



Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης
Πολυτεχνική Σχολή
Τμήμα Ηλεκτρολόγων Μηχανικών &
Μηχανικών Υπολογιστών
Τομέας Ηλεκτρονικής και Υπολογιστών

Διπλωματική Εργασία

Title not yet specified

Εκπόνηση:
Παναγιώτου Κωνσταντίνος
ΑΕΜ: 7316

Επίβλεψη:
Αν. Καθ. Πέτρου Λουκάς
Υπ. Δρ. Μουσουλιώτης
Παναγιώτης

Θεσσαλονίκη, Σεπτέμβριος 2016

The robots are going to help us find our crystal...

ΕΥΧΑΡΙΣΤΙΕΣ

Ευχαριστίες here!

Περίληψη

Title

Thesis Title here!

Abstract

Konstantinos Panayiotou
Electrical & Computer Engineering Department,
Aristotle University of Thessaloniki, Greece
September 2016

Περιεχόμενα

Ευχαριστίες	iii
Περίληψη	v
Abstract	vii
Ακρωνύμια	xiii
1 Εισαγωγή	1
1.1 Περιγραφή του Προβλήματος	2
1.2 Σκοπός - Συνεισφορά της Διπλωματικής Εργασίας	3
1.3 Διάρθρωση της Αναφοράς	3
2 Επισκόπηση της Ερευνητικής Περιοχής	6
3 Τεχνικές Βαθιάς Μηχανικής Μάθησης και Αναγνώριση Αντικειμένων στον χώρο	8
3.1 Εισαγωγή στην Επιστήμη της Σύγχρονης Μηχανικής Μάθησης	9
3.2 Νευρωνικά Δίκτυα με Βάθος	17
3.2.1 Συναρτήσεις Ενεργοποίησης	20
3.2.2 Ο αλγόριθμος Backpropagation	21
3.2.3 Αρχιτεκτονικές	21
3.3 Νευρωνικά Δίκτυα Συνέλιξης	21
3.4 Εφαρμογές Νευρωνικών Δικτύων Συνέλιξης στην Αναγνώριση Αντικειμένων στον Χώρο	21
4 Εργαλεία Hardware/Software που χρησιμοποιήθηκαν	22
4.1 NVIDIA Jetson TK1 development board	22
4.2 Full software stack for developing Deep Neural Networks	25
5 Implementations	26
5.1 YOLO implementation with keras DNN framework	26
5.2 Desktop PC configuration for optimal performance	26
5.3 Jetson TK1 dev board setup and applied optimizations	26
6 Πειράματα - Αποτελέσματα	28
7 Συμπεράσματα	30
8 Μελλοντικές επεκτάσεις	32

Βιβλιογραφία

33

Κατάλογος Σχημάτων

3.1	Διάγραμμα Venn των διαφόρων κατηγοριών μηχανικής μάθησης . . .	11
3.2	Παράδειγμα διαφορετικών αναπαραστάσεων των δεδομένων	15
3.3	Απλό μοντέλο Autocoder με ένα κρυφό επίπεδο	15
3.4	Παράδειγμα απεικόνισης των φίλτρων ενός μοντέλου CNN για ανα- γνώριση προσώπου	16
3.5	Παράδειγμα απεικόνισης των φίλτρων ενός μοντέλου CNN για ανα- γνώριση προσώπου	17
3.6	Βιολογικός Νευρώνας	18
3.7	Μαθηματικό μοντέλο του νευρώνα	18
3.8	Υλοποίηση πύλης AND με χρήση του μαθηματικού μοντέλου του perceptron	20
3.9	Απλό μοντέλο NN με ένα κρυφό επίπεδο	20
4.1	Tegra K1 SOC	23
4.2	Jetson TK1 development board	24
4.3	ARM Cortex-A15 processor	25

Κατάλογος πινάκων

Κατάλογος Αλγορίθμων

3.1	Υπολογισμός εξόδων ενός NN	21
-----	--------------------------------------	----

Ακρωνύμια Εγγράφου

Παρακάτω παρατίθενται ορισμένα από τα πιο συχνά χρησιμοποιούμενα ακρωνύμια της παρούσας διπλωματικής εργασίας:

AI	→ Artificial Intelligence
ML	→ Machine Learning
NN	→ Neural Network
DL	→ Deep Learning
DNN	→ Deep Neural Network
MLP	→ Multilayer Perceptron
CNN	→ Convolutional Neural Network
ConvNet	→ Convolutional Networks
RNN	→ Recurrent Neural Network
YOLO	→ You Only Look Once
SOC	→ System On Chip
OS	→ Operating System

1

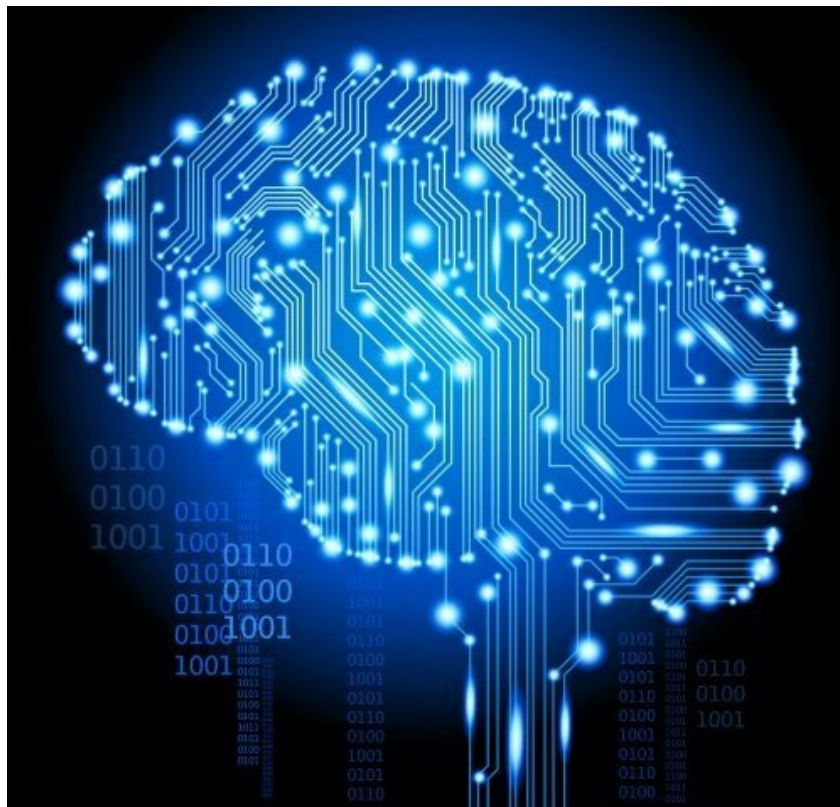
Εισαγωγή

Ο όρος *Τεχνητή Νοημοσύνη* αναφέρεται στην ικανότητα των υπολογιστικών συστημάτων να μιμούνται στοιχεία της ανθρώπινης συμπεριφοράς. Η επιθυμία των ανθρώπων να κατασκευάσουν "έξυπνες" μηχανές, καταγράφεται από τα έτη της αρχαίας Ελλάδας. Μυθικές μορφές όπως οι Πυγμαλίωνας, Δαίδαλος και Ήφαιστος μπορούν να θεωρηθούν ως θρυλικοί εφευρέτες και δημιουργοί νοούντων μηχανών όπως η Γαλάτεια, ο Τάλος και η Πανδώρα.

Ένας πιο ολοκληρωμένος ορισμός της Τεχνητή Νοημοσύνη είναι:

Ο κλάδος/τομέας της επιστήμης της πληροφορικής, που ασχολείται με την σχεδίαση και κατασκευή ευφυών συστημάτων, δηλαδή συστημάτων που κατέχουν χαρακτηριστικά που σχετίζονται με την ανθρώπινη νοημοσύνη και συμπεριφορά.

Με την εμφάνιση των πρώτων ηλεκτρονικών και (επανα)προγραμματιζόμενων υπολογιστικών συστημάτων, οι άνθρωποι ξεκίνησαν να σκέφτονται τρόπους για να κατασκευάσουν "έξυπνες" μηχανές. Η ραγδαία εξέλιξη στον κλάδο της επιστήμης της πληροφορικής, τις τελευταίες δεκαετίες, έφερε και την εξέλιξη στην επιστήμη της Τεχνητής Νοημοσύνης. Το 1997, η IBM κατασκεύασε ένα υπολογιστικό σύστημα το οποίο μπορούσε να πέξει σκάκι (Deep Blue) [1], το οποίο κέρδισε τον παγκόσμιο τότε πρωταθλητή στο σκάκι, Garry Kasparov. Το σκάκι έχει εξηντατέσσερις θέσεις και τριάνταδύο πιόνια που μπορούν να κινούνται με συγκεκριμένο τρόπο. Η μηχανή Deep Blue, μπορούσε να εκτιμήσει και να αξιολογήσει διακόσια εκατομύρια πιθανές καταστάσεις της σκακιέρας. Ωστόσο, πρέπει να σημειώσουμε ότι η επίλυση του πρόβληματος του σκακιού, είναι ένα απλό πρόβλημα το οποίο μπορεί να περιγραφεί πλήρως μέσα από μια λίστα με κανόνες. Η δυνατότητα ενός "ευφυούς" ρομποτικού συστήματος, να αντιλαμβάνεται το περιβάλλον του, είναι απαραίτητη ικανότητα που πρέπει να διαθέτει. Ένας ρομποτικός πράκτορας, σε πληθώρα εφαρμογών, πρέπει να διαθέτει τόσο την ικανότητα να αναγνωρίζει, μέσω εικόνων λήψης από κάμερας, διάφορα αντικείμενα, όσο και την ικανότητα να εντοπίζει επακριβώς



την θέση του εκάστοτε αντικειμένου στον χώρο. Δεν θα είχε νόημα για ένα ρομπότ, να έχει άποψη για την παρουσία αντικειμένων γύρω του, αν δεν έχει και την ικανότητα να γνωρίζει και την θέση των αντικειμένων αυτών.

Σήμερα, το γενικότερο πρόβλημα της αναγνώρισης και εντοπισμού αντικειμένων σε εικόνες λύνεται με την χρησιμοποίηση Νευρωνικών Δικτύων Συνέλιξης (Convolutional Neural Networks - CNN). Το γεγονός ότι τα CNN έχουν την δυνατότητα να κατηγοριοποιήσουν αντικείμενα, σε προβλήματα όπου οι κλάσεις αντικειμένων κυμάνονται από δεκάδες έως και χιλιάδες, θέτει το ερώτημα της εφαρμογής τους σε εφαρμογές πραγματικού χρόνου εφαρμογές με ιδιαίτερο ενδιαφέρον στην επιστήμη της ρομποτικής.

TODO: More...!!!!???

1.1 ΠΕΡΙΓΡΑΦΗ ΤΟΥ ΠΡΟΒΛΗΜΑΤΟΣ

Παρόλο που τα CNN είναι ικανά να δώσουν λύσεις με μεγάλη ακρίβεια, απαιτούν μεγάλο όγκο επεξεργαστικής ισχύς, τόσο για την εκπαίδευσή τους, όσο και για την εκτέλεση ενός πειράματος, όταν το πρόβλημα για το οποίο έχουν σχεδιαστεί να δώσουν λύση είναι περίπλοκο. Η απαίτηση αυτή προέρχεται από το βάθος των μοντέλων CNN για αναγνώριση και εντοπισμό αντικειμένων σε εικόνες. Για παράδειγμα, ένα από τα πρώτα μοντέλα CNN το οποίο σχεδιάστηκε για την αναγνώριση αντικειμένων σε εικόνες, αποτελείται από 16 επίπεδα (AlexNet) και έχει εξήντα εκατομμύρια (60,000,000) παραμέτρους και εξακόσες-πενήντα χιλιάδες (650,000)

νευρώνες από τους οποίους οι περισσότεροι εκτελούν πράξεις συνέλιξης. Ο Alex Krizhevsky απέδειξε το 2012 ότι η χρησιμοποίηση των σύγχρονων GPU για την εκτέλεση πράξεων συνέλιξης, φέρει σαν αποτέλεσμα την εκαπαίδευση μοντέλων CNN σε χρόνους έως και δύο τάξεις μεγέθους πιο κάτω σε σχέση με έναν ισχυρό επεξεργαστή [2]. Ο χρόνος εκτέλεσης ενός πειράματος του Δικτύου AlexNet έχει μετρηθεί στα 7.39 δευτερόλεπτα σε οκτα-πύρηνο επεξεργαστή Haskwell @2.9Ghz και στα 0.71 δευτερόλεπτα σε μονάδα GPU, Nvidia K520 [3].

Είναι ιδιαίτερα σημαντικό, ένα ρομπότ να μπορεί να αντιλαμβάνεται το περιβάλλον του; να μπορεί να αναγνωρίζει ανθρώπους, ζώα, αντικείμενα γενικότερα. Ωστόσο, θέλουμε τα ρομπότ να είναι και όσο πιο "ελκιστικά" γίνεται στον άνθρωπο ή/και μικρότερα. Αυτό, φέρει σαν αποτέλεσμα να μην μπορούμε να τοποθετήσουμε ογκώδη, άρα με μεγάλη υπολογιστική ισχύ, υπολογιστικά συστήματα κατευθείαν πάνω στα ρομπότ.

Παρόλο που σήμερα έχουν σχεδιαστεί μοντέλα CNN, τα οποία έχουν την δυνατότητα να αναγνωρίζουν και να εντοπίσουν αντικείμενα από χιλιάδες, αν όχι και περισσότερες, κλάσεις, ο χρόνος που απαιτείται για να κατηγοριοποιήσει αντικείμενα σε μία εικόνα είναι αρκετά μεγάλος, της τάξης των μερικών δευτερολέπτων σε σύγχρονους υπολογιστές. Αυτό κάνει την χρήση CNN σε εφαρμογές πραγματικού χρόνου, όπως για παράδειγμα στην ρομποτική, ακατάλληλη. Ωστόσο, η επιστημονική κοινότητα σήμερα προσπαθεί να δώσει λύσεις στο συγκεκριμένο πρόβλημα εστιάζοντας το ενδιαφέρον στην εξέλιξη των ενσωματωμένων συστημάτων και την σχεδίαση γρήγορων λογισμικών για υλοποιήσεις μοντέλων DNN τα οποία εκμεταλλεύονται κυρίως την υπολογιστική ισχύ των μονάδων GPU, αλλά και άλλων πολυπύρηνων επεξεργαστικών μονάδων.

1.2 ΣΚΟΠΟΣ - ΣΥΝΕΙΣΦΟΡΑ ΤΗΣ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Η παρούσα διπλωματική εργασία έχει στόχο την υλοποίηση και σύγκριση διαφόρων μοντέλων CNN για αναγνώριση και εντοπισμό αντικειμένων σε εικόνες (object detection), σε ένα ρομποτικό σύστημα.

Περεταίρω εξετάζεται η χρήση υβριδικών ενσωματωμένων συστημάτων, τα οποία φέρουν μονάδες CPU αλλά και GPU (GPU+CPU SOC), σε εφαρμογές αναγνώρισης και εντοπισμού αντικειμένων σε πραγματικό χρόνο (Real-Time Object Detection).

...

1.3 ΔΙΑΦΩΡΩΣΗ ΤΗΣ ΑΝΑΦΟΡΑΣ

Η διάρθρωση της παρούσας διπλωματικής εργασίας είναι η εξής:

- **Κεφάλαιο 2:** Παρατίθεται η ανασκόπηση της ερευνητικής περιοχής αναφορικά με τα αντικείμενα στα οποία επιδιώκει να παρουσιάσει λύσεις η διπλωματική εργασία.

- **Κεφάλαιο 3:** Περιγράφονται τα βασικά θεωρητικά στοιχεία στα οποία βασίστηκαν οι υλοποιήσεις.
- **Κεφάλαιο 4:** Παρουσιάζονται οι διάφορες τεχνικές και τα εργαλεία που χρησιμοποιήθηκαν.
- **Κεφάλαιο 5:** Πλήρες περιγραφή των υλοποιήσεων.
- **Κεφάλαιο 6:** Παρουσιάζεται αναλυτικά η μεθοδολογία των πειραμάτων.
- **Κεφάλαιο 7:** Παρουσιάζονται τα συμπεράσματα στα οποία καταλήξαμε.
- **Κεφάλαιο 8:** Στο τελευταίο αυτό κεφάλαιο αναφέρονται τα προβλήματα που προέκυψαν και προτείνονται θέματα για μελλοντική μελέτη, αλλαγές και επεκτάσεις.

2

Επισκόπηση της Ερευνητικής Περιοχής

Τόσο η αναγνώριση αντικειμένων (object recognition) όσο και η εντοπισμός της θέσης των αντικειμένων αυτών (detection/localization) σε εικόνες λήψης είναι μία ερευνητική περιοχή με τεράστιο ενδιαφέρον και η οποία απασχολεί πληθώρα ερευνητών. Η επιστήμη της Μηχανικής Όρασης (Computer Vision - ML), στοχεύει στο να δώσει λύσεις στα συγκεκριμένα προβλήματα, εισάγοντας μαθηματικά μοντέλα, τόσο αναλυτικά, όσο και πιθανοτικά.

Ο κλάδος της Βαθιάς Μηχανικής Μάθησης (Deep Learning - DL) [4], ανάγει το πρόβλημα της εύρεσης χαρακτηριστικών σημείων, για την αναγνώριση αντικειμένων, στην εκμάθηση αναπαραστάσεων [5], με την χρήση Νευρωνικών Δικτύων Συνέλιξης (Convolutional Neural Networks - CNN). Οι πρώτες εφαρμογές Νευρωνικών Δικτύων Συνέλιξης, για την αναγνώριση αντικειμένων σε εικόνες, αναπτύχθηκαν το 1990 από τον Yann LeCun. Η πιο γνωστή και επιτυχής είναι το δίκτυο LeNet [6], το οποίο χρησιμοποιήθηκε για την αναγνώριση ψηφίων σε εικόνες. Ωστόσο, η εισαγωγή των CNN στον κλάδο της Μηχανικής Όρασης έγινε το 2012 με την ανάπτυξη του δικτύου AlexNet [2], από τους Alex Krizhevsky, Ilya Sutskever και Geoffrey E. Hinton. Το δίκτυο AlexNet χρησιμοποιήθηκε στον διαγωνισμό ImageNet ILSVRC challenge, το 2012, κερδίζοντας με διαφορά 10,9%, στο σφάλμα αναγνώρισης αντικειμένων σε σύνολο 1000 κλάσεων. Με την εμφάνιση του δικτύου AlexNet, η ερευνητική κοινότητα ξεκίνησε να πιστεύει στην αποτελεσματικότητα των Νευρωνικών Δικτύων Συνέλιξης σε εφαρμογές αναγνώρισης αντικειμένων σε εικόνες. Συνέχεια στο έργο του Alex Krizhevsky έδωσαν οι ερευνητές αναπτύσσοντας, το 2013, το ZF-Net [7], το οποίο είναι βασισμένο στην αρχιτεκτονική του δικτύου AlexNet. Μέχρι σήμερα, έχουν σχεδιαστεί και αναπτυχθεί διάφορα μοντέλα Νευρωνικών Δικτύων Συνέλιξης για αναγνώριση αντικειμένων, με πιο πρόσφατο το ResNet [8], το οποίο αναπτύχθηκε από τον Kaiming He. Το ResNet (Residual Network) έχει την ιδιαιτερότητα απουσίας πλήρες συνδεδεμένων επιπέδων και είναι από τα πιο δημοφιλείς μοντέλα που εφαρμόζονται σε πρακτικά προβλήματα αναγνώρισης αντικειμένων σε εικόνες [9].

Τα προαναφερθέντα μοντέλα Νευρωνικών Δικτύων Συνέλιξης δίνουν λύσεις μόνο στο πρόβλημα της αναγνώρισης και όχι του εντοπισμού της θέσης των αντικειμένων αυτών. Το 2013, ερευνητές εργαζόμενοι στην Google Inc., σχεδίασαν και υλοποίησαν ένα μοντέλο Νευρωνικού Δικτύου το οποίο δίνει λύσεις στο πρόβλημα της ταυτόχρονης αναγνώρισης και εντοπισμού αντικειμένων πάνω σε πλαίσια εικόνων [10]. Το μοντέλο αυτό, που φέρει το όνομα DetectorNet, είναι ομαδική εργασία των Christian Szegedy, Alexander Toshev και Dumitru Erhan. Το μοντέλο αυτό είναι πιθανοτηκό αφού "προβλέπει" τις οριοθετημένες θέσεις για διάφορες κλάσεις αντικειμένων στον πλαίσιο μίας εικόνας. Ωστόσο, ένα βασικό μειονέκτημα του DetectorNet που το κάνει ακατάλληλο για εφαρμογή σε προβλήματα πραγματικού χρόνου, όπως για παράδειγμα σε ένα ρομποτικό σύστημα, είναι οι τεράστιες απαιτήσεις του σε πόρους και χρόνο.

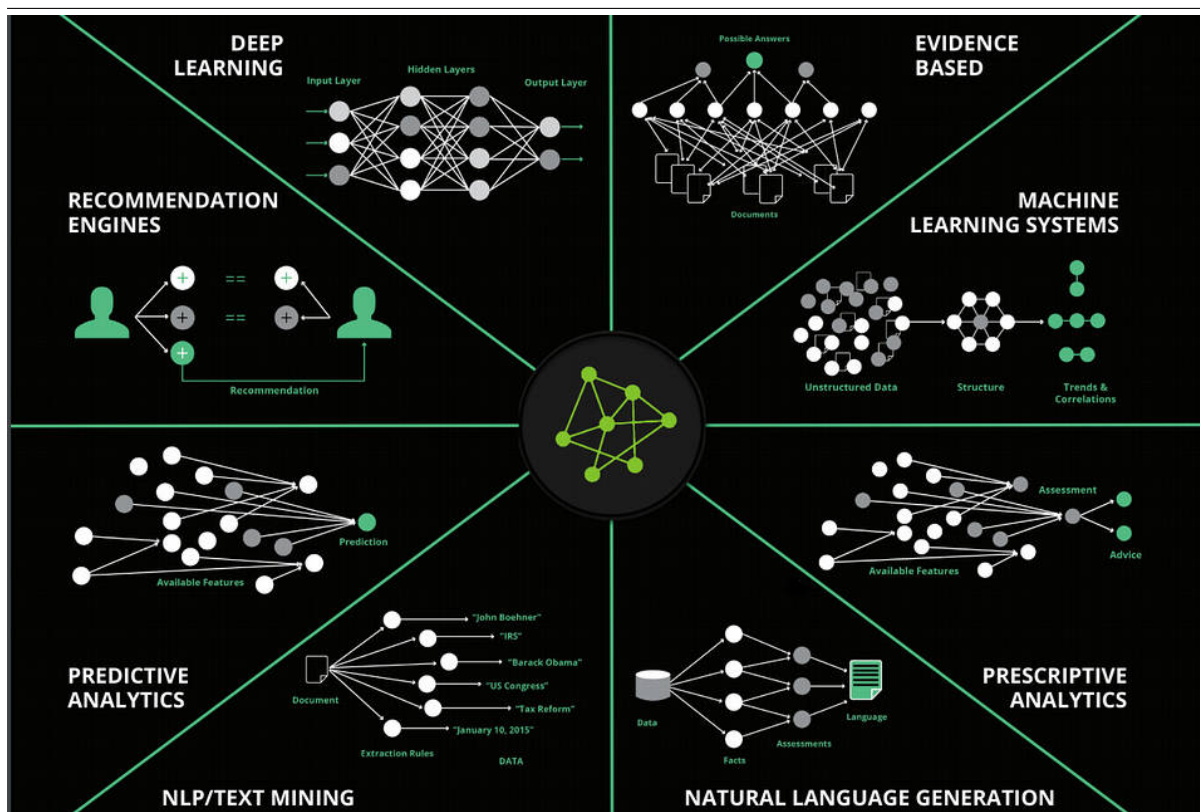
TODO: A few words about the applications of DNN models in robotics!!!!

3

Τεχνικές Βαθιάς Μηχανικής Μάθησης και Αναγνώριση Αντικειμένων στον χώρο

Τα τελευταία χρόνια, ο κλάδος της Τεχνητής Νοημοσύνης είναι ένας από τους πιο ραγδαία αναπτυσσόμενους κλάδους της επιστήμης της πληροφορικής με τεράστιο ερευνητικό και πρακτικό ενδιαφέρον; διαιρείται σε δύο υπο___; την συμβολική τεχνητή νοημοσύνη και την υποσυμβολική τεχνητή νοημοσύνη. Η πρώτη προσπαθεί να επιλύσει τα προβλήματα χρησιμοποιώντας αλγοριθμικές διαδικασίες, δηλαδή σύμβολα και λογικούς κανόνες, ενώ η δεύτερη προσπαθεί να αναπαράγει την ανθρώπινη "ευφυΐα" μέσα από την χρήση αριθμητικών μοντέλων που με την σύνθεσή τους προσομοιώνουν την λειτουργία του ανθρώπινου εγκεφάλου (υπολογιστική νοημοσύνη). Η ικανότητα ενός νοούμενου (AI) συστήματος, να αποκτά από μόνο του γνώση, εξάγοντας πρότυπα ή/και χαρακτηριστικά σημεία από τα δεδομένα, είναι γνωστή ως *Μηχανική Μάθηση* (ML).

3.1. ΕΙΣΑΓΩΓΗ ΣΤΗΝ ΕΠΙΣΤΗΜΗ ΤΗΣ ΣΥΓΧΡΟΝΗΣ ΜΗΧΑΝΙΚΗΣ ΜΑΘΗΣΗΣ



Στο κεφάλαιο αυτό, παρουσιάζονται και αναλύονται τεχνικές και αλγόριθμοι Μηχανικής Μάθησης, με επίκεντρο τα νευρωνικά δίκτυα με βάθος (Deep Neural Networks - DNN). Στόχος είναι ο αναγνώστης να αντιληφθεί και να κατανοήσει τις βασικές αρχές και λειτουργίες των νευρωνικών αυτών δικτύων, αφού είναι οι βάσεις για την περαιτέρω μελέτη των νευρωνικών δικτύων συνέλιξης (CNN) και των εφαρμογών αυτών στο πρόβλημα της αναγνώρισης και εντοπισμού αντικειμένων σε εικόνες.

3.1 ΕΙΣΑΓΩΓΗ ΣΤΗΝ ΕΠΙΣΤΗΜΗ ΤΗΣ ΣΥΓΧΡΟΝΗΣ ΜΗΧΑΝΙΚΗΣ ΜΑΘΗΣΗΣ

Η εισαγωγή του κλάδου της μηχανικής μάθησης στην επιστήμη των υπολογιστών, επέτρεψε στους υπολογιστές να μπορούν να αντιμετωπίσουν προβλήματα που εμπλέκουν την αντίληψη για τον πραγματικό κόσμο και να πέρνουν υποκειμενικές αποφάσεις.

Η χρησιμοποίηση αλγορίθμων ML, επιτρέπουν σε συστήματα AI να προσαρμόζονται εύκολα σε καινούργια έργα, με ελάχιστη επέμβαση από τον άνθρωπο. Για παράδειγμα, ένα Νευρωνικό Δίκτυο που έχει εκπαιδευτεί να αναγνωρίζει γάτους σε εικόνες, δεν απαιτεί να σχεδιαστεί και να εκπαιδευτεί από το μηδέν για να έχει την ικανότητα να αναγνωρίζει και σκύλους.

Πολλά προβλήματα, ενώ μέχρι και πριν μία μερικά χρόνια λύνονταν με "χειρόγραφη", προγραμματισμένη από τον άνθρωπο γνώση, σήμερα χρησιμοποιούνται

ΚΕΦΑΛΑΙΟ 3. ΤΕΧΝΙΚΕΣ ΒΑΘΙΑΣ ΜΗΧΑΝΙΚΗΣ ΜΑΘΗΣΗΣ ΚΑΙ ΑΝΑΓΝΩΡΙΣΗ ΑΝΤΙΚΕΙΜΕΝΩΝ ΣΤΟΝ ΧΩΡΟ

αλγόριθμοι ML για την επίλυσή τους. Πιο κάτω παρατίθενται μερικά παραδείγματα:

- Αναγνώριση ομιλίας - Speech Recognition
- Μηχανική όραση - Computer Vision
 - Αναγνώριση αντικειμένων σε εικόνες - Object Recognition
 - Αναγνώριση και εντοπισμός της θέσης αντικειμένων σε εικόνες - Object Detection
- Αναγνώριση ηλεκτρονικών επιθέσεων στο διαδίκτυο - Cyberattack detection
- Επεξεργασία φυσικής γλώσσας - Natural Language Processing
 - Κατανόηση της φυσικής γλώσσας του ανθρώπου - Natural Language Understanding
 - Μοντελοποίηση και χρησιμοποίηση της φυσικής γλώσσας του ανθρώπου από μηχανές - Natural Language Generation
- Μηχανές αναζήτησης
- Αναπαράσταση γνώσης - Knowledge Representation
- Ρομποτική

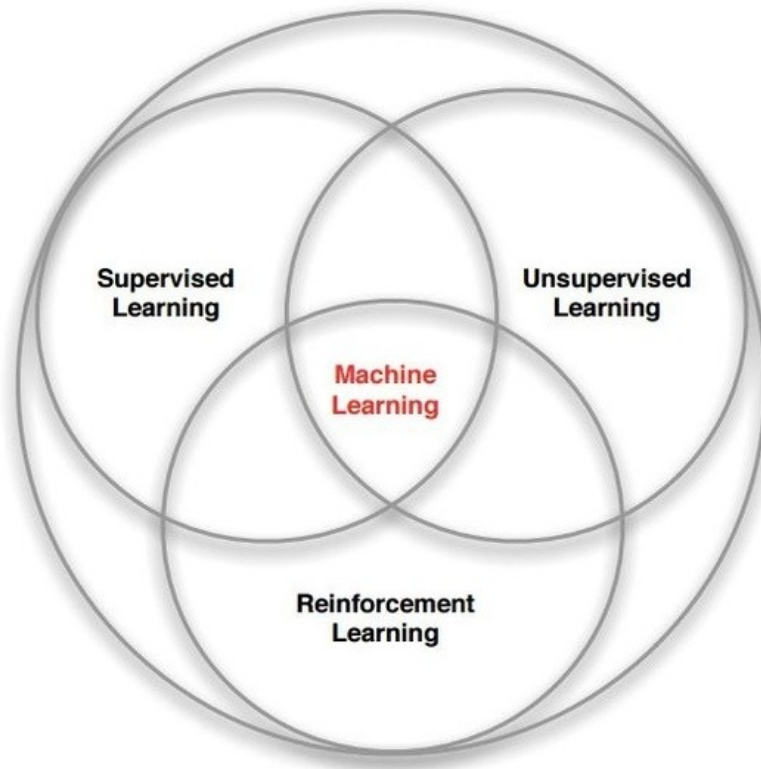
Τα προβλήματα Μηχανικής Μάθησης χωρίζονται σε τρεις μεγάλες κατηγορίες:

- Υπό επίβλεψη Μάθηση - Supervised Learning: Στο υπολογιστικό σύστημα δίνονται παραδείγματα εισόδου και επιθυμητής εξόδου, δηλαδή στα δεδομένα έχουν προηγουμένως ανατεθεί ετικέτες(labels), και στόχος είναι να μάθει ένα γενικό κανόνα αντιστοίχισης της εισόδου στην επιθυμητή έξοδο. Η αναγνώριση αντικειμένων σε εικόνες είναι ένα πρόβλημα που ανήκει σε αυτή την κατηγορία.
- Χωρίς επίβλεψη Μάθηση - Unsupervised Learning: Τα δεδομένα δεν έχουν ετικέτες (labels), αφήνοντας έτσι τον αλγόριθμο ML να βρεί από μόνος του δομές στα δεδομένα εισόδου.
- Εκμάθηση δια ανταμοιβής - Reinforcement Learning: Ο πράκτορας αλληλεπιδρά με ένα δυναμικό περιβάλλον στο οποίο πρέπει να εκτελέσει ένα συγκεκριμένο στόχο, χωρίς την ύπαρξη ενός "δασκάλου" που να ορίζει ρητά αν έχει φθάσει κοντά στον στόχο. Ένα παράδειγμα εφαρμογής είναι η αυτόματη πλοήγηση ενός οχήματος.

Επιπλέον, κάποια προβλήματα είναι υβριδικά, δηλαδή συνδυασμός των πιο πάνω. Στο [σχήμα 3.1](#) απεικονίζεται το διάγραμμα Venn των διαφόρων κατηγοριών προβλημάτων ML.

Περεταίρω, οι Supervised Learning αλγόριθμοι χωρίζονται σε 2 κατηγορίες, ανάλογα με την επιθυμητή μορφή της εξόδου του αλγόριθμου ML:

- Ταξινόμησης - Classification: Όταν η έξοδος παίρνει διακριτές τιμές (discrete).



Σχήμα 3.1: Διάγραμμα Venn των διαφόρων κατηγοριών μηχανικής μάθησης

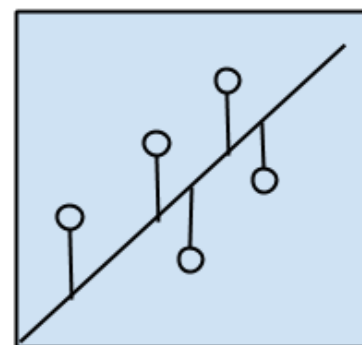
- Regression: Όταν η έξοδος παίρνει συνεχείς τιμές.

Γενικότερα, οι αλγόριθμοι ML ομαδοποιούνται και ανάλογα με την ομοιότητα τους σε σχέση με την λειτουργία που εκτελούν. Πιο κάτω αναφέρονται οι πιο δημοφιλείς αλγόριθμοι ML, ομαδοποιημένοι με βάση την λειτουργία τους

Regression

Ασχολείται με τη μοντελοποίηση της σχέσης μεταξύ των μεταβλητών που επαναληπτικά ανανεώνονται χρησιμοποιώντας ένα μέτρο σφάλματος για τις προβλέψεις που γίνονται από το μοντέλο

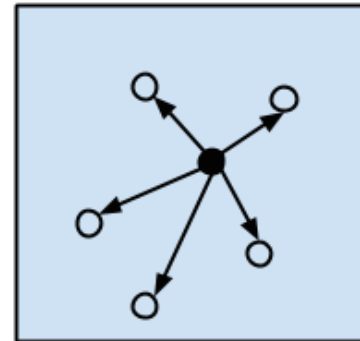
- Ordinary Least Squares Regression (OLSR)
- Linear Regression
- Logistic Regression
- Stepwise Regression
- Multivariate Adaptive Regression Splines (MARS)
- Locally Estimated Scatterplot Smoothing (LOESS)



Instance-based

Αυτές οι μεθόδους δημιουργούν μία βάση δεδομένων με παραδείγματα δεδομένων και συγκρίνουν τα νέα δεδομένα με αυτά που έχουν καταχωρηθεί στην βάση δεδομένων χρησιμοποιώντας ένα μέτρο ομοιότητας, για την εύρεση της καλύτερης αντιστοιχίας, πιθανοτικά.

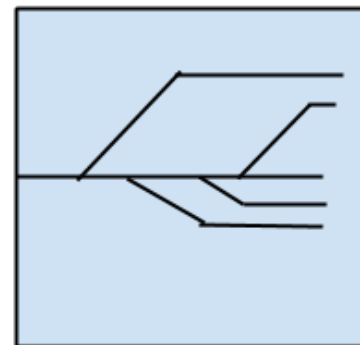
- k-Nearest Neighbour (kNN)
- Learning Vector Quantization (LVQ)
- Self-Organizing Map (SOM)
- Locally Weighted Learning (LWL)



Regularization

Χρησιμοποιούνται σαν επεκτάσεις άλλων μεθόδων και "τιμωρούν" μοντέλα, βασισμένα στην πολυπλοκότητα τους, ευνοώντας έτσι απλούστερα μοντέλα τα οποία είναι παράλληλα καλύτερα στην γενίκευση της επίλυσης του εκάστοτε προβλήματος.

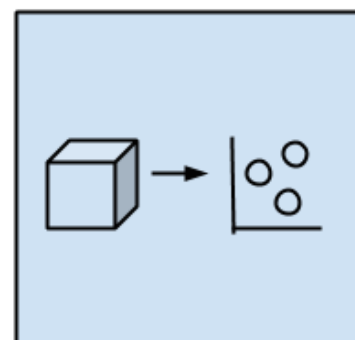
- Ridge Regression
- Least Absolute Shrinkage and Selection Operator (LASSO)
- Least-Angle Regression (LARS)
- Elastic Net



Dimensionality Reduction

Χρησιμοποιούνται για την αφαίρεση σχεδόν ασήμαντης πληροφορίας από τα δεδομένα. Πολλές από τις μεθόδους αυτές χρησιμοποιούνται σαν επεκτάσεις σε μοντέλα επίλυσης προβλημάτων regression και classification

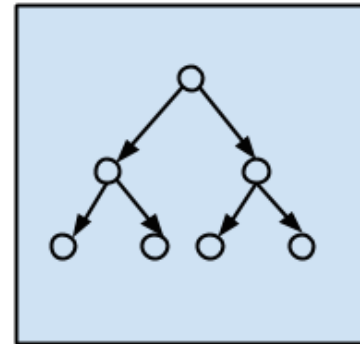
- Principal Component Analysis (PCA)
- Discriminant Analysis: Linear (LDA), Mixture (MDA), Quadratic (QDA), Flexible (FDA)
- Principal Component Regression (PCR)
- Multidimensional Scaling (MDS)



Decision Trees

Χρησιμοποιούνται για την κατασκευή μοντέλων λήψης αποφάσεων, τα οποία χρησιμοποιούν τις πραγματικές τιμές των χαρακτηριστικών των δεδομένων.

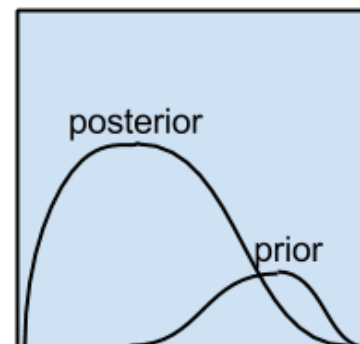
- Classification and Regression Tree (CART)
- Conditional Decision Trees
- M5



Bayesian

Εφαρμόζουν το θεώρημα του Bayes για την επίλυση τόσο προβλημάτων regression, αλλά και classification

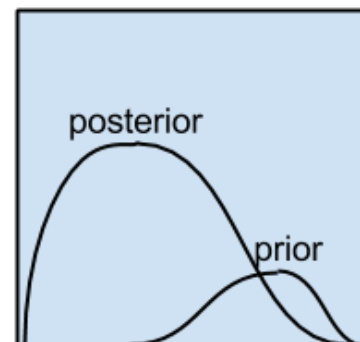
- Naive Bayes
- Gaussian Naive Bayes
- Bayesian Network (BN)
- Bayesian Belief Network (BBN)



Clustering

Περιγράφουν τις κλάσεις του προβλήματος

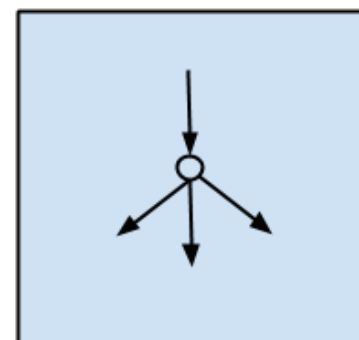
- k-Means
- k-Medians
- Expectation Maximisation (EM)
- Hierarchical Clustering



Artificial Neural Networks (ANN)

Μοντέλα εμπνευσμένα από τη δομή ή/και την λειτουργία των βιολογικών νευρωνικών δικτύων. Χρησιμοποιούνται στην επίλυση προβλημάτων classification ή/και regression

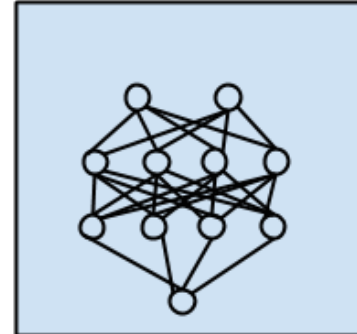
- Perceptron
- Back-Propagation
- Radial Basis Function Network (RBFN)



Deep Learning (DL)

Οι αλγόριθμοι DL είναι η σύγχρονη επέκταση των ANN, τα οποία εκμεταλλεύονται την αφθονία υπολογιστικής ισχύς των σύγχρονων υπολογιστικών συστημάτων.

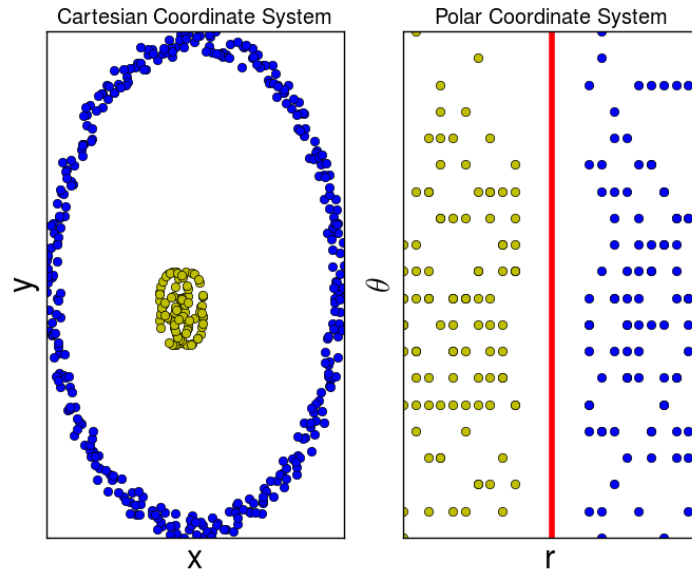
- Autocoder
- Multilayer Perceptron (MLP)
- Deep Boltzmann Machine (DBM)
- Deep Belief Networks (DBN)
- Convolutional Neural Network (CNN)
- Stacked Auto-Encoders
- Recurrent Neural Networks (RNN)



Η μορφή της αναπαράστασης των δεδομένων αποτελεί σημαντικό παράγοντα στην απόδοση των αλγορίθμων ML. Μία αναπαράσταση αποτελείται από χαρακτηριστικά (features). Για παράδειγμα, ένα χρήσιμο χαρακτηριστικό, στην ταυτοποίηση ομιλιτή, από δεδομένα ήχου, είναι η εκτίμηση του μεγέθους της φωνητικής έκτασης του ομιλιτή. Έτσι, πολλά προβλήματα τεχνητής νοημοσύνης, μπορούν να λυθούν με κατάλληλη σχεδίαση και επιλογή των χαρακτηριστικών, για το συγκεκριμένο πρόβλημα. Το σύνολο των χαρακτηριστικών αυτών αποτελεί την αναπαράσταση των δεδομένων, σε ένα πιο υψηλό και αφαιρετικό επίπεδο αντίληψης για τους υπολογιστές, η οποία στην συνέχεια δίνεται σαν είσοδος σε έναν απλό ML αλγόριθμο, ο οποίος έχει μάθει να αντιστοιχεί την αναπαράσταση των δεδομένων στην επιθυμητή έξοδο.

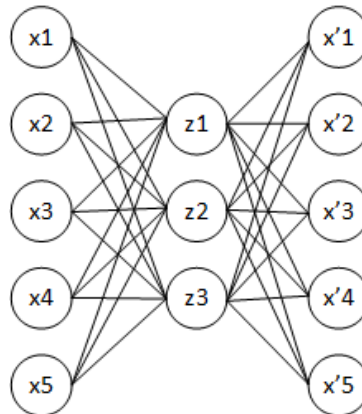
Ένα απλό και κατανοητό παράδειγμα το οποίο δείχνει την εξάρτηση της επίδοσης των αλγορίθμων ML από την μορφή της αναπαράστασης που του δίνεται, φαίνεται στο [σχήμα 3.2](#). Έστω ότι θέλουμε να διαχωρίσουμε τα δεδομένα μας σε δύο κλάσεις, χαράζοντας μία ευθεία μεταξύ τους. Αν αναπαραστήσουμε τα δεδομένα στο Καρτεσιανό σύστημα συντεταγμένων (αριστερό διάγραμμα), τότε η επίλυση του προβλήματος είναι αδύνατη αφού δεν υπάρχει καμία ευθεία που να διαχωρίζει τις δύο κλάσεις. Ωστόσο, αν αναπαραστήσουμε τα δεδομένα στο Πολικό σύστημα συντεταγμένων (δεξί διάγραμμα), τότε το πρόβλημα λύνεται εύκολα, χαράζοντας μία κάθετη ευθεία, με $r = a, a \in [r1, r2]$.

Σε πληθώρα προβλημάτων τεχνητής νοημοσύνης, η επιλογή κατάλληλων χαρακτηριστικών είναι δύσκολο και χρονοβόρο έργο. Έστω για παράδειγμα ότι θέλουμε να αναγνωρίσουμε πρόσωπα σε εικόνες. Ένα χαρακτηριστικό θα μπορούσε να είναι τα μάτια. Δυστυχώς όμως, η αναγνώριση ματιών είναι και αυτό ένα δύσκολο πρόβλημα, αφού δεν μπορεί να περιγραφεί πάντα επακριβώς έχοντας σαν δεδομένα τις τιμές των pixel της εικόνας. Η γεωμετρική, για παράδειγμα, μορφή των ματιών σε μία εικόνα λήψης εξαρτάται από την γωνία λήψης της εικόνας, τον φωτισμό, τις ανακλάσεις του φωτισμού, την απόσταση από την οποία γίνεται η λήψη, την ανάλυση της κάμερας, κτλ.



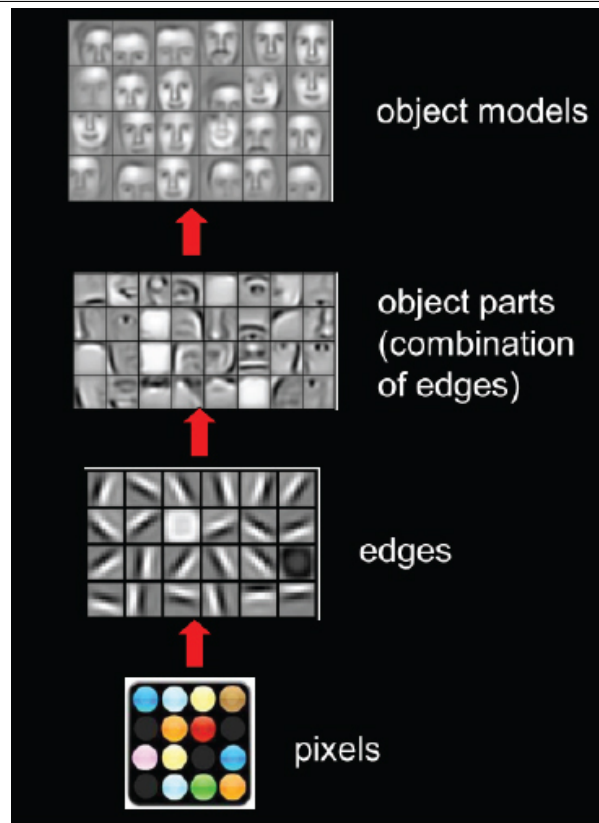
Σχήμα 3.2: Παράδειγμα διαφορετικών αναπαραστάσεων των δεδομένων

Το πρόβλημα αυτό, της επιλογής κατάλληλης αναπαράστασης, μπορεί να λυθεί χρησιμοποιώντας τεχνικές μηχανικής μάθησης για την εκμάθηση της ίδιας της αναπαράστασης. Αυτή η προσεγγίση είναι γνωστή ως *Εκμάθηση Αναπαραστάσεων* (*Representation Learning*). Οι αλγόριθμοι εκμάθησης αναπαραστάσεων είναι ικανοί να "μάθουν" ένα καλό σετ χαρακτηριστικών (features). Ένα απλό παράδειγμα αλγό-



Σχήμα 3.3: Απλό μοντέλο Autoencoder με ένα κρυφό επίπεδο.

ριθμοι εκμάθησης αναπαραστάσεων είναι αυτό του Autoencoder [11] που φαίνεται στο (σχήμα 3.3). Ο Autoencoder, στην πιο απλή του μορφή, είναι ο συνδυασμός ενός κωδικοποιητή (encoder) ο οποίος μετατρέπει τα δεδομένα εισόδου σε μία διαφορετική αναπαράσταση, και ενός αποκωδικοποιητή ο οποίος επαναφέρει την αναπαράσταση αυτή στην αρχική μορφή της αναπαράστασης των δεδομένων εισόδου. Οι Autoencoders ανήκουν στην κατηγορία των Νευρωνικών Δικτύων και είναι Unsupervised ML αλγόριθμοι. Ένα συχνά εμφανιζόμενο πρόβλημα σε εφαρμογές τεχνητής νοημοσύνης είναι η εύρεση και εξαγωγή χαρακτηριστικών υψηλού επιπέδου

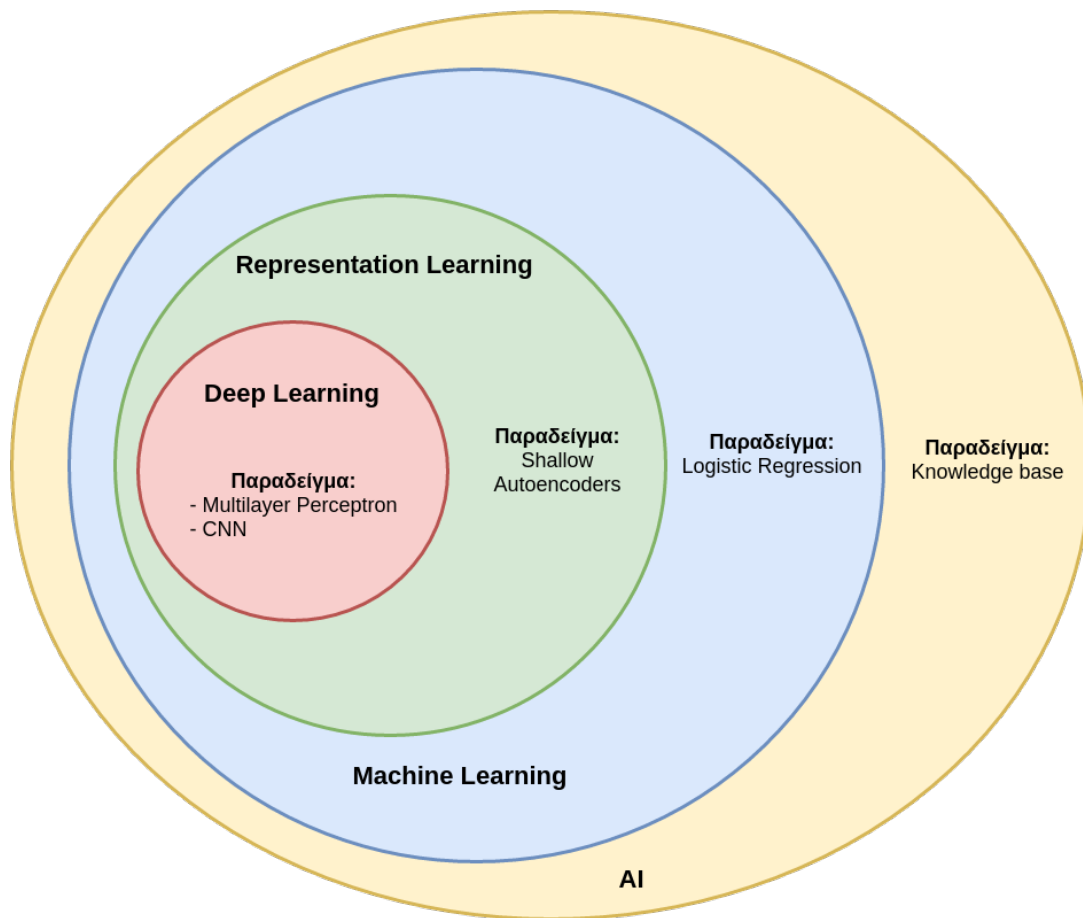


Σχήμα 3.4: Παράδειγμα απεικόνισης των φίλτρων ενός μοντέλου CNN για αναγνώριση προσώπου

από τα δεδομένα. Τα μοντέλα *Βαθιάς Μηχανικής Μάθησης* δίνουν λύσεις σε αυτό το πρόβλημα εκμάθησης αναπαραστάσεων με την εισαγωγή αναπαραστάσεων οι οποίες εκφράζονται με βάση άλλες, απλούστερες αναπαραστάσεις. Αυτή η προσέγγιση δίνει την δυνατότητα στους υπολογιστές να κατασκευάζουν σύνθετες έννοιες χρησιμοποιώντας πιο απλές έννοιες. Για παράδειγμα, η αναγνώριση αντικειμένων μπορεί να εκφραστεί με έννοιες όπως το γεωμετρικό σχήμα των αντικειμένων, το οποίο με την σειρά του ορίζεται από γωνίες και περιγράμματα. Περαιτέρω, οι γωνίες και τα περιγράμματα ορίζονται από ακμές. Στο [σχήμα 3.4](#), παρουσιάζονται τα φίλτρα που έμαθε ένα μοντέλο CNN για αναγνώριση προσώπων σε εικόνες. Το συγκεκριμένο μοντέλο CNN έχει 3 κρυφά επίπεδα (hidden layers); το πρώτο κρυφό επίπεδο εξάγει από τα δεδομένα εισόδου (τιμές των πίξελ) πληροφορία σχετικά με τις ακμές, το δεύτερο, έχοντας σαν είσοδο την πληροφορία παρουσίας ακμών, εξάγει πληροφορία σχετικά με τις γωνίες και τα περιγράμματα, και το τρίτο πέρνει σαν είσοδο την πληροφορία αυτή και κατασκευάζει μοντέλα αντικειμένων, δηλαδή εξάγει πληροφορία σχετικά με το γεωμετρικό σχήμα των αντικειμένων.

Συνοψίζοντας, η βαθιά μηχανική μάθηση, είναι μία προσέγγιση της σύγχρονης τεχνητής νοημοσύνης. Πιο συγκεκριμένα, είναι ένα είδος μηχανικής μάθησης, η οποία προσδίδει σε υπολογιστικά συστήματα ευφυΐα, δηλαδή την ικανότητα εξέλιξης με την χρήση εμπειρίας και δεδομένων. Σύμφωνα με τους *Ian Goodfellow, Yoshua Bengio και Aaron Courville*, η μηχανική μάθηση είναι η μόνη βιώσιμη προσέγγιση στην κατασκευή

συστημάτων ΑΙ τα οποία μπορούν να αντεπεξέλθουν σε πολύπλοκα περιβάλλοντα και προβλήματα [4]. Η βαθιά μηχανική μάθηση καταφέρνει να μαθίνει να αναπαριστά τον κόσμο ως μία ένθετη ιεραρχία εννοιών όπου η κάθε έννοια ορίζεται σε σχέση με άλλες πιο απλές έννοιες, και πιο αφηρημένες μορφές αναπαραστάσεων σε σχέση με λιγότερο αφηρημένες. Το διάγραμμα Venn που φαίνεται στο [σχήμα 3.5](#) δείχνει η βαθιά μηχανική μάθηση ανήκει στην κατηγορία της εκμάθησης αναπαραστάσεων, η οποία με την σειρά της είναι ένα είδος μηχανικής μάθησης που χρησιμοποιείται για την κατασκευή νοούμενων συστημάτων.



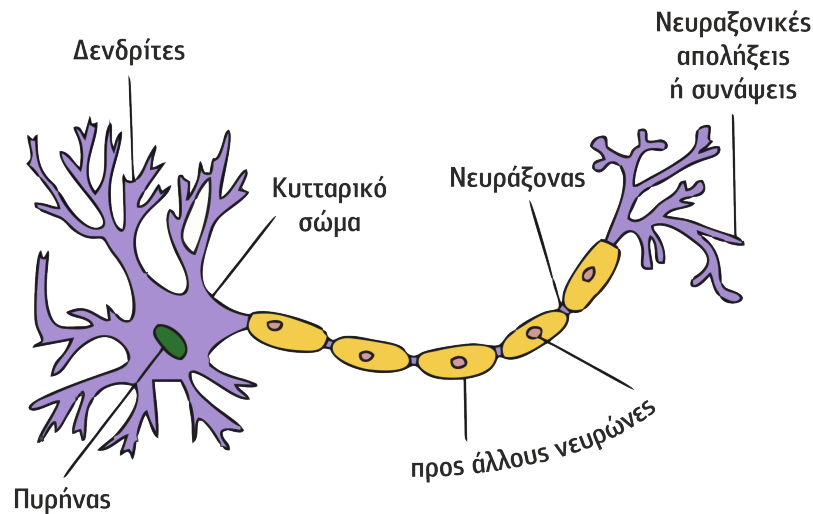
Σχήμα 3.5: Παράδειγμα απεικόνισης των φίλτρων ενός μοντέλου CNN για αναγνώριση προσώπου

3.2 ΝΕΥΡΩΝΙΚΑ ΔΙΚΤΥΑ ΜΕ ΒΑΘΟΣ

Τα Νευρωνικά Δίκτυα είναι εμπνευσμένα από το βιολογικό νευρικό σύστημα του του ανθρώπου. Η βασική επεξεργαστική μονάδα του εγκεφάλου είναι ο νευρώνας. Το ανθρώπινο νευρικό σύστημα αποτελείτε από περίπου 86 εκατομμύρια νευρώνες και περίπου $10^{14} - 10^{15}$ διασυνδέσεις. Στο [σχήμα 3.6](#) φαίνεται η μορφή και τα μέλη ενός βιολογικού νευρώνα. Τα κύρια μέρη ενός νευρώνα είναι τα εξής:

- **Δενδρίτης (Dendrites):** Δέχεται είσοδο από άλλους νευρώνες.

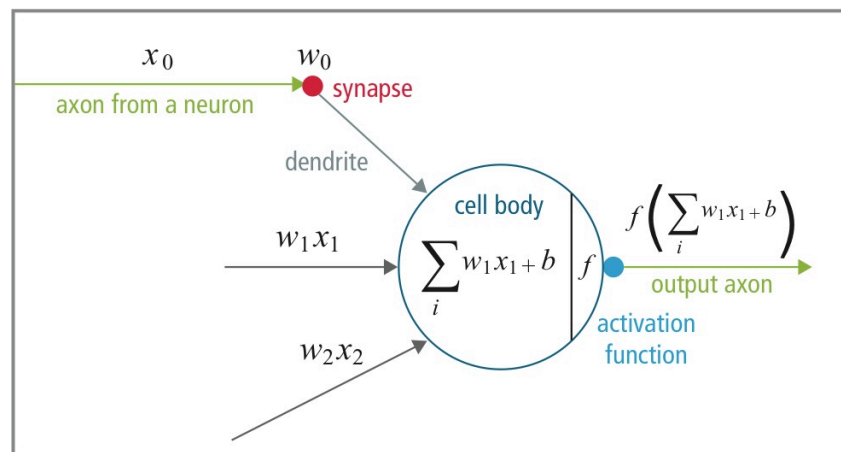
ΚΕΦΑΛΑΙΟ 3. ΤΕΧΝΙΚΕΣ ΒΑΘΙΑΣ ΜΗΧΑΝΙΚΗΣ ΜΑΘΗΣΗΣ ΚΑΙ ΑΝΑΓΝΩΡΙΣΗ ΑΝΤΙΚΕΙΜΕΝΩΝ ΣΤΟΝ ΧΩΡΟ



Σχήμα 3.6: Βιολογικός νευρώνας.

- **Σώμα του κυττάρου (Cell body):** Εξάγει συμπεράσματα, με βάση τις εισόδους.
- **Νευράξονας (Axon):** Συνδέει την έξοδο που λαμβάνεται από το σώμα του κυττάρου με τις νευρωνικές απολήψεις
- **Νευραξονικές απολήψεις:** Συνδέει τον νευράξονα του εκάστοτε νευρώνα με τους τερματικούς κόμβους από όπου και μεταφέρεται η πληροφορία σε άλλους νευρώνες. είσοδο άλλων νευρώνων.

Κάθε νευρώνας δέχεται είσοδο από άλλους νευρώνες μέσω των δενδρίτων του. Στην συνέχεια επεξεργάζεται το σήμα που λαμβάνει στην είσοδο και στέλνει το αποτέλεσμα στον νευράξονα. Τέλος άλλοι νευρώνες συνδέονται με αυτόν μέσω των συνάψεων του.



Σχήμα 3.7: Μαθηματικό μοντέλο του νευρώνα

Το αντίστοιχο μαθηματικό μοντέλο του νευρώνα, φαίνεται στο [σχήμα 3.7](#). Η πληροφορία που μεταφέρεται από τις νευραξονικές απολήψεις (x_0), πρώτου στους δενδρίτες των επόμενων νευρώνων, αλληλεπιδρά πολλαπλασιαστικά με τις συνάψεις ($w_0 * x_0$). Οι πολλαπλασιαστικοί παράγοντες w_n ονομάζονται βάρη και αποτελούν τις μεταβλητές παραμέτρους ενός νευρώνα. Η τιμή των παραμέτρων αυτών ελέγχουν την επίδραση μεταξύ των νευρώνων. Η συνάρτηση ενεργοποίησης f ελέγχει την ροή της πληροφορίας στους συνδεδεμένους νευρώνες, και προσδίδει ευελιξία και ικανότητα εκτίμησης όσον αφορά πολύπλοκες μή γραμμικές σχέσεις στα δεδομένα εισόδου. Η πιο συχνά χρησιμοποιούμενη και απλή στην λειτουργία συνάρτηση ενεργοποίησης είναι η σιγμοειδής συνάρτηση $\sigma(x) = 1/(1+e^{-x})$. Εναλλακτικά, η σιγμοειδής συνάρτηση ενεργοποίησης μπορεί να εκφραστεί σε διακριτή μορφή ως

$$f(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$$

Γενικότερα, ο νευρώνας μπορεί να είναι και πολωμένος (bias - b) και έτσι το μαθηματικό μοντέλο που τον περιγράφει πλήρως παίρνει την μορφή:

$$f(\sum_i \vec{X}_i \vec{W}_i + b) = \frac{1}{1+e^{-\sum_i \vec{X}_i \vec{W}_i + b}}$$

Θα μπορούσαμε να ερμηνεύσουμε το αποτέλεσμα της εφαρμογής της σιγμοειδής συνάρτησης ενεργοποίησης ως την πιθανότητα μίας από τις κλασεις:

$$P(y_i = 1|x_i; w) \\ P(y_i = 0|x_i; w) = 1 - P(y_i = 1|x_i; w)$$

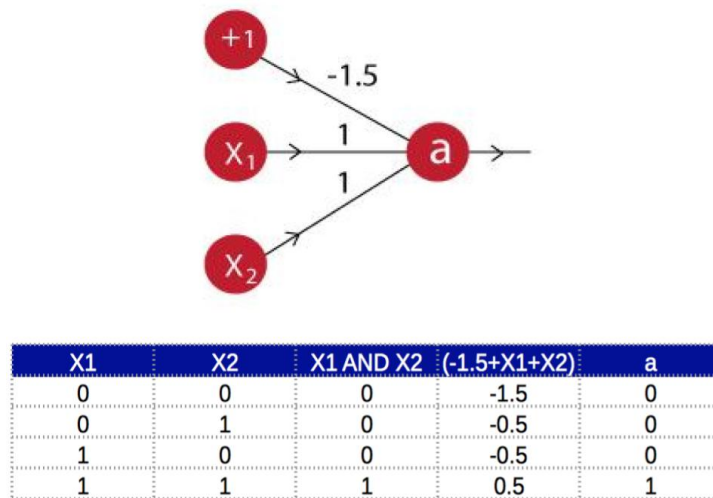
Σαν παρατήρηση, με εφαρμογή κατάλληλης συνάρτησης σφάλματος στην έξοδο, ο νευρώνας μετατρέπεται σε ένα γραμμικό ταξινομητή (linear classifier). Πιο συγκεκριμένα, σε περίπτωση που χρησιμοποιήσουμε την *cross-entropy* συνάρτηση σφάλματος ο νευρώνας μετατρέπεται σε δυαδικό ταξινομητή **Softmax**, τον οποίο και θα συναντήσουμε στην συνέχεια.

Μία από τις τρεις θεμελιώδεις εφαρμογές του μαθηματικού μοντέλου του νευρώνα είναι η υλοποίηση της δυαδικής πράξης **AND**, η οποία και φαίνεται στο [σχήμα 3.8](#). Οι άλλες δύο είναι προφανώς οι πράξεις **OR** και **NOT**.

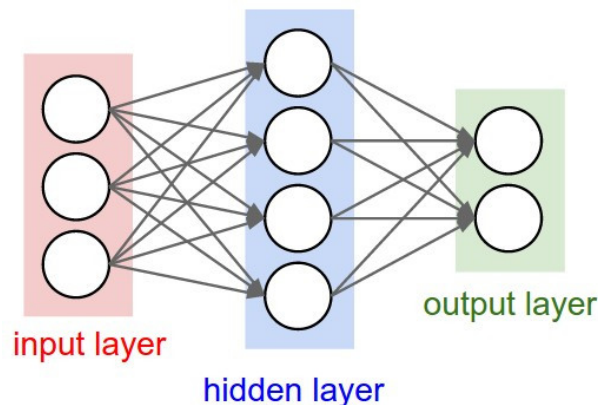
Η σύνδεση πολλών νευρώνων σε έναν γράφο δομεί ένα *Νευρωνικό Δίκτυο*, το οποίο έχει την μορφή του σχήματος [σχήμα 3.9](#). Το συγκεκριμένο μοντέλο NN ονομάζεται *Perceptron* ή αλλιώς *Feedforward Artificial Neural Network*, ο οποίος έχει τα εξής χαρακτηριστικά:

- Οι διασυνδέσεις μεταξύ των νευρώνων δεν σχηματίζουν σε καμία περίπτωση κύκλο.
- Αποτελείτε από 2 επίπεδα; ένα κρυφό επίπεδο και το επίπεδο εξόδου
- Χρησιμοποιείται η σιγμοειδής συνάρτηση ενεργοποίησης

Τα μοντέλα NN συχνά οργανώνονται σε διακριτά επίπεδα από νευρώνες, όπου οι νευρώνες που ανήκουν σε ένα επίπεδο συνδέονται με νευρώνες του προηγούμενου



Σχήμα 3.8: Υλοποίηση πύλης AND με χρήση του μαθηματικού μοντέλου του perceptron



Σχήμα 3.9: Απλό μοντέλο NN με ένα κρυφό επίπεδο

και του επόμενου επιπέδου. Όσον αφορά την αρίθμηση των επιπέδων ενός νευρωνικού δικτύου, αναφέρουμε ότι συμβατικά το πρώτο επίπεδο δεν υπολογίζεται αφού πρακτικά αποτελεί τα ίδια τα δεδομένα εισόδου.

Η επιλογή εφαρμογής της σιγμοειδής συνάρτησης, ως συνάρτηση ενεργοποίησης των νευρώνων που αποτελούν ένα NN δεν είναι τυχαία, αφού όπως θα δούμε στην συνέχεια επιτρέπει την εφαρμογή του αλγορίθμου *Backpropagation* για την εκπαίδευση των νευρωνικών δικτύων.

Ο [αλγόριθμος 3.1](#) υλοποιεί την συνάρτηση για τον υπολογισμό της εξόδου ενός νευρωνικού δικτύου (forward pass), έχοντας σαν δεδομένα τα βάρη και τις τιμές της πόλωσης των νευρώνων του κάθε επιπέδου, καθώς και τα δεδομένα εισόδου.

3.2.1 Συναρτήσεις Ενεργοποίησης

TODO!

Αλγόριθμος 3.1 Υπολογισμός εξόδων ενός NN

```
1: procedure NN_FORWARD( $X, W, B, \text{num\_layers}$ )
2:    $k \leftarrow 0$ 
3:   while  $k < \text{num\_layers}$  do
4:      $X \leftarrow 1/(1 + e^{-\sum_i X \odot W_k + B_k})$ 
5:      $k \leftarrow k + 1$ 
6:   end while
7: end procedure
```

3.2.2 Ο αλγόριθμος Backpropagation

TODO!

3.2.3 Αρχιτεκτονικές

TODO!

3.3 ΝΕΥΡΩΝΙΚΑ ΔΙΚΤΥΑ ΣΥΝΕΛΙΞΗΣ

3.4 ΕΦΑΡΜΟΓΕΣ ΝΕΥΡΩΝΙΚΩΝ ΔΙΚΤΥΩΝ ΣΥΝΕΛΙΞΗΣ ΣΤΗΝ ΑΝΑΓΝΩ- ΡΙΣΗ ΑΝΤΙΚΕΙΜΕΝΩΝ ΣΤΟΝ ΧΩΡΟ

4

Εργαλεία Hardware/Software που χρησιμοποιήθηκαν

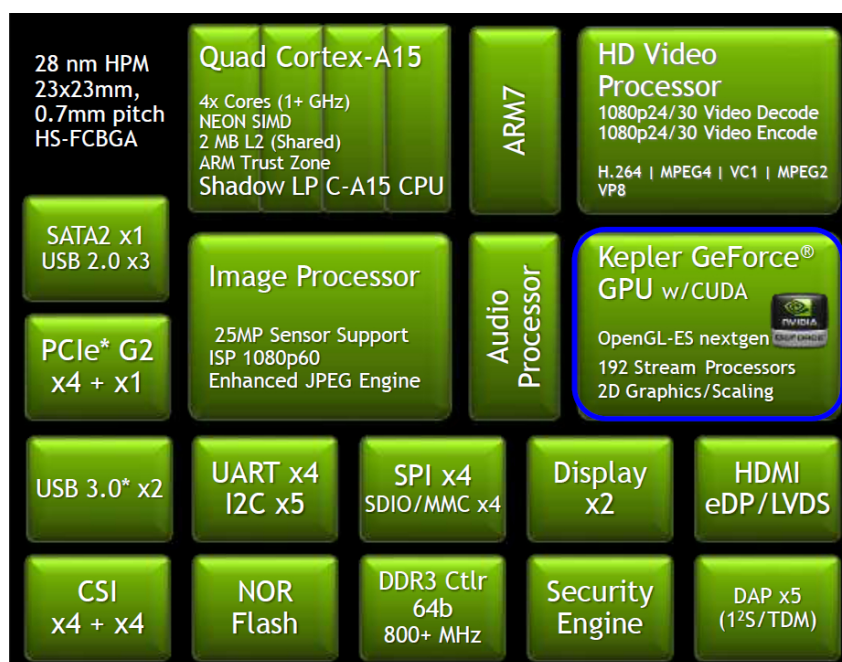
Στο κεφάλαιο αυτό παρουσιάζονται τα βασικά εργαλεία που χρησιμοποιήθηκαν τόσο για τις υλοποιήσεις όσο και για τα πειράματα. Στο [υποκεφάλαιο 4.1](#) παρουσιάζεται το ενσωματωμένο σύστημα της NVIDIA, Jetson TK1, ενώ τα εργαλεία λογισμικού που χρησιμοποιήθηκαν παρουσιάζονται στο [υποκεφάλαιο 4.2](#).

4.1 NVIDIA JETSON TK1 DEVELOPMENT BOARD

Στο κεφάλαιο αυτό παρουσιάζεται η ενσωματωμένη πλατφόρμα Jetson TK1 της NVIDIA, η οποία χρησιμοποιήθηκε για εφαρμογή των υλοποιήσεων για *Αναγνώριση και Εντοπισμό Αντικειμένων* σε συστήματα πραγματικού χρόνου, με Νευρωνικά Δίκτυα Συνέλιξης.

Ο *Tegra K1* είναι το πρώτο SOC της NVIDIA, για φορητές συσκευές, με προηγμένη αρχιτεκτονική και χαρακτηριστικά, καθώς και χαμηλή κατανάλωση ισχύος. Η μέγιστη κατανάλωση είναι στα *3Watt*, τιμή που δικαιολογεί την ενσωμάτωσή του σε φορητές συσκευές (tablets, smartphones), καθώς και σε προηγμένα ενσωματωμένα συστήματα (embedded systems) για εφαρμογές πραγματικού χρόνου.

4.1. NVIDIA JETSON TK1 DEVELOPMENT BOARD



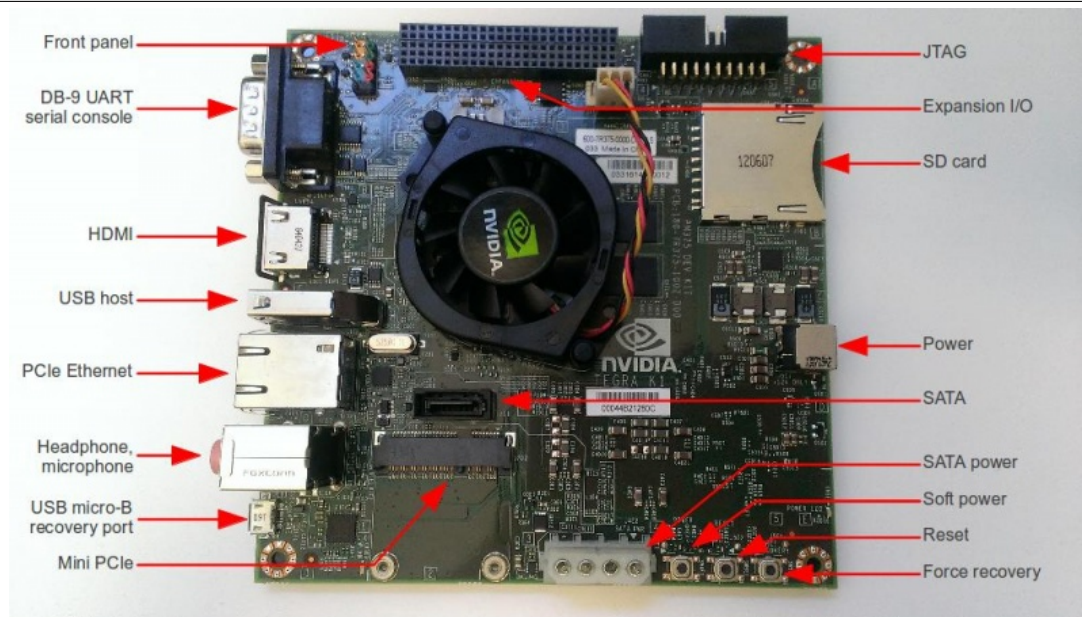
Σχήμα 4.1: Tegra K1 SOC

Όπως φαίνεται στο [σχήμα 4.1](#), τα βασικά τεχνικά χαρακτηριστικά του Tegra K1 SOC:

- CPU: Quad-core ARM Cortex-A15 CPU, 2.3Ghz
- GPU: GK20A Kepler-based GPU with 192 CUDA cores
- RAM: DDR3L/LPDDR3, up to 8GB
- Peripherals I/O: USB, eMMC/SD-card, LVDS, HDMI, SPI, UART, I2C, SATA, PCIe
- ISP: Image processor

Στα πλαίσια της παρούσας διπλωματικής εργασίας, επιλέχτηκε να χρησιμοποιήσουμε το *Jetson TK1 development board* της NVIDIA, που φαίνεται στο [σχήμα 4.2](#). Το Jetson TK1 ενσωματώνει το Tegra K1 SOC (CPU+GPU+ISP) και είναι πλήρες συμβατό με διάφορες διανομές λειτουργικών συστημάτων Linux (Ubuntu, Debian, Arch, Fedora, openSUSE, Gentoo). Η πλήρης συμβατότητα και υποστήριξη λειτουργικού συστήματος Linux ήταν βασικό κριτήριο στην επιλογή του συγκεκριμένου ενσωματωμένου συστήματος αφού επιτρέπει την εγκατάσταση εργαλείων με τον ίδιο τρόπο όπως σε ένα σταθερό υπολογιστικό σύστημα (Desktop PC) το οποίο τρέχει Linux OS. Πέρα από την συμβατότητα με κλασσικές διανομές Linux OS, η NVIDIA έχει αναπτύξει δικό της λειτουργικό σύστημα, *Linux4Tegra*, το οποίο έχει σαν βάση τα Ubuntu-14.04, με κάποιες επεκτάσεις για πλήρη υποστήριξη του hardware του Jetson TK1. Επιπλέον παράγοντας στην επιλογή της συγκεκριμένης πλατφόρμας είναι η πληθώρα των περιφερειακών διεπαφών που διαθέτει, κάνοντας το χρήσιμο για εφαρμογή σε ρομποτικά συστήματα όπου η σύνδεση διαφόρων περιφερειακών

ΚΕΦΑΛΑΙΟ 4. ΕΡΓΑΛΕΙΑ HARDWARE/SOFTWARE ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΗΘΗΚΑΝ



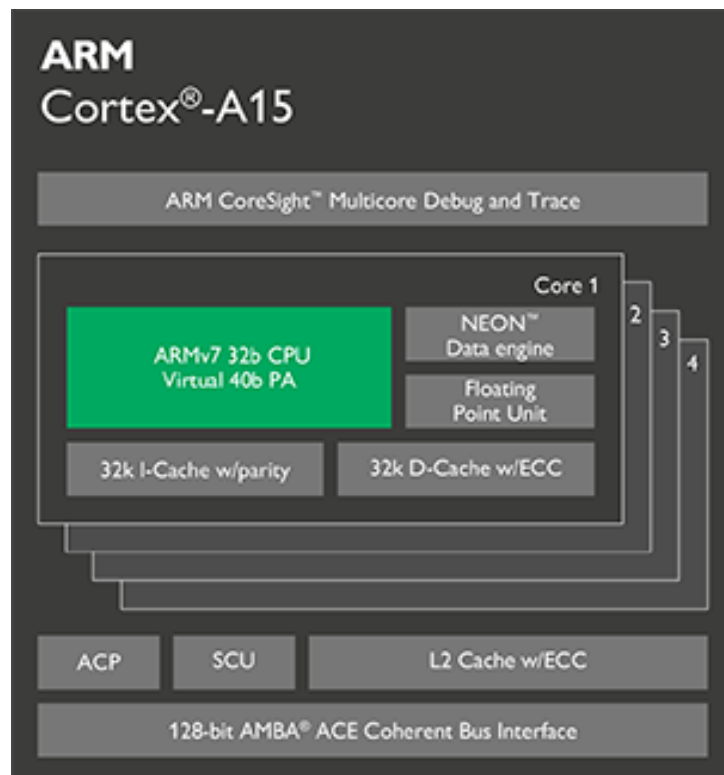
Σχήμα 4.2: Jetson TK1 development board

συσκευών, όπως για παράδειγμα αισθητήρες, κάμερες, κινητήρες, σερβο-κινητήρες, είναι απαραίτητη. Πιο κάτω δίνονται οι βασικές διεπαφές που προσφέρει η πλατφόρμα Jetson TK1:

- mini-PCIe: Σύνδεση πρόσθετων συσκευών στον δίαυλο PCI-Express όπως, Wifi cards, SSD δίσκων, κτλ.
- USB 2.0 port: Για σύνδεση συσκευών ή/και αισθητήρων με διεπαφή eHCI (Extended Host Controller Interface)
- USB 3.0 port: Για σύνδεση συσκευών ή/και αισθητήρων με διεπαφή xHCI (eXtensible Host Controller Interface)
- HDMI: Δίνει την δυνατότητα σύνδεσης οθόνης
- RS232 port: Παρόλο που το RS232 είναι αρκετά παλιό πρωτόκολλο επικοινωνίας, ακόμη ενσωματώνεται σε διάφορες συσκευές που δεν απαιτούν μεγάλο όγκο μεταφοράς δεδομένων, όπως οι οδηγοί κινητήρων
- Audio IO
- Gigabit Ethernet LAN: Δικτύωση της πλατφόρμας με τον "έξω" κόσμο
- SATA: Επιτρέπει την σύνδεση σκληρού δίσκου SATA
- JTAG port: Το JTAG προσφέρει την δυνατότητα σύνδεσης συσκευής/προγράμματος εντοπισμού σφαλμάτων (debugger), για επαγγελματική αποσφαλμάτωση
- UART port
- I2C ports: Διαθέτει τρεις θύρες I2C για σύνδεση αισθητήρων/συσκευών που οδηγούνται με το συγκεκριμένο πρωτόκολλο
- GPIO: Προσφέρει δύο θύρες επέκτασης (expansion ports), με 50 και 75 ακροδέκτες αντίστοιχα. Χρησιμο κυρίως για, επικοινωνία συσκευών με SPI, οδήγηση σερβο-κινητήρων, διακλάδωση τροφοδοσίας σε τρίτες συσκευές, κτλ.

4.2. FULL SOFTWARE STACK FOR DEVELOPING DEEP NEURAL NETWORKS

Όσον αφορά την υλοποίηση και ανάπτυξη Νευρωνικών Δικτύων σε ενσωματωμένα συστήματα, η πλατφόρμα Jetson TK1 θεωρείται ιδανική αφού υποστηρίζει CUDA και cuDNN. Η cuDNN (CUDA Deep Neural Network library) είναι GPU-accelerated βιβλιοθήκη για Νευρωνικά Δίκτυα, η οποία αναπτύχθηκε από την NVIDIA και προσφέρει υψηλού επιπέδου υλοποιήσεις για ρουτίνες όπως συνέλιξη, κανονικοποίηση δεδομένων, pooling, επίπεδα ενεργοποίησης, κτλ. Η βιβλιοθήκη cuDNN χρησιμοποιείται και ενσωματώνεται στα πιο δημοφιλή σήμερα frameworks για σχεδίαση και υλοποίηση μοντέλων DNN, όπως Caffe [12], Tensorflow [13], Theano [14][15][16], Torch [17][18][19], CNTK και Keras.



Σχήμα 4.3: ARM Cortex-A15 processor

Ένα από τα μειονεκτήματα της πλατφόρμας Jetson TK1 είναι αρχιτεκτονική των 32 bit του επεξεργαστή ARM Cortex-A15 MPCore που έχει ενσωματωμένο (σχήμα 4.3). Με την εμφάνιση των επεξεργαστών ARM Cortex-A57, οι οποίοι είναι αρχιτεκτονικής 64 bit, η υποστήριξη σε λογισμικά, τόσο από την πλευρά της NVIDIA όσο και από την ανοικτού-κώδικα (open-source) κοινότητα, μειώθηκε. Η NVIDIA σταμάτησε να υποστηρίζει μηχανήματα αρχιτεκτονικής 32 bit, τις βιβλιοθήκες CUDA και cuDNN, από την έκτη και δεύτερη έκδοση αντίστοιχα. Η CUDA είναι σήμερα στην έκδοση 8 και η cuDNN στην έκδοση 5, οι οποίες φέρουν τρομερές αναβαθμίσεις στην επίδοση των αλγορίθμων βαθιάς εκμάθησης (<https://developer.nvidia.com/cudnn>).

4.2 FULL SOFTWARE STACK FOR DEVELOPING DEEP NEURAL NETWORKS

5

Implementations

TODO Introduction here!
Implementations on i7 and jetsonTK1 board

5.1 YOLO IMPLEMENTATION WITH KERAS DNN FRAMEWORK

TODO!!

5.2 DESKTOP PC CONFIGURATION FOR OPTIMAL PERFORMANCE

TODO!!

5.3 JETSON TK1 DEV BOARD SETUP AND APPLIED OPTIMIZATIONS

6

Πειράματα - Αποτελέσματα

TODO!!

7

Συμπεράσματα

TODO!!

8

Μελλοντικές επεκτάσεις

TODO!!

Βιβλιογραφία

- [1] Murray Campbell, A Joseph Hoane, and Feng-hsiung Hsu. “*Deep blue*“. Artificial intelligence, 134(1):57–83, 2002.
- [2] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. “*ImageNet Classification with Deep Convolutional Neural Networks*“. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, “*Advances in Neural Information Processing Systems 25*“, pages 1097–1105. Curran Associates, Inc., 2012.
- [3] Firas Abuzaid. “*Optimizing CPU Performance for Convolutional Neural Networks*“.
- [4] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. “*Deep Learning*“. Book in preparation for MIT Press, 2016. URL <http://www.deeplearningbook.org>.
- [5] Yoshua Bengio, Aaron Courville, and Pascal Vincent. “*Representation learning: A review and new perspectives*“. IEEE transactions on pattern analysis and machine intelligence, 35(8):1798–1828, 2013.
- [6] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. “*Gradient-based learning applied to document recognition*“. Proceedings of the IEEE, 86(11):2278–2324, 1998.
- [7] Matthew D. Zeiler and Rob Fergus. “*Visualizing and Understanding Convolutional Networks*“. CoRR, abs/1311.2901, 2013.
- [8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. “*Deep Residual Learning for Image Recognition*“. CoRR, abs/1512.03385, 2015.
- [9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. “*Identity Mappings in Deep Residual Networks*“. CoRR, abs/1603.05027, 2016.
- [10] Christian Szegedy, Alexander Toshev, and Dumitru Erhan. “*Deep neural networks for object detection*“. In “*Advances in Neural Information Processing Systems*“, pages 2553–2561, 2013.
- [11] Pierre Baldi. “*Autoencoders, unsupervised learning, and deep architectures.*“. ICML unsupervised and transfer learning, 27(37-50):1, 2012.
- [12] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. “*Caffe: Convolutional Architecture for Fast Feature Embedding*“. arXiv preprint arXiv:1408.5093, 2014.

- [13] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek Gordon Murray, Benoit Steiner, Paul A. Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zhang. “*TensorFlow: A system for large-scale machine learning*“. CoRR, abs/1605.08695, 2016.
- [14] Rami Al-Rfou, Guillaume Alain, Amjad Almahairi, Christof Angermueller, Dzmitry Bahdanau, Nicolas Ballas, Frédéric Bastien, Justin Bayer, Anatoly Belikov, Alexander Belopolsky, Yoshua Bengio, Arnaud Bergeron, James Bergstra, Valentin Bisson, Josh Blecher Snyder, Nicolas Bouchard, Nicolas Boulanger-Lewandowski, Xavier Bouthillier, Alexandre de Brébisson, Olivier Breuleux, Pierre-Luc Carrier, Kyunghyun Cho, Jan Chorowski, Paul Christiano, Tim Cooijmans, Marc-Alexandre Côté, Myriam Côté, Aaron Courville, Yann N. Dauphin, Olivier Delalleau, Julien Demouth, Guillaume Desjardins, Sander Dieleman, Laurent Dinh, Mélanie Ducoffe, Vincent Dumoulin, Samira Ebrahimi Kahou, Dumitru Erhan, Ziyi Fan, Orhan Firat, Mathieu Germain, Xavier Glorot, Ian Goodfellow, Matt Graham, Caglar Gulcehre, Philippe Hamel, Iban Harlouchet, Jean-Philippe Heng, Balázs Hidasi, Sina Honari, Arjun Jain, Sébastien Jean, Kai Jia, Mikhail Korobov, Vivek Kulkarni, Alex Lamb, Pascal Lamblin, Eric Larsen, César Laurent, Sean Lee, Simon Lefrancois, Simon Lemieux, Nicholas Léonard, Zhouhan Lin, Jesse A. Livezey, Cory Lorenz, Jeremiah Lowin, Qianli Ma, Pierre-Antoine Manzagol, Olivier Mastropietro, Robert T. McGibbon, Roland Memisevic, Bart van Merriënboer, Vincent Michalski, Mehdi Mirza, Alberto Orlandi, Christopher Pal, Razvan Pascanu, Mohammad Pezeshki, Colin Raffel, Daniel Renshaw, Matthew Rocklin, Adriana Romero, Markus Roth, Peter Sadowski, John Salvatier, François Savard, Jan Schlüter, John Schulman, Gabriel Schwartz, Iulian Vlad Serban, Dmitriy Serdyuk, Samira Shabanian, Étienne Simon, Sigurd Spieckermann, S. Ramana Subramanyam, Jakub Sygnowski, Jérémie Tanguay, Gijs van Tulder, Joseph Turian, Sebastian Urban, Pascal Vincent, Francesco Visin, Harm de Vries, David Warde-Farley, Dustin J. Webb, Matthew Willson, Kelvin Xu, Lijun Xue, Li Yao, Saizheng Zhang, and Ying Zhang. “*Theano: A Python framework for fast computation of mathematical expressions*“. arXiv e-prints, abs/1605.02688, May 2016.
- [15] James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. “*Theano: a CPU and GPU Math Expression Compiler*“. In “*Proceedings of the Python for Scientific Computing Conference (SciPy)*“, June 2010. Oral Presentation.
- [16] Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, James Bergstra, Ian J. Goodfellow, Arnaud Bergeron, Nicolas Bouchard, and Yoshua Bengio. “*Theano: new features and speed improvements*“. Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop, 2012.

- [17] Ronan Collobert, Samy Bengio, and Johnny Mariéthoz. “*Torch: a modular machine learning software library*“. Technical report, Idiap, 2002.
- [18] Ronan Collobert, Koray Kavukcuoglu, and Clément Farabet. “*Torch7: A matlab-like environment for machine learning*“. In “*BigLearn, NIPS Workshop*“, number EPFL-CONF-192376, 2011.
- [19] Ronan Collobert, Koray Kavukcuoglu, and Clément Farabet. “*Implementing neural networks efficiently*“. In “*Neural Networks: Tricks of the Trade*“, pages 537–557. Springer, 2012.