυδρομείου σας, στην τράπεζα και στο ιατρείο σας. Η τεχνολογία της βαθιάς μάθησης επιτρέπει πιο σύνθετα και αυτόνομα προγράμματα, όπως αυτοκινούμενα αυτοκίνητα ή ρομπότ που εκτελούν προηγμένες χειρουργικές επεμβάσεις.

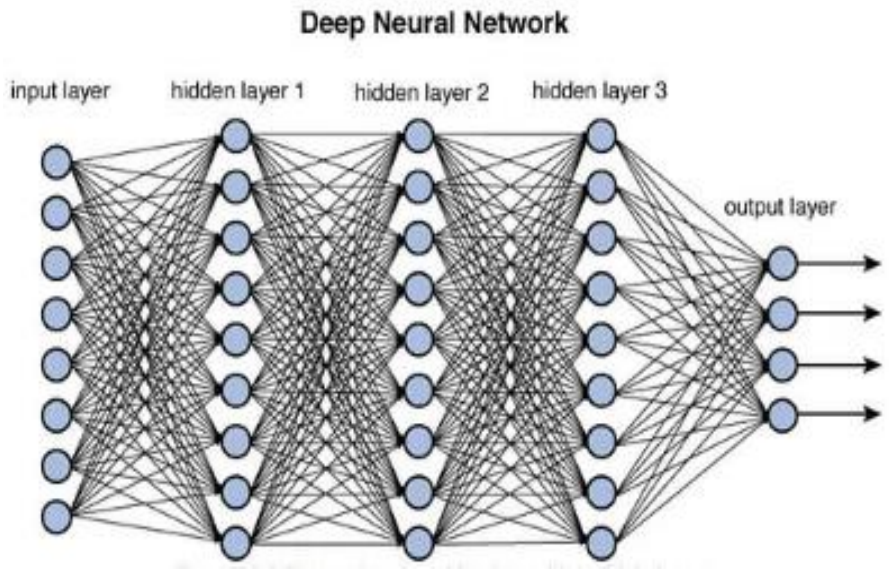
1.5 Πώς Λειτουργεί η Βαθιά Μάθηση

Η βαθιά μάθηση βασίζεται στην έννοια των νευρωνικών δικτύων, τα οποία έχουν ως πρότυπο τη δομή και τη λειτουργία του ανθρώπινου εγκεφάλου. Στο παρακάτω κείμενο, εξηγείται λεπτομερώς πώς λειτουργεί η βαθιά μάθηση, βήμα προς βήμα.

Το πρώτο βήμα στη βαθιά μάθηση είναι η προετοιμασία των δεδομένων. Αυτό περιλαμβάνει τη συλλογή και τον καθαρισμό των δεδομένων, τη μετατροπή τους σε αριθμητική μορφή και τον διαχωρισμό τους σε σύνολα εκπαίδευσης (train), επικύρωσης (validation) και δοκιμής (test). Το σύνολο εκπαίδευσης χρησιμοποιείται για την εκπαίδευση του νευρωνικού δικτύου, ενώ το σύνολο επικύρωσης χρησιμοποιείται για τη βελτιστοποίηση των υπερπαραμέτρων του μοντέλου. Τέλος, το σύνολο δοκιμής χρησιμοποιείται για την αξιολόγηση της απόδοσης του τελικού μοντέλου.

Το επόμενο βήμα είναι ο σχεδιασμός της αρχιτεκτονικής του νευρωνικού δικτύου. Αυτό περιλαμβάνει την επιλογή του τύπου και του αριθμού των στρωμάτων, του αριθμού των νευρώνων σε κάθε στρώμα, των συναρτήσεων ενεργοποίησης και του αλγορίθμου βελτιστοποίησης. Ο πιο συνηθισμένος τύπος νευρωνικού δικτύου που χρησιμοποιείται στη βαθιά μάθηση είναι το νευρωνικό δίκτυο τροφοδότησης, το οποίο αποτελείται από ένα στρώμα εισόδου, ένα ή περισσότερα κρυφά στρώματα και ένα στρώμα εξόδου.

Αφού σχεδιαστεί η αρχιτεκτονική του νευρωνικού δικτύου, το επόμενο βήμα είναι το πέρασμα των δεδομένων εισόδου μέσω του δικτύου για την παραγωγή μιας εξόδου. Στο στρώμα εισόδου, κάθε νευρώνας λαμβάνει ένα χαρακτηριστικό των δεδομένων εισόδου. Στα κρυφά στρώματα, κάθε νευρώνας υπολογίζει ένα σταθμισμένο άθροισμα των εισόδων, εφαρμόζει μια συνάρτηση ενεργοποίησης και μεταβιβάζει την έξοδο στο επόμενο στρώμα. Στο στρώμα εξόδου, κάθε νευρώνας παράγει μια πρόβλεψη με βάση τα μαθημένα χαρακτηριστικά.

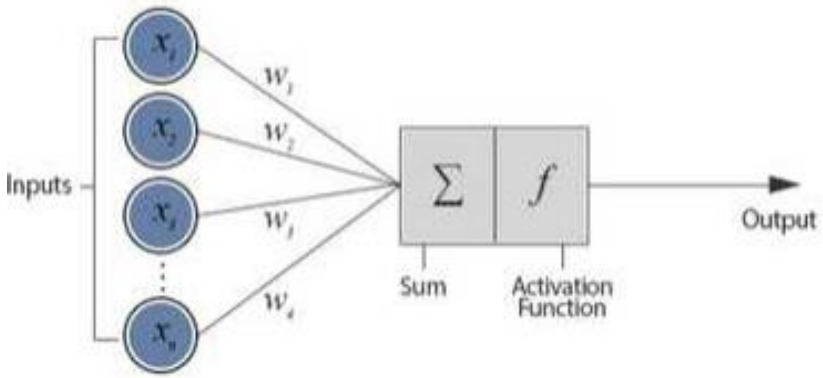


Εικόνα 4:Βαθιά Αρχιτεκτονική Δικτύου με Πολλαπλά Στρώματα

Το επόμενο βήμα είναι ο υπολογισμός της συνάρτησης απώλειας. Η συνάρτηση απώλειας μετρά τη διαφορά μεταξύ της προβλεπόμενης εξόδου και της πραγματικής εξόδου. Υπάρχουν διαφορετικοί τύποι συναρτήσεων απώλειας, ανάλογα με την εργασία, όπως το μέσο τετραγωνικό σφάλμα (mean square error) για εργασίες παλινδρόμησης και η διασταυρούμενη εντροπία (cross entropy) για εργασίες ταξινόμησης.

Στην συνέχεια, σειρά έχει ο υπολογισμός της κλίσης της συνάρτησης απώλειας σε σχέση με τις παραμέτρους του δικτύου και τη χρήση της για την ενημέρωση των βαρών και των προκαταλήψεων των νευρώνων. Η κλίση υπολογίζεται διαδίδοντας το σφάλμα προς τα πίσω μέσω του δικτύου. Ο αλγόριθμος βελτιστοποίησης, όπως και η στοχαστική κάθοδος κλίσης, χρησιμοποιούνται για την ενημέρωση των παραμέτρων και την ελαχιστοποίηση της συνάρτησης απώλειας.

Οι παραπάνω διαδικασίες επαναλαμβάνονται για πολλές εποχές (epochs) ή επαναλήψεις, έως ότου η συνάρτηση απώλειας συγκλίνει σε ένα ελάχιστο. Κατά τη διάρκεια κάθε επανάληψης, το νευρωνικό δίκτυο μαθαίνει να εξάγει πιο σημαντικά χαρακτηριστικά από τα δεδομένα εισόδου και να παράγει πιο ακριβείς προβλέψεις.



Εικόνα 5: Υπολογισμός Εξόδου Νευρωνικού Δικτύου

Αφού ολοκληρωθεί η εκπαίδευση, το τελικό βήμα είναι η αξιολόγηση της απόδοσης του μοντέλου στο σύνολο των δοκιμών. Οι μετρικές (metrics) που χρησιμοποιούνται για την αξιολόγηση εξαρτώνται από το ζητούμενο πρόβλημα, όπως το μέσο απόλυτο σφάλμα (mean absolute error) για εργασίες παλινδρόμησης και η ακρίβεια (accuracy) για εργασίες ταξινόμησης. Η απόδοση μπορεί επίσης να απεικονιστεί με τη χρήση διαφόρων τεχνικών, όπως οι πίνακες σύγχυσης (confusion matrix).

		Predicted Class: Positive	**Predicted Class: Negative**	
Actual Class	**Positive**	True Positive (TP)	False Negative (FN) **Type II Error**	**Sensitivity** $\frac{TP}{(TP+FN)}$
	Negative	False Positive (FP) **Type I Error**	True Negative (TN)	**Specificity** $\frac{TN}{(TN+FP)}$
		Precision $\frac{TP}{(TP+FP)}$	**Negative Predictive Value** $\frac{TN}{(TN+FN)}$	**Accuracy** $\frac{TP+TN}{(TP+TN+FP+FN)}$

Εικόνα 6: Πίνακας Σύγχυσης (Confusion Matrix)

1.6 Αλγόριθμοι Βαθιάς Μάθησης

Υπάρχουν πολλαπλοί αλγόριθμοι βαθιάς μάθησης που χρησιμοποιούνται για την επίλυση διαφορετικών τύπων προβλημάτων. Ορισμένοι από τους πιο συνηθισμένους αλγορίθμους βαθιάς μάθησης είναι τα Τεχνητά Νευρωνικά Δίκτυα (Artificial Neural Networks), τα Συνελικτικά Νευρωνικά Δίκτυα (Convolutional Neural Networks) για την αναγνώριση εικόνων και τα Αναδρομικά Νευρωνικά Δίκτυα (Recurrent Neural Networks) για τη μοντελοποίηση ακολουθιών και την επεξεργασία γλώσσας.

Επιπλέον, υπάρχουν και τεχνικές Μεταφοράς Μάθησης (Transfer Learning) που επιτρέπουν τη χρήση προ-εκπαιδευμένων μοντέλων ως αφετηρία για νέα προβλήματα. Η επιλογή του αλγορίθμου εξαρτάται από τη φύση του προβλήματος, τον τύπο των δεδομένων και την επιθυμητή έξοδο.

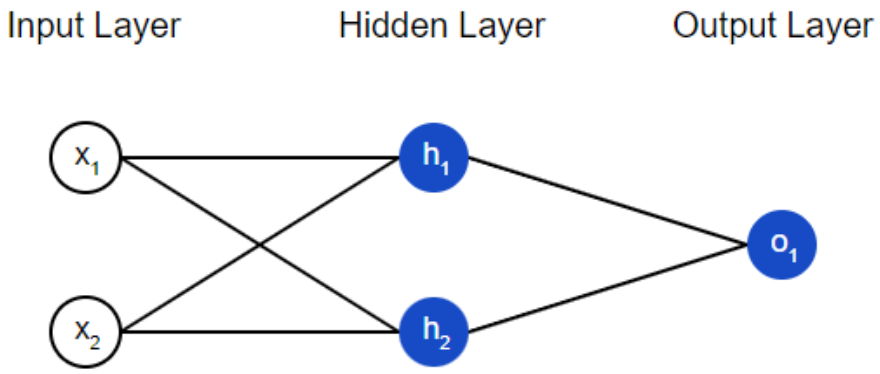
Καθώς η βαθιά μάθηση συνεχίζει να εξελίσσεται, αναπτύσσονται νέοι αλγόριθμοι για την αντιμετώπιση ακόμη πιο σύνθετων και απαιτητικών προβλημάτων.

1.6.1 Τεχνητά Νευρωνικά Δίκτυα (ANN)

Τα Τεχνητά Νευρωνικά Δίκτυα (ANN) είναι μια κατηγορία μοντέλων βαθιάς μάθησης που έχουν σχεδιαστεί για να αναγνωρίζουν μοτίβα σε δεδομένα χρησιμοποιώντας ένα σύνολο διασυνδεδεμένων κόμβων επεξεργασίας ή νευρώνων. Χρησιμοποιούνται συνήθως στην αναγνώριση εικόνας και ομιλίας, στην επεξεργασία φυσικής γλώσσας και στην προγνωστική ανάλυση.

Τα ANNs αποτελούνται από πολλαπλά επίπεδα, συμπεριλαμβανομένου ενός επιπέδου εισόδου, ενός ή περισσότερων κρυφών επιπέδων και ενός επιπέδου εξόδου. Κάθε νευρώνας σε ένα ANN συνδέεται με άλλους νευρώνες μέσω σταθμισμένων συνδέσεων (Weighted Links) και τα σήματα εισόδου υποβάλλονται σε επεξεργασία μέσω του δικτύου για τη δημιουργία μιας εξόδου. Χρησιμοποιούν, επίσης, μια ποικιλία συναρτήσεων ενεργοποίησης, όπως η σιγμοειδής συνάρτηση ή η συνάρτηση της διορθωμένης γραμμικής μονάδας (ReLU), για να εισάγουν μη γραμμικότητα στο μοντέλο.

Τέλος, εκπαιδεύονται χρησιμοποιώντας την οπισθοδιάδοση, μια τεχνική μάθησης με επίβλεψη που προσαρμόζει τα βάρη των συνδέσεων μεταξύ των νευρώνων ώστε να ελαχιστοποιείται η διαφορά μεταξύ της προβλεπόμενης εξόδου και της πραγματικής εξόδου.



Εικόνα 7: Artificial Neural Network

1.6.2 Συνελικτικά Νευρωνικά Δίκτυα (CNN)

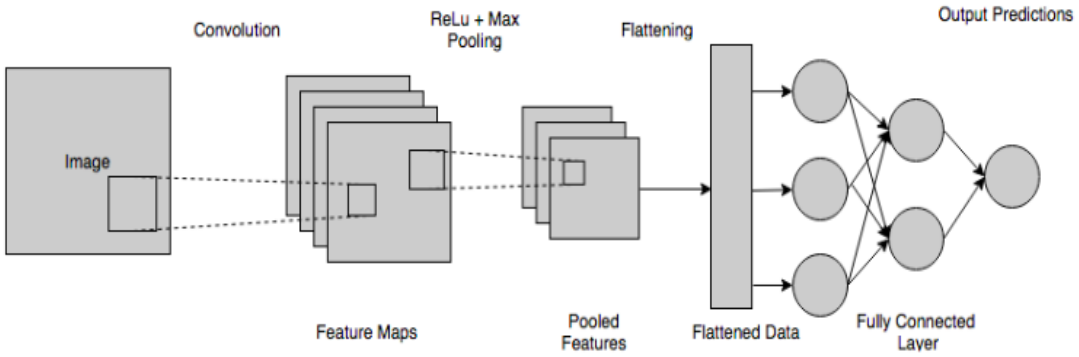
Τα συνελικτικά νευρωνικά δίκτυα (CNN) είναι μια κατηγορία μοντέλων βαθιάς μάθησης που έχουν σχεδιαστεί κυρίως για την αναγνώριση και την επεξεργασία εικόνων. Βασίζονται στην έννοια της συνέλιξης, η οποία περιλαμβάνει την εφαρμογή ενός φίλτρου σε μια εικόνα εισόδου για την εξαγωγή σχετικών χαρακτηριστικών.

Τα CNNs αποτελούνται από πολλαπλά στρώματα, συμπεριλαμβανομένου ενός στρώματος συνέλιξης (Convolutional Layer), ενός στρώματος συγκέντρωσης (Pooling Layer) και ενός πλήρως συνδεδεμένου στρώματος (Fully Connected Layer).

Το στρώμα συνελικτικής επεξεργασίας εφαρμόζει ένα σύνολο φίλτρων στην εικόνα εισόδου για την εξαγωγή σχετικών χαρακτηριστικών. Επιπλέον, το στρώμα συγκέντρωσης μειώνει τις χωρικές διαστάσεις της εξόδου από το στρώμα συνέλιξης με την δειγματοληψία των χαρτών χαρακτηριστικών (Featured Maps). Τέλος, το πλήρως συνδεδεμένο στρώμα εφαρμόζει ένα σύνολο βαρών στην έξοδο του στρώματος συγκέντρωσης για να δημιουργήσει μια τελική έξοδο.

Τα CNNs χρησιμοποιούν και αυτά συναρτήσεις ενεργοποίησης, όπως η συνάρτηση ReLU, για την εισαγωγή μη γραμμικότητας στο μοντέλο, ενώ εκπαιδεύονται με τη χρήση backpropagation, μια τεχνική μάθησης με επίβλεψη που προσαρμόζει τα βάρη

των συνδέσεων μεταξύ των νευρώνων ώστε να ελαχιστοποιείται η διαφορά μεταξύ της προβλεπόμενης εξόδου και της πραγματικής εξόδου, ακριβώς όπως και τα ANN.



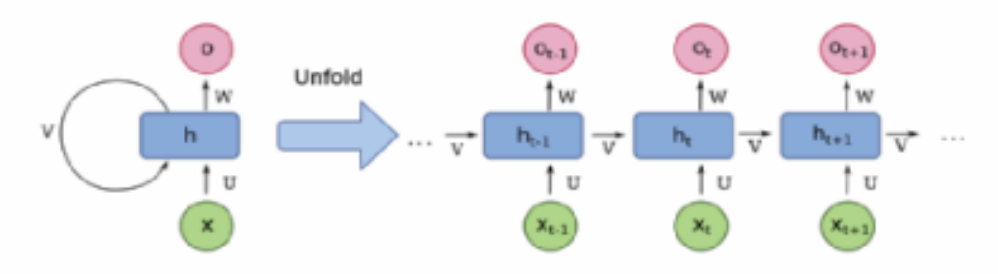
Εικόνα 8: Convolutional Neural Network

1.6.3 Επαναλαμβανόμενα Νευρωνικά Δίκτυα (RNN)

Τα επαναλαμβανόμενα νευρωνικά δίκτυα (RNN) είναι μια κατηγορία μοντέλων βαθιάς μάθησης που έχουν σχεδιαστεί για να επεξεργάζονται διαδοχικά δεδομένα, όπως δεδομένα χρονοσειρών ή δεδομένα φυσικής γλώσσας. Χρησιμοποιούν συνδέσεις ανατροφοδότησης για να εισάγουν μνήμη στο μοντέλο, επιτρέποντας στο δίκτυο να επεξεργάζεται ακολουθίες μεταβλητού μήκους.

Τα RNNs αποτελούνται από ένα επαναλαμβανόμενο στρώμα που εισάγει συνδέσεις ανατροφοδότησης στο μοντέλο. Το επαναλαμβανόμενο στρώμα επιτρέπει στο δίκτυο να διατηρεί μνήμη προηγούμενων εισόδων, επιτρέποντας στο δίκτυο να επεξεργάζεται διαδοχικά δεδομένα. Τα RNNs χρησιμοποιούν συναρτήσεις ενεργοποίησης, όπως η συνάρτηση υπερβολικής εφαπτομένης ή η σιγμοειδής συνάρτηση, για την εισαγωγή μη γραμμικότητας στο μοντέλο.

Τέλος, μπορούν να εκπαιδευτούν χρησιμοποιώντας την οπισθοδιάδοση μέσω του χρόνου, μια παραλλαγή του backpropagation που λαμβάνει υπόψη τις χρονικές εξαρτήσεις στα δεδομένα.



Εικόνα 9: Recurrent Neural Network

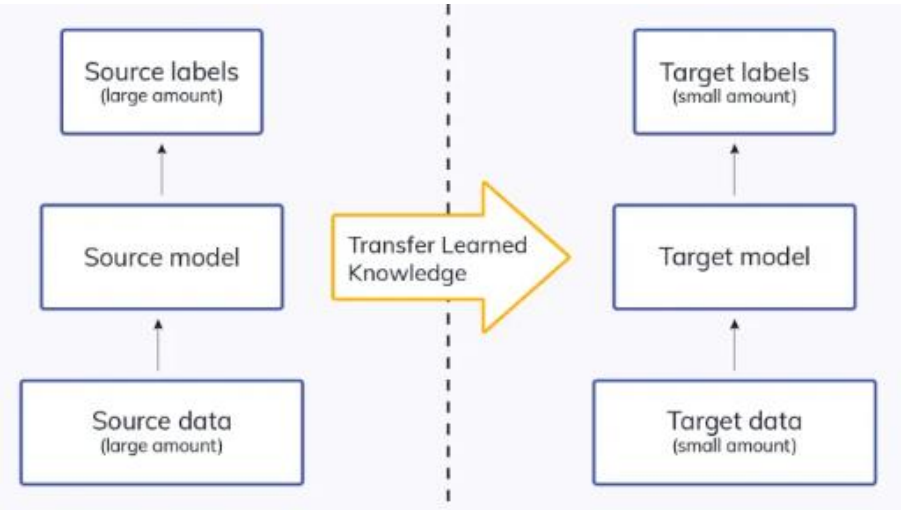
1.7 Μεταφορά Μάθησης (Transfer Learning)

Η μάθηση μεταφοράς για τη μηχανική μάθηση είναι όταν στοιχεία ενός προ-εκπαιδευμένου μοντέλου επαναχρησιμοποιούνται σε ένα νέο μοντέλο μηχανικής μάθησης. Εάν τα δύο μοντέλα έχουν αναπτυχθεί για να εκτελούν παρόμοιες εργασίες, τότε η γενικευμένη γνώση μπορεί να μοιραστεί μεταξύ τους. Αυτή η προσέγγιση στην ανάπτυξη της μηχανικής μάθησης μειώνει τους πόρους και την ποσότητα των επισημειωμένων δεδομένων (labeled data) που απαιτούνται για την εκπαίδευση νέων μοντέλων. Γίνεται σημαντικό μέρος της εξέλιξης της μηχανικής μάθησης και χρησιμοποιείται όλο και περισσότερο ως τεχνική στο πλαίσιο της διαδικασίας ανάπτυξης.

Με τη μάθηση μεταφοράς, βασικά προσπαθούμε να εκμεταλλευτούμε αυτό που έχουμε μάθει σε μια εργασία για να βελτιώσουμε τη γενίκευση σε μια άλλη. Μεταφέρουμε τα βάρη που έχει μάθει ένα δίκτυο στην "εργασία A" σε μια νέα "εργασία B". Η γενική ιδέα είναι να χρησιμοποιήσουμε τη γνώση που έχει μάθει ένα μοντέλο από μια εργασία με πολλά διαθέσιμα επισημασμένα δεδομένα εκπαίδευσης σε μια νέα εργασία που δεν έχει πολλά δεδομένα. Αντί να ξεκινήσουμε τη διαδικασία μάθησης από το μηδέν, ξεκινάμε με πρότυπα που μάθαμε από την επίλυση μιας σχετικής εργασίας.

Το transfer learning είναι χρήσιμο για πολλού λόγους. Αρχικά, συμβάλλει στην εξοικονόμηση χρόνου και πόρων από την εκπαίδευση πολλαπλών μοντέλων μηχανικής μάθησης από το μηδέν για την ολοκλήρωση παρόμοιων εργασιών. Επίσης, προσφέρει αποδοτικότητα σε τομείς της μηχανικής μάθησης που απαιτούν μεγάλες ποσότητες πόρων, όπως η κατηγοριοποίηση εικόνων ή η επεξεργασία φυσικής γλώσσας. Τέλος, εάν υπάρχει έλλειψη επισημειωμένων δεδομένων εκπαίδευσης από

έναν οργανισμό, με τη χρήση προ-εκπαιδευμένων μοντέλων, αυτό παύει να αποτελεί πρόβλημα στην αποδοτική εκπαίδευση κάποιου νέου μοντέλου.



Εικόνα 10: Transfer Learning

1.7.1 Εφαρμογή Μεταφοράς Μάθησης

Προκειμένου να εφαρμοστεί το transfer learning στην εκπαίδευση κάποιου μοντέλου, θα πρέπει να ακολουθήσουμε έξι συγκεκριμένα βήματα:

1. Το πρώτο βήμα είναι να αποκτηθεί το επιθυμητό προ-εκπαιδευμένο μοντέλο για το πρόβλημά που αντιμετωπίζεται.
2. Το δεύτερο βήμα είναι η δημιουργία ενός βασικού μοντέλου (base model). Το προ-εκπαιδευμένο μοντέλο θα έχει συνήθως περισσότερες μονάδες στο τελικό επίπεδο εξόδου από όσες χρειάζονται. Κατά τη δημιουργία του βασικού μοντέλου, θα πρέπει, επομένως, να αφαιρεθεί το τελικό στρώμα εξόδου και αργότερα, να προστεθεί ένα τελικό στρώμα εξόδου που είναι συμβατό με το πρόβλημά που αντιμετωπίζεται.
3. Το πάγωμα των επιπέδων (layers freezing) από το προ-εκπαιδευμένο μοντέλο αποτελεί το επόμενο βήμα και είναι πολύ σημαντικό. Αυτό συμβαίνει επειδή δεν θέλετε να αρχικοποιηθούν εκ νέου τα βάρη στα νέα τα στρώματα. Εάν γίνει αυτό, τότε θα χάσετε όλη τη μάθηση που έχει ήδη πραγματοποιηθεί. Σε περίπτωση που συμβεί αυτό, τότε θα είναι σαν να γίνεται η εκπαίδευση του μοντέλου από την αρχή.

4. Το επόμενο βήμα είναι η προσθήκη νέων εκπαιδεύσιμων επιπέδων που θα μετατρέψουν τα παλιά χαρακτηριστικά σε προβλέψεις στο νέο σύνολο δεδομένων. Αυτό είναι σημαντικό επειδή το προ-εκπαιδευμένο μοντέλο φορτώνεται χωρίς το τελικό στρώμα εξόδου.
5. Στη συνέχεια, σειρά έχει η εκπαίδευση των νέων επιπέδων στο σύνολο δεδομένων. Δηλαδή, για τον λόγο ότι τα προ-εκπαιδευμένα μοντέλα μπορεί να δίνουν έξοδο πολλαπλές κλάσεις, ενώ το νέο μοντέλο έχει λιγότερες, θα πρέπει να εκπαιδευτεί το μοντέλο με ένα νέο επίπεδο εξόδου. Επομένως, θα προστεθούν κάποια νέα πυκνά στρώματα κατά βούληση, αλλά κυρίως ένα τελικό πυκνό στρώμα με μονάδες που αντιστοιχούν στον αριθμό των εξόδων που αναμένονται από το επιθυμητό μοντέλο.
6. Τελικό βήμα στην εφαρμογή της μεταφοράς μάθησης, αποτελεί η βελτίωση του μοντέλου, προαιρετικά, μέσω της λεπτομερούς ρύθμισης (fine-tuning). Η λεπτομερής ρύθμιση γίνεται με το ξεπάγωμα του βασικού μοντέλου ή μέρους αυτού και την εκ νέου εκπαίδευση ολόκληρου του μοντέλου σε ολόκληρο το σύνολο δεδομένων με πολύ χαμηλό ρυθμό μάθησης. Ο χαμηλός ρυθμός μάθησης θα αυξήσει την απόδοση του μοντέλου στο νέο σύνολο δεδομένων, ενώ παράλληλα θα αποτρέψει την υπερβολική προσαρμογή (overfitting). Ο ρυθμός μάθησης (learning rate) πρέπει να είναι χαμηλός επειδή το μοντέλο είναι αρκετά μεγάλο ενώ το σύνολο δεδομένων είναι μικρό. Το αντίθετο θα οδηγούσε στην υπερπροσαρμογή, κάτι το οποίο χρειάζεται να αποφευχθεί.

1.8 Υπερπροσαρμογή (Overfitting)

Η υπερπροσαρμογή είναι μια έννοια στην επιστήμη των δεδομένων, η οποία συμβαίνει όταν ένα στατιστικό μοντέλο ταιριάζει ακριβώς με τα δεδομένα εκπαίδευσης. Όταν συμβαίνει αυτό, ο αλγόριθμος δυστυχώς δεν μπορεί να αποδώσει με ακρίβεια έναντι αθέατων - νέων δεδομένων, με αποτέλεσμα να μην επιτυγχάνεται ο σκοπός του. Η γενίκευση ενός μοντέλου σε νέα δεδομένα είναι αυτό που μας επιτρέπει να χρησιμοποιούμε καθημερινά αλγόριθμους μηχανικής μάθησης για να κάνουμε προβλέψεις και να ταξινομούμε δεδομένα.

Όταν κατασκευάζονται αλγόριθμοι μηχανικής μάθησης, αξιοποιούν ένα σύνολο δειγματικών δεδομένων για την εκπαίδευση του μοντέλου. Ωστόσο, όταν το μοντέλο εκπαιδεύεται για πολύ καιρό σε δειγματικά δεδομένα ή όταν το μοντέλο είναι πολύ περίπλοκο, μπορεί να αρχίσει να μαθαίνει τον "θόρυβο" (“noise”) ή τις άσχετες πληροφορίες μέσα στο σύνολο δεδομένων. Όταν το μοντέλο απομνημονεύει το θόρυβο και προσαρμόζεται πολύ στενά στο σύνολο εκπαίδευσης, το μοντέλο γίνεται "υπερβολικά προσαρμοσμένο" (“overfitted”) και δεν μπορεί να γενικεύσει καλά νέα δεδομένα. Εάν συμβεί αυτό, τότε δεν θα είναι σε θέση να εκτελέσει τις εργασίες ταξινόμησης ή πρόβλεψης για τις οποίες προοριζόταν.

Τα χαμηλά ποσοστά σφάλματος και η υψηλή διακύμανση (variance) είναι καλοί δείκτες υπερπροσαρμογής. Προκειμένου να αποφευχθεί αυτού του είδους η συμπεριφορά, μέρος του συνόλου δεδομένων εκπαίδευσης συνήθως τίθεται στην άκρη ως "σύνολο δοκιμής" (“test set”) για τον έλεγχο της υπερπροσαρμογής. Εάν τα δεδομένα εκπαίδευσης έχουν χαμηλό ποσοστό σφάλματος και τα δεδομένα δοκιμής έχουν υψηλό ποσοστό σφάλματος, αυτό σηματοδοτεί overfitting.

1.8.1 Αποφυγή Υπερπροσαρμογής

Σημαντικό μέρος της εκπαίδευσης ενός μοντέλου αποτελεί η κατανόηση των τρόπων αποφυγής του overfitting. Μερικοί από τους τρόπους αυτούς αναγράφονται παρακάτω:

- Εκπαίδευση με περισσότερα δεδομένα: Η επέκταση του συνόλου εκπαίδευσης ώστε να περιλαμβάνει περισσότερα δεδομένα μπορεί να αυξήσει την ακρίβεια του μοντέλου παρέχοντας περισσότερες ευκαιρίες για την ανάλυση της κυρίαρχης σχέσης μεταξύ των μεταβλητών εισόδου και εξόδου.
- Αύξηση δεδομένων (Data augmentation): Μερικές φορές προστίθενται θορυβώδη δεδομένα για να γίνει ένα μοντέλο πιο σταθερό. Ωστόσο, αυτή η μέθοδος θα πρέπει να εφαρμόζεται αραιά και με προσοχή.
- Πρόωρη διακοπή (Early stopping): Η μέθοδος αυτή επιδιώκει να διακόψει την εκπαίδευση πριν το μοντέλο αρχίσει να μαθαίνει το θόρυβο μέσα στο

μοντέλο. Αυτή η προσέγγιση ενέχει τον κίνδυνο να σταματήσει η διαδικασία εκπαίδευσης πολύ σύντομα, οδηγώντας στο αντίθετο πρόβλημα της υποπροσαρμογής.

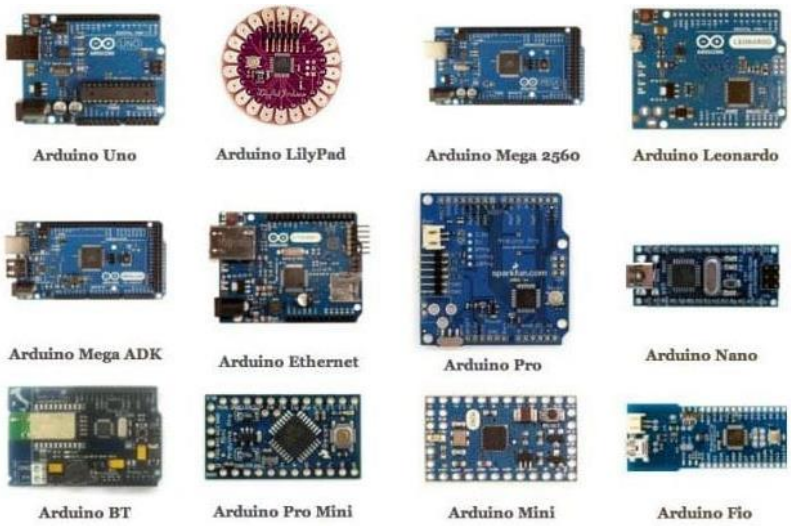
- Επιλογή χαρακτηριστικών (Feature selection): Πολλές φορές κάποια από τα χαρακτηριστικά των δεδομένων μπορεί να είναι περιττά σε σχέση με άλλα. Η επιλογή χαρακτηριστικών είναι η διαδικασία εντοπισμού των πιο σημαντικών μέσα στα δεδομένα εκπαίδευσης και, στη συνέχεια, η εξάλειψη των άσχετων ή περιττών.
- Κανονικοποίηση (Regularization): Εάν δεν γνωρίζουμε ποια χαρακτηριστικά πρέπει να αφαιρέσουμε από το μοντέλο μας, οι μέθοδοι κανονικοποίησης μπορούν να φανούν ιδιαίτερα χρήσιμες. Η κανονικοποίηση εφαρμόζει μια "ποινή" (“penalty”) στις παραμέτρους εισόδου με τους μεγαλύτερους συντελεστές, η οποία στη συνέχεια περιορίζει το ποσό της διακύμανσης στο μοντέλο.

Κεφάλαιο 2: Πλακέτα Arduino

2.1 Εισαγωγή στο Arduino

Το Arduino είναι μια ηλεκτρονική πλατφόρμα ανοικτού κώδικα που βασίζεται σε εύχρηστο υλικό και λογισμικό και έχει σχεδιαστεί για να διευκολύνει τους ανθρώπους να δημιουργούν διαδραστικές και έξυπνες ηλεκτρονικές συσκευές. Η πλατφόρμα αποτελείται τόσο από στοιχεία υλικού όσο και από στοιχεία λογισμικού και έχει γίνει δημοφιλής μεταξύ ερασιτεχνών, φοιτητών και επαγγελματιών για την ευκολία χρήσης, την ευελιξία και την προσιτή τιμή της.

Το Arduino γεννήθηκε στο Ivrea Interaction Design Institute ως ένα εύκολο εργαλείο για γρήγορη κατασκευή πρωτοτύπων, που απευθυνόταν σε κοινό χωρίς υπόβαθρο στα ηλεκτρονικά και τον προγραμματισμό. Μόλις έφτασε σε μια ευρύτερη κοινότητα, η πλακέτα Arduino άρχισε να αλλάζει ώστε προσαρμοστεί σε νέες ανάγκες και προκλήσεις. Αυτό οδήγησε στην εξέλιξή των πλακετών από απλές 8-bit σε προϊόντα για εφαρμογές IoT (Internet of Things), wearables, τρισδιάστατη εκτύπωση και ενσωματωμένα περιβάλλοντα (embedded environments).



Εικόνα 11: Διάφοροι Τύποι Arduino

Επιπλέον, το Arduino έχει σχεδιαστεί για να είναι ιδιαίτερα ευέλικτο και προσαρμόσιμο, γεγονός που το καθιστά ιδανικό για ένα ευρύ φάσμα εφαρμογών. Μπορεί να χρησιμοποιηθεί για την κατασκευή οτιδήποτε, από απλά ηλεκτρονικά

παιχνίδια και παιχνίδια μέχρι σύνθετα ρομποτικά συστήματα και διαδραστικές εγκαταστάσεις. Η πλατφόρμα είναι επίσης εξαιρετικά επεκτάσιμη, με μια μεγάλη και ενεργή κοινότητα χρηστών που συνεισφέρουν βιβλιοθήκες, κώδικα παραδειγμάτων και άλλους πόρους που διευκολύνουν τους ανθρώπους να ξεκινήσουν με το Arduino και να κατασκευάσουν πιο προηγμένα έργα.

2.2 Πλεονεκτήματα Arduino

Υπάρχουν διάφοροι λόγοι για τους οποίους το Arduino αποτελεί μια δημοφιλή πλατφόρμα για την κατασκευή ψηφιακών συσκευών και διαδραστικών συστημάτων. Μερικά από τα κύρια πλεονεκτήματα της χρήσης του Arduino περιλαμβάνουν:

- ✓ Ευκολία στην χρήση: Το Arduino διαθέτει ένα φιλικό προς το χρήστη περιβάλλον εργασίας και μια απλή γλώσσα προγραμματισμού που διευκολύνει τους αρχάριους να ξεκινήσουν. Το περιβάλλον ανάπτυξης λογισμικού είναι δωρεάν και ανοικτού κώδικα και υπάρχουν πολλά διαδικτυακά σεμινάρια και πόροι που είναι διαθέσιμοι για να βοηθήσουν τους χρήστες να μάθουν και να αντιμετωπίσουν προβλήματα.

- ✓ Ευελιξία: Το Arduino μπορεί να χρησιμοποιηθεί για ένα ευρύ φάσμα εφαρμογών, από απλές εργασίες μέχρι σύνθετα έργα ρομποτικής. Η πλατφόρμα είναι εξαιρετικά προσαρμόσιμη και μπορεί να καλύψει ποικίλες ανάγκες και απαιτήσεις.

- ✓ Χαμηλό κόστος: Οι πλακέτες Arduino είναι σχετικά φθηνές, καθιστώντας τις προσιτές σε ένα ευρύ φάσμα χρηστών. Η πλατφόρμα είναι επίσης ανοικτού κώδικα, πράγμα που σημαίνει ότι οι χρήστες μπορούν να δημιουργήσουν τις δικές τους προσαρμοσμένες πλακέτες και εξαρτήματα, εάν χρειάζονται κάτι που δεν είναι διαθέσιμο στο ράφι.

- ✓ Υποστήριξη από την κοινότητα: Το Arduino διαθέτει μια μεγάλη και ενεργή κοινότητα χρηστών, προγραμματιστών και ενθουσιωδών που μοιράζονται τις γνώσεις και την εμπειρία τους μέσω διαδικτυακών φόρουμ, ιστολογίων και

μέσων κοινωνικής δικτύωσης. Αυτή η υποστήριξη της κοινότητας μπορεί να είναι ανεκτίμητη για τους αρχάριους που μόλις ξεκινούν, καθώς και για τους πιο έμπειρους χρήστες που επιθυμούν να λύσουν πολύπλοκα προβλήματα.

- ✓ Διαλειτουργικότητα: Το Arduino είναι συμβατό με ένα ευρύ φάσμα αισθητήρων, ενεργοποιητών και άλλων ηλεκτρονικών εξαρτημάτων, καθιστώντας εύκολη τη σύνδεση και τον έλεγχο διαφορετικών συσκευών. Η πλατφόρμα υποστηρίζει επίσης μια ποικιλία πρωτοκόλλων επικοινωνίας, όπως Bluetooth, Wi-Fi και Ethernet, τα οποία μπορούν να χρησιμοποιηθούν για τη σύνδεση συσκευών Arduino στο Διαδίκτυο ή σε άλλα δίκτυα.

- ✓ Ταχεία κατασκευή πρωτοτύπων: Το Arduino είναι ιδανικό για ταχεία δημιουργία πρωτοτύπων, καθώς επιτρέπει στους χρήστες να δημιουργούν και να δοκιμάζουν γρήγορα τις ιδέες τους χωρίς να χρειάζονται ακριβό εξοπλισμό ή εξειδικευμένες δεξιότητες. Αυτό διευκολύνει την επανάληψη και την τελειοποίηση των σχεδίων και την ταχύτερη μεταφορά από την ιδέα στην πραγματικότητα.

Συνοψίζοντας, το Arduino είναι μια ευέλικτη και φιλική προς τον χρήστη πλατφόρμα που προσφέρει μια σειρά από πλεονεκτήματα για την κατασκευή ψηφιακών συσκευών και διαδραστικών αντικειμένων. Το χαμηλό κόστος, η ευελιξία, η υποστήριξη της κοινότητας και η διαλειτουργικότητά του το καθιστούν δημοφιλή επιλογή για εκπαιδευτικούς και επαγγελματίες και άλλους.

2.3 Τρόπος λειτουργίας πλακέτας

Η πλακέτα Arduino βασίζεται σε έναν μικροελεγκτή, ο οποίος είναι ένας μικρός υπολογιστής που μπορεί να προγραμματιστεί για να εκτελεί διάφορες εργασίες. Διαθέτει ακίδες (pins) εισόδου/εξόδου (Input (I) / Output (O)) που μπορούν να χρησιμοποιηθούν για τη σύνδεση αισθητήρων, ενεργοποιητών και άλλων

ηλεκτρονικών εξαρτημάτων. Η πλακέτα διαθέτει επίσης θύρα USB που της επιτρέπει να επικοινωνεί με έναν υπολογιστή για προγραμματισμό και μεταφορά δεδομένων.

Το περιβάλλον ανάπτυξης λογισμικού για το Arduino βασίζεται στη γλώσσα προγραμματισμού C++ και περιλαμβάνει έναν επεξεργαστή κώδικα (code editor), έναν μεταγλωττιστή (compiler) και έναν φορτωτή εκκίνησης (bootloader). Ο επεξεργαστής κώδικα παρέχει μια απλή και εύχρηστη διεπαφή για τη συγγραφή κώδικα, ενώ ο μεταγλωττιστής μεταφράζει τον κώδικα σε γλώσσα μηχανής που μπορεί να κατανοήσει ο μικροελεγκτής. Ο φορτωτής εκκίνησης είναι ένα κομμάτι λογισμικού που εκτελείται στον μικροελεγκτή και επιτρέπει τον προγραμματισμό της πλακέτας μέσω της θύρας USB.

Για να χρησιμοποιηθεί μια πλακέτα Arduino, πρέπει πρώτα να συνδεθεί στον υπολογιστή χρησιμοποιώντας ένα καλώδιο USB. Στη συνέχεια, μπορεί να γραφθεί ένα πρόγραμμα, ή ένα σκίτσο (sketch), χρησιμοποιώντας το περιβάλλον ανάπτυξης λογισμικού Arduino (Arduino Integrated Development Environment – IDE). Το σκίτσο μπορεί να ελέγχει τη συμπεριφορά της πλακέτας, για παράδειγμα, διαβάζοντας δεδομένα από αισθητήρες και ενεργοποιώντας LED.

Αφού ο κώδικας ολοκληρωθεί, στη συνέχεια γίνεται η μεταφόρτωση (upload) στην πλακέτα Arduino χρησιμοποιώντας τη σύνδεση USB. Ο bootloader της πλακέτας λαμβάνει το sketch και το αποθηκεύει στη μνήμη του μικροελεγκτή. Στη συνέχεια, ο μικροελεγκτής εκτελεί τις εντολές που του έχουν δοθεί, οι οποίες μπορούν να ελέγξουν τη συμπεριφορά της πλακέτας και των συνδεδεμένων ηλεκτρονικών εξαρτημάτων.

Συνοπτικά, οι πλακέτες Arduino είναι σε θέση να διαβάζουν εισόδους από έναν αισθητήρα ή κάποιο κουμπί (button) και να τις μετατρέπουν σε έξοδο, ανάβοντας ένα LED, ενεργοποιώντας ένα μοτέρ, ή στέλνοντας δεδομένα σε κάποια συσκευή. Μπορείτε να πείτε στην πλακέτα σας τι να κάνει στέλνοντας ένα σύνολο εντολών στον μικροελεγκτή της πλακέτας.

2.4 Λογισμικό Arduino

Το λογισμικό (Software) Arduino είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) που χρησιμοποιείται για τον προγραμματισμό κάθε πλακέτας Arduino. Το λογισμικό είναι δωρεάν, ανοικτού κώδικα και διατίθεται για λήψη από τον επίσημο ιστότοπο του Arduino. Επίσης, τρέχει σε όλα λειτουργικά συστήματα.

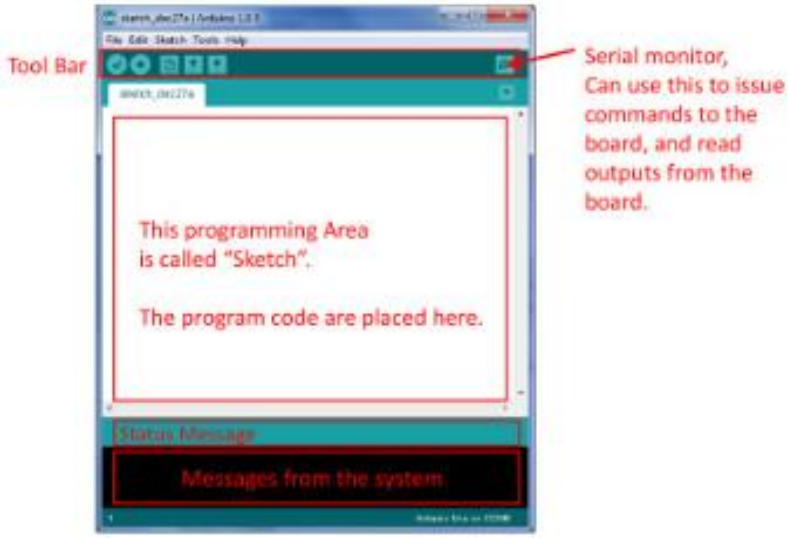
Το IDE του Arduino αποτελείται από τρία κύρια μέρη: τον επεξεργαστή (editor), τον μεταγλωττιστή (compiler) και τον μεταφορτωτή (uploader). Στον επεξεργαστή γράφει ο χρήστης τον κώδικα, ο μεταγλωττιστής μετατρέπει τον κώδικα σε μορφή αναγνώσιμη από μηχανήματα και ο μεταφορτωτής μεταφέρει τον κώδικα στην πλακέτα Arduino.

Η γλώσσα προγραμματισμού του Arduino βασίζεται σε μια απλοποιημένη έκδοση της C++, η οποία διευκολύνει τους αρχάριους να ξεκινήσουν τον προγραμματισμό. Το λογισμικό συνοδεύεται επίσης από μια μεγάλη βιβλιοθήκη προ-γραμμένου κώδικα, που ονομάζεται "σκίτσα" (“sketches”), τα οποία οι χρήστες μπορούν να τροποποιήσουν για να δημιουργήσουν τα έργα τους.

Ένα από τα βασικά χαρακτηριστικά του λογισμικού Arduino είναι η ικανότητά του να επικοινωνεί με την πλακέτα Arduino. Το λογισμικό περιλαμβάνει μια σειριακή οθόνη (serial monitor) που επιτρέπει στους χρήστες να στέλνουν και να λαμβάνουν δεδομένα μεταξύ της πλακέτας και του υπολογιστή. Αυτό είναι χρήσιμο για την αποσφαλμάτωση (debugging) και την αντιμετώπιση προβλημάτων.

Το IDE περιλαμβάνει επίσης μια σειρά εργαλείων για την εργασία με αισθητήρες, κινητήρες και άλλα ηλεκτρονικά εξαρτήματα. Το λογισμικό υποστηρίζει διάφορα πρωτόκολλα επικοινωνίας, καθιστώντας το συμβατό με ένα ευρύ φάσμα αισθητήρων και συσκευών.

Εκτός από το τυπικό λογισμικό Arduino, υπάρχουν επίσης πολλές διαθέσιμες βιβλιοθήκες και εργαλεία τρίτων κατασκευαστών που μπορούν να χρησιμοποιηθούν για την επέκταση της λειτουργικότητάς του.



Εικόνα 12: Arduino IDE

2.4.1 Επεξεργαστής Κώδικα

Ο επεξεργαστής ή συντάκτης κώδικα (code editor) είναι το σημείο όπου οι χρήστες γράφουν τον κώδικά τους. Περιλαμβάνει λειτουργίες όπως η επισήμανση συντακτικού, η συμπλήρωση κώδικα και ο έλεγχος σφαλμάτων, οι οποίες βοηθούν τους χρήστες να γράψουν σωστό κώδικα. Ο επεξεργαστής διαθέτει επίσης μια γραμμή εργαλείων που παρέχει πρόσβαση σε συχνά χρησιμοποιούμενες εντολές και λειτουργίες.

2.4.2 Σκίτσα

Στο Arduino IDE, τα προγράμματα ονομάζονται σκίτσα (sketches). Ένα σκίτσο είναι ένα σύνολο εντολών που θα εκτελέσει η πλακέτα Arduino. Το Arduino IDE περιλαμβάνει μια σειρά από ενσωματωμένα παραδείγματα και βιβλιοθήκες που οι χρήστες μπορούν να τροποποιήσουν για να δημιουργήσουν τα δικά τους σκίτσα.

2.4.3 Σειριακή Οθόνη

Η σειριακή οθόνη (serial monitor) είναι ένα εργαλείο που επιτρέπει στους χρήστες να επικοινωνούν με την πλακέτα Arduino. Εμφανίζει τα δεδομένα που η πλακέτα στέλνει πίσω στον υπολογιστή και μπορεί επίσης να χρησιμοποιηθεί για την αποστολή δεδομένων στην πλακέτα. Αυτό είναι χρήσιμο για την αποσφαλμάτωση και την αντιμετώπιση προβλημάτων.

2.4.4 Εργαλεία

Από το πεδίο εργαλεία (tools) του IDE υπάρχει η δυνατότητα να προβείς σε διάφορες ενέργειες, δύο εκ των οποίων είναι η διαχείριση πλακετών και βιβλιοθηκών.

Ο διαχειριστής πλακετών (board manager) χρησιμοποιείται για την προσθήκη υποστήριξης για διαφορετικούς τύπους πλακετών Arduino στο IDE. Περιλαμβάνει μια λίστα με τις υποστηριζόμενες πλακέτες και οι χρήστες μπορούν να κατεβάσουν πρόσθετα πακέτα για να προσθέσουν υποστήριξη και για άλλες πλακέτες.

Ο διαχειριστής βιβλιοθηκών (library manager) χρησιμοποιείται για την προσθήκη βιβλιοθηκών στο IDE. Οι βιβλιοθήκες είναι συλλογές κώδικα που παρέχουν πρόσθετη λειτουργικότητα στην πλακέτα Arduino. Ο διαχειριστής βιβλιοθηκών διευκολύνει την εύρεση και εγκατάσταση βιβλιοθηκών από το αποθετήριο βιβλιοθηκών του Arduino.

2.5 Υλικό Arduino

Το υλικό (Hardware) της κάθε πλακέτας Arduino αποτελείται από διαφορετικά εξαρτήματα. Έχουν δηλαδή, συγκεκριμένα χαρακτηριστικά και δυνατότητες.

Υπάρχουν, όμως κάποια βασικά εξαρτήματα που συναντώνται συνήθως σε όλες τις πλακέτες Arduino. Μερικά από αυτά είναι:

- Ο Μικροελεγκτής - Αποτελεί την καρδιά της πλακέτας και είναι, ουσιαστικά, ένας μικρός υπολογιστής σε ένα μόνο τσιπ. Είναι υπεύθυνος για την εκτέλεση του κώδικα προγράμματος που μεταφορτώνεται σε αυτόν μέσω υπολογιστή.

- Η Τροφοδοσία Ρεύματος - Οι πλακέτες Arduino μπορούν να τροφοδοτηθούν με ρεύμα από ένα καλώδιο USB ή από ένα εξωτερικό τροφοδοτικό. Η πλακέτα διαθέτει επίσης έναν ρυθμιστή τάσης που ρυθμίζει την εισερχόμενη τάση σε μια σταθερή τάση 5 βολτ (Volt) που μπορεί να χρησιμοποιηθεί για την τροφοδοσία άλλων εξαρτημάτων.

- Οι Ψηφιακές Ακίδες - Η πλακέτα διαθέτει αρκετές ψηφιακές ακίδες (Digital pins) εισόδου / εξόδου (I/O). Αυτές μπορούν να χρησιμοποιηθούν για την ανάγνωση ψηφιακών αισθητήρων ή τον έλεγχο ψηφιακών συσκευών, όπως LED ή κινητήρες (motors).

- Οι Αναλογικές Ακίδες - Η πλακέτα Arduino διαθέτει επίσης αρκετές ακίδες αναλογικής εισόδου (Analog pins), οι οποίες μπορούν να χρησιμοποιηθούν για την ανάγνωση αναλογικών αισθητήρων, όπως αισθητήρες θερμοκρασίας ή φωτός και ποτενσιόμετρα.

- Η Σειριακή Διεπαφή - Η πλακέτα Arduino διαθέτει σειριακή διεπαφή (Serial Interface), η οποία της επιτρέπει να επικοινωνεί με άλλες συσκευές, όπως υπολογιστές, άλλους μικροελεγκτές και αισθητήρες.

- Το Κουμπί Επαναφοράς – Κάθε πλακέτα περιλαμβάνει ένα κουμπί επαναφοράς (Reset Button) για την επαναφορά του μικροελεγκτή και την εκκίνηση του κώδικα του προγράμματος από την αρχή.

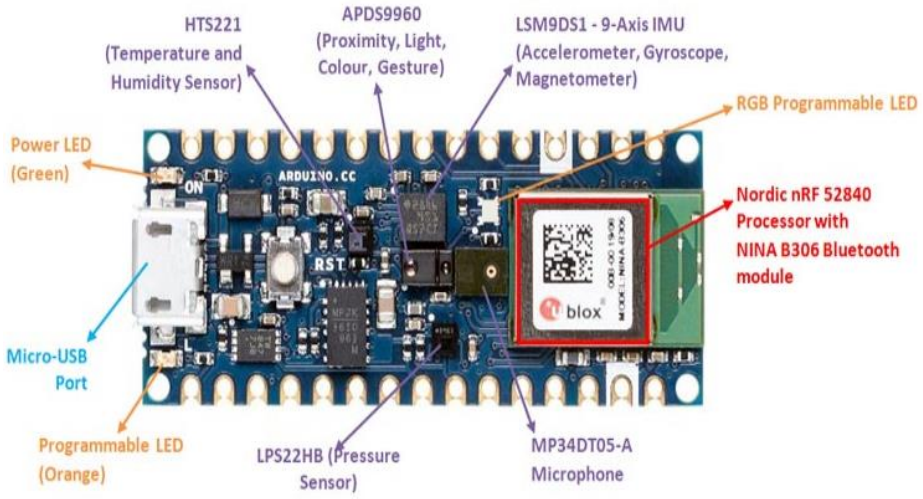
- Υποδοχή USB – Η υποδοχή USB (USB Connector) χρησιμοποιείται για τη σύνδεση της πλακέτας Arduino με έναν υπολογιστή για τον προγραμματισμό και την τροφοδοσία της.

2.6 Arduino Nano 33 BLE Sense

Στην διπλωματική αυτή εργασία, επέλεξα να εργαστώ με το Arduino Nano 33 BLE Sense λόγω των ισχυρών δυνατοτήτων και των ιδιαίτερων χαρακτηριστικών του. Το μικρό μέγεθος και η χαμηλή κατανάλωση ενέργειας της πλακέτας την καθιστούν ιδανική για εφαρμογές IoT, ενώ οι ενσωματωμένοι αισθητήρες, συμπεριλαμβανομένου ενός IMU (Inertial Measurement Unit) 9 αξόνων και αισθητήρων θερμοκρασίας, υγρασίας και πίεσης, παρέχουν αξιόπιστα δεδομένα για ανάλυση. Επιπλέον, η συνδεσιμότητα Bluetooth 5.0 και Bluetooth Low Energy (BLE) της πλακέτας επιτρέπουν την εύκολη επικοινωνία με άλλες συσκευές.

Πιο συγκεκριμένα, το Arduino Nano 33 BLE Sense, βασίζεται στο τσιπ Nordic nRF52840, έναν ισχυρό μικροελεγκτή Cortex-M4F που λειτουργεί στα 64 MHz με μνήμη flash 1 MB και μνήμη RAM 256 KB. Η πλακέτα μπορεί να προγραμματιστεί χρησιμοποιώντας το Arduino IDE ή άλλες γλώσσες προγραμματισμού όπως η Python, η JavaScript ή η C++.

Συνολικά, η πλακέτα Arduino Nano 33 BLE Sense αποτελεί κατάλληλη επιλογή για την διπλωματική μου εργασία, παρέχοντάς μου τα εργαλεία που χρειάζομαι για τη συλλογή και ανάλυση δεδομένων για την αναγνώριση δραστηριοτήτων. Παρακάτω, παρουσιάζονται οι τεχνικές προδιαγραφές της, οι πιθανές εφαρμογές της και οι δυνατότητές του για την ανάπτυξη μοντέλων βαθιάς μάθησης.



Εικόνα 13: Arduino Nano 33 BLE Sense - Hardware

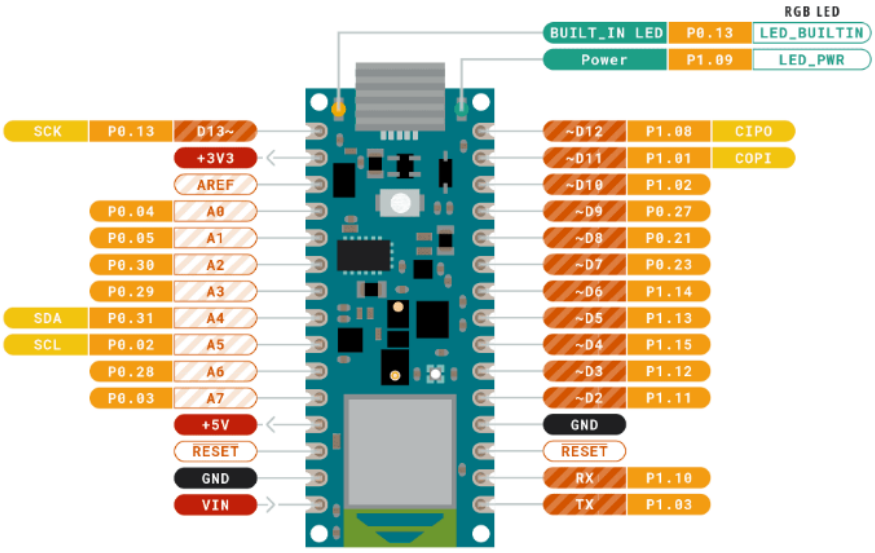
2.6.1 Τεχνικά Χαρακτηριστικά

Οι τεχνικές προδιαγραφές του Arduino Nano 33 BLE Sense το καθιστούν μια ισχυρή πλακέτα μικροελεγκτή με ευρύ φάσμα δυνατοτήτων. Η πλακέτα αποτελείται από το τσιπ Nordic nRF52840 και λειτουργεί στα 64 MHz, ενώ έχει μνήμη flash 1 MB και μνήμη RAM 256 KB. Η πλακέτα περιλαμβάνει, επίσης, συνδεσιμότητα Bluetooth 5.0 και Bluetooth Low Energy (BLE). Η επικοινωνία BLE υποστηρίζεται από μια ενσωματωμένη κεραία, παρέχοντας εμβέλεια έως και 30 μέτρα. Επιπλέον, η πλακέτα φέρει μία ποικιλία αισθητήρων, όπως μια IMU 9 αξόνων (LSM9DS), αισθητήρες θερμοκρασίας, υγρασίας και πίεσης. Η IMU περιλαμβάνει επιταχυνσιόμετρο 3 αξόνων, γυροσκόπιο 3 αξόνων και μαγνητόμετρο 3 αξόνων, παρέχοντας ένα ευρύ φάσμα δυνατοτήτων ανίχνευσης κίνησης. Οι αισθητήρες θερμοκρασίας, υγρασίας και πίεσης είναι όλοι αισθητήρες υψηλής ακρίβειας και χαμηλού θορύβου, παρέχοντας αξιόπιστα δεδομένα για ποικίλες εφαρμογές. Η πλακέτα περιλαμβάνει, επίσης, ένα ενσωματωμένο μικρόφωνο για εφαρμογές ήχου και ένα τσιπ κρυπτογράφησης για ασφαλή επικοινωνία. Τέλος, έχει διαστάσεις 18mm πλάτος και 45mm μήκος, ενώ το βάρος της είναι μόλις 5gr, στοιχεία που την καθιστούν καλή επιλογή για υλοποίηση συσκευών μικρού μεγέθους.

Clock speed	**Processor**	nRF52840 64MHz
Memory	**nRF52840**	256 KB SRAM, 1MB flash
Connectivity	**Bluetooth®**	NINA-B306
Sensors	**IMU**	LSM9DS
	Microphone	MP34DT05
	Gesture, light, proximity	APDS9960
	Barometric pressure	LPS22HB
	Temperature, humidity	HTS221
Dimensions	**Weight**	5gr
	Width	18 mm
	Length	45 mm

Εικόνα 14: Τεχνικά Χαρακτηριστικά Πλακέτας

Όσο αφορά τις επιλογές συνδεσμολογίας και τις ακίδες (pins), το Arduino Nano 33 BLE Sense, περιλαμβάνει 8 αναλογικές (analog) και 14 ψηφιακές (digital), ενώ 5 από αυτές είναι PWM (Pulse Width Modulation), για την λήψη αναλογικών αποτελεσμάτων με ψηφιακά μέσα. Επιπροσθέτως, συναντώνται 2 ακίδες γείωσης (GND), μια ακίδα για είσοδο 3,3V, μία για 5V, καθώς και η Vin, η οποία μπορεί να λάβει τιμές από 4,5V, έως 21V. Τέλος, το Arduino χρησιμοποιεί σειριακή επικοινωνία για τη μεταφορά δεδομένων μεταξύ μικροελεγκτή και υπολογιστή ή με οποιονδήποτε άλλο μικροελεγκτή. Για την επικοινωνία αυτή χρησιμοποιείται ένας σειριακός δίαυλος, ο οποίος αποτελείται από δύο ακροδέκτες, ο ένας για την αποστολή δεδομένων και ο άλλος για τη λήψη δεδομένων. Έτσι, όλες οι συσκευές που χρησιμοποιούν σειριακό πρωτόκολλο έχουν δύο σειριακές ακίδες, την RX, που είναι ο δέκτης (Receiver) και την TX, που είναι ο πομπός (Transmitter).



Εικόνα 15: Ακίδες (Pins) Arduino Nano 33 BLE Sense

2.6.2 Πιθανές Εφαρμογές

Το Arduino Nano 33 BLE Sense έχει ένα ευρύ φάσμα πιθανών εφαρμογών σε διάφορους τομείς, όπως η υγειονομική περίθαλψη, ο αθλητισμός και ο βιομηχανικός αυτοματισμός. Στην υγειονομική περίθαλψη, η πλακέτα θα μπορούσε να χρησιμοποιηθεί για την παρακολούθηση ζωτικών σημείων και τη συλλογή δεδομένων για ανάλυση. Για παράδειγμα, οι αισθητήρες θερμοκρασίας και υγρασίας θα μπορούσαν να χρησιμοποιηθούν για την παρακολούθηση των περιβαλλοντικών

συνθηκών ενός ασθενούς, ενώ η IMU θα μπορούσε να χρησιμοποιηθεί για την παρακολούθηση των κινήσεών του. Στον αθλητισμό, η πλακέτα θα μπορούσε να χρησιμοποιηθεί για την παρακολούθηση των επιδόσεων των αθλητών και την παρακολούθηση των περιβαλλοντικών συνθηκών. Για παράδειγμα, οι αισθητήρες θερμοκρασίας και υγρασίας θα μπορούσαν να χρησιμοποιηθούν για την παρακολούθηση των καιρικών συνθηκών, ενώ η IMU θα μπορούσε να χρησιμοποιηθεί για την παρακολούθηση των κινήσεων του αθλητή. Στον βιομηχανικό αυτοματισμό, η πλακέτα θα μπορούσε να χρησιμοποιηθεί για τον έλεγχο των μηχανημάτων και την παρακολούθηση των διαδικασιών παραγωγής. Για παράδειγμα, ο αισθητήρας θερμοκρασίας θα μπορούσε να χρησιμοποιηθεί για την παρακολούθηση της θερμοκρασίας ενός μηχανήματος, ενώ η IMU θα μπορούσε να χρησιμοποιηθεί για την παρακολούθηση των κινήσεών του.

2.6.3 Ανάπτυξη Μοντέλων Βαθιάς Μάθησης

Το Arduino Nano 33 BLE Sense είναι μια ισχυρή προγραμματίσιμη πλακέτα που μπορεί να χρησιμοποιηθεί για τη δημιουργία μοντέλων μηχανικής και βαθιάς μάθησης. Για να τη χρησιμοποιηθεί αποτελεσματικά, πρέπει να ακολουθηθούν μια σειρά από βήματα:

Το πρώτο βήμα είναι η συλλογή δεδομένων που θα χρησιμοποιηθούν για την εκπαίδευση του μοντέλου μηχανικής μάθησης. Αυτό μπορεί να γίνει χρησιμοποιώντας τους ενσωματωμένους αισθητήρες της πλακέτας ή συνδέοντας εξωτερικούς αισθητήρες σε αυτήν. Για παράδειγμα, για την κατασκευή ενός μοντέλου για την αναγνώριση δραστηριοτήτων, μπορούν να συλλεχθούν δεδομένα από τους αισθητήρες κίνησης.

Μόλις συλλεχθούν τα δεδομένα, θα πρέπει να τα προεπεξεργαστούν κατάλληλα πριν από την εκπαίδευση του μοντέλου. Αυτό περιλαμβάνει τον καθαρισμό και την προετοιμασία των δεδομένων, ώστε να μπορούν να χρησιμοποιηθούν αποτελεσματικά στο μοντέλο.

Μετά την προεπεξεργασία των δεδομένων, πρέπει να επιλεχθεί μία δομή (framework) μηχανικής μάθησης που είναι συμβατή με την πλακέτα και τον τύπο του μοντέλου που θέλετε να δημιουργήσετε. Υπάρχουν διάφορα frameworks μηχανικής μάθησης που μπορούν να χρησιμοποιηθούν με το Arduino Nano 33 BLE Sense, συμπεριλαμβανομένων των TensorFlow Lite και Edge Impulse.

Στη συνέχεια, μπορεί να ξεκινήσει η εκπαίδευση του μοντέλου. Αυτό περιλαμβάνει τη χρήση των προεπεξεργασμένων δεδομένων για να διδαχθεί στο μοντέλο πώς να κάνει προβλέψεις. Η διαδικασία εκπαίδευσης μπορεί να διαρκέσει αρκετό χρόνο, ανάλογα με την πολυπλοκότητα του μοντέλου και τον όγκο των δεδομένων που χρησιμοποιείτε.

Αφού εκπαιδευτεί το μοντέλο, μπορεί να βελτιστοποιηθεί για να γίνει πιο αποδοτικό. Αυτό μπορεί να περιλαμβάνει την προσαρμογή των παραμέτρων του μοντέλου, τη λεπτομερή ρύθμιση της διαδικασίας εκπαίδευσης ή την προσθήκη πρόσθετων δεδομένων στο μοντέλο.

Μόλις το μοντέλο εκπαιδευτεί και βελτιστοποιηθεί, μπορεί να αναπτυχθεί στο Arduino Nano 33 BLE Sense. Αυτό επιτρέπει στην πλακέτα να κάνει προβλέψεις σε πραγματικό χρόνο, χωρίς την ανάγκη σύνδεσης στο cloud. Η διαδικασία ανάπτυξης μπορεί να περιλαμβάνει τη μετατροπή του μοντέλου σε μορφή που μπορεί να χρησιμοποιηθεί από το framework μηχανικής μάθησης (π.χ. TensorFlow Lite) και τη μεταφόρτωσή του στην πλακέτα.

Τέλος, μπορείτε να δοκιμαστεί και να βελτιωθεί το μοντέλο με βάση την απόδοσή του. Ανάλογα με το πόσο ακριβή είναι τα αποτελέσματα της κατηγοριοποίησης και με το πόσο ικανοποιητικά λειτουργεί το μοντέλο, θα αποφασιστεί αν θα χρειαστεί η επαναληπτική εκπαίδευσή του.

Συνολικά, η επιλογή του Arduino Nano 33 BLE Sense για τη δημιουργία ενός μοντέλου μηχανικής μάθησης είναι ιδανική. Το ισχυρό υλικό και οι αισθητήρες της πλακέτας την καθιστούν κατάλληλη για ένα ευρύ φάσμα εφαρμογών μηχανικής μάθησης, όπως η αναγνώριση εικόνας, η αναγνώριση ομιλίας, αλλά και η αναγνώριση κινήσεων ή δραστηριοτήτων.

2.7 Πλεονεκτήματα Arduino Nano 33 BLE Sense

Ανάμεσα στις πολλές πλακέτες Arduino, η συγκεκριμένη πλακέτα αποτελεί την πλέον κατάλληλη για την υλοποίηση της συγκεκριμένης διπλωματικής και, πιο συγκεκριμένα για την δημιουργία μιας συσκευής αναγνώρισης δραστηριότητας σε πραγματικό χρόνο.

Το Arduino Nano 33 BLE Sense χρησιμοποιήθηκε καθώς έχει κάποια χαρακτηριστικά και δυνατότητες που άλλες πλακέτες δεν παρουσιάζουν, γεγονός που το οδηγεί στο να έχει περισσότερα πλεονεκτήματα. Τα σημαντικότερα από αυτά αναγράφονται παρακάτω:

- ✓ Αισθητήρες: Η πλακέτα διαθέτει διάφορους αισθητήρες, όπως ένα επιταχυνσιόμετρο, ένα γυροσκόπιο, ένα μαγνητόμετρο, έναν αισθητήρα πίεσης, έναν αισθητήρα θερμοκρασίας και υγρασίας και ένα μικρόφωνο. Αυτοί οι αισθητήρες επιτρέπουν στην πλακέτα να συλλέγει δεδομένα σχετικά με το περιβάλλον της, καθιστώντας την ιδανική επιλογή για την ανάπτυξη εφαρμογών στον τομέα της υγείας και της ευεξίας, της φυσικής κατάστασης και της παρακολούθησης του περιβάλλοντος.

- ✓ Δυνατότητες μηχανικής μάθησης: Η πλακέτα Arduino Nano 33 BLE Sense διαθέτει έναν ενσωματωμένο επεξεργαστή μηχανικής μάθησης (ML), τον SAMD21, ο οποίος της επιτρέπει να εκτελεί εργασίες ML τοπικά χωρίς να χρειάζεται σύνδεση στο cloud. Αυτό το χαρακτηριστικό το καθιστά κατάλληλο για την ανάπτυξη εφαρμογών που απαιτούν λήψη αποφάσεων σε πραγματικό χρόνο, όπως η προγνωστική συντήρηση και η ανίχνευση ανωμαλιών.

- ✓ Συνδεσιμότητα BLE: Το Arduino Nano 33 BLE Sense είναι εξοπλισμένο με συνδεσιμότητα Bluetooth Low Energy (BLE), η οποία του επιτρέπει να επικοινωνεί ασύρματα με άλλες συσκευές. Αυτό το χαρακτηριστικό το καθιστά κατάλληλο για την ανάπτυξη εφαρμογών που απαιτούν απομακρυσμένη παρακολούθηση και έλεγχο, όπως συσκευές IoT, wearables και έξυπνες οικιακές συσκευές.

- ✓ Μέγεθος: Η πλακέτα έχει μικρό μέγεθος, γεγονός που καθιστά εύκολη την τοποθέτησή της σε στενούς χώρους. Είναι επίσης ελαφριά και έχει χαμηλή κατανάλωση ενέργειας, γεγονός που την καθιστά ιδανική για φορητές εφαρμογές και εφαρμογές που λειτουργούν με μπαταρία.

- ✓ Οικοσύστημα Arduino: Η πλακέτα Arduino Nano 33 BLE Sense είναι πλήρως συμβατή με το οικοσύστημα Arduino, πράγμα που σημαίνει ότι μπορεί να προγραμματιστεί χρησιμοποιώντας το Arduino IDE και μπορεί να ενσωματωθεί με άλλες πλακέτες και ασπίδες (shields) Arduino. Αυτή η συμβατότητα διευκολύνει τους προγραμματιστές να ξεκινήσουν με την πλακέτα και να αξιοποιήσουν τις υπάρχουσες βιβλιοθήκες και τα εργαλεία του οικοσυστήματος Arduino.

Συνολικά, το Arduino Nano 33 BLE Sense είναι μια ισχυρή και ευέλικτη πλακέτα που είναι κατάλληλη για την ανάπτυξη ενός ευρέος φάσματος εφαρμογών. Η συνδεσιμότητα BLE, οι αισθητήρες, οι δυνατότητες μηχανικής μάθησης, το μέγεθος και η συμβατότητα με το οικοσύστημα Arduino την καθιστούν ιδανική επιλογή για πολλές εφαρμογές, όπως και για την δημιουργία συστήματος αναγνώρισης δραστηριότητας με βαθιά μάθηση.

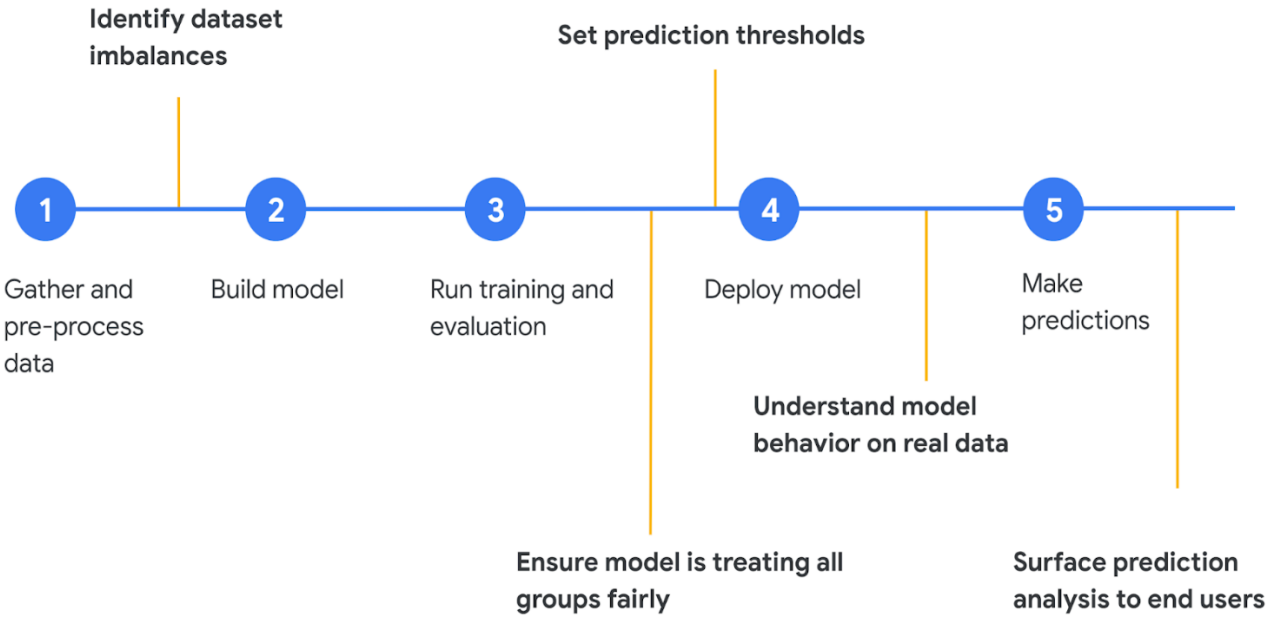
Κεφάλαιο 3: Σύστημα Αναγνώρισης Δραστηριότητας

3.1 Μεθοδολογία

Προκειμένου να σχεδιαστεί και να υλοποιηθεί ένα σύστημα αναγνώρισης δραστηριότητας με βαθιά μάθηση, καθώς και να εφαρμοστεί σε μία πλακέτα, όπως αυτή του Arduino, θα πρέπει να πραγματοποιηθούν συγκεκριμένες διαδικασίες.

Το πρώτο βήμα είναι η επιλογή της συσκευής που θα χρησιμοποιηθεί, καθώς και του σημείου στο σώμα για την τοποθέτηση της. Μόλις επιλεγεί το σημείο για τη συσκευή, το επόμενο βήμα είναι η συλλογή των δεδομένων στα οποία θα εκπαιδευτεί το μοντέλο βαθιάς μάθησης. Αφού συλλεχθούν τα δεδομένα, πρέπει να υποστούν προεπεξεργασία για να προετοιμαστούν για την εκπαίδευση του μοντέλου.

Το επόμενο βήμα είναι η εκπαίδευση ενός μοντέλου βαθιάς μάθησης στα προεπεξεργασμένα δεδομένα. Στη συνέχεια, το μοντέλο πρέπει να αξιολογηθεί στο σύνολο δοκιμών για να μετρηθεί η απόδοσή του. Τέλος, για την ανάπτυξη του μοντέλου βαθιάς μάθησης στο Arduino, το μοντέλο πρέπει να μετατραπεί σε μορφή που να μπορεί να εκτελεστεί στον μικροελεγκτή. Το μετατρεπόμενο μοντέλο μπορεί στη συνέχεια να μεταφορτωθεί στο Arduino και να ενσωματωθεί με τους αισθητήρες.



Εικόνα 16: Μεθοδολογία Υλοποίησης Συστήματος

3.2 Επιλογή Συσκευής και Σημείου Τοποθέτησης

Προτού ξεκινήσει η όποια διαδικασία για την συλλογή ή προετοιμασία δεδομένων για την εκπαίδευση του μοντέλου, μεγάλη σημασία έχει το να βρεθεί η κατάλληλη συσκευή, αλλά και το πιο σωστό μέρος σώματος για την τοποθέτηση της.

Η επιλογή της συσκευής αποτέλεσε σημαντικό ρόλο στην υλοποίηση της διπλωματικής. Μετά από έρευνα κατέληξα στην πλακέτα Arduino Nano 33 BLE Sense, καθώς έχει όλα όσα μου ήταν απαραίτητα. Δηλαδή, οι ενσωματωμένοι αισθητήρες, αλλά και η συνδεσιμότητα Bluetooth, βοήθησαν στην πραγματοποίηση της ιδέας. Επίσης, το μικρό μέγεθος της πλακέτας μου επέτρεψε να την αξιοποιήσω όπως επιθυμώ.

Η επιλογή του σημείου εξαρτάται από τις δραστηριότητες που πρέπει να αναγνωριστούν. Στην συγκεκριμένη διπλωματική επιλέχθηκαν τέσσερεις βασικές δραστηριότητες της καθημερινότητας. Το περπάτημα (walking), το τρέξιμο (running), η στιγμή που στεκόμαστε όρθιοι (standing), αλλά και καθιστοί (sitting). Οι δραστηριότητες αυτές περιλαμβάνουν κυρίως κινήσεις του κάτω μέρους του σώματος.

Έπειτα από πολλαπλές δοκιμές για την συλλογή των δεδομένων από διάφορα σημεία του σώματος, εξακριβώθηκε πως τα καλύτερα αποτελέσματα ήρθαν όταν οι αισθητήρες της συσκευής ήταν σε κάποιο σημείο κάτω από την μέση. Επομένως, η συσκευή αποφασίστηκε να τοποθετηθεί ψηλά στο πόδι.

Εικόνα 17: Σημείο Τοποθέτησης Συσκευής

3.3 Συλλογή Δεδομένων

Η συλλογή δεδομένων αποτελεί μια κρίσιμη διαδικασία για τη δημιουργία ενός μοντέλου βαθιάς μάθησης για την αναγνώριση ανθρώπινης δραστηριότητας. Περιλαμβάνει την απόκτηση δεδομένων αισθητήρων που αντιπροσωπεύουν τις φυσικές κινήσεις ενός ατόμου.

Το Arduino Nano 33 BLE Sense είναι μια δημοφιλής πλακέτα μικροελεγκτή που είναι εξοπλισμένη με μια σειρά αισθητήρων, συμπεριλαμβανομένου ενός επιταχυνσιόμετρου 3 αξόνων και ενός γυροσκοπίου 3 αξόνων, οι οποίοι και χρησιμοποιήθηκαν. Αυτοί οι αισθητήρες συνεργάζονται για τη μέτρηση της γραμμικής επιτάχυνσης και της γωνιακής ταχύτητας, αντίστοιχα.

Κατά τη διάρκεια της συλλογής δεδομένων, το άτομο φοράει την πλακέτα Arduino Nano 33 BLE Sense στο σώμα του στο επιλεγμένο σημείο, πιο συγκεκριμένα στον μηρό, και εκτελεί τις ζητούμενες δραστηριότητες (περπάτημα, τρέξιμο, κάθισμα και ορθοστασία). Τα δεδομένα των αισθητήρων που συλλέχθηκαν από την πλακέτα, στη συνέχεια αποθηκεύτηκαν σε ένα αρχείο.

3.3.1 Συλλογή Μέσω Arduino IDE

Για τη συλλογή δεδομένων με τη χρήση των αισθητήρων IMU στο Arduino Nano 33 BLE Sense, η πλακέτα έπρεπε πρώτα να προγραμματιστεί για να διαβάζει τις τιμές του επιταχυνσιομέτρου και του γυροσκοπίου, ώστε να τις αποθηκεύει σε κάποιο αρχείο. Αυτό επιτεύχθηκε χρησιμοποιώντας το Arduino IDE.

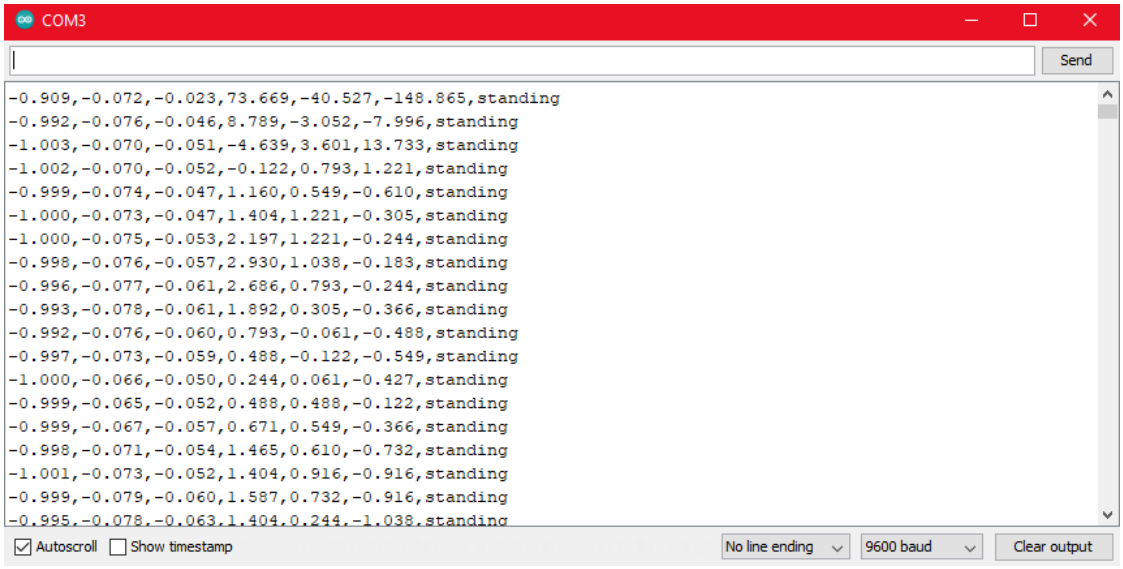
Εκεί, δημιουργήθηκε ένα σκίτσο (sketch) Arduino που, με την βοήθεια της ενσωματωμένης βιβλιοθήκης αισθητήρων ολοκλήρωσε την διασύνδεση με τους αισθητήρες και τη συλλογή των δεδομένων. Το σκίτσο ρυθμίστηκε, μέσω κώδικα ώστε να παίρνει δείγματα από τους αισθητήρες σε συγκεκριμένη συχνότητα ή να ενεργοποιείται από συγκεκριμένο εύρος κίνησης, έχοντας προκαθορισμένη ευαισθησία στους αισθητήρες.

```
#include <Arduino_LSM9DS1.h> //necessary library for IMU

const float accelerationThreshold = 1.4; //threshold of significant motion
const int numSamples = 1000; //samples gathered per 10 seconds

int samplesRead = numSamples; //a value for each sample

void setup() {
  Serial.begin(9600); //start serial monitor
  while (!Serial); //start the proccess when serial monitor is open

  if (!IMU.begin()) {  // print a message if IMU fails to start
    Serial.println("Failed to initialize IMU!");
    while (1);
  }
}

void loop() {
  float aX, aY, aZ, gX, gY, gZ; //variables for 3-axis accelerometer
  char action[] = "standing";   // and 3-axis gyroscope

  // wait for significant motion
  while (samplesRead == numSamples) {
    if (IMU.accelerationAvailable()) {
      // read the acceleration data
      IMU.readAcceleration(aX, aY, aZ);

      // sum up the absolutes
      float aSum = fabs(aX) + fabs(aY) + fabs(aZ);

      // check if it's above the threshold
      if (aSum >= accelerationThreshold) {
        // reset the sample read count
        samplesRead = 0;
        break;
      }
    }
  }
```

```
  // check if the all the required samples have been read since
  // the last time the significant motion was detected
  while (samplesRead < numSamples) {
    // check if both new acceleration and gyroscope data is
    // available
    if (IMU.accelerationAvailable() && IMU.gyroscopeAvailable()) {
      // read the acceleration and gyroscope data
      IMU.readAcceleration(aX, aY, aZ);
      IMU.readGyroscope(gX, gY, gZ);

      samplesRead++;

      // print the data in CSV format
      Serial.print(aX, 3);
      Serial.print(',');
      Serial.print(aY, 3);
      Serial.print(',');
      Serial.print(aZ, 3);
      Serial.print(',');
      Serial.print(gX, 3);
      Serial.print(',');
      Serial.print(gY, 3);
      Serial.print(',');
      Serial.print(gZ, 3);
      Serial.print(',');
      Serial.print("standing");
      Serial.println();

      if (samplesRead == numSamples) {
        // add an empty line if it's the last sample
        Serial.println();
      }
    }
  }
}
```

Εικόνα 18: Συλλογή Δεδομένων μέσω IDE

Με την χρήση του συγκεκριμένου κώδικα, σε ένα sketch του Arduino IDE, είχα σαν αποτέλεσμα κάποιες τιμές από τους τρεις άξονες του επιταχυνσιομέτρου και από τους τρεις άξονες του γυροσκοπίου. Ανάλογα με το πόσο χρόνο λειτουργούσαν οι αισθητήρες, καθώς και ποια δραστηριότητα αντιπροσώπευε η κάθε μια κίνηση, εμφανίζονταν τα αντίστοιχα αποτελέσματα στο Serial Monitor του IDE.



Εικόνα 19: Δεδομένα αισθητήρων IMU στο Serial Monitor

3.3.2 Δημιουργία CSV Αρχείου

Τα δεδομένα αυτά ήταν προγραμματισμένο έτσι ώστε να χωρίζονται με κόμμα μεταξύ των κάθε χαρακτηριστικών. Αυτό έγινε με σκοπό την δημιουργία ενός ολοκληρωμένου CSV αρχείου, το οποίο τελικά περιλαμβάνει κάθε τιμή από όλους τους άξονες (x, y και z) των αισθητήρων που χρησιμοποιήθηκαν. Επίσης, ενώ τα χαρακτηριστικά των δεδομένων ήταν έξι εν τέλη (accX, accY, accZ, gyrX, gyrY, gyrZ), η τελευταία στήλη αναφέρεται στην δραστηριότητα που αντιστοιχεί στην κάθε σειρά. Δηλαδή, κάθισμα, ορθοστασία, περπάτημα ή τρέξιμο.

	A	B	C	D	E	F	G
59994	0.169,-0.976,0.066,1.587,0.305,0.793,sitting						
59995	0.169,-0.977,0.062,2.136,0.244,0.916,sitting						
59996	0.390,-0.834,-0.387,1.221,-0.183,0.549,sitting						
59997	0.388,-0.833,-0.386,1.648,0.000,0.610,sitting						
59998	0.387,-0.832,-0.387,1.648,0.061,0.671,sitting						
59999	0.387,-0.832,-0.388,1.587,0.183,0.610,sitting						
60000	0.389,-0.832,-0.388,1.770,0.244,0.671,sitting						
60001	-0.989,0.008,0.096,8.057,1.587,-3.662,standing						
60002	-0.985,-0.004,0.100,7.324,1.221,-3.418,standing						
60003	-0.983,-0.006,0.100,6.104,0.793,-3.052,standing						
60004	-0.973,-0.005,0.101,4.944,0.427,-2.686,standing						
60005	-0.979,-0.004,0.097,3.479,-0.671,-2.502,standing						
60006	-0.990,0.004,0.090,2.563,-1.648,-2.258,standing						
60007	-0.996,0.011,0.090,2.441,-1.953,-1.770,standing						

activities_dataset

Εικόνα 20: Δημιουργία CSV δραστηριοτήτων

3.4 Python και Jupyter

Για την διαδικασία της προεπεξεργάσιας των δεδομένων, αλλά και για κάθε επόμενη διαδικασία που ακολουθεί αυτήν, έπρεπε να επιλεχθεί μία γλώσσα προγραμματισμού η οποία μπορεί να υποστηρίξει πλήρως το ζητούμενο. Εκτός αυτού μεγάλη βαρύτητα είχε και το περιβάλλον ανάπτυξης του κώδικα. Για την διπλωματική αυτή λοιπόν χρησιμοποιήθηκε η γλώσσα προγραμματισμού Python, καθώς και το περιβάλλον ανάπτυξης Jupyter, μέσω των οποίων ολοκληρώθηκε η εργασία.

Η Python είναι μια δημοφιλής γλώσσα προγραμματισμού που χρησιμοποιείται για την ανάλυση δεδομένων, τη μηχανική μάθηση και τους επιστημονικούς υπολογισμούς. Έχει γίνει μια από τις πιο ευρέως χρησιμοποιούμενες γλώσσες προγραμματισμού για την επιστήμη των δεδομένων λόγω της ευελιξίας, της ευκολίας χρήσης και των εκτεταμένων βιβλιοθηκών της.

Το Jupyter είναι ένα διαδικτυακό διαδραστικό περιβάλλον ανάπτυξης που παρέχει μια πλατφόρμα για την ανάλυση δεδομένων, τον επιστημονικό υπολογισμό και τη μηχανική μάθηση σε Python. Επιτρέπει στους χρήστες να δημιουργούν, να επεξεργάζονται και να εκτελούν κώδικα σε ένα διαδραστικό περιβάλλον. Επιπλέον. παρέχει μια ποικιλία εργαλείων για την οπτικοποίηση δεδομένων, το οποίο βοηθάει στην κατανόησή τους.

3.5 Εισαγωγή Βιβλιοθηκών

Για την προεπεξεργασία δεδομένων με τη χρήση της Python και του Jupyter, αλλά και για την εκπαίδευση και έλεγχο της επίδοσης ενός μοντέλου μηχανικής μάθησης, απαραίτητο βήμα είναι η εισαγωγή των απαιτούμενων βιβλιοθηκών.

Για την εκπαίδευση του μοντέλου αναγνώρισης δραστηριότητας έγινε η εισαγωγή των παρακάτω βιβλιοθηκών:

- Pandas - Παρέχει γρήγορες, ευέλικτες και εκφραστικές δομές δεδομένων, σχεδιασμένες για να κάνουν την εργασία με "σχεσιακά" (“relational”) ή

"επισημασμένα" (“labeled”) δεδομένα εύκολη και διαισθητική. Στόχος της είναι να αποτελέσει το θεμελιώδες δομικό στοιχείο υψηλού επιπέδου για την πρακτική ανάλυση δεδομένων στην Python.

- NumPy - Παρέχει διάφορες λειτουργίες που είναι ικανές να εκτελούν αριθμητικούς υπολογισμούς με υψηλή ταχύτητα, αλλά και διάφορες ισχυρές δομές δεδομένων, υλοποιώντας πολυδιάστατους πίνακες. Η NumPy είναι γρήγορη, γεγονός που το καθιστά λογικό να εργάζεται με ένα μεγάλο σύνολο δεδομένων.
- Matplotlib - είναι η βασική βιβλιοθήκη απεικόνισης ή σχεδίασης της γλώσσας προγραμματισμού Python και ένα ισχυρό εργαλείο για την εκτέλεση ποικίλων εργασιών. Είναι σε θέση να δημιουργεί διαφορετικούς τύπους αναφορών οπτικοποίησης, όπως γραμμικά διαγράμματα, διαγράμματα διασποράς, ιστογράμματα, ραβδογράμματα, κυκλικά διαγράμματα, ακόμα και τρισδιάστατες γραφικές παραστάσεις.
- Scikit-learn - Παρέχει μια επιλογή αποτελεσματικών εργαλείων για μηχανική μάθηση και στατιστική μοντελοποίηση, συμπεριλαμβανομένης της ταξινόμησης, της παλινδρόμησης, της ομαδοποίησης και της μείωσης της διαστατικότητας. Χρησιμοποιείται επίσης για τον διαχωρισμό των δεδομένων σε ¨εκπαίδευσης¨ και ¨ελέγχου¨.
- Keras - Είναι ένα υψηλού επιπέδου API βαθιάς μάθησης που αναπτύχθηκε από την Google για την εύκολη υλοποίηση νευρωνικών δικτύων. Υποστηρίζει επίσης πολλαπλούς υπολογισμούς νευρωνικών δικτύων backend.
- TensorFlow - είναι μια βιβλιοθήκη ανοικτού κώδικα που αναπτύχθηκε από την Google κυρίως για εφαρμογές βαθιάς μάθησης (υποστηρίζει επίσης την παραδοσιακή μηχανική μάθηση). Δέχεται δεδομένα με τη μορφή πολυδιάστατων πινάκων υψηλότερων διαστάσεων που ονομάζονται tensors. Οι πολυδιάστατοι πίνακες είναι πολύ χρήσιμοι για το χειρισμό μεγάλων ποσοτήτων δεδομένων. Το TensorFlow λειτουργεί με βάση τους γραφήματα ροής δεδομένων που έχουν κόμβους και ακμές.

```
#Importing the Libraries
import pandas as pd
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split

import keras
from tensorflow.keras import Sequential
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.layers import Dropout, Flatten, Dense
```

Εικόνα 21: Εισαγωγή Βιβλιοθηκών

3.6 Προεπεξεργασία Δεδομένων

Η προεπεξεργασία των δεδομένων είναι ένα βασικό βήμα για τη δημιουργία ενός μοντέλου μηχανικής μάθησης και αποτελείται από την προετοιμασία των ακατέργαστων δεδομένων αισθητήρων για χρήση στην εκπαίδευση ενός μοντέλου. Η διαδικασία αυτή περιλαμβάνει συνήθως τον καθαρισμό των δεδομένων, τον μετασχηματισμό των δεδομένων και την κανονικοποίηση των δεδομένων. Τα δεδομένα, επίσης, θα πρέπει στο τέλος να χωριστούν σε σύνολα εκπαίδευσης (train) και δοκιμής (test).

3.6.1 Εισαγωγή και Οπτικοποίηση Δεδομένων

Το επόμενο βήμα είναι η φόρτωση των δεδομένων σε ένα πλαίσιο δεδομένων pandas. Μόλις τα δεδομένα φορτωθούν σε ένα πλαίσιο δεδομένων, μπορούν να επιθεωρηθούν, να οπτικοποιηθούν και να ελεγχθούν για τυχόν σφάλματα.

Παρακάτω, φαίνεται η εισαγωγή, με την προσθήκη των χαρακτηριστικών των στηλών και η προβολή σε μορφή πίνακα τω δεδομένων. Επιπλέον, για να εμφανιστούν τα κατανοηθεί περισσότερο η μορφή των δεδομένων που συλλέχθηκαν, έγινε διαχωρισμός αυτών σε ομάδες, ανάλογα με την δραστηριότητα στην οποία αντιστοιχούνταν. Στη συνέχεια δημιουργήθηκαν γραφήματα τα οποία οπτικοποιούν

τις τιμές του επιταχυνσιομέτρου και του γυροσκοπίου των τεσσάρων δραστηριοτήτων. Δηλαδή, κάθισμα, ορθοστασία, περπάτημα και τρέξιμο, αντίστοιχα.

Δίνεται ο κώδικας των εργασιών αυτών:

```
#Reading CSV file with data and adding columns labels
columns = ['accX','accY','accZ','gyrX','gyrY','gyrZ','activity']
data = pd.read_csv('activities/activities_dataset.csv', names = columns)

data.head()
```

	accX	accY	accZ	gyrX	gyrY	gyrZ	activity
0	-0.219	-0.814	0.502	-44.739	22.949	-14.343	sitting
1	-0.256	-0.843	0.500	9.888	-4.395	10.071	sitting
2	-0.257	-0.846	0.500	12.268	-4.517	7.324	sitting
3	-0.258	-0.846	0.500	1.099	0.000	1.221	sitting
4	-0.257	-0.845	0.498	-0.427	0.549	0.610	sitting

```
data.shape
```

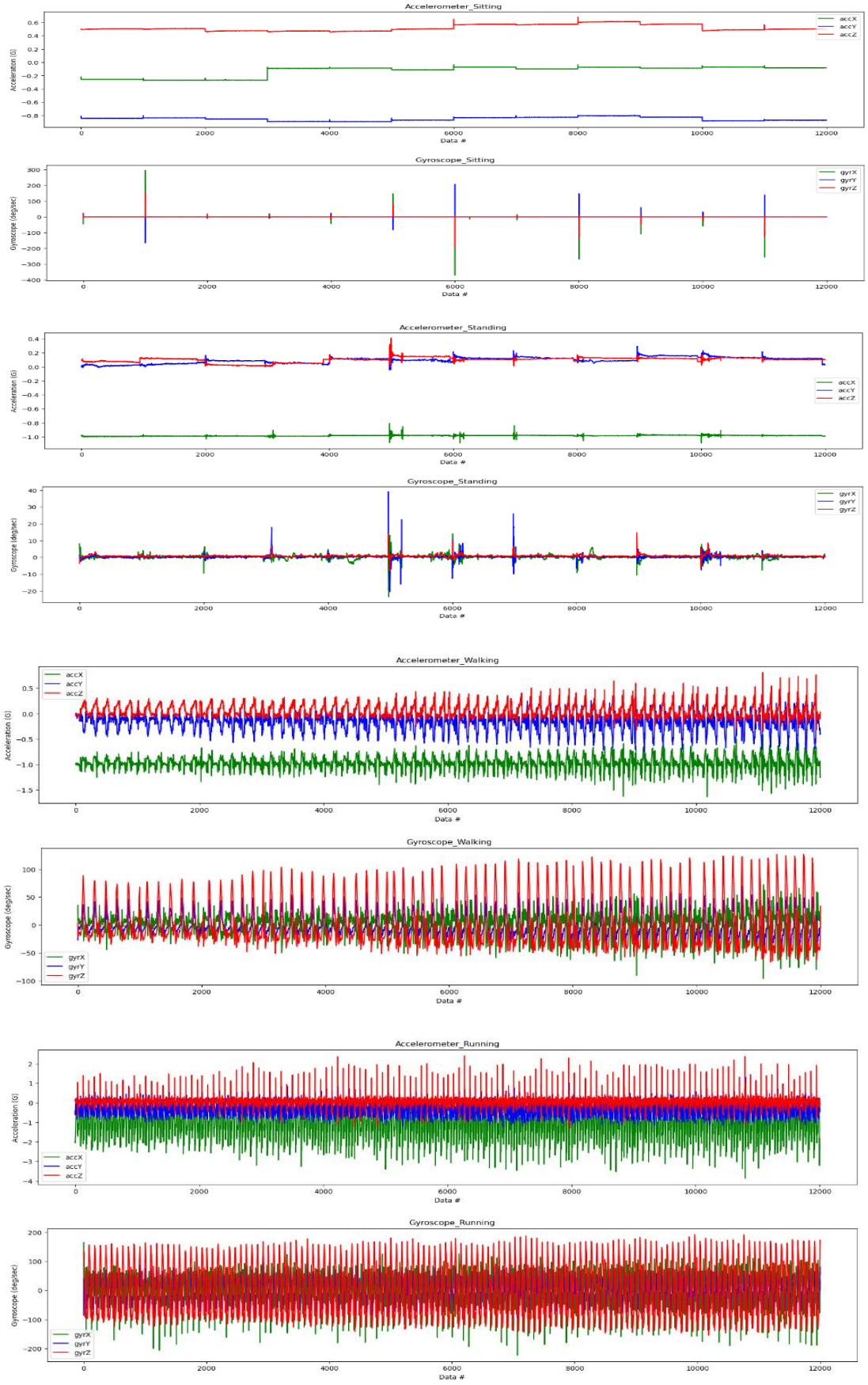
```
(240000, 7)
```

Εικόνα 22: Εισαγωγή Αρχείου Δεδομένων

```
#Grouping the data according to their labels to plot graphs
grouped = data.groupby(data.activity)
```

```
index = range(1, len(data_sitting[0:12000]) + 1)

plt.rcParams["figure.figsize"] = (20,4)

plt.plot(index, data_sitting[0:12000]['accX'], 'g.', label='accX', linestyle='solid', marker=',')
plt.plot(index, data_sitting[0:12000]['accY'], 'b.', label='accY', linestyle='solid', marker=',')
plt.plot(index, data_sitting[0:12000]['accZ'], 'r.', label='accZ', linestyle='solid', marker=',')
plt.title("Accelerometer_Sitting")
plt.xlabel("Data #")
plt.ylabel("Acceleration (G)")
plt.legend()
plt.show()

plt.plot(index, data_sitting[0:12000]['gyrX'], 'g.', label='gyrX', linestyle='solid', marker=',')
plt.plot(index, data_sitting[0:12000]['gyrY'], 'b.', label='gyrY', linestyle='solid', marker=',')
plt.plot(index, data_sitting[0:12000]['gyrZ'], 'r.', label='gyrZ', linestyle='solid', marker=',')
plt.title("Gyroscope_Sitting")
plt.xlabel("Data #")
plt.ylabel("Gyroscope (deg/sec)")
plt.legend()
plt.show()
```

Εικόνα 23: Ομαδοποίηση και Δημιουργία Γραφημάτων Δεδομένων



Εικόνα 24: Γραφήματα Δραστηριοτήτων

3.6.2 Δημιουργία Samples και Κανονικοποίηση

Για την σωστή εκπαίδευση του μοντέλου αναγνώρισης δραστηριότητας, θα πρέπει από τα δεδομένα των αισθητήρων να διεξαχθούν συγκεκριμένου μήκους δείγματα ή αλλιώς samples. Η δημιουργία των Samples περιλαμβάνει την τμηματοποίηση των δεδομένων σε μικρότερα, πιο εύχρηστα κομμάτια και την εξαγωγή σχετικών χαρακτηριστικών από κάθε τμήμα. Τα δείγματα χωρίστηκαν, λοιπόν, σε τετρακόσια (400) samples των εκατό πενήντα (150) δεδομένων της κάθε δραστηριότητας.

Με τη δημιουργία δειγμάτων, μπορεί να μειωθεί η πολυπλοκότητα των δεδομένων γεγονός που μπορεί να βελτιώσει την απόδοση του μοντέλου μηχανικής μάθησης, επιτρέποντάς του να μαθαίνει από πιο δομημένες και ουσιαστικές αναπαραστάσεις των δεδομένων.

Επιπλέον, η δημιουργία δειγμάτων μπορεί να βοηθήσει στην αντιμετώπιση του προβλήματος της ανισορροπίας των κλάσεων, το οποίο είναι σύνηθες σε σύνολα δεδομένων ανάλυσης κίνησης. Με τη δημιουργία ισορροπημένων δειγμάτων από κάθε κλάση, μπορεί να διασφαλιστεί ότι το μοντέλο μηχανικής μάθησης μαθαίνει να αναγνωρίζει εξίσου καλά κάθε κλάση, γεγονός που μπορεί να βελτιώσει τη συνολική ακρίβειά του.

Επόμενο στάδιο της προεπεξεργασίας των δεδομένων αποτέλεσε η κανονικοποίηση (normalization), η διασφάλιση, δηλαδή, ότι οι τιμές κάθε χαρακτηριστικού βρίσκονται στην ίδια κλίμακα. Αυτό βοήθησε στη βελτίωση της απόδοσης του αλγορίθμου βαθιάς μάθησης και στην αποφυγή μεροληψίας προς ορισμένα χαρακτηριστικά.

Οι τιμές των δεδομένων κλιμακώθηκαν σε ένα εύρος μεταξύ 0 και 1. Αυτό έγινε καθώς ορισμένα μοντέλα μηχανικής μάθησης, είναι ευαίσθητα στην κλίμακα των χαρακτηριστικών εισόδου. Εάν τα χαρακτηριστικά έχουν διαφορετικές κλίμακες, ο αλγόριθμος μπορεί να δώσει μεγαλύτερη βαρύτητα στα χαρακτηριστικά με μεγαλύτερες τιμές, γεγονός που μπορεί να οδηγήσει σε μη βέλτιστη απόδοση.

Για την κανονικοποίηση των τιμών σε ένα σύνολο δεδομένων ώστε να είναι μεταξύ 0 και 1, χρησιμοποιείται ο ακόλουθος τύπος:

$$z_i = (x_i - min(x)) / (max(x) - min(x))$$

```
One_Hot_Encoded_Activities = np.eye(Num_Activities)
```

```
Samples_per_Activity = 150
inputs = []
outputs = []

# read each csv file and push an input and output
for index in range(Num_Activities):
  activity = Activities[index]
  print(f"Processing index {index} for activity '{activity}'.")

  output = One_Hot_Encoded_Activities[index]

  df = pd.read_csv(os.path.join("activities", activity + ".csv"))

  # calculate the number of activity recordings in the file
  num_recordings = int(df.shape[0] / Samples_per_Activity)

  print(f"\tThere are {num_recordings} recordings of the {activity} activity.")

  for i in range(num_recordings):
    tensor = []
    for j in range(Samples_per_Activity):
      index = i * Samples_per_Activity + j
      # normalize the input data, between 0 to 1:
      tensor += [
          (df['aX'][index] + 4) / 5,
          (df['aY'][index] + 3) / 5,
          (df['aZ'][index] + 3) / 6,
          (df['gX'][index] + 370) / 740,
          (df['gY'][index] + 210) / 420,
          (df['gZ'][index] + 220) / 450
      ]

    inputs.append(tensor)
    outputs.append(output)

print('Data processing complete.')
```

```
Processing index 0 for activity 'sitting'.
        There are 400 recordings of the sitting activity.
Processing index 1 for activity 'standing'.
        There are 400 recordings of the standing activity.
Processing index 2 for activity 'walking'.
        There are 400 recordings of the walking activity.
Processing index 3 for activity 'running'.
        There are 400 recordings of the running activity.
```

Εικόνα 25: Samples και Κανονικοποίηση Δεδομένων

3.6.3 Διαχωρισμός Δεδομένων

Το τελικό στάδιο της προεπεξεργασίας των δεδομένων είναι ο διαχωρισμός τους σε δεδομένα εκπαίδευσης, δοκιμής και επικύρωσης. Σκοπός είναι η αποφυγή της υπερπροσαρμογής.

Η υπερπροσαρμογή συμβαίνει όταν ένα μοντέλο μαθαίνει το θόρυβο και τις ιδιαιτερότητες των δεδομένων εκπαίδευσης και όχι τα υποκείμενα πρότυπα που

μπορούν να γενικευτούν σε νέα, αόρατα δεδομένα. Αξιολογώντας το μοντέλο σε ένα ξεχωριστό σύνολο δοκιμών, μπορούμε να πάρουμε μια εκτίμηση του πόσο καλά θα αποδώσει το μοντέλο σε νέα δεδομένα.

Ο διαχωρισμός εκπαίδευσης - δοκιμής - επικύρωσης πραγματοποιείται συνήθως με την τυχαία ανάθεση ενός ποσοστού των δεδομένων στο σύνολο εκπαίδευσης και δοκιμής και του υπόλοιπου ποσοστού στο σύνολο επικύρωσης. Ο ακριβής ποσοστιαίος διαχωρισμός εξαρτάται από το μέγεθος του συνόλου δεδομένων και το συγκεκριμένο πρόβλημα.

Στο συγκεκριμένο πρόβλημα δόθηκε το 60% του συνόλου των δεδομένων στην εκπαίδευση (train) και το 20% στην δοκιμή (test), για την αξιολόγηση της απόδοσης του μοντέλου. Οπότε, το υπόλοιπο 20% των δεδομένων δόθηκε στην επικύρωση (validation). Το σύνολο επικύρωσης χρησιμοποιείται συνήθως για τη λεπτομερή ρύθμιση των υπερπαραμέτρων του μοντέλου, επειδή παρέχει έναν τρόπο εκτίμησης του σφάλματος γενίκευσης του μοντέλου, χωρίς τη χρήση του συνόλου δοκιμής.

Ο διαχωρισμός πραγματοποιήθηκε με την χρήση της NumPy:

```
num_inputs
```

```
1600
```

```
# Split the recordings (group of samples) into three sets: training, testing and
Train_Split = int(0.6 * num_inputs)
Test_Split = int(0.2 * num_inputs)

X_train, X_test, X_val = np.split(inputs, [Train_Split, Test_Split])
y_train, y_test, y_val = np.split(outputs, [Train_Split, Test_Split])

print("Dataset splitting complete.")
```

```
Dataset splitting complete.
```

Εικόνα 26: Διαχωρισμός σε Δεδομένα Εκπαίδευσης – Δοκιμής – Επικύρωσης

3.7 Εκπαίδευση και Απόδοση Μοντέλου

Μόλις η διαδικασία της προεπεξεργασίας των δεδομένων ολοκληρώθηκε, το επόμενο βήμα ήταν η δημιουργία ενός μοντέλου βαθιάς μάθησης για την βέλτιστη κατηγοριοποίηση των δραστηριοτήτων. Μετά τη δημιουργία του, σειρά είχε η μεταγλώττισή (compile) του και η προσαρμογή (fit) του.

Έπειτα, με τις κατάλληλες μεθόδους, σειρά είχε η μελέτη των αποτελεσμάτων του εκπαιδευμένου μοντέλου. Έτσι, ελέγχθηκε η απόδοση του κάθε μοντέλου, καθώς και το σφάλμα τους, παρατηρώντας γραφήματα ή πίνακες τιμών.

Τα δεδομένα δόθηκαν σε διάφορους τύπους μοντέλων, όπως Τεχνητά Νευρωνικά Δίκτυα (ANN) ή Συνελικτικά Νευρωνικά Δίκτυα (CNN). Τα μοντέλα, μετά την εκπαίδευσή τους, συγκρίθηκαν μεταξύ τους, για να βρεθεί η καλύτερη απόδοση (accuracy), με τη μικρότερη δυνατή απώλεια (loss).

3.7.1 Εκπαίδευση με ANN

Για την εύρεση του βέλτιστου μοντέλου δοκιμάστηκαν πολλές περιπτώσεις τεχνητών νευρωνικών δικτύων, με διαφορετικούς αριθμούς νευρώνων ή στρωμάτων, το καθένα.

Προσπάθεια 1η:

Ένα από τα καλύτερα μοντέλα, όσο αφορά την υψηλή απόδοση και την χαμηλή απώλεια, αποτελεί το παρακάτω:

```
model = tf.keras.Sequential([
    tf.keras.layers.Dense(16, activation='relu'),

    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dropout(0.2),

    tf.keras.layers.Dense(128, activation='relu'),

    tf.keras.layers.Dense(NUM_GESTURES, activation='softmax')
])
```

Εικόνα 27: Δημιουργία 1ου Μοντέλου

Το μοντέλο είναι ένα διαδοχικό (sequential) μοντέλο, που σημαίνει ότι τα στρώματα προστίθενται ένα προς ένα με γραμμικό τρόπο.

Το μοντέλο έχει τέσσερα στρώματα. Ένα πυκνό στρώμα με 16 νευρώνες και μια συνάρτηση ενεργοποίησης ReLU (Rectified Linear Unit). Ένα πυκνό στρώμα με 64 νευρώνες και μια συνάρτηση ενεργοποίησης ReLU, ακολουθούμενο από ένα στρώμα εγκατάλειψης που θέτει τυχαία το 20% των μονάδων εισόδου στο 0 κατά τη διάρκεια της εκπαίδευσης για να αποτρέψει την υπερπροσαρμογή. Ένα πυκνό στρώμα με 128 νευρώνες και μια συνάρτηση ενεργοποίησης ReLU. Ένα πυκνό στρώμα με αριθμό νευρώνων ίσο με τον αριθμό των δραστηριοτήτων που ταξινομούνται και μια συνάρτηση ενεργοποίησης softmax, η οποία παράγει μια κατανομή πιθανότητας πάνω στις κλάσεις.

Σκοπός του μοντέλου είναι να εκτελεί ταξινόμηση δραστηριοτήτων, δεδομένων ορισμένων δεδομένων εισόδου. Τα δεδομένα εισόδου θα πρέπει να περάσουν στο πρώτο στρώμα, και η έξοδος του τελικού στρώματος θα είναι μια κατανομή πιθανότητας πάνω στις διάφορες δραστηριότητες. Το μοντέλο θα εκπαιδευτεί χρησιμοποιώντας μια κατάλληλη συνάρτηση απώλειας (loss function) και έναν βελτιστοποιητή (optimizer) για την ελαχιστοποίηση της απώλειας, με στόχο τη βελτίωση της ικανότητας του μοντέλου να ταξινομεί σωστά τις χειρονομίες.

```
model.compile(optimizer=Adam(learning_rate = 0.0001),
              loss='categorical_crossentropy',
              metrics=['accuracy'])
```

Το συγκεκριμένο απόσπασμα κώδικα διαμόρφωσε το μοντέλο για εκπαίδευση χρησιμοποιώντας τον βελτιστοποιητή Adam με χαμηλό ρυθμό εκμάθησης και βελτιστοποιώντας την απώλεια κατηγορικής διασταυρούμενης εντροπίας (categorical crossentropy), ενώ χρησιμοποιήθηκε η ακρίβεια (accuracy) ως μετρική αξιολόγησης.

- Ο βελτιστοποιητής Adam υπολογίζει προσαρμοστικούς ρυθμούς μάθησης για κάθε παράμετρο, με βάση τις εκτιμήσεις της πρώτης και της δεύτερης στιγμής των κλίσεων. Ο ρυθμός μάθησης (learning rate) καθορίζει το μέγεθος του βήματος που λαμβάνει ο βελτιστοποιητής προς την κατεύθυνση της αρνητικής κλίσης της συνάρτησης απώλειας και ένας μικρότερος ρυθμός μάθησης σημαίνει ότι ο βελτιστοποιητής λαμβάνει μικρότερα βήματα στο χώρο των παραμέτρων.

- Η συνάρτηση απώλειας categorical crossentropy μετρά τη διαφορά μεταξύ των προβλεπόμενων πιθανοτήτων κλάσης και των πραγματικών πιθανοτήτων κλάσης. Η συνάρτηση απώλειας ενθαρρύνει το μοντέλο να αποδίδει υψηλές πιθανότητες στη σωστή κλάση και χαμηλές πιθανότητες στις λανθασμένες κλάσεις.
- Η μετρική της ακρίβειας είναι το κλάσμα των σωστά ταξινομημένων παραδειγμάτων επί του συνόλου των παραδειγμάτων του συνόλου δεδομένων. Κατά τη διάρκεια της εκπαίδευσης, το μοντέλο θα εξάγει τη μετρική ακρίβειας στο τέλος κάθε εποχής (epoch), η οποία δίνει μια εκτίμηση του πόσο καλά τα πάει το μοντέλο στα δεδομένα εκπαίδευσης.

```
history = model.fit(inputs_train, outputs_train, epochs=100, batch_size=1, validation_data=(inputs_validate, outputs_validate))
```

Κατά τη διάρκεια της εκπαίδευσης, το μοντέλο ενημερώνει τις παραμέτρους του χρησιμοποιώντας τον βελτιστοποιητή που καθορίζεται στη μέθοδο compile() και τη συνάρτηση απώλειας που καθορίζεται στην ίδια μέθοδο. Η μέθοδος fit() επιστρέφει ένα αντικείμενο history που περιέχει πληροφορίες σχετικά με τη διαδικασία εκπαίδευσης, όπως η απώλεια και η ακρίβεια του μοντέλου σε κάθε εποχή.

Στο συγκεκριμένο μοντέλο ορίστηκαν 100 epoch, δηλαδή, τόσες φορές που το μοντέλο επαναλαμβάνει ολόκληρο το σύνολο εκπαίδευσης, καθώς ο ρυθμός μάθησης είναι μικρός. Επίσης, ορίζοντας το batch size = 1, χρησιμοποιείται 1 παράδειγμα σε κάθε παρτίδα εκπαίδευσης, πράγμα που σημαίνει ότι το μοντέλο θα ενημερώνει τις παραμέτρους του μετά από κάθε παράδειγμα.

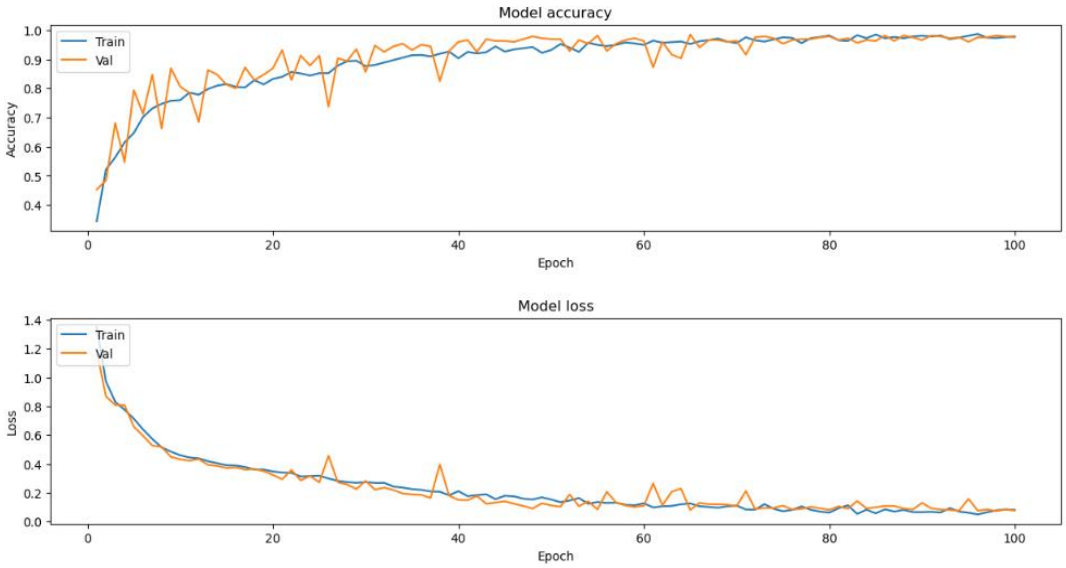
```
Epoch 95/100
960/960 [==============================] - 1s 929us/step - loss: 0.0618 - accuracy: 0.9802 - val_loss: 0.1585 - val_accuracy:
0.9594
Epoch 96/100
960/960 [==============================] - 1s 915us/step - loss: 0.0503 - accuracy: 0.9865 - val_loss: 0.0760 - val_accuracy:
0.9750
Epoch 97/100
960/960 [==============================] - 1s 942us/step - loss: 0.0652 - accuracy: 0.9740 - val_loss: 0.0843 - val_accuracy:
0.9750
Epoch 98/100
960/960 [==============================] - 1s 920us/step - loss: 0.0784 - accuracy: 0.9729 - val_loss: 0.0738 - val_accuracy:
0.9812
Epoch 99/100
960/960 [==============================] - 1s 920us/step - loss: 0.0846 - accuracy: 0.9760 - val_loss: 0.0850 - val_accuracy:
0.9781
Epoch 100/100
960/960 [==============================] - 1s 914us/step - loss: 0.0817 - accuracy: 0.9781 - val_loss: 0.0770 - val_accuracy:
0.9750
```

Εικόνα 28: Αποτελέσματα 1ου Μοντέλου

Το μοντέλο στο τελευταίο epoch έδωσε 97% ακρίβεια, ενώ η απώλεια ήταν στο 8%, γεγονός που σημαίνει πως τα αποτελέσματα ήταν πολύ ικανοποιητικά.

```
def plot_learningCurve(history, epochs):
  epoch_range = range(1, epochs+1)
  plt.plot(epoch_range, history.history['accuracy'])
  plt.plot(epoch_range, history.history['val_accuracy'])
  plt.title('Model accuracy')
  plt.ylabel('Accuracy')
  plt.xlabel('Epoch')
  plt.legend(['Train', 'Val'], loc='upper left')
  plt.show()

  plt.plot(epoch_range, history.history['loss'])
  plt.plot(epoch_range, history.history['val_loss'])
  plt.title('Model loss')
  plt.ylabel('Loss')
  plt.xlabel('Epoch')
  plt.legend(['Train', 'Val'], loc='upper left')
  plt.show()
```

Τελικό στάδιο της διαδικασίας εκπαίδευσης του μοντέλου αποτελεί η αναπαράσταση των γραφικών παραστάσεων της ακρίβειας και της απώλειας, ώστε να οπτικοποιηθούν τα αποτελέσματα.



Εικόνα 29: Γραφήματα Αποτελεσμάτων 1ου Μοντέλου

Επιπλέον, το μοντέλο χρησιμοποιήθηκε για να γίνουν προβλέψεις σε νέα δεδομένα, με τα οποία δεν είχε εκπαιδευτεί, και τα αποτελέσματα ήταν εξίσου ικανοποιητικά.

```
predictions = model.predict(inputs_test)
# print the predictions and the expected ouputs
print("predictions =\n", np.round(predictions, decimals=3))
print("actual =\n", outputs_test)
```

```
predictions =
 [[0.    0.985 0.014 0.   ]
 [1.    0.    0.    0.   ]
 [0.    0.018 0.975 0.007]
 ...
 [0.    0.157 0.843 0.   ]
 [0.    0.944 0.056 0.   ]
 [0.    0.    0.941 0.059]]
actual =
 [[0. 1. 0. 0.]
 [1. 0. 0. 0.]
 [0. 0. 1. 0.]
 ...
 [0. 0. 1. 0.]
 [0. 1. 0. 0.]
 [0. 0. 1. 0.]]
```

Εικόνα 30: Προβλέψεις 1ου Μοντέλου

Προσπάθεια 2η:

Στην δεύτερη προσπάθεια δημιουργίας και εκπαίδευσης ενός τεχνητού νευρωνικού δικτύου, χρησιμοποιήθηκαν 4 στρώματα, αλλά λιγότεροι νευρώνες.

```
model = tf.keras.Sequential([
    tf.keras.layers.Dense(8, activation='relu'),

    tf.keras.layers.Dense(16, activation='relu'),

    tf.keras.layers.Dense(32, activation='relu'),

    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dropout(0.2),

    tf.keras.layers.Dense(Num_Activities, activation='softmax')
])
```

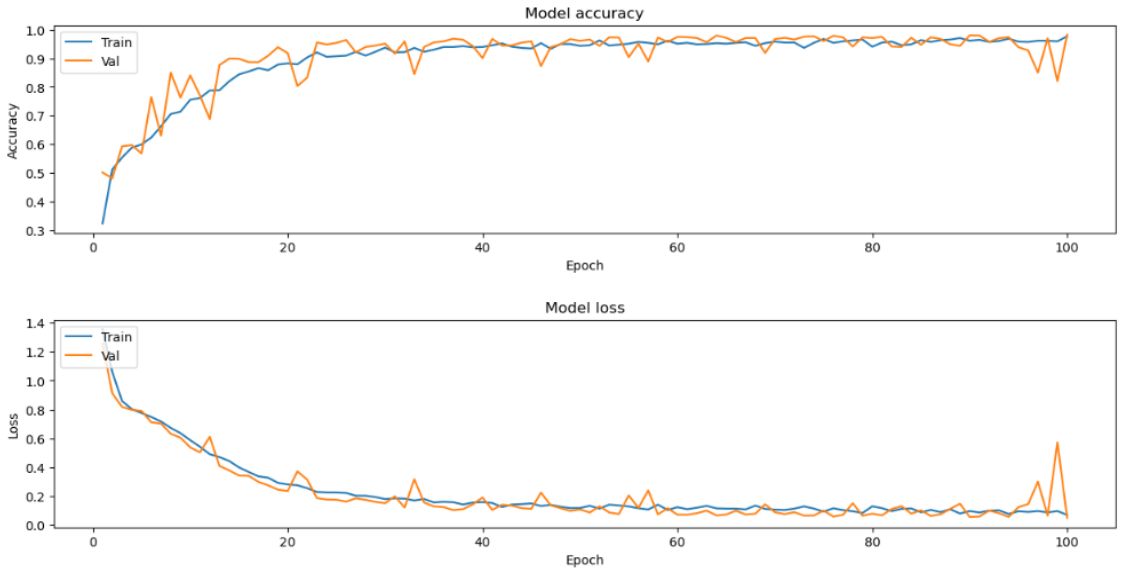
Εικόνα 31: Δημιουργία 2ουΜοντέλου

Αφήνοντας τα υπόλοιπα χαρακτηριστικά και συναρτήσεις ίδια, όπως και στην προηγούμενη προσπάθεια, τα αποτελέσματα ήταν παρόμοια, ίσως και λίγο καλύτερα.

```
Epoch 95/100
960/960 [==============================] - 1s 2ms/step - loss: 0.0953 - accuracy: 0.9583 - val_loss: 0.1239 - val_accuracy:
0.9391
Epoch 96/100
960/960 [==============================] - 1s 2ms/step - loss: 0.0907 - accuracy: 0.9573 - val_loss: 0.1449 - val_accuracy:
0.9273
Epoch 97/100
960/960 [==============================] - 1s 2ms/step - loss: 0.0965 - accuracy: 0.9615 - val_loss: 0.2998 - val_accuracy:
0.8500
Epoch 98/100
960/960 [==============================] - 1s 1ms/step - loss: 0.0872 - accuracy: 0.9615 - val_loss: 0.0645 - val_accuracy:
0.9703
Epoch 99/100
960/960 [==============================] - 1s 2ms/step - loss: 0.0966 - accuracy: 0.9594 - val_loss: 0.5727 - val_accuracy:
0.8203
Epoch 100/100
960/960 [==============================] - 1s 1ms/step - loss: 0.0698 - accuracy: 0.9771 - val_loss: 0.0490 - val_accuracy:
0.9828
```

Εικόνα 32: Αποτελέσματα 2ου Μοντέλου

Η ακρίβεια παρέμεινε υψηλή, κοντά στο 98%, ενώ η απώλεια έπεσε στο 7% περίπου, δίνοντας τα παρακάτω διαγράμματα:



Εικόνα 33: Γραφήματα Αποτελεσμάτων 2ου Μοντέλου

Παράλληλα, οι προβλέψεις σε νέα, μη ορατά στην εκπαίδευση δεδομένα, το μοντέλο είχε πολύ κοντινές προβλέψεις σε αυτές των πραγματικών τιμών.

```
predictions =
 [[1.    0.    0.    0.   ]
 [1.    0.    0.    0.   ]
 [0.    0.002 0.998 0.   ]
 ...
 [0.    0.    0.999 0.001]
 [1.    0.    0.    0.   ]
 [1.    0.    0.    0.   ]]
actual =
 [[1. 0. 0. 0.]
 [1. 0. 0. 0.]
 [0. 0. 1. 0.]
 ...
 [0. 0. 1. 0.]
 [1. 0. 0. 0.]
 [1. 0. 0. 0.]]
```

Εικόνα 34: Προβλέψεις 2ου Μοντέλου

3.7.2 Εκπαίδευση με CNN

Το επόμενο μοντέλο που δοκιμάστηκε ήταν ένα συνελικτικό νευρωνικό δίκτυο. Το συγκεκριμένο είναι ένα CNN, με είσοδο 1D, το οποίο χρησιμοποιείται συχνά για δεδομένα χρονοσειρών, όπως στην περίπτωση αυτή.

Πιο συγκεκριμένα, σε ένα CNN 1D, το στρώμα συνέλιξης ολισθαίνει ένα φίλτρο κατά μήκος του σήματος και εκτελεί ένα γινόμενο μεταξύ των βαρών του φίλτρου και των τιμών του σήματος σε κάθε θέση. Έτσι παράγεται ένας χάρτης χαρακτηριστικών που αποτυπώνει την παρουσία ορισμένων μοτίβων ή χαρακτηριστικών στην ακολουθία εισόδου. Ο χάρτης χαρακτηριστικών (feature map) εξόδου καθορίζεται από τον αριθμό των φίλτρων που χρησιμοποιούνται, το μέγεθος των φίλτρων, το βήμα της συνέλιξης και το γέμισμα.

```
model = keras.Sequential([
    keras.layers.Conv1D(filters=16, kernel_size=3, activation='relu', input_shape=(X_train.shape[1], X_train.shape[2])),
    keras.layers.MaxPooling1D(pool_size=2),
    keras.layers.Flatten(),
    keras.layers.Dropout(0.2),

    keras.layers.Dense(32, activation='relu'),
    keras.layers.Dropout(0.2),

    keras.layers.Dense(num_classes, activation='softmax')])
```

Εικόνα 35: Δημιουργία 3ουΜοντέλου

Το μοντέλο αποτελείται από διάφορα επίπεδα. Ένα στρώμα συνελικτικού 1D με 16 φίλτρα, μέγεθος πυρήνα (kernel) 3 και συνάρτηση ενεργοποίησης ReLU. Το σχήμα εισόδου ορίζεται ως (X_train.shape[1], X_train.shape[2]), πράγμα που σημαίνει ότι το μοντέλο αναμένει ακολουθίες εισόδου σχήματος (χρονικά βήματα (timesteps), χαρακτηριστικά (features)), δηλαδή (150, 6).

Επίσης, έχει ένα στρώμα max pooling με μέγεθος pool 2, το οποίο μειώνει το μέγεθος της εξόδου του προηγούμενου στρώματος. Ένα επίπεδο ισοπέδωσης (Flatten layer), το οποίο μετατρέπει την έξοδο του προηγούμενου επιπέδου σε ένα διάνυσμα 1D που μπορεί να τροφοδοτηθεί σε ένα πλήρως συνδεδεμένο επίπεδο. Ένα στρώμα εγκατάλειψης (Dropout layer) με ρυθμό 0,2, το οποίο εγκαταλείπει τυχαία το 20% της εξόδου του προηγούμενου στρώματος κατά τη διάρκεια της εκπαίδευσης για να αποτρέψει την υπερπροσαρμογή. Ένα πλήρως συνδεδεμένο στρώμα με 32 μονάδες και συνάρτηση ενεργοποίησης ReLU, καθώς και άλλο ένα στρώμα εγκατάλειψης με ρυθμό 0,2.

Τέλος, αποτελείται από ένα πυκνό στρώμα με μονάδες num_classes, δηλαδή 4, και μια συνάρτηση ενεργοποίησης softmax, η οποία εξάγει τις προβλεπόμενες πιθανότητες για κάθε κλάση.

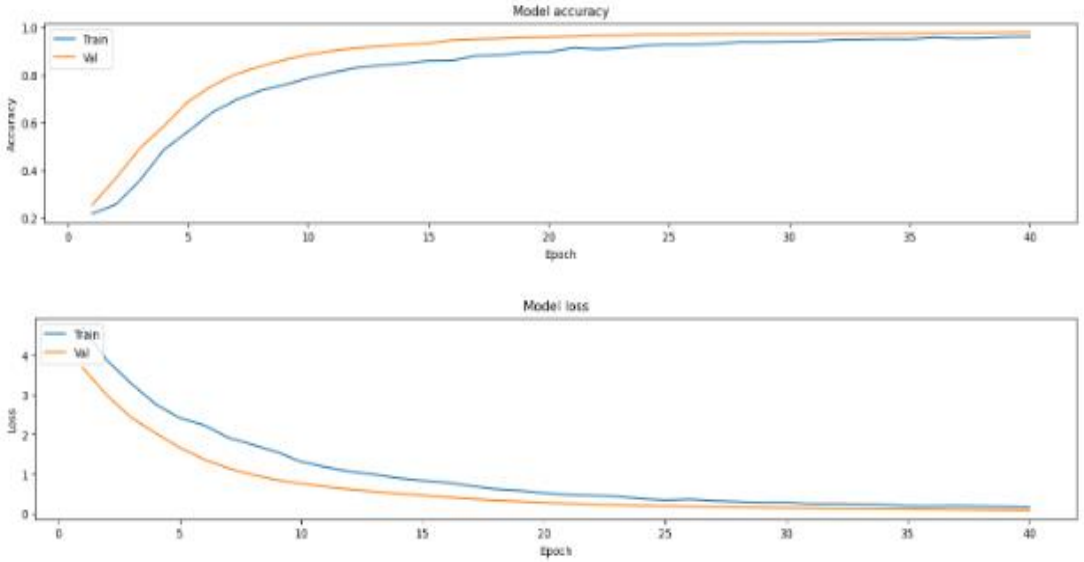
Το μοντέλο συντάσσεται με μια κατηγορική συνάρτηση απώλειας cross-entropy και έναν βελτιστοποιητή Adam, και θα εκπαιδευτεί σε ακολουθίες εισόδου σχήματος (χρονικά βήματα, χαρακτηριστικά) για να προβλέψει μία από τις num_classes πιθανές κλάσεις.

Μετά την εκπαίδευση του μοντέλου τα αποτελέσματα είναι αρκετά καλά, καθώς φτάνουν το 97% ακρίβειας, ενώ η απώλεια είναι στο 9%.

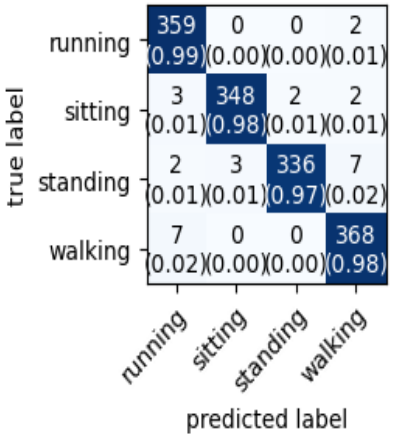
```
Epoch 95/100
105/105 [==============================] - 0s 4ms/step - loss: 0.1028 - accuracy: 0.9684 - val_loss: 0.0657 - val_accuracy:
0.9903
Epoch 96/100
105/105 [==============================] - 0s 4ms/step - loss: 0.0892 - accuracy: 0.9717 - val_loss: 0.0658 - val_accuracy:
0.9903
Epoch 97/100
105/105 [==============================] - 0s 5ms/step - loss: 0.1003 - accuracy: 0.9735 - val_loss: 0.0653 - val_accuracy:
0.9903
Epoch 98/100
105/105 [==============================] - 0s 4ms/step - loss: 0.1052 - accuracy: 0.9696 - val_loss: 0.0664 - val_accuracy:
0.9903
Epoch 99/100
105/105 [==============================] - 0s 4ms/step - loss: 0.0923 - accuracy: 0.9735 - val_loss: 0.0653 - val_accuracy:
0.9903
Epoch 100/100
105/105 [==============================] - 0s 4ms/step - loss: 0.0958 - accuracy: 0.9735 - val_loss: 0.0654 - val_accuracy:
0.9903
```

Εικόνα 36: Αποτελέσματα 3ου Μοντέλου

Στη συνέχεια έγινε εξαγωγή των γραφημάτων ακρίβειας και απώλειας, ενώ δημιουργήθηκαν παράλληλα ένας πίνακας σύγχυσης (confusion matrix) και η αναφορά ταξινόμησης (classification report) με τα αποτελέσματα.



Εικόνα 37: Γραφήματα Αποτελεσμάτων 3ου Μοντέλου



	precision	recall	f1-score	support
sitting	0.97	0.99	0.98	361
standing	0.99	0.98	0.99	355
walking	0.99	0.97	0.98	348
running	0.97	0.98	0.98	375
accuracy			0.98	1439
macro avg	0.98	0.98	0.98	1439
weighted avg	0.98	0.98	0.98	1439

Εικόνα 38: Πίνακας Σύγχυσης και Αναφορά Ταξινόμησης 3ου Μοντέλου

3.8 Μετατροπή σε TensorFlow Lite

Μετά την ολοκλήρωση της διαδικασίας εκπαίδευσης των μοντέλων, σειρά είχε η τροποποίηση του κατάλληλου για την υλοποίηση του συστήματος αναγνώρισης δραστηριότητας. Το μοντέλο, θα έπρεπε να εφαρμοστεί στο Arduino Nano 33 BLE Sense, οπότε χρειάζεται να τηρεί κάποιες προδιαγραφές.

Ένα από τα σημαντικότερα χαρακτηριστικά είναι το μέγεθος του μοντέλου, ο χώρος δηλαδή που καταναλώνει στην μνήμη της συσκευής. Οπότε, η μετατροπή του σε TensorFlow Lite μορφή, είναι απαραίτητη.

Η μετατροπή ενός μοντέλου μηχανικής μάθησης σε μορφή TensorFlow Lite περιλαμβάνει τη μετατροπή του μοντέλου σε μορφή που είναι βελτιστοποιημένη για εκτέλεση σε μικρές συσκευές και, στη συνέχεια, τη χρήση του μετατροπέα TensorFlow Lite για τη μετατροπή του μοντέλου σε δυαδική μορφή που μπορεί να φορτωθεί και να εκτελεστεί στη συσκευή.

Μόλις το μοντέλο μετατραπεί σε μορφή TensorFlow Lite, μπορεί να αναπτυχθεί και να εκτελεστεί στο Nano 33 BLE Sense, επιτρέποντάς του να εκτελεί εργασίες μηχανικής μάθησης απευθείας στη συσκευή.

Το μέγεθος ενός μοντέλου μηχανικής μάθησης που μπορεί να αναπτυχθεί σε ένα Nano 33 BLE Sense εξαρτάται από τη διαθέσιμη μνήμη και την αποθηκευτική ικανότητα της συσκευής, καθώς και από την πολυπλοκότητα του ίδιου του μοντέλου. Το Nano 33 BLE Sense διαθέτει 256 KB μνήμης RAM και 1 MB μνήμης Flash, γεγονός που περιορίζει το μέγεθος του μοντέλου που μπορεί να φορτωθεί στη συσκευή.

```
# Convert the model to the TensorFlow Lite format without quantization
converter = tf.lite.TFLiteConverter.from_keras_model(model)
tflite_model = converter.convert()

# Save the model to disk
open("activities_model.tflite", "wb").write(tflite_model)

import os
basic_model_size = os.path.getsize("activities_model.tflite")
print("Model is %d bytes" % basic_model_size)
```

```
Model is 43444 bytes
```

Εικόνα 39: Μετατροπή σε TensorFlow Lite μορφή

Στη συνέχεια, το μοντέλο, εφόσον θα εφαρμοστεί στο Arduino, μέσω του IDE, μετατράπηκε σε πίνακα κατάλληλο για προγραμματισμό στην γλώσσα C. Τέλος, αποθηκεύτηκε σαν αρχείο με ονομασία activities_model.h, ώστε να μπορεί να συμπεριληφθεί στον κώδικα του Arduino.

```
# Function: Convert some hex value into an array for C programming
def hex_to_c_array(hex_data, var_name):

  c_str = ''

  # Create header guard
  c_str += '#ifndef ' + var_name.upper() + '_H\n'
  c_str += '#define ' + var_name.upper() + '_H\n\n'

  # Add array length at top of file
  c_str += '\nunsigned int ' + var_name + '_len = ' + str(len(hex_data)) + ';\n'

  # Declare C variable
  c_str += 'unsigned char ' + var_name + '[] = {'
  hex_array = []
  for i, val in enumerate(hex_data) :

    # Construct string from hex
    hex_str = format(val, '#04x')

    # Add formatting so each line stays within 80 characters
    if (i + 1) < len(hex_data):
      hex_str += ','
    if (i + 1) % 12 == 0:
      hex_str += '\n '
    hex_array.append(hex_str)

  # Add closing brace
  c_str += '\n ' + format(' '.join(hex_array)) + '\n};\n\n'

  # Close out header guard
  c_str += '#endif //' + var_name.upper() + '_H'

  return c_str
```

```
c_model_name = 'activities_model'
```

```
# Write TFLite model to a C source (or header) file
with open(c_model_name + '.h', 'w') as file:
  file.write(hex_to_c_array(tflite_model, c_model_name))
```

Εικόνα 40: Δημιουργία Αρχείου Τελικού Μοντέλου για Γλώσσα C

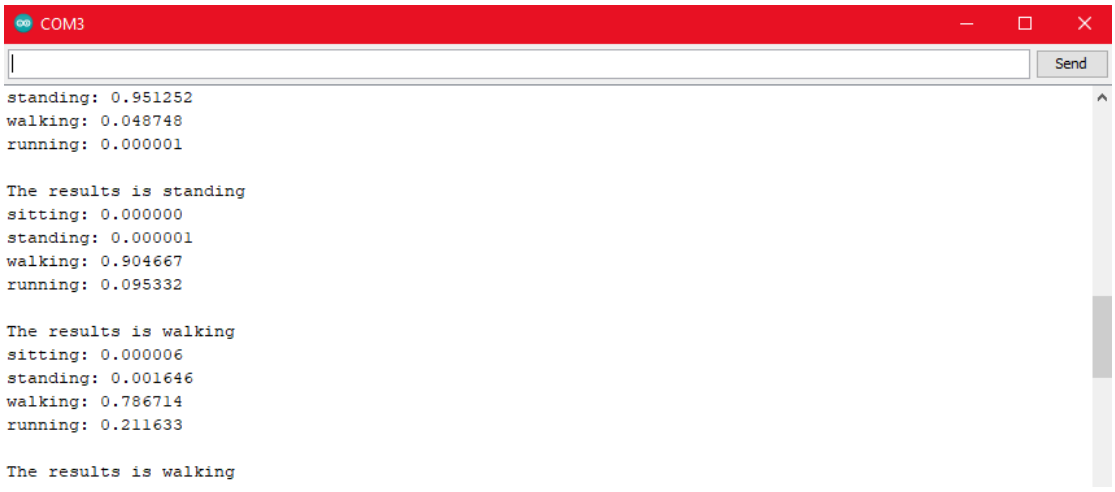
3.9 Σχεδιασμός Συστήματος μέσω Arduino IDE

Τελευταίο βήμα για την ολοκλήρωση του συστήματος αναγνώρισης δραστηριότητας, αποτελεί η εισαγωγή του μοντέλου στο Arduino IDE, για την ταξινόμηση των δραστηριοτήτων σε πραγματικό χρόνο, μέσω της συσκευής. Εκτός από την εισαγωγή του μοντέλου, το σκίτσο (sketch) ξεκινά με την ενσωμάτωση των απαραίτητων βιβλιοθηκών, οι οποίες είναι η βιβλιοθήκη Arduino_LSM9DS1 για τη διασύνδεση με την IMU και η βιβλιοθήκη TensorFlowLite για την εκτέλεση της μηχανικής μάθησης.

Στη συνέχεια, ορίζονται ορισμένες μεταβλητές, όπως η ευαισθησία εκκίνησης και ο αριθμός των δειγμάτων προς ανάγνωση. Ορίζεται επίσης ένας αναμεταδότης σφαλμάτων (error reporter) για χρήση στο TensorFlow Lite.

Το αρχείο μοντέλου φορτώνεται χρησιμοποιώντας τη συνάρτηση TensorFlowLite GetModel(), η οποία μετατρέπει την αναπαράσταση του μοντέλου από πίνακα byte σε μοντέλο TensorFlow Lite. Στη συνέχεια δημιουργείται ένας διερμηνέας (interpreter) για την εκτέλεση του μοντέλου και κατανέμεται μνήμη για τους tensors εισόδου και εξόδου του μοντέλου.

Στον κύριο βρόχο (main loop), το σκίτσο περιμένει να ανιχνευτεί σημαντική κίνηση από την IMU. Μόλις ανιχνευθεί σημαντική κίνηση, ο βρόχος διαβάζει δεδομένα επιτάχυνσης και γυροσκοπίου από την IMU και τα αποθηκεύει στον tensor εισόδου του μοντέλου. Μόλις διαβαστούν αρκετά δείγματα, το μοντέλο εκτελείται χρησιμοποιώντας τη συνάρτηση Invoke( ) του διερμηνέα και η προβλεπόμενη δραστηριότητα εκτυπώνεται στη σειριακή οθόνη (serial monitor).



Εικόνα 41: Εμφάνιση Πρόβλεψης Δραστηριότητας

3.9.1 Σχεδιασμός Hardware Συστήματος

Όσον αφορά το κομμάτι του σχεδιασμού του hardware του συστήματος, το σκίτσο περιλαμβάνει, αρχικά, μια συνάρτηση Volts( ), η οποία διαβάζει μια αναλογική τάση από τον ακροδέκτη A0 και τη μετατρέπει σε μια τιμή τάσης float. Αυτή η συνάρτηση καλείται στον κύριο βρόχο για να διαβάσει την τιμή της τάσης και να την εκτυπώσει στη σειριακή οθόνη.

Η λειτουργία αυτή αποσκοπεί στην αναγνώριση της τάσης μίας εξωτερικής μπαταρίας 9V, η οποία είναι συνδεδεμένη στη συσκευή με την προσθήκη και ενός διακόπτη (slide switch), για την ενεργοποίηση ή απενεργοποίηση της. Η μπαταρία συνδέεται, με τη βοήθεια ενός διαιρέτη τάσης και τις απαραίτητες αντιστάσεις (R1=10K, R2=20K), στον ακροδέκτη A0. Ένας διαιρέτης τάσης είναι ένα απλό κύκλωμα που μπορεί να μειώσει την τάση της μπαταρίας σε ένα επίπεδο που μπορεί να διαβάσει το Arduino. Αποτελείται από δύο αντιστάσεις σε σειρά, με την τάση της μπαταρίας να εφαρμόζεται σε ολόκληρο το κύκλωμα.

Επιλέγοντας τις κατάλληλες τιμές για τις αντιστάσεις, μπορεί να δημιουργηθεί ένας διαιρέτης τάσης που θα παράγει μια τάση εξόδου ανάλογη με την τάση εισόδου, αλλά σε χαμηλότερο επίπεδο. Με τον τρόπο αυτό, η συσκευή καταλαβαίνει πότε η μπαταρία είναι χαμηλή και να ανάβει κάποιο LED για την ειδοποίηση του χρήστη.

Επιπλέον, ορισμένοι ακροδέκτες ορίζονται ως έξοδοι και τίθενται σε LOW στη συνάρτηση setup( ). Οι συγκεκριμένοι ακροδέκτες αποτελούνται από το LED ένδειξης της μπαταρίας και ένα RGB LED, το οποίο ανάβει (HIGH) ανάλογα με την κάθε δραστηριότητα η οποία αναγνωρίζεται. Δηλαδή, πράσινο σταθερό για ορθοστασία, πράσινο που ανάβει και σβήνει για το κάθισμα, μπλε για το περπάτημα και κόκκινο για το τρέξιμο. Σημαντική είναι και η προσθήκη των επιθυμητών αντιστάσεων (220 Ohm), για την ανάλογη φωτεινότητα στις συνδέσεις των LED για την αποφυγή βλάβης.

Τέλος, για την εξωτερική εμφάνιση της συσκευής κατασκευάστηκε μία θήκη – βάση για το Arduino Nano 33 BLE Sense, την μπαταρία, καθώς και για όλο το υπόλοιπο κύκλωμα. Σε συνδυασμό με μία ζώνη, η ολοκληρωμένη συσκευή αναγνώρισης δραστηριότητας, παραμένει σταθερή επάνω στον μηρό του χρήστη.

Εικόνα 42: Hardware Συστήματος

Εικόνα 43: Τελική Εμφάνιση Συσκευής

Παρακάτω δίνεται το sketch του Arduino IDE με τον κώδικα που χρησιμοποιήθηκε για την κατηγοριοποίηση των δραστηριοτήτων σε πραγματικό χρόνο, καθώς και για την ειδοποίηση του χρήστη για την αλλαγή μεταξύ αυτών:

classify_activities | activities_model.h

```
#include <Arduino_LSM9DS1.h>

#include <TensorFlowLite.h>
#include <tensorflow/lite/micro/all_ops_resolver.h>
#include <tensorflow/lite/micro/micro_error_reporter.h>
#include <tensorflow/lite/micro/micro_interpreter.h>
#include <tensorflow/lite/schema/schema_generated.h>
#include <tensorflow/lite/version.h>

#include "activities_model.h"

const float accelerationThreshold = 1.0; // threshold of significant in G's
const int numSamples = 150;

int samplesRead = numSamples;

// global variables used for TensorFlow Lite (Micro)
tflite::MicroErrorReporter tflErrorReporter;

// pull in all the TFLM ops
tflite::AllOpsResolver tflOpsResolver;

const tflite::Model* tflModel = nullptr;
tflite::MicroInterpreter* tflInterpreter = nullptr;
TfLiteTensor* tflInputTensor = nullptr;
TfLiteTensor* tflOutputTensor = nullptr;

// Create a static memory buffer for TFLM, the size may need to
// be adjusted based on the model you are using
constexpr int tensorArenaSize = 15 * 1024;
byte tensorArena[tensorArenaSize];

// array to map gesture index to a name
const char* Activities[] = {
  "sitting",
  "standing",
  "walking",
  "running"
};

#define Num_Activities (sizeof(Activities) / sizeof(Activities[0]))
```

```
int analogvoltage;
float voltage;

void setup() {
  pinMode(2, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);

  digitalWrite(2, LOW);
  pinMode(A0, INPUT);

  Serial.begin(9600);
//  while (!Serial);

  // initialize the IMU
  if (!IMU.begin()) {
    Serial.println("Failed to initialize IMU!");
    while (1);
  }

  // print out the samples rates of the IMUs
  Serial.print("Accelerometer sample rate = ");
  Serial.print(IMU.accelerationSampleRate());
  Serial.println(" Hz");
  Serial.print("Gyroscope sample rate = ");
  Serial.print(IMU.gyroscopeSampleRate());
  Serial.println(" Hz");

  Serial.println();

  // get the TFL representation of the model byte array
  tflModel = tflite::GetModel(activities_model);
  if (tflModel->version() != TFLITE_SCHEMA_VERSION) {
    Serial.println("Model schema mismatch!");
    while (1);
  }

  // Create an interpreter to run the model
  tflInterpreter = new tflite::MicroInterpreter(tflModel, tflOpsResolver, tensorArena, tensorArenaSize, &tflErrorReporter);

  // Allocate memory for the model's input and output tensors
  tflInterpreter->AllocateTensors();

  // Get pointers for the model's input and output tensors
  tflInputTensor = tflInterpreter->input(0);
  tflOutputTensor = tflInterpreter->output(0);
}


void loop() {
  float aX, aY, aZ, gX, gY, gZ;
  Volts();

  // wait for significant motion
  while (samplesRead == numSamples) {
    if (IMU.accelerationAvailable()) {
      // read the acceleration data
      IMU.readAcceleration(aX, aY, aZ);

      // sum up the absolutes
      float aSum = fabs(aX) + fabs(aY) + fabs(aZ);

      // check if it's above the threshold
      if (aSum >= accelerationThreshold) {
        // reset the sample read count
        samplesRead = 0;
        break;
      }
    }
  }

  // check if the all the required samples have been read since
  // the last time the significant motion was detected
  while (samplesRead < numSamples) {
    // check if new acceleration AND gyroscope data is available
    if (IMU.accelerationAvailable() && IMU.gyroscopeAvailable()) {
      // read the acceleration and gyroscope data
      IMU.readAcceleration(aX, aY, aZ);
      IMU.readGyroscope(gX, gY, gZ);

      // normalize the IMU data between 0 to 1 and store in the model's
      // input tensor
      tflInputTensor->data.f[samplesRead * 6 + 0] = (aX + 4.0) / 5.0;
      tflInputTensor->data.f[samplesRead * 6 + 1] = (aY + 3.0) / 5.0;
      tflInputTensor->data.f[samplesRead * 6 + 2] = (aZ + 3.0) / 6.0;
      tflInputTensor->data.f[samplesRead * 6 + 3] = (gX + 370.0) / 740.0;
      tflInputTensor->data.f[samplesRead * 6 + 4] = (gY + 210.0) / 420.0;
      tflInputTensor->data.f[samplesRead * 6 + 5] = (gZ + 220.0) / 450.0;

      samplesRead++;
```

```
      if (samplesRead == numSamples) {
        // Run inferencing
        TfLiteStatus invokeStatus = tflInterpreter->Invoke();
        if (invokeStatus != kTfLiteOk) {
          Serial.println("Invoke failed!");
          while (1);
          return;
        }

        // Loop through the output tensor values from the model
        float confidences[Num_Activities];
        for (int i = 0; i < Num_Activities; i++) {
          Serial.print(Activities[i]);
          Serial.print(": ");
          Serial.println(tflOutputTensor->data.f[i], 6);
          confidences[i] = tflOutputTensor->data.f[i];
        }
        Serial.println();

        int maxConfidence = getIndexOfMaximumValue(confidences, Num_Activities);
        Serial.print("The results is ");
        Serial.println(Activities[maxConfidence]);
        // Activities[maxConfidence] is the class that has the highest confidence
        if (Activities[maxConfidence] == "sitting") {
            digitalWrite(4, LOW);
            digitalWrite(5, HIGH);
            digitalWrite(6, LOW);
            delay(600);
            digitalWrite(4, LOW);
            digitalWrite(5, LOW);
            digitalWrite(6, LOW);
        } else if (Activities[maxConfidence] == "standing") {
            digitalWrite(4, LOW);
            digitalWrite(5, HIGH);
            digitalWrite(6, LOW);
        } else if (Activities[maxConfidence] == "walking") {
            delay(800);
            digitalWrite(4, LOW);
            digitalWrite(5, LOW);
            digitalWrite(6, HIGH);

        } else if (Activities[maxConfidence] == "running") {
            delay(800);
            digitalWrite(4, HIGH);
            digitalWrite(5, LOW);
            digitalWrite(6, LOW);
        }
      }
    }
  }
}

void Volts()
{
  int sensorValue = analogRead(A0); //read the A0 pin value
  float voltage = sensorValue * (5.00 / 1023.00) * 2; //convert the value to a true voltage.
  if (voltage < 7.20) //set the voltage considered low battery here
  {
    digitalWrite(2, HIGH);
  }
}

int getIndexOfMaximumValue(float* array, int size){
 int maxIndex = 0;
 float max = array[maxIndex];
 for (int i = 0; i < size; i++){
   if (array[i] > max){
     max = array[i];
     maxIndex = i;
   }
 }
 return maxIndex;
}
```

Εικόνα 44: Κατηγοριοποίηση Δεδομένων μέσω IDE

Επίλογος

Εν κατακλείδι, η παρούσα διπλωματική εργασία είχε ως στόχο να διερευνήσει τις δυνατότητες της βαθιάς μάθησης για την αναγνώριση της ανθρώπινης δραστηριότητας χρησιμοποιώντας ένα Arduino Nano 33 BLE Sense. Η έρευνα που παρουσιάστηκε σε αυτή τη διπλωματική εργασία έδειξε ότι η βαθιά μάθηση είναι μια πολλά υποσχόμενη προσέγγιση για την αναγνώριση ανθρώπινης δραστηριότητας λόγω της ικανότητάς της να εξάγει χαρακτηριστικά υψηλού επιπέδου από ακατέργαστα δεδομένα αισθητήρων.

Η διατριβή ξεκίνησε με μια επισκόπηση των εννοιών της τεχνητής νοημοσύνης, της μηχανικής μάθησης και της βαθιάς μάθησης. Αυτό αποτέλεσε τη βάση για την κατανόηση των μοντέλων νευρωνικών δικτύων που χρησιμοποιήθηκαν στο σύστημα αναγνώρισης δραστηριότητας.

Η διπλωματική εργασία παρείχε επίσης λεπτομερή ανάλυση του Arduino Nano 33 BLE Sense, τονίζοντας τα βασικά χαρακτηριστικά και πλεονεκτήματά του. Αυτή η συσκευή επιλέχθηκε για το έργο αναγνώρισης δραστηριοτήτων κυρίως λόγω των ενσωματωμένων αισθητήρων οι οποίοι επέτρεψαν την παρακολούθηση των ανθρώπινων δραστηριοτήτων σε πραγματικό χρόνο.

Η κύρια συνεισφορά της παρούσας διατριβής είναι ο σχεδιασμός και η υλοποίηση ενός συστήματος αναγνώρισης δραστηριοτήτων με χρήση βαθιάς μάθησης. Το σύστημα εκπαιδεύτηκε και αξιολογήθηκε χρησιμοποιώντας ένα σύνολο δεδομένων, επιτυγχάνοντας υψηλή ακρίβεια στην ταξινόμηση τεσσάρων διαφορετικών δραστηριοτήτων. Τα αποτελέσματα καταδεικνύουν τις δυνατότητες της βαθιάς μάθησης για την αναγνώριση ανθρώπινων δραστηριοτήτων και την αποτελεσματικότητα του Arduino Nano 33 BLE Sense για την κατασκευή τέτοιων συστημάτων.

Το έργο καταδεικνύει τελικά τη δυνατότητα κατασκευής μιας συσκευής αναγνώρισης δραστηριότητας η οποία μπορεί να επεκταθεί περαιτέρω και να εφαρμοστεί σε διάφορους τομείς, όπως η υγειονομική περίθαλψη, ο αθλητισμός ή ο οικιακός αυτοματισμός.

Βιβλιογραφία

1. McCarthy, John (1999), What is AI?, archived from the original on 4 December 2022, retrieved 4 December 2022.
2. Deng, L.; Yu, D. (2014). "Deep Learning: Methods and Applications" (PDF). Foundations and Trends in Signal Processing. 7 (3–4): 1–199. doi:10.1561/2000000039. Archived (PDF) from the original on 14 March 2016. Retrieved 18 October 2014.
3. Heath, Nick (11 December 2020). "What is AI? Everything you need to know about Artificial Intelligence". ZDNet. Archived from the original on 2 March 2021. Retrieved 1 March 2021.
4. Anadiotis, George (1 October 2020). "The state of AI in 2020: Democratization, industrialization, and the way to artificial general intelligence". ZDNet. Archived from the original on 15 March 2021. Retrieved 1 March 2021.
5. Schmidhuber, J. (2015). "Deep Learning in Neural Networks: An Overview". Neural Networks. 61: 85–117. arXiv:1404.7828. doi:10.1016/j.neunet.2014.09.003. PMID 25462637. S2CID 11715509.
6. Russell, Stuart J.; Norvig, Peter (2003), Artificial Intelligence: A Modern Approach (2nd ed.), Upper Saddle River, New Jersey: Prentice Hall, ISBN 0-13-790395-2.
7. Nils J. Nilsson, Introduction to Machine Learning Archived 2019-08-16 at the Wayback Machine.
8. Ethem Alpaydin (2004). Introduction to Machine Learning, MIT Press, ISBN 978-0-262-01243-0.
9. Kevin P. Murphy (2021). Probabilistic Machine Learning: An Introduction Archived 2021-04-11 at the Wayback Machine, MIT Press.
10. Schulz, Hannes; Behnke, Sven (1 November 2012). "Deep Learning". KI - Künstliche Intelligenz. 26 (4): 357–363. doi:10.1007/s13218-012-0198-z. ISSN 1610-1987. S2CID 220523562.
11. Deng, L.; Yu, D. (2014). "Deep Learning: Methods and Applications" (PDF). Foundations and Trends in Signal Processing. 7 (3–4): 1–199. doi:10.1561/2000000039. Archived (PDF) from the original on 2016-03-14. Retrieved 2014-10-18.
12. Bengio, Yoshua (2009). "Learning Deep Architectures for AI" (PDF). Foundations and Trends in Machine Learning. 2 (1): 1–127. CiteSeerX 10.1.1.701.9550.
13. Haykin, Simon S. (1999). Neural Networks: A Comprehensive Foundation. Prentice Hall. ISBN 978-0-13-273350-2.
14. Banzi, Massimo; Shiloh, Michael (2022). Make: Getting Started With Arduino: The Open Source Electronics Prototyping Platform (4th ed.). Make Community. ISBN 978-1680456936.
15. "Board; Nano 33 IoT; Docs". Arduino. Archived from the original on May 9, 2023.
16. "Getting Started with the Arduino Nano 33 BLE Sense". Arduino.

17. Monk, Simon (2018). Programming Arduino Next Steps: Going Further with Sketches (2nd ed.). McGraw-Hill Education. ISBN 978-1260143249.
18. "3D human gesture capturing and recognition by the IMMU-based data glove". Retrieved 2022-09-02.
19. "GNSS/INS". Xsens 3D motion tracking. Archived from the original on 2019-01-22. Retrieved 2019-01-22.
20. Tanzeem Choudhury, Gaetano Borriello, et al. The Mobile Sensing Platform: An Embedded System for Activity Recognition. Appears in the IEEE Pervasive Magazine – Special Issue on Activity-Based Computing, April 2008.
21. Nishkam Ravi, Nikhil Dandekar, Preetham Mysore, Michael Littman. Activity Recognition from Accelerometer Data. Proceedings of the Seventeenth Conference on Innovative Applications of Artificial Intelligence (IAAI/AAAI 2005).
22. Derek Hao Hu, Sinno Jialin Pan, Vincent Wenchen Zheng, Nathan NanLiu, and Qiang Yang. Real world activity recognition with multiple goals. In Proceedings of the 10th international conference on Ubiquitous computing, Ubicomp, pages 30–39, New York, NY, USA, 2008. ACM.
23. TLM van Kasteren, Gwenn Englebienne, BJA Kröse. "Human activity recognition from wireless sensor network data: Benchmark and software." Activity Recognition in Pervasive Intelligent Environments, 165–186, Atlantis Press
24. https://www.sap.com/greece/insights/what-is-machine-learning.html
25. https://iguru.gr/einai-kai-pos-leitourgei-vathia-mathisi/
26. https://www.mathworks.com/discovery/machine-learning.html
27. https://www.javatpoint.com/machine-learning
28. https://www.ibm.com/topics/deep-learning
29. https://flatironschool.com/blog/deep-learning-vs-machine-learning/
30. https://builtin.com/machine-learning/deep-learning
31. https://www.seldon.io/transfer-learning
32. https://neptune.ai/blog/transfer-learning-guide-examples-for-images-and-text-in-keras
33. https://www.ibm.com/topics/overfitting
34. https://www.arduino.cc/en/Guide/Introduction
35. https://www.makeuseof.com/tag/4-reasons-everyone-learn-arduino-now/
36. https://circuitdigest.com/microcontroller-projects/arduino-nano-33-ble-sense-board-review-and-getting-started-guide
37. http://www.starlino.com/imu_guide.html A Guide To using IMU (Accelerometer and Gyroscope Devices) in Embedded Applications