ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΥΠΟΛΟΓΙΣΤΩΝ

Δημιουργία ενός Real-Time Object Detector με τη χρήση του Tensorflow

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ιάσων Ι. Μαυρομμάτης

Επιβλέπων : Πέτρος Στεφανέας
Επ. Καθηγητής Ε.Μ.Π

Αθήνα, Ιούλιος 2022

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΥΠΟΛΟΓΙΣΤΩΝ

Δημιουργία ενός Real-Time Object Detector με τη χρήση του Tensorflow

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ιάσων Ι. Μαυρομμάτης

Επιβλέπων : Πέτρος Στεφανέας
Επ. Καθηγητής Ε.Μ.Π

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 27η Ιουλίου 2022.

.....	.....	.....
Πέτρος Στεφανέας	Παναγιώτης Τσανάκας	Αριστείδης Παγουρτζής
Επ. Καθηγητής Ε.Μ.Π	Καθηγητής Ε.Μ.Π	Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούλιος 2022

.....
Ιάσων Ι. Μαυρομμάτης

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Η παρούσα διπλωματική εργασία έχει ως σκοπό τη δημιουργία μιας εφαρμογής αναγνώρισης φύλλων τράπουλας σε πραγματικό χρόνο με την χρήση της κάμερας του υπολογιστή. Στο πρώτο κεφάλαιο θα γίνει παρουσίαση της Τεχνητής Νοημοσύνης. Θα αναφέρουμε κάποια ιστορικά γεγονότα, τους τύπους της, τα στοιχεία που την απαρτίζουν και τους κλάδους αυτής. Στο δεύτερο κεφάλαιο θα εστιάσουμε στην ανίχνευση αντικειμένων, όπου θα αναφερθούν οι μέθοδοι με τους οποίους μπορεί να γίνει. Θα αναφερθεί συνοπτικά ο τρόπος με τον οποίο τελείται η ανίχνευση αντικειμένων και στο τέλος θα εστιάσουμε σε πρακτικές εφαρμογές της ανίχνευσης αντικειμένων. Στο τρίτο κεφάλαιο θα παρουσιαστεί το θεωρητικό υπόβαθρο, στο οποίο στηρίζεται η ανίχνευση αντικειμένων. Πιο συγκεκριμένα θα αναλύσουμε τον τρόπο λειτουργίας των Νευρωνικών και των Συνελικτικών Νευρωνικών Δικτύων και στο τέλος του κεφαλαίου θα παρουσιαστούν έννοιες, ορισμοί και μετρικές που χρησιμοποιούνται για την Ανίχνευση Αντικειμένων. Στο τέταρτο κεφάλαιο θα αναφέρουμε το τεχνικό υπόβαθρο, το οποίο χρησιμοποιήθηκε για την δημιουργία της εφαρμογής. Στο πέμπτο κεφάλαιο θα παρουσιαστεί η πειραματική διάταξη της εφαρμογής καθώς και τα αποτελέσματα τα οποία προέκυψαν από αυτή. Στο τέλος θα προτείνουμε μελλοντικές επεκτάσεις που μπορούν να γίνουν στην παρούσα διπλωματική.

Λέξεις κλειδιά

Ανίχνευση αντικειμένων, Tensorflow, Νευρωνικά Δίκτυα, Συνελικτικά Νευρωνικά Δίκτυα, Τεχνητή Νοημοσύνη, SSD, Ηθική

Abstract

This thesis has as a goal the creation of a real-time Object Detector application, which can detect cards from a card deck through the usage of a web-camera. The first chapter presents the AI. More specificly, we are going to present the history, sub-types, sub-fields and characteristics of AI. In the second chapter we are going to make an introduction into Object Detection. This chapter informs the reader about the methods of Object Detection, the process of Object Detection and some uses of Object Detection in the real world. The third chapter is going to present the theoritical backround in which the Object Detection leans on. We are going to present how Neural Networks and Convolutional Neural Networks work and after that we are going to present meanings, definitions and metrics of Object Detection. In the fourth chapter, we are going to present the technical backround, which is used for the creation of the application. In the fifth chapter, we are going to show the expiremental layout of the Custom Detector and the results from it. Finally, we are going to make suggestions on further extensions on this thesis.

Key words

Object Detection, Tensorflow, Neural Networks, Convolutional Neural Networks, Artificial Intelligence, SSD, Ethics

Ευχαριστιές

Με την παρούσα διπλωματική εργασία κλείνει ένας καθοριστικός κύκλος για εμένα, αυτός των προπτυχιακών σπουδών. Μέσω αυτής της πορείας κατάφερα να αναβαθμιστώ τόσο ως άνθρωπος όσο και ως επιστήμονας μέσα από συνεχείς προκλήσεις και εμπειρίες. Σε αυτό το σημείο θα ήθελα να ευχαριστήσω τα άτομα που με συντρόφευσαν σε αυτήν την πορεία.

Αρχικά θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου κύριο Πέτρο Στεφανέα για την ανάθεση του ενδιαφέροντος θέματος με το οποίο ασχολήθηκα. Θα ήθελα επίσης να τον ευχαριστήσω για τη συνέχη στήριξή του τόσο σε ψυχολογικό όσο και σε επιστημονικό επίπεδο, την ενθάρρυνση που μου παρείχε και την πολύτιμη βοήθεια του καθόλη την εκπόνηση της διπλωματικής εργασίας. Επίσης, θα ήθελα να ευχαριστήσω τον κύριο Πέτρο Ποτίκα για τη συνεισφορά του και τη συνδρομή του κατά τη διάρκεια ενασχόλησής μου με την παρούσα διπλωματική εργασία. Τέλος, θα ήθελα να ευχαριστήσω τους καθηγητές μου, και ιδίως, τον κύριο Παναγιώτη Τσανάκα και τον κύριο Αριστείδη Παγουρτζή για το ενδιαφέρον που μου καλλιέργησαν για τον κλάδο της Επιστήμης των Υπολογιστών.

Σε προσωπικό επίπεδο, θα ήθελα να ευχαριστήσω τους συμφοιτητές μου Δήμητρα, Θοδωρή, Βλάσση, Ιάσονα, Αργύρη και Νίκο, οι οποίοι με βοήθησαν και έκαναν το ταξίδι αυτό ακόμα πιο ευχάριστο. Από φίλους θα ήθελα να ευχαριστήσω τον Ρέι και την Ήλια για τη διαρκή και ουσιαστική στήριξη που μου παρείχαν.

Ευχαριστώ επίσης τους γονείς μου Αιμιλία και Ιωάννη, τις αδερφές μου Μαρία, Στεφανία και Αθηνά για την υπόμονή,τη συνεχή υποστήριξή τους και την ενθάρρυνση καθόλη τη διάρκεια των σπουδών.

Περιεχόμενα

Κατάλογος σχημάτων

Κεφάλαιο 1 : Τεχνητή Νοημοσύνη

1.0 - Εισαγωγή

Figure 1: Τεχνητή Νοημοσύνη

Η τεχνολογία στις μέρες μας εξελίσσεται με ιλλιγιώδεις ρυθμούς. Αποτέλεσμα αυτής της εξέλιξης είναι και η δημιουργία του κλάδου της Τεχνητής Νοημοσύνης. Αξίζει να αναφέρουμε λοιπόν, κάποια ιστορικά στοιχεία που την αφορούν.

Πρώτος ο Turing ή αλλιώς «Πατέρας της Επιστήμης των Υπολογιστών» για πολλούς, το 1950 έκανε την πρώτη αναφορά στη Τεχνητή Νοημοσύνη με το περίφημο «Turing Test». Το test έχει ως εξής: Ένας άνθρωπος συμμετέχει σε μια συζήτηση με μια μηχανή. Αν ο άνθρωπος δεν καταλάβει ότι μιλάει σε μηχανή τότε η μηχανή περνάει το test. Αυτό το test σηματοδοτεί την αρχή της Τεχνητής Νοημοσύνης.

Σύμφωνα με δήλωση του John McCarthy, το 2004, «η Τεχνητή Νοημοσύνη είναι η επιστήμη και ο μηχανισμός, με τον οποίο δημιουργείται ένα έξυπνο μηχάνημα και πιο ειδικά ένα έξυπνο πρόγραμμα. Αλλιώς θα μπορούσαμε να πούμε ότι είναι ο σχεδιασμός συστημάτων που έχουν παρόμοια χαρακτηριστικά με την ανθρώπινη ευφυΐα.» Πιο αναλυτικά, είναι η υλοποίηση συστημάτων που είναι ικανά να τελέσουν διαδικασίες που κανονικά απαιτούν ανθρώπινη νοημοσύνη, όπως λήψη αποφάσεων, αναγνώριση αντικειμένων, επίλυση πολύπλοκων προβλημάτων, η εκμάθηση, η συλλογιστική, η κατανόηση της γλώσσας και άλλα.

Ο Stuart Russell και Peter Norvig στο σύγγραμμα τους «Artificial Intelligence: A Modern Approach» εμβαθύνουν στις τέσσερις προσεγγίσεις για να καθορίσουν την Τεχνητή Νοημοσύνη:

1. Systems that think like humans («Συστήματα που σκέφτονται σαν άνθρωποι»)

2. Systems that think rationally («Συστήματα που σκέφτονται λογικά»)

3. Systems that act like humans («Συστήματα που συμπεριφέρονται σαν άνθρωποι»)

4. Systems that act rationally («Συστήματα που συμπεριφέρονται λογικά»)

Οι πρώτες δύο προσεγγίσεις ασχολούνται με τη σκέψη και τη λογική, ενώ οι υπόλοιπες επικεντρώνουν το ενδιαφέρον τους στη δράση-συμπεριφορά. Με αυτόν τον τρόπο, βλέπουμε λοιπόν, ότι ο ορισμός του Turing εμπίπτει πιο πολύ στη προσέγγιση «systems that act like humans».

Συμπεραίνουμε λοιπόν, ότι η Τεχνητή Νοημοσύνη αποτελεί ένα εργαλείο για την επίλυση προβλημάτων, μέσω του συνδυασμού Computer Science και δεδομένων. Η Μηχανική Μάθηση (Machine Learning) και Βαθιά Μάθηση (Deep Learning) αποτελούν τα δύο κυριότερα υπο-πεδία της Τεχνητής Νοημοσύνης. Αυτοί οι δύο υπο-κλάδοι καταφέρνουν μέσα από AI αλγορίθμους να σχεδιάσουν συστήματα είτε πρόβλεψης (prediction systems) είτε κατηγοριοποίησης (classification systems). Βλέπουμε λοιπόν ότι η Τεχνητή Νοημοσύνη είναι ένα πεδίο στην Επιστήμη των Υπολογιστών, το οποίο με την πάροδο του χρόνου αποκτά ολοένα και μεγαλύτερο ενδιαφέρον, με αποτέλεσμα ολοένα και περισσότεροι developers και ερευνητές να ασχολούνται με αυτή.

1.1 - Στοιχεία της Τεχνητής Νοημοσύνης

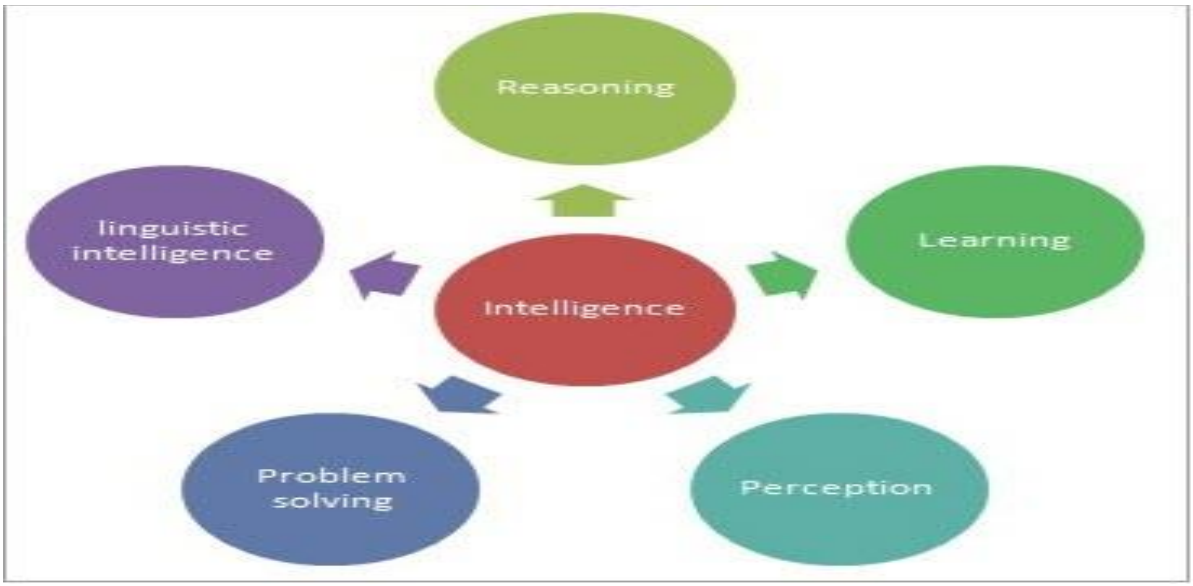


Figure 2: Στοιχεία Τεχνητής Νοημοσύνης

1) **Reasoning (Συλλογιστική):** Είναι η διαδικασία που μας διευκολύνει να παρέχουμε τα βασικά κριτήρια και τις κατευθυντήριες γραμμές για την κρίση, την πρόβλεψη και τη λήψη αποφάσεων σε οποιοδήποτε πρόβλημα. Χωρίζεται σε δύο τύπους:

   a) Generalized reasoning που βασίζεται στα γενικά παρατηρούμενα γεγονότα (incidences) και δηλώσεις (statements). Τα αποτελέσματα που προκύπτουν με βάση αυτή τη συλλογιστική ορισμένες φορές είναι λανθασμένα.

   b) Logical reasoning που βασίζεται σε γεγονότα (facts), τύπους (figures), συγκεκριμένες δηλώσεις (statements) και γεγονότα (incidences). Κατά αυτόν τον τρόπο, τα συμπεράσματα που απορρέουν είναι σωστά και λογικά.

2) **Learning (Μάθηση):** Είναι η διαδικασία, με την οποία παρέχεται γνώση και αναπτύσσονται δεξιότητες μέσω διάφορων πηγών όπως βιβλία, αληθινά περιστατικά, εμπειρίες, διδασκαλία από κάποιους ειδικούς και άλλα. Την δυνατότητα της μάθησης δεν την έχουν μόνο οι άνθρωποι, αφού και έξυπνα συστήματα (intelligent systems) μπορούν να την αποκτήσουν.

3) **Problem Solving (Επίλυση προβλημάτων):** Είναι η διαδικασία με την οποία αναγνωρίζεται το πρόβλημα και διερευνώνται τρόποι πιθανής επίλυσής του. Αυτό επιτυγχάνεται με την ανάλυση του προβλήματος, τη λήψη αποφάσεων και την αναζήτηση επιπλέον λύσεων, προκειμένου να βρεθεί η καλύτερη δυνατή λύση. Με τον όρο αυτό συνήθως εννοούμε την απόκτηση των καλύτερων δυνατών αποτελέσματων στον λιγότερο δυνατό χρόνο ή μια καλή ισορροπία

ανάμεσα σε αυτά τα δύο χαρακτηριστικά.

4) **Perception (Αντίληψη) :** Είναι η διαδικασία της απόκτησης, εξαγωγής συμπερασμάτων, επιλογής και συστηματοποίησης των χρήσιμων δεδομένων από τα ακατέργαστα δεδομένα (raw data). Στους ανθρώπους, η αντίληψη προέρχεται από τα αισθητήρια όργανα, εμπειρίες και τις συνθήκες του περιβάλλοντος. Σε αντίθεση, η αντίληψη της τεχνητής νοημοσύνης αποκτάται από τεχνητούς αισθητήρες σε συνδυασμό με την επεξεργασία δεδομένων με λογικό τρόπο.

5) **Linguistic Intelligence (Γλωσσική Νοημοσύνη):** Είναι η ικανότητα κάποιου να αναπτύσσει, να κατανοεί, να διαβάζει και να γράφει λεκτικά σύνολα σε διαφορετικές γλώσσες. Είναι το βασικό συστατικό του τρόπου επικοινωνίας μεταξύ δύο ή περισσότερων ατόμων και απαραίτητο επίσης για την αναλυτική και λογική κατανόηση.

1.2 - Πεδία της Τεχνητής Νοημοσύνης

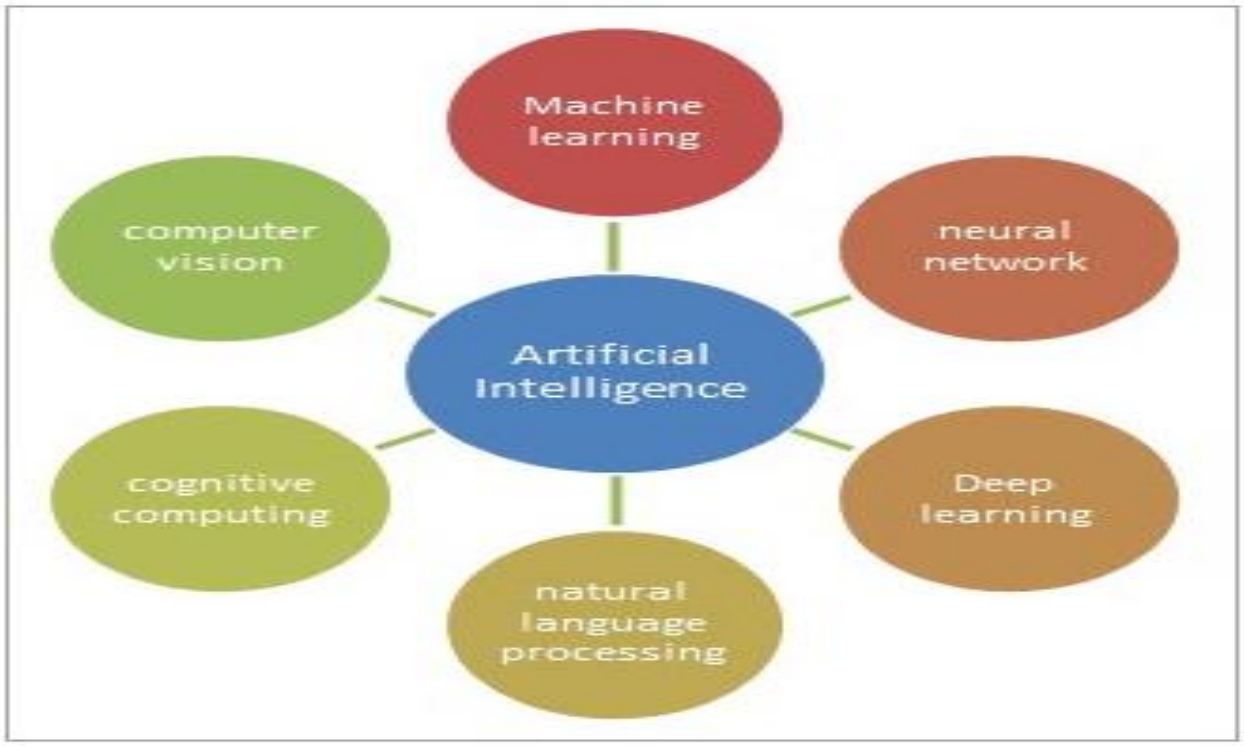


Figure 3: Πεδία Τεχνητής Νοημοσύνης

1) **Machine Learning (Μηχανική Μάθηση):** Η Μηχανική Μάθηση παρέχει τη δυνατότητα στον υπολογιστή μέσα από τη συλλογή δεδομένων ή τις περιπτώσεις που έχει αντιμετωπίσει, να μαθαίνει για το εκάστοτε πρόβλημα χωρίς να χρειάζεται περαιτέρω και πιο ειδικός προγραμματισμός. Η Μηχανική Μάθηση δίνει έμφαση στην ανάπτυξη αλγορίθμων που είναι ικανοί να ελέγχουν δεδομένα και να κάνουν προβλέψεις. Το Pattern Recognition (Αναγνώριση Προτύπων) είναι μια υποκατηγορία της Μηχανικής Μάθησης. Μπορεί να οριστεί ως η αυτόματη αναγνώριση του σχεδιαγράμματος (blueprint) από ακατέργαστα

δεδομένα με την χρήση αλγορίθμων. Αναλύουμε για λόγους πληρότητας τα στάδια του Pattern recognition:

a) Απόκτηση Δεδομένων: Περιλαμβάνει τη συλλογή ακατέργαστων δεδομένων (raw data), όπως φυσικές μεταβλητές και μετρικών (όπως συχνότητα, εύρος, ανάλυση).

b) Προεπεξεργασία των δεδομένων εισόδου: Σε αυτό το στάδιο περιλαμβάνεται το φιλτράρισμα των ανεπιθύμητων δεδομένων, όπως τυχόν θόρυβος και το σήμα-είσοδος υπόκειται σε επεξεργασία.

c) Εξαγωγή Χαρακτηριστικών: Διάφοροι αλγόριθμοι εκτελούνται όπως ο αλγόριθμος αντιστοίχισης προτύπων για να βρεθεί το αντίστοιχο μοτίβο «pattern», καθώς αυτό απαιτείται από την πλευρά των χαρακτηριστικών.

d) Classification (Ταξινόμηση): Η τάξη (class) αποδίδεται στο μοτίβο (pattern), βάσει των εξόδων των αλγορίθμων που εκτελέστηκαν και τα διάφορα μοντέλα εκμάθησης για την απόκτηση του matching pattern.

e) Post-processing: Εδώ προκύπτει το τελικό αποτέλεσμα και διασφαλίζεται ότι το τελικό output που θα επιτευχθεί είναι σχεδόν το ίδιο με αυτό που πιθανώς χρειαζόμαστε.

Model for Pattern Recognition:

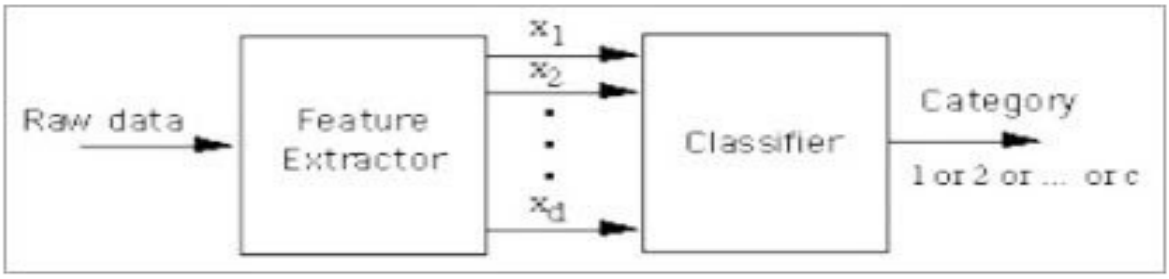


Figure 4:Model for Pattern Recognition

Παράδειγμα αναγνώρισης ενός τριγώνου.

Για να καταλάβουμε για τι είδους σχήμα πρόκειται, το χαρακτηριστικό (feature) το οποίο θέλουμε να εξαχθεί, είναι το πλήθος των πλευρών. Ένα τρίγωνο ξέρουμε ότι έχει τρεις πλευρές. Το σχήμα που ακολουθεί μας βοηθά να κατανοήσουμε το ανωτέρω μοντέλο.

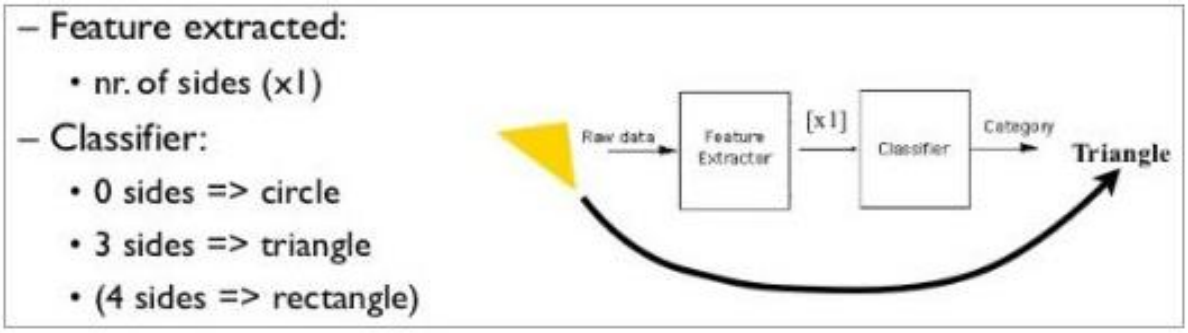


Figure 5: Classifier Σχημάτων

Εναλλακτικά το ίδιο πρόβλημα θα μπορούσε να επιλυθεί αν μπορούσαμε να εξάγουμε όλες τις γωνίες του σχήματος. Θα αποτελούσε τρίγωνο αν και μόνο αν το άθροισμα των γωνιών ήταν ίσο με 180 μοίρες. Δηλαδή το μοντέλο μας αναγνωρίζει τις γωνίες μας, εξάγει τις μοίρες τους και τελικά αν το άθροισμα όλων αυτών είναι 180, τότε αποτελεί τρίγωνο.

2) **Deep Learning (Βαθιά Μάθηση)**: Η Βαθιά Μάθηση είναι ένα υποπεδίο της Μηχανικής Μάθησης και προσπαθεί να μιμηθεί την ανθρώπινη νοημοσύνη μέσα από το συνδυασμό δεδομένων, βαρών (weights) και της προκατάληψης (bias). Η Βαθιά Μάθηση διαφοροποιείται από τη Μηχανική Μάθηση βάσει του τύπου δεδομένων που επεξεργάζεται όσο και βάσει των μεθόδων που χρησιμοποιεί για να μάθει. Μέσω ειδικών αλγορίθμων, η Βαθιά Μάθηση, χρησιμοποιώντας ακατέργαστα, μη δομημένα δεδομένα (raw unstructured data), όπως φωτογραφίες, κείμενα, βίντεο κ.α., αυτοματοποιεί τη διαδικασία για εξαγωγή χαρακτηριστικών, χωρίς την παρέμβαση του ανθρώπινου παράγοντα. Για παράδειγμα, ας υποθέσουμε ότι έχουμε ένα set από φωτογραφίες κατοικιδίων (σκύλος, γάτα, hamster). Στο σύστημα Μηχανικής Μάθησης χρειάζεται ένας ειδικός, ο οποίος θα εξάγει την ιεραρχία των χαρακτηριστικών που διαφοροποιούν τα κατοικίδια (αυτιά, μύτη, πόδια κ.α.) για την αναγνώρισή τους. Από την άλλη, ένας αλγόριθμος Βαθιάς Μάθησης μπορεί να καταλάβει και να εξάγει τα χαρακτηριστικά αυτά αυτόματα. Τόσο η Μηχανική Μάθηση όσο και η Βαθιά Μάθηση κατηγοριοποιούνται σε:

a) Supervised Learning (Επιβλεπόμενη Μάθηση): Γίνεται κατηγοριοποίηση και τα δεδομένα υπόκεινται σε Labeling. (δράση του ανθρώπινου παράγοντα).

b) Unsupervised Learning (Μη Επιβλεπόμενη Μάθηση): Δεν απαιτείται η κατηγοριοποίηση και το Labeling. Ο αλγόριθμος είναι ικανός να εξάγει τα μοτίβα (patterns) από τα δεδομένα.

c) Reinforcement Learning (Ενισχυτική Μάθηση): Αυτού του είδους τα μοντέλα έχουν σκοπό να πετύχουν περισσότερη ακρίβεια/αποτελεσματικότητα αλληλεπιδρώντας με ένα συγκεκριμένο περιβάλλον στηριζόμενα στην ανατροφοδότησγη (feedback) με σκοπό την μεγιστοποίηση ενός επάθλου/σκορ. Απλό παράδειγμα Reinforcement Learning συστήματος

αποτελεί ένα μοντέλο που τρέχει πάνω σε ένα παιχνίδι (πίστα Super Mario) για να πετύχει όσο το δυνατόν καλύτερο σκορ.

Όπως αναφέραμε προηγουμένως, η Βαθιά Μάθηση προσπαθεί να μιμηθεί τον τρόπο με τον οποίο λειτουργεί ο ανθρώπινος εγκέφαλος. Αυτό το πετυχαίνει με τη δημιουργία και χρήση νευρωνικών δικτύων (neural networks) αξιοποιώντας και συνδυάζοντας δεδομένα εισόδου (input data), βάρη (weights) και της προκατάληψης (bias), προκειμένου να αναγνωρίσει με ακρίβεια, να ορίσει την τάξη (classify) και να περιγράψει αντικείμενα στα δεδομένα. Τα Βαθιά Νευρωνικά Δίκτυα (Deep Neural Networks-DNN) αποτελούνται από πολλαπλά στρώματα (layers), τα οποία με τη σειρά τους αποτελούνται από ένα σύνολο διασυνδεδεμένων κόμβων. Το καθένα από αυτά τα στρώματα βασίζεται στο προηγούμενο επίπεδο για να βελτιώσει και να βελτιστοποιήσει την πρόβλεψη ή την κατηγοριοποίηση. Η εκτέλεση υπολογισμών από τα αρχικά στρώματα προς τα τελικα στρώματα στο Βαθύ Νευρωνικό Δίκτυο ονομάζεται *forward propagation*. Το *forward propagation* είναι η διαδικασία με την οποία εξάγονται τα επιθυμητά αποτελέσματα. Το στρώμα εισόδου (input layer) και το στρώμα εξόδου (output layer) ονομάζονται ορατά στρώματα (*visible layers)*. Το στρώμα εισόδου είναι το πρώτο στρώμα του Μοντέλου Βαθίας Μάθησης και σε αυτό εισάγονται τα δεδομένα που θα υποστούν επεξεργασία. Το στρώμα εξόδου αποτελεί το τελευταίο στρώμα και σε αυτό γίνεται η πρόβλεψη ή η κατηγοριοποίηση. Μια άλλη διαδικασία που ονομάζεται *backpropagation* , χρησιμοποιεί αλγορίθμους όπως gradient descent (απότομης καθόδου) για να υπολογίσει τα σφάλματα στις προβλέψεις και έπειτα προσαρμόζει τα weights και biases πηγαίνοντας από τα πισω στρώματα προς τα μπρος. Αποτέλεσμα αυτής της διαδικασίας είναι σταδιακά το μοντέλο να εκπαιδεύεται , να ελαχιστοποιεί τα σφάλματα και να πετυχαίνει μεγαλύτερη ακρίβεια. Παραδείγματα Βαθιών Νευρωνικών Δικτύων αποτελούν τα:

i) Συνελικτικά Νευρωνικά Δίκτυα (Convolutional Neural Networks-CNNs): Χρησιμοποιούνται κατα κύριο λόγο σε θέματα Όρασης Υπολογιστών (Computer Vision) και εφαρμογές Ταξινόμησης Εικόνων (Image Classification). Είναι ικανά να εντοπίσουν χαρακτηριστικά και μοτίβα μέσα σε μια φωτογραφία, δίνοντας την ικανότητα της Ανίχνευσης Αντικειμένων (Object Detection ή Recognition).

ii) Αναδρομικά Νευρωνικά Δίκτυα (Recurrent Neural Networks-RNNs): Χρησιμοποιούνται στη φυσική γλώσσα και σε speech recognition εφαρμογές αναγνώρισης ομιλίας (speech recognition apps).

3) Νευρωνικά Δίκτυα (Neural Networks): Τα νευρωνικά δίκτυα πήραν την ονομασία τους από τους νευρώνες του ανθρώπινου εγκεφάλου, διότι η συμπεριφορά τους μιμείται την ανθρώπινη εγκεφαλική δραστηριότητα. Χωρίζονται στις τρεις βασικές κατηγορίες Τεχνητά Νευρωνικά Δίκτυα (Artificial Neural Networks-ANNs), Προσομοιωμένα Νευρωνικά Δίκτυα (Simulated Neural

Networks-SNNs) και Βαθιά Νευρωνικά Δίκτυα (Deep Neural Networks-DNNs).

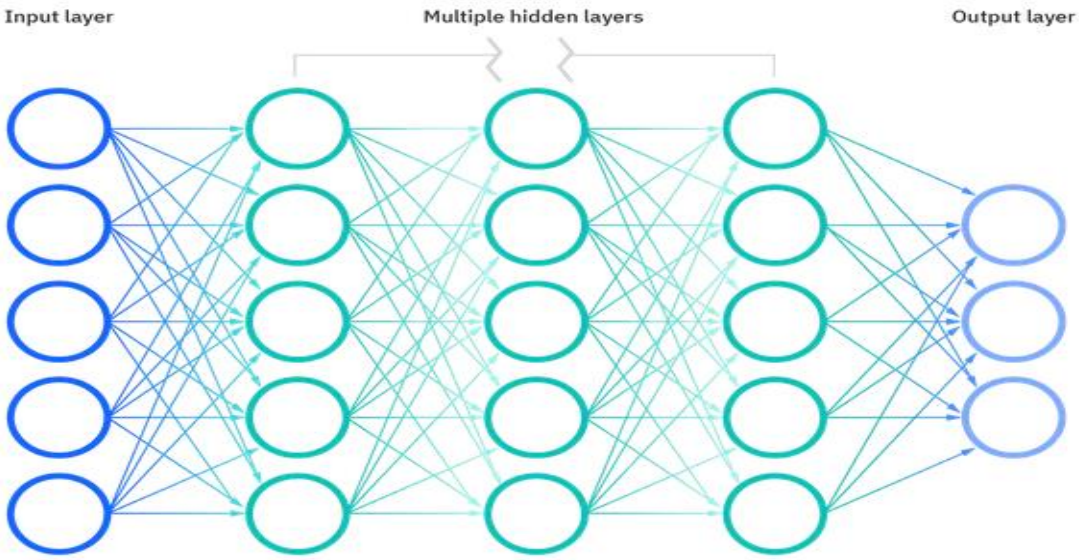


Figure 6:Neural Network Example

Όπως αναφέραμε προηγουμένως τα Νευρωνικά Δίκτυα χωρίζονται σε στρώματα όπως φαίνεται και στο σχήμα. Τα Νευρωνικά Δίκτυα περιέχουν ένα στρώμα εισόδου, ένα ή περισσότερα κρυμμένα στρώματα και τελικά ένα στρώμα εξόδου. Κάθε κόμβος (node) ή αλλιώς τεχνητός νευρώνας (artificial neuron) συνδέεται με άλλους και έχει συγκεκριμένο βάρος (weight) και κατώφλι threshold (threshold). Ο τρόπος με τον οποίο επικοινωνούν τα διάφορα στρώματα θα αποδοθεί αργότερα. Τα Νευρωνικά Δίκτυα στηρίζονται στην εκπαίδευση (training) για να μάθουν και να βελτιώνουν την ακρίβειά τους. Η δημιουργία τέτοιων δικτύων μας βοηθά να προβλέπουμε και να κατηγοριοποιούμε με μεγάλη ταχύτητα. Ένα παράδειγμα Νευρωνικού Δικτύου γνωστό σε όλους αποτελεί ο αλγόριθμος αναζήτησης της Google.

4) Γνωσιακή Υπολογιστική (Cognitive Computing): Είναι ένα υποπεδίο του AI, προσομοιώνει τις διαδικασίες ανθρώπινης σκέψης σε μηχανές, χρησιμοποιώντας αλγορίθμους αυτοεκμάθησης μέσα από την εξόρυξη δεδομένων (data mining), την αναγνώριση προτύπων (pattern recognition) και την επεξεργασία φυσικής γλώσσας (natural language processing). Αυτά τα τεχνητά περιβάλλοντα βασίζονται σε αλγορίθμους Βαθιάς Μάθησης και Νευρωνικά Δίκτυα για την επεξεργασία πληροφοριών, συγκρίνοντάς τες με ένα εκπαιδευτικό σύνολο δεδομένων. Μιμούμενοι τις ανθρώπινες διαδικασίες σκέψης, οι υπολογιστές βοηθούν τους ανθρώπους να λαμβάνουν καλύτερες και ευκολότερες αποφάσεις. Ένα παράδειγμα Cognitive Computing αποτελεί το Google Assistant από την Google.

5) Επεξεργασία Φυσικής Γλώσσας (Natural Language Processing): Μέσω αυτού του υποπεδίου του AI, οι υπολογιστές μπορούν να ερμηνεύουν, να αναγνωρίζουν, να εντοπίζουν και να επεξεργάζονται την ανθρώπινη γλώσσα και ομιλία. Μέσω αυτού οι μηχανές μπορούν να αλληλεπιδρούν με φυσική γλώσσα, είτε σε μορφή κειμένου είτε ομιλίας και να παρέχουν λογικές απαντήσεις στους

ανθρώπους. Το Natural Language Generation (NLG) επεξεργάζεται και αποκωδικοποιεί την ανθρώπινη λεκτική επικοινωνία, ενώ το Natural Language Understanding (NLU) δίνει έμφαση στη γραπτή επικοινωνία. Παραδείγματα τέτοιων εφαρμογών αποτελεί το Google translate (NLU based) και το Google Voice Assistant (NLG based).

6) Όραση Υπολογιστών (Computer Vision): Είναι το πεδίο της Τεχνητής Νοημοσύνης που επιτρέπει σε υπολογιστές και συστήματα να αντλούν σημαντικές πληροφορίες από ψηφιακές εικόνες, βίντεο κ.α. και να προβαίνουν σε ενέργειες ή να κάνουν προτάσεις με βάση αυτές τις πληροφορίες. Αν για παράδειγμα, η Τεχνητή Νοημοσύνη δίνει τη δυνατότητα στους υπολογιστές να σκέφτονται, η Όραση Υπολογιστών τους δίνει τη δυνατότητα να βλέπουν, να παρατηρούν και να κατανοούν το περιβάλλον τους. Η Όραση Υπολογιστών με την αξιοποίηση αλγορίθμων, καμερών και δεδομένων παρέχει δυνατότητες όρασης που ορισμένες φορές ξεπερνούν αυτές των ανθρώπων.

   Η Όραση Υπολογιστών απαιτεί πολλά δεδομένα. Αναλύει επανειλλημμένως τα δεδομένα μέχρι να διακρίνει διαφορές και τελικά να αναγνωρίζει εικόνες. Για παράδειγμα, για να εκπαιδευτεί ένας υπολογιστής, ώστε να αναγνωρίζει ελαστικά αυτοκινήτου, πρέπει να τροφοδοτηθεί με τεράστιο πλήθος εικόνων ελαστικών και αντικειμένων που σχετίζονται με τα ελαστικά για να μάθει τις διαφορές και τελικά να αναγνωρίσει ένα ελαστικό. Για να επιτευχθεί αυτό χρειάζονται δύο τεχνολογίες. Η μία είναι η Βαθιά Μάθηση και η άλλη ένα Συνελικτικό Νευρωνικό Δίκτυο.

   Η Βαθιά Μάθηση χρησιμοποιεί αλγοριθμικά μοντέλα που επιτρέπουν σε έναν υπολογιστή να διδάξει τον εαυτό του όσον αφορά θέματα Όρασης Υπολογιστών. Εάν τροφοδοτηθούν αρκετά δεδομένα μέσω του μοντέλου, ο υπολογιστής θα «κοιτάξει» τα δεδομένα και θα μάθει να ξεχωρίζει τη μια εικόνα από την άλλη. Όλο αυτό γίνεται αυτόματα, χωρίς να χρειάζεται η ανθρώπινη παρέμβαση.

   Ένα Συνελικτικό Νευρωνικό Δίκτυο βοηθά το μοντέλο Βαθιάς Μάθησης μέσω της διάσπασης μιας εικόνας σε pixel, στα οποία έχουν δοθεί tags ή labels. Το Συνελικτικό Νευρωνικό Δίκτυο εκτελεί συνελίξεις και ελέγχει την ακρίβεια των προβλέψεών του με συνεχόμενες επαναλήψεις, μέχρι να αρχίσουν να πραγματοποιούνται οι προβλέψεις. Ένα Συνελικτικό Νευρωνικό Δίκτυο διακρίνει πρώτα άκρες και απλά σχήματα και μετά εμπλουτίζει με νέες πληροφορίες και γνώση. Ένα Συνελικτικό Νευρωνικό Δίκτυο χρησιμοποιείται για την κατανόηση μεμονωμένων εικόνων. Κατά συνέπεια, ο υπολογιστής, με αυτό το εργαλείο, μπορεί να αναγνωρίζει αντικείμενα όπως ο άνθρωπος.

   Η Όραση Υπολογιστών επικεντρώνει το ενδιαφέρον της στις κάτωθι κατηγορίες προβλημάτων:

   a) Image Classification – Ταξινόμηση εικόνων: Το μοντέλο δέχεται σαν είσοδο (input) μια εικόνα και αποδίδει σαν έξοδο (output) την πιθανότητα η εικόνα

να ανήκει σε μία κλάση, στηριζόμενη σε ολόκληρη την εικόνα.

b) Object Detection – Ανίχνευση Αντικειμένων: Το μοντέλο δέχεται σαν είσοδο (input) μια εικόνα και αποδίδει σαν έξοδο (output) την κλάση και την θέση των αντικειμένων που βρίσκονται μέσα στην εικόνα.

c) Semantic Segmentaition – Σημασιολογική Κατάτμηση εικόνων: Το μοντέλο δέχεται σαν είσοδο (input) μια εικόνα και αποδίδει σαν έξοδο (output) την κλάση για κάθε εικονοστοιχείο.

d) Instance Segmentation – Κατάτμηση Στιγμιοτύπων εικόνων: Το μοντέλο δέχεται σαν είσοδο (input) μια εικόνα και αποδίδει σαν έξοδο (output) τα αντικείμενα και την κλάση τους σε επίπεδο εικονοστοιχείου. Άρα συνδιάζει το Object Detection και το Semantic Segmentation σαν προβλήματα.

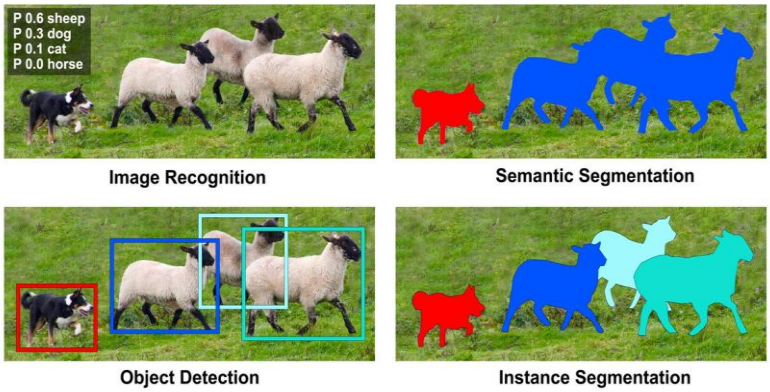


Figure 7: Κατηγορίες προβλημάτων Computer Vision

Το Computer Vision έχει αρχίσει και αποκτά ιδιαίτερο ενδιαφέρον καθώς έχει πολλές εφαρμογές στην καθημερινή ζωή. Ορισμένες από αυτές είναι τα αυτόοδηγούμενα αυτοκίνητα, αναγνώριση ανθρώπων σε συστήματα ασφαλείας, αναγνώριση ιατρικών χαρακτηριστικών κ.α.

Κεφάλαιο 2 : Object Detection (Ανίχνευση Αντικειμένων)

2.0 – Εισαγωγή

Η Αναγνώριση Αντικειμένου (Object Detection) είναι ένα υποπεδίο της Όρασης Υπολογιστών (Computer Vision) που μας επιτρέπει να αναγνωρίζουμε και να εντοπίζουμε αντικείμενα μέσα από μια φωτογραφία ή ένα βίντεο. Το ειδικό χαρακτηριστικό σχετικά με την Αναγνώριση Αντικειμένου είναι ότι προσδιορίζει την κλάση (άνθρωπος, σκύλος, γάτα κλπ) του αντικειμένου και τις ακριβείς συντεταγμένες του στη δεδομένη εικόνα ή βίντεο. Η θέση επισημαίνεται σχεδιάζοντας ένα πλαίσιο οριοθέτησης (bounding box) γύρω από το αντικείμενο. Η ικανότητα να εντοπίζεται σωστά και με ακρίβεια το αντικείμενο προσδιορίζει την απόδοση του αλγορίθμου που χρησιμοποιήθηκε για την Αναγνώριση Αντικειμένου.

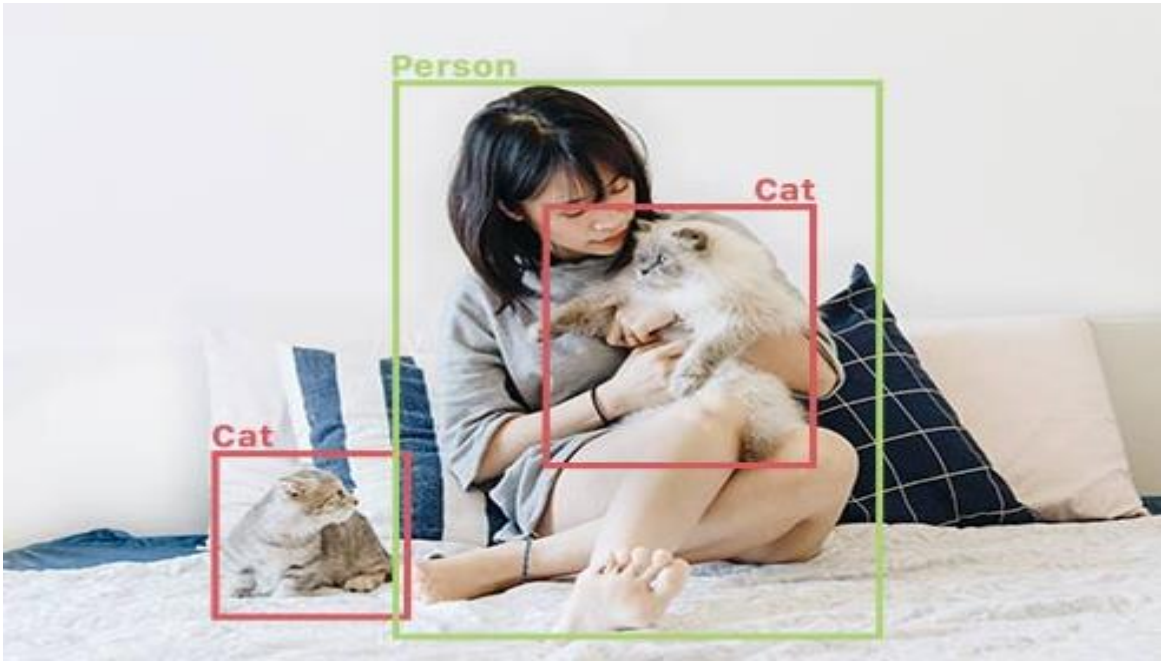


Figure 8: Object Detection Example

Η Αναγνώριση Αντικειμένων μπορεί να χρησιμοποιεί προεκπαιδευμένα αλγοριθμικά μοντέλα (pre-trained models) ή μοντέλα τα οποία δημιουργούνται από το μηδέν. Η δημιουργία ενός αλγοριθμικού μοντέλου αναγνώρισης αντικειμένου είναι μια πολύ δύσκολη και χρονοβόρα διαδικασία και για αυτό το λόγο δημιουργείται από ειδικούς σε θέματα Όρασης Υπολογιστών. Στη παρούσα διπλωματική εργασία εστιάζουμε στην ανάπτυξη ενός Προσαρμοσμένου Ανιχνευτή Αντικειμένων (Custom Object Detector) χρησιμοποιώντας έτοιμα προεκπαιδευμένα μοντέλα.

2.1 – Μέθοδοι της Ανίχνευσης Αντικειμένων

Οι μέθοδοι με τις οποίες μπορεί να γίνει η Αναγνώριση Αντικειμένων είναι είτε στηριζόμενες σε Νευρωνικά Δίκτυα (neural networks) ή ανεξάρτητες από αυτά. Σε αυτές που είναι ανεξάρτητες από τα νευρωνικά δίκτυα αρχικά πρέπει να καθοριστούν τα χαρακτηριστικά και κατόπιν χρησιμοποιούνται τεχνικές όπως, τα Support Vector Machines (SVM) για να γίνει η κατηγοριοποίηση. Από την άλλη, οι μέθοδοι στηριζόμενες σε Νευρωνικά Δίκτυα είναι ικανές να κάνουν μία end-to-end Ανίχνευση Αντικειμένων, χωρίς να χρειάζεται ή να απαιτείται ο προσδιορισμός των χαρακτηριστικών και συνήθως χρησιμοποιούν Συνελικτικά Νευρωνικά Δίκτυα.

Μερικές Non-neural προσεγγίσεις είναι οι εξής:

- Viola-Jones object detection framework βασισμένη σε Haar features.
- Scale-invariant feature transform (SIFT)
- Histogram of oriented gradients (HOG) features

Μερικές Neural-network προσεγγίσεις είναι οι εξής:

- Region Proposals (R-CNN, Fast R-CNN, Faster R-CNN, cascade R-CNN)
- Single Shot MultiBox Detector (SSD)
- You Only Look Once (YOLO)
- Single-Shot Refinement Neural Network for Object Detection (RefineDet)
- Retina-Net

2.2 - Τρόπος λειτουργίας της Ανίχνευσης Αντικειμένων

Γενικά ο τρόπος λειτουργίας της Ανίχνευσης Αντικειμένων γίνεται σε τρεις φάσεις:

- Η είσοδος «κόβεται» σε μικρά τμήματα, όπως φαίνεται στην παρακάτω εικόνα. Όπως βλέπουμε το τεράστιο πλήθος από τα μικρά τμήματα καλύπτει ολόκληρη την εικόνα.

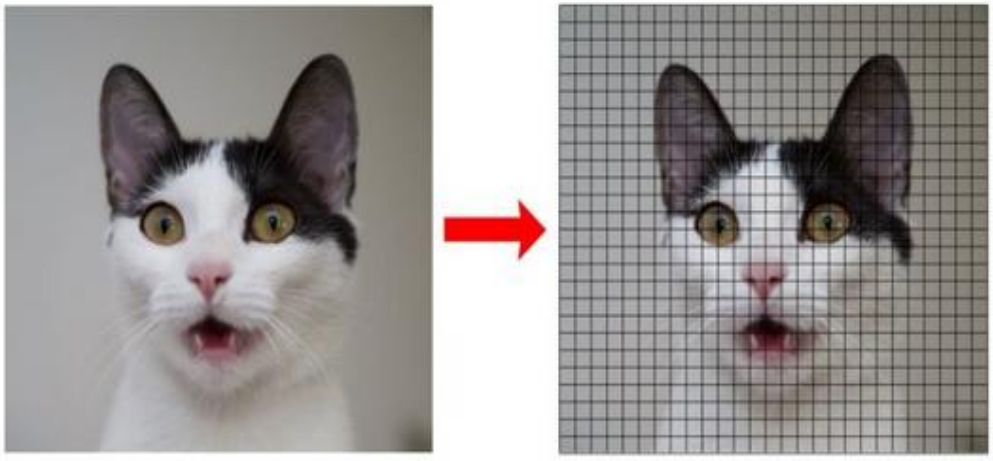


Figure 9: Step 1 of Object Detection

- Έπειτα εκτελείται εξαγωγή των features για κάθε τμηματοποιημένη ορθογώνια περιοχή για να προβλεφθεί αν το ορθογώνιο περιέχει ένα έγκυρο αντικείμενο.

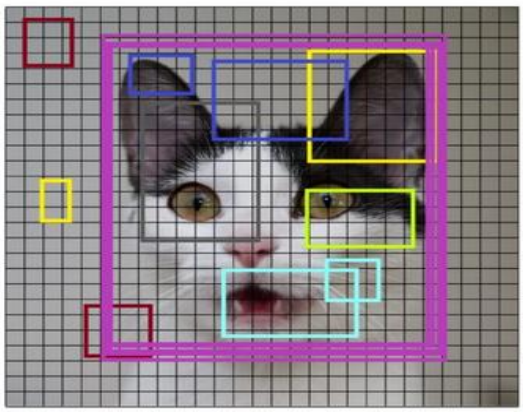


Figure 10: Step 2 of Object Detection

- Τα επικαλυπτόμενα πλαίσια (overlapping boxes) συνδυάζονται σε ένα ενιαίο πλαίσιο οριοθέτησης, όπως φαίνεται στο σχήμα καθώς αποδίδεται και η κλάση του πλαισίου οριοθέτησης.

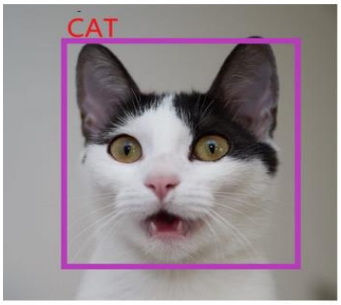


Figure 11: Final Prediction of Object Detection

2.3 - Πρακτικές εφαρμογές της Ανίχνευσης Αντικειμένων

Σε αυτήν την ενότητα θα παρουσιάσουμε πρακτικές εφαρμογές της Ανίχνευσης Αντικειμένων. Αν και έχουμε αναφέρει αρκετές εφαρμογές προηγουμένως, στόχος μας είναι να τις αναλύσουμε περισσότερο και να εξερευνήσουμε τον αντίκτυπο της Ανίχνευσης Αντικειμένων σε όλους τους κλάδους.

Οι εφαρμογές που θα αναλύσουμε είναι:

1. Βιντεοπαρακολούθηση:

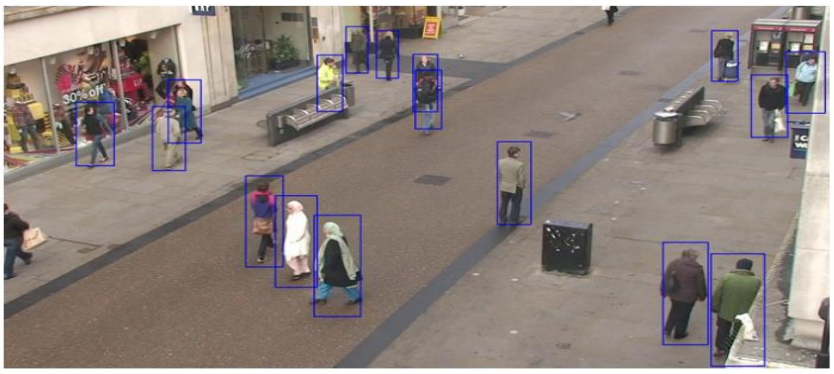


Figure 12: Βιντεοπαρακολούθηση

Η Ανίχνευση Αντικειμένων μπορεί να αναγνωρίζει, να ανιχνεύει με ακρίβεια πολλαπλές εμφανίσεις ενός αντικειμένου. Αυτό συνετέλεσε στο να ενσωματωθεί σε αυτόματα συστήματα παρακολούθησης μέσω κάμερας. Τα μοντέλα Ανίχνευσης Αντικειμένων είναι ικανά να παρακολουθούν πολλαπλά άτομα ταυτόχρονα, σε πραγματικό χρόνο, καθώς κινούνται σε μια καθορισμένη σκηνή (ορατή από την κάμερα). Από καταστήματα μέχρι εργοστασιακές εγκαταστάσεις, αυτού του είδους η λεπτομερής παρακολούθηση μπορεί να προσφέρει ανεκτίμητες δυνατότητες όπως η φύλαξη χώρων, η απόδοση και ασφάλεια εργαζομένων και άλλα.

2. Καταμέτρηση πλήθους (Crowd counting)

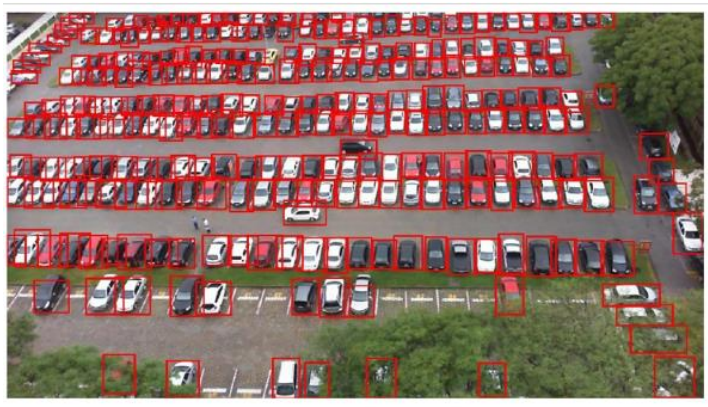


Figure 13: Crowd Counting

Άλλη μια σημαντική εφαρμογή της Ανίχνευσης Αντικειμένων είναι η καταμέτρηση του πλήθους των αντικειμένων. Για πυκνοκατοικημένες περιοχές όπως θεματικά πάρκα, εμπορικά κέντρα και πλατείες πόλεων, η Ανίχνευση Αντικειμένων μπορεί να βοηθήσει επιχειρήσεις και δήμους να μετρήσουν πιο αποτελεσματικά τα διαφορετικά είδη κίνησης (είτε με τα πόδια, με οχήματα ή με άλλο τρόπο). Αυτού του είδους η πληροφορία μπορεί να βοηθήσει τις επιχειρήσεις να βελτιστοποιήσουν οποιαδήποτε λειτουργία επιθυμούν, όπως διαχείριση αποθέματος και προγραμματισμός παραγωγής. Σε επίπεδο δημοτικής οργάνωσης, τα μοντέλα Ανίχνευσης Αντικειμένων μπορούν να συνδράμουν στην αντιμετώπιση προβλημάτων, όπως αυτό της κυκλοφοριακής συμφόρησης σε δρόμους και της διαθεσιμότητας ελεύθερων χώρων στάθμευσης.

3. Ανίχνευση Ανωμαλιών (Anomaly Detection)

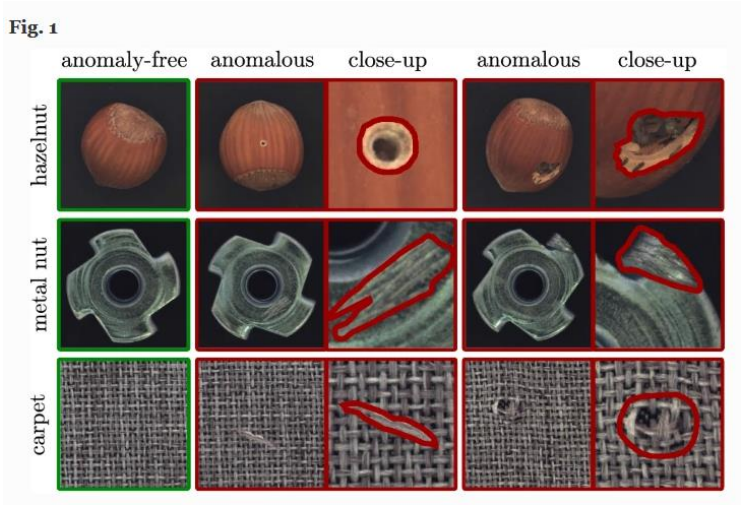


Figure 14: Anomaly Detection

Η ανίχνευση ανωμαλιών αποτελεί μια ακόμα σπουδαία εφαρμογή της Ανίχνευσης Αντικειμένων. Κατά κύριο λόγο, χρησιμοποιείται τόσο στην επιστήμη της Γεωργίας όσο και της Ιατρικής. Από την πλευρά της Γεωργίας ένα μοντέλο Ανίχνευσης Αντικειμένων μπορεί να ανιχνεύσει αν το φυτό νοσεί ή έχει κάποια ανωμαλία και να βοηθήσει τον γεωργό να το θεραπεύσει, συμβάλλοντας στην αύξηση της παραγωγής του και να προστατεύσει την καλλιέργειά του. Το ίδιο ισχύει και για τον κλάδο της Ιατρικής, όπου μπορούν να ανιχνευθούν ανωμαλίες με σκοπό αντιμετωπιστούν το συντομότερο δυνατόν. Αυτό επιτυγχάνεται μέσω καμερών υψηλής ανάλυσης, οι οποίες ξεπερνούν κατά πολύ τις δυνατότητες του γυμνού οφθαλμού.

4. Αυτοοδηγούμενα αυτοκίνητα (Self-Driving cars)

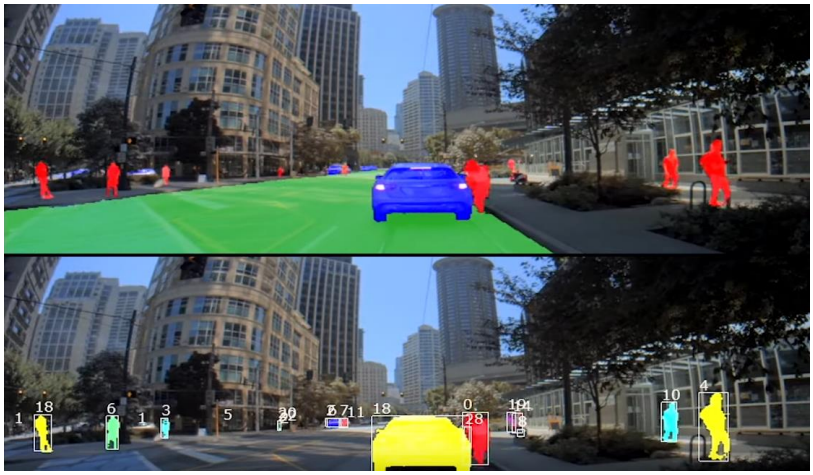


Figure 15: Self-Driving Cars

Η ανίχνευση σε πραγματικό χρόνο (real-time detection) είναι το σημαντικότερο στοιχείο των αυτοοδηγούμενων αυτοκινήτων. Αυτά τα οχήματα μέσω της Ανίχνευσης Αντικειμένων μπορούν να αναγνωρίζουν, να εντοπίζουν και παρακολουθούν αντικείμενα στο περιβάλλον τους. Αυτό έχει σαν αποτέλεσμα αυτού του είδους τα οχήματα να είναι πιο ασφαλή σε σχέση με τα συμβατικά αυτοκίνητα. Αυτό σημαίνει πρακτικά ότι ένα δίκτυο αποτελούμενο από αυτόνομα αυτοκίνητα θα μπορούσε να είναι πιο ασφαλές και γρήγορο σε σχέση με τα υπάρχοντα οδικά δίκτυα.

Κεφάλαιο 3 : Θεωρητικό Υπόβαθρο

3.0 – Εισαγωγή

Σκοπός αυτού του κεφαλαίου είναι να πληροφορηθούμε και να κατανοήσουμε τις τεχνολογίες που χρησιμοποιούνται από το προεκπαιδευμένο (pre-trained) μοντέλο που χρησιμοποιήθηκε για τη δημιουργία του Προσαρμοσμένου Ανιχνευτή Αντικειμένων σε πραγματικό χρόνο (real-time Custom Object Detector). Θα αναφέρουμε τον τρόπο λειτουργίας και τα στοιχεία από τα οποία αποτελείται ένα Νευρωνικό Δίκτυο. Έπειτα, θα αναφερθούμε στα Συνελικτικά Νευρωνικά Δίκτυα. Τέλος, θα παρουσιαστεί το προεκπαιδευμένο μοντέλο που χρησιμοποιούμε για να κάνουμε ανίχνευση και κατηγοριοποίηση.

3.1 – Επιβλεπόμενη Μάθηση – Supervised Learning

Όπως έχουμε αναφέρει προηγουμένως, η Επιβλεπόμενη Μάθηση είναι η υποκατηγορία της Μηχανικής Μάθησης. Χαρακτηρίζεται από τη χρήση δεδομένων με ετικέτες (labeled data). Στόχος είναι να βρεθεί το μοντέλο που θα κάνει είτε προβλέψεις, είτε κατηγοριοποίηση και θα ελαχιστοποιεί την συνάρτηση κόστους. Ο Προσαρμοσμένος Ανιχνευτής Αντικειμένων που θα υλοποιήσουμε ανήκει σε αυτήν την κατηγορία. Οι προγραμματιστές που χρησιμοποιούν αυτόν τον τύπο Μηχανικής Μάθησης είναι απαραίτητο να προεπεξεργαστούν τα δεδομένα τους. Ύστερα τα δεδομένα χωρίζονται σε:

1. Δεδομένα Εκπαίδευσης (Training data): Είναι τα δεδομένα, τα οποία μέσω της διαδικασίας «μάθησης» εξάγουν το τελικό επιθυμητό μοντέλο.

2. Δεδομένα Αξιολόγησης (Evaluation data): Είναι τα δεδομένα, τα οποία χρησιμοποιούνται στο τελικό εξαχθέν μοντέλο, προκειμένου να αξιολογηθεί και να μετρηθεί η απόδοσή του.

Συνήθως τα training data αποτελούν το 80%-90% του γενικού πλήθους των δεδομένων, ενώ τα evaluation data είναι το υπόλοιπο 10%-20%. Στόχος του μοντέλου που θα προκύψει είναι να αφομιώσει τη μορφολογική δομή των δεδομένων μέσα από την διαδικασία μάθησης, ώστε να κάνει λογικές-ορθές γενικεύσεις. Το τελικό μοντέλο που μπορεί να προκύψει μπορεί να υπάγεται σε μία από τις παρακάτω κατηγορίες:

1. Underfitting: Το μοντέλο εμφανίζει μεγάλη απόκλιση στα training data. Στη συγκεκριμένη περίπτωση το μοντέλο είναι αρκετά απλοποιημένο και δεν μπορεί να βρει μια συσχέτιση ανάμεσα στα δεδομένα εισόδου που του δίνονται και στα αποτελέσματα που αποσκοπεί να πετύχει.

2. Overfitting: Το μοντέλο συμπεριφέρεται αρκετά καλά στα training data, αλλά δεν πετυχαίνει τα επιθυμητά αποτελέσματα στα evaluation data. Συνήθως ένα

τέτοιο μοντέλο μαθαίνει από τα training data και το «θόρυβο» που εμπεριέχουν. Ένα overfitting μοντέλο χαρακτηρίζεται από μεγάλη διακύμανση.

3. Balanced: Το μοντέλο αυτό συμπεριφέρεται καλά με ελάχιστες αποκλίσεις τόσο στα training όσο και στα evaluation data. Είναι το αποτέλεσμα που προσπαθούμε να πετύχουμε. Ένα τέτοιο μοντέλο χαρακτηρίζεται από μια ισορροπημένη σχέση ανάμεσα στην απόκλιση και τη διακύμανση και είναι ικανό να κάνει γενικεύσεις.

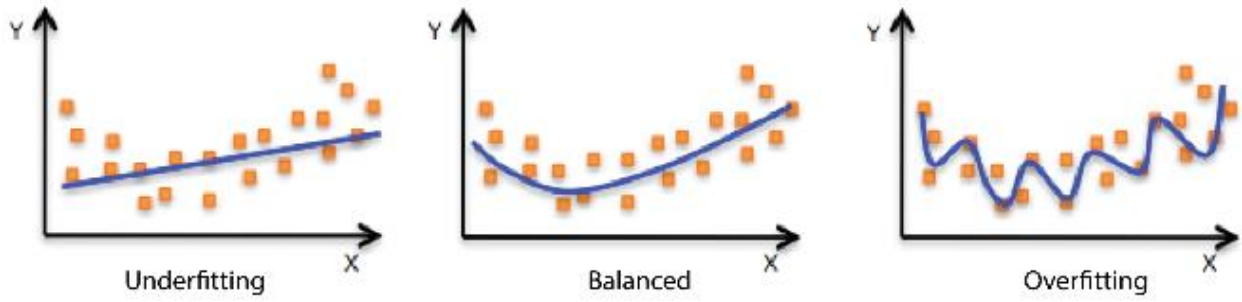


Figure 16: Underfitting, Overfitting, Balanced Models

3.2 - Νευρωνικά Δίκτυα

3.2.1 - Δομή και λειτουργία ενός Νευρωνικού Δικτύου

Τα Νευρωνικά Δίκτυα όπως έχουμε αναφέρει προηγουμένως μιμούνται τον τρόπο που λειτουργεί ο ανθρώπινος εγκέφαλος. Ένα Νευρωνικό Δίκτυο αποτελείται από ένα στρώμα εισόδου (input layer) ακολουθούμενο από ένα ή περισσότερα κρυφά στρώματα (hidden layers) και καταλήγει σε ένα στρώμα εξόδου (output layer). Το Νευρωνικό Δίκτυο περιέχει διασυνδεδεμένους κόμβους μεταξύ τους, οι οποίοι ονομάζονται νευρώνες. Σκοπός των νευρώνων είναι να επεξεργάζονται την πληροφορία που τους δίνεται σαν input και να «περνούν» νέες πληροφορίες στο επόμενο στρώμα για να εξαχθεί τελικά το επιθυμητό αποτέλεσμα. Η δομή ενός πλήρως συνδεδεμένου Νευρωνικού Δικτύου δίνεται σχηματικά παρακάτω.

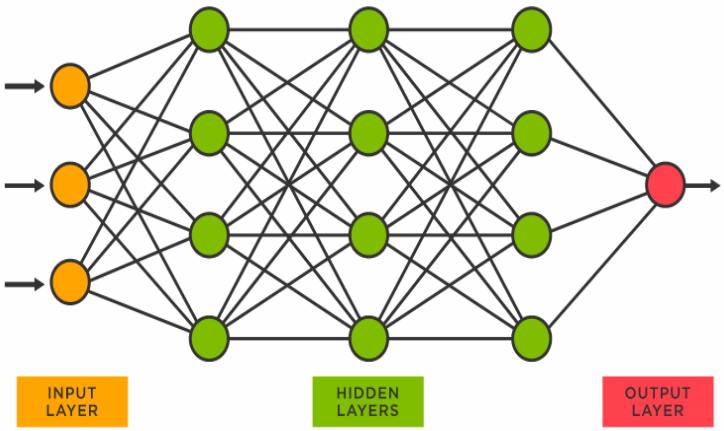


Figure 17:Πλήρως συνδεδεμένο Νευρωνικό Δίκτυο

Το στρώμα εισόδου είναι αυτό που δέχεται τα ακατέργαστα δεδομένα (raw data) και αποτελεί πάντα το πρώτο στρώμα. Κάθε νευρώνας-κόμβος δέχεται input από κόμβους του προηγούμενου στρώματος, με τους οποίους συνδέεται και στέλνει output στους κόμβους του επόμενου στρώματος, με τους οποίους συνδέεται. Μπορεί να υπάρχουν και κόμβοι σε ένα νευρωνικό δίκτυο, οι οποίοι δεν παιρνουν input από προηγούμενο στρώμα. Αυτοί ερμηνεύονται σαν bias (προκατάληψη).

Ο τρόπος με τον οποίο λειτουργεί το Νευρωνικό Δίκτυο είναι ο εξής: Το στρώμα εισόδου δέχεται δεδομένα. Τα δεδομένα αυτά υπόκεινται σε επεξεργασία στο στρώμα εισόδου και όταν η επεξεργασία τελειώσει, η παραγόμενη έξοδος αποδίδεται σαν input για το πρώτο κρυφό στρώμα. Έπειτα, το πρώτο κρυφό στρώμα, αφού τελειώσει την επεξεργασία δεδομένων περνάει την παραγόμενη έξοδο σαν input στο δεύτερο κρυφό στρώμα. Η διαδικασία συνεχίζεται με παρόμοιο τρόπο μέχρι να παραχθεί το επιθυμητό αποτέλεσμα στο στρώμα εξόδου.

Αξίζει να σημειωθεί ότι το στρώμα εισόδου μπορεί να έχει διαφορετικό τύπο από το στρώμα εξόδου. Αυτό σημαίνει ότι το πρώτο μπορεί να είναι μια φωτογραφία, ενώ το δεύτερο μπορεί να είναι ένα ποσοστό που δηλώνει πόσο σίγουρο είναι το Νευρωνικό Δίκτυο ότι ένα αντικείμενο της φωτογραφίας ανήκει σε μια τάξη (γάτα, σκύλος κλπ).

3.2.2 - Δομή και λειτουργία ενός νευρώνα

Προηγουμένως παρουσιάσαμε πως λειτουργεί ένα Νευρωνικό Δίκτυο συνολικά. Σκοπός τώρα είναι να δούμε τη λειτουργία και τη δομή σε επίπεδο νευρώνα. Η δομή ενός νευρώνα έχει την παρακάτω μορφή.

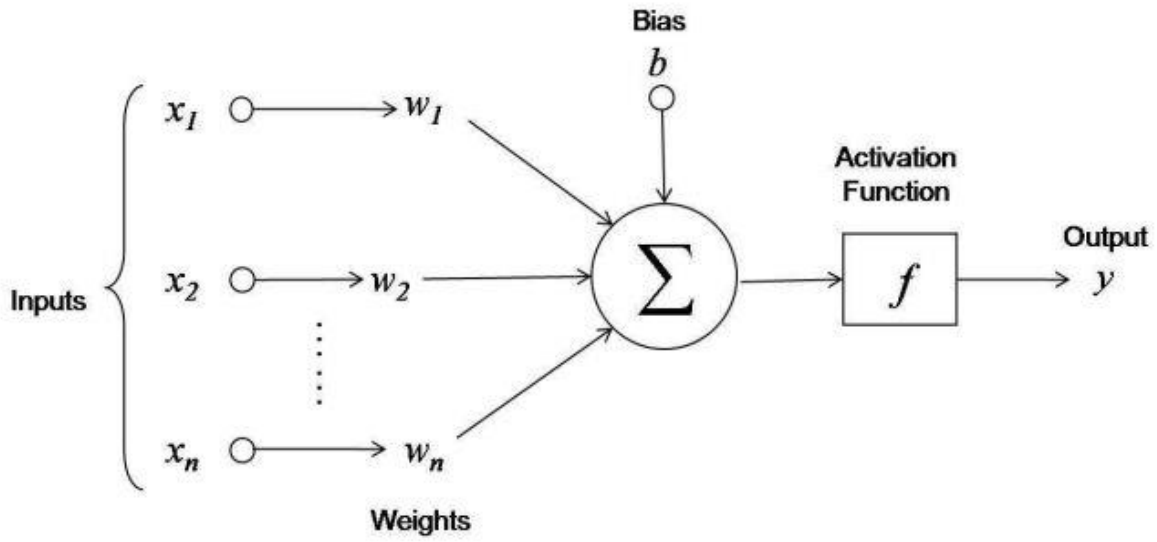


Figure 18: Δομή ενός νευρώνα

Κάθε νευρώνας δέχεται ένα input, στην περίπτωσή μας x_1, x_2, …, x_n. Για καθένα x_i ο νευρώνας διαθέτει ένα αντίστοιχο βάρος w_i. Η διαδικασία που επιτελεί ο νευρώνας είναι αρχικά να υπολογίζει το σταθμισμένο άθροισμα ($x_1w_1+x_2w_2+...+x_nw_n$), έπειτα σε αυτό προσθέτει το bias (αν υπάρχει) και η τελική τιμή περνά σαν όρισμα μέσα από μία συνάρτηση ενεργοποίησης (activation function). Το αποτέλεσμα της συνάρτησης ενεργοποίησης αποτελεί το output του νευρώνα και όπως προαναφέραμε, αποτελεί input για τους νευρώνες του επομένου στρώματος, με τους οποίους συνδέεται. Τα βάρη w_i για την κάθε μεταβλητή x_i είναι διαφορετικά. Η σημασία του βάρους w_i είναι το πόσο σημαντική θεωρείται η μεταβλητή x_i. Τα βάρη w_i και το bias λαμβάνουν και ανανεώνουν τις τιμές τους από τη διαδικασία εκπαίδευσης του νευρωνικού δικτύου.

3.2.3 – Συναρτήσεις Ενεργοποίησης (Activation functions)

Μια συνάρτηση ενεργοποίησης αποφασίζει αν ένας νευρώνας πρέπει να «ενεργοποιηθεί» ή όχι. Με άλλα λόγια η συνάρτηση ενεργοποίησης αποφασίζει αν το input ενός νευρώνα είναι σημαντικό ή όχι για το ευρύτερο Νευρωνικό Δίκτυο χρησιμοποιώντας μαθηματικούς τύπους και συναρτήσεις. Η σημαντικότητα αυτή προσδιορίζεται με την τιμή που παράγει η συνάρτηση ενεργοποίησης. Χωρίς αυτές τις συναρτήσεις το νευρωνικό δίκτυο ανεξάρτητα από το πλήθος των στρωμάτων θα μπορούσε να αναπαρασταθεί από μια γραμμική συνάρτηση κάτι που δεν βολεύει στις περισσότερες περιπτώσεις, διότι τα περισσότερα φυσικά φαινόμενα δεν έχουν «τελείως» γραμμική συμπεριφορά. Επιπλέον αξίζει να σημειωθεί ότι αυτές οι συναρτήσεις για την εκπαίδευση του δικτύου πρέπει να είναι διαφορίσιμες. Οι συναρτήσεις ενεργοποίησης χωρίζονται σε γραμμικες και μη γραμμικες. Οι γραμμικές συναρτήσεις είναι αρκετά απλές και πλέον έχουν αντικατασταθεί από τις μη-γραμμικές.

Οι πιο διαδεδομένες activation functions είναι:

1. Μοναδιαία Βηματική Συνάρτηση (Binary Step Function)

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

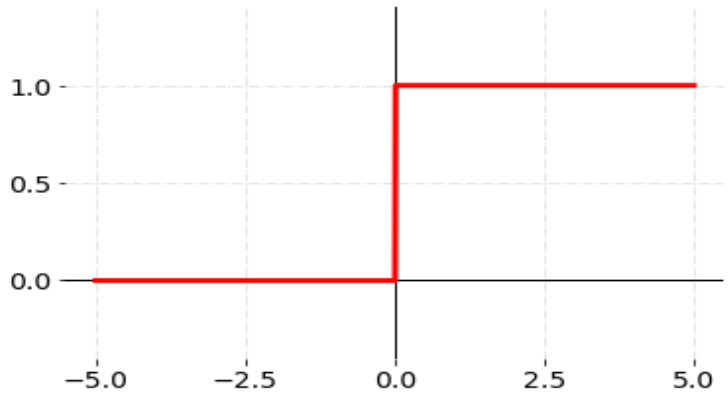


Figure 19: Binary step function

Η Binary step fuction αποτελεί συνάρτηση που βασίζεται σε ένα κατώφλι (threshold), που σημαίνει ότι αν το σταθμισμένο άθροισμα βρίσκεται πάνω από το κατώφλι ενεργοποιείται ο εκάστοτε νευρώνας αλλίως απενεργοποιείται. Στη παραπάνω περίπτωση το κατώφλι είναι 0. Αυτή η συνάρτηση χρησιμοποιείται για δυαδική ταξινόμηση (binary classification) και δεν μπορεί να χρησιμοποιηθεί σε ταξινόμηση πολλαπλών τάξεων.

2. Γραμμική Συνάρτηση (Linear function)

$$f(x) = x$$

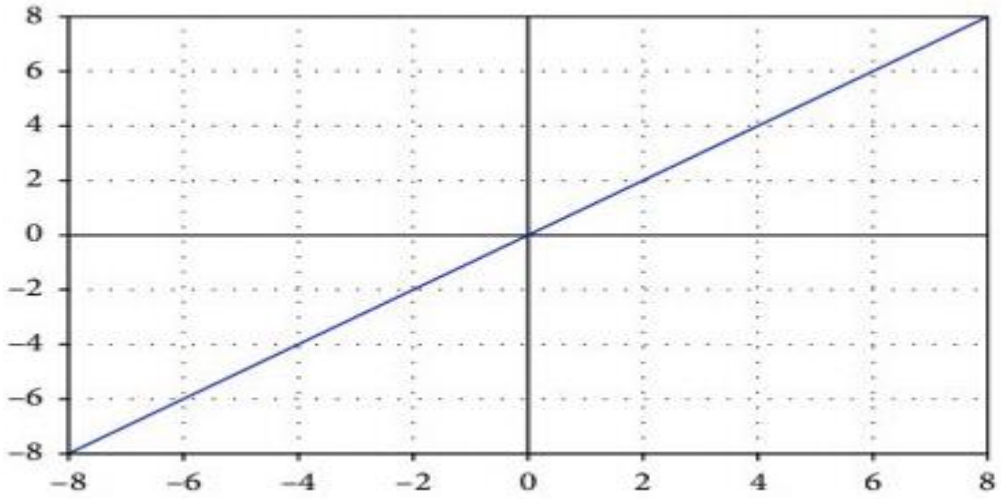


Figure 20: Linear function

Το output της Γραμμικής Συνάρτησης είναι το σταθμισμένο άθροισμα. Αυτή η συνάρτηση μπορεί να κάνει ταξινόμηση πολλαπλών τάξεων όμως παρουσιάζει αρκετά θέματα. Κατά πρώτον, δεν βοηθά στην εκπαίδευση, αφού η παράγωγος είναι η σταθερή συνάρτηση και δεν έχει καμία σχέση με την μεταβλητή x. Με την χρήση αυτής της συνάρτησης σαν συνάρτηση ενεργοποίησης όλα τα στρώματα του κρυφού στώματος δεν έχουν νόημα και το Νευρωνικό Δίκτυο μπορεί να αντικατασταθεί με Νευρωνικό Δίκτυο με μόνο ένα στρώμα. Αυτή η συνάρτηση δεν βοηθά το Νευρωνικό Δίκτυο να αντιμετωπίζει σύνθετα προβλήματα.

3. Σιγμοειδής/Λογιστική Συνάρτηση (Sigmoid/Logistic Function)

$$f(x) = \frac{1}{1 + e^{-x}}$$

Είναι από τις πιο διαδεδομένες συναρτήσεις ενεργοποίησης και χρησιμοποιείται σε πολλές περιπτώσεις. Έχει την ικανότητα να κανονικοποιεί το output κάθε νευρώνα. Απευθύνεται κυρίως σε μοντέλα για να προβλέψει πιθανότητα, αφού το αποτέλεσμα της συνάρτησης είναι στο διάστημα 0 με 1. Το μοναδικό πρόβλημα που εντοπίζεται σε αυτή τη συνάρτηση είναι η

συμπεριφορά της για πολύ μεγάλες και πολύ μικρές τιμές.

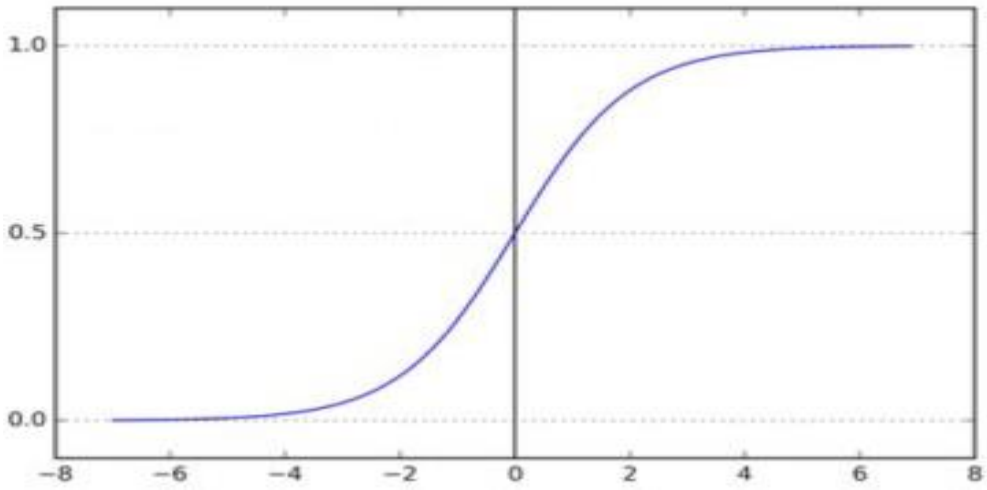


Figure 21: Sigmoid function

4. Υπερβολική Εφαπτομένη (Tanh function)

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

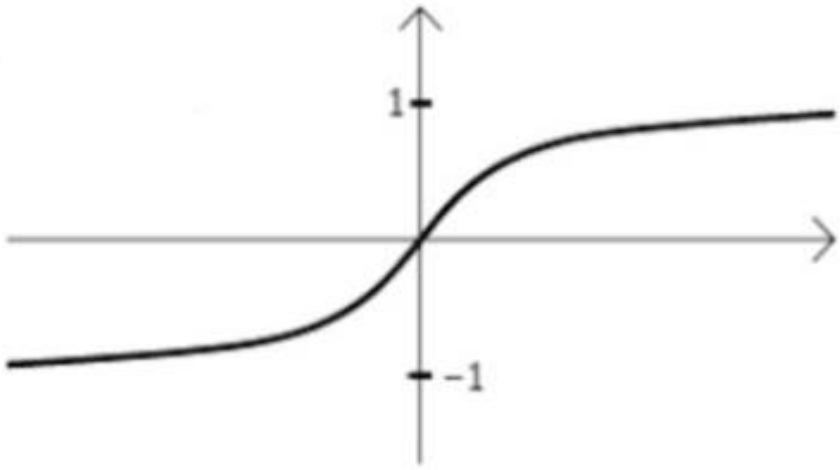


Figure 22: Tahn function

Η συνάρτηση αυτή έχει παρόμοιες ιδιότητες με την προηγούμενη. Το εύρος τιμών είναι από το -1 εώς το 1 δηλαδή κεντραρισμένο ως προς το μηδεν. Αυτή η ιδιότητα βοηθάει στο κεντράρισμα των δεδομένων.

5. Συνάρτηση Διορθωμένης Γραμμικής Μονάδας (ReLU (Rectified Linear Unit) Function)

$$f(x) = max(0, x)$$

Αν και μας δίνει την εντύπωση ότι μοιάζει με τη Γραμμική Συνάρτηση η ReLU έχει παράγωγο και μας επιτρέπει τις διαδικασίες εκπαίδευσης, ενώ παράλληλα είναι υπολογιστικά καλύτερη από τις Sigmoid και Tanh. Δεν ενεργοποιεί όλους τους νευρώνες, παρά μόνο αυτούς που είναι μεγαλύτεροι από το μηδέν. Ο μόνος περιορισμός είναι η παράγωγος γύρω από το 0.

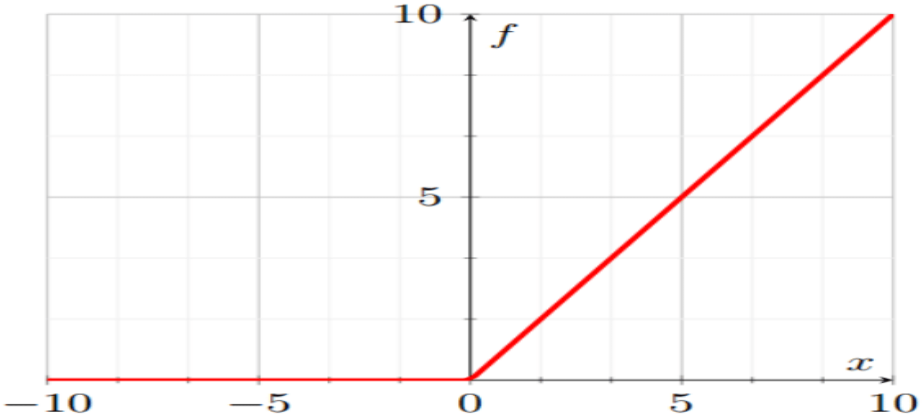


Figure 23: ReLu function

6. Parametric ReLU Function

$$f(x) = max(x, a * x), \qquad 0 < a \leq 1$$

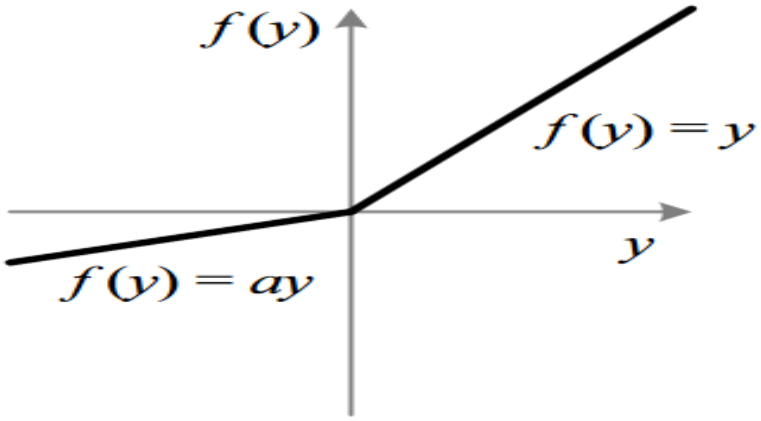


Figure 24: Parametric ReLU function

Είναι μια διαφοροποίηση της ReLU. Είναι η πιο γενική περίπτωση της ReLU και χρησιμοποιείται ευρέως σαν συνάρτηση ενεργοποίησης. Η γραφικής της αναπαράσταση φαίνεται παραπάνω. Τόσο η ReLU όσο και η Parametric ReLU χρησιμοποιούνται πολύ συχνά στους νευρώνες του κρυφού στρώματος.

7. Softmax activation function

$$\sigma(\boldsymbol{x})_i = \frac{e^{x_i}}{\sum_{i=1}^{N} e^{x_i}}, for\ i = 1, \dots, N\ and\ \boldsymbol{x} = (x_1, \dots, x_n) \in \mathbb{R}^N$$

Η συνάρτηση δέχεται ένα διάνυσμα τάξης N πραγματικών αριθμών και το

κανονικοποιεί, παράγοντας Ν πιθανότητες, αφού το σύνολο τιμών που παίρνει φαίνεται ότι είναι το (0,1).

Η συνάρτηση Softmax χρησιμοποιείται στο τελευταίο στρώμα και ο λόγος είναι για να κάνουμε την ταξινόμηση. Όπως καταλαβαίνουμε, σε ένα Νευρωνικό Δίκτυο το τελευταίο επίπεδο αποτελείται από Ν νευρώνες, όπου Ν ο αριθμός των τάξεων που θέλουμε να αναγνωρίζει το Νευρωνικό Δίκτυο. Μέσω της Softmax παράγονται Ν πιθανότητες για καθέναν από τους τελικούς νευρώνες και η μεγαλύτερη τιμή από αυτές μας υποδεικνύει την τάξη του αντικειμένου στην οποία υπάγεται.

Στην πράξη τόσο η Binary step function όσο και η Linear function δεν χρησιμοποιούνται αρκετά λόγω της γραμμικής τους συμπεριφοράς. Οι υπόλοιπες συναρτήσεις είναι αυτές που χρησιμοποιούνται κατα κύριο λόγο σε όλα τα Νευρωνικά Δίκτυα.

3.2.4 - Συναρτήσεις Σφάλματος (Loss Functions) & Αλγόριθμος Απότομης Καθόδου (Gradient Descent Algorithm)

Προηγουμένως αναφέραμε τη διαδικασία λειτουργίας του Νευρωνικού Δικτύου. Όπως είπαμε, τα Νευρωνικά Δίκτυα έχουν την ικανότητα να εκπαιδεύονται. Αυτό γίνεται με την αλλαγή των τιμών των βαρών w_i με στόχο να ελαχιστοποιηθούν τα σφάλματα. Με τον όρο σφάλμα εννοούμε μετρικές συναρτήσεις που έχουν υλοποιηθεί, οι οποίες ποσοτικοποιούν την «διαφορά» ανάμεσα στην έξοδο που μας αποδίδει το Νευρωνικό Δίκτυο και στην έξοδο που θα θέλαμε να πετύχουμε. Για να επιτευχθεί αυτό, γίνεται διαχωρισμός των δεδομένων δεδομένα εκπαίδευσης και δεδομένα αξιολόγησης. Έπειτα, με τον καθορισμό της Συνάρτησης Σφάλματος (Loss Function) εφαρμόζουμε αλλαγές στα w_i με αλγοριθμικό τρόπο μέχρι να πετύχουμε το μικρότερο δυνατό σφάλμα για τα δεδομένα εκπαίδευσής μας.

Οι πιο γνωστές και ευρέως χρησιμοποιούμενες στα Νευρωνικά Δίκτυα Συναρτήσεις Σφάλματος είναι:

1. Μέσο τετραγωνικό σφάλμα (MSE-Mean Square Error):

$$MSE = \frac{1}{N}\sum_{i=1}^{n}(Y_i - Y_i^*)^2$$

Όπου n είναι το πλήθος των output nodes, Y_i η έξοδος που υπολογίζεται από το Νευρωνικό Δίκτυο και Y_i^* η επιθυμητή έξοδος που θα θέλαμε να πετύχουμε.

2. Cross-Entropy Error Function

Είναι πολύ διαδεδομένη και χρησιμοποιείται κατά κύριο λόγο στα Νευρωνικά

Δίκτυα, τα οποία κάνουν ταξινόμηση. Η συνάρτηση ορίζεται με τον κάτωθι τρόπο:

$$Cross\ Entropy = \sum_{i=1}^{N} \left(-t_i * log(s_i)\right)$$

Όπου n το πλήθος των νευρώνων στο στρώμα εξόδου, t_i (τιμές είναι είτε 0 είτε 1) πιθανότητα πραγματικότητας (η πιθανότητα ground truth), s_i η πιθανότητα όπως υπολογίζεται από το δίκτυο με τη χρήση της συνάρτησης Softmax.

Για να το κατανοήσουμε παραθέτουμε το εξής παράδειγμα

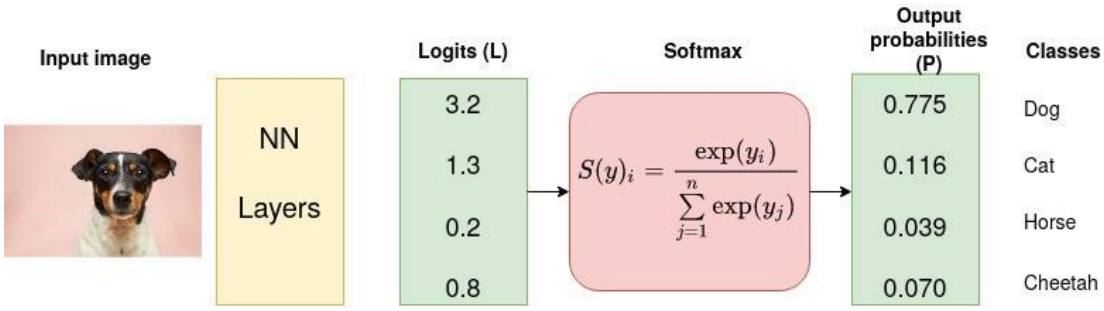


Figure 25: Exaple of Cross Entropy Loss Fuction

Όπως φαίνεται έχουμε ένα Νευρωνικό με τέσσερις κλάσεις (Dog Cat Horse Cheetah). Στο στρώμα εξόδου έχουμε τις πιθανότητες η φωτογραφία μας να ανήκει στην αντίστοιχη κλάση. Η φωτογραφία πρόκειται για δεδομένα εκπαίδευσης και άρα ξέρουμε τον ground truth πίνακα των κλάσεων.

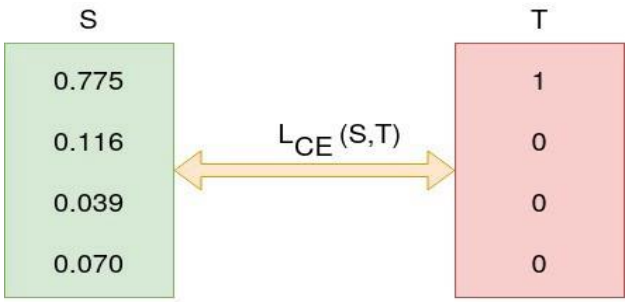


Figure 26:Scores Table and Truth Table

Ο πίνακας S είναι οι πιθανότητες που προκύπτουν από τα πραγματικά δεδομένα και ο πίνακας T είναι ο πίνακας πιθανοτήτων των ground truths. Άρα το Cross Entropy Loss είναι:

$$Cross\ Entropy = \sum_{i=1}^{4} \left(-t_i * log(s_i)\right)$$

$$Cross\ Entropy = -1 * log(0.775) - 0 * \log(0.116) - 0 * \log(0.039) - 0 * \log(0.070) \approx 0.51$$

Αφού επιλεγεί η κατάλληλη συνάρτηση σφάλματος για το μοντέλο μας, στόχος είναι η εύρεση των κατάλληλων w_i, έτσι ώστε το σφάλμα που προκύπτει να είναι το ελάχιστο. Για την επίτευξη αυτού του στόχου χρησιμοποιούμε τον αλγόριθμο Απότομης Καθόδου (Gradient Descent). Ο αλγόριθμος αυτός εκφράζει την συνάρτηση σφάλματος ως προς τα w_i, έπειτα με χρήση μερικών παραγώγων ως προς τα w_i και την ελαχιστοποίηση της συνάρτησης σφάλματος βρίσκουμε τα κατάλληλα w_i. Ο λόγος που γίνεται χρήση των μερικών παραγώγων ως προς w_i είναι ότι μας δείχνουν την πληροφορία του πώς θα μεταβληθεί το σφάλμα με την αλλαγή των w_i. Η επαναληπτική μέθοδος αλλαγής των w_i είναι η εξής:

$$w_i^{new} = w_i^{old} - a\frac{\partial F_{loss}(w)}{\partial w_i}$$

Η μέθοδος συγκλίνει σε κατάλληλα w_i με την χρήση κατάλληλης συνάρτησης σφάλματος και επιλογή αρχικών βαρών. Το w_i^{new} είναι η νέα τιμή που προκύπτει από την παλαιά w_i^{old}, a είναι το learning rate και F_{loss} η συνάρτηση σφάλματος. Αυτή η επαναληπτική αριθμητική μέθοδος εφαρμόζεται για όλα τα w_i.

Εναλλαγές αυτού του αλγορίθμου αποτελούν ο αλγόριθμος Stochastic Gradient Descent και Batch Gradient Descent, τους οποίους αναφέρουμε πληροφοριακά και δεν θα εξηγήσουμε περαιτέρω στα πλαίσια της παρούσας διπλωματικής εργασίας.

3.3 - Συνελικτικά Νευρωνικά Δίκτυα

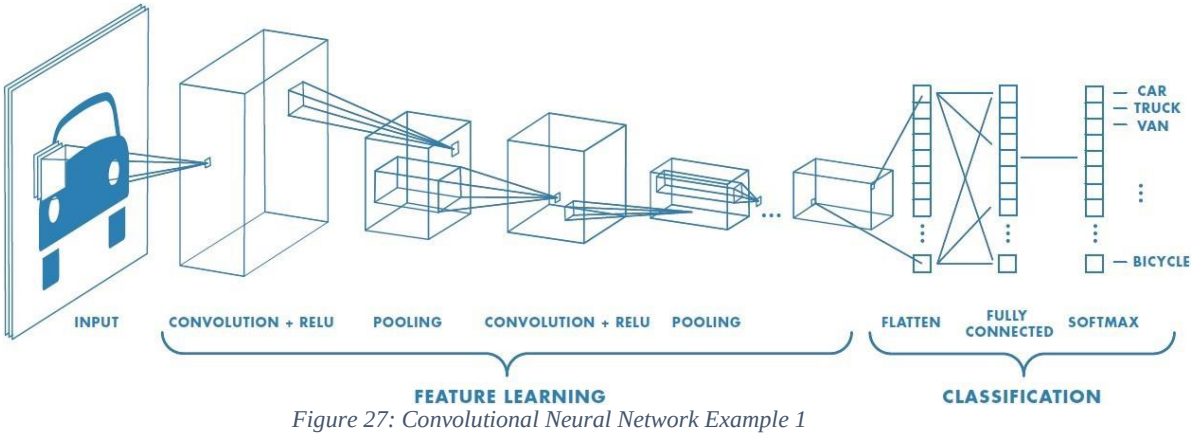


Figure 27: Convolutional Neural Network Example 1

Τα Συνελικτικά Νευρωνικά Δίκτυα είναι ο πιο διαδεδομένος και χρησιμοποιούμενος τύπος Νευρωνικών Δικτύων που χρησιμοποιείται στην Όραση Υπολογιστών. Υπερέχουν έναντι των πλήρως συνδεδεμένων Νευρωνικών Δικτύων, καθώς είναι υπολογιστικά πιο «φτηνά» (έχουν μικρότερο πλήθος παραμέτρων σε σχέση με τα πλήρως συνδεδεμένα Νευρωνικά Δίκτυα) και μας επιτρέπουν χωρικές συσχετίσεις. Στα Συνελικτικά Νευρωνικά Δίκτυα κάθε νευρώνας ενός στρώματος συνδέεται με συγκεκριμένου πλήθους νευρώνες με το επόμενο στρώμα. Σε αυτά επιτελείται η πράξη της συνέλιξης, όπως ορίζεται για την επεξεργασία εικόνας με την εφαρμογή

κατάλληλων φίλτρων. Για τους νευρώνες που βρίσκονται στο ίδιο στρώμα τα φίλτρα έχουν ίδια βάρη. Επίσης, τα φίλτρα οργανώνονται σε πλέγματα και εφαρμόζονται σε διαφορετικές περιοχές της εικόνας.

3.3.1 – Συνελικτικό Στρώμα (Convolution Layer)

Από τα σήματα είναι γνωστή η συνέλιξη, η οποία συμβολίζεται με το * και έχει τον κάτωθι τύπο.

$$(F * G)\,[n] \;=\; F[n] \;*\; G[n] \;=\; \sum_{i=-\infty}^{+\infty} F[i]G[n-i]$$

Στα Συνελικτικά Στρώματα για την επεξεργασία της εικόνας γίνεται η πράξη της συνέλιξης. Πριν προχωρήσουμε στον ορισμό αυτής της πράξης, είναι απαραίτητο να δώσουμε κάποιους ορισμούς. Κάθε εικόνα έχει διαστάσεις ύψος , πλάτος, κανάλια (height,width,channels) , όπου κανάλια είναι το πλήθος των βασικών χρωμάτων για κάθε στοιχείο της εικόνας. Συνήθως ο αριθμός των καναλιών είναι ίσος με τον αριθμό 3 για μια εικόνα RGB (red, green, blue).

Figure 28: Channels of an image

Αντίστοιχα, το φίλτρο μπορεί να θεωρηθεί ως ένα παράθυρο με διαστάσεις ύψος επί πλάτος και με p πλήθος καναλιών χρώματος. Για κάθε φίλτρο οι διαστάσεις ύψους και πλάτους είναι συνήθως μικρές και ίσες και μας δείχνουν τις χωρικές διαστάσεις του φίλτρου. Ο αριθμός p του φίλτρου εκφράζει το βάθος του. Ο αριθμός p πρέπει να είναι ίσος είτε με τα κανάλια της εικόνας, είτε με το βάθος του προηγούμενου στρώματος που δέχεται σαν είσοδο. Εναλλακτική ονομασία του φίλτρου είναι αλλιώς πυρήνας (kernel). Δίνουμε τον ορισμό της συνέλιξης δισδιάστατων διακριτών σημάτων :

$$(F * G)\,[x,y] \;=\; F[x,y] \;*\; G[x,y] \;=\; \sum_{i=-\infty}^{+\infty}\sum_{j=-\infty}^{+\infty} F[i,j]G[x-i,y-j]$$

όπου F, δισδιάστατο διακριτό σήμα εικόνας και G, δισδιάστατο διακριτό σήμα φίλτρου/πυρήνα. Το αποτέλεσμα αυτής της πράξης ονομάζεται χάρτης

ενεργοποίησης (activation map) ή αλλιώς χάρτης χαρακτηριστικών (feature map). Αν το μέγεθος του πίνακα της εικόνας είναι MxN και το μέγεθος του φίλτρου είναι FxF το μέγεθος του χάρτη χαρακτηριστικών είναι (M-F+1)x(N-F+1). Η σχηματική απεικόνιση ενός Συνελικτικού Δικτύου σε μορφή Νευρωνικού Δικτύου φαίνεται στο παρακάτω σχήμα:

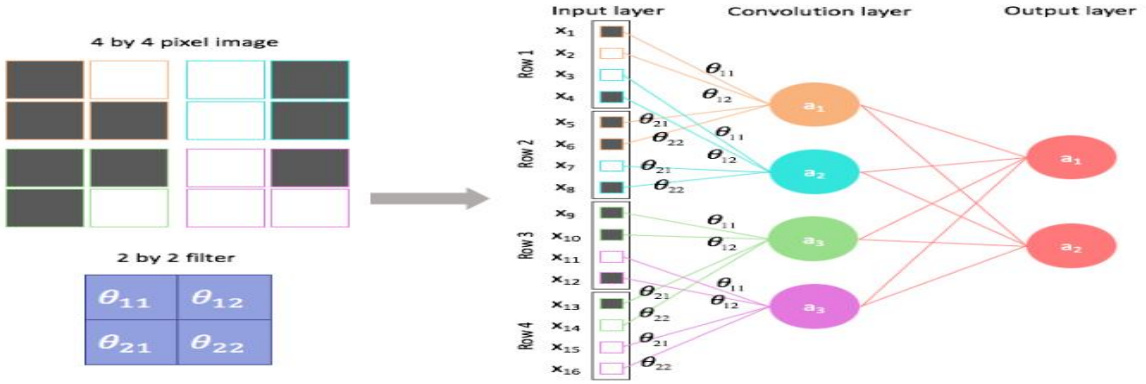


Figure 29: Σχηματική απαράσταση Συνελικτικού Δικτύου

Η τιμή κάθε στοιχείου του χάρτη χαρακτηριστικών δηλώνει την πιθανότητα το επιθυμητό χαρακτηριστικό να βρίσκεται στην αντίστοιχη περιοχή της εικόνας. Για περαιτέρω κατανόηση της συνέλιξης παραθέτουμε το παράδειγμα της εικόνας :

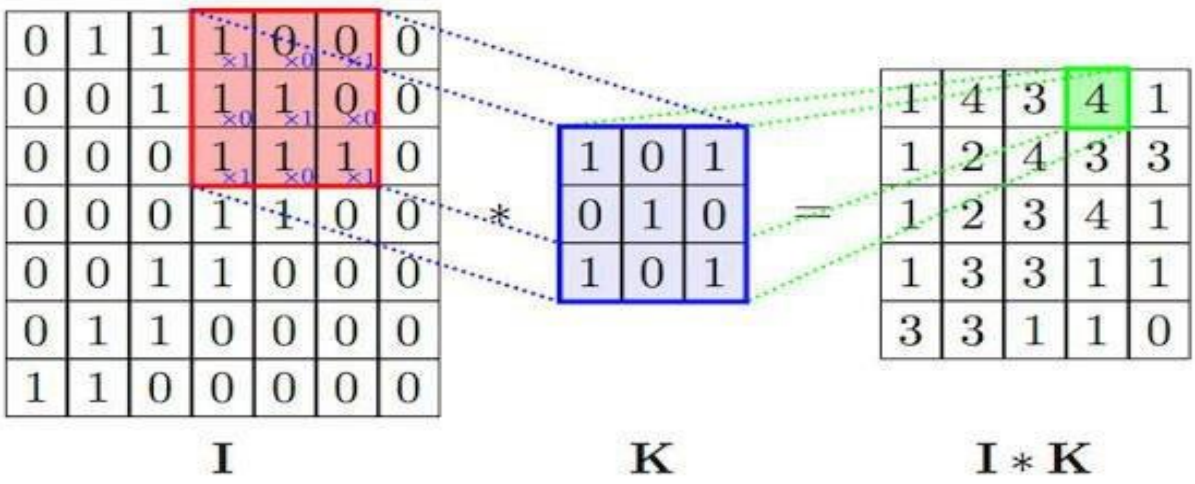


Figure 30: Convolution Example 1

Όπου I το δισδιάστατο σήμα της εικόνας και K το φίλτρο. Το επιθυμητό στοιχείο στον χάρτη χαρακτηριστικών που φαίνεται με το πράσινο χρώμα υπολογίζεται ως :

$$1 * 1 + 0 * 0 + 0 * 1 + 1 * 0 + 1 * 1 + 0 * 0 + 1 * 1 + 1 * 0 + 1 * 1 = 4$$

Για τον υπολογισμό του χάρτη χαρακτηριστικών, παίρνουμε το φίλτρο και το ολισθαίνουμε πάνω στην εικόνα ή στο προηγούμενο στρώμα. Ο υπολογισμός του κάθε στοιχείου του χάρτη χαρακτηριστικών υπολογίζεται με τον πολλαπλασιασμό των τιμών του φίλτρου με τις αντίστοιχες τιμές της εικόνας που καλύπτει και έπειτα αθροίζουμε όλα αυτά τα γινόμενα. Παρακάτω παρουσιάζεται πως γίνεται αυτή η ολίσθηση και πως προκύπτει σταδιακά ο χάρτης χαρακτηριστικών.

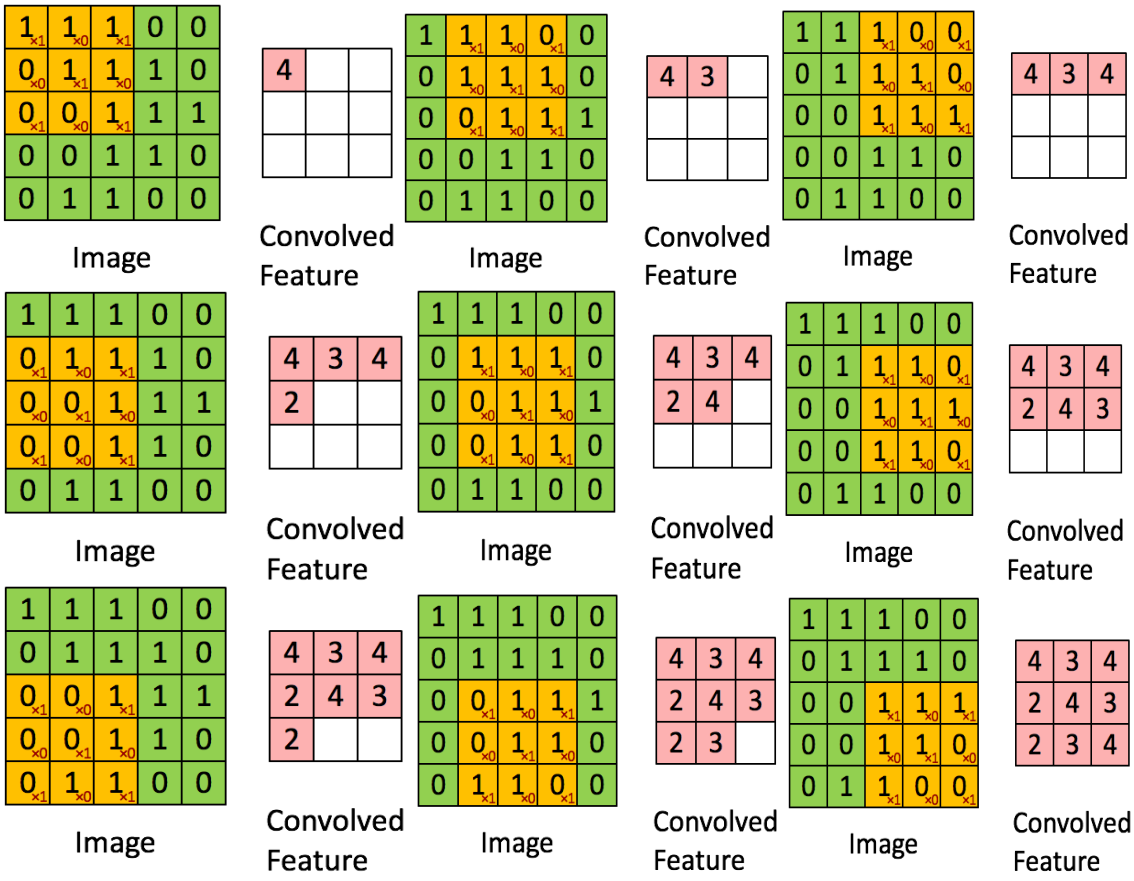


Figure 31: Step by step Convolution

Στο κίτρινο γραμμοσκιασμένο μέρος της εικόνας φαίνεται ο τρόπος που εφαρμόζεται το φίλτρο. Οι τιμές του φίλτρου φαίνονται με κόκκινο χρώμα και βλέπουμε ότι πολλαπλασιάζονται με τις αντίστοιχες τιμές της εικόνας. Η τιμή του κάθε στοιχείου του χάρτη χαρακτηριστικών φαίνεται με ροζ χρώμα και προκύπτει από το άθροισμα των γινομένων.

Κάθε στοιχείο του πυρήνα-φίλτρου του συνελικτικού στρώματος αποτελεί ένα αντίστοιχο βάρος. Αυτά τα βάρη έχουν επιρροή στο αποτέλεσμα χάρτη χαρακτηριστικών της συνέλιξης και αποτελούν παραμέτρους που αλλάζουν με την διαδικασία της εκπαίδευσης.

Το βήμα ολίσθησης ονομάζεται stride. Στον ορισμό της συνέλιξης που δώσαμε προηγουμένως, το stride είναι ίσο με 1 τόσο οριζοντίως όσο και καθέτως ή αλλιώς (1,1), όμως ορισμένες φορές είναι μεγαλύτερο από αυτό, με αποτέλεσμα να μειώνονται οι χωρικές διαστάσεις (ύψους και πλάτους) του χάρτη χαρακτηριστικών. Δηλαδή, αν για παράδειγμα το stride είναι (3,3) αυτό σημαίνει ότι μεταφέρουμε το φίλτρο ανά τρεις θέσεις οριζοντίως και ανά τρεις καθέτως. Συνήθως το βήμα οριζόντια είναι ίσο με το βήμα κατακόρυφα. Σε αυτή την περίπτωση για μια εικόνα με διαστάσεις MxN, ένα φίλτρο με διαστάσεις FxF και βήμα stride S, ίδιο για οριζόντιο και κατακόρυφο βήμα, το αποτέλεσμα της συνέλιξης έχει διαστάσεις KxL όπου:

$$K = \frac{M-F}{S} + 1$$

$$L = \frac{N-F}{S} + 1$$

Αν το $\frac{M-F}{S}$ ή το $\frac{N-F}{S}$ δεν είναι ακέραιοι παίρνουμε το πλησιέστερο ακέραιο προς τα κάτω.

Συχνά χρησιμοποιείται παραπάνω από ένα φίλτρο ανά στρώμα συνέλιξης με αποτέλεσμα να προκύπτουν πολλαπλοί χάρτες χαρακτηριστικών (κάθε φίλτρο δίνει και από έναν χάρτη χαραικτηριστικών) που αντιστοιχούν σε διαφορετικά χαρακτηριστικά. Τα αποτελέσματα που προκύπτουν στοιβάζονται το ένα πίσω από το άλλο όπως φαίνεται παρακάτω. Το πλήθος αυτών των χαρτών χαρακτηριστικών που προκύπτουν, ονομάζεται βάθος (depth).

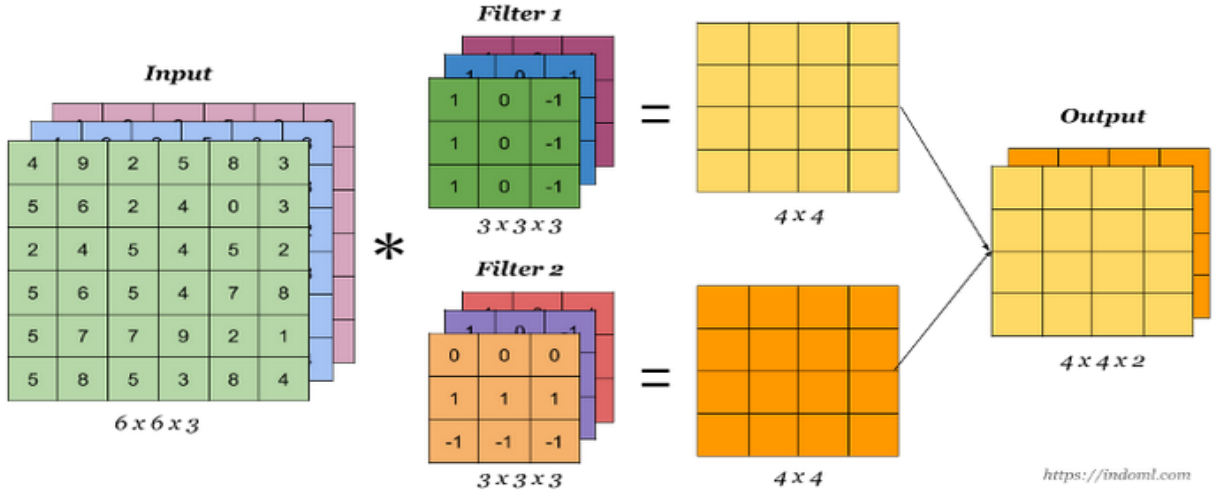


Figure 32:Multiple filters

Το πρόβλημα που προκύπτει από τη διαδικασία της συνέλιξης είναι ότι οι τιμές στην περίμετρο της εικόνας τείνουν να χαθούν. Οπτικοποιούμε το παρακάτω με το εξής παράδειγμα:

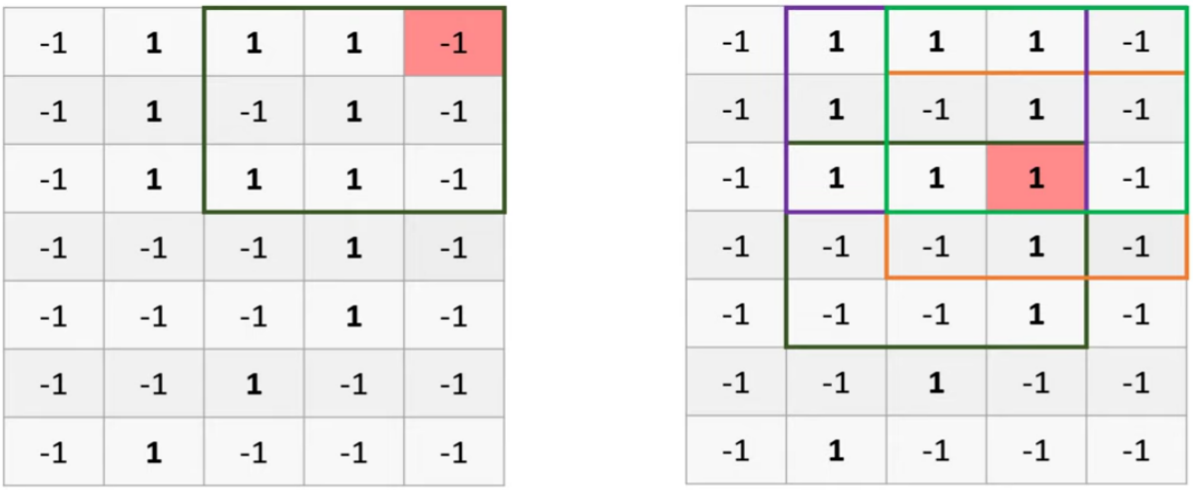


Figure 33:Convolution without padding problem

Φαίνεται στη δεξιά εικόνα ότι όταν ένα φίλτρο 3x3 με stride 1 εφαρμόζεται στον πίνακα της εικόνας, το στοιχείο (4,0) συμμετέχει μόνο μία φορά, ενώ το στοιχείο (2,3) συμμετέχει σαφώς περισσότερες φορές. Για να λυθεί αυτό το πρόβλημα και όλες οι τιμές να συμμετέχουν με τον ίδιο τρόπο αρκετές φορές, χρησιμοποιείται η μέθοδος του padding. Αυτή η μέθοδος αυξάνει το πλήθος των γραμμών και των στηλών συνήθως κατά δύο και γεμίζει με μηδενικά συνήθως τις γύρω γύρω θέσεις. Μετά το padding γίνεται η πράξη της συνέλιξης.

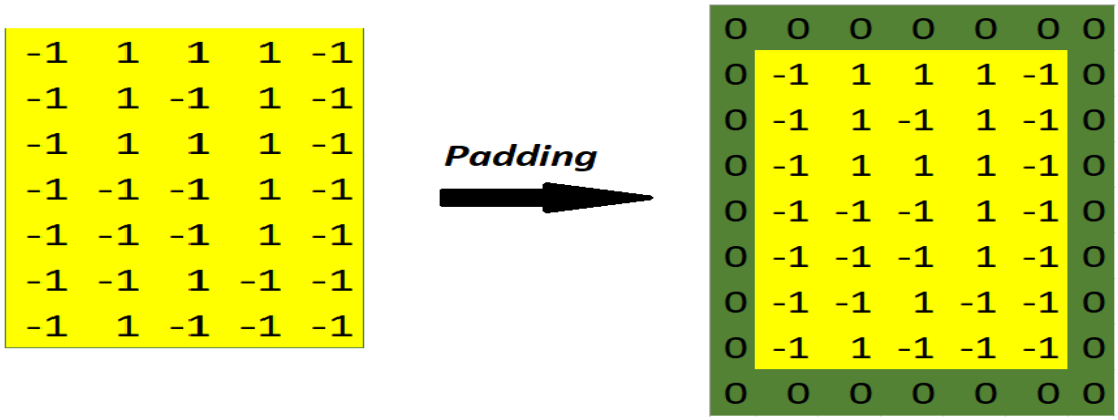


Figure 34: Padding

Αφού έχουμε αναφέρει τη συνέλιξη μεταξύ εικόνας και φίλτρου, στόχος μας τώρα είναι να αναδείξουμε τους λόγους για τους οποίους γίνεται αυτό μέσα από ένα παράδειγμα. Ένα φίλτρο πρακτικά αποτελεί έναν ανιχνευτή μοτίβων (pattern detector). Στο παρακάτω παράδειγμα της εικόνας, αριστερά επάνω φαίνεται η ασπρόμαυρη αναπαράσταση του αριθμού επτά. Στην πρώτη γραμμή βλέπουμε τους πίνακες των φίλτρων που έχουν διαστάσεις 3x3. Στη δεύτερη γραμμή, βλέπουμε οπτική αναπαράσταση του φίλτρου. Το -1 αναπαρίσταται οπτικά με το μαύρο, το 0 με το γκρι και το 1 με το άσπρο. Στην τελευταία γραμμή βλέπουμε το αποτέλεσμα της συνέλιξης της εικόνας με το εκάστοτε φίλτρο σαν εικόνα. Τα εικονοστοιχεία του κάθε αποτελέσματος που είναι άσπρα είναι αυτά που εντοπίζει το κάθε φίλτρο. Το φίλτρο 1 εντοπίζει πάνω οριζόντιες γραμμές, το φίλτρο 2 εντοπίζει αριστερές κάθετες γραμμές, φίλτρο 3 κάτω οριζόντιες γραμμές και φίλτρο 4 δεξιές κάθετες γραμμές. Αυτά τα φίλτρα είναι αρκετά απλά και ονομάζονται ανιχνευτές ακμών (edge detectors).

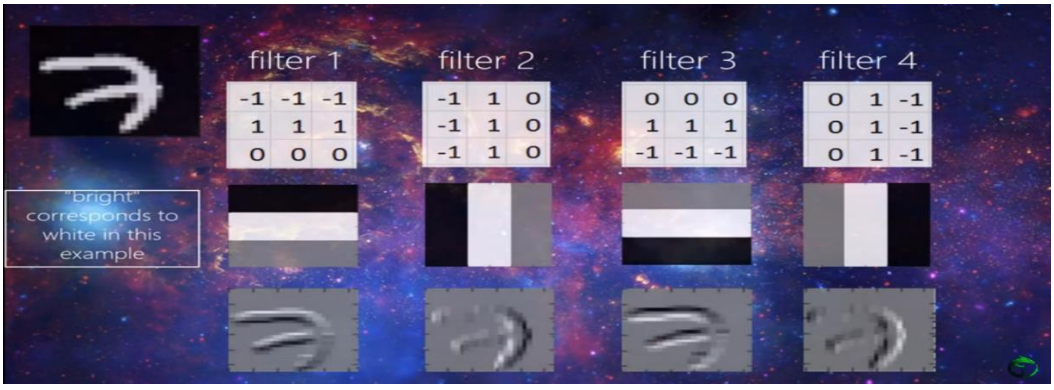


Figure 35: Edge Detectors

Ένα συνελικτικό δίκτυο στα πρώτα στρώματα μαθαίνει να αναγνωρίζει βασικά χαρακτηριστικά όπως γραμμές, γωνίες κλπ. Στα μεσαία στρώματα χρησιμοποιούνται φίλτρα για την αναγνώριση μερών από αντικείμενα, για παράδειγμα, σε ένα πρόσωπο αναγνωρίζουν τη μύτη, το στόμα, τα μάτια κλπ. Στα τελευταία στρώματα τα φίλτρα είναι ικανά να αναγνωρίζουν ολόκληρα αντικείμενα σε διαφορετικές θέσεις και σχήματα.

3.3.2 – Στρώμα Ενεργοποίησης (Activation Layer)

Στις περισσότερες περιπτώσεις τα συστήματα, τα οποία επιδιώκουμε να προσεγγίσουμε δεν έχουν γραμμική συμπεριφορά. Για τον λόγο αυτό, για να ενταχθεί αυτή η μη γραμμική συμπεριφορά στα Συνελικτικά Νευρωνικά Δίκτυα, τοποθετούμε ένα στρώμα ενεργοποίησης (activation layer) έπειτα από κάθε συνελικτικό στρώμα (convolution layer). Όπως έχουμε πει και προηγουμένως, η συνάρτηση που χρησιμοποιείται σε αυτό το στρώμα λέγεται συνάρτηση ενεργοποίησης και συνήθως είναι η ReLU, διότι δεν είναι απόλυτα γραμμική. Αυτή η συνάρτηση, όπως έχουμε αναφέρει, μας βοηθάει, γιατί απλοποιεί το back-propagation δηλαδή την εκπαίδευση του νευρωνικού δικτύου. Εναλλακτικά, γίνεται χρήση της συνάρτησης Tanh ανάλογα με τη φύση του προβλήματος.

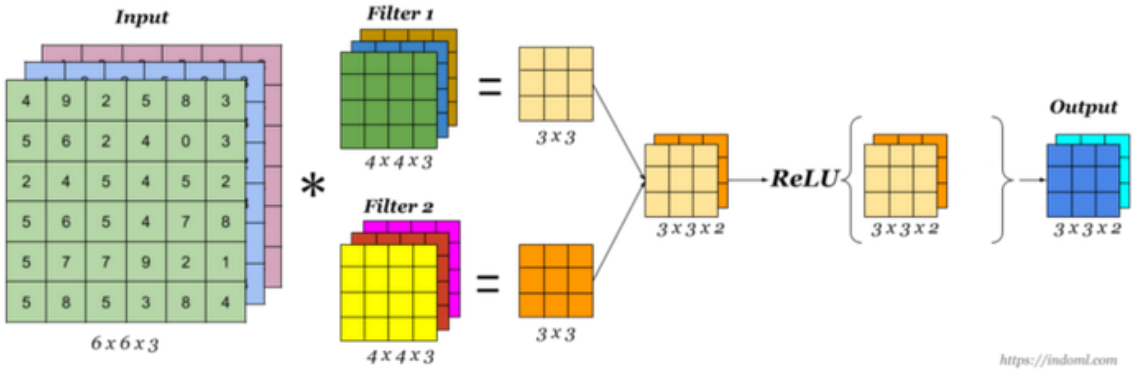


Figure 36: Activation Layer CNN

3.3.3 – Στρώμα Υποδειγματοληψίας (Pooling Layer)

Στόχος των στρωμάτων υποδειγματοληψίας είναι να μειώνουν τις διαστάσεις των χαρτών χαρακτηριστικών που προκύπτουν σαν αποτέλεσμα στο συνελικτικό στρώμα. Αυτό επιτυγχάνεται με χωρισμό του χάρτη χαρακτηριστικών σε μη επικαλυπτόμενα κομμάτια. Κάθε ένα από αυτά τα κομμάτια ανάλογα με την μέθοδο υποδειγματοληψίας αντικαθίσταται από μία τιμή «αντιπρόσωπο».

Πιο γνωστοί μέθοδοι υποδειγματοληψίας που χρησιμοποιούνται ευρέως στα Συνελικτικά Νευρωνικά Δίκτυα είναι:

1. Συγκράτηση Μέγιστης τιμής (Max pooling): Από κάθε μη επικαλυπτόμενο τμήμα του χάρτη χαρακτηριστικών επιλέγεται ως αντιπρόσωπος η

μεγαλύτερη τιμή.

2. Συγκράτηση Μέσης τιμής (Average pooling): Από κάθε μη επικαλυπτόμενο τμήμα του χάρτη χαρακτηριστικών επιλέγεται ως αντιπρόσωπος η μέση τιμή του τμήματος

3. Επιλογή Τυχαίας τιμής (Stochastic pooling): Από κάθε μη επικαλυπτόμενο τμήμα του χάρτη χαρακτηριστικών επιλέγεται ως αντιπρόσωπος ένας τυχαίος αριθμός του τμήματος.

Αποτέλεσμα της υποδειγματοληψίας είναι να μειώνονται οι παράμετροι και να βελτιώνεται η διαδικασία εκπαίδευσης .

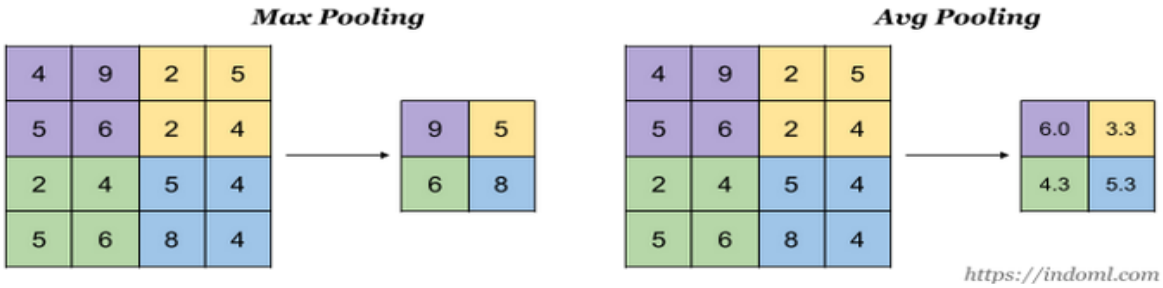


Figure 37: Max and Average Pooling

3.3.4 – Στρώμα Κανονικοποίησης Παρτίδας (Batch Normalization Layer)

Η κανονικοποίηση είναι η μέθοδος, με την οποία διαφορετικά αριθμητικά δεδομένα αποδίδονται σε μια κοινή κλίμακα, χωρίς να παραμορφώνεται το «σχήμα» τους. Αρκετές φορές τα χαρακτηριστικά έχουν μεγάλες διαφορές στις τιμές τους (άλλα χαρακτηριστικά είναι πολύ μικρά και άλλα αρκετά μεγάλα), με αποτέλεσμα να δυσχεραίνεται η διαδικασία της εκμάθησης. Το Στρώμα Κανονικοποίησης Παρτίδας επιχειρεί να πάρει το output διαφορετικών στρωμάτων και να το κανονικοποιήσει με τρόπο ώστε η μέση τιμή να είναι μηδέν και η διακύμανση να ισούται με τη μονάδα. Η ένταξη αυτού του στρώματος φαίνεται να αυξάνει την απόδοση της εκμάθησης και την επιταχύνει.

3.3.5 – Πλήρως Συνδεδεμένο Στρώμα (Fully Connected Layer)

Ένα Πλήρως Συνδεδεμένο Στρώμα τοποθετείται στο τέλος του Συνελικτικού Νευρωνικού Δικτύου. Οι διαφορετικοί χάρτες χαρακτηριστικών που έχουν προκύψει στο προηγούμενο στρώμα υπόκεινται σε «flatten» και έτσι δημιουργείται ένα διάνυσμα. Αυτό το δίανυσμα δίνεται σαν input στο Πλήρως Συνδεδεμένο Στρώμα με σκοπό να γίνει η ταξινόμηση της εικόνας. Στο τελευταίο στάδιο αυτού του επιπέδου χρησιμοποιείται σαν συνάρτηση ενεργοποίησης η συνάρτηση Softmax που αναφέραμε προηγουμένως.

3.4 – Έννοιες, ορισμοί και μετρικές της Ανίχνευσης Αντικειμένων

3.4.1 – Πλαίσιο Οριοθέτησης (Bounding Box)

Το Bounding Box ή αλλιώς Πλαίσιο Οριοθέτησης είναι ένα από τα πιο αναγνωρισμένα και συνήθως χρησιμοποιούμενα εργαλεία στην Ανίχνευση Αντικειμένων. Σε μια εικόνα ένα Πλαίσιο Οριοθέτησης είναι το ορθογώνιο, το οποίο περικλείει ένα ολόκληρο αντικείμενο. Ιδανικά, θα θέλαμε αυτό το ορθογώνιο να είναι το μικρότερο από όλα το ορθογώνια που θα μπορούσαν να περικλείσουν ένα αντικείμενο. Η αναπαράσταση ενός Πλαίσιου Οριοθέτησης γίνεται συνήθως με έναν από τους εξής τρόπους:

1. Δίνοντας τις συντεταγμένες δύο σημείων του ορθογωνίου, δηλαδή ($x_{downleft}$, $y_{downleft}$) και ($x_{upright}$, $y_{upright}$). Όπου ($x_{downleft}$, $y_{downleft}$) συντεταγμένες της κάτω αριστερής γωνίας και ($x_{upright}$, $y_{upright}$) συντεταγμένες της άνω δεξιάς γωνίας.

2. Δίνοντας τις συντεταγμένες του σημείου του κέντρου του Πλαισίου Οριοθέτησης (x_{center} , y_{center}) και έπειτα του ύψους (height) και του πλάτους (width).

3. Δίνοντας τις συντεταγμένες της κάτω αριστερής γωνίας ($x_{downleft}$, $y_{downleft}$), του ύψους (height) και του πλάτους (width).

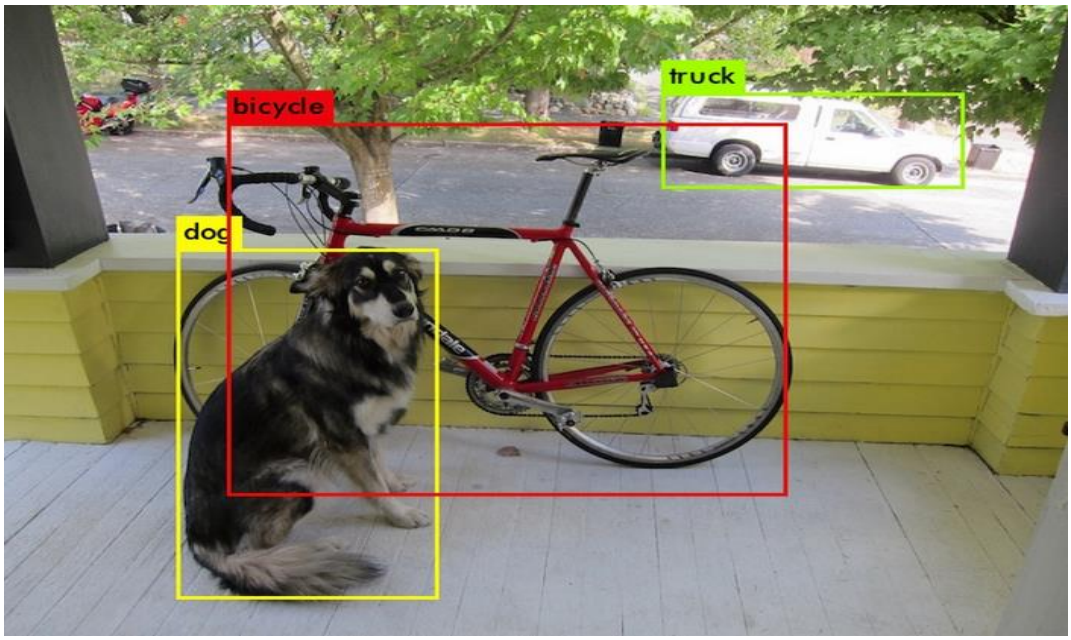


Figure 38:Bounding Box Example

3.4.2 – Λόγος Τομής προς Ένωση (Intersection over Union)

Intersection over Union (IoU) ή αλλιώς Λόγος Τομής προς Ένωση είναι μια από τις πιο δημοφιλείς μετρικές που χρησιμοποιούνται στην Ανίχνευση Αντικειμένων. Όπως

έχουμε αναφέρει προηγουμένως, η Ανίχνευση Αντικειμένων έχει δύο στόχους. Ο ένας είναι το localization, δηλαδή ο εντοπισμός της τοποθεσίας του αντικειμένου σε μία εικόνα και ο άλλος είναι το classification, δηλαδή ο χαρακτηρισμός της τάξης του αντικειμένου.
Για να υπολογίσουμε το IoU απαιτούνται :

1. Ground truth bounding box (b_{gt}) : Είναι το Ορθογώνιο Πλαίσιο, το οποίο αντιπροσωπεύει την αλήθεια, δηλαδή το πραγματικό Πλαίσιο Οριοθέτησης του αντικειμένου.

2. Prediction bounding box (b_p) : Είναι το Ορθογώνιο Πλαίσιο, το οποίο προκύπτει ως output από το Νευρωνικό Δίκτυο.

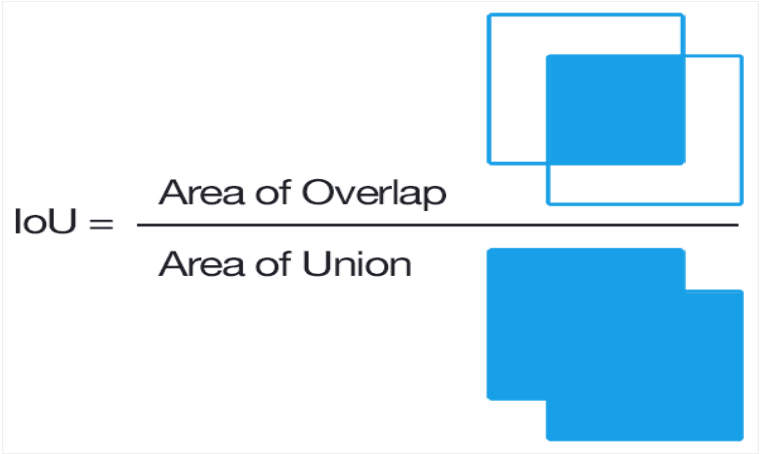


Figure 39:Intersection over Union

Όπως φαίνεται, το IoU είναι ίσο με το εμβαδόν της τομής των b_p και b_{gt} προς το εμβαδόν της ένωσης των b_p και b_{gt}. Η ποσότητα αυτή κυμαίνεται από 0 εώς 1,όπου το 0 δηλώνει ότι δεν υπάρχει καμία επικάλυψη ανάμεσα στα δύο Πλαίσια Οριοθέτησης και 1 δηλώνει ότι τα δύο Πλαίσια Οριοθέτησης ταυτίζονται. Η μετρική IoU χρησιμοποιεί συνήθως ένα κατώφλι για παράδειγμα a. Το κατώφλι αυτό μας βοηθάει να αποφασίσουμε αν η ανίχνευση που έγινε, είναι σωστή ή όχι. Συνήθεις τιμές που χρησιμοποιούνται για το κατώφλι a είναι 0,75 και 0,5.

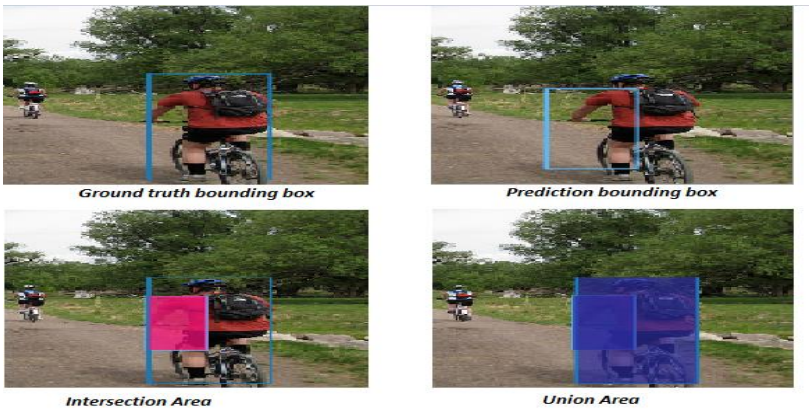


Figure 40:IoU Example

3.4.3 - Χαρακτηρισμός Προβλέψεων

Στην Ανίχνευση Αντικειμένων είναι σημαντικό να μπορούμε να χαρακτηρίσουμε για κάθε πρόβλεψη μιας συγκεκριμένης κλάσης πόσο καλή είναι και γενικά να κατηγοριοποιήσουμε τις προβλέψεις. Για να γίνει αυτό θα γίνει χρήση του IoU που αναφέραμε προηγουμένως. Έτσι λοιπόν, κάθε πρόβλεψη μπορεί να χαρακτηριστεί:

1. True Positive (TP): Σε αυτή την κατηγορία έγινε η σωστή πρόβλεψη. Ως True Positive ορίζουμε τις προβλέψεις, για τις οποίες η κλάση που έχει αποδώσει ο ανιχνευτής είναι η σωστή και ισχύει ότι $IoU(B_p,B_{gt})\geq a$ όπου a το κατωφλι του IoU.

2. False Positive: Σε αυτή την κατηγορία έγινε λάθος πρόβλεψη. Ως False Positive ορίζουμε τις προβλέψεις, για τις οποίες είτε η κλάση που έχει αποδώσει το μοντέλο είναι η σωστή και ισχύει ότι $IoU(B_p,B_{gt})<a$ όπου a το κατωφλι του IoU, είτε η κλάση που απέδωσε ο ανιχνευτής είναι λάθος.

3. False Negative: Σε αυτή την κατηγορία ο ανιχνευτής δεν κατάφερε να ανιχνεύσει την αλήθεια (ground truth).

Figure 41: TP, FP, FN Example

Από τον ορισμό του IoU βλέπουμε λοιπόν, ότι από το διάστημα [0,1] κάνουμε μια αντιστοίχιση σε TP, FP, FN. Το κατώφλι a που χρησιμοποιείται, είναι αυτό που μας υποδεικνύει πότε μια πρόβλεψη είναι αποδεκτή. Επίσης εύκολα μπορούμε να καταλάβουμε ότι το άθροισμα των TP και FP είναι το σύνολο όλων των ανιχνεύσεων που έκανε ο ανιχνευτής, και το άθροισμα των TP και FN είναι το πλήθος όλων των ground truths του ανιχνευτή.

3.4.4 – Ακρίβεια (Precision) & Ανάκληση (Recall)

Precision ή αλλιως Ακρίβεια είναι η μετρική που μας δείχνει το βαθμό του μοντέλου να ανιχνεύει μόνο τα αντικείμενα που επιθυμούμε. Ορίζεται ως το πλήθος των TP ανιχνεύσεων προς το πλήθος των συνολικών ανιχνεύσεων.

$$Precision = \frac{TP}{TP + FP} = \frac{TP}{ALL\ DETECTIONS}$$

Recall ή αλλιώς Ανάκληση είναι η μετρική που δείχνει την ικανότητα του μοντέλου να ανιχνεύει όλες τις αλήθειες (ground truths). Ορίζεται ως το πλήθος των TP ανιχνεύσεων προς το πλήθος όλων των ground truths.

$$Recall = \frac{TP}{TP + FN} = \frac{TP}{ALL\ GROUND - TRUTHS}$$

Ένα μοντέλο ιδανικά θα θέλαμε να έχει το μεγαλύτερή δυνατή Ακρίβεια και τη μεγαλύτερη δυνατή Ανάκληση. Όπως φαίνεται η Ανάκληση και η Ακρίβεια εξαρτώνται άμεσα από τα TP, FP, FN, τα οποία εξαρτώνται από το κατώφλι του IoU. Αν αυξήσουμε το κατώφλι, τότε περισσότερα αντικείμενα δεν θα ανιχνευθούν από τον ανιχνευτή. Άρα θα προκύψει μεγαλύτερο πλήθος από FN σε σχέση με πριν και ως αποτέλεσμα η Ανάκληση θα μειωθεί και η Ακρίβεια θα αυξηθεί. Από την άλλη, αν χαμηλώσουμε την τιμή του κατωφλίου, τότε θα αυξηθούν τα FP. Αυτό σημαίνει, ότι θα σημειωθεί μείωση στην Ακρίβεια και αύξηση στην Ανάκληση.

3.4.5 - Average Precision (AP) και mean Average Precision (mAp)

Οι μετρικές Precision και Recall που αναφέραμε προηγουμένως είναι αρκετά απλοϊκές για την Ανίχνευση Αντικειμένων. Η μετρική που χρησιμοποιείται κυρίως για την αξιολόγηση της Ανίχνευσης Αντικειμένων είναι η mean Average Precision. Αυτή η μετρική υπολογίζεται συνήθως για συγκεκριμένο κατώφλι IoU.

Για να υπολογιστεί αυτή η μετρική, είναι απαραίτητο πρώτα να υπολογιστεί το Average Precision. Με την σειρά της, η μετρική Average Precision υπολογίζεται από την καμπύλη Precision ως συνάρτηση του Recall. Το Average Precision υπολογίζεται για κάθε κλάση ξεχωριστά.

Ο τρόπος υπολογισμού της καμπύλης είναι ο εξής:

1. Από τα δεδομένα αξιολόγησης παίρνουμε όλα τα Πλαίσια Οριοθέτησης των προβλέψεων, δηλαδή την πιθανότητα να ανήκει ένα αντικείμενο στην αντίστοιχη κλάση.

2. Για τα δεδομένα αξιολόγησης επίσης έχουμε όλα τα ground truths που βοηθούν στον χαρακτηρισμό των προβλέψεων (δηλαδή αν η πρόβλεψη ισχύει ή όχι).

3. Ταξινομούμε τις πιθανότητες των προβλέψεων σε φθίνουσα σειρά.

4. Χαρακτηρίζουμε αληθή ή ψευδή κάθε πρόβλεψη σύμφωνα με το ground truth.

5. Βρίσκουμε τα σημεία ($Precision_i$, $Recall_i$) της κάθε πρόβλεψης ως εξής:

$$Precision_i = \frac{Past\ True\ Predictions\ +\ ispredtrue}{i}$$

$$Recall_i = \frac{Past\ True\ Predictions\ +\ ispredtrue}{ALL\ GROUND\ TRUTHS}$$

$$ispredtrue = \begin{cases} 1, & if\ i-th\ prediction\ is\ true \\ 0, & if\ i-th\ prediction\ is\ false \end{cases}$$

Για να το κατανοήσουμε περισσότερο, παραθέτουμε το εξής παράδειγμα. Έχουμε ένα evaluation dataset που φαίνονται 5 μήλα (άρα ground truth είναι 5) και το μοντέλο μας έκανε 10 προβλέψεις. Παραθέτουμε τον ταξινομημένο πίνακα ως προς τις πιθανότητες για το detection της κλάσης μήλου.

Πρόβλεψη	Πιθανότητα	Correct	Precision	Recall
1	0.98	TRUE	1/1	1/5
2	0.95	TRUE	2/2	2/5
3	0.84	FALSE	2/3	2/5
4	0.75	FALSE	2/4	2/5
5	0.74	FALSE	2/5	2/5
6	0.71	TRUE	3/6	3/5
7	0.64	TRUE	4/7	4/5
8	0.6	FALSE	4/8	4/5
9	0.55	FALSE	4/9	4/5
10	0.5	TRUE	5/10	5/5

Figure 42: Παράδειγμα πίνακα μήλου για την εύρεση του AP, mAP

Οι τιμές του $Precision_i$, $Recall_i$ προκύπτουν ως εξής για το στοιχείο i,j του πίνακα.

$Precision_1 = \frac{0+1}{1} = \frac{1}{1}$, $Recall_1 = \frac{0+1}{5} = \frac{1}{5}$ Άρα το πρώτο σημείο είναι το (1,0.2)
$Precision_2 = \frac{1+1}{2} = \frac{2}{2}$, $Recall_2 = \frac{1+1}{5} = \frac{2}{5}$ Άρα το πρώτο σημείο είναι το (1,0.4)
…
$Precision_{10} = \frac{4+1}{10} = \frac{5}{10}$, $Recall_{10} = \frac{4+1}{5} = \frac{5}{5}$ Άρα το πρώτο σημείο είναι το (0.5,1)

Για το παράδειγμά μας η καμπύλη του Precision ως προς το Recall που προκύπτει είναι η εξής:

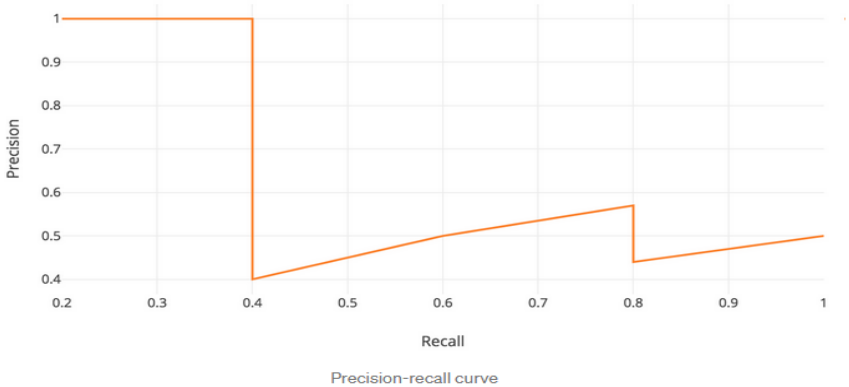


Figure 43: Καμπύλη Precision-Recall

Κάθε κλάση έχει και την δική της καμπύλη Precision-Recall. Επίσης φαίνεται ότι η καμπύλη ακολουθεί ένα zigzag μοτίβο.

Ο γενικός ορισμός του Average Precision για την ανίχνευση μια κλάσης είναι το εμβαδόν κάτω από την καμπύλη:

$$AP = \int_0^1 p(r)dr$$

Συνήθως για την Ανίχνευση Αντικειμένων τείνουμε να κάνουμε ομαλοποίηση της καμπύλης Precision-Recall όπως φαίνεται παρακάτω:

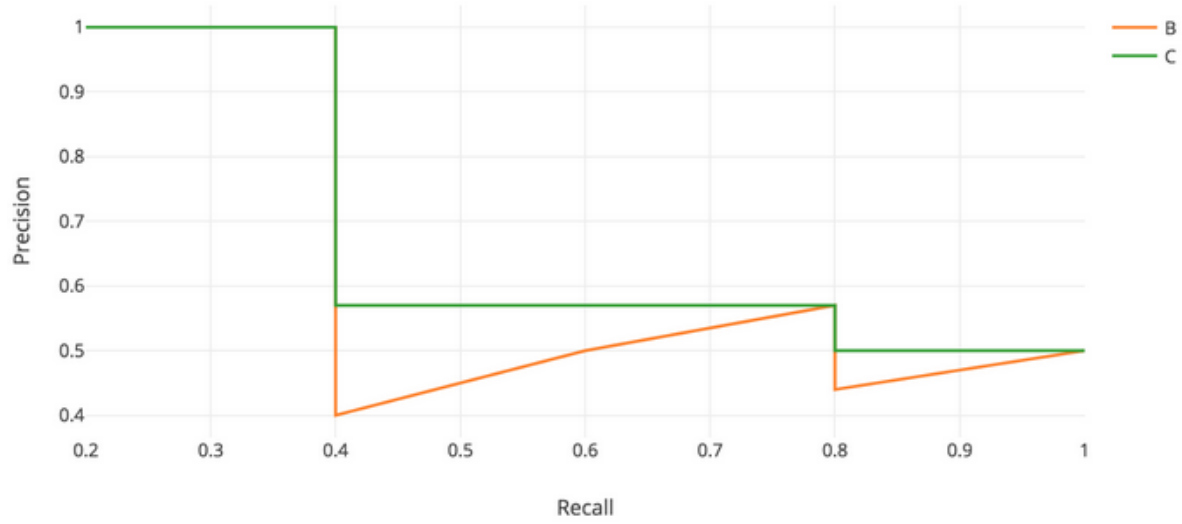


Figure 44: Ομαλοποιημένη καμπύλη Precision-Recall

Ο τύπος από τον οποίο προκύπτει αυτή η καμπύλη είναι ο εξής:

$$p_{inter}(x) = max_{r' \leq r} p(r')$$

Γραφικά αυτό που κάνουμε, είναι ότι κάθε τιμή του Precision στη καμπύλη Precision-Recall αντικαθίσταται με την μέγιστη τιμή προς τα δεξιά.

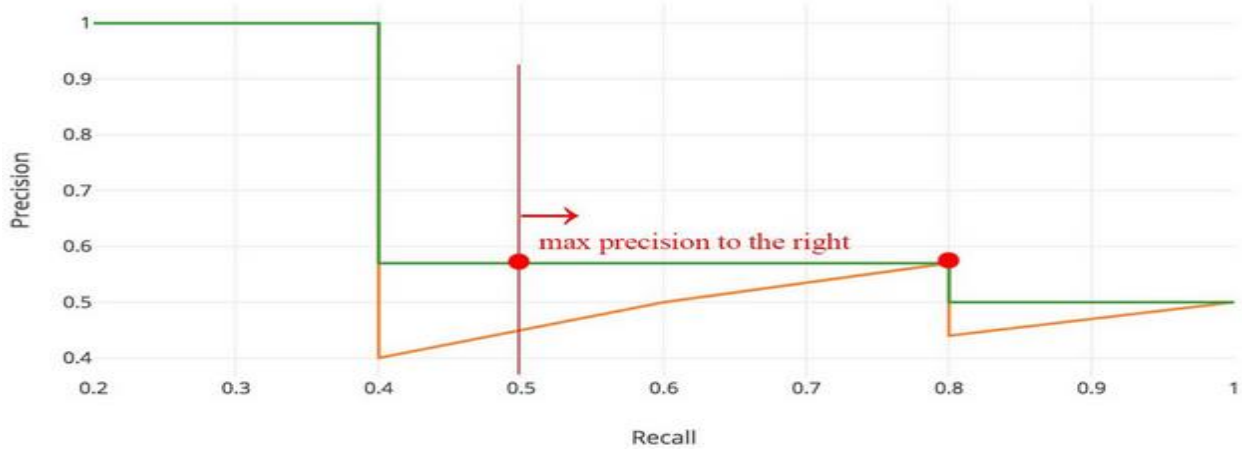


Figure 45: Τρόπος Δημιουργίας Ομαλοποιημένης Καμπύλης Precision-Recall

Σύμφωνα με το Pascal VOC 2008 , το Average Precision της κάθε κλάσης ισούται με:

$$AP = \frac{1}{11}\sum_{i=0}^{10} p_{inter}(\frac{i}{10})$$

Είναι δηλαδή ο μέσος όρος των P_{inter} για Recall ίσο με {0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1}. Ονομάζεται αλλιώς Interpolated Average Precision.

Ως mean Average Precision (mAP) ορίζεται:

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP(i)$$

όπου N ο αριθμός των κλάσεων, $AP(i)$ το Average Precision της i-οστής κλάσης. Είναι δηλαδή ο μέσος όρος των Average Precision των κλάσεων.

3.4.6 – Καταστολή Μη Μεγίστων (Non Maximum Suppression)

Ένα πρόβλημα που εμφανίζεται αρκετά συχνά στην Ανίχνευση Αντικειμένων είναι η ύπαρξη πολλών διαφορετικών επικαλυπτόμενων Πλαισίων Οριοθέτησης για το ίδιο αντικείμενο. Ο αλγόριθμος Καταστολή μη Μεγίστων (Non Maximum Suppression) επιλύει αυτό το πρόβλημα και πρακτικά επιλέγει το καλύτερο Πλαίσιο Οριοθέτησης για ένα αντικείμενο.

Ο τρόπος που το κάνει αυτό είναι να απορρίπτει και να συγχωνεύει τα Πλαίσια Οριοθέτησης. Ο αλγόριθμος δέχεται σαν όρισμα όλα τα Πλαίσια Οριοθέτησης και την πιθανότητα/σκορ να ανήκουν σε συγκεκριμένη κλάση. Ο ψευδοκώδικας του NMS είναι ο εξής:

Step 1: Επίλεξε το Πλαίσιο Οριοθέτησης με τη μεγαλύτερη πιθανότητα.

Step 2: Σύγκρινε την επικάλυψη (IoU) του Πλαισίου αυτού με τα άλλα Πλαίσια Οριοθέτησης.

Step 3: Αφαίρεσε όλα τα Πλαίσια Οριοθέτησης με επικάλυψη μεγαλύτερη από μια συγκεκριμένη τιμή.

Step 4: Επίλεξε το επόμενο Πλαίσιο Οριοθέτησης με το μεγαλύτερο σκορ.

Step 5: Επανάλαβε τα Steps 2-4 μέχρι να μην αφαιρούνται άλλα Πλαίσια Οριοθέτησης.

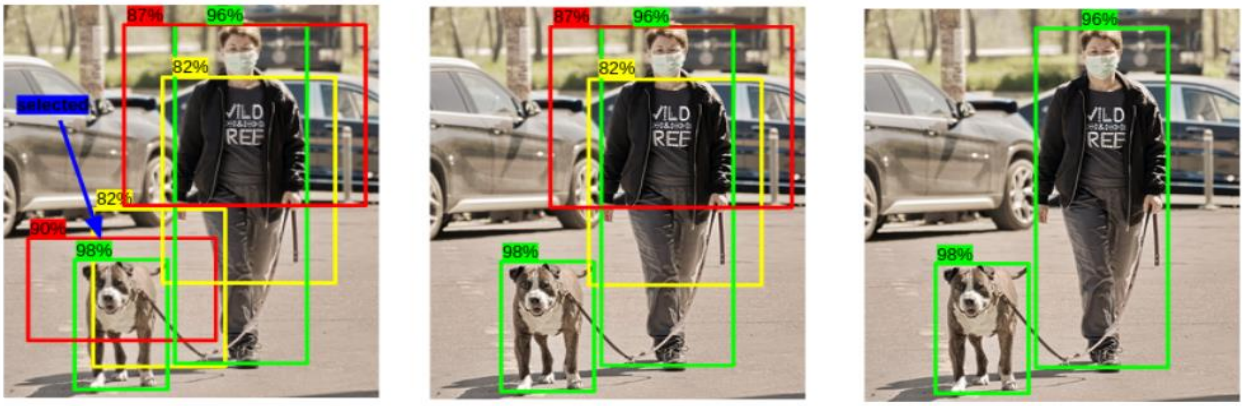


Figure 46: Non Maximum Suppression Example

3.4.7 - Single-Shot Multibox Detector (SSD)

Το προεκπαιδευμένο μοντέλο που χρησιμοποιήθηκε για την παρούσα διπλωματική εργασία είναι το SSD. Χρησιμοποιείται ευρέως για Ανίχνευση Αντικειμένων σε πραγματικό χρόνο. Το μοντέλο αυτό δημοσιεύτηκε στα τέλη Νοεμβρίου του 2016 και κατάφερε να πετύχει παραπάνω από 74% mAP στα 59 frames per second σε δημοφιλή Dataset όπως το PascalVOC και COCO. Η ονομασία του SSD δίνει σημαντικές πληροφορίες για την αρχιτεκτονική του:

1. Single Shot: Αυτό σημαίνει ότι ο εντοπισμός της τοποθεσίας και η ταξινόμηση του αντικειμένου γίνεται σε ένα «πέρασμα» (single forward pass) του δικτύου.

2. MultiBox: Είναι το όνομα της τεχνικής του Bounding Box Regression δηλαδή της εύρεσης του καλύτερου Πλαισίου Οριοθέτησης για ένα αντικείμενο.

3. Detector: Είναι ένα Συνελικτικό Νευρωνικό Δίκτυο που χρησιμοποιείται για Ανίχνευση Αντικειμένων.

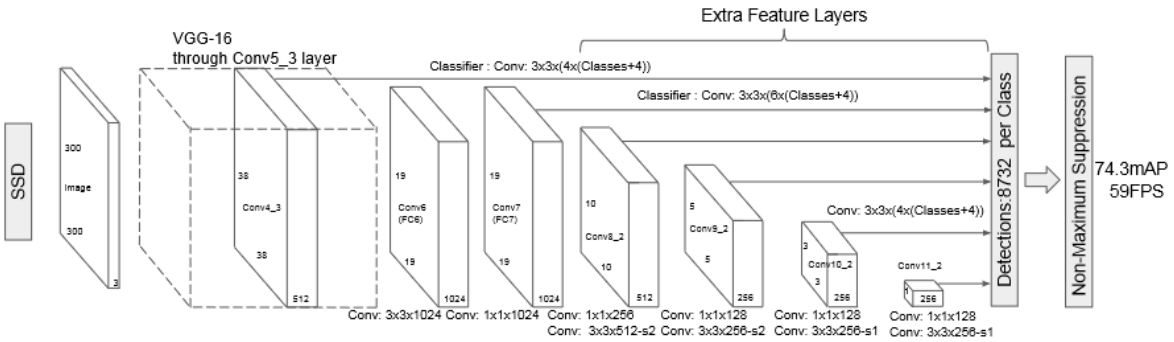


Figure 47: SSD Architecture

Όπως φαίνεται από το σχήμα η αρχιτεκτονική του SSD στηρίζεται στο VGG-16 (αρχιτεκτονική CNN για visual object recognition), χωρίς να έχει καθόλου Πλήρως Συνδεδεμένα Στρώματα. Ο λόγος που χρησιμοποιήθηκε το VGG-16 ως βάση του δικτύου είναι η καλή απόδοση και καλή ποιότητα κατηγοριοποίησης εικόνων. Αντί για Πλήρως Συνδεδεμένα Στρώματα, χρησιμοποιείται ένα σετ από βοηθητικά

Συνελικτικά Στρώματα (από το conv6 και έπειτα), τα οποία έξαγουν χαρακτηριστικά και προοδευτικά μειώνουν το μέγεθος του input και επιτρέπουν να γίνονται ανιχνεύσεις σε διαφορετικές κλίμακες.

Το SSD στηρίζεται σε ένα feed-forward CNN, το οποίο παράγει σταθερού πλήθους Πλαίσια Οριοθέτησης και πιθανότητες/σκορς για την ύπαρξη Αντικειμένων μέσα στα Πλαίσια αυτά. Στο τέλος χρησιμοποιεί τον αλγόριθμο Καταστολής Μη Μεγίστων για να παράξει τις τελικές ανιχνεύσεις. Τα πρώτα στρώματα του δικτύου ή αλλιώς βάση του δικτύου στηρίζονται στη VGG-16 αρχιτεκτονική CNN και πετυχαίνουν υψηλής ποιότητας κατηγοριοποίηση εικόνων. Έπειτα γίνεται προσθήκη βοηθητικών Συνελικτικών Στρωμάτων στο δίκτυο με τα ακόλουθα βασικά χαρακτηριστικά:

1. Προστίθενται Συνελικτικά Στρώματα διαφορετικής κλίμακας στο τέλος του δικτύου (extra features layers). Αυτά τα στρώματα προοδευτικά μειώνουν τις διαστάσεις του input και επιτρέπουν ανιχνεύσεις σε διαφορετικές κλίμακες. Το συνελικτικό μοντέλο πρόβλεψης είναι διαφορετικό για κάθε ένα από αυτά τα στρώματα.

2. Κάθε προστιθέμενο στρώμα στο τέλος του δικτύου μπορεί να παράγει συγκεκριμένο πλήθος προβλέψεων χρησιμοποιώντας ένα σετ από συνελικτικά φίλτρα.

3. Το μοντέλο παράγει Πλαίσια Οριοθέτησης σε συγκεκριμένες θέσεις και καθορισμένων διαστάσεων/σχήματος (aspect ratios) προκειμένου να κάνει προβλέψεις σε αυτά. Είναι παρόμοια λογική με τα anchor boxes που χρησιμοποιείται στην Faster R-CNN μοντέλο.

Από την αρχιτεκτονική του σχήματος φαίνεται ότι το μοντέλο μπορεί και κάνει 8732 ανιχνεύσεις για κάθε κλάση. Στην παρούσα διπλωματική θα χρησιμοποιήσουμε μια διαφοροποιημένη έκδοση από την αρχική δηλαδή την SSD MobileNet V2 FPNLite 320x320. Για τους στόχους της διπλωματικής δηλαδή την δημιουργία ενός Ανιχνευτή Αντικειμένων σε πραγματικό χρόνο θεωρούμε σκόπιμο να αποδώσουμε ποιοτικά την λειτουργία του προεκπαιδευμένου μοντέλου.

Κεφάλαιο 4 : Τεχνικό υπόβαθρο

4.1 - Python

Η Python είναι μία γλώσσα υψηλού επιπέδου(high-level), γενικού σκοπού (general-purpose). Είναι μια γλώσσα διερμηνευτή (interpreted) και δημιουργήθηκε απο τον Guido van Rossum το 1991. Είναι μια γλώσσα ανοιχτού κώδικα (open source). Υπάρχουν αρκετές εκδόσεις της, με την Python 3 να αποτελεί την τελευταία της και να επικρατεί στις περισσότερες εφαρμογές.

Υποστηρίζει πολλά προγραμματιστικά μοντέλα όπως το διαδικαστικό προγραμματισμό (Procedural programming), αντικειμενοστραφή προγραμματισμο (Object Oriented programming) και συναρτησιακό προγραμματισμό (Functional programming). Σαν προτερήματα της γλώσσας Python λογίζεται η ευκολία τόσο στη χρήση της όσο και στην αναγνωσιμότητα. Σαν γλώσσα, η Python είναι αρκετά απλή στην εκμάθηση, κάνοντας εύκολη τη συντήρηση προγραμμάτων της. Οι λόγοι που επιλέγεται κυρίως, είναι ότι δεν υπάρχει το κομμάτι της μεταγλώττισης του προγράμματος (compilation) και η αποσφαλμάτωση (debugging) γίνεται υπερβολικά γρήγορα.

4.2 - Βασικές Βιβλιοθήκες που χρησιμοποιούνται από τον Real - Time Object Detector

4.2.1 - Tensorflow 2.0

Το Tensorflow 2.0 είναι μια βιβλιοθήκη της Python που παρέχει πληθώρα από εργαλεία σε προγραμματιστές (developers), ερευνητές (researchers) και οργανισμούς που θέλουν να φτιάξουν εφαρμογές Μηχανικής Μάθησης και Βαθιάς Μηχανικής Μάθησης. Η βιβλιοθήκη αυτή δημιουργήθηκε και δημοσιεύτηκε το 2015 από την ομάδα Google Brain της Google για την δημιουργία μοντέλων τόσο μηχανικής μάθησης όσο και βαθιάς μηχανικής μάθησης.

Η αρχιτεκτονική δομή των εργασιών στο Tensorflow χωρίζεται σε τρία μέρη:

1. Προεπεξεργασία των δεδομένων

2. Δημιουργία του μοντέλου

3. Εκπαίδευση και Αξιολόγηση του μοντέλου

Όπως και η Python έτσι και το Tensorflow είναι open source. Μπορεί να εκτελεστεί σε GPU και CPU και είναι ένα από τα καλύτερα API για την δημιουργία εφαρμογών Τεχνητής Νοημοσύνης.

4.2.2 - NumPy

Η NumPy είναι μια βιβλιοθήκη ανοιχτού κώδικα (open-source) της Python. Χρησιμοποιείται στην Python κυρίως για την επεξεργασία και διαχείριση πινάκων. Εμπεριέχει μαθηματικές συναρτήσεις καθώς και βασικές μαθηματικές πράξεις. Το όνομα της κανονικά είναι Numerical Python και δημιουργήθηκε απο τον Travis Oliphant το 2005.

4.2.3 - OpenCV

Η OpenCV είναι μια βιβλιοθήκη ανοιχτού κώδικα (open-source). Χρησιμοποιείται ευρέως στον κλάδο της Όρασης Υπολογιστών και στη Μηχανική Μάθηση. Περιέχει πάνω από 2500 αλγορίθμους κλασικούς και state-of-art αλγορίθμους για την Όραση Υπολογιστών και τη Μηχανική Μάθηση. Αυτοί οι αλγόριθμοι χρησιμοποιούνται για να αναγνωρίζουν αντικείμενα ή πρόσωπα, να παρακολουθούν κινούμενα αντικείμενα, αλλαγές κ.α. Η βιβλιοθήκη δημιουργήθηκε από την Intel. Το OpenCV δημοσιεύτηκε το 2000 και έχει γραφτεί σε C/C++.

4.2.4 - Matplotlib

Είναι μια βιβλιοθήκη ανοιχτού κώδικα (open-source) της Python. Δημιουργήθηκε από τον John D. Hunter. Ο κύριος λόγος χρήσης της είναι η οπτικοποίηση δεδομένων και η παραγωγή γραφικών παραστάσεων. Παρέχει διαφορετικούς τύπους διάγραμματων όπως σχεδιάγραμμα εικόνας (image plots), σχεδιάγραμμα γραμμών (line plots), ιστογράμματα (histograms) κ.α.

4.2.5 - Tensorboard

Είναι η βιβλιοθήκη open-source και εγκαθίστανται συνήθως μαζί με το Tensorflow. Μας παρέχει δυνατότητες οπτικοποίησης και εργαλεία απαραίτητα για πειραματικές διατάξεις Μηχανικής Μάθησης. Το Tensorboard μπορεί να παρακολουθεί ένα μοντέλο κατα την περίοδο του χρόνου και να οπτικοποιεί μετρικές όπως το σφάλμα (loss) και η ακρίβεια (accuracy). Έχει την δυνατότητα να αναπαραστήσει τον γράφο του μοντέλου, παρέχει το ιστόγραμμα των βαρών (weights), των προκαταλήψεων (biases) και άλλων μετρικών. Μπορεί να αναπαραστήσει εικόνα, κείμενο, ήχο κ.α. Στην παρούσα διπλωματική εργασία θα χρησιμοποιηθεί για την παρακολούθηση τόσο της εκπαίδευσης όσο και της αξιολόγησης του μοντέλου.

Κεφάλαιο 5 : Πειραματική Διάταξη

5.0 - Εισαγωγή

Σκοπός αυτού του κεφαλαίου είναι να παρουσιάσει και να αναλύσει την ροή των εργασιών που απαιτούνται για τη δημιουργία του δικού μας Ανιχνευτή Αντικειμένων σε πραγματικό χρόνο. Θα αναφερόμαστε στον ανιχνευτή ως real-time Custom Object Detector από εδώ και στο εξής. Ο real-time Custom Object Detector της παρούσας διπλωματικής αναγνωρίζει από μια τράπουλα τέσσερα φύλλα της. Πιο συγκεκριμένα αναγνωρίζει το Ρήγα σπαθί, Ντάμα σπαθί, Βαλέ σπαθί και το μαύρο Τζοκερ. Ο real-time Custom Object Detector της παρούσας διπλωματικής στηρίζεται στο project του Nicholas Renotte με repository: https://github.com/nicknochnack/TFODCourse.

Είναι ένα project ανοιχτού κώδικα και στόχος είναι να δείξει στον καθένα πως μπορεί να φτιάξει τον δικό του ανιχνευτή χρησιμοποιώντας το Tensorflow και έτοιμα προεκπαιδευμένα μοντέλα που παρέχονται ελεύθερα στο repository του Tensorflow: https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md.

5.1 - Hardware

Η εκτέλεση, η εκπαίδευση και η αξιολόγηση των πειραμάτων έγινε σε προσωπικό υπολογιστή. Ο προσωπικός υπολογιστής διαθέτει:

- 16gb Ram χρονισμένες στα 3200mHz
- i9-9900k , χρονισμένος στα 4.7gHz
- nVidia GTX 1080ti με 11GB διαθέσιμη μνήμη
- Windows 10 , Version 10.0.19043

5.2 - Λογισμικό

SSD MobileNet V2 FPNLite 320x320: Είναι το μοντέλο, το οποίο χρησιμοποιήθηκε για την παρούσα εργασία. Προσφέρεται σαν ανοιχτός κώδικας στο repository του *TensorFlow 2 Detection Model Zoo.* Μέσω αυτού και την παραμετροποίηση του Config file μπορούμε να χτίσουμε το δικό μας Ανιχνευτή Αντικειμένων. Στο *TensorFlow 2 Detection Model Zoo* πέρα από το μοντέλο που επιλέχθηκε, προσφέρονται και άλλα μοντέλα, τα οποία μπορούν να κάνουν και αυτά Ανίχνευση

Αντικειμένων. Ο λόγος που επιλέχθηκε αυτό το μοντέλο είναι η ισορροπία στο mAP και στο speed. Μπορεί και επεξεργάζεται μια εικόνα (frame) σε 22ms δηλαδή αρκετά γρήγορα (δηλαδή μπορεί να επεξεργάζεται 45fps, όπου fps είναι τα frames per second). Επιλέγουμε το SSD μοντέλο, γιατί συμπεριφέρεται καλά σε ανίχνευση σε πραγματικό χρόνο. Από τον παρακάτω πίνακα φαίνεται ότι καθώς ο χρόνος σε ms μεγαλώνει δηλαδή η επεξεργασία της φωτογραφίας αργεί τότε η μετρική mAP μεγαλώνει, δηλαδή το απόδοση/ακρίβεια βελτιώνεται. Αυτό σημαίνει ότι έχουμε ένα trade-off ανάμεσα σε ταχύτητα και αποτελεσματικότητα. Οι μετρικές του παρακάτω πίνακα έχουν προκύψει από την εκπαίδευση των προεκπαιδευμένων μοντέλων στο COCO Dataset.

Model name	Speed (ms)	COCO mAP	Outputs
SSD MobileNet v2 320x320	19	20.2	Boxes
SSD MobileNet V1 FPN 640x640	48	29.1	Boxes
SSD MobileNet V2 FPNLite 320x320	22	22.2	Boxes
SSD MobileNet V2 FPNLite 640x640	39	28.2	Boxes
SSD ResNet50 V1 FPN 640x640 (RetinaNet50)	46	34.3	Boxes
SSD ResNet50 V1 FPN 1024x1024 (RetinaNet50)	87	38.3	Boxes
SSD ResNet101 V1 FPN 640x640 (RetinaNet101)	57	35.6	Boxes
SSD ResNet101 V1 FPN 1024x1024 (RetinaNet101)	104	39.5	Boxes

Figure 48:Pre-trained Models

Επίσης στο πίνακα φαίνεται ότι τα μοντέλα αυτά δίνουν σαν output Πλαίσια Οριοθέτησης που είναι τα αποτελέσματα που θέλουμε προκειμένου να γίνει η Ανίχνευση Αντικειμένων.

5.3 - Στάδια Δημιουργίας του Object Detector

5.3.1 - Συλλογή Φωτογραφιών

Για να δημιουργήσουμε το δικό μας real-time Custom Object Detector απαιτείται συλλογή φωτογραφιών . Οι φωτογραφίες τραβήχτηκαν με τη χρήση κινητού τηλεφώνου (Redmi Note 8 Pro). Συλλέχθηκαν συνολικά 130 φωτογραφίες :

- 33 για την κλάση του Βαλέ σπαθί με όνομα κλάσης Jack.
- 33 για την κλάση του μαύρου Τζοκερ με όνομα κλάσης Joker.
- 34 για την κλάση του Ρήγα σπαθί με όνομα κλάσης King.

- 30 για την κλάση της Ντάμας σπαθί με όνομα κλάσης Queen.

Ο λόγος που περιοριστήκαμε σε 130 φωτογραφίες είναι καθαρά υπολογιστικός. Όσο περισσότερα δεδομένα αποδοθούν στο μοντέλο τόσο το καλύτερο. Όμως ένα μεγάλο σύνολο δεδομενων, πέρα από περισσότερο χρόνο εκπαίδευσης χρειάζεται και περισσότερους υπολογιστικούς πόρους, για να μπορεί να κάνει γενικεύσεις. Παρόλα αυτά, στο σύνολο δεδομένων που δόθηκε, το μοντέλο που δημιουργήθηκε κατάφερε να πετύχει καλό mAP και να αναγνωρίζει αρκετά ικανοποιητικά τα αντικείμενα, τα οποία τροφοδοτούνται από την web-camera. Οι διαστάσεις των φωτογραφιών που επιλέχθηκαν είναι μικρότερες από 1280x720 προκειμένου το μέγεθος να μην είναι πολύ μεγάλο, καθώς αυτό επιβραδύνει την διαδικασία εκπαίδευσης. Οι φωτογραφίες τραβήχτηκαν σε τυχαία μέρη του σπιτιού με τυχαία κλήση και τυχαία απόσταση από την κάμερα, ώστε να το μοντέλο μας να κάνει καλές γενικεύσεις. Δεν είναι όλες οι φωτογραφίες των ίδιων διαστάσεων, αφού ορισμένες έχουν επεξεργαστεί και πιο συγκεκριμένα υποστεί περικοπή και περιστροφή (crop & rotate). Ξεκινήσαμε τη διαδικασία από τις 40 φωτογραφίες και πιο συγκεκριμένα 10 για κάθε κλάση και αυξάναμε σταδιακά το σύνολο δεδομένων προκειμένου να γίνει πιο αποτελεσματικό το μοντέλο. Συνολικά το μοντέλο που φτιάξαμε έχει τέσσερις εκδόσεις.

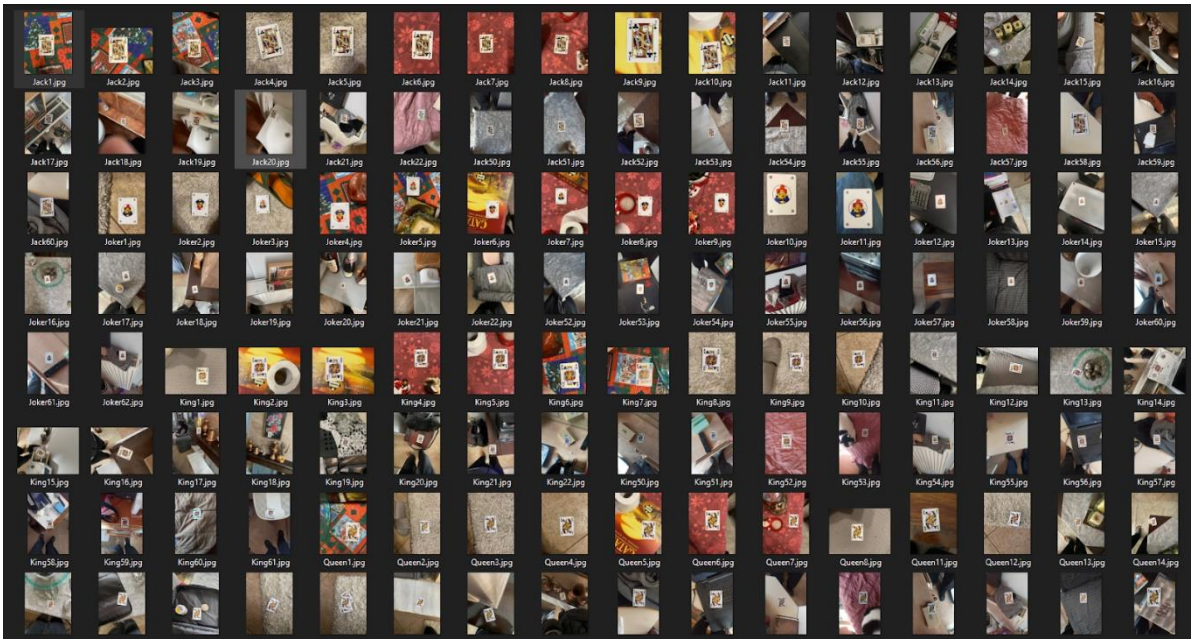


Figure 49: Dataset

5.3.2 - Δημιουργία ετικετών

Οι φωτογραφίες για να τροφοδοτηθούν στο προεκπαιδευμένο μοντέλο είναι απαραίτητο να υποστούν labelling. Για το labelling των φωτογραφιών απαιτείται συνήθως πρόγραμμα επεξεργασίας εικόνας ή κάποιο API συλλογής ετικετών. Στην παρούσα διπλωματική χρησιμοποιήσαμε το LabelIMG. Για κάθε μία από τις φωτογραφίες απαιτείται καθορισμός του ground truth, δηλαδή ο προσδιορισμός του Πλαισίου Οριοθέτησης και χαρακτηρισμός της κλάσης. Γραφικά το περιβάλλον του LabelIMG φαίνεται στην παρακάτω εικόνα:

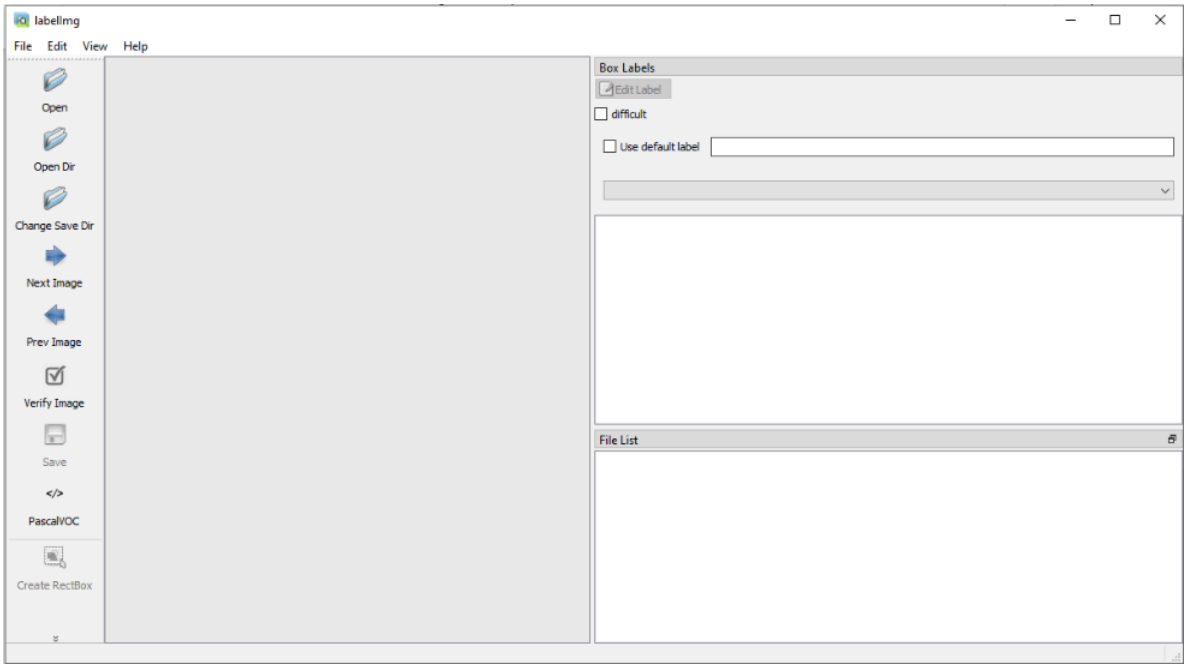


Figure 50: LabelImg

Για όλες τις φωτογραφίες μας επιλέγουμε το PascalVOC annotation (προτελευταία γραμμή στην πρώτη στήλη της εικόνας). Ύστερα περιηγούμαστε μέσω της επιλογής Open Dir στο φάκελο που περιέχει τις φωτογραφίες μας και για κάθε φωτογραφία ορίζουμε το Πλαίσιο Οριοθέτησης και την κλάση. Πατώντας το πλήκτρο W, δίνεται η επιλογή για σχεδιασμό του Πλαισίου Οριοθέτησης που περιέχει το αντικείμενο πάνω στην εικόνα. Αφού σχεδιαστεί το Πλαίσιο Οριοθέτησης, το πρόγραμμα ζητά να οριστεί η κλάση.

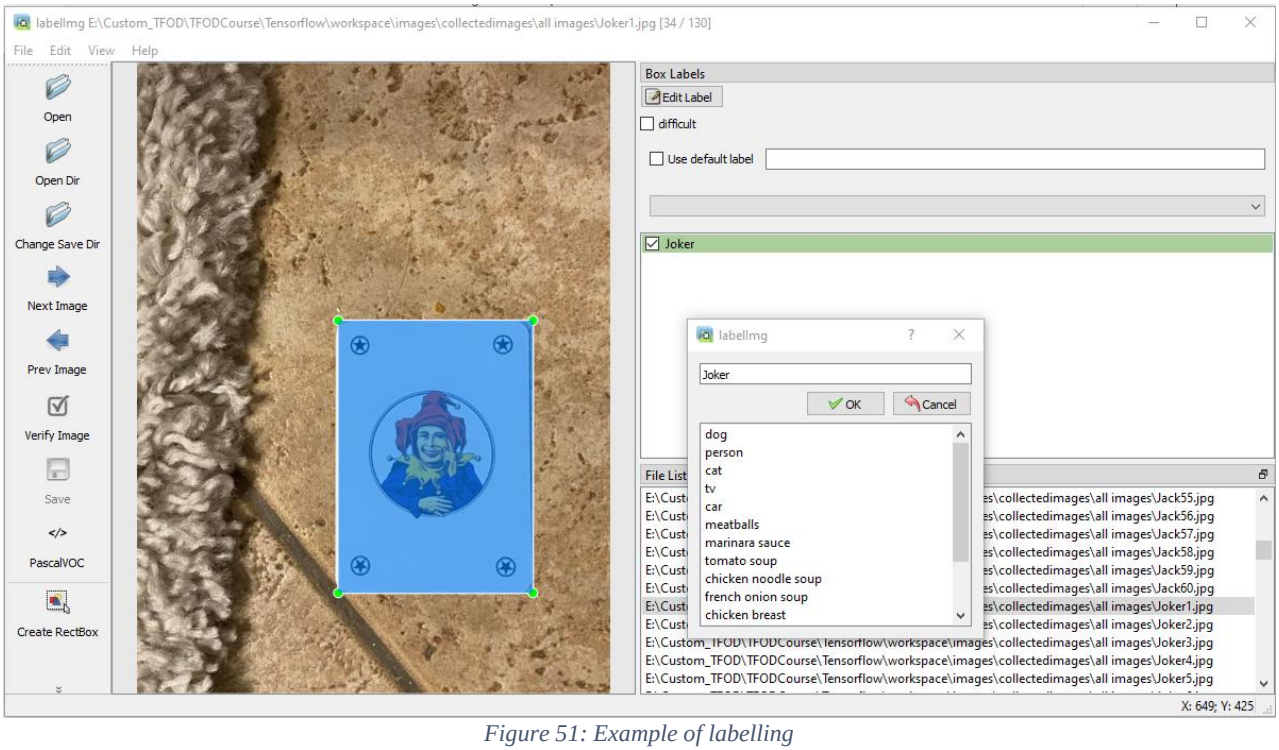


Figure 51: Example of labelling

Αφού συμπληρώσουμε και την κλάση, αποθηκεύουμε το label της κάθε φωτογραφίας και προκύπτουν xml αρχεία των labels, τα οποία έχουν το ίδιο όνομα με τη φωτογραφία μας.

Joker1.xml

```xml
<annotation>
    <folder>joker</folder>
    <filename>Joker1.jpg</filename>
    <path>E:\Custom_TFOD\TFODCourse\Tensorflow\workspace\images\collectedimages\joker\Joker1.jpg</path>
    <source>
        <database>Unknown</database>
    </source>
    <size>
        <width>618</width>
        <height>824</height>
        <depth>3</depth>
    </size>
    <segmented>0</segmented>
    <object>
        <name>Joker</name>
        <pose>Unspecified</pose>
        <truncated>0</truncated>
        <difficult>0</difficult>
        <bndbox>
            <xmin>263</xmin>
            <ymin>337</ymin>
            <xmax>517</xmax>
            <ymax>693</ymax>
        </bndbox>
    </object>
</annotation>

```

Figure 52:Annotation file example

Το xml αρχείο έχει τις απαραίτητες πληροφορίες για κάθε εικόνα. Πιο συγκεκριμένα, το πεδίο <size> δηλώνει τις διαστάσεις της φωτογραφίας. Το πεδίο <object> δηλώνει το αντικείμενο που περιέχεται μέσα στην εικόνα. Μέσα στο πεδίο <object> περιέχεται το όνομα της κλάσης στο πεδίο <name> και το Πλαίσιο Οριοθέτησης στο πεδίο <bndbox> που το περιβάλλει το αντικείμενο. Το Πλαίσιο Οριοθέτησης προσδιορίζεται από τα σημεία $A(x_{min}, y_{min})$ και $B(x_{max}, y_{max})$. Για το σχεδιασμό του Πλαισίου Οριοθέτησης προσπαθούμε να βρούμε το μικρότερο πλαίσιο που περιβάλλει ολόκληρο το αντικείμενο προκειμένου να περιέχει τον λιγότερο δυνατό θόρυβο.

5.3.3 - Δημιουργία των Label Map

Η δημιουργία του αρχείου label_map.pbtxt είναι απαραίτητη, αφού αυτό το αρχείο ορίζει τις κλάσεις. Είναι απαραίτητο να είναι σε συγκεκριμένη μορφή έτσι ώστε να μπορούν να παραχθούν τα TFRecords. Η δομή του αρχείου label_map.pbtxt φαίνεται στο παρακάτω σχήμα.

label_map.pbtxt

```
item {
    name:'Joker'
    id:1
}
item {
    name:'Jack'
    id:2
}
item {
    name:'King'
    id:3
}
item {
    name:'Queen'
    id:4
}
```

Figure 53:Label Map

Το item περιγράφει τις κλάσεις. Μέσα σε αυτό το πεδίο name περιγράφει το όνομα της κλάσης και το πεδίο id το αριθμητικό χαρακτηριστικό αυτής.

5.3.4 - Χωρισμός των Δεδομένων σε Δεδομένα Εκπαίδευσης και δημιουργία των TFRecords

Σε αυτό το στάδιο είναι απαραίτητος ο διαχωρισμός των φωτογραφιών με τα xml αρχεία που τα συνοδεύουν σε δεδομένα αξιολόγησης και εκπαίδευσης. Ιδανικά, θέλουμε τα δεδομένα εκπαίδευσης να αποτελούν το 80-90% και τα δεδομένα αξιολόγησης το υπόλοιπο 10-20% των συνολικών δεδομένων, χωρίς όμως αυτός ο διαχωρισμός να αποτελεί κανόνα.

Για να γίνει η διαδικασία της εκπαίδευσης και της αξιολόγησης στο Tensorflow 2.0 είναι απαραίτητη η μετατροπή των φωτογραφιών και των xml αρχείων σε ένα τύπο αρχείων (format) που μπορεί να χειριστεί το Tensorflow δηλαδή TFRecords. Η διαδικασία της εκπαίδευσης γίνεται υπολογιστικά πιο φθηνή με την δημιουργία και τη χρήση των TFRecords. Για την δημιουργία των TFRecords χρησιμοποιήθηκε το πρόγραμμα *generate_tfrecord.py*, το οποίο υπάρχει και παρέχεται ελεύθερα στο TensorFlow 2 Object Detection API. Αρχικά δημιουργούμε το TFRecord των δεδομένων εκπαίδευσης και έπειτα δημιουργούμε το TFRecord των δεδομένων αξιολόγησης. Για να δημιουργηθούν αυτά τα δύο αρχεία πρέπει τα labels στα xml αρχεία να συμφωνούν με τα labels στο label_map.pbtxt.

5.3.5 - Παραμετροποίηση του προεκπαιδευμένου μοντέλου

Αφού κατεβάσουμε το προεκπαιδευμένο μοντέλο πρέπει να παραμετροποιήσουμε το pipeline του, έτσι ώστε να δημιουργήθει ο δικός μας Ανιχνευτής Αντικειμένων. Τα στοιχεία που παραμετροποιήθηκαν στο pipeline.config αρχείο είναι τα εξής:

1. Ορίσαμε τον αριθμό των κλάσεων σε 4 όσες είναι και οι κλάσεις στο label_map.pbtxt.

   num_classes:4

2. Ρίξαμε τον αριθμό του batch_size από 128 που είναι το default σε 4. Η τιμή αυτού του πεδίου σχετίζεται με την υπολογιστική ισχύ. Επειδή πρόκειται για ένα προσωπικό υπολογιστή η τιμή που προτείνεται είναι 4. Η τιμή του batch_size ορίζει το σύνολο των αρχείων που θα επεξεργαστεί το μοντέλο ταυτόχρονα.

   batch_size:4

3. Ορίσαμε το αρχικό σημείο (checkpoint) από το οποίο ξεκινά η εκπαίδευση.

   fine_tune_checkpoint:"Tensorflow\\workspace\\pre-trained-models\\ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8\\checkpoint\\ckpt-0"

4. Ορίσαμε ότι με το μοντέλο θέλουμε να κάνουμε ανίχνευση αντικειμένων.

   fine_tune_checkpoint_type: "detection"

5. Δίνουμε τα path των Labels:

   label_map_path: "Tensorflow\\workspace\\annotations\\label_map.pbtxt"

6. Δίνουμε τα paths των train και testing TFRecords:

   Για το train:
   tf_record_input_reader {
    input_path: "Tensorflow\\workspace\\annotations\\train.record"
   }

   Για το test:
   tf_record_input_reader {
    input_path: "Tensorflow\\workspace\\annotations\\test.record"
   }

5.3.6 - Διαδικασία Εκπαίδευσης του μοντέλου

Αφού έχουμε ορίσει όλα τα παραπάνω είμαστε σε θέση να ξεκινήσουμε την διαδικασία της εκπαίδευσης. Για να γίνει αυτό στο command line εκτελούμε την εντολή:

```
python Tensorflow\models\research\object_detection\model_main_tf2.py
--model_dir=Tensorflow\workspace\models\my_ssd_mobnetV4 --
pipeline_config_path=Tensorflow\workspace\models\my_ssd_mobnetV4\pipeline.co
nfig
--num_train_steps=7000
```

Για να γίνει αυτό μέσα από το φάκελο του Tensorflow εκτελούμε το *model_main_tf2.py* και δίνουμε σαν ορίσματα τον φάκελο που θα γίνει αποθήκευση το Tensorflow, το pipeline και τα training steps. Σαν training step λογίζεται ένα update στις παραμέτρους (weights) του δικτύου από τον αλγόριθμο Gradient. Όσο περισσότερα είναι τα training steps τόσο καλύτερα «μαθαίνει» το μοντέλο μας τα δεδομένα εκπαίδευσης. Υπερβολικά μεγάλος αριθμός στο πεδίο αυτό θα οδηγήσει σε υπερεκπαίδευση του μοντέλου (overfitting), δηλαδή πέρα από την μορφολογική δομή των δεδομένων, το μοντέλο θα αφομοιώσει και τον θόρυβο που περιέχουν.

Με το πέρας της διαδικασίας παράγονται τα αρχεία όπως φαίνεται στην παρακάτω εικόνα.

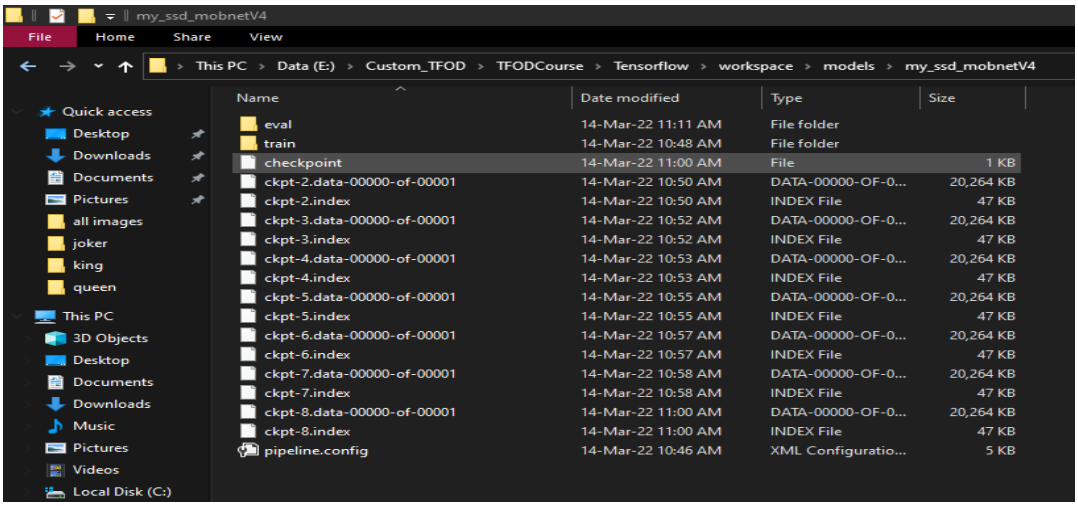


Figure 54: Checkpoint files

Τα checkpoints (π.χ. ckpt-8.index) έχουν όλες τις τιμές για τις παραμέτρους του δικτύου (weights). Ο αριθμός μετά τη παύλα δείχνει το πότε παρήχθησαν. Πρώτα δημιουργείται το ckpt-2, από αυτό έπειτα δημιουργείται το ckpt-3 κτλπ. Για τον Ανιχνευτή μας επιλέγουμε το τελευταίο checkpoint δηλαδή ckpt-8.index. Κατά την εκτέλεση της εκπαίδευσης το Tensorflow παρέχει εποπτεία των συναρτήσεων σφάλματος (loss functions). Βλέπουμε ότι με την πάροδο του χρόνου και την εκτέλεση ολοένα και περισσότερων steps τα σφάλματα μειώνονται κάτι που είναι

φυσιολογικό, αφού μετα από κάθε step το μοντέλο προσαρμόζεται και μαθαίνει καλύτερα τα δεδομένα.

```
 'learning_rate': 0.07910804}
I0314 18:04:03.419913 15252 model_lib_v2.py:708] {'Loss/classification_loss': 0.049315836,
 'Loss/localization_loss': 0.018583678,
 'Loss/regularization_loss': 0.13845651,
 'Loss/total_loss': 0.20635602,
 'learning_rate': 0.07910804}
INFO:tensorflow:Step 4400 per-step time 0.116s
I0314 18:04:14.759742 15252 model_lib_v2.py:705] Step 4400 per-step time 0.116s
INFO:tensorflow:{'Loss/classification_loss': 0.058454394,
 'Loss/localization_loss': 0.026505867,
 'Loss/regularization_loss': 0.13860226,
 'Loss/total_loss': 0.22356251,
 'learning_rate': 0.07905338}
I0314 18:04:14.760740 15252 model_lib_v2.py:708] {'Loss/classification_loss': 0.058454394,
 'Loss/localization_loss': 0.026505867,
 'Loss/regularization_loss': 0.13860226,
 'Loss/total_loss': 0.22356251,
 'learning_rate': 0.07905338}
INFO:tensorflow:Step 4500 per-step time 0.115s
I0314 18:04:26.097590 15252 model_lib_v2.py:705] Step 4500 per-step time 0.115s
INFO:tensorflow:{'Loss/classification_loss': 0.04041533,
 'Loss/localization_loss': 0.016921988,
 'Loss/regularization_loss': 0.137954,
 'Loss/total_loss': 0.19529131,
 'learning_rate': 0.07899711}
I0314 18:04:26.097590 15252 model_lib_v2.py:708] {'Loss/classification_loss': 0.04041533,
 'Loss/localization_loss': 0.016921988,
 'Loss/regularization_loss': 0.137954,
 'Loss/total_loss': 0.19529131,
 'learning_rate': 0.07899711}
INFO:tensorflow:Step 4600 per-step time 0.114s
I0314 18:04:37.674808 15252 model_lib_v2.py:705] Step 4600 per-step time 0.114s
INFO:tensorflow:{'Loss/classification_loss': 0.08658714,
 'Loss/localization_loss': 0.017134264,
 'Loss/regularization_loss': 0.13719963,
 'Loss/total_loss': 0.24092102,
 'learning_rate': 0.078939244}
I0314 18:04:37.675806 15252 model_lib_v2.py:708] {'Loss/classification_loss': 0.08658714,
 'Loss/localization_loss': 0.017134264,
 'Loss/regularization_loss': 0.13719963,
 'Loss/total_loss': 0.24092102,
 'learning_rate': 0.078939244}
```

Figure 55:Output during Training

5.3.6 - Αξιολόγηση Αποτελεσμάτων

Υλοποιήθηκαν 4 versions για τον Ανιχνευτή μας. Σταδιακά όπως είπαμε και προηγουμένως αυξάναμε το σύνολο δεδομένων (των δεδομένων εκπαίδευσης και αξιολόγησης), προκειμένου να δούμε τη συμπεριφορά των διαφόρων μετρικών. Εστιάζουμε στις versions 3 και 4 του μοντέλου. Το μοντέλο v3 έχει λιγότερα δεδομένα εκπαίδευσης από το v4. Και οι δύο εκδόσεις εκτελούνται σε 7000 training steps και έχουν το ίδια δεδομένα αξιολόγησης. Όπως φαίνεται παρακάτω η αύξηση των δεδομένων εκπαίδευσης οδηγεί σε επίτευξη υψηλότερου mAP και άρα το μοντέλο v4 είναι πιο αξιόπιστο/ακριβές μοντέλο. Όπως φαίνεται η μετρική mAP υπολογίζεται για τα διάφορα IoU.

Version 4 Evaluation mAP:

```
Accumulating evaluation results...
DONE (t=0.02s).
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.823
 Average Precision  (AP) @[ IoU=0.50       | area=   all | maxDets=100 ] = 0.983
 Average Precision  (AP) @[ IoU=0.75       | area=   all | maxDets=100 ] = 0.983
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = -1.000
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.830
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.863
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.863
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.863
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = -1.000
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.863
INFO:tensorflow:Eval metrics at step 7000
```

Figure 56: Evaluation Metrics V4

Version 3 Evaluation mAP:

```
Accumulating evaluation results...
DONE (t=0.02s).
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.798
 Average Precision  (AP) @[ IoU=0.50       | area=   all | maxDets=100 ] = 0.963
 Average Precision  (AP) @[ IoU=0.75       | area=   all | maxDets=100 ] = 0.963
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = -1.000
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.798
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.844
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.844
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.844
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = -1.000
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.844
INFO:tensorflow:Eval metrics at step 7000
```

Figure 57: Evaluation Metrics V3

Παραθέτουμε επίσης από το Tensorboard τις φωτογραφίες που συγκρίνουν τα predictions (αριστερή εικόνα) με το ground truths (δεξιά εικόνα) των εικόνων αξιολόγησης.

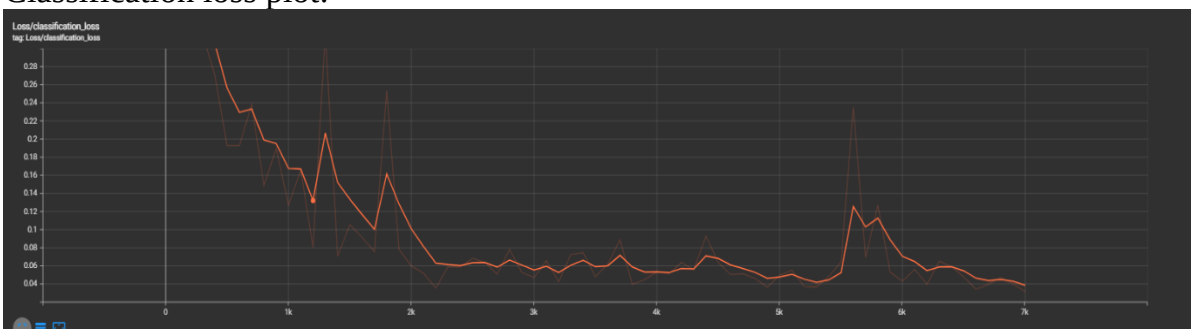
Figure 58: Jack Prediction - Ground Truth

Figure 59:King Prediction - Ground Truth

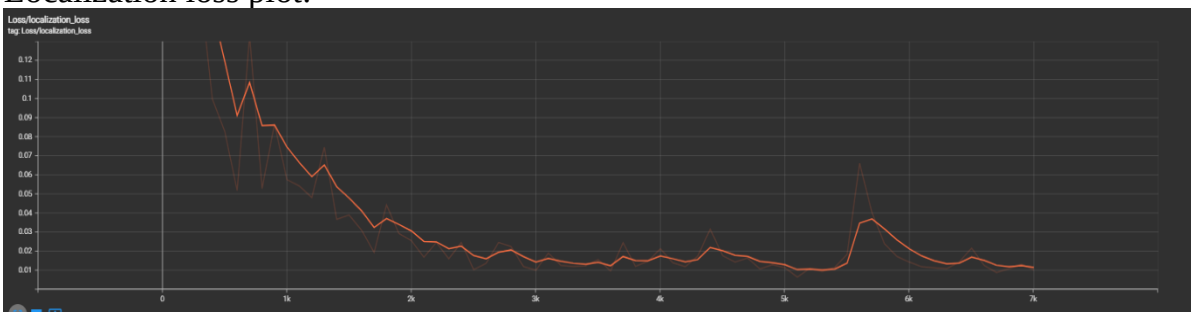
Figure 60:Joker Prediction - Ground Truth

Παραθέτουμε επίσης τα γραφήματα των loss function για την V4 του μοντέλου μας. Βλέπουμε ότι με την πάροδο του χρόνου και με την εκτέλεση ολοένα και περισσότερων steps τα σφάλματα τείνουν να πέσουν.

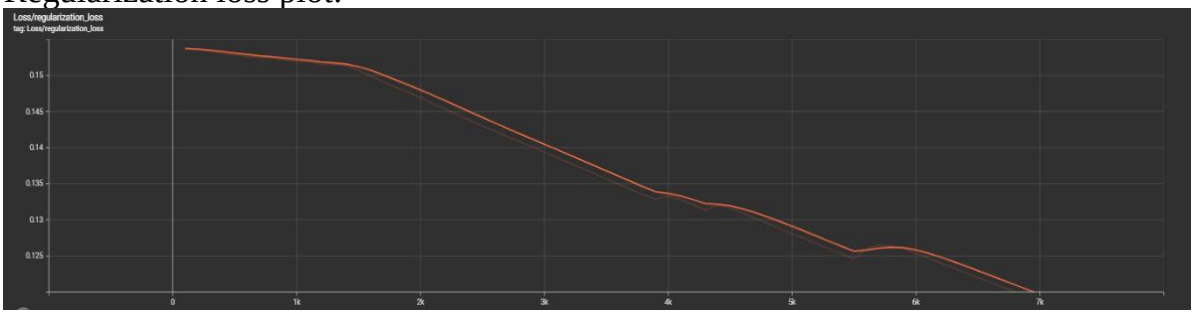
Classification loss plot:



Localization loss plot:



Regularization loss plot:



Παρατηρούμε ότι με την εκτέλεση όλων και περισσοτέρων steps όλα τα σφάλματα φθίνουν πράγμα που σημαίνει ότι το μοντέλο κάνει καλύτερες γενικεύσεις.

5.3.7 – Ανίχνευση σε Πραγματικό Χρόνο (Real-Time Object Detection)

Χρησιμοποιώντας τον οδηγό-κώδικα που παρέχει στο github ο Nicholas Rennote καταφέραμε να πετύχουμε Ανίχνευση σε Πραγματικό Χρόνο. Για τη σύνδεση χρειάστηκε να προσαρμόσουμε κατάλληλα όλα τα paths, να συνδέσουμε την κάμερα και να ορίσουμε το checkpoint που προέκυψε από την εκπαίδευση.

Η δυνατότητα να επεξεργάζεται πολύ γρήγορα το προεκπαιδευμένο μοντέλο τις εικόνες είναι αυτό που το κάνει ιδανικό για Ανίχνευση Αντικειμένων σε πραγματικό χρόνο. Παραθέτουμε τον κώδικα που χρησιμοποιήθηκε:

```
# Load pipeline config and build a detection model
configs = config_util.get_configs_from_pipeline_file(files['PIPELINE_CONFIG'])
detection_model = model_builder.build(model_config=configs['model'], is_training=False) #Using the config to make sure that its

# Restore checkpoint
ckpt = tf.compat.v2.train.Checkpoint(model=detection_model)
ckpt.restore(os.path.join(paths['CHECKPOINT_PATH'], 'ckpt-8')).expect_partial() #taking the latest checkpoint from training

#Detect function (input is an image and output all detections)
@tf.function
def detect_fn(image):
    image, shapes = detection_model.preprocess(image) #resing the image to 320x320 so that the model can perform predictions
    prediction_dict = detection_model.predict(image, shapes) #make the predictions
    detections = detection_model.postprocess(prediction_dict, shapes)
    return detections #returning all the predictions
```

Figure 61: detect_fn

Η συνάρτηση detect_fn δέχεται σαν input μια εικόνα σε μορφή tensor. Έπειτα προσαρμόζει το ύψος και το πλάτος, ώστε οι διαστάσεις να γίνουν 320x320, με σκοπό να τροφοδοτηθεί στο προεκπαιδευμένο μοντέλο. Κάνει την πρόβλεψη και επιστρέφει ένα dictionary που περιέχει όλα τα Πλαίσια Οριοθέτησης των προβλέψεων, όλα τα scores και κλάσεις για καθένα από τα αντίστοιχα Πλαίσια Οριοθέτησης.

```
cap = cv2.VideoCapture(0)


while cap.isOpened():
    ret, frame = cap.read() #capture the frame
    image_np = np.array(frame) #turn image into a numpy array in order to work with Tensorflow

    input_tensor = tf.convert_to_tensor(np.expand_dims(image_np, 0), dtype=tf.float32) #turn the array into tensor so that it car
    detections = detect_fn(input_tensor)

    num_detections = int(detections.pop('num_detections'))
    detections = {key: value[0, :num_detections].numpy()
                  for key, value in detections.items()}
    detections['num_detections'] = num_detections

    # detection_classes should be ints.
    detections['detection_classes'] = detections['detection_classes'].astype(np.int64)

    label_id_offset = 1 # first label is number 0
    image_np_with_detections = image_np.copy()

    viz_utils.visualize_boxes_and_labels_on_image_array(
                image_np_with_detections,
                detections['detection_boxes'],
                detections['detection_classes']+label_id_offset,
                detections['detection_scores'],
                category_index,
                use_normalized_coordinates=True,
                max_boxes_to_draw=4,
                min_score_thresh=.75,
                agnostic_mode=False)

    cv2.imshow('object detection',  cv2.resize(image_np_with_detections, (1000, 800))) #dimensions of the pop up window

    if cv2.waitKey(10) & 0xFF == ord('q'): #exit buy pressing Q button
        cap.release()
        cv2.destroyAllWindows()
        break
```

Figure 62:Real Time Object Detection

Το παραπάνω πρόγραμμα αποτελεί το πρόγραμμα του Real-Time Object Detection. Στην αρχή συνδέουμε την κάμερα. Από την κάμερα παίρνουμε κάθε frame, το μετατρέπουμε σε tensor, ώστε να μπορεί να γίνει η Ανίχνευση Αντικειμένου από τη συνάρτηση detect_fn. Η συνάρτηση που αλλάζει την εικόνα, ώστε να φαίνονται και τα αντικείμενα είναι η viz_utils.visualize_boxes_and_labels_on_image_array. Αυτή

δέχεται το αρχικό frame, τα Πλαίσια Οριοθέτησης, πιθανότητες/σκορς και κλάσεις που έχουν προκύψει από τις προβλέψεις της detect_fn και εμφανίζει τα Πλαίσια Οριοθέτησης, πιθανότητες/σκορς και κλάσεις που έχουν πιθανότητα/σκορ μεγαλύτερη από το min_score_thresh. Έχουμε ορίσει να φαίνονται το πολύ 4 Πλαίσια Οριοθέτησης όσα και τα max_boxes_to_draw.

Παραθέτουμε κάποια στιγμιότυπα από την εφαρμογή που υλοποιήθηκε.

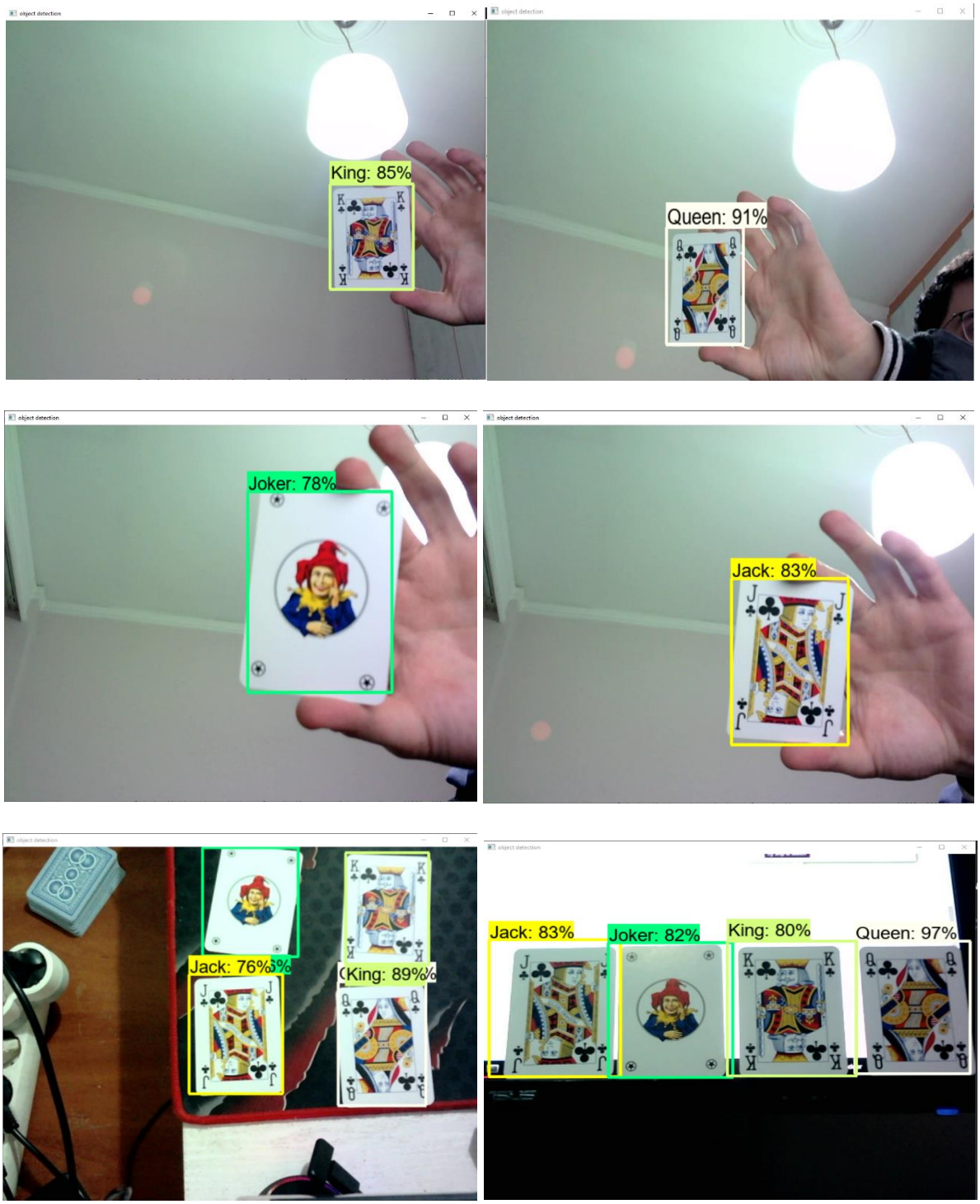


Figure 63: Examples of Real-Time Object Detection

Κεφάλαιο 6 : Όραση Υπολογιστών και Ηθική

Όπως έχει γίνει πλέον αντιληπτό από τα προεκτιθέμενα, η Όραση Υπολογιστών και γενικότερα η Τεχνητή Νοημοσύνη αποτελούν σημαντικά εργαλεία για την επιστημονική έρευνα, αλλά και για την αντιμετώπιση πρακτικών καθημερινών προβλημάτων. Είναι φανερό ότι βρισκόμαστε στην απαρχή μιας εποχής που συμβατικές τακτικές δίνουν τη θέση τους σε ρηξικέλευθες τεχνολογίες. Κάθε αλλαγή ωστόσο, συνοδεύεται από πληθώρα νέων, αδιανόητων μέχρι τώρα, πολλές φορές, προβληματισμών. Έτσι και με όλες τις τελευταίες εξελίξεις στο πεδίο της Όρασης Υπολογιστών και της Τεχνητής Νοημοσύνης, εγείρεται μια σειρά ηθικών διλημμάτων και ζητημάτων που χρίζουν αντιμετώπισης. Είναι αυτονόητο ότι τα σπουδαία μέσα που μας παρέχουν δεν πρέπει να μας αποπροσανατολίζουν από την επιτακτική ανάγκη συνετής και ρυθμιζόμενης χρήσης τους. Η αλόγιστη και αυθαίρετη χρήση των ανωτέρω τεχνολογιών μπορεί να θέσει σε κίνδυνο ουσιώδεις συλλογικές κατακτήσεις χρόνων και να πληγώσει ανεπανόρθωτα ιδανικά και αξίες πάνω στις οποίες έχουν δομηθεί οι σύγχρονες κοινωνίες. Σε αυτό το κεφάλαιο θα προσπαθήσουμε να προσεγγίσουμε βασικά ηθικά ζητήματα που ανακύπτουν κατά την επεξεργασία δεδομένων από έξυπνα υπολογιστικά ζητήματα μέσω της τεχνολογίας της Όρασης Υπολογιστών.

Απάτες: Η τεχνολογία αναγνώρισης προσώπου μπορεί να διευκολύνει ουσιωδώς την καθημερινότητά μας, απλοποιώντας τη διαδικασία πρόσβασης σε προσωπικές συσκευές και λογαριασμούς. Το πλεονέκτημα της ταχύτητας δεν είναι το μοναδικό ωστόσο, καθώς η σύνδεση της δυνατότητας εισόδου σε συστήματα βάσει ατομικών, μοναδικών χαρακτηριστικών, όπως τα εκάστοτε χαρακτηριστικά του προσώπου, αντί της χρήσης μοτίβων και κωδικών (οι οποίοι συχνά υποκλέπτονται ή λησμονούνται) καθιστά τη διαδικασία ακόμα πιο ασφαλή. Μια δεύτερη ανάγνωση του ζητήματος, ωστόσο, θα μας αποκαλύψει περιπτώσεις απατεώνων και χάκερς, οι οποίοι έχουν καταφέρει να ξεγελάσουν συστήματα αναγνώρισης προσώπου με σκοπό να αποκτήσουν πρόσβαση σε λογαριασμούς που δεν τους ανήκουν αποσπώντας προσωπικά δεδομένα ή χρηματικά ποσά. Δυστυχώς τα συστήματα αυτά παρουσιάζουν ακόμα κενά που επιτρέπουν τη χειραγώγησή τους, με συνέπεια την προσπέλαση λογαριασμών από άτομα με κακόβουλους σκοπούς που δεν συνδέονται με αυτούς με κανέναν τρόπο.

Προκατάληψη - Ρατσισμός: Μια κοινή χρήση που έχει ανά διαστήματα προταθεί για την τεχνολογία αναγνώρισης προσώπου, είναι η χρήση της ανωτέρω τεχνολογίας για την αναγνώριση πιθανών εν δυνάμει εγκληματιών σε μια προσπάθεια μείωσης της εγκληματικότητας. Ωστόσο, αξίζει να σημειωθεί ότι έχει παρατηρηθεί ότι συστήματα αναγνώρισης προσώπου τείνουν να κάνουν περισσότερα λάθη στην αναγνώριση έγχρωμων και ασιατικών προσώπων σε σχέση με πρόσωπα λευκών. Επίσης τέτοια συστήματα επιδρούν αρνητικά σε έρευνες και στατιστικές, διότι είναι ικανά να ανιχνεύουν με μεγαλύτερη ευκολία λευκούς μεσήλικες σε σχέση με άτομα άλλων φυλών, παιδιά και γέρους. Αυτό έχει ως αποτέλεσμα, να διαιωνίζονται

αναχρονιστικά στερεότυπα και να προωθούνται ρατσιστικές συμπεριφορές, όπως φυσικά και να ενοχοποιούνται αβασίμως άτομα με καταστροφικές συνέπειες για την υπόληψη, την κοινωνική θέση και τον βιοπορισμό τους. Αυτός είναι ο βασικός λόγος που τέτοιες τεχνολογίες αντιμετωπίζονται με μεγάλο σκεπτικισμό και εν τέλει δεν χρησιμοποιούνται σε αρκετές Πολιτείες της Αμερικής.

Παραβιάσεις Νομικού Πλαισίου: Η αναγνώριση προσώπου καθώς και η βιντεοπαρακολούθηση στηρίζεται στην συλλογή προσωπικών δεδομένων. Τα δεδομένα αυτά έπειτα υφίστανται επεξεργασία προκειμένου να εξαχθεί το επιθυμητό αποτέλεσμα. Ωστόσο, στις περισσότερες περιπτώσεις, οι συνθήκες κάτω από τις οποίες αποθηκεύονται και αξιοποιούνται αυτά τα δεδομένα παραμένουν αδιαφανείς, με αποτέλεσμα η επεξεργασία και κατοχή τέτοιων δεδομένων να παραβιάζει τον νόμο GDPR (Γενικός Κανονισμός Προστασίας Δεδομένων Προσωπικού Χαρακτήρα). Η αποθήκευση των δεδομένων επίσης τις περισσότερες φορές γίνεται σε τοπικούς διακομιστές, γεγονός που συνεπάγεται σοβαρούς κινδύνους για την ασφάλεια των δεδομένων εξαιτίας κυβερνοεπιθέσεων που επιτρέπουν τα κενά ασφαλείας των εν λόγω διακομιστών. Για τη διασφάλιση των προσωπικών δεδομένων κρίνεται απαραίτητη η κρυπτογράφησή τους. Βασικό ζήτημα που ανακύπτει επιπλέον με τη χρήση της τεχνολογίας αναγνώρισης προσώπου, είναι το πολύ συχνό φαινόμενο της απουσίας συνειδητής συγκατάθεσης και διαφάνειας από τα υποκείμενα. Οι τεχνολογίες αναγνώρισης προσώπου στηρίζονται στην αξιοποίηση μιας όσο το δυνατόν μεγαλύτερης βάσης δεδομένων, από την οποία θα αντλούνται τα κρίσιμα χαρακτηριστικά προκειμένου να σχηματιστούν τα κατάλληλα μοτίβα και να επιτευχθεί η επιθυμητή εξαγωγή αποτελεσμάτων. Ωστόσο, η προέλευση των δεδομένων αυτών είναι συχνά αμφίβολη όπως και το κατά πόσο έχει ληφθεί η συγκατάθεση των υποκειμένων για την επεξεργασία, αφού έχει προηγηθεί η ενδελεχής ενημέρωσή τους για το αντικείμενο και τους σκοπούς της παρούσας επεξεργασίας. Με αυτόν τον τρόπο, ο άνθρωπος αντιμετωπίζεται ως μέσο και στερείται του πρωταρχικού δικαιώματός του στην αυτοδιάθεση. Επίσης, η ενδεχόμενη χρήση της αναγνώρισης προσώπου σε συστήματα παρακολούθησης στερεί το υποκείμενο από το συνταγματικό και αναφαίρετο δικαίωμά του στην ιδιωτική ζωή και καταλύει τη σφαίρα του ιδιωτικού βίου.

Στρατιωτικές Χρήσεις: Η Όραση Υπολογιστών μπορεί να ενσωματωθεί σε οπλικά συστήματα κάνοντας τα πιο έξυπνα και πιο αυτόνομα. Τέτοιες εφαρμογές, όπως γίνεται κατανοητό, μπορεί να έχουν καταστροφικές συνέπειες για την ανθρώπινη ζωή και να οδηγήσουν στη διασάλευση των παγκοσμίων ισορροπιών.

Λαμβάνοντας λοιπόν υπόψη μας όλα τα ανωτέρω, καταλήγουμε στο συμπέρασμα ότι η Όραση Υπολογιστών πέρα από τις δυνατότητες που προσφέρει μπορεί να δημιουργήσει προβλήματα στην ανθρώπινη υπόσταση και στις κοινωνικές ισορροπίες όπως τις γνωρίζουμε μέχρι σήμερα. Είναι σημαντικό, η εξέλιξη τέτοιων τεχνολογιών και εφαρμογών να συνοδεύεται από επαρκή έλεγχο και έρευνα των ενδεχομένων αρνητικών επιπτώσεων προκειμένου αυτές αποφευχθούν. Η συλλογή των δεδομένων πρέπει πάντα να γίνεται κατόπιν συνειδητής συναίνεσης, για σκοπούς που δεν έρχονται σε αντίθεση με βασικά ανθρωπιστικά ιδεώδη, με τη

δυνατότητα ελεύθερης πρόσβασης των υποκειμένων στα δεδομένα τους. Η συναίνεση των υποκειμένων πρέπει να διασφαλίζεται επίσης και σε κάθε περίπτωση διαμοιρασμού τους και ύψιστη προτεραιότητα πρέπει να αποτελεί η ασφάλειά τους, η διαφανής διαχείρισή τους καθώς και η υποχρέωση σε λογοδοσία των φορέων επεξεργασίας για κάθε στάδιο της διαδικασίας.

Κεφάλαιο 7 : Επίλογος & Μελλοντικές επεκτάσεις

Στην παρούσα διπλωματική εργασία εξερευνήσαμε και μάθαμε τις διάφορες πτυχές και την ιστορία της Τεχνητής Νοημοσύνης. Μάθαμε για τα διάφορα μοντέλα Μηχανικής Μάθησης (Overfitting, Underfitting, Balanced). Ενημερωθήκαμε για τη δομη και λειτουργία τόσο των Νευρωνικών Δικτύων όσο και των Συνελικτικών Νευρωνικών Δικτύων. Ασχοληθήκαμε με το Tensorflow που αποτελεί ένα από τα πιο δημοφιλή API για την δημιουργία εφαρμογών Τεχνητής Νοημοσύνης. Συλλέξαμε δεδομένα, τα επεξεργαστήκαμε με τρόπο, ώστε να τα τροφοδοτήσουμε και να φτιάξουμε το δικό μας μοντέλο. Δημιουργήσαμε το δικό μας Ανιχνευτή Αντικειμένων που αναγνωρίζει φύλλα τράπουλας και φτιάξαμε μια εφαρμογή που εντοπίζει σε πραγματικό χρόνο τα φύλλα της τράπουλας χρησιμοποιώντας τη web-camera του προσωπικού μας υπολογιστή. Μετρήσαμε την απόδοση και την αποτελεσματικότητα της εφαρμογής μας.

Γενικά ασχοληθήκαμε με μια σύγχρονη πρόκληση, αυτή της Ανίχνευσης Αντικειμένων και πιο συγκεκριμένα της Ανίχνευσης σε Πραγματικό Χρόνο. Η συμβολή της Βαθίας Μηχανικής Μάθησης και της Όρασης Υπολογιστών μπορεί να βοηθήσει σε μεγάλο βαθμό την ανθρωπότητα, καθώς επεκτείνει τις δυνατότητες της ανθρώπινης όρασης. Η Ανίχνευση σε Πραγματικό Χρόνο αποτελεί αναπόσπαστο κομμάτι για τα αυτοοδηγούμενα/αυτόνομα αυτοκίνητα, καθώς τα βοηθάει να αναγνωρίζουν αντικείμενα του περιβάλλοντός τους όπως γραμμές, πινακίδες, ανθρώπους, εμπόδια κ.α. με αποτέλεσμα να κατανοούν το περιβάλλον τους. Χρησιμοποιώντας αυτή τη γνώση και με δημιουργία κανόνων τα αυτοοδηγούμενα αυτοκίνητα οδηγούν με περισσότερη ασφάλεια από τους κανονικούς οδηγούς καθώς ανά πάσα ώρα και στιγμή μπορούν να πάρουν τη «σωστή» απόφαση. Από την άλλη η Ανίχνευση Αντικειμένων σε Πραγματικό Χρόνο μπορεί να χρησιμοποιηθεί ώς μέσο βιντεοπαρακολούθησης με σκοπό να βοηθήσει το άνθρωπο είτε για την ασφάλεια του είτε να μπλοκάρει ανεπιθύμητες ενέργειες. Αξίζει να ειπωθεί ότι η δημιουργία τέτοιων εφαρμογών πρέπει να σέβονται την ανθρώπινη υπόσταση. Πολλές φορές οι δυνατότητες τέτοιων εφαρμογών δημιουργούν ανεργία, καθώς υπερτερούν από την ανθρώπινη εργασία. Είναι σημαντικό να θεσπιστεί και νομοθετικό πλαίσιο, το οποίο να εντάσσει τέτοιες τεχνολογίες όμως έχοντας σαν ύψιστη αξία τον άνθρωπο.

Στόχος της παρούσας διπλωματικής ήταν η δημιουργία του Real-Time Custom Object Detector. Για την δημιουργία υπήρξαν περιορισμοί στο πλήθος δεδομένων που χρησιμοποιήθηκαν και στους διαθέσιμους πόρους πάνω στους οποίους χτίστηκε η εφαρμογή. Παρουσιάζουμε κάποιες από τις επεκτάσεις που μπορούν να εφαρμοστούν :

- Συλλογή μεγαλύτερου πλήθους δεδομένων καθώς και χρήση σαφώς καλύτερου εξοπλισμού για την δημιουργία ενός ακόμη καλύτερου και αξιόπιστου μοντέλου.

- Το προεκπαιδευμένο μοντέλο που χρησιμοποιήθηκε είναι ιδανικό για να τρέξει πέρα από υπολογιστές και σε κινητά τηλέφωνα με την δημιουργία κατάλληλης εφαρμογής Android. Αυτό μπορεί να υλοποιηθεί με την χρήση του Android Studio και την μετατροπή του μοντέλου σε Tensorflow Lite. Αντίστοιχα μπορεί να ενσωματωθεί και σε διαδικτυακές εφαρμογές αν το μοντέλο μας μετατραπεί σε Tensorflow JS.

- Για την αύξηση της απόδοσης μια επέκταση αποτελεί η εκπαίδευση των δεδομένων σε άλλο προεκπαιδευμένο μοντέλο που πετυχαίνει καλύτερη ακρίβεια (mAP).

- Χρήση προεκπαιδευμένων μοντέλων που χρησιμοποιούν εικόνες με ανάλυση μεγαλύτερη από 320x320.

- Για την δημιουργία του συνόλου δεδομένων χρησιμοποιήθηκε μια τράπουλα. Το μοντέλο που δημιουργήθηκε μπορεί και αναγνωρίζει τα συγκεκριμένα φύλλα μόνο σε αυτή την τράπουλα. Για να μπορεί να κάνει καλύτερες γενικεύσεις το μοντέλο μας, δηλαδή να αναγνωρίζει τα ίδια φύλλα και από άλλες τράπουλες που έχουν διαφορετική γραφική αναπαράσταση απαιτείται ο εμπλουτισμός των δεδομένων με εικόνες από διαφορετικές τράπουλες .

Βιβλιογραφία

[1] Introduction to AI, From Builtin.com

[2] Artificial Intelligence (AI), Author IBM

[3] What are the branches of Artificial Intelligence?, From H2kinfosys.com

[4] Narrow AI, From DeepAI.org

[5] What is strong AI?, Author IBM

[6] What Is Artificial Intelligence: Definition & Sub-Fields Of AI, From Softwaretestinghelp.com

[7] Deep Learning, Author IBM

[8] Neural Networks, Author IBM

[9] Computer Vision, Author IBM

[10] Real-Time Object Detection Using TensorFlow, From Mygreatlearning.com

[11] Object Detection Guide, From Fritz AI

[12] Object Detection, From Wiki

[13] What is Python? By Python.org

[14] About OpenCV by OpenCV.org

[15] Matplotlib By Wiki

[16] 12 Types of Neural Network Activation Functions, Author Pragati Baheti

[17] Softmax Activation Function with Python, Author Jason Brownlee

[18] Convolutional Neural Network, Author Md Shahid

[19] Supervised Learning, IBM

[20] Model Fit: Underfitting vs. Overfitting, Amazon

[21] "Μελέτη και υλοποίηση Deep Learning τεχνικών στον τομέα της υπολογιστικής όρασης" ΜΟΝΑΧΟΠΟΥΛΟΣ ΚΩΝΣΤΑΝΤΙΝΟΣ

[22] Padding and Stride, Dive into Deep Learning

[23] Beginners Guide to Convolutional Neural Networks, Author Sabina Pokhrel

[24] Everything You Should Know About Dropouts And BatchNormalization In CNN, Author Rohit Dwivedi

[25] Introduction to Batch Normalization, Author Shipra Saxena

[26] Object Detection Metrics With Worked Example, Author Kiprono Elijah Koech

[27] Generalized Intersection over Union: A Metric and A Loss for Bounding Box Regression, Authors Hamid Rezatofighi Nathan Tsoi JunYoung Gwak Amir Sadeghian Ian Reid Silvio Savarese

[28] mAP (mean Average Precision) for Object Detection, Author Jonathan Hui

[29] Selecting the Right Bounding Box Using Non-Max Suppression , Author Aishwarya Singh

[30] Single Shot Multibox Detector, Authors Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg

[31] SSD object detection: Single Shot MultiBox Detector for real-time processing, Author Jonathan Hui

[32] Ανάλυση ιατρικών εικόνων με χρήση τεχνικών βαθιάς μάθησης, Συγγραφείς Κωστόπουλος Χαράλαμπος , Τσανάκας Παναγιώτης

[33] Ανίχνευση Παραλιών σε Δορυφορικές Εικόνες με Χρήση Συνελικτικών Νευρωνικών Δικτύων, Συγγραφείς Μαθιουλάκη Ελένη, Κόλλιας Στέφανος

[34] The PASCAL Visual Object Classes (VOC) Challenge, Authors Mark Everingham · Luc Van Gool · Christopher K. I. Williams

[35] Detection and Segmentation through ConvNets, Author Ravindra Parmar

[36] Tensorflow Object Detection Walkthrough, Author Nicholas Renotte

[37] An Introduction to Convolutional Neural Networks, Author Keiron O'Shea, Ryan Nash

[38] Ethical Issues in Computer Vision and Strategies for Success, From Innodata.com

[39] The Ethics of AI Image Recognition, Author Bethann Noble

[40] Ethics of Facial Recognition: Key Issues and Solutions, Author Katam Raju Gangarapu

[41] Ethical considerations, From Mypeer.org

[42] The ethical questions that haunt facial-recognition research, Author Richard Van Noorden