ΔΙΙΔΡΥΜΑΤΙΚΟ ΔΙΑΤΜΗΜΑΤΙΚΟ ΠΡΟΓΡΑΜΜΑ
ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ
ΣΤΡΑΤΙΩΤΙΚΗΣ ΣΧΟΛΗΣ ΕΥΕΛΠΙΔΩΝ ΚΑΙ
ΣΧΟΛΗΣ ΜΗΧΑΝΙΚΩΝ ΠΑΡΑΓΩΓΗΣ ΚΑΙ ΔΙΟΙΚΗΣΗΣ
ΤΟΥ ΠΟΛΥΤΕΧΝΕΙΟΥ ΚΡΗΤΗΣ

ΣΥΓΓΡΑΦΗ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Του:

<Νικόλαου Τσαβαλή >

Ανίχνευση [προαιρετικά και ταξινόμηση] επικίνδυνων αντικειμένων σε χρόνο πραγματικό (real time) από δεδομένα video, με χρήση μεθόδων βαθιάς μάθησης

ΣΣΕ, Ακαδημαϊκό Έτος 2018-19

Η σελίδα αυτή είναι σκόπιμα λευκή.

Πίνακας περιεχομένων

Περίληψη

Στην παρούσα διατριβή θα γίνει μία ιστορική αναδρομή της υπολογιστικής όρασης (computer vision) και των δυνατοτήτων που προσφέρει σε ένα ευρύ φάσμα εφαρμογών και επιστημών, μέσω των συνελικτικών νευρωνικών δικτύων (ΣΝΔ). Θα αναπτυχθούν οι έννοιες της αρχιτεκτονικής δομής και τα επίπεδα (layers) των ΣΝΔ καθώς και οι τεχνικές εκπαίδευσης (training), χωρικής υποδειγματοληψίας (pooling), κανονικοποίησης (normalisation) και ενεργοποίησης (activation) που τα διέπουν. Θα δοθεί ο παραγόμενος κώδικας γραμμένος σε γλώσσα προγραμματισμού Python και οι τεχνολογίες που χρησιμοποιήθηκαν για την αναγνώριση τυχόν οπλισμού, αρχικά σε στατική εικόνα και στη συνέχεια σε βίντεο. Μέσω labelled data, θα υπάρξει η δυνατότητα αναγνώρισης του παραπάνω οπλισμού με αναγραφή του τύπου αυτού και τυχόν άλλων χαρακτηριστικών του όπλου.

Abstract

This thesis will analyze computer vision and its potential in a wide range of applications and sciences through convolutional neural networks (CNN). The concepts of architectural structure and layers of the CNN will be developed as well as the techniques of training, pooling, normalization and activation that govern them. The generated code written in Python programming language and the technologies used to identify any weaponry will be given, first in static image and then in video. Through labeled data, it will be possible to identify the above weapon by indicating this type and any other weapon characteristics.

Εικόνες

Πίνακες

Ευχαριστίες

Η παρούσα διπλωματική εργασία αποτελεί το επιστέγασμα των προσπαθειών μου στην επιτυχή ολοκλήρωση των σπουδών μου στο Μεταπτυχιακό Πρόγραμμα «Σχεδίαση και Επεξεργασία Συστημάτων (Systems Engineering)». Για το αποτέλεσμα αυτό θα ήθελα να ευχαριστήσω θερμά τον Καθηγητή κύριο Νικόλαο Παπαδάκη, χωρίς τη βοήθεια και την πολύτιμη καθοδήγησή του δε θα ήταν δυνατή η ολοκλήρωση της εργασίας αυτής. Επίσης, θα ήθελα να εκφράσω τις ειλικρινείς ευχαριστίες μου όλους τους καθηγητές μου που συνέβαλλαν σε αυτή την προσπάθεια και έγιναν συνοδοιπόροι και αρωγοί στην επίτευξη των στόχων που έθεσα. Τέλος, θα ήθελα να ευχαριστήσω τη Ζωή, την Ειρήνη και την Κωνσταντίνα, οι οποίες αποδέχθηκαν τις επιλογές μου και μου παρείχαν στήριξη όλο αυτό το διάστημα, χωρίς την οποία τίποτα από όσα έχω καταφέρει δε θα γινόντουσαν πραγματικότητα.

Συντμήσεις – Ακρώνυμα

ANN: Artificial Neural Network

CNN: Convolutional Neural Network

FN: False Negative

FP: False Positive

MLP: Multilayer Perceptron

R-CNN: Region Convolutional Neural Network

TN(en): True Negative

TP: True Positive

TPU: Tensor Processing Units

YOLO: You Only Look Once

ΣΝΔ: Συνελικτικά Νευρωνικά Δίκτυα

ΤΝ(ελ): Τεχνητή Νοημοσύνη

ΤΝΔ: Τεχνητά Νευρωνικά Δίκτυα

Κεφάλαιο 1

Ο νευρώνας, βασικό συστατικό των νευρωνικών δικτύων

1.1 Εισαγωγή

Η μεγάλη, εκθετική και εντυπωσιακή ανάπτυξη της τεχνολογίας και η συνεχής ώθηση της επιστήμης στα όρια της επιστημονικής, για το πρόσφατο παρελθόν, φαντασίας, προσφέρει την τεράστια δυνατότητα να δημιουργήσει ο άνθρωπος ένα αυτόνομο να πράττει, αλλά ακόμα πιο σημαντικό, να σκέφτεται, δημιούργημα, όμοιο με τον άνθρωπο, αλλά δημιουργημένο με κώδικα σε γλώσσα προγραμματισμού. Το παραπάνω είναι γνωστό σαν τεχνητή νοημοσύνη. Η δυνατότητα αυτή έχει ξεκινήσει ως θεωρία μερικές δεκαετίες πριν, αλλά τα τελευταία χρόνια με τη συνεχή αύξηση της υπολογιστικής ισχύος, τείνει να γίνει πράξη και να εδραιωθεί στην καθημερινή ζωή.

Οι τομείς στους οποίους εφαρμόζεται η τεχνητή νοημοσύνη είναι πάρα πολλοί, καθώς εφαρμόζονται σε ολοένα περισσότερες πτυχές και εφαρμογές της καθημερινότητας, τόσο στις επιχειρήσεις όσο και στον ιδιωτικό βίο. Η τεχνητή νοημοσύνη επηρεάζει, ανατρέπει και διαμορφώνει τη ζωή, εισάγοντας νέα δεδομένα στην αντιμετώπιση τετριμμένων και μη προβλημάτων, που άλλοτε η λύση τους, φαινόταν πολλές φορές αδύνατη.

1.2 Τεχνητή Νοημοσύνη

Ο ορισμός της Τεχνητής Νοημοσύνης (εφεξής ΤΝ) είναι πολυποίκιλος, αφού υπάρχουν διάφοροι αποδεκτοί ορισμοί. Ένας από αυτούς είναι ο εξής: “ΤΝ είναι ο τομέας της Επιστήμης των Υπολογιστών που ασχολείται με τη σχεδίαση και την υλοποίηση προγραμμάτων τα οποία είναι ικανά να μιμηθούν τις ανθρώπινες γνωστικές ικανότητες, εμφανίζοντας έτσι χαρακτηριστικά που αποδίδουμε συνήθως σε ανθρώπινη συμπεριφορά,

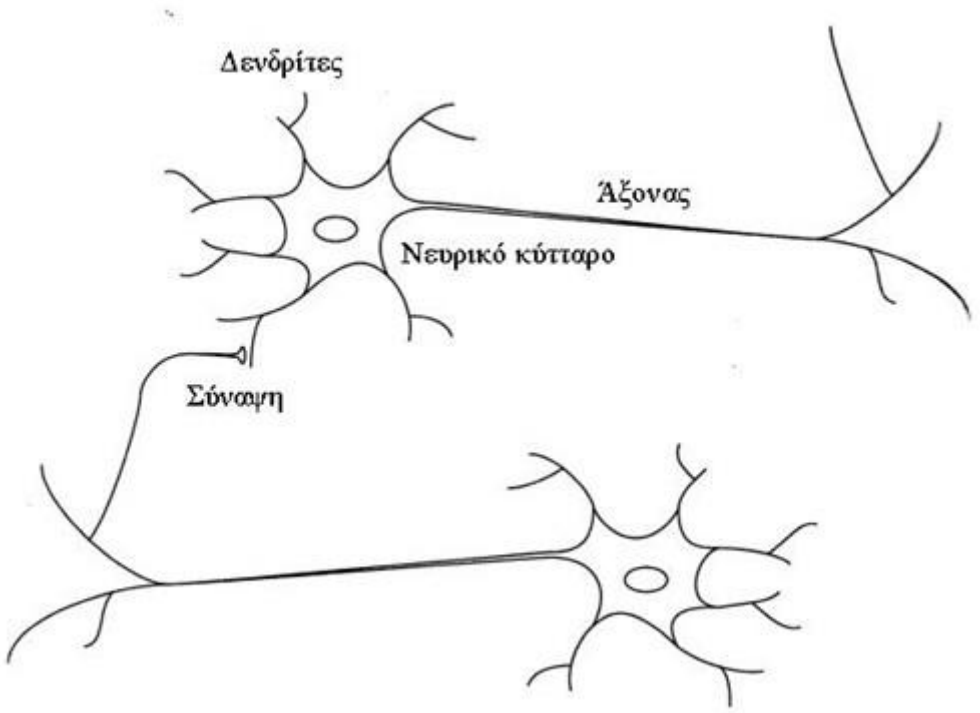
όπως η επίλυση προβλημάτων, η αντίληψη μέσω της όρασης, η μάθηση, η εξαγωγή συμπερασμάτων, η κατανόηση της φυσικής γλώσσας κτλ" [1]

Παρακάτω αναφέρονται ορισμένες εφαρμογές της ΤΝ. Τέτοιες είναι η απόδειξη θεωρημάτων, επεξεργασία φυσικής γλώσσας, η μηχανική μάθηση, αυτόνομα ρομπότ, ευφυείς πράκτορες, έμπειρα συστήματα και συστήματα γνώσης, υπολογιστική όραση. Με το πεδίο της υπολογιστικής όρασης θα ασχοληθεί η παρούσα εργασία.

1.3 Νευρώνας - Perceptron

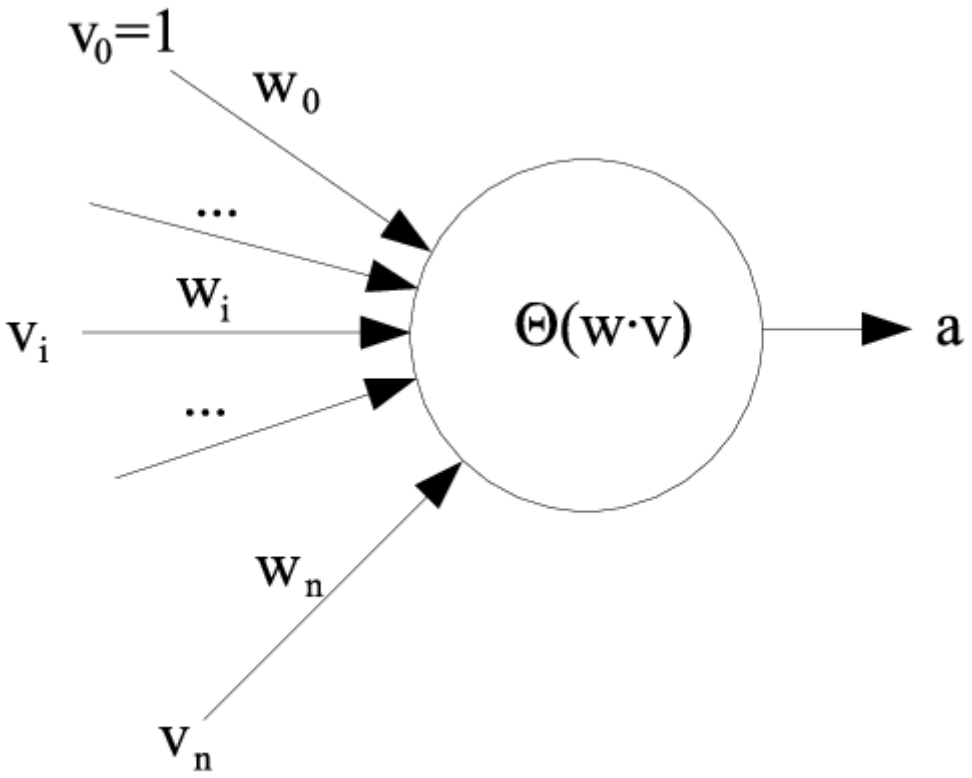
Πως προκύπτει, όμως η ΤΝ; Η ΤΝ βασίζεται στα λεγόμενα νευρωνικά δίκτυα. Τα νευρωνικά δίκτυα είναι ο κορμός της και σχεδόν όλες οι εφαρμογές χτίζονται πάνω σε αυτά, σε διάφορες παραλλαγές. Τα Τεχνητά Νευρωνικά Δίκτυα (εφεξής ΤΝΔ) με βάση τον ορισμό των **Aleksander & Morton, 1990**[2] είναι ένας τεράστιος παράλληλος επεξεργαστής με κατανεμημένη αρχιτεκτονική, ο οποίος αποτελείται από απλές μονάδες επεξεργασίας και έχει από τη φύση του τη δυνατότητα να αποθηκεύει εμπειρική γνώση και να την καθιστά διαθέσιμη για χρήση. Μοιάζει με τον ανθρώπινο εγκέφαλο σε δύο σημεία. Πρώτον, γνώση αποκτιέται από το δίκτυο με τη διαδικασία της μάθησης δηλαδή τη δοκιμή και το σφάλμα. και δεύτερον, οι δυνάμεις σύνδεσης μεταξύ των νευρώνων (ονομάζονται και συναπτικά βάρη), χρησιμοποιούνται για την αποθήκευση της αποκτηθείσας γνώσης.

Στην **Εικόνα 1** παρουσιάζεται ένας βιολογικός νευρώνας. Ο νευρώνας αυτός αποτελείται από το νευρικό κύτταρο, τον άξονά του και τους δενδρίτες. Τα ηλεκτρικά σήματα του εγκεφάλου φτάνουν από τους άλλους νευρώνες στον εκάστοτε νευρώνα, μέσω των δενδριτών, ενώ ο νευρώνας αποστέλλει τα σήματα του, μέσω του άξονα του.



Εικόνα 1. Αποτύπωση Νευρώνα Εγκεφάλου

Στην **Εικόνα 2** παρουσιάζεται το πιο απλό παράδειγμα υπολογιστικού νευρώνα, το perceptron[3]. Το perceptron έχει έναν αριθμό εισόδων (Δενδρίτες), κάθε μία εκ των οποίων έχουν μία ειδική τιμή (βάρος) Στο σώμα του perceptron, υπολογίζεται μία συνάρτηση και η έξοδός της, είναι η έξοδος του perceptron.



Εικόνα 2. Αποτύπωση Υπολογιστικού Νευρώνα

Όταν ένας νευρώνας ενεργοποιείται, υπολογίζει μία συνάρτηση από το άθροισμα των γινομένων των βαρών και των εισόδων του, και συγκρίνει την τιμή της συνάρτησης αυτής με μια τιμή κατωφλίου η οποία είναι χαρακτηριστική για τον νευρώνα αυτόν. Αν η τιμή της συνάρτησης είναι μεγαλύτερη από την τιμή κατωφλίου, τότε ο νευρώνας υπολογίζει την έξοδο. Η έξοδος αυτή χρησιμοποιείται ως είσοδος στον επόμενο νευρώνα, ο οποίος θα παράγει και αυτός αντίστοιχα μία έξοδο. Οι τιμές των βαρών αλλάζουν δυναμικά, δηλαδή κατά την διάρκεια της εκπαίδευσης, προσαρμόζονται ανάλογα το επιθυμητό αποτέλεσμα και τη μέθοδο εκπαίδευσης που χρησιμοποιείται. Όπως ακριβώς και οι βιολογικοί νευρώνες δημιουργούν ένα δίκτυο στον ανθρώπινο εγκέφαλο, έτσι και οι υπολογιστικοί νευρώνες δημιουργούν τα λεγόμενα τεχνητά νευρωνικά δίκτυα (Artificial Neural Networks -ANN), τα οποία αναλύονται στο επόμενο κεφάλαιο.

Πριν την ανάλυση των ΤΝΔ, σκόπιμο είναι να αναφερθούν συνοπτικά οι μέθοδοι που προκαλούν τις παραπάνω αλλαγές στα βάρη, δηλαδή οι τρόποι εκπαίδευσης ενός ΤΝΔ. Αυτοί είναι οι εξής: η εποπτευόμενη μέθοδος, η μη-εποπτευόμενη μέθοδος και η αυτο-εποπτευόμενη μέθοδος. Στην εποπτευόμενη μάθηση οι αρχικές τιμές των βαρών είναι τυχαίες αλλά οι τιμές των εισόδων και των εξόδων που πρέπει να μάθει το δίκτυο, είναι δεδομένες. Κατά την διαδικασία εκπαίδευσης το δίκτυο αλλάζει τις τιμές των βαρών διορθώνοντας τες, ανάλογα με την έξοδο του δικτύου και την επιθυμητή έξοδο. Στην μη-εποπτευόμενη εκπαίδευση δίνουμε μόνο την πληροφορία στο δίκτυο, αλλά δε γίνεται κάποιος έλεγχος, όπως παραπάνω. Στην αυτο-εποπτευόμενη εκπαίδευση το δίκτυο χρησιμοποιεί ένα μηχανισμό ανάδρασης (feedback) και αυτο-ελέγχει τον εαυτό του, ώστε διορθώνει τα σφάλματα στα δεδομένα. Η εκπαίδευση είναι επιτυχής όταν οι τιμές των βαρών του ΤΝΔ σταματούν να αλλάζουν.

1.4 Βασικές διαφορές μεταξύ εποπτευόμενης και μη εποπτευόμενης μάθησης

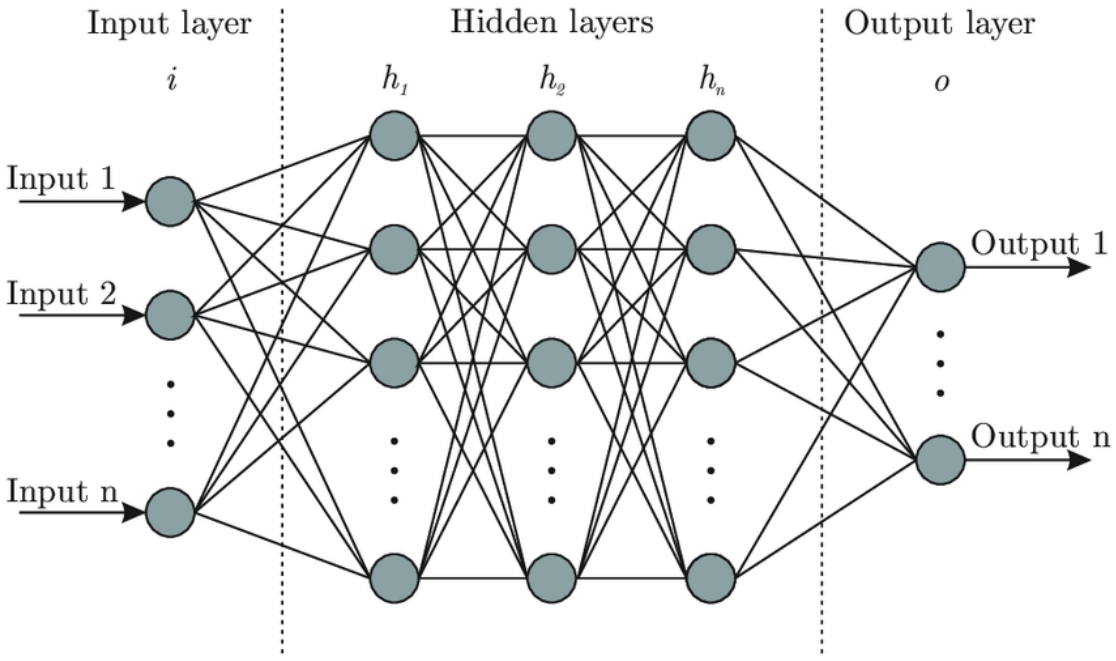
- Η εποπτευόμενη τεχνική μάθησης ασχολείται με τα επισημασμένα δεδομένα, όπου τα δεδομένα δεδομένων εξόδου είναι γνωστά στο σύστημα. Αντιθέτως, η μάθηση χωρίς επίβλεψη λειτουργεί με μη επισημασμένα δεδομένα στα οποία η παραγωγή βασίζεται απλώς στη συλλογή των αντιλήψεων.
- Όσον αφορά την πολυπλοκότητα, η εποπτευόμενη μέθοδος μάθησης είναι λιγότερο περίπλοκη ενώ η μη εποπτευόμενη μέθοδος μάθησης είναι πιο περίπλοκη.
- Η εποπτευόμενη μάθηση μπορεί επίσης να διεξάγει ανάλυση εκτός σύνδεσης, ενώ η μη επιτηρούμενη μάθηση χρησιμοποιεί ανάλυση σε πραγματικό χρόνο.
- Το αποτέλεσμα της εποπτευόμενης τεχνικής μάθησης είναι πιο ακριβές και αξιόπιστο. Αντίθετα, η μάθηση χωρίς επίβλεψη παράγει μέτρια αλλά αξιόπιστα αποτελέσματα.
- Η ταξινόμηση και η παλινδρόμηση είναι οι τύποι προβλημάτων που επιλύονται με τη μέθοδο εποπτευόμενης μάθησης. Αντιστρόφως, η μη επιτηρούμενη μάθηση περιλαμβάνει προβλήματα ομαδοποίησης και συσχετιστικού εξόρυξης κανόνα.

Κεφάλαιο 2

Τεχνητά Νευρωνικά Δίκτυα

2.1 Εισαγωγή

Όπως αναφέρθηκε στο προηγούμενο κεφάλαιο, το ΤΝΔ είναι ένα δίκτυο από υπολογιστικούς νευρώνες κατάλληλα συνδεδεμένους με ακμές οι οποίες έχουν βάρη τα οποία χρησιμοποιούνται στην εκπαίδευση του νευρωνικού δικτύου. Στην Εικόνα 3 φαίνεται ένα τυπικό ΤΝΔ. Το ΤΝΔ έχει πολλαπλές εισόδους, αριθμό ενδιάμεσων (κρυφών) επιπέδων και τουλάχιστον μία έξοδο. Παρακάτω, εξηγείται η διαδικασία μάθησης.



Εικόνα 3: Πρότυπη Αποτύπωση Τεχνητού Νευρωνικού Δικτύου

2.2 Εκπαίδευση ΤΝΔ

2.2.1 Εποπτευόμενη Μέθοδος Εκπαίδευσης

Σε αυτό τον τρόπο εκπαίδευσης, δίνονται στο νευρωνικό δίκτυο ζευγάρια διανυσμάτων εισόδου και επιθυμητής εξόδου [4]. Το δίκτυο, με βάση τη δομή του και τις τιμές των βαρών του υπολογίζει στην έξοδο, μία τιμή η οποία είναι πιθανόν να διαφέρει κατά μία ποσότητα από την επιθυμητή έξοδο. Η διαφορά της επιθυμητής και πραγματικής τιμής εξόδου ονομάζεται σφάλμα. Για να ελαχιστοποιηθεί το σφάλμα, χρησιμοποιείται ένας αλγόριθμος εκπαίδευσης, ώστε να αναπροσαρμοστούν κατάλληλα τα βάρη.

Τα διανύσματα που χρησιμοποιούνται λέγονται *δεδομένα με ετικέτα* (*labelled data).* Η εποπτευόμενη μέθοδος χωρίζεται στην *Ταξινόμηση(Classification)* και στην *Παλινδρόμηση(Regression).* Η ταξινόμηση χρησιμοποιείται για την ταξινόμηση – ομαδοποίηση των δεδομένων σε κλάσεις – ομάδες με βάση μία ή περισσότερες μεταβλητές. Παραδείγματα είναι η εύρεση ανεπιθύμητης αλληλογραφίας, ανάλυση συναισθημάτων κτλ Η Οπισθοχώρηση είναι η τεχνική για την πρόβλεψη των τιμών ενός σεναρίου, και βασίζεται σε προηγούμενες παρατηρήσιμες τιμές. Παραδείγματα είναι ο προσδιορισμός της τιμής ενός σπιτιού σε μία δεδομένη περιοχή, η εύρεση τιμής μετοχής κάτω από ορισμένες συνθήκες, το ποσοστό ύψους – βάρους ενός ανθρώπου κτλ.

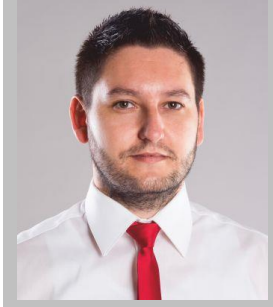
Οι αλγόριθμοι που χρησιμοποιούνται είναι οι παρακάτω[5]:

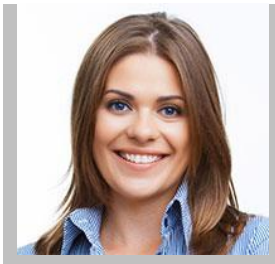
a. Support Vector Machines (SVM)
b. Linear Regression
c. Logistic Regression
d. Naïve Bayes
e. Linear Discriminant Analysis
f. Decision Trees
g. K-nearest Neighbour Algorithm (KNN)
h. Neural Networks
i. Similarity Learning

Για την εκπαίδευση του ΤΝΔ αρχικά καθορίζεται ο τύπος των παραδειγμάτων που θα χρησιμοποιηθούν ως πρότυπα. Από το σύνολο των προτύπων, εξάγονται χαρακτηριστικά τα οποία θα χρησιμοποιηθούν στις εισόδους του δικτύου. Συγκεντρώνονται όσο το δυνατόν περισσότερα χαρακτηριστικά και δηλώνονται οι επιθυμητές έξοδοι – στόχοι της εκπαίδευσης. Καθορίζεται ο τύπος του δικτύου και ο αλγόριθμος που θα χρησιμοποιηθεί. Κατά την εκπαίδευση και αφού έχει τροφοδοτηθεί ο αλγόριθμος με τις εισόδους και τους

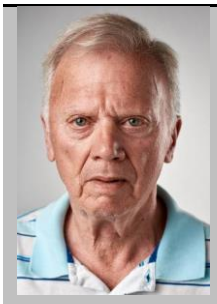
στόχους, οι τιμές των βαρών αλλάζουν, έως ότου συγκλίνουν σε συγκεκριμένες τιμές. Το δίκτυο θεωρείται ότι έχει εκπαιδευτεί και στη συνέχεια αξιολογείται η απόδοση του, εισάγοντας στο δίκτυο δεδομένα τα οποία είναι άγνωστα για αυτό μέχρι στιγμής. Η επιδίωξη της εκπαίδευσης είναι η δυνατότητα γενίκευσης του δικτύου, να μπορεί να εξάγει δηλαδή σωστά αποτελέσματα, με βάση άγνωστες για αυτό εισόδους.

Στη συνέχεια παρουσιάζονται από ένα παράδειγμα της ταξινόμησης και της παλινδρόμησης. Όπως αναφέρθηκε παραπάνω, η ταξινόμηση χρησιμοποιείται για τον προσδιορισμό της κλάσης στην οποία ανήκει ένα αντικείμενο. Η κλάση μπορεί να είναι το χρώμα του αντικειμένου, το φύλο του ανθρώπου, το είδος του φαγητού κτλ. Στο παρόν παράδειγμα θα γίνει η κατηγοριοποίηση εικόνων ανθρώπινων προσώπων στα δύο φύλα των ανθρώπων. Αν υποθέσουμε ότι, τα βιομετρικά χαρακτηριστικά του ανθρώπινου προσώπου, όπως απόσταση ματιών, μέγεθος μύτης, μετώπου κτλ μπορούν να αναπαρασταθούν με ένα διάνυσμα k τιμών με $0<k<1$, τότε η είσοδος φωτογραφιών ανθρώπινων προσώπων, θα παρήγαγε το παραπάνω διάνυσμα $v_i(k_1,k_2,\ldots,k_n)$, όπου i ο αριθμός των φωτογραφιών και n το πλήθος των βιομετρικών χαρακτηριστικών. Στη φάση της εκπαίδευσης, για κάθε φωτογραφία, δίνεται η επιθυμητή έξοδος (Άνδρας, Γυναίκα) και παράγεται η πρόβλεψη του δικτύου. Ανάλογα την επιτυχία ή όχι του δικτύου, γίνονται διορθώσεις στις τιμές των βαρών αυτού. Μετά τη φάση της εκπαίδευσης και αφού έχουν σταθεροποιηθεί οι τιμές των βαρών, δίνονται άγνωστα, με το προηγούμενο σύνολο, πρότυπα και αξιολογείται η απόδοση του δικτύου. Τα παραπάνω παρουσιάζονται στους Πίνακες 1 και 2.

Πρότυπο Εκπαίδευσης	Βιομετρικά Χαρακτηριστικά(v_i)	Στόχος	Πρόβλεψη Δικτύου	Αλλαγή Βαρών
	(0.1,0.5,…,0.8)	Άνδρας	Γυναίκα	Ναι

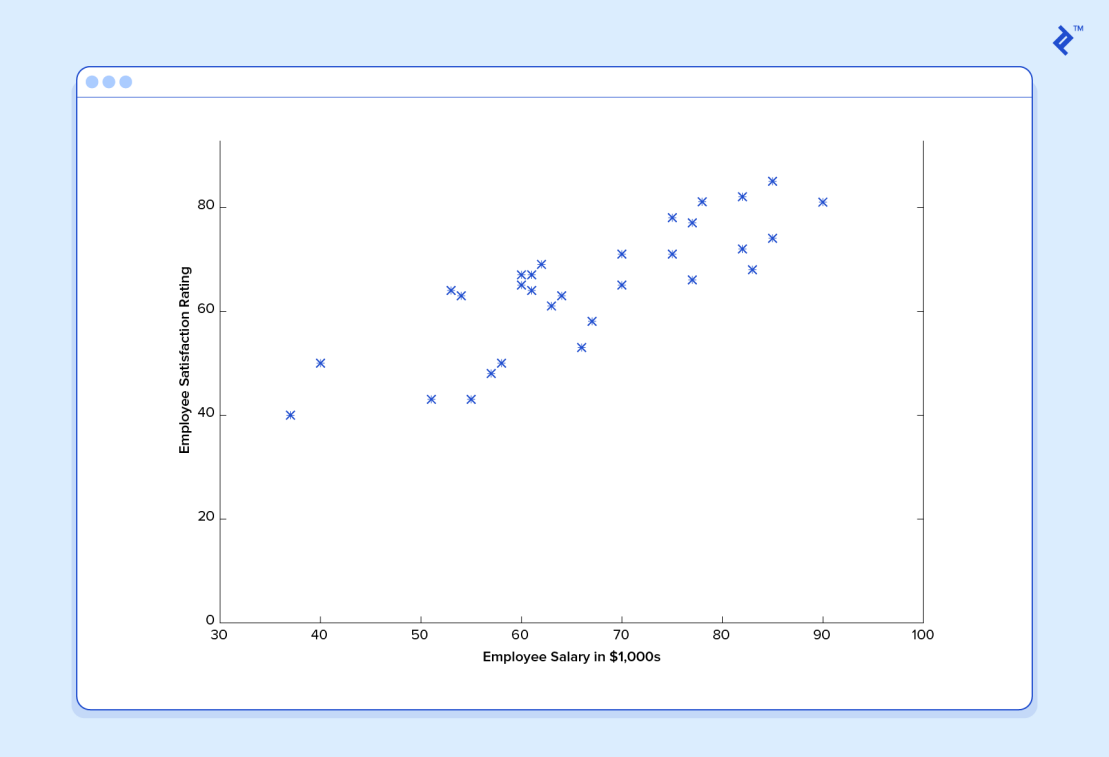
	(0.3,0.5,…,0.9)	Γυναίκα	Γυναίκα	Όχι
	(0.6,0.1,…,0.7)	Γυναίκα	Άνδρας	Ναι

Πίνακας 1. Φάση Εκπαίδευσης Δικτύου

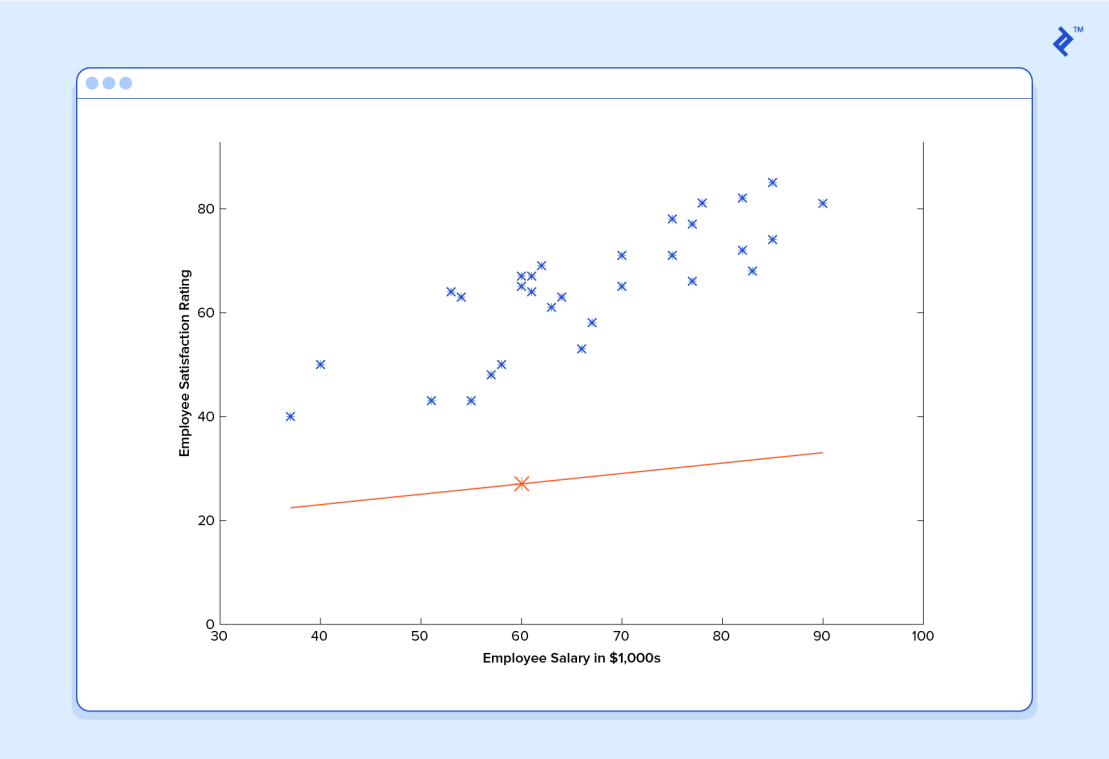
Άγνωστα Πρότυπα	Βιομετρικά Χαρακτηριστικά(v_i)	Πρόβλεψη Δικτύου	Επιτυχές
	(0.2,0.7,…,0.3)	Άνδρας	Ναι
	(0.5,0.4,…,0.5)	Γυναίκα	Ναι
	(0.6,0.7,…,0.9)	Γυναίκα	Όχι

Πίνακας 2. Φάση Αξιολόγησης Δικτύου

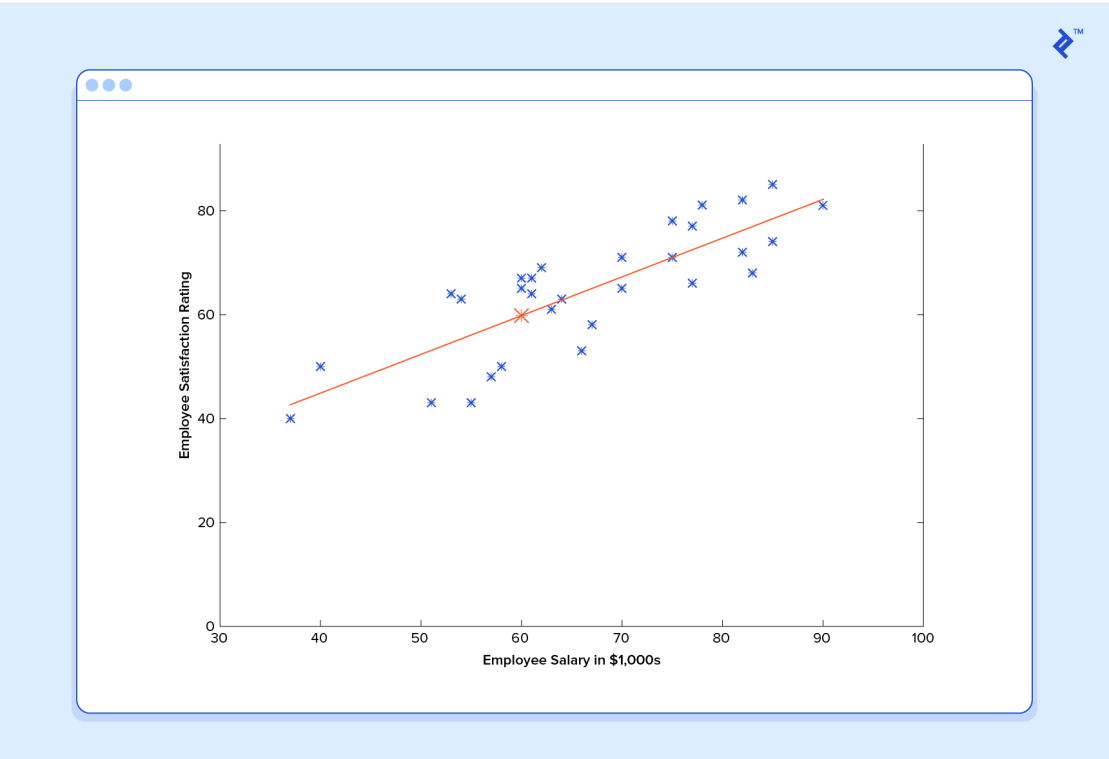
Η Παλινδρόμηση(Regression)[6] δίνει ως έξοδο μία ή περισσότερες συνεχείς αριθμητικές τιμές. Παράδειγμα της παλινδρόμησης αποτελεί η ικανοποίηση μιας εταιρείας σε σχέση με το μισθό τους. Στην Εικόνα 4 παρουσιάζονται τα δεδομένα σε γράφημα. Είναι φανερό ότι η έξοδος για την πρόβλεψη μελλοντικών δεδομένων θα είναι μία ευθεία γραμμή που θα τέμνει τους άξονες του γραφήματος. Στόχος του αλγόριθμου εκπαίδευσης είναι να ελαχιστοποιήσει το σφάλμα μεταξύ του στόχου και των προβλέψεων. Ένα τέτοιο σφάλμα, φαίνεται στην Εικόνα 5. Τελικά μετά από την εκπαίδευση των δεδομένων, λαμβάνεται μία γραμμική εξίσωση, από την οποία εξάγονται ασφαλείς τιμές σχετικά με την αξιοπιστία του δικτύου, όταν εισάγονται άγνωστα σε αυτό δεδομένα. Το τελικό αποτέλεσμα φαίνεται στην Εικόνα 6.



Εικόνα 4. Δεδομένα Εκπαίδευσης Παλινδρόμησης



Εικόνα 5. Σφάλμα Εξόδου Παλινδρόμησης

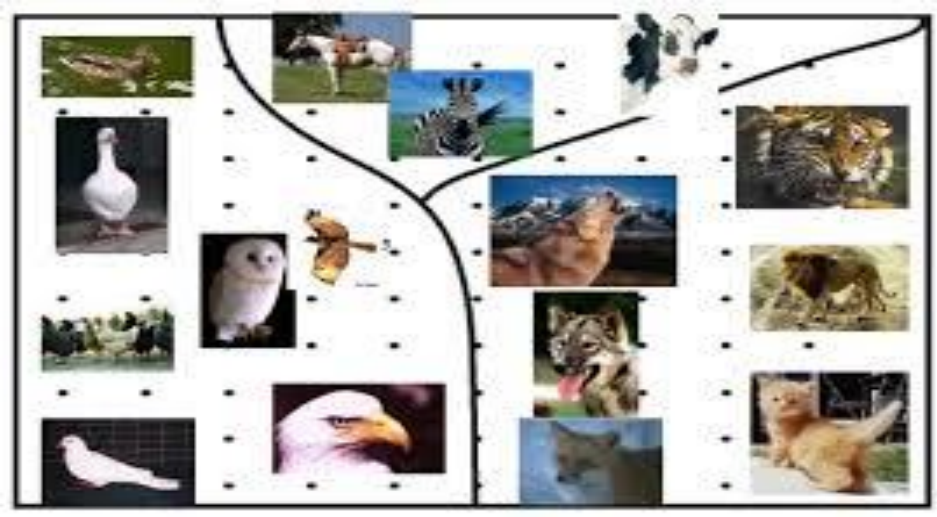


Εικόνα 6. Έξοδος Εκπαιδευμένου Δικτύου Παλινδρόμησης

2.2.2 Μη Εποπτευόμενη Μέθοδος (Unsupervised Learning)

Στη μη εποπτευόμενη μέθοδο[7] χρησιμοποιούνται δεδομένα προς επεξεργασία, χωρίς το ΤΝΔ να έχει εκπαιδευτεί με δεδομένα – ετικέτες (στόχους), όπως στην προηγούμενη μέθοδο. Έτσι, το δίκτυο καλείται να εξάγει συμπεράσματα με βάση μόνο τους δοθέντες αλγορίθμους και άγνωστα δεδομένα. Στην περίπτωση αυτή το δίκτυο αναζητεί πρότυπα (patterns) – σχέσεις στα δεδομένα, ώστε οι έξοδοι να έχουν τα ίδια χαρακτηριστικά με τις εισόδους. Όταν βρεθούν οι σχέσεις αυτές, το δίκτυο θεωρείται εκπαιδευμένο και η αξιολόγηση του θα είναι ικανοποιητική σε νέα δεδομένα. Η μη εποπτευόμενη μέθοδος προσπαθεί να προσομοιάσει τον ανθρώπινο εγκέφαλο, ώστε όταν του δίνονται άγνωστα δεδομένα, θα μπορεί να «καταλάβει» τι είναι αυτά και ποια η σχέση τους με άλλα δεδομένα.

Η μέθοδος αυτή χρησιμοποιείται για ομαδοποίηση δεδομένων (clustering) και εκτίμηση πυκνότητας δεδομένων (density estimation). Δύο περιπτώσεις χρήσης είναι η διερευνητική ανάλυση (exploratory analysis) και η μείωση διαστάσεων (dimensionality reduction). Η διερευνητική ανάλυση είναι πολύ σημαντική. καθώς επιτρέπει την αναγνώριση προτύπων και δομών σε ένα σύνολο δεδομένων. Ένα απλό παράδειγμα φαίνεται στη Εικόνα 7. Σε πιο δύσκολες, όμως περιπτώσεις, όπως η ομαδοποίηση των καταναλωτών με βάση τα προϊόντα μίας εταιρείας, η μη εποπτευόμενη μέθοδος εξάγει ικανοποιητικά αποτελέσματα, κάτι που για τον υπεύθυνο του μάρκετινγκ, για παράδειγμα, θα ήταν επίπονο και χρονοβόρο. Η μείωση διαστάσεων χρησιμοποιείται για την παρουσίαση των δεδομένων με λιγότερες στήλες ή χαρακτηριστικά. Τα χαρακτηριστικά που παρουσιάζονται κάνουν την επεξεργασία και την ανάλυση των δεδομένων λιγότερο πολύπλοκη, περισσότερο κατανοητή, και η λήψη αποφάσεων επί αυτών καθίσταται λιγότερο επισφαλής.



Εικόνα 7. Ομαδοποίηση Βάση Χαρακτηριστικών

Συνήθεις αλγόριθμοι που χρησιμοποιούνται στην μη εποπτευόμενη μέθοδο είναι οι εξής:

a. **Hierarchical clustering:** Δημιουργείται ένα δέντρο ομαδοποίησης με αποτέλεσμα να χτίζεται μία πολυεπίπεδη ιεραρχία από ομάδες

b. **k-Means clustering:** Τα δεδομένα βασίζονται σε k διακριτές ομάδες με βάση την απόστασή τους από το κέντρο κάθε ομάδας

c. **Gaussian mixture models:** Ομάδες μοντέλων ως ένα μίγμα πολυμεταβλητών συνιστωσών κανονικής πυκνότητας

d. **Self-organizing maps:** Παράγει μία χαμηλής διάστασης (συνήθως δισδιάστατη), διακριτή αναπαράσταση του χώρου εισόδου των δεδομένων εκπαίδευσης και τη διασπορά αυτών

e. **Hidden Markov models:** Χρησιμοποιεί τα παρατηρούμενα δεδομένα για να ανακτήσει την ακολουθία καταστάσεων

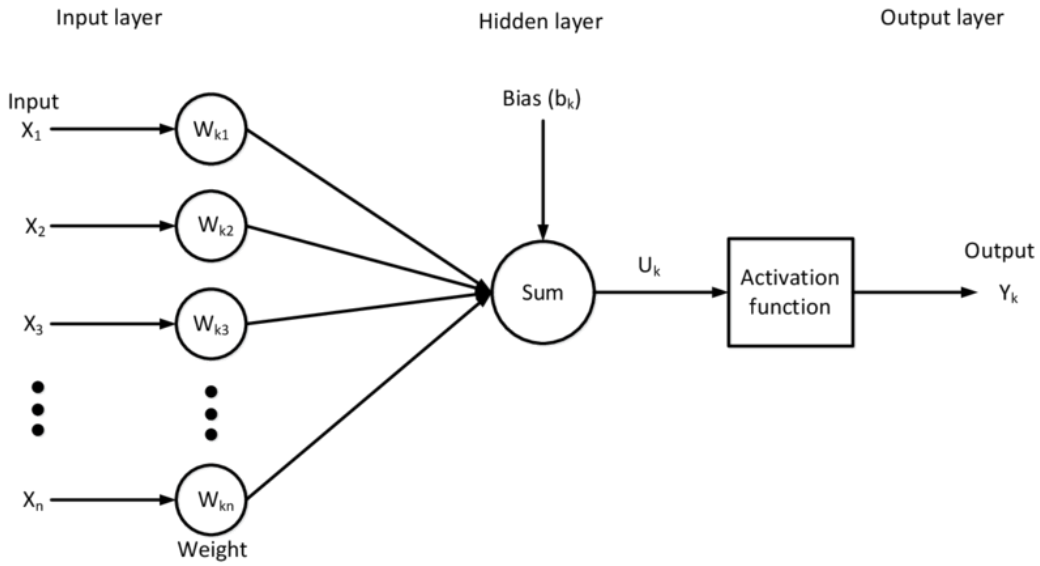
Χρήσεις των μη εποπτευόμενων μεθόδων μάθησης γίνονται στη βιοπληροφορική για την ανάλυση αλληλουχιών και γενετική ομαδοποίηση, στην εξόρυξη δεδομένων (data mining) για εξόρυξη ακολουθιών και προτύπων, στην ιατρική απεικόνιση και στην υπολογιστική όραση για την αναγνώριση αντικειμένων.

2.2.3 Ενισχυμένη Μάθηση

Στην ενισχυμένη μάθηση, όπως αναφέρθηκε παραπάνω, δε δίνεται στο δίκτυο η επιθυμητή έξοδος, αλλά αν αυτή είναι σωστή ή λάθος. Χρησιμοποιείται η μέθοδος της επιβράβευσης σε κάθε σωστή έξοδο και μία ποινή σε κάθε λανθασμένη έξοδο. Η μάθηση αυτή χρησιμοποιείται στα προβλήματα βελτιστοποίησης ελέγχου, στην αναγνώριση δηλαδή των βέλτιστων ενεργειών σε κάθε κατάσταση του συστήματος, τα οποία μπορεί να έχουν έναν πολύ μεγάλο αριθμός από αυτές (>>1000). Παραδείγματα τέτοιων συστημάτων είναι τα βιομηχανικά ρομπότ και οι προσομοιώσεις στην επιστήμη, βιομηχανία και αλλού.

2.2.4 Μαθηματικό Υπόβαθρο

Στην Εικόνα 8 φαίνεται ακόμα ένα πρότυπο ΤΝΔ με εισόδους, βάρη, κρυμμένο επίπεδο και εξόδους. Οι είσοδοι συμβολίζονται με x_i, τα βάρη με w_i και οι έξοδοι με y_i και i ≥0. Η είσοδος bias (πόλωση) παίρνει τιμές {-1|1} και χρησιμοποιείται για τον προσδιορισμό της ενεργοποίησης του νευρώνα. Κάθε νευρώνας έχει έναν όρο bias, ο οποίος αλλάζει την τιμή του μαζί με τα υπόλοιπα βάρη. Οι είσοδοι πολλαπλασιασμένοι με τα βάρη και προστιθέμενοι μεταξύ τους, είναι η είσοδος για τη λεγόμενη συνάρτηση ενεργοποίησης (activation function).



Εικόνα 8. Τεχνητό Νευρωνικό Δίκτυο (Μαθηματικό Πρότυπο)

Η συνάρτηση ενεργοποίησης λαμβάνει ως είσοδο τη συνάρτηση $\boldsymbol{u = \sum_{j=1}^{n}(x_i w_i) + b_i, n > 0}$. Για τη συνάρτηση ενεργοποίησης, συνήθως χρησιμοποιούνται η γραμμική συνάρτηση, η σιγμοειδής συνάρτηση, η υπερβολική εφαπτομένη η συνάρτηση ReLu (Rectified Linear Units) και άλλες, όπως φαίνεται και στην Εικόνα 9.

Name	Plot	Equation	Derivative
Identity		$f(x) = x$	$f'(x) = 1$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1+e^{-x}}$	$f'(x) = f(x)(1-f(x))$
TanH		$f(x) = \tanh(x) = \frac{2}{1+e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2+1}$
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Parameteric Rectified Linear Unit (PReLU) [2]		$f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Exponential Linear Unit (ELU) [3]		$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
SoftPlus		$f(x) = \log_e(1 + e^x)$	$f'(x) = \frac{1}{1+e^{-x}}$

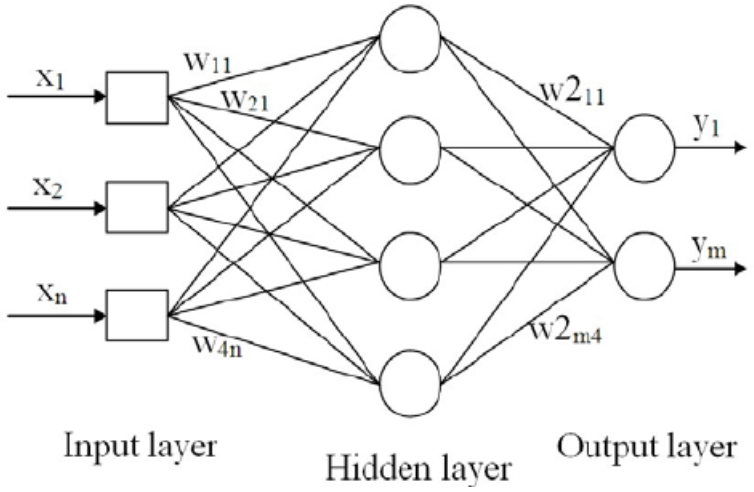
Εικόνα 9. Συναρτήσεις Ενεργοποίησης

Για ένα πολυστρωματικό ΤΝΔ, όπως αυτό της Εικόνας 10, χρησιμοποιούνται οι εξής όροι εκφρασμένοι σε πίνακες:

Διάνυσμα εισόδου: $\vec{\boldsymbol{x}} = (\boldsymbol{x_{11}}, \boldsymbol{x_{12}}, \dots, \boldsymbol{x_{1n}},)^{\boldsymbol{T}}$

Έξοδος ΤΝΔ: $\vec{\boldsymbol{y}} = (\boldsymbol{y_{11}}, \boldsymbol{y_{12}}, \dots, \boldsymbol{y_{1m}},)^{\boldsymbol{T}}$, με **n, m** όχι απαραίτητα ίσα.

Συνεπώς, οι συναρτήσεις που εφαρμόζονται επί του ΤΝΔ είναι $\boldsymbol{u_{ij}} = \sum_{\boldsymbol{j=1}}^{\boldsymbol{n}} \left(\boldsymbol{w_{ij}} \boldsymbol{x_{ij}}\right) + \boldsymbol{b_{ij}}$, i = αριθμός των επιπέδων, j = αριθμός των νευρώνων/επίπεδο $\boldsymbol{y_{ij}} = \boldsymbol{f}(\boldsymbol{u_{ij}})$, f(u) = χρησιμοποιούμενη συνάρτηση ενεργοποίησης και $\begin{bmatrix} \boldsymbol{u_{11}} \\ \vdots \\ \boldsymbol{u_{1n}} \end{bmatrix} = \begin{pmatrix} \boldsymbol{w_{11}} & \dots & \boldsymbol{w_{1n}} \\ \vdots & \ddots & \vdots \\ \boldsymbol{w_{n1}} & \dots & \boldsymbol{w_{nn}} \end{pmatrix} \begin{pmatrix} \boldsymbol{x_{11}} \\ \vdots \\ \boldsymbol{x_{1n}} \end{pmatrix} + \begin{pmatrix} \boldsymbol{b_{11}} \\ \vdots \\ \boldsymbol{b_{1n}} \end{pmatrix}, \begin{bmatrix} \boldsymbol{y_{11}} \\ \vdots \\ \boldsymbol{y_{1n}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f_{11}}(\boldsymbol{u_{11}}) \\ \vdots \\ \boldsymbol{f_{1n}}(\boldsymbol{u_{1n}}) \end{bmatrix}$



Εικόνα 10. Multilayer Perceptron (MLP)

Για τη διόρθωση των βαρών θα πρέπει να συγκριθεί η έξοδος $\vec{y}$ με την επιθυμητή έξοδο. Αυτή η διαφορά είναι το σφάλμα, το οποίο πρέπει να ελαχιστοποιηθεί. Όμως, το σφάλμα προέρχεται φαινομενικά μόνο από τις εξόδους, ενώ δεν υπάρχει κάποια ένδειξη για τα σφάλματα που προκαλούνται από τα κρυφά επίπεδα. Στην παράγραφο αυτή περιγράφεται η διόρθωση του σφάλματος έως ότου τα βάρη οριστικοποιηθούν σε συγκεκριμένες τιμές, ώστε το ΤΝΔ να θεωρείται εκπαιδευμένο. **Εποχή (epoch)** ονομάζεται ένας κύκλος εκπαίδευσης κατά τον οποίο έχουν χρησιμοποιηθεί όλα τα παραδείγματα διαδοχικά, μία φορά για την ενημέρωση των βαρών. Σημειώνεται ότι, επειδή μπορεί ο αλγόριθμος μπορεί να «τρέχει» ατέρμονα, θα πρέπει να καθοριστεί ένας μέγιστος αριθμός εποχών, πέρα από τον οποίο, ο αλγόριθμος τερματίζει.

Έστω, ένα ζευγάρι δεδομένων εκπαίδευσης (x(k),y(k)), όπου **x** είναι η είσοδος στον νευρώνα, **y** είναι η επιθυμητή έξοδος του νευρώνα και **k** η τρέχουσα εποχή. Δηλώνεται ένας

θετικός πραγματικός αριθμός **α (α > 0)** που ονομάζεται **ρυθμός μάθησης** και καθορίζει το πόσο μεγάλες θα είναι οι μεταβολές των τιμών των βαρών. Σε κάθε εποχή, εκτελούνται τα εξής βήματα:

a. Για το παραπάνω ζευγάρι, υπολογίζεται η έξοδος του νευρώνα **o(k) = g(w^T(k)x(k))**
b. Ενημέρωση των βαρών του νευρώνα με τον τύπο **w(k+1) = w(k) + a[y(k)-o(k)]**
c. Αυξάνεται κατά μία η εποχή: **k = k +1**

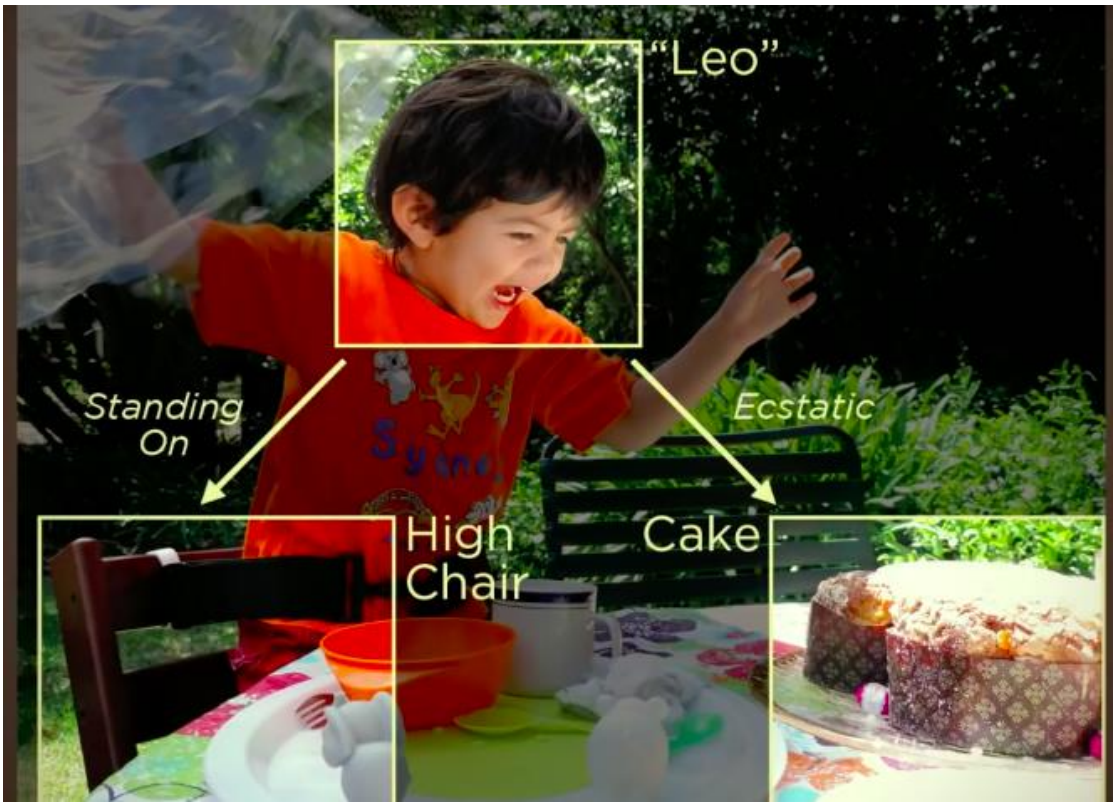
Αποδεικνύεται ότι αν το σύνολο εκπαίδευσης είναι γραμμικά διαχωρίσιμο και χρησιμοποιηθούν όλα τα παραδείγματα, τότε ο αλγόριθμος μετά από πεπερασμένα βήματα (εποχές) θα καταλήξει σε τιμές βαρών, τα οποία θα ταξινομούν σωστά τα παραδείγματα εκπαίδευσης, ανεξάρτητα από τη τιμή του ρυθμού μάθησης.

Κεφάλαιο 3

Θεωρία Συνελικτικών Νευρωνικών Δικτύων

3.1 Ο εγκέφαλος ως πρότυπο των ΣΝΔ

Πριν ορίσουμε τα συνελικτικά νευρωνικά δίκτυα (Convolutional Neural Networks – CNN), θα αναλύσουμε πως λειτουργεί ο ανθρώπινος εγκέφαλος, καθώς αποτέλεσε την έμπνευση για τη δημιουργία των δικτύων αυτών. Ο άνθρωπος για να μπορέσει να καταλάβει το τι συμβαίνει γύρω του, χρησιμοποιεί γνώση του παρελθόντος. Βασίζεται δηλαδή, σε αυτά που έχει ήδη μάθει, κατηγοριοποιεί και «βάζει» ετικέτες στα αντικείμενα που βλέπει. Οι ενέργειες αυτές γίνονται ασυναίσθητα, χωρίς ο παρατηρητής να δίνει σημασία στο πως γίνεται όλη αυτή η διαδικασία. Στο παράδειγμα της Εικόνας 11, φαίνεται ένα χαρούμενο παιδί πάνω σε μία καρέκλα, μπροστά σε ένα τραπέζι ή το παιδί ουρλιάζει, καθώς βλέπει το κέικ πάνω στο τραπέζι. Ο εγκέφαλος, αμέσως αναγνωρίζει αντικείμενα, εκφράσεις και συμπεριφορές βασιζόμενος σε ετικέτες των αντικειμένων που βλέπει. Πως, όμως, φτάνει στη διαχείριση αυτή του περιβάλλοντός του;

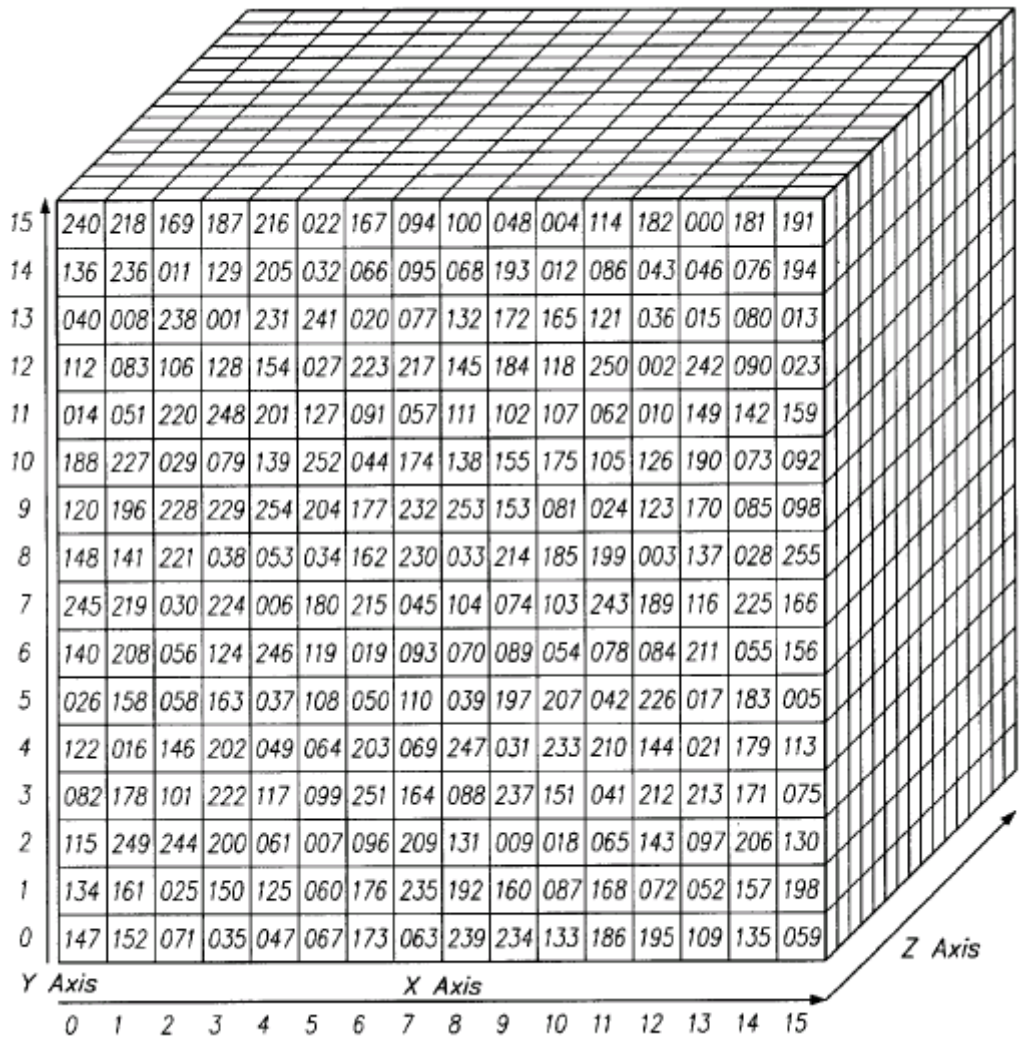


Εικόνα 11. Αναγνώριση Αντικειμένων από τον Εγκέφαλο

Το φως μεταφέρει την εικόνα μέσω των ματιών και συγκεκριμένα μέσω του οπτικού νεύρου στον πρωτεύων οπτικό φλοιό. Αυτός είναι υπεύθυνος για την κατανόηση της εικόνας και των αντικειμένων της. Αυτό που δε γίνεται κατανοητό από τον άνθρωπο είναι το τεράστιο, πολύπλοκο και ιεραρχικό δίκτυο των βιολογικών νευρώνων που δίνουν την ικανότητα να θυμόμαστε και να κατηγοριοποιούμε τα αντικείμενα. Με τη μάθηση από την εκπαίδευση, ο άνθρωπος μαθαίνει να ξεχωρίζει τα σχήματα, να διαχωρίζει τα χρώματα και τα χαρακτηριστικά διαφόρων αντικειμένων και έτσι μπορεί να γενικεύσει, βασιζόμενος σε προηγούμενη εμπειρία. Αυτή η γενίκευση του επιτρέπει να ξεχωρίζει το βιβλίο από το πιάτο, τον αετό από το περιστέρι, την καρέκλα από το τραπέζι. Με παρόμοιο, λοιπόν, τρόπο που ένα παιδί μαθαίνει, το ΣΝΔ μαθαίνει να ξεχωρίζει τα αντικείμενα.

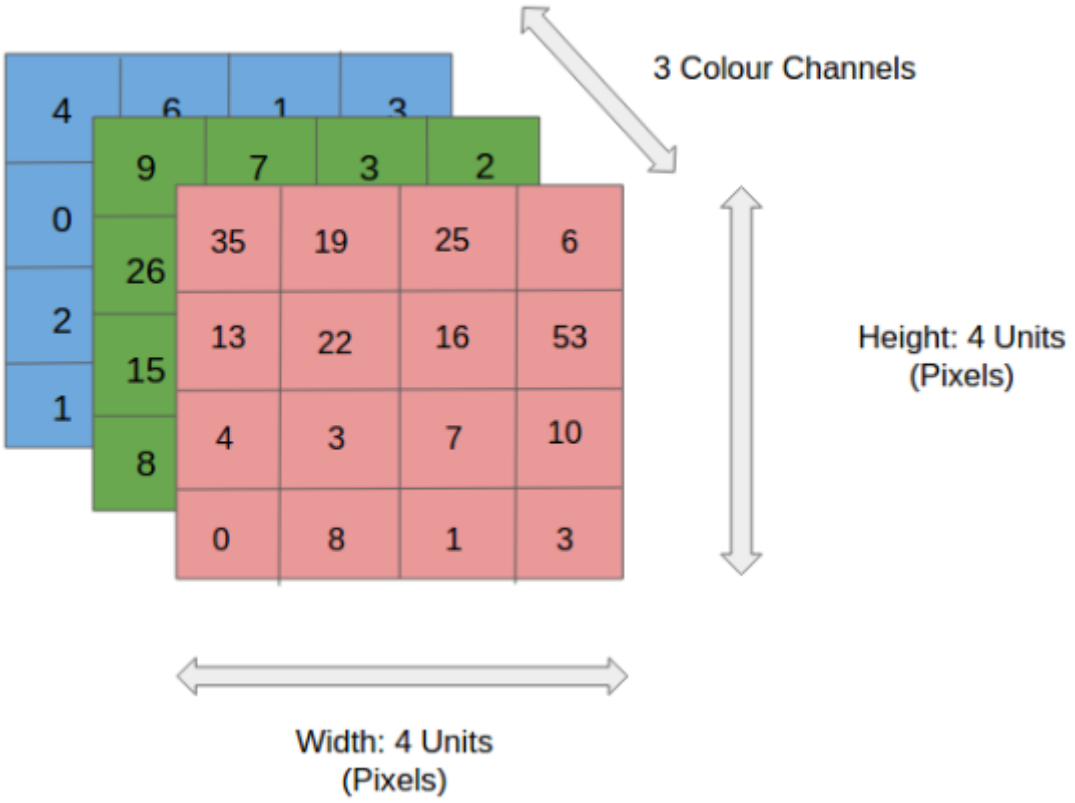
3.2 Εισαγωγή στα Συνελικτικά Δίκτυα

Τα ΣΝΔ[8] είναι μία τεχνολογία η οποία επιτρέπει την υπολογιστική όραση (Computer Vision). Επειδή οι υπολογιστές βασίζονται στον ηλεκτρισμό, αυτό που μεταφράζουν οι υπολογιστές, είναι ο δυαδικός κώδικας. Έτσι, μία εικόνα μπορεί να παρασταθεί ως πίνακας αριθμών, γνωστοί ως pixels. Ένα παράδειγμα φαίνεται στην Εικόνα 11.



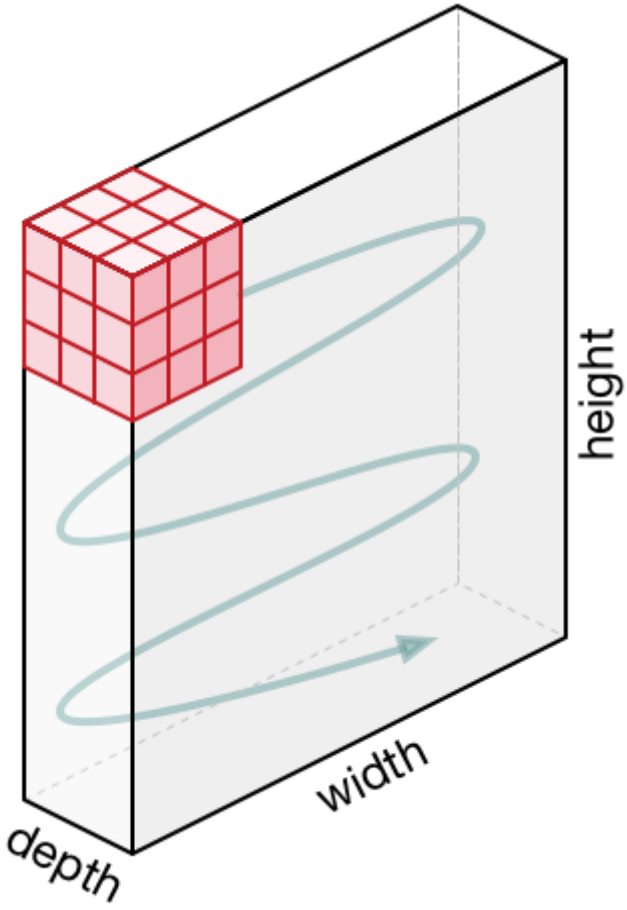
Εικόνα 12. Ανάλυση Εικόνας σε Pixels

Δημιουργείται έτσι ένας πίνακας ΥxΠx3, όπου Υ είναι το ύψος της εικόνας σε pixel, Π το πλάτος της εικόνας σε pixels και 3 είναι ο αριθμός των καναλιών χρωμάτων (Red – Green – Blue – Εικόνα 12). Κάθε επίπεδο του ΣΝΔ είναι και ένα επίπεδο επεξεργασίας από το οποίο περνάει η εικόνα εισόδου. Αυτό το επίπεδο μπορεί να είναι ένα συνελικτικό επίπεδο (convolutional layer), ένα συγκεντρωτικό επίπεδο (pooling layer), ένα επίπεδο κανονικοποίησης (normalization layer), ένα πλήρες συνδεδεμένο επίπεδο (fully connected layer) ή ένα επίπεδο απωλειών (loss layer).

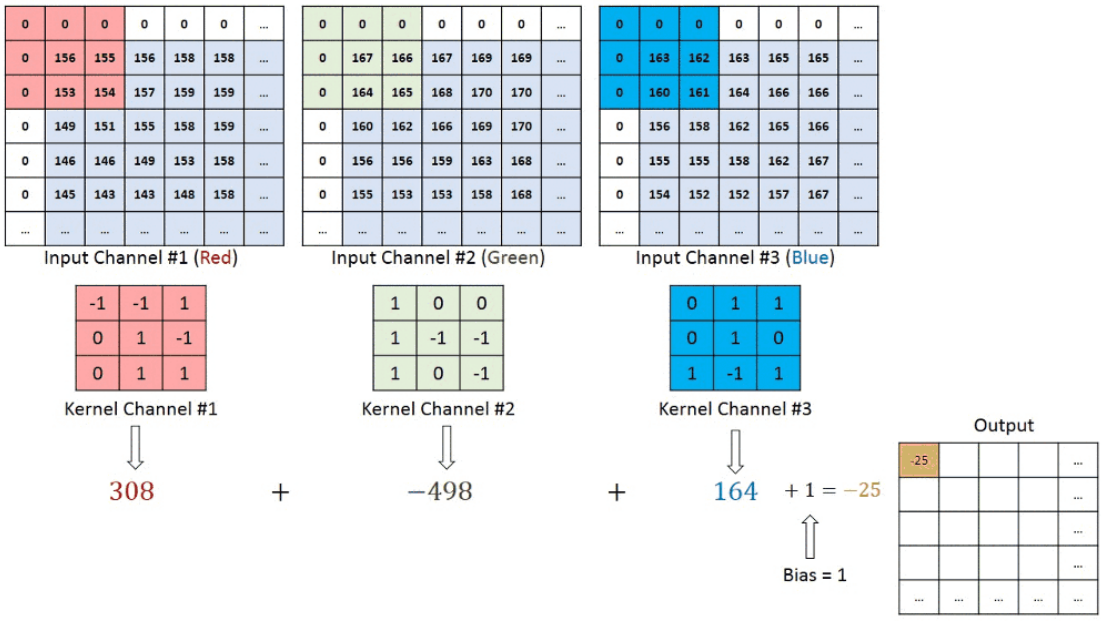


Εικόνα 13. Τρία Κανάλια Χρωμάτων

Σκοπός του συνελικτικού επιπέδου είναι η εξαγωγή χαρακτηριστικών της εικόνας που δίνεται ως είσοδος. Για να γίνει αυτό χρησιμοποιούνται αριθμητικοί πίνακες που ονομάζονται φίλτρα (ή πυρήνες-kernels). Τα φίλτρα συνελίσσονται με την εικόνα εισόδου κατά ύψος και κατά πλάτος δίνοντας στην έξοδο διάφορα χαρακτηριστικά, όπως κάθετες και οριζόντιες γραμμές. Η συνέλιξη αυτή είναι το εσωτερικό γινόμενο του φίλτρου με του ίδιου μεγέθους κομμάτι της εικόνας εισόδου. Οι πίνακες που δημιουργούνται ονομάζονται πίνακες χαρακτηριστικών (feature maps). Επειδή οι εικόνες εισάγονται σε τρισδιάστατη μορφή, η έξοδος θα είναι και αυτή σε τρισδιάστατη μορφή. Η «διαδρομή» των πράξεων φαίνονται στην Εικόνα 13 και ένα παράδειγμα πράξεων στην Εικόνα 14. Στον Πίνακα 3, παρουσιάζονται μερικά παραδείγματα φίλτρων που χρησιμοποιούνται.



Εικόνα 14. Κίνηση του Φίλτρου



Εικόνα 15. Συνέλιξη Εικόνας ΥxΠx3 με Φίλτρο 3x3x3

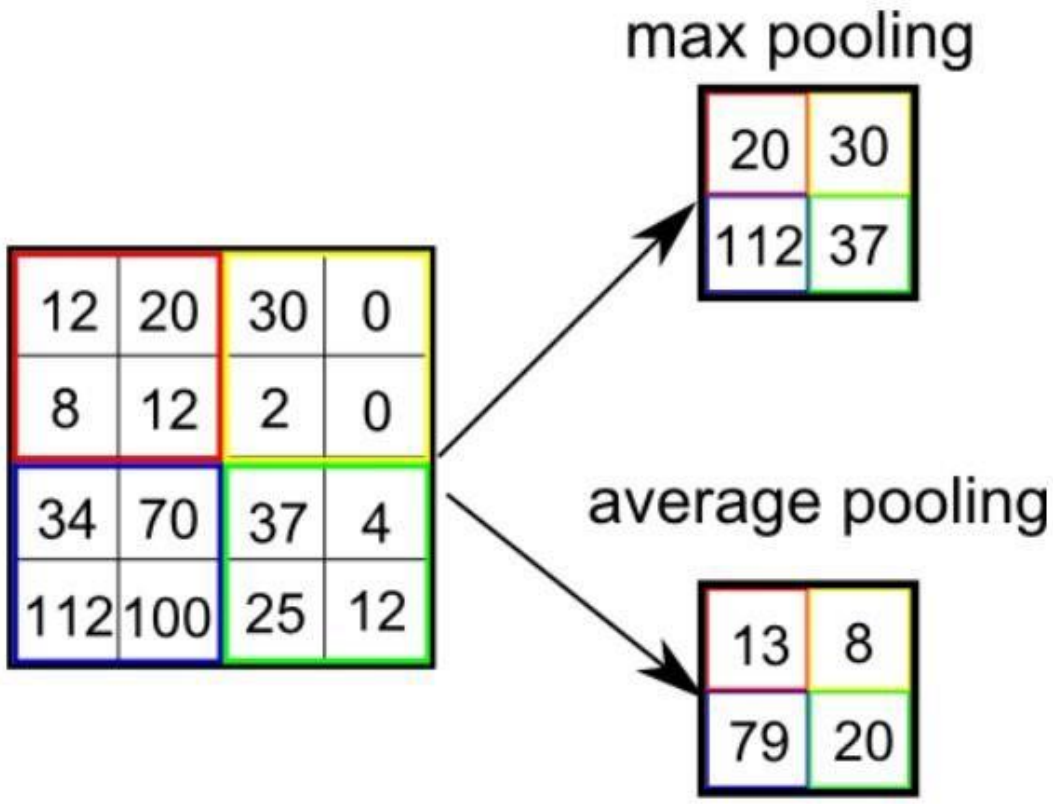
Operation	Filter	Convolved Image
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur (approximation)	$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	

Πίνακας 3. Φίλτρα ΣΝΔ

3.3 Αρχιτεκτονική και Παραδείγματα ΣΝΔ

Η εικόνα εισόδου περνά από διάφορα επίπεδα επεξεργασίας μέχρι να εξαχθεί το τελικό αποτέλεσμα. Το συνελικτικό επίπεδο αναλύθηκε στην προηγούμενη παράγραφο. Το συγκεντρωτικό επίπεδο (pooling layer) συνήθως εισάγεται μεταξύ διαδοχικών συνελικτικών επιπέδων. Βασική επιδίωξη είναι να μειωθεί το χωρικό μέγεθος της εισόδου και κατ' επέκταση η υπολογιστική ισχύς που χρειάζεται για την επεξεργασία της εικόνας. Παρά τη μείωση του μεγέθους τα βασικά χαρακτηριστικά και πληροφορίες της εισόδου διατηρούνται. Όπως και στο συνελικτικό επίπεδο, χρησιμοποιείται ένα σταθερό μέγεθος πίνακα, για παράδειγμα πίνακας 2x2, με τον οποίο μπορούν να χρησιμοποιηθούν οι μέθοδοι του μεγαλύτερου στοιχείου (max pooling), της μέσης τιμής των στοιχείων (average pooling) ή

του αθροίσματος των στοιχείων του πίνακα. Ένα παράδειγμα των δύο πρώτων μεθόδων, φαίνεται στην Εικόνα 16.

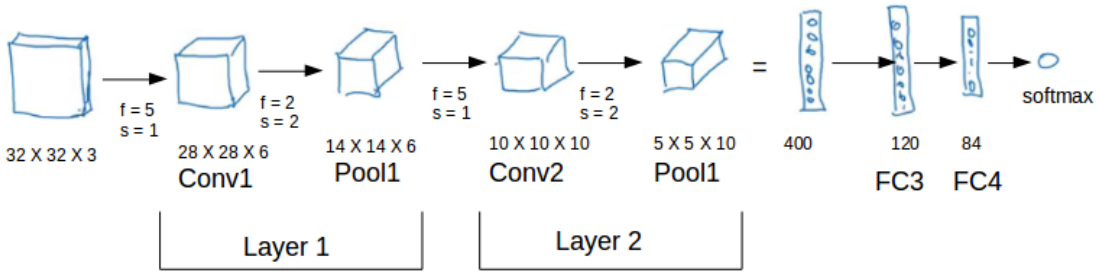


Εικόνα 16. Μέθοδοι Συγκεντρωτικού Επιπέδου

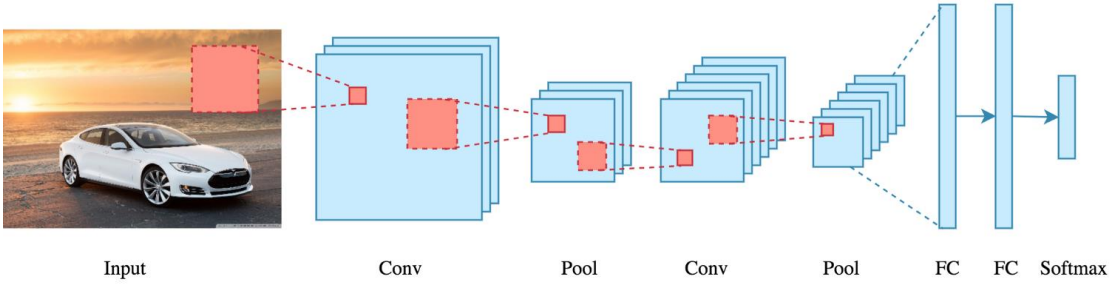
Έχει αποδειχθεί ότι η μέθοδος του μέγιστου στοιχείου είναι περισσότερο αποτελεσματική. Το πλήθος τέτοιων επιπέδων μπορεί να αυξηθεί για τον προσδιορισμό περισσότερων λεπτομερειών, αλλά θα έχει ως απαίτηση αυξημένη υπολογιστική ισχύ.

Το πλήρες συνδεδεμένο επίπεδο (Fully Connected Layer) είναι ένα επίπεδο όπου κάθε νευρώνας του προηγούμενου επιπέδου συνδέεται με όλους τους νευρώνες του επόμενου επιπέδου. Στην έξοδο του χρησιμοποιείται μία συνάρτηση ενεργοποίησης (softmax). Το επίπεδο αυτό χρησιμοποιείται για να κατηγοριοποιήσει τις εξόδους από τα προηγούμενα αναφερθέντα επίπεδα σε κλάσεις και πιθανώς με πιθανότητες εμφάνισης, με βάση τα δεδομένα εκπαίδευσης. Τελικά, φαίνεται ότι το ΣΝΔ εκτελεί δύο εργασίες. Την εξαγωγή χαρακτηριστικών της εικόνας και την κατηγοριοποίηση αυτής. Γενικά, όσο περισσότερα συνελικτικά βήματα χρησιμοποιούνται, τόσο περισσότερα πολύπλοκα χαρακτηριστικά μπορεί να αναγνωρίζει το δίκτυο που εκπαιδεύεται.

Κάποια απλά παραδείγματα παρουσιάζονται στις Εικόνες 17 και 18.



Εικόνα 17. Παράδειγμα Συνελικτικού Δικτύου



Εικόνα 18. Παράδειγμα Συνελικτικού Δικτύου

Κεφάλαιο 4

Υπολογιστική Όραση (Computer Vision)

4.1 Εισαγωγή

Η μηχανική όραση[9] είναι ένα διεπιστημονικό πεδίο στο οποίο γίνεται προσπάθεια οι υπολογιστές να αποκτήσουν τη δυνατότητα κατανόησης αντικειμένων σε ψηφιακές εικόνες ή βίντεο. Επιχειρείται, λοιπόν, η αλγοριθμική αναπαραγωγή της αίσθησης της όρασης στο μέγεθος του ανθρώπινου επιπέδου, τουλάχιστον, σε μηχανικά συστήματα. Η εισαγωγή που περιγράφηκε στα προηγούμενα κεφάλαια έθεσε τις βάσεις για την επεξήγηση της μηχανικής όρασης, των τεχνικών που εφαρμόζονται και τις δυνατότητες που προσφέρει η ανάπτυξη της τεχνολογίας αυτής.

Η μηχανική όραση έχει πολύ μεγάλη σημασία σε ένα ευρύ φάσμα του ανθρώπινου πολιτισμού. Χρησιμοποιείται στην ψυχαγωγία όπως στις κονσόλες παιχνιδιών, στην ασφάλεια, με την αναγνώριση επικίνδυνων αντικειμένων ή συμπεριφορών και στην πρόληψη ατυχημάτων στο οδικό δίκτυο ή την ανάπτυξη των αυτόνομων οχημάτων. Έχει μεγάλη χρησιμότητα και στην ιατρική, όπου οι ακτινογραφίες και τομογραφίες αναλύονται σε βάθος και με λεπτομέρεια που πιθανόν δεν μπορεί να αντιληφθεί το ανθρώπινο μάτι, οδηγώντας στην έγκαιρη διάγνωση επιβλαβών ασθενειών. Ένας ακόμα τομέας είναι η βιομηχανία. Η μηχανική μάθηση χρησιμοποιείται στην ανίχνευση βεβλαμμένων αντικειμένων κατά τη διαδικασία παραγωγής τους, στην βαθμονόμηση των βιομηχανικών ρομπότ και στην απομάκρυνση ανεπιθύμητων ουσιών και υλικών από τα τρόφιμα της γεωργικής γραμμής παραγωγής. Τέλος, έχει μεγάλη εφαρμογή και στο στρατιωτικό τομέα με τα συστήματα με επανδρωμένων αεροσκαφών, για την αναγνώριση αγνώστου εδάφους, εχθρικών οχημάτων και προσωπικού και άλλα στοιχεία εχθρικού ενδιαφέροντος.

Η ανάπτυξη της τεχνολογίας αυτής συνεχώς εξελίσσεται, καθώς βασίζεται τόσο στο υλικό όσο και στο λογισμικό των συστημάτων που χρησιμοποιούνται. Συνεχής έρευνα γίνεται τόσο στην ανάπτυξη καμερών μεγαλύτερης ανάλυσης, τη χρησιμοποίηση ισχυρότερου

υλικού για την ταχύτερη υλοποίηση του λογισμικού αλλά και βελτίωσης των αλγορίθμων και μεθόδων αναγνώρισης των αντικειμένων ή ανάπτυξη νέων και πιο αποδοτικών.

4.2 Χρωματισμός

Το χρώμα είναι το αποτέλεσμα της αλληλεπίδρασης του φυσικού φωτός στο περιβάλλον με το οπτικό μας σύστημα. Μια ψυχολογική ιδιότητα των εμπειριών μας όταν βλέπουμε τα αντικείμενα και τα φώτα, όχι μια φυσική ιδιότητα αυτών των αντικειμένων ή φώτων[10]. Το φως αποτελείται από κύματα διαφορετικών μηκών. Το οπτικό φως κυμαίνεται από 400nm έως 700nm και οι άνθρωποι είναι πιο ευαίσθητοι στο φως με μήκη κύματος στη μέση αυτού του φάσματος. Οι άνθρωποι βλέπουν μόνο ορατό φως επειδή ο Ήλιος εκπέμπει κίτρινο φως περισσότερο από οποιοδήποτε άλλο χρώμα και λόγω της θερμοκρασίας του. Οποιαδήποτε πηγή φωτός μπορεί να περιγραφεί πλήρως φυσικά από το φάσμα του (για παράδειγμα την ποσότητα ενέργειας που εκπέμπεται, ανά μονάδα χρόνου, σε κάθε μήκος κύματος 400-700nm). Οι επιφάνειες έχουν φάσματα ανάκλασης: το ανακλώμενο φως επικεντρώνεται σε μια συγκεκριμένη πλευρά του φάσματος του ορατού φωτός. Για παράδειγμα, οι μπανάνες αντικατοπτρίζουν κυρίως το κίτρινο φως, και οι ντομάτες αντανακλούν σχεδόν το κόκκινο φως.

Η ισορροπία λευκού είναι η επεξεργασία της ρύθμισης των δεδομένων εικόνας που λαμβάνονται από τους αισθητήρες ώστε να αποδίδουν σωστά τα ουδέτερα χρώματα (λευκό, γκρι, κ.λπ.). Αυτή η ρύθμιση εκτελείται αυτόματα από ψηφιακές κάμερες. Οι μη διορθωμένες εικόνες έχουν ένα αφύσικο χρώμα, όπως φαίνεται στην Εικόνα 19 για τους παρακάτω λόγους, κάνοντας τη λευκή εξισορρόπηση πολύ σημαντική:

a. Οι αισθητήρες στις κάμερες είναι πολύ διαφορετικοί από αυτούς που υπάρχουν στα ανθρώπινα μάτια.

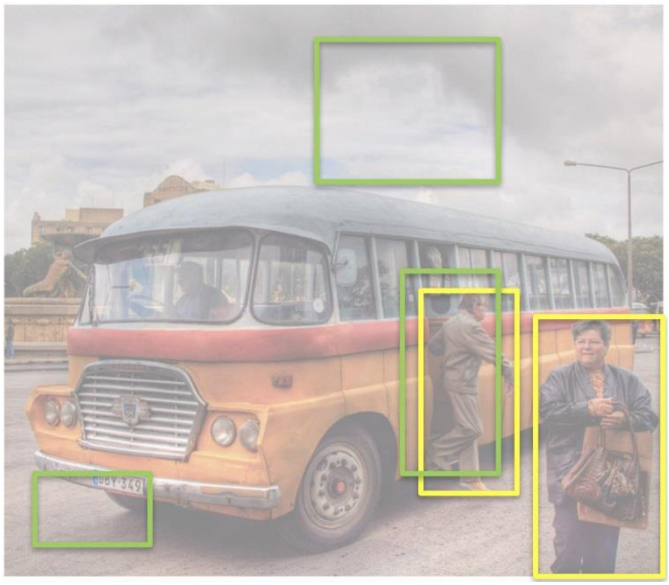
b. Τα διαφορετικά μέσα προβολής αποδίδουν διαφορετικά τις εικόνες.

c. Οι συνθήκες προβολής κατά τη λήψη της εικόνας συνήθως διαφέρουν από τις συνθήκες προβολής εικόνων με το ανθρώπινο μάτι.

Εικόνα 19. Δύο Εικόνες με Διαφορετικές Ισορροπίες Άσπρου Χρώματος

4.3 Αξιολόγηση Αναγνώρισης Αντικειμένων

Κατά την αξιολόγηση ενός αλγορίθμου ανίχνευσης αντικειμένων, θέλουμε να συγκρίνουμε τις προβλέψεις (έξοδοι του αλγόριθμου) με την επιθυμητή έξοδο[11]. Η επιθυμητή έξοδος παρέχεται από τους ανθρώπους που ταξινομούν και εντοπίζουν αντικείμενα στις εικόνες. Ένα παράδειγμα φαίνεται στην Εικόνα 20, όπου τα ορθογώνια με κίτρινο χρώμα είναι η επιθυμητή έξοδος και τα ορθογώνια με πράσινο χρώμα είναι οι προβλέψεις του αλγορίθμου.



Εικόνα 20. Χρωματισμός Πρόβλεψης & Επιθυμητής Εξόδου

Όταν συγκρίνονται οι προβλέψεις με την επιθυμητή έξοδο, υπάρχουν τέσσερις διαφορετικές δυνατότητες, όπως φαίνεται στον Πίνακα :

a. True Positive (TP)

Τα TP είναι αντικείμενα που εντοπίζουν τόσο ο αλγόριθμος (πρόβλεψη) όσο και ο άνθρωπος (επιθυμητή έξοδος). Οι προβλέψεις θεωρούνται αληθινά θετικές (TP) όταν η αλληλοεπικάλυψη μεταξύ της πρόβλεψης και της επιθυμητής εξόδου είναι μεγαλύτερη από 0.5 (Εικόνα 22). Η αλληλοεπικάλυψη μεταξύ της πρόβλεψης και της επιθυμητής εξόδου ορίζεται ως η ένωση της πρόβλεψης και της επιθυμητής εξόδου. Ορισμένες φορές αναφέρονται και ως χτυπήματα (hits).

b. False Positive (FP)

Τα FP συμβαίνουν όταν ο αλγόριθμος εντοπίζει αντικείμενα, ενώ ο άνθρωπος δεν τα εντοπίζει. Δηλαδή ο αλγόριθμος δε βρίσκει τα σωστά αντικείμενα, ενώ θεωρεί ότι έχουν βρεθεί, όπως στην Εικόνα 23. Σε αυτή την περίπτωση, η επικάλυψη της πρόβλεψης και της επιθυμητής εξόδου είναι μικρότερη από 0,5. Τα FP αναφέρονται και ως ψευδής συναγερμός (false alarms).

c. False Negative (FN)

Τα FN συμβαίνουν όταν ο αλγόριθμος δεν μπορεί να εντοπίσει τα αντικείμενα της επιθυμητής εξόδου, όπως στην Εικόνα 24. Τα FP αναφέρονται και ως αποτυχίες (misses).

d. True Negative (TN)

Τα TN συμβαίνουν όταν ούτε ο αλγόριθμός ούτε ο άνθρωπος παρήγαγε κάποιο ορθογώνιο. Ορισμένες φορές αναφέρονται και ως σωστά απορριφθέντα (true rejections).

	Class 1 Predicted	*Class 2 Predicted*
Class 1 Actual	**TP**	**FN**
Class 2 Actual	**FP**	**TN**

Πίνακας 4. TP, FP, FN. TN

Εικόνα 21. True Positive

Εικόνα 22.False Positive

Εικόνα 23. True Negative

Όπως γίνεται φανερό, στόχος είναι η μείωση των False Positive και False Negative και η αύξηση των True Positive και True Negative.

Από τα παραπάνω μέτρα εκτίμησης, εξάγεται η αποδοτικότητα του αλγορίθμου όσον αφορά στην ικανότητα πρόβλεψης. Οι μετρικές που μετρώνται αφορούν στην ακρίβεια (Accuracy), το λόγο των σωστά προβλεπόμενων αντικειμένων προς το σύνολο των αντικειμένων που εντοπίστηκαν (Precision), το λόγο λάθους (Error Rate), το ποσοστό των TP (Recall), το ποσοστό των TN (Specificity), το ποσοστό των FP (False Positive Rate) και το ποσοστό των FN (False Negative Rate). Τα παραπάνω υπολογίζονται ως εξής:

Η ακρίβεια είναι ο λόγος της επιθυμητής εξόδου προς το σύνολο ελέγχου:

$$\boldsymbol{Accuracy} = \frac{\boldsymbol{TP} + \boldsymbol{TN}}{\boldsymbol{TP} + \boldsymbol{TN} + \boldsymbol{FP} + \boldsymbol{FN}}$$

Ο λόγος των σωστά προβλεπόμενων αντικειμένων προς το σύνολο των αντικειμένων που εντοπίστηκαν είναι:

$$\boldsymbol{Precision} = \frac{\boldsymbol{TP}}{\boldsymbol{TP} + \boldsymbol{FP}}$$

Ο λόγος λάθους είναι το άθροισμα των αποτυχιών και των λάθος συναγερμών προς το σύνολο ελέγχου:

$$\boldsymbol{Error\ Rate} = \frac{\boldsymbol{FP} + \boldsymbol{FN}}{\boldsymbol{TP} + \boldsymbol{TN} + \boldsymbol{FP} + \boldsymbol{FN}}$$

Το ποσοστό των θετικών στιγμιότυπων που ταξινομούνται σωστά είναι:

$$\boldsymbol{Recall} = \frac{\boldsymbol{TP}}{\boldsymbol{TP} + \boldsymbol{FN}}$$

Το ποσοστό των αρνητικών στιγμιότυπων που ταξινομούνται σωστά είναι:

$$\boldsymbol{Specificity} = \frac{\boldsymbol{TN}}{\boldsymbol{TN} + \boldsymbol{FP}}$$

Το ποσοστό των αρνητικών στιγμιότυπων που ταξινομούνται λάθος είναι:

$$\boldsymbol{False\ Positive\ Rate} = \frac{\boldsymbol{FP}}{\boldsymbol{TN} + \boldsymbol{FP}}$$

Το ποσοστό των θετικών στιγμιότυπων που ταξινομούνται λάθος είναι:

$$\boldsymbol{False\ Negative\ Rate} = \frac{\boldsymbol{FN}}{\boldsymbol{TP} + \boldsymbol{FN}}$$

Οι μετρικές precision και recall είναι οι πιο βασικές για τη μέτρηση της απόδοσης των ταξινομητών. Αυτές οι δύο δεν είναι δυνατόν να μεγιστοποιηθούν παράλληλα. Μπορεί, όμως, να χρησιμοποιηθεί μία άλλη μετρική που ονομάζεται F1 –μετρική με τύπο

$$\boldsymbol{F1 - Score} = \frac{\boldsymbol{2 * Precision * Recall}}{\boldsymbol{Precision + Recall}}$$

που συνδυάζει τις Precision και Recall και ως στόχος να είναι η μεγιστοποίηση αυτής.

4.4 Αλγόριθμοι & Τεχνολογίες

Λόγω του μεγάλου όγκου δεδομένων εκπαίδευσης, δηλαδή ψηφιακών εικόνων, απαιτείται ανάπτυξη αλγορίθμων που έχουν ως αποτέλεσμα την ταχύτατη εκπαίδευση των δεδομένων αυτών και την ταχύτατη αναγνώριση των δεδομένων προς επεξεργασία. Οι αλγόριθμοι που θα αναφερθούν στην παράγραφο, αποσκοπούν σε αυτή την κατεύθυνση, ενώ,

όπως προαναφέρθηκε, η έρευνα για την εξεύρεση νέων και βελτιωμένων αλγορίθμων συνεχίζεται.

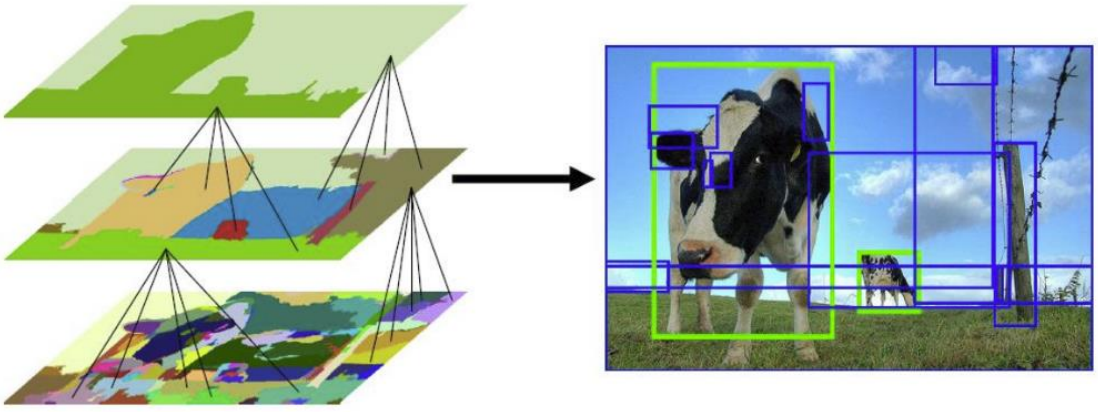
4.4.1 R - CNN

Οι χρησιμοποιούμενοι αλγόριθμοι είναι οι R - CNN, Fast R - CNN, Faster R - CNN, YOLO[12]. Ο R – CNN (Region CNN)[13] χρησιμοποιείται επιλεκτική αναζήτηση για να εξαχθούν μόνο 2000 περιοχές από την εικόνα και ονομάζονται προτάσεις περιοχής (region proposal). Επομένως, αντί να γίνει ταξινόμηση ενός τεράστιου αριθμού περιοχών, ο αλγόριθμος εργάζεται με 2000 περιοχές. Αυτές επιλέγονται χρησιμοποιώντας τον αλγόριθμο επιλεκτικής αναζήτησης ως εξής[14]:

a. Επειδή στις εικόνες υπάρχουν διαφορετικά χρώματα, υφές και περιοχές, ξεκινώντας από το χαμηλό επίπεδο προς το υψηλότερο, επιλέγονται πολύ μικρές περιοχές της εικόνας.

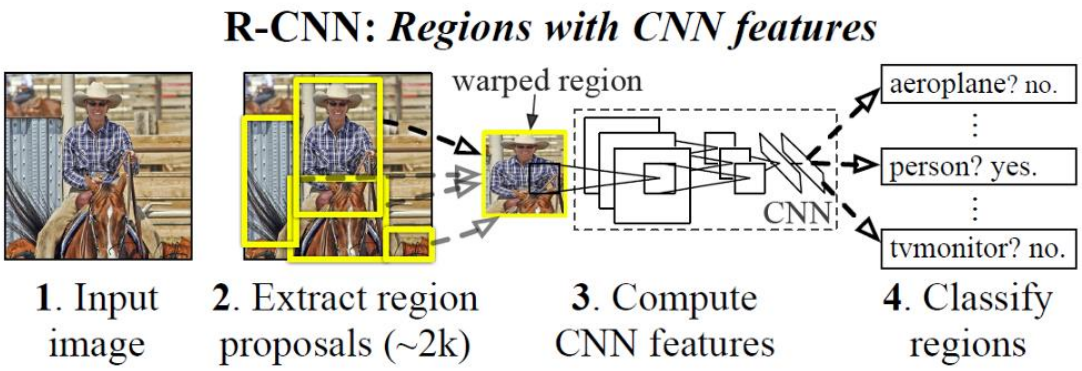
b. Με επαναληπτική διαδικασία, ενώνονται οι γειτονικές περιοχές που έχουν παρόμοιο χρώμα και υφή, δημιουργώντας μεγαλύτερες περιοχές

c. Δημιουργούνται τελικά 2000 περιοχές προς επεξεργασία.



Εικόνα 24. Αλγόριθμος Επιλεκτικής Αναζήτησης

Εν συνεχεία, όπως περιγράφηκε και ανωτέρω, οι περιοχές αυτές εισάγονται στο ΣΝΔ και τελικά ταξινομούνται στην έξοδο. Η Εικόνα 26 παρουσιάζει συνοπτικά το R – CNN.

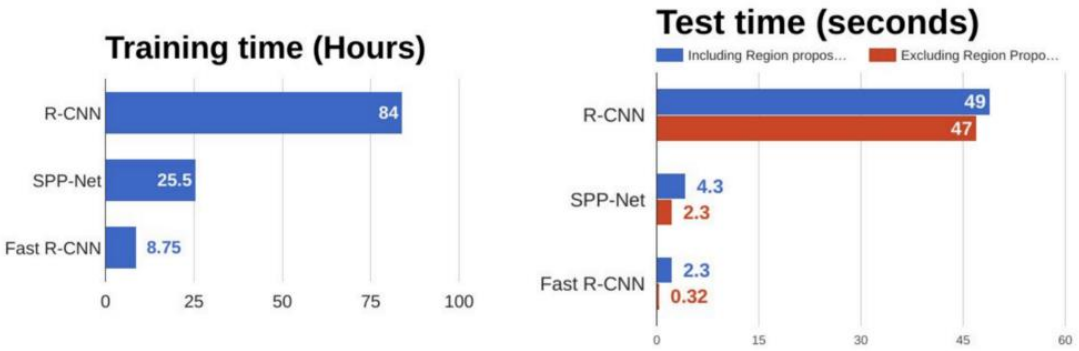


Εικόνα 25. Ο Αλγόριθμος R – CNN

Προβλήματα που σχετίζονται με τον αλγόριθμο είναι ο χρόνος εκπαίδευσης, ο οποίος είναι μεγάλος, καθώς χρειάζεται η ταξινόμηση 2000 περιοχών ανά εικόνα, δεν μπορεί να εφαρμοστεί σε πραγματικό χρόνο, αφού κάθε εικόνα χρειάζεται 47 δευτερόλεπτα για να ταξινομηθεί και τέλος, επειδή ο αλγόριθμος είναι στατικός, που σημαίνει ότι δεν γίνεται κάποια εκπαίδευση, μπορεί να οδηγηθεί σε λάθος προτάσεις περιοχών.

4.4.2 Fast R - CNN

Ο αλγόριθμος Fast R – CNN[15] διορθώνει τα προβλήματα (Εικόνα 26) με το να εισάγει την εικόνα στο ΣΝΔ ώστε να δημιουργηθεί ένας χάρτης χαρακτηριστικών (feature map), αντί να χρησιμοποιήσει ως είσοδο τις προτάσεις περιοχών. Από το χάρτη χαρακτηριστικών, εντοπίζονται οι προτάσεις περιοχών και χρησιμοποιούνται ως είσοδος σε συγκεντρωτικά επίπεδα και στη συνέχεια στο πλήρες συνδεδεμένο επίπεδο, κατά τα γνωστά. Ο Fast R-CNN είναι ταχύτερος από τον R-CNN διότι δεν τροφοδοτούνται κάθε φορά 2000 περιοχές στο ΣΝΔ, αλλά η συνέλιξη γίνεται μία φορά ανά εικόνα κατά την οποία εξάγεται ο χάρτης περιοχής.

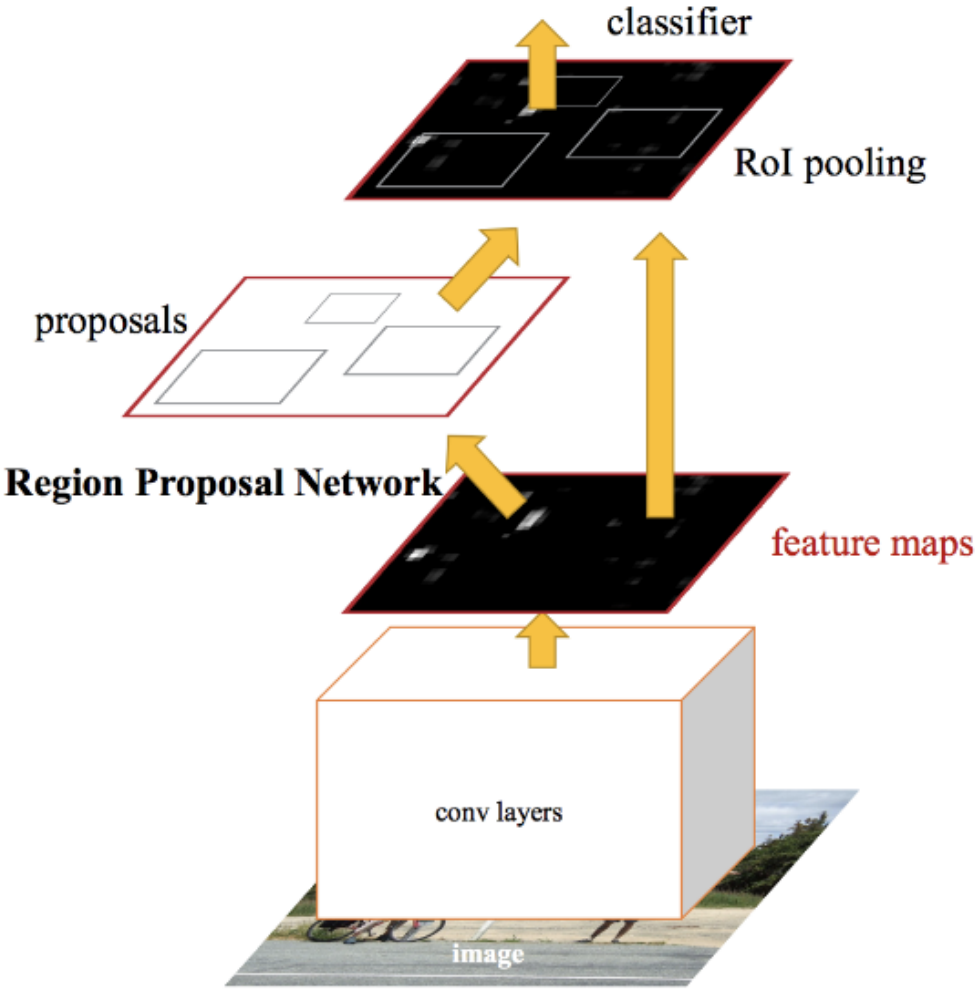


Εικόνα 26. Σύγκριση R - CNN & Fast R - CNN

4.4.3 Faster R - CNN

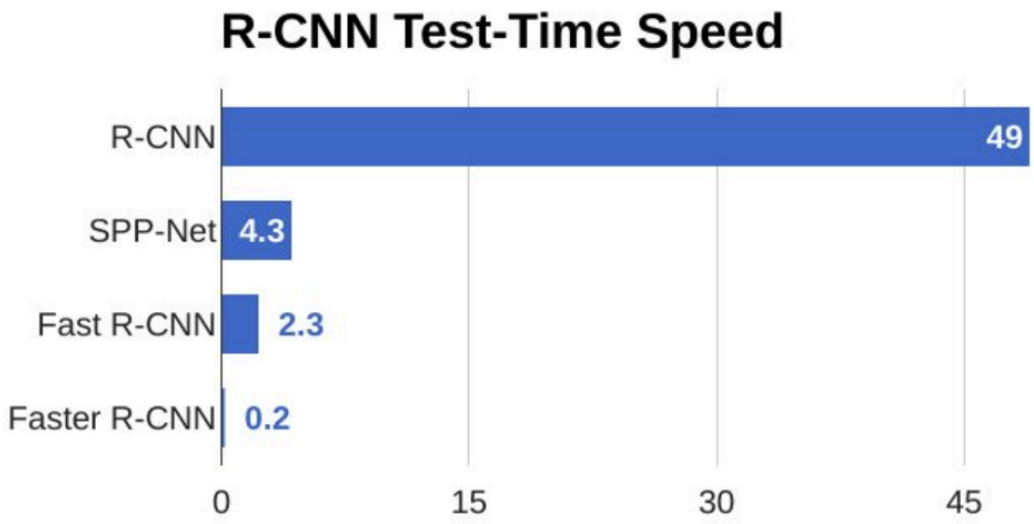
Η κύρια διαφορά είναι ότι το R-CNN εισάγει τις προτάσεις περιοχής για ανίχνευση στο CNN σε επίπεδο pixel, ενώ η Fast R-CNN εισάγει τις προτάσεις περιοχής σε επίπεδο χάρτη χαρακτηριστικών. Το Faster R-CNN[16] χρησιμοποιεί την επιλεκτική αναζήτηση. Με το να χρησιμοποιηθεί ένα μικρό συνελικτικό δίκτυο που ονομάζεται Δίκτυο Προτάσεων Περιοχών (Regional Proposal Network - RPN), η ταχύτητα αυξάνεται ακόμα περισσότερο.

Για να χειριστείτε τις παραλλαγές του λόγου διαστάσεων (aspect ratio) και της κλίμακας αντικειμένων (scale of objects), το Faster R-CNN χρησιμοποιεί τα κιβώτια αγκύρωσης (anchor boxes). Σε κάθε θέση, χρησιμοποιούνται 3 είδη κιβωτίων αγκύρωσης για κλίμακες 128x128, 256x256 και 512x512. Ομοίως, για λόγο διαστάσεων, χρησιμοποιεί τρεις αναλογίες διαστάσεων 1: 1, 2: 1 και 1: 2. Έτσι, συνολικά σε κάθε θέση, υπάρχουν 9 πλαίσια στα οποία το RPN προβλέπει την πιθανότητα να είναι ένα φόντο ή να βρίσκεται μπροστά. Στη συνέχεια, εφαρμόζεται επαναληπτικά ο ορισμός κιβωτίων οριοθέτησης για να βελτιωθούν αυτά σε κάθε θέση. Επομένως, το RPN εξάγει πλαίσια οριοθέτησης διαφόρων μεγεθών με τις αντίστοιχες πιθανότητες κάθε τάξης. Τα ποικίλα μεγέθη των πλαισίων οριοθέτησης μπορούν να γίνουν είσοδος στο υπόλοιπο δίκτυο το οποίο είναι παρόμοιο με το Fast-RCNN. Έτσι, το δίκτυο RPN πρέπει να ελέγξει εκ των προτέρων ποια τοποθεσία περιέχει αντικείμενο. Και οι αντίστοιχες θέσεις και τα κιβώτια οριοθέτησης θα περάσουν στο δίκτυο ανίχνευσης για την ανίχνευση της κλάσης αντικειμένου και την επιστροφή του πλαισίου οριοθέτησης αυτού του αντικειμένου. Στην Εικόνα 28 παρουσιάζεται ο αλγόριθμος Faster R – CNN.



Εικόνα 27. Αλγόριθμος Faster R - CNN

Όπως φαίνεται στην Εικόνες 29 και στον Πίνακα 5, ο Faster R - CNN είναι 10 φορές ταχύτερος από το Fast R – CNN. Αυτός είναι ο λόγος για τον οποίο το Faster-RCNN υπήρξε ένας από τους πιο ακριβείς αλγόριθμους ανίχνευσης αντικειμένων, καθώς, επίσης, μπορεί να χρησιμοποιηθεί για την ανίχνευση αντικειμένων σε πραγματικό χρόνο.



Εικόνα 28. Σύγκριση Faster R - CNN με άλλους Αλγόριθμους

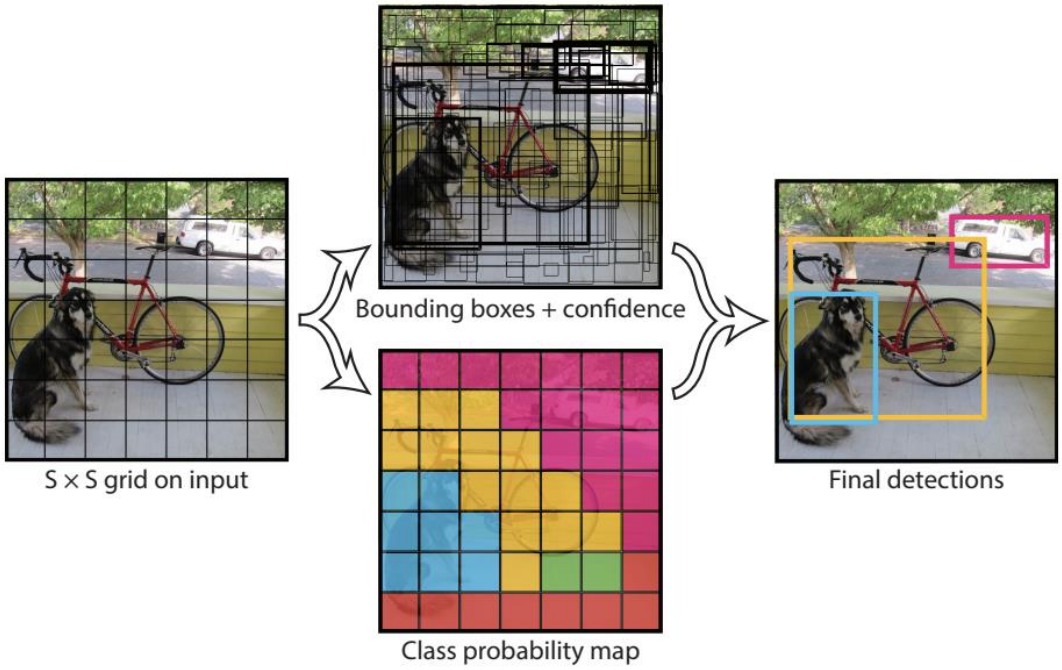
	R-CNN	Fast R-CNN	Faster R-CNN
Test Time per Image	50 Seconds	2 Seconds	0.2 Seconds
Speed Up	1x	25x	250x

Πίνακας 5. Σύγκριση Faster R - CNN με άλλους Αλγόριθμους

4.4.4 Αλγόριθμος YOLO

Ο αλγόριθμος YOLO[17] (You Only Look Once) Όλοι οι προηγούμενοι αλγόριθμοι ανίχνευσης αντικειμένων χρησιμοποιούν περιοχές για να εντοπίσουν το αντικείμενο μέσα στην εικόνα. Το δίκτυο δεν εξετάζει την πλήρη εικόνα, αλλά τμήματα της εικόνας που έχουν μεγάλες πιθανότητες να περιέχουν το αντικείμενο. Για το YOLO, η ανίχνευση είναι ένα απλό πρόβλημα παλινδρόμησης το οποίο παίρνει μια εικόνα εισόδου και μαθαίνει τις πιθανότητες των κλάσεων και τις συντεταγμένες του πλαισίου οριοθέτησης

Ο YOLO διαιρεί κάθε εικόνα σε πλέγμα S x S και κάθε πλέγμα προβλέπει N οριοθετούμενα πλαίσια και ένα βαθμό σιγουριάς. Ο βαθμός αυτός αντανακλά την ακρίβεια του πλαισίου και αν το πλαίσιο περιέχει πράγματι ένα αντικείμενο, ανεξάρτητα από την κλάση στην οποία θέλουμε να κατηγοριοποιήσουμε τα αντικείμενα. Ο YOLO προβλέπει επίσης τη βαθμολογία των κατηγοριών για κάθε πλαίσιο, για κάθε κατηγορία στη φάση της εκπαίδευσης. Επομένως, προβλέπονται συνολικά πλαίσια SxSxN. Ωστόσο, τα περισσότερα από αυτά τα πλαίσια έχουν χαμηλή βαθμολογία αξιοπιστίας. Θέτοντας μία επιθυμητή τιμή κατωφλίου, κάθε πλαίσιο με βαθμό μεγαλύτερο του κατωφλίου, επιλέγεται και χρησιμοποιείται για την ανεύρεση ενός αντικειμένου. Στην Εικόνα 28, φαίνονται τα παραπάνω.



Εικόνα 29. Αλγόριθμος YOLO

Παρατηρείται ότι κατά το χρόνο εκτέλεσης ο αλγόριθμος διατρέχει την εικόνα στο ΣΝΔ μόνο μία φορά. Ως εκ τούτου, ο YOLO είναι τάξεις μεγέθους ταχύτερος (45 καρέ ανά δευτερόλεπτο) από τους άλλους αλγόριθμους ανίχνευσης αντικειμένων. Μια άλλη βασική διαφορά είναι ότι το YOLO βλέπει την πλήρη εικόνα ταυτόχρονα σε αντίθεση με την εξέταση μόνο προτάσεων περιοχής που δημιουργήθηκαν στις προηγούμενες μεθόδους. Έτσι, αυτές οι πληροφορίες που παρέχονται από τον αλγόριθμο, συμβάλλουν στην αποφυγή ψευδών-θετικών. Ωστόσο, ένας περιορισμός για το YOLO είναι ότι προβλέπει μόνο 1 τύπο κλάσης σε ένα πλέγμα, επομένως, δεν αναγνωρίζει πολύ μικρά αντικείμενα, για παράδειγμα μπορεί να έχει δυσκολίες στην ανίχνευση ενός κοπαδιού πουλιών. Το σύστημα που θα αναπτυχθεί στην παρούσα εργασία, θα υλοποιηθεί με τον αλγόριθμο YOLO.

4.4.5 Python – TensorFlow – Keras

Σε συνδυασμό με τον παραπάνω αλγόριθμο, θα χρησιμοποιηθεί η γλώσσα προγραμματισμού **Python**. Η Python είναι μία «ώριμη» γλώσσα, περιέχει έναν μεγάλο αριθμό βιβλιοθηκών και χρησιμοποιείται κατά κόρον στη μηχανική μάθηση. Το **TensorFlow** είναι μια πλατφόρμα ανοιχτού κώδικα για μηχανική μάθηση, ανεπτυγμένο από την εταιρεία Google. Έχει ένα ολοκληρωμένο, ευέλικτο οικοσύστημα εργαλείων, βιβλιοθηκών και με την συνεισφορά προγραμματιστών εκτός εταιρείας, επιτρέπει στους ερευνητές να αναπτύξουν

περαιτέρω τη μηχανική μάθηση και στους προγραμματιστές να δημιουργήσουν εύκολα εφαρμογές.

Το TensorFlow είναι ένα σύστημα που λειτουργεί σε προγράμματα μεγάλης κλίμακας και σε ετερογενή περιβάλλοντα. Το TensorFlow χρησιμοποιεί γραφήματα ροής δεδομένων για να αντιπροσωπεύει τον υπολογισμό, την κοινή κατάσταση και τις λειτουργίες που μεταλλάζουν αυτή την κατάσταση. Χαρτογράφει τους κόμβους ενός γραφήματος ροής δεδομένων ανάμεσα σε πολλά μηχανήματα ενός συμπλέγματος (**cluster**) ή σε ένα μόνο μηχάνημα με πολλαπλές υπολογιστικές μονάδες, συμπεριλαμβανομένων CPU πολλαπλών επεξεργαστών, GPU γενικού σκοπού και προσαρμοσμένων ASIC[10] (**application –specific integrated circuit**), που ονομάζονται Tensor Processing Units (**TPU**). Αυτή η αρχιτεκτονική παρέχει ευελιξία στους προγραμματιστές εφαρμογών να δοκιμάσουν νέες βελτιστοποιήσεις και αλγόριθμους κατάρτισης. Το TensorFlow υποστηρίζει μια ποικιλία εφαρμογών, με επίκεντρο την εκπαίδευση των νευρωνικών δικτύων και τα βαθιά νευρωνικά δίκτυα.

Το Keras είναι μία βιβλιοθήκη νευρωνικών δικτύων υψηλού επιπέδου, γραμμένο στη γλώσσα Python και μπορεί να «τρέχει» πάνω από πολλές τεχνολογίες συμπεριλαμβανομένου του TensorFlow που θα χρησιμοποιηθεί στην παρούσα εργασία. Έχει τη δυνατότητα ανάπτυξης προγραμμάτων, γρήγορα και εύκολα λόγω της φιλικής διεπαφής του, των πολλών μονάδων προγραμματισμού του (**modules**) και της δυνατότητας να προστίθενται νέες βιβλιοθήκες και μονάδες προγραμματισμού. Χρησιμοποιείται τόσο στα συνελικτικά δίκτυα, όσο και στα recurrent νευρωνικά δίκτυα. Τέλος, η χρήση του δεν επιβαρύνει τους πόρους της CPU και της GPU.

Κεφάλαιο 5

Υλοποίηση Υπολογιστικής Όρασης με τον Αλγόριθμο YOLOv3

5.1 Προλογισμός

Στο προηγούμενο κεφάλαιο αναπτύχθηκε ο τρόπος που λειτουργεί ο αλγόριθμος YOLO. Στην υλοποίηση του συστήματος θα χρησιμοποιηθεί η τρίτη έκδοση του αλγορίθμου, η οποία αυξάνει σημαντικά την ταχύτητα εκπαίδευσης και αναγνώρισης, καθώς και την ακρίβεια. Καταρχήν θα πρέπει να έχουμε δύο (2) σετ από εικόνες. Ένα σετ θα περιέχει τις εικόνες στις οποίες θα εκπαιδευτεί το δίκτυο και ένα σετ στο οποίο θα κάνει την επαλήθευση. Οι εικόνες περιέχουν υλικό από τυφέκια, πιστόλια και μαχαίρια. Αυτό μπορεί να γίνει με δύο (2) τρόπους. Είτε προγραμματιστικά είτε με ένα extension προγράμματος περιήγησης. Για τους παραπάνω τρόπους χρησιμοποιείται η Google.

5.2 Απόκτηση Εικόνων

5.2.1 Προγραμματιστικός τρόπος

Δημιουργώντας ένα πρόγραμμα σε γλώσσα προγραμματισμού Python[18], θα «κατέβουν» οι φωτογραφίες μαζικά από τον ιστότοπο των google images μετά από σχετική αναζήτηση. Χρησιμοποιείται το WebDriver είναι ένα εργαλείο ανοιχτού κώδικα για αυτοματοποιημένες δοκιμές web apps σε πολλά προγράμματα περιήγησης. Παρέχει δυνατότητες πλοήγησης σε ιστοσελίδες, εισαγωγή χρηστών, εκτέλεση JavaScript και πολλά άλλα. Το ChromeDriver.exe είναι ένας αυτόνομος διακομιστής που υλοποιεί το πρωτόκολλο καλωδίου του WebDriver για το Chromium. Ο σχετικός κώδικας δίνεται παρακάτω:

```
import ast
import os
import general
import urllib.request as ulib
from selenium import webdriver
from bs4 import BeautifulSoup as Soup

chromePath = r'C:\Users\silav\Desktop\chromedriver.exe'
driver=webdriver.Chrome(chromePath)
url=general.url
directory=general.directory
failed = 0
succeded = 0

# Will get the images urls
def getURL(url):
    driver.get(url)
    desired_URLs=[]
    print('Press me now')
    a=input()
    get_pages=driver.page_source
    soup =Soup(get_pages,'lxml')
    # Gets only the url of each image
    URLs_in_line = soup.findAll('div',{'class':'rg_meta notranslate'})
    # and put them in a list
    for urls in URLs_in_line:
        current_URL = urls.text
        current_URL = ast.literal_eval(current_URL)['ou']
        desired_URLs.append(current_URL)
    #returns a list with urls
    return desired_URLs

 # Saves and renames the images on the directory with the name
xxxx.jpg, where xxxx is number
def save_images(URLs,directory):
    if not os.path.isdir(directory):
        os.mkdir(directory)
    for i,url in enumerate(URLs):
        savePath = os.path.join(directory,'{:04}.jpg'.format(i))
        global succeded
        global failed
        try:
            ulib.urlretrieve(url,savePath)
            succeded = succeded + 1
        except:
            print('I failed with ',url)
            failed=failed + 1
```

```
URLs = getURL(url)

save_images(URLs,directory)
driver.close()
print('Finished ')
print('Failed: ', failed)
print('Succeded: ', succeded)
print('Success Ration: ', (succeded/(failed+succeded))*100)
```

Ο παραπάνω κώδικας ανοίγει μία προεπιλεγμένη ιστοσελίδα με εικόνες που περιέχουν το υλικό της εργασίας και στη συνέχεια αφού ξεκινήσει(“Press me now”) κατεβάζει και αποθηκεύει τις εικόνες στον επιθυμητό φάκελο για να επεξεργαστούν περαιτέρω.

5.2.2 Με extension του προγράμματος περιήγησης

Σε αυτό το σημείο μπορεί να χρησιμοποιηθεί το extension **Fatkun Batch Download Image** από το Web Store της Google. Αυτό κάνει την ίδια εργασία με την προγραμματιστική διαδικασία που περιγράφηκε παραπάνω αλλά είναι περισσότερο φιλικό στο χρήστη, με περισσότερες επιλογές. Στην παρούσα εργασία, χρησιμοποιήθηκε ο τρόπος που περιγράφεται στο παρόν εδάφιο. Οι εικόνες που χρησιμοποιήθηκαν για την εκπαίδευση και την επαλήθευση, αποθηκεύθηκαν για την επεξεργασία τους, σε προσωρινό φάκελο με όνομα “word _ Google Search”, όπου word αντιστοιχεί στη λέξη αναζήτησης. Αυτές μετονομάστηκαν με βάση τον παρακάτω κώδικα σε γλώσσα προγραμματισμού Python για την εύκολη μεταχείρισή τους και αναγνώριση τους:

```
# renamePictures.py
import os

# Function to rename multiple files
def main():
    i = 0

    path = r'C:\Users\silav\Desktop\knives _ Google Search\\'
    path1 =r'C:\Users\silav\Desktop\knives\\'
    for filename in os.listdir(path):
        dst='{:04}.jpg'.format(i)
        src =path + filename
        dst =path1 + dst

        # rename() function will
        # rename all the files
        os.rename(src, dst)
        i += 1
```

```
# Driver Code
if __name__ == '__main__':

    # Calling main() function
    main()
```

Με τον παρακάτω κώδικα καταγράφω στο αρχείο **train.txt** τα ονόματα των εικόνων που θα χρησιμοποιηθούν στην εκπαίδευση:

```
#listfiles.py
import os

folder = 'data/temp/'
with open(data/train.txt', 'w') as f:
    with os.scandir(folder) as entries:
        for entry in entries:
            f.write("data/img/" + entry.name + "\n")
```

5.3 "Σχόλια" επί των Εικόνων

Σε αυτή την παράγραφο περιγράφεται ο «σχολιασμός» των εικόνων. Θα σημειωθούν δηλαδή τα όρια πλαισίων που περιέχουν οπλισμό, χειροκίνητα, για κάθε εικόνα. Υπάρχουν πολλά εργαλεία για να γίνει αυτό, τα οποία είναι παρόμοια μεταξύ τους στη λειτουργία και τη διεπαφή χρήστη, αλλά για την εργασία θα χρησιμοποιηθεί το **Yolo Mark** του προγραμματιστή AlexeyAB, του οποίου την έκδοση εκτελούμε.

Οι εικόνες είναι αποθηκευμένες όλες μαζί σε έναν προσωρινό φάκελο και με ονόματα ως εξής:

a. Τυφέκια με τρεις (3) θέσεις ακεραίων: xxx.jpg

b. Μαχαίρια με τέσσερις (4) θέσεις ακεραίων: xxxx.jpg

c. Πιστόλια με πέντε (5) θέσεις ακεραίων: xxxxx.jpg

Στη συνέχεια εκτελείτε ο παρακάτω κώδικας για το διαχωρισμό των εικόνων που θα εκπαιδευτούν και των εικόνων που θα επαληθευτούν:

```
# process.py
import glob, os

# Current directory
current_dir = os.path.dirname(os.path.abspath(__file__))
```

```
current_dir = '/home/darknet/data/temp_train_images/'
# Directory where the data will reside

# Percentage of images to be used for the test set
percentage_test = 10;

# Create and/or truncate train.txt and test.txt
file_train = open('/home/darknet/data/train.txt', 'w')
file_test  = open('/home/darknet/data/test.txt', 'w')

# Populate train.txt and test.txt
counter = 1
index_test = round(100 / percentage_test)
i=0
j=0

for pathAndFilename in glob.iglob(os.path.join(current_dir, "*.jpg")):
    # title=00000x(.jpg)
    title, ext = os.path.splitext(os.path.basename(pathAndFilename))
    i+=1
    # when counter equals to index_test, image's path is written to
test.txt
    if counter == index_test:
        counter = 1
        j+=1
        file_test.write(current_dir + "/" + title + '.jpg ' + "\n")
    else:
        # while counter not equal to index_test, image's path is
written to train.txt
        file_train.write(current_dir + "/" + title + '.jpg ' + "\n")
        counter+=1

# Statistics
print ('\n\n1.Number of images: '+ str(i))
print ('-----------\n2.Train Set: ' + str(i-j))
print ('-----------\n3.Test Set: ' + str(j))
print ('-----------\n4.Test Set Percentage: '+str(j/i) +'%\n-----------
\n')
```

Αφού έχουν δημιουργηθεί τα αρχεία train.txt και test.txt τα οποία περιέχουν τις διαδρομές προς τις αντίστοιχες εικόνες, πρέπει να δημιουργήσουμε το αρχείο classes.txt που περιέχει τα ονόματα των τριών (3) κλάσεων που θα χρησιμοποιηθούν:

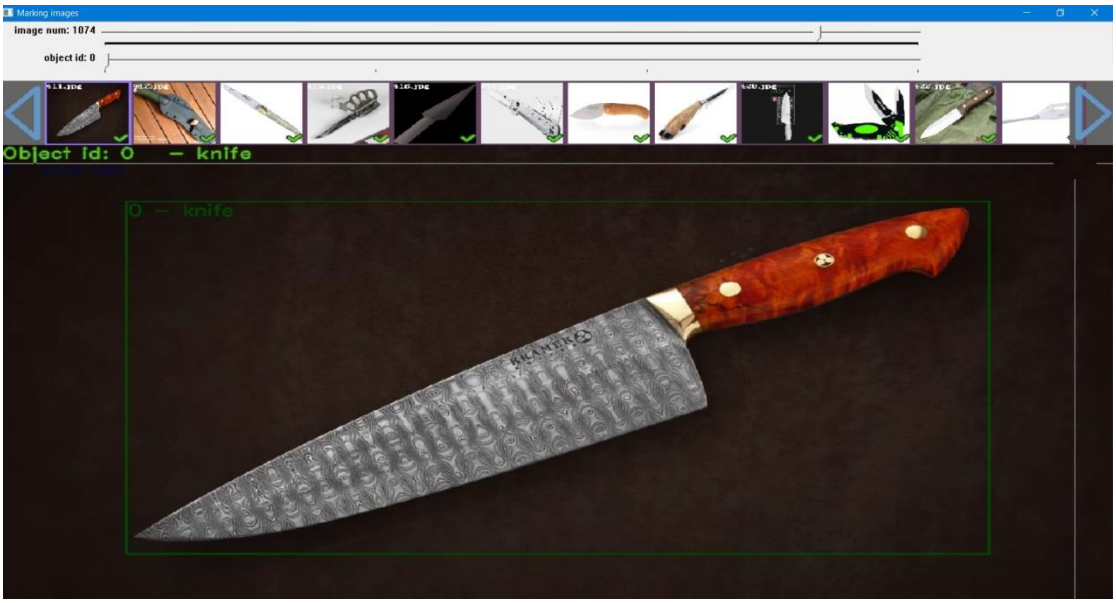
```
rifle
pistol
```

```
knive
```

Στη συνέχεια, εκτελείται το πρόγραμμα Yolo Mark, για το annotation των εικόνων. Εικόνες από την εκτέλεση του παραπάνω προγράμματος είναι οι παρακάτω:

Εικόνα 30. Screenshot από Εκπαίδευση Τυφεκίου

Εικόνα 31. Screenshot από Εκπαίδευση Πιστολίου



Εικόνα 32. Screenshot από Εκπαίδευση Μαχαιριού

Οι εικόνες που εκπαιδεύονται είναι αποθηκευμένες στο φάκελο data\img και χρησιμοποιούνται επίσης τα αρχεία obj.data obj.names και train.txt. Το πρόγραμμα εκκινεί με την εντολή:

```
yolo_mark.exe data/img data/train.txt data/obj.names
```

Όπως φαίνεται στις εικόνες, τα αντικείμενα καταγράφονται με ένα object id που είναι η κλάση κάθε αντικειμένου και υπάρχουν επιλογές για τον χειρισμό των εικόνων με πληκτρολόγιο. Για κάθε εικόνα αποθηκεύονται τα αποτελέσματα σε ένα αρχείο txt με ίδιο όνομα, με το όνομα της εικόνας. Ένα παράδειγμα είναι το παρακάτω:

```
2 0.501953 0.479167 0.983594 0.908333
```

στο οποίο φαίνεται η κλάση (2-εκκινώντας από το 0) και οι συντεταγμένες του κουτιού που περικλείει το αντικείμενο που αναγνωρίζεται.

Συνοψίζοντας, μέχρι τώρα έχουν δημιουργηθεί τα αρχεία train.txt, που περιέχουν τη διαδρομή των εικόνων με τις οποίες θα εκπαιδευτεί το δίκτυο, το test.txt με τη διαδρομή των εικόνων με τις οποίες θα επαληθευτεί το δίκτυο και τα αρχεία .txt που περιλαμβάνουν τις συντεταγμένες του οπλισμού.

5.4 Κατάλογος YOLO

Τα αρχεία του προγράμματος, θα πρέπει να μπουν σε συγκεκριμένες διαδρομές, ώστε να μπορέσει να τρέξει το πρόγραμμα. Πέραν των αρχείων του προγράμματος, που υπάρχουν από τον προγραμματιστή και τα αρχεία που αναφέρθηκαν στο προηγούμενο εδάφιο, θα προστεθούν και configuration αρχεία που περιλαμβάνουν τις διαδρομές στα παραπάνω

αρχεία, τον αριθμό των κλάσεων και τα ονόματά τους. Τα αρχεία στα οποία πρόκειται να γίνει επεξεργασία και να δημιουργηθούν είναι τα παρακάτω:

5.4.1 yolov3-tiny.cfg

Σε αυτό το αρχείο ρυθμίσεων υπάρχουν οι ρυθμίσεις για κάθε επίπεδο του συνελικτικού δικτύου. Χρησιμοποιείται και για την εκπαίδευση και την επαλήθευση. Αυτό που πρέπει να αλλάξει είναι ο αριθμός των κλάσεων που χρησιμοποιούνται (στην περίπτωση αυτή είναι ίσο με τρία (3)) και ο αριθμός των φίλτρων στα συνελικτικά επίπεδα πριν τα yolo επίπεδα, με τιμή **(classes +5)*3** (στην περίπτωση αυτή είναι ίσα με 24). Τέλος, οι άγκυρες (anchors) υπολογίζονται από την εντολή «**darknet.exe detector calc_anchors data/voc.data -num_of_clusters 9 -width 416 -height 416».** Το αρχείο είναι το παρακάτω:

```
[net]
# Testing
#batch=1
#subdivisions=1
# Training
batch=64
subdivisions=64
width=832
height=832
channels=3
momentum=0.9
decay=0.0005
angle=0
saturation = 1.5
exposure = 1.5
hue=.1

learning_rate=0.0005
burn_in=2000
max_batches = 9000
policy=steps
steps=7200,8100
scales=.1,.1

[convolutional]
batch_normalize=1
filters=16
size=3
stride=1
pad=1
activation=leaky

[maxpool]
```

```
size=2
stride=2

[convolutional]
batch_normalize=1
filters=32
size=3
stride=1
pad=1
activation=leaky

[maxpool]
size=2
stride=2

[convolutional]
batch_normalize=1
filters=64
size=3
stride=1
pad=1
activation=leaky

[maxpool]
size=2
stride=2

[convolutional]
batch_normalize=1
filters=128
size=3
stride=1
pad=1
activation=leaky

[maxpool]
size=2
stride=2

[convolutional]
batch_normalize=1
filters=256
size=3
stride=1
pad=1
activation=leaky

[maxpool]
```

```
size=2
stride=2

[convolutional]
batch_normalize=1
filters=512
size=3
stride=1
pad=1
activation=leaky

[maxpool]
size=2
stride=1

[convolutional]
batch_normalize=1
filters=1024
size=3
stride=1
pad=1
activation=leaky

###########

[convolutional]
batch_normalize=1
filters=256
size=1
stride=1
pad=1
activation=leaky

[convolutional]
batch_normalize=1
filters=512
size=3
stride=1
pad=1
activation=leaky

[convolutional]
size=1
stride=1
pad=1
filters=24
activation=linear
```

```
[yolo]
mask = 3,4,5
anchors = 214,145,  89,469, 533,121, 246,332, 528,221, 355,475, 518,329
, 553,441, 546,549
classes=3
num=9
jitter=.3
ignore_thresh = .7
truth_thresh = 1
random=0

[route]
layers = -4

[convolutional]
batch_normalize=1
filters=128
size=1
stride=1
pad=1
activation=leaky

[upsample]
stride=2

[route]
layers = -1, 8

[convolutional]
batch_normalize=1
filters=256
size=3
stride=1
pad=1
activation=leaky

[convolutional]
size=1
stride=1
pad=1
filters=24
activation=linear

[yolo]
mask = 0,1,2
```

```
anchors = 214,145,  89,469, 533,121, 246,332, 528,221, 355,475, 518,329
, 553,441, 546,549
classes=3
num=9
jitter=.3
ignore_thresh = .7
truth_thresh = 1
random=1
max=200
```

Παρατίθενται εξηγήσεις για αρχείο:

Οι εικόνες που εξετάζονται είναι σε πακέτα των **64**(batch = 64)

Το μέγιστο ύψος και πλάτος των εικόνων είναι **832** (πολλαπλάσιο του 32)

anchors: προκαθορισμένο σύνολο κιβωτίων με ιδιαίτερες αναλογίες ύψους-πλάτους, το οποίο υπολογίζεται με βάση τις εικόνες στις οποίες έχει γίνει annotation και την εντολή

```
darknet.exe detector calc_anchors data/voc.data -num_of_clusters 6 -
width 416 -height 416
```

mask: λίστα αναγνωριστικών των πλαισίων οριοθέτησης (άγκυρες) που το επίπεδο (layer) είναι υπεύθυνο για την πρόβλεψη

num: συνολικός αριθμός αγκυρών

filter = (κλάσεις + 5) * k όπου k = ο αριθμός της μάσκας σε ένα στρώμα yolo (**24)**

max_batches = οι επαναλήψεις που εκτελεί ο αλγόριθμος

Το YOLOv3 προβλέπει προβλέψεις από ένα προκαθορισμένο σύνολο πλαίσιο με συγκεκριμένες αναλογίες ύψους-πλάτους (άγκυρες). Οι άγκυρες είναι αρχικά μεγέθη (πλάτος, ύψος), μερικά από τα οποία (το πλησιέστερο στο μέγεθος του αντικειμένου) θα αλλάζουν μέγεθος στο μέγεθος του αντικειμένου. Κάθε επίπεδο [yolo] πρέπει να γνωρίζει όλα τα πλαίσια άγκυρας, αλλά είναι υπεύθυνο μόνο για ένα υποσύνολο αυτών. Η μάσκα "λέει" στο στρώμα ποιό από τα πλαίσια οριοθέτησης πρέπει να χρησιμοποιήσει για την πρόβλεψη.

Το παραπάνω αρχείο θα μετονομαστεί σε yolov3-tiny-obj.cfg και βρίσκεται στη διαδρομή darknet\cfg.

5.4.2 objects.data

Στο αρχείο αυτό εγγράφεται ο αριθμός των κλάσεων, η διαδρομή του αρχείου train.txt, η διαδρομή του αρχείου test.txt, η διαδρομή των ονομάτων των κλάσεων και ο φάκελος **backup**, στον οποίο αποθηκεύονται τα βάρη του εκπαιδευμένου δικτύου. Το αρχείο που θα χρησιμοποιηθεί είναι το παρακάτω:

```
classes =3
train = data/train.txt
```

```
valid = data/test.txt
names = data/objects.names
backup = backup/
```

Το παραπάνω αρχείο βρίσκεται στη διαδρομή darknet/data.

5.4.3 objects.names

Στο αρχείο αυτό εγγράφονται τα ονόματα των κλάσεων. Βρίσκεται στη διαδρομή darknet/data και είναι το παρακάτω:

```
rifle
pistol
knives
```

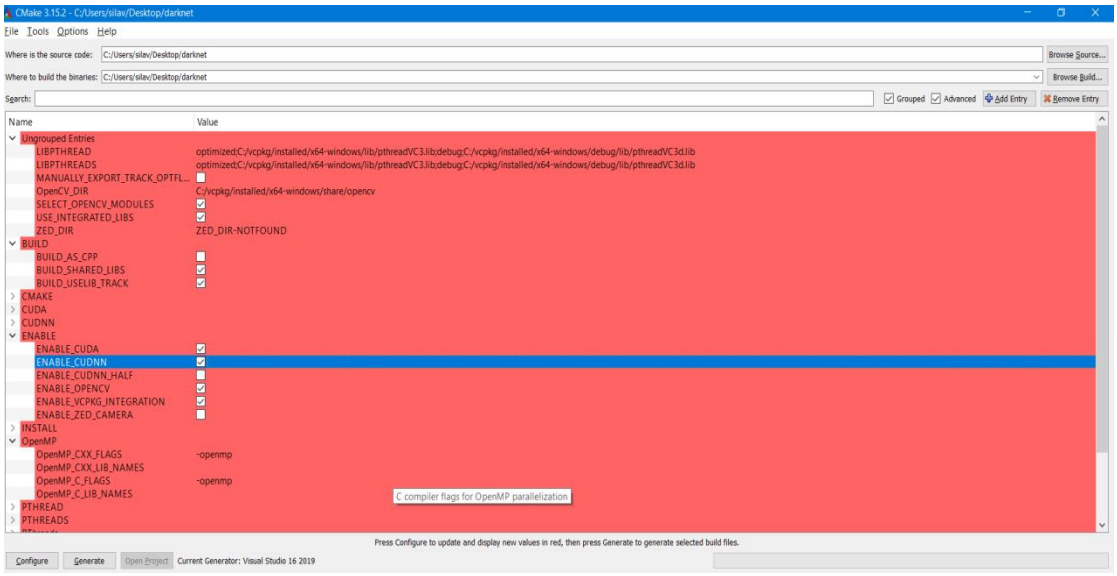
5.5 Εγκατάσταση YOLOv3

Για την εγκατάσταση ακολουθήθηκαν τα παρακάτω βήματα:

1. Εγκατάσταση του OpenCV.
2. Εγκατάσταση ή ενημέρωση του Visual Studio 16 2019
3. Εγκατάσταση των CUDA και cuDNN. Αυτά είναι βιβλιοθήκες ανεπτυγμένες από την εταιρεία NVIDIA με σκοπό να βοηθήσουν τους προγραμματιστές να "τρέξουν" προγράμματα σε κάρτες γραφικών(GPU) της NVIDIA.
4. Εγκατάσταση των git and cmake για Windows περιβάλλον και δημιουργία διαδρομής στο σύστημα.
5. Εγκατάσταση του vcpkg. Ανεπτυγμένο από τη Microsoft, το vcpkg είναι ένας διαχειριστής βιβλιοθηκών C++ , γλώσσα στην οποία είναι ανεπτυμένη η συγκεκριμένη έκδοση του yolo που χρησιμοποιείται.
6. Προσδιορισμός των περιβαλλοντικών μεταβλητών (environment variables) **VCPKG_ROOT**, στην διαδρομή εγκατάστασης του vcpkg.
7. Προσδιορισμός των περιβαλλοντικών μεταβλητών **VCPKG_DEFAULT_TRIPLET** με τιμή **x64-windows.**
8. Στο περιβάλλον Powershell πληκτρολογούνται οι παρακάτω εντολές:

```
PS \>cd $env:VCPKG_ROOT
PS Code\vcpkg>.\vcpkg install pthreads opencv[cuda,ffmpeg]
```

9. Εκτέλεση του προγράμματος CMake-GUI με source code και το χώρο εγκατάστασης των binaries στη διαδρομή του φακέλου darknet και επιλογή "Configure"
10. Ενημέρωση των επιλογών, όπως φαίνεται στην παρακάτω εικόνα:

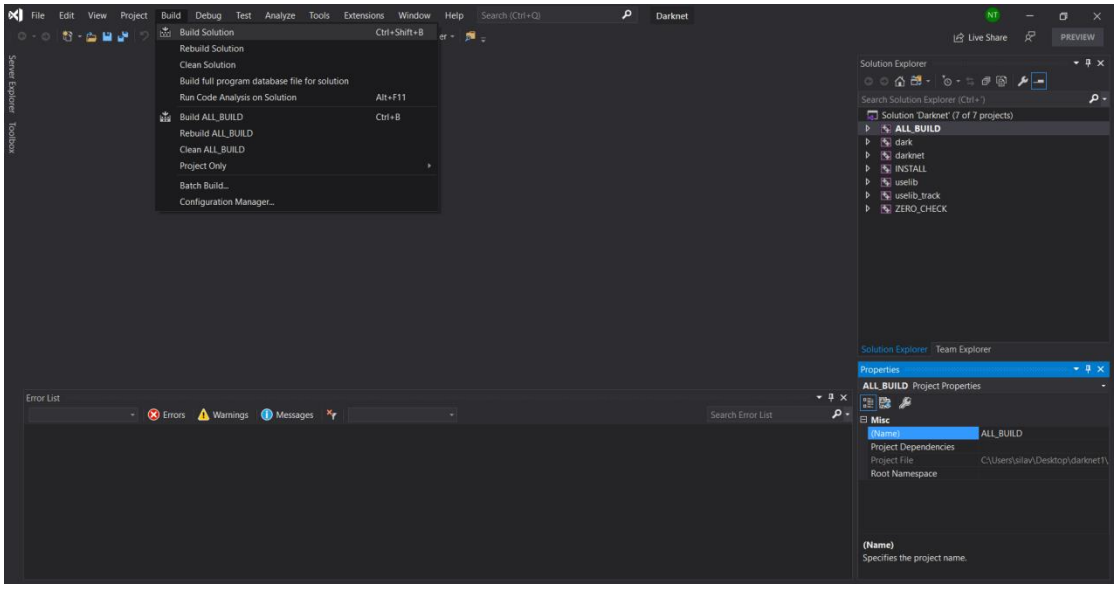


Εικόνα 33. CMake - GUI

11. Επιλογή Generate και Open Project.

12. Επιλογή των Release και x64

13. Επιλογή Build -> Build Solution:



Εικόνα 34. Building On Visual Studio

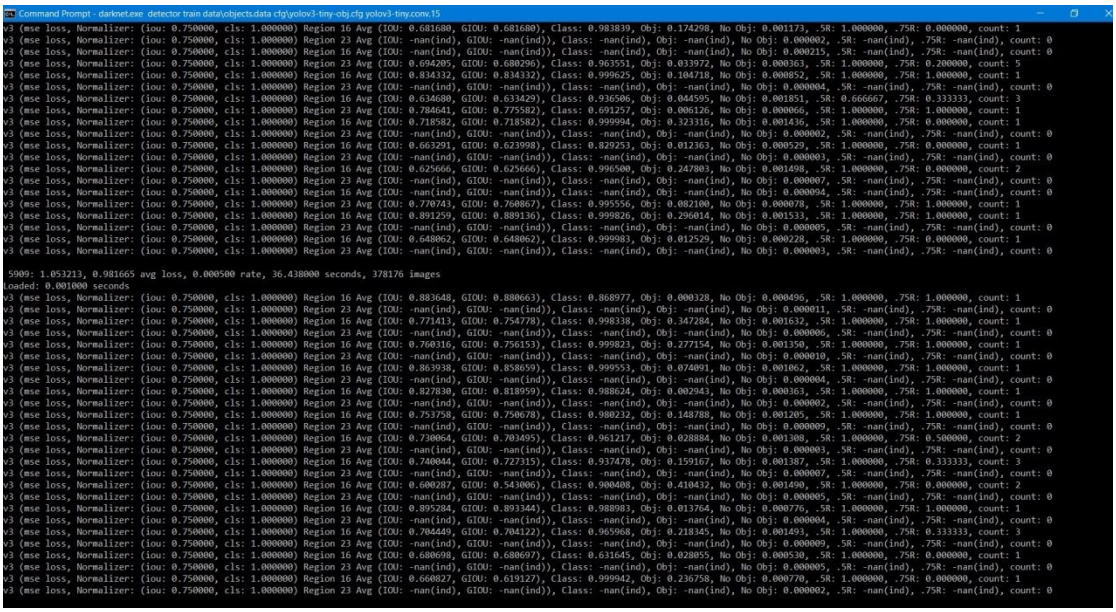
14. Έχει δημιουργηθεί το darknet.exe στην παραπάνω διαδρομή και είναι έτοιμο προς χρήση.

5.6 Εκπαίδευση Στατικών Εικόνων

Για την εκπαίδευση του συστήματος, χρησιμοποιούμε την παρακάτω εντολή:

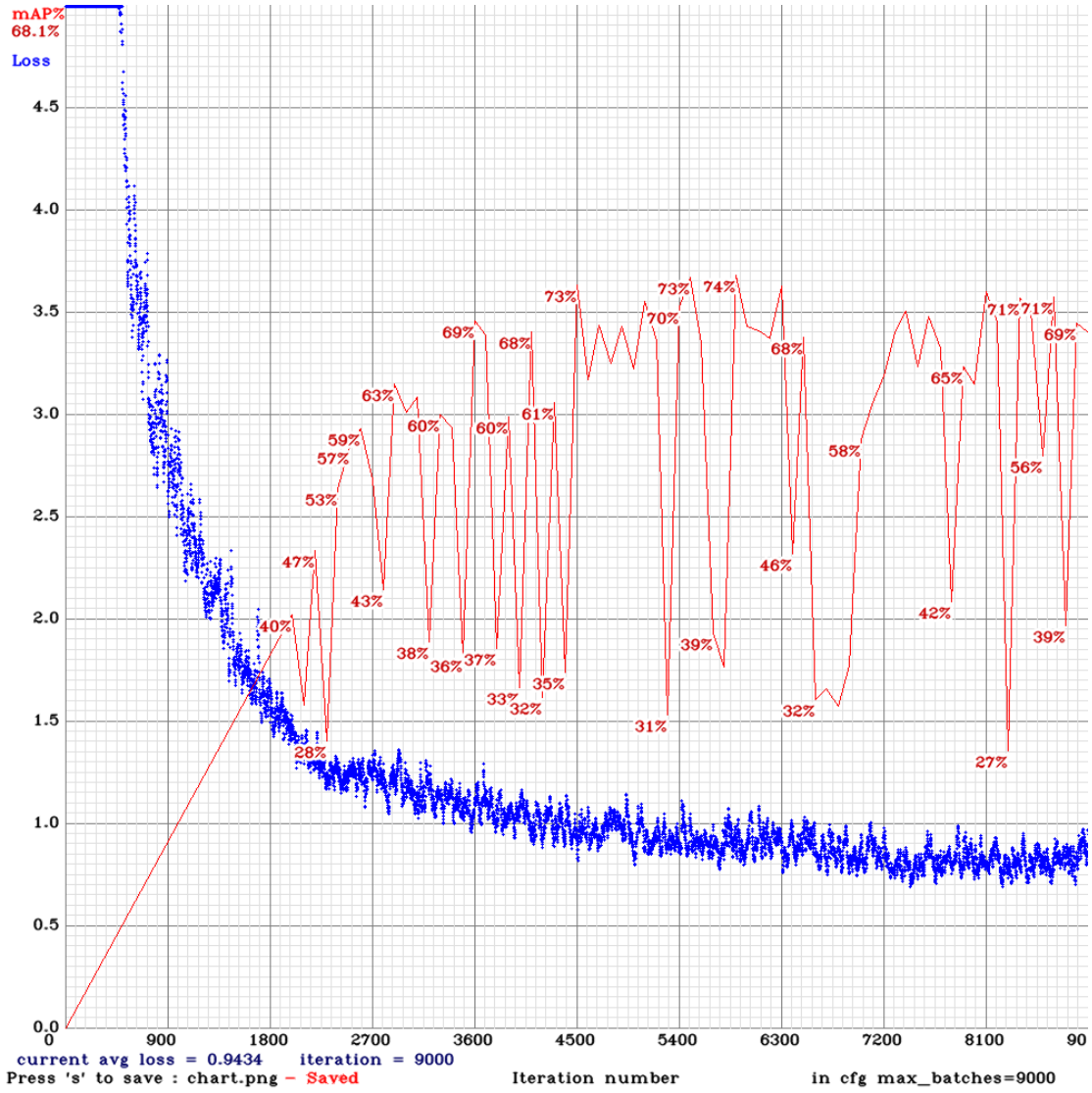
```
darknet.exe detector train data\objects.data cfg\yolov3-tiny-obj.cfg
yolov3-tiny.conv.15
```

Ενδεικτική εικόνα της εκτέλεσης της παραπάνω εντολής είναι η εξής:



Εικόνα 35. Στιγμιότυπο Εκπαίδευσης Δικτύου

Μία σημαντική παράμετρος είναι ο όρος mAP (mean Average Precision) το οποίο είναι ένα μέτρο ακρίβειας για τα δεδομένα. Κατά την εκπαίδευση, είναι δυνατόν να αποδοθεί γραφικά η παράμετρος αυτή, καθώς επίσης και το **average loss,** για τον έλεγχο της σωστής εκπαίδευσης του δικτύου, όπως φαίνεται στην παραπάνω εικόνα.

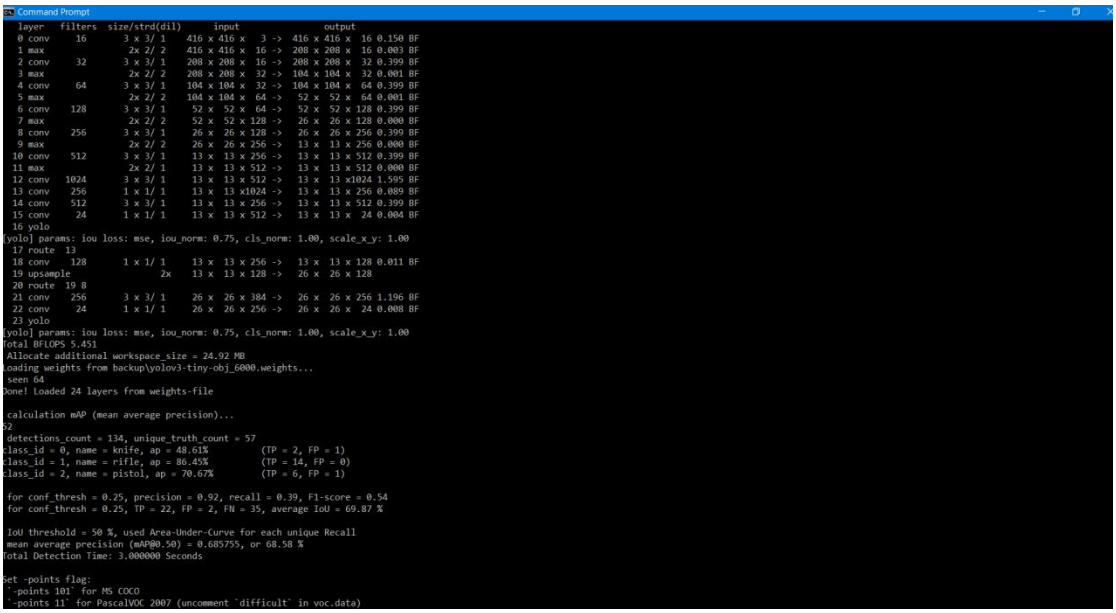


Εικόνα 36. Γράφημα mAP & Average Loss

Με την κάτωθι εντολή, μετράται η παράμετρος mAP σε συγκεκριμένα βάρη:

```
darknet.exe detector map data\objects.data cfg\yolov3-tiny-obj.cfg
backup#yolov3-tiny-obj_6000.weights
```

και έξοδο:



5.7 Αναγνώριση Αντικειμένων

Η αναγνώριση των αντικειμένων σε στατική εικόνα γίνεται με την παρακάτω εντολή:

```
darknet.exe detector test data\objects.data cfg\yolov3-tiny-obj-test.cfg backup\yolov3-tiny-obj_6000.weights data\img\0000.jpg
```

Το πρόγραμμα δίνει τη δυνατότητα αναγνώρισης αντικειμένων σε βίντεο με την παρακάτω εντολή:

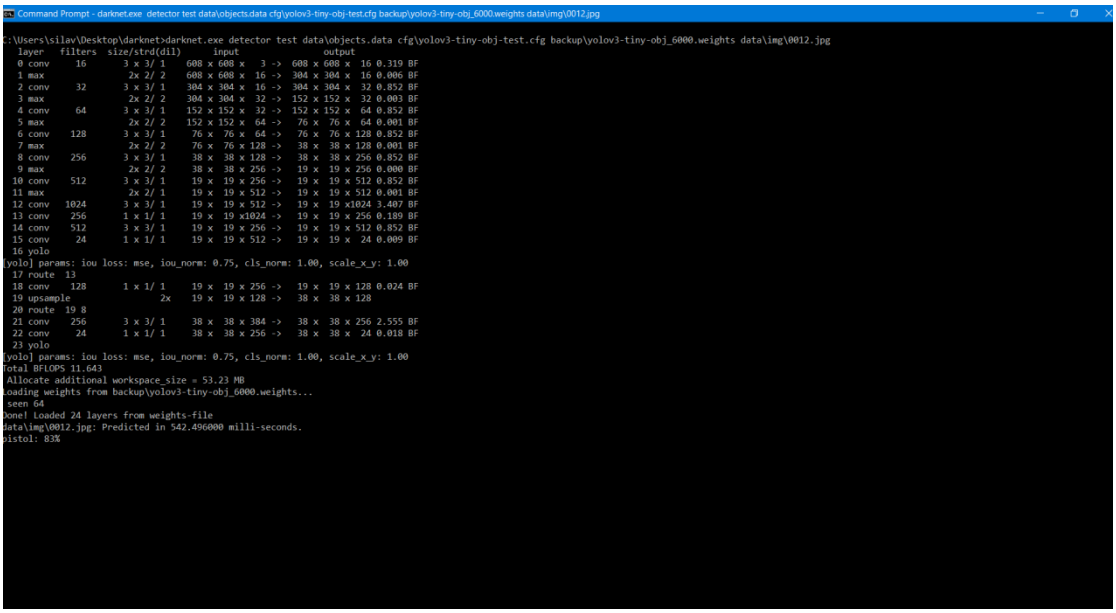
```
darknet.exe detector demo data\objects.data cfg\yolov3-tiny-obj-test.cfg backup\yolov3-tiny-obj_6000.weights guns.mp4
```

Η ταχύτητα αναγνώρισης σε βίντεο εξαρτάται από τη «δύναμη» του υλικού, αφού το πρόγραμμα εξετάζει το βίντεο ανά εικόνες (frames) για την εύρεση των αντικειμένων. Αντίστοιχα, προσφέρεται η επιλογή αναγνώρισης μέσω κάμερας με live feed και την εντολή:

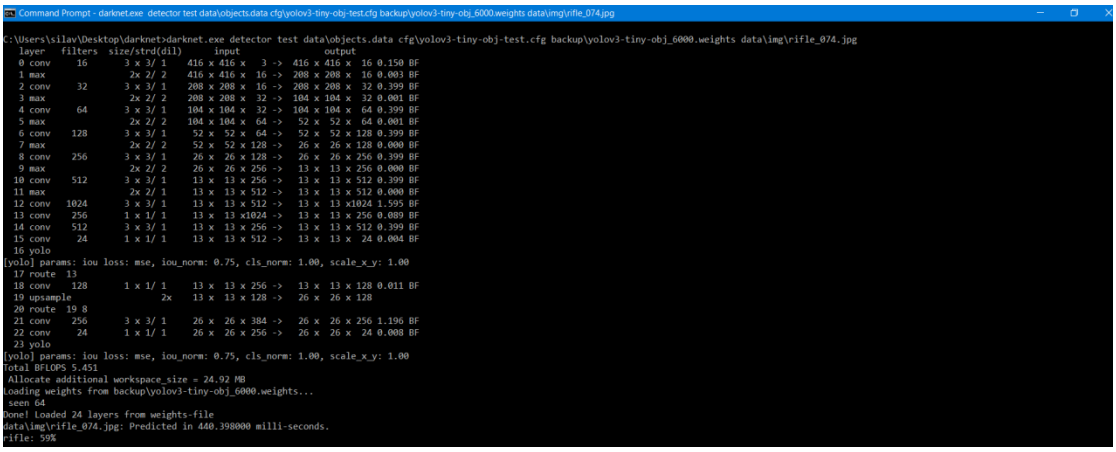
```
darknet.exe detector demo data\objects.data cfg\yolov3-tiny-obj-test.cfg backup\yolov3-tiny-obj_6000.weights
```

Ενδεικτικές εικόνες από την αναγνώριση οπλισμού είναι οι παρακάτω:

Εικόνα 37. Αναγνώριση Πιστολιού με 83% Πιθανότητα



Εικόνα 38. Αναγνώριση Τυφεκίου με Πιθανότητα 59%



5.8 *Προβλήματα Αναγνώρισης Αντικειμένων*

Τα επικίνδυνα υλικά που εξετάζονται στην παρούσα διατριβή είναι ο ατομικός οπλισμός. Λόγω του μικρού μεγέθους τους, του σκούρου χρώματος και των πολλών τρόπων που φέρονται, είναι αρκετές φορές δύσκολο να αναγνωριστούν σε όλες τις γωνίες που μπορεί να ληφθούν από την κάμερα. Τρόπος για την επίλυση αυτού του προβλήματος είναι η λήψη όσο το δυνατόν περισσότερων φωτογραφιών από πολλές γωνίες λήψης και από διάφορες συνθήκες φωτισμού και πιθανόν σε ειδικό για τούτο χώρο. Επίσης, η χρησιμοποίηση ισχυρού υπολογιστή με μεγάλη δύναμη σε υπολογιστική δύναμη μέσω καρτών γραφικών είναι πλεονέκτημα για την εξαγωγή μεγαλύτερης πιθανότητας (σιγουριάς) στην αναγνώριση του οπλισμού.

5.9 *Χρήση της Τεχνολογίας*

Είναι φανερό ότι η τεχνολογία αυτή είναι απαραίτητη για την αντιμετώπιση ενός συνόλου θεμάτων που σχετίζονται με την ασφάλεια, τόσο σε εθνικό επίπεδο όσο και ατομικό – επιχειρησιακό. Η αναγνώριση επικίνδυνων υλικών προλαμβάνει επιπτώσεις που μπορεί να έχουν καταστροφικές συνέπειες στην ανθρώπινη ζωή κυρίως, αλλά και σε κτιριακές – οικονομικές υποδομές. Η βελτίωση και αναβάθμιση αυτής με ισχυρό υλικό, συνεχή εκπαίδευση και ενημέρωση των προγραμμάτων που χρησιμοποιούνται, θα εμποδίσουν τις απώλειες στη ζωή και στην παραγωγικότητα των εταιρειών, ειδικά όσο αυξάνονται οι ασύμμετρες απειλές από άτομα που δρουν μόνα τους, ανεξέλεγκτα και ξαφνικά.

Βιβλιογραφία

[1] Ιωάννης Βλαχάβας, Νικόλαος Βασιλειάδης, Φώτης Κόκκορας, Ηλίας Σακελλαρίου, Πέτρος Κεφάλας, *Τεχνητή νοημοσύνη Β' Έκδοση*. Εκδόσεις Πανεπιστημίου Μακεδονίας, 2011.

[2] Igor Aleksander, Helen Morton, *An introduction to neural computing*. 1990.

[3] "https://en.wikipedia.org/wiki/Perceptron."

[4] Stuart J. Russell, Peter Norvig, "Artificial Intelligence: A Modern Approach, Third Edition." 2010.

[5] "https://en.wikipedia.org/wiki/Supervised_learning#Algorithm_choice."

[6] "https://towardsdatascience.com/supervised-learning-basics-of-linear-regression-1cbab48d0eba."

[7] "https://en.wikipedia.org/wiki/Unsupervised_learning."

[8] "https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53."

[9] David A. Forsyth, Jean Ponce, *Computer Vision: A Modern Approach (2nd Edition)*. Pearson, 2011.

[10] S. E. Palmer, "Vision science: Photons to phenomenology." MIT press, 1999.

[11] "https://blog.zenggyu.com/en/post/2018-12-16/an-introduction-to-evaluation-metrics-for-object-detection/."

[12] "https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e."

[13] "https://arxiv.org/abs/1311.2524." .

[14] "http://www.huppelen.nl/publications/selectiveSearchDraft.pdf." .

[15] "https://arxiv.org/abs/1504.08083." .

[16] "https://papers.nips.cc/paper/5638-faster-r-cnn-towards-real-time-object-detection-with-region-proposal-networks.pdf." .

[17] "https://pjreddie.com/darknet/yolo/." .

[18] "https://github.com/ivangrov." .