ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΛΟΠΟΝΝΗΣΟΥ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ

ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Μελέτη, σχεδίαση και αξιολόγηση τεχνικών επίλυσης προβλημάτων αναγνώρισης αντικειμένων και εντοπισμού σχετικών θέσεων τοποθέτησης αντικειμένων, βασισμένων σε κυρίαρχες τεχνολογίες υπολογιστικής όρασης.

ΣΠΥΡΙΔΩΝ ΜΑΥΡΙΚΗΣ

ΕΠΙΒΛΕΠΩΝ: Αντωνόπουλος Χρήστος
Επίκουρος Καθηγητής

ΠΑΤΡΑ 2021

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή

Πάτρα, Ημερομηνία

ΕΠΙΤΡΟΠΗ ΑΞΙΟΛΟΓΗΣΗΣ

1.

2.

3.

Υπεύθυνη Δήλωση Φοιτητή
Βεβαιώνω ότι είμαι συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης έχω αναφέρει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επίσης βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά ειδικά για τη συγκεκριμένη εργασία.

Η έγκριση της διπλωματικής εργασίας από το Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Πανεπιστημίου Πελοποννήσου δεν υποδηλώνει απαραιτήτως και αποδοχή των απόψεων του συγγραφέα εκ μέρους του Τμήματος.

Περίληψη

Στόχος της παρούσας διπλωματικής εργασίας, είναι η συγκριτική μελέτη, σχεδίαση αλλά και ανάπτυξη τεχνικών για την αποδοτικότερη αναγνώριση ύπαρξης ή μη αντικειμένων, την αναγνώριση της χωρικής τοποθέτησης σε περιπτώσεις πολλαπλής αναγνώρισης, όπως επίσης και η επίλυση προβλημάτων τοποθέτησης αντικειμένων από μια σταθερή θέση ταυτοποίησης προς μια άλλη, με ίδια χαρακτηριστικά του αντικειμένου. Χρησιμοποιούμαι δυο από τις πιο σύγχρονες μεθόδους για την επίτευξη αυτών των στόχων. Αυτή του TensorFlow framework και της βιβλιοθήκης του OpenCV. Στα πλαίσια της αποδοτικότερης μελέτης της εργασίας, θα δημιουργήσουμε ένα custom demo που θα περιλαμβάνει παραμέτρους όπως, τρία είδη γεωμετρικών σχημάτων, συγκεκριμένους χρωματισμούς για τα τρία είδη σχημάτων, τυχαία τοποθέτηση των σχημάτων στο καρτεσιανό επίπεδο αλλά και κάμερα λήψης εικόνας υπό αμετάβλητη γωνία. Έτσι με γνώμονα το κοινό μοτίβο εργασίας, επιτυγχάνεται μια ενδελεχή και αντικειμενική σύγκριση των αποδόσεων των δύο μεθοδολογιών στο κομμάτι της αναγνώρισης αντικειμένων σε πραγματικό χρόνο.

Το συγκεκριμένο θέμα είναι μεγάλου τεχνολογικού ενδιαφέροντος, καθώς η μηχανική όραση έχει καταφέρει να αφομοιωθεί από ολοένα και περισσότερους επιστημονικούς τομείς, καθιστώντας την πλέον κάτι παραπάνω από απαραίτητη, στην προς το κοινό όφελος αλματώδη εξέλιξή τους, αλλά και με πολλές δυνατότητες ιδιαίτερα σημαντικών επεκτάσεων. Κατά την ανάπτυξη της συγκεκριμένης εργασίας, έχει εφαρμοστεί τόσο αλγοριθμική προσέγγιση μέσω της βιβλιοθήκης του OpenCV και της αντικειμενοστρέφειας της Python γλώσσας προγραμματισμού, όσο και η εξαρχής δημιουργία δικού μας μοντέλου αναγνώρισης αντικειμένων μέσω του framework του TensorFlow, αξιοποιώντας έτσι στο μέγιστο τις δυνατότητες των δύο εργαλείων και προσφέροντας αποτελέσματα σε πραγματικό χρόνο με εξαιρετική ακρίβεια, μέσω του custom demo.

Εν τέλει θα γίνει μία εις βάθος ανάλυση της απόδοσης των δύο προσεγγίσεων, παραθέτοντας έτσι, τόσο τα θετικά τους χαρακτηριστικά, όσο και τα μελανά τους σημεία σε πολλαπλά σενάρια πέραν του κυρίως demo, προσπαθώντας με αυτό τον τρόπο να καλύψουμε ένα ευρύ φάσμα παραγόντων όπου μπορούν να επηρεάσουν την απόδοση και το γενικότερο αποτέλεσμα.

Abstract

Aim of this thesis, is the comparative study, design and development of techniques for a more efficient identification of existence or non-existence of objects, the recognition of spatial placement in cases of multiple identification, as well as the solution of placement problems of objects from one fixed location, to an other with the same object characteristics. Two state of the art approaches are used to achieve these goals. That of the TensorFlow framework and the OpenCV library. As part of the most efficient study of the thesis, we will create a custom demo that will include parameters such as three types of geometric shapes, specific colors for the three types of shapes, random placement of the shapes on the Cartesian plane and a camera for video capturing at a constant angle. Thus, guided by the common working pattern, a thorough and objective comparison of the performances of the two methodologies in the field of real-time object recognition is achieved.

This topic is of great technological interest, as mechanical vision has managed to be assimilated by more and more scientific fields, rendering it more than necessary, in the common interest of their leapfrog development, but also with many possibilities of particularly important extensions. During the development of this thesis, both an algorithmic approach has been applied through the OpenCV library and the object-oriented Python programming language, as well as the creation of our own object detection classifier model through the TensorFlow framework, thus making the most of the two tools and offering real-time results with exceptional accuracy, through the custom demo.

Eventually there will be an in-depth analysis of the performance of the two approaches, listing both their positive features and their black spots in multiple scenarios beyond the main demo, in order to cover a wide range of factors where they can affect performance and overall outcome.

Εισαγωγή

Το πρώτο σημαντικό κομμάτι σε ότι αφορά το τεχνολογικό ενδιαφέρον της εργασίας, είναι η γενικότερη έννοια του Computer Vision, οι πολλαπλές τεχνικές οι οποίες απαρτίζουν την έννοια, αλλά και τα πολυποίκιλες εφαρμογές που μπορεί να έχει. Επίσης επιδιώκουμε μια πρώτη επαφή με τον κόσμο του Machine Learning, του Deep Learning, αλλά και των Neural Networks, τον θεμέλιο λίθο και των δύο.

Το Computer Vision μπορεί να οριστεί ως ένα από τα υποπεδία της επιστήμης της τεχνητής νοημοσύνης, που επικεντρώνεται στο να αναπτύξει τεχνικές που βοηθούν τους υπολογιστές να «βλέπουν» και να κατανοούν το περιεχόμενο οπτικών δεδομένων, όπως φωτογραφίες και βίντεο με τον ίδιο τρόπο που το κάνουν οι άνθρωποι. Έχει ήδη ένα τεράστιο εύρος εφαρμογών όπως στην υγεία, την αστρονομία, τον στρατό, την παραγωγική βιομηχανία αλλά και πολλά ακόμη και συνεχίζει καθημερινά να αφομοιώνεται από ολοένα και νέους τομείς της επιστήμης, αλλά και της καθημερινής μας ζωής. Προσεγγίζουμε επίσης κάποιες από τις κυριότερες Computer Vision τεχνικές όπως Object Detection για εντοπισμό αντικειμένων και Object Recognition για αναγνώριση αντικειμένων με συγκεκριμένο μοτίβο. Οι παραπάνω τεχνικές αποτελούν δύο από τις βασικότερες τεχνικές που εφαρμόζονται στο Computer Vision και είναι και μερικές από τις οποίες θα εφαρμόσουμε και στην εργασία μας.

Δεν θα μπορούσαμε φυσικά να μην αναφερθούμε στις έννοιες του Machine Learning, του Deep Learning και των Neural Networks. Τεχνολογίες που χρησιμοποιούμε σχεδόν όλοι καθημερινά χωρίς να το γνωρίζουμε μέσα από το auto complete στις διάφορες μηχανές αναζήτησης, τις πολύ συγκεκριμένες διαφημίσεις που δεχόμαστε στα social media ανάλογα με προηγούμενες αναζητήσεις μας, ακόμα και τα προτεινόμενα βίντεο που αποτελούν αλγοριθμικά συμπεράσματα σύμφωνα με τις δικές μας προτιμήσεις. Το Machine Learning είναι μια τεχνική ανάλυσης δεδομένων που διδάσκει στους υπολογιστές να κάνουν ότι εμείς οι άνθρωποι κάνουμε αυθόρμητα. Το Deep Learning από την άλλη μεριά, ως κομμάτι του Machine Learning, δημιουργήθηκε και αναπτύχθηκε με την φιλοδοξία της μίμησης του ανθρώπινου εγκεφάλου από τους υπολογιστές, πράγμα που τροφοδότησε και την αρχική ανάπτυξη των Neural Networks. Ένα Neural Network ή πιο στοχευμένα ένα Artificial Neural Networks είναι μια σειρά αλγορίθμων που προσπαθούν να αναγνωρίσουν τις υποκείμενες σχέσεις σε ένα σύνολο δεδομένων, μέσω μιας διαδικασίας που μιμείται τον τρόπο λειτουργίας του ανθρώπινου εγκεφάλου. Εμείς ως κομμάτι της εργασία μας θα χρησιμοποιήσουμε τα

Convolutional Neural Networks για την εκπαίδευση και δημιουργία του δικού μας Object Detection Classifier.

Θεωρητικό υπόβαθρο

Στο συγκεκριμένο κεφάλαιο θα γίνει μια εις βάθος ανάλυση του τρόπου λειτουργίας των Neural Networks, όπως επίσης και του TensorFlow framework και του OpenCV library, των δύο προσεγγίσεων ουσιαστικά που θα εφαρμόσουμε για το πέρας της εργασίας μας. Θα εξοικειωθούμε με Neural Network έννοιες όπως, Perceptron and Multilayer Perceptron μορφολογίες νευρώνων, Backpropagation αλγορίθμους, Overfitting και Underfitting περιπτώσεις και πολλές άλλες.

Το TensorFlow είναι μια ολοκληρωμένη πλατφόρμα ανοιχτού κώδικα για μηχανική μάθηση. Δημιουργήθηκε από την ομάδα της Google Brains και επιτρέπει στους ερευνητές να προωθήσουν την τελευταία λέξη της τεχνολογίας στο Machine Learning. Το TensorFlow επιτρέπει την δημιουργία γραφημάτων ροής δεδομένων, δομές που ουσιαστικά περιγράφουν τον τρόπο με τον οποίο τα δεδομένα κινούνται μέσω γραφήματος ή μια σειρά κόμβων επεξεργασίας. Κάθε κόμβος στο γράφημα αντιπροσωπεύει μια μαθηματική λειτουργία και κάθε σύνδεση ή άκρη μεταξύ των κόμβων είναι ένας πολυδιάστατος πίνακας δεδομένων ή tensor. Αποτελεί μία από τις δύο προσεγγίσεις που θα έχουμε στην εργασία μας και μέσα από αυτό θα εκπαιδεύσουμε το δικό μας Object Detection Classifier που θα χρησιμοποιήσουμε για την αναγνώριση των αντικειμένων.

Η δεύτερη προσέγγιση θα γίνει απόλυτα αλγοριθμικά μέσω του OpenCV, που είναι μια open source Computer Vision και Machine Learning software library που παρέχει μια πληθώρα λειτουργειών που θα μας χρησιμεύσουν στο πέρας της εργασίας μας. Θα χρησιμοποιήσουμε functions όπως την cv2.findContours για την εύρεση των περιγραμμάτων των σχημάτων, την cv2.Moments για την εύρεση του κέντρου βάρους των σχημάτων και πολλές άλλες. Επιπλέον θα δημιουργήσουμε μια λειτουργία ανίχνευσης τριών ξεχωριστών χρωμάτων, όπως επίσης και ξεχωριστό κομμάτι στο frame της κάμερας όπου θα πραγματοποιούνται μέσα ξεχωριστές και ανεξάρτητες από το υπόλοιπο frame λειτουργίες. Η προσέγγιση του OpenCV διαφέρει αρκετά από αυτή του TensorFlow, καθώς ως library είμαστε σε θέση να χρησιμοποιήσουμε ήδη

υπάρχουσες functions για την επίτευξη του στόχου μας και δεν τίθεται απαραίτητη η εκπαίδευση ενός classifier.

Σχεδίαση Και Υλοποίηση Λύσης

Σε αυτό το κεφάλαιο θα αναφερθούμε στην Python γλώσσα προγραμματισμού, την γλώσσα δηλαδή που χρησιμοποιήσαμε για το development της εργασίας μας, την διαδικασία εκπαίδευσης του μοντέλου, αλλά και την ανάπτυξη του αλγορίθμου αναγνώρισης μέσω του OpenCV.

Η Python είναι μια ερμηνευτική, αντικειμενοστραφής γλώσσα προγραμματισμού υψηλού επιπέδου με δυναμική σημασιολογία, εύκολο εντοπισμό σφαλμάτων των προγραμμάτων και με γρήγορο κύκλο edit-test-debug, που την καθιστά απλή και πολύ αποτελεσματική στην προσέγγιση. Επίσης μας προσφέρει ένα τεράστιο εύρος διαθέσιμων βιβλιοθηκών για πολλαπλές χρήσεις, ένα μικρό μέρος από το οποίο θα χρησιμοποιήσουμε και εμείς, όπως αυτό της NumPy. Η NumPy είναι βιβλιοθήκη η οποία έχει δημιουργηθεί για να προσθέτει υποστήριξη για μεγάλες πολυδιάστατες συστοιχίες και πίνακες, μαζί με μια μεγάλη συλλογή μαθηματικών συναρτήσεων υψηλού επιπέδου για λειτουργία σε αυτές τις συστοιχίες.

Επίσης θα γίνει μια λεπτομερής ανάλυση της συνολικής διαδικασίας που ακολουθήσαμε για τη δημιουργία του Object Detection Classifier model μέσω του TensorFlow. Θα γίνει αναφορά στη συλλογή και επισήμανση των επιθυμητών αντικειμένων μέσα στις φωτογραφίες, την εκκίνηση της εκπαιδευτικής διαδικασίας του μοντέλου και την εξαγωγή του frozen inference graph. Παράλληλα με την εκπαίδευση του μοντέλου θα παρακολουθούμε την πορεία του TotalLoss γραφήματος για την εξέλιξη και επάρκεια της εκπαίδευσης του μοντέλου μας.

Θα αναφερθούμε επιπλέον στην διαδικασία ανάπτυξης του συνολικού αλγορίθμου αναγνώρισης με την βοήθεια των OpenCV functions. Θα αναλύσουμε τις μεθοδολογίες που χρησιμοποιήσαμε, την χρησιμότητά τους, αλλά και πως μπορούν να εναρμονιστούν με την φύση της εργασίας μας.Τέλος θα γίνει λόγος στις δυσκολίες ανάπτυξης της εργασίας, όσων αφορά τις δύο παραπάνω προσεγγίσεις.

Ανάλυση Απόδοσης

Τέλος θα γίνει μια ανάλυση της απόδοσης των δυο μεθοδολογιών σε παράγοντες όπως:

- Φωτισμός και υπόβαθρο, για την ανάλυση της διαστρέβλωσης των αποτελεσμάτων σε περιπτώσεις διαφορετικών υποβάθρων και φωτισμού

- Γωνία και απόσταση κάμερας, για την ανάλυση της διαστρέβλωσης των αποτελεσμάτων σε περιπτώσεις λήψης εικόνας υπό διαφορετική γωνία ή υπό διαφορετική απόσταση.

- Σειρά αναγνώρισης αντικειμένων, για την ανάλυση της σειρά και του τρόπου που πραγματοποιούν την αναγνώριση των αντικειμένων οι δύο προσεγγίσεις

- Επεξεργαστική ταχύτητα σε πραγματικό χρόνο, για τον προσδιορισμό των fps που δεχόμαστε ως feedback κατά την running διαδικασία των προσεγγίσεων

Ευχαριστίες

Η συγκεκριμένη πτυχιακή εργασία διεξήχθει στα πλαίσια του προγράμματος σπουδών του πρώην τμήματος Μηχανικών Πληροφορικής Τ.Ε.Ι. Δυτικής Ελλάδας, νυν τμήμα Ηλεκτρολόγων Μηχανικών Και Μηχανικών Υπολογιστών του Πανεπιστημίου Πελοποννήσου

Θα ήθελα καταρχάς να ευχαριστήσω τον επιβλέπων καθηγητή μου κύριο Χρήστο Αντωνόπουλο, για την πολύτιμη βοήθεια και καθοδήγηση που μου προσέφερε καθ'όλη την διάρκεια έρευνας και υλοποίησης της πτυχιακής εργασίας.

Θα ήθελα επίσης να ευχαριστήσω τον διδακτορικό φοιτητή του τμήματος Ηλεκτρολόγων Μηχανικών Και Μηχανικών Υπολογιστών του Πανεπιστημίου Πελοποννήσου Αλέξανδρο Σπουρνιά, για την εθελοντική βοήθεια που απλόχερα μου προσέφερε.

Τέλος θα ήθελα να ευχαριστήσω τους γονείς μου, για την ψυχολογική στήριξη που μου προσέφεραν σε όλη την διάρκεια την φοίτησής μου.

Περιεχόμενα

Κεφάλαιο 1-Εισαγωγή

1.1 Υπολογιστική Όραση (Computer Vision)

Εδώ και πολλές δεκαετίες, οι άνθρωποι ονειρεύονταν να δημιουργήσουν μηχανές με τα χαρακτηριστικά της ανθρώπινης νοημοσύνης, όπως να μπορούν να σκέφτονται και να ενεργούν σαν τους ανθρώπους. Μία από τις πιο συναρπαστικές ιδέες ήταν να δοθεί στους υπολογιστές η δυνατότητα να "βλέπουν" και να ερμηνεύουν τον κόσμο γύρω τους. Η φαντασία του χθες έχει γίνει η πραγματικότητα του σήμερα. Χάρη στις εξελίξεις στην τεχνητή νοημοσύνη και την υπολογιστική ισχύ, η τεχνολογία της μηχανικής οράσεως (Computer Vision) έχει κάνει ένα τεράστιο άλμα προς την ένταξή της στην καθημερινή μας ζωή. Σε αυτή την πτυχιακή εργασία, θα προσπαθήσουμε να αναλύσουμε διεξοδικά την έννοια του Computer Vision και θα μοιραστούμε μερικά εξαιρετικά παραδείγματα όπου αυτή η τεχνολογία μπορεί να εφαρμοστεί στη ζωή μας.

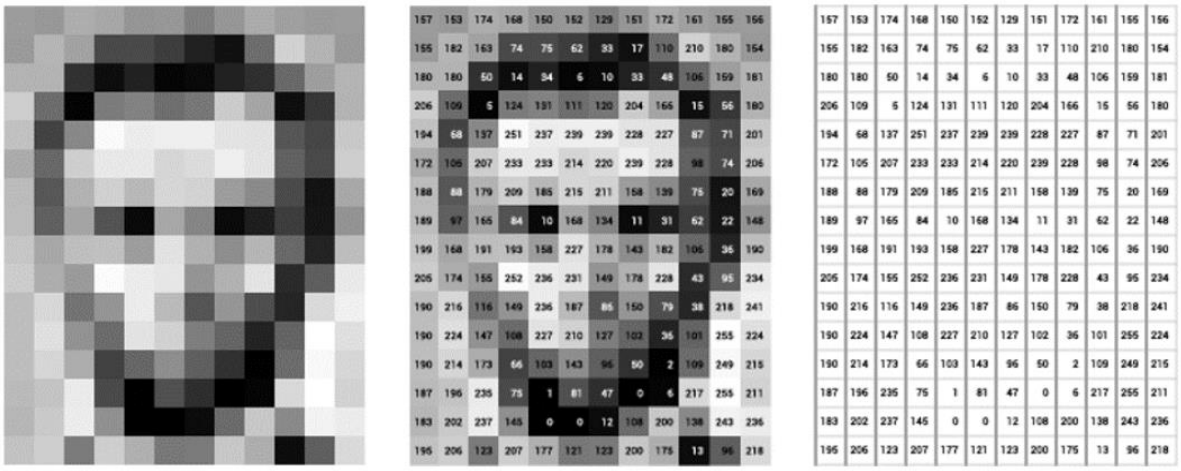
Τι είναι λοιπόν η έννοια του Computer Vision; Το Computer Vision μπορεί να οριστεί ως ένα από τα υποπεδία της επιστήμης της τεχνητής νοημοσύνης, που επικεντρώνεται στο να αναπτύξει τεχνικές που βοηθούν τους υπολογιστές να «βλέπουν» και να κατανοούν το περιεχόμενο οπτικών δεδομένων, όπως φωτογραφίες και βίντεο με τον ίδιο τρόπο που το κάνουν οι άνθρωποι. Είναι ένα διεπιστημονικό πεδίο λοιπόν, το οποίο μπορεί να περιλαμβάνει τη χρήση εξειδικευμένων αλγορίθμων μάθησης (learning algorithms) για την επεξεργασία της εικόνας σε επίπεδο pixel, την ανάκτηση της οπτικής πληροφορίας και την αντίδραση του υπολογιστή σε αυτή. Ως διεπιστημονικό πεδίο μελέτης λοιπόν, το Computer Vision μοιράζεται αλγόριθμους και τεχνικές από τον χώρο της μηχανικής μάθησης (Machine Learning), όπως επίσης και μέρη του Machine Learning έχουν αναπτυχθεί αποκλειστικά για το Computer Vision. Η έννοια του Machine Learning θα αναπτυχθεί περαιτέρω σε επόμενη ενότητα της εισαγωγής, καθώς και η βαθύτερη σχέση αυτού με το Computer Vision.

Η επιστήμη του Computer Vision δεν αποτελεί νέο στην επιστήμη των υπολογιστών καθώς ξεκίνησε να παίρνει μορφή ως επιστημονικό πεδίο στη δεκαετία του 1960, αλλά παραμένει σε μεγάλο βαθμό ακόμη και σήμερα ένα πρόβλημα, που βασίζεται τόσο στην περιορισμένη κατανόηση της βιολογικής όρασης, όσο και στην πολυπλοκότητας της αντίληψης της όρασης σε έναν δυναμικό και σχεδόν απεριόριστο φυσικό κόσμο. Τα πρώιμα

πειράματα στο Computer Vision πραγματοποιήθηκαν πίσω στο 1950 και ξεκίνησε η εμπορική χρήση της τεχνολογίας το 1970, με σκοπό την ερμηνεία δακτυλογραφημένου και πληκτρολογημένου κειμένου. Εκείνη την εποχή, οι διαδικασίες ανάλυσης του Computer Vision ήταν σχετικά απλές αλλά απαιτούσαν εκτεταμένη δουλειά από τον ανθρώπινο παράγοντα που έπρεπε να παρέχει δείγματα δεδομένων για ανάλυση, με μη αυτοματοποιημένο τρόπο. Όπως είναι αντιληπτό, ήταν δύσκολο να παρέχεται ο απαραίτητος τεράστιος όγκος δεδομένων χειροκίνητα, γεγονός που καθυστερούσε σημαντικά τα εκάστοτε πειράματα. Επιπλέον, η υπολογιστική ισχύς δεν ήταν αρκετά καλή, επομένως το περιθώριο σφάλματος για αυτήν την ανάλυση ήταν αρκετά υψηλό. Σήμερα όμως όπου δεν αντιμετωπίζουμε προβλήματα έλλειψης ισχύος και σε συνδυασμό με τεχνολογίες όπως Cloud Computing και robust αλγορίθμους, είμαστε σε θέση να επιλύσουμε ακόμη και τα πιο περίπλοκα προβλήματα με εξαιρετική ακρίβεια.

Για να έχουμε μια πιο στοχευμένη προσέγγιση της τεχνολογίας του Computer Vision, πρέπει να εμβαθύνουμε περισσότερο στην λογική στην οποία βασίστηκε η ανάπτυξη της, στους αλγορίθμους που την απαρτίζουν, αλλά να δούμε και ένα γενικό πλάνο του πως λειτουργεί. Η τεχνολογία του Computer Vision λοιπόν τείνει να μιμείται τον τρόπο λειτουργίας του ανθρώπινου εγκεφάλου, όπου μία από τις πιο δημοφιλείς υποθέσεις, δηλώνει πως ο εγκέφαλός μας βασίζεται σε μοτίβα, για την αποκωδικοποίηση μεμονωμένων αντικειμένων. Οπότε η δημιουργία Computer Vision αλγορίθμων βασίζεται και αυτή στη χρήση μοτίβων αναγνώρισης. Έτσι εκπαιδεύουμε τους υπολογιστές πάνω σε μια τεράστια ποσότητα όμοιων οπτικών δεδομένων, οι υπολογιστές επεξεργάζονται τις εικόνες, επισημαίνουν αντικείμενα πάνω τους και βρίσκουν μοτίβα αναγνώρισης για αυτά τα αντικείμενα. Αυτή η διαδικασία, είναι η διαδικασία δημιουργίας μοντέλου προβλέψεων (Predictive Model) και είναι το αποτέλεσμα της εκτεταμένης και πολυεπίπεδης ανάλυσης του κάθε pixel όλων των οπτικών δεδομένων, μέσω νευρωνικών δικτύων (Neural Networks) και αποτελεί μια από τις πολλές Computer Vision τεχνικές. Ως ένας διεπιστημονικός κλάδος μελέτης, μπορεί να φαίνεται ακατάστατος, με τεχνικές δανειζόμενες και επαναχρησιμοποιούμενες από ένα μεγάλο εύρος διαφορετικών τομέων μηχανικής και πληροφορικής. Υπό διαφορετική σκοπιά, ο Golan Levin, στο άρθρο του Image Processing and Computer Vision, παρέχει τεχνικές λεπτομέρειες σχετικά με τη διαδικασία που ακολουθούν οι υπολογιστές στην ερμηνεία εικόνων. Εν ολίγοις, οι υπολογιστές ερμηνεύουν τις εικόνες ως μια σειρά από pixel, το καθένα με το δικό του σύνολο τιμών χρώματος. Για παράδειγμα, παρακάτω στην εικόνα (Εικόνα 1.1) απεικονίζεται ο Αβραάμ Λίνκολν. Η φωτεινότητα κάθε pixel σε αυτήν την εικόνα αντιπροσωπεύεται από έναν

8-bit αριθμό, που κυμαίνεται από 0 (μαύρο) έως 255 (λευκό). Αυτοί οι αριθμοί είναι αυτό που βλέπει το λογισμικό όταν εισάγετε μια εικόνα. Αυτά τα δεδομένα παρέχονται ως είσοδος στον Computer Vision αλγόριθμο που θα είναι υπεύθυνος για περαιτέρω ανάλυση και λήψη αποφάσεων.



Εικόνα 1.1 - Οι τιμές των χρωμάτων μεμονωμένων pixels, μετατρέπονται σε ένα πίνακα αριθμών που χρησιμοποιείται ως είσοδος για τον Computer Vision αλγόριθμο

Οι Computer Vision εφαρμογές λοιπόν, όντας πολύπλευρες και πολυποίκιλες καταφέρνουν να αφομοιώνονται από ολοένα και περισσότερα μέρη της καθημερινής μας ζωής, των επιστημών και της βιομηχανίας λόγω των πιθανών οφελών που μπορούμε να αποκομίσουμε από την αντικατάσταση ενός ανθρώπινου παράγοντα με έναν υπολογιστή σε συγκεκριμένους τομείς εξαλείφοντας τα περιθώρια ανθρώπινου λάθους, αλλά και λόγω της διευκόλυνσης πολλές φορές της εργασίας και αύξησης της παραγωγικότητας σε περιπτώσεις που το Computer Vision μπορεί να χρησιμοποιηθεί ως μέσο βοήθειας, λειτουργώντας συνεργατικά με τον άνθρωπο. Μερικές περιπτώσεις εφαρμογής του Computer Vision είναι εξής:

- **Αυτόνομη Οδήγηση**

  Το Computer Vision επιτρέπει στα αυτοκίνητα να κατανοήσουν το περιβάλλον τους. Ένα έξυπνο όχημα έχει μερικές κάμερες που καταγράφουν βίντεο από διαφορετικές οπτικές γωνίες και το στέλνουν ως σήμα εισόδου στο λογισμικό όρασης του υπολογιστή. Το σύστημα επεξεργάζεται το βίντεο σε πραγματικό χρόνο και εντοπίζει αντικείμενα όπως σήμανση δρόμου, αντικείμενα κοντά στο αυτοκίνητο (όπως πεζοί ή άλλα αυτοκίνητα), φανάρια κ.λπ. Ένα από τα πιο αξιοσημείωτα παραδείγματα εφαρμογών αυτής της τεχνολογίας είναι ο αυτόματος πιλότος σε αυτοκίνητα Tesla .

- **Παραγωγική Βιομηχανία**

  Το Computer Vision σε συνδυασμό με αισθητήρες μπορεί να κάνει θαύματα στην απόδοση και παραγωγή μια βιομηχανίας. Σήμερα, η τεχνολογία χρησιμοποιείται για τον έλεγχο των εγκαταστάσεων και του εξοπλισμού, της ταχύτερης και αποδοτικότερης παραγωγής, την ασφάλεια των εργαζομένων, στη μείωση των ελαττωμάτων στην παραγωγή και πολλά άλλα. Πολλές εταιρείες χρησιμοποιούν την μέθοδο της προληπτικής συντήρησης για να διατηρήσουν τον παραγωγικό τους εξοπλισμό σε καλή κατάσταση. Για παράδειγμα, το λογισμικό ZDT που κατασκευάζεται από τη FANUC είναι ένα λογισμικό προληπτικής συντήρησης που έχει σχεδιαστεί για τη συλλογή εικόνων από κάμερα συνδεδεμένη με ρομπότ. Στη συνέχεια, αυτά τα δεδομένα υποβάλλονται σε επεξεργασία για να παρέχουν διάγνωση προβλημάτων και να εντοπίζουν πιθανά προβλήματα.

- **Λιανεμπόριο**

  Η εταιρεία Walmart χρησιμοποιεί το Computer Vision για να παρακολουθεί τις περιπτώσεις κλοπής και να αποτρέπει με αυτό τον τρόπο τη συνολική ζημία σε 1.000 καταστήματα σε ολόκληρη τη Αμερική. Έχουν αναπτύξει ένα Missed Scan Detection πρόγραμμα που ανιχνεύει τις περιπτώσεις εσφαλμένης ή μη σάρωσης των προϊόντων, που χρησιμοποιεί κάμερες για τον εντοπισμό αυτών

των περιπτώσεων. Μόλις εντοπιστεί ένα σφάλμα, η συγκεκριμένη τεχνολογία ενημερώνει τους διαχειριστές ταμείων ώστε να μπορούν να το αντιμετωπίσουν. Αυτή η πρωτοβουλία συμβάλλει δραστικά στη μείωση της ζημίας που περιλαμβάνει κλοπή, σφάλματα σάρωσης και λοιπές απάτες. Προς το παρόν, το πρόγραμμα έχει αποδειχθεί αποτελεσματικό στην ψηφιοποίηση της παρακολούθησης των ταμείων και στην πρόληψη των απωλειών.

- **Υγεία**

  Επί του παρόντος, υπάρχουν αρκετοί τομείς στην υγειονομική περίθαλψη όπου χρησιμοποιείται το Computer Vision και ωφελεί τους επαγγελματίες υγείας να κάνουν αποτελεσματικότερη διάγνωση και θεραπεία στους ασθενείς. Οι εφαρμογές αυτές περιλαμβάνου τομείς όπως, ανάλυση ιατρικής απεικόνισης, προγνωστική ανάλυση, παρακολούθηση της υγείας, μικροχειρουργική, πυρηνική ιατρική, μείωση χρόνου διάγνωσης κ.τ.λ.

- **Γεωργία**

  Το Computer Vision μπορεί να συμβάλει στη χαρτογράφηση, την ανάλυση του εδάφους, τη μέτρηση των ζώων, την αξιολόγηση της απόδοσης των καλλιεργειών και της ωριμότητάς της, αλλά και πολλά άλλα. Το RSIP όραμα ανέπτυξε πολλές γεωργικές λύσεις χρησιμοποιώντας το Deep Learning, συστήματα αισθητήρων και δορυφορικές εικόνες, ώστε οι αγρότες να μπορούν να εκτιμήσουν την εποχική απόδοση πριν από τη συγκομιδή χρησιμοποιώντας τα smartphones ή τα tablets τους.

- **Αστρονομία**

  Ομοίως και στην αστρονομία, η χρήση του Computer Vision είναι εκτεταμένη αν όχι καθολική. Τρανό παράδειγμα αποτελεί το πρόγραμμα τεχνητής νοημοσύνης Morpheus, που αναπτύχθηκε από ερευνητές στο UC Santa Cruz που μπορεί να αναλύσει δεδομένα αστρονομικής εικόνας και να ταξινομήσει γαλαξίες και αστέρες με χειρουργική ακρίβεια. Το πρόγραμμα αυτό αξιοποιεί το Deep Learning και εφαρμόζει Computer Vision αλγόριθμους για την

ταξινόμηση αντικειμένων με βάση τα πρωτογενή δεδομένα που ρέουν από τηλεσκόπια.

- **Στρατός**

  Οι στρατιωτικές εφαρμογές είναι πιθανώς ένας από τους μεγαλύτερους τομείς για το Computer Vision. Το προφανές παραδείγματα είναι η ανίχνευση εχθρικών στρατιωτών ή οχημάτων και καθοδήγηση πυραύλων σε εχθρικούς στόχους. Σύγχρονες στρατιωτικές έννοιες, όπως η "Επίγνωση στο πεδίο της μάχης", υπαινίσσονται πως η χρήση αισθητήρων, συμπεριλαμβανομένων των αισθητήρων εικόνας, παρέχουν ένα πλούσιο σύνολο πληροφοριών για ένα σκηνικό πολέμου, που μπορεί να χρησιμοποιηθεί για την υποστήριξη στρατηγικών αποφάσεων.

- **Επιτήρηση**

  Η τεχνολογία του Computer Vision εδώ, επιτρέπει την ασφάλεια δημόσιων χώρων όπως χώρους στάθμευσης, μετρό, σιδηροδρομικούς σταθμούς και σταθμούς λεωφορείων, δρόμους και αυτοκινητόδρομους κ.λπ. Η εφαρμογή του Computer Vision για λόγους ασφαλείας είναι διαφορετική. Εμπεριέχει έννοιες όπως, αναγνώριση προσώπου, ανίχνευση πλήθους, ανίχνευση ανθρώπινης μη φυσιολογικής συμπεριφοράς, παράνομη ανίχνευση στάθμευσης, ανίχνευση υπερβολικής ταχύτητας και πολλά άλλα. Η τεχνολογία συμβάλλει στην ενίσχυση της ασφάλειας και στην πρόληψη διαφόρων ειδών ατυχημάτων.

1.1.1 Βασικές Λειτουργίες Υπολογιστικής Όρασης (Computer Vision Tasks)

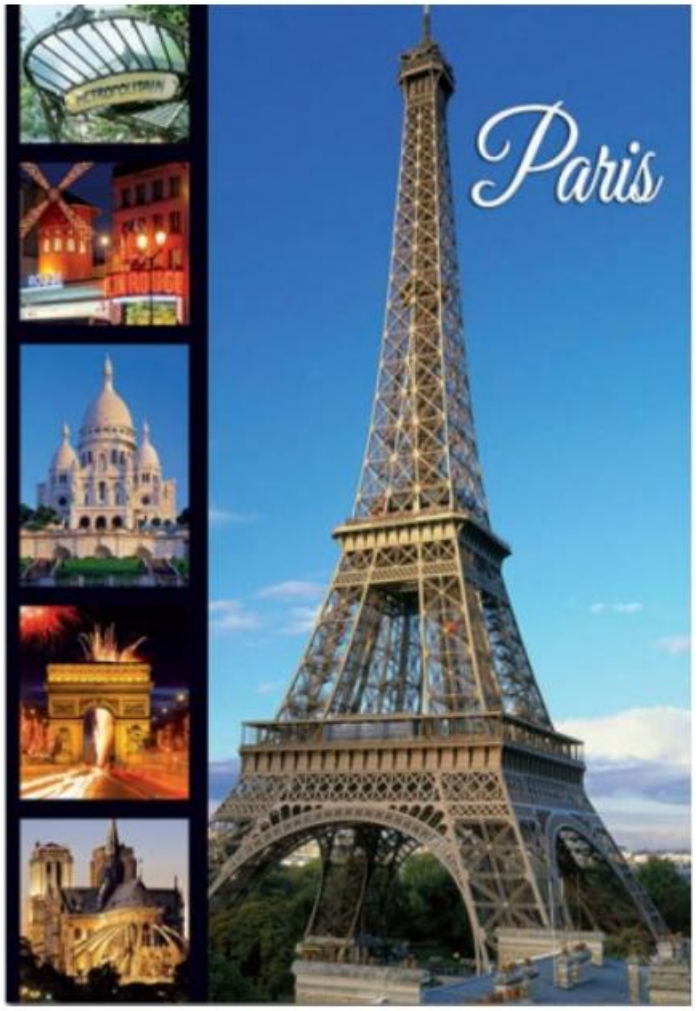
Όπως είναι αντιληπτό, για να είναι αποτελεσματική η χρήση του Computer Vision στις παραπάνω εφαρμογές, πρέπει να βασίζεται σε ένα ευρύ φάσμα διαφορετικών εργασιών, τα οποία συνδυαστικά θα επιτελέσουν στην επίτευξη εξειδικευμένων εφαρμογών. Οι πιο συχνές εργασίες στο Computer Vision είναι η αναγνώριση εικόνας και βίντεο, οι οποίες ουσιαστικά επιτελούν στον προσδιορισμό των διαφορετικών αντικειμένων που περιέχει μια εικόνα. Μερικές τυπικές εργασίες του Computer Vision είναι οι εξής:

- **Ταξινόμηση Εικόνας (Image Classification)**

  Πιθανώς ένα από τα πιο γνωστά έργα του Computer Vision είναι η ταξινόμηση εικόνας. Το Image Classification επιτρέπει την ταξινόμηση μιας δεδομένης εικόνας σε ένα σύνολο προκαθορισμένων κατηγοριών. Ας πάρουμε ένα απλό δυαδικό παράδειγμα, στο οποίο θέλουμε να κατηγοριοποιήσουμε εικόνες ανάλογα με το αν περιέχουν τουριστικό αξιοθέατο ή όχι. Ας υποθέσουμε ότι έχει δημιουργηθεί ένας ταξινομητής (classifier) για αυτόν τον σκοπό και ότι παρέχεται η παρακάτω εικόνα (Εικόνα 1.2).

Εικόνα 1.2

Ο classifier θα αποκριθεί ότι η εικόνα ανήκει στην ομάδα εικόνων που περιέχουν τουριστικά αξιοθέατα. Αυτό δεν σημαίνει ότι έχει αναγνωρίσει απαραίτητα τον Πύργο του Άιφελ, αλλά ότι έχει δει προηγουμένως φωτογραφίες του πύργου και ότι γνωρίζει πως αυτές οι εικόνες περιέχουν τουριστικό αξιοθέατο. Μια πιο φιλόδοξη έκδοση του classifier θα μπορούσε να έχει περισσότερες από δύο κατηγορίες. Για παράδειγμα, θα μπορούσε να υπάρχει μια κατηγορία για κάθε συγκεκριμένο τύπο τουριστικού αξιοθέατου που θέλουμε να αναγνωρίσουμε, όπως Πύργος του Άιφελ, Αψίδα του Θριάμβου, Sacré-Coeur κ.λπ. Σε ένα τέτοιο σενάριο, οι απαντήσεις ανά είσοδο εικόνας θα μπορούσαν να είναι πολλαπλές, όπως στο η περίπτωση της παρακάτω κάρτας (Εικόνα 1.3)

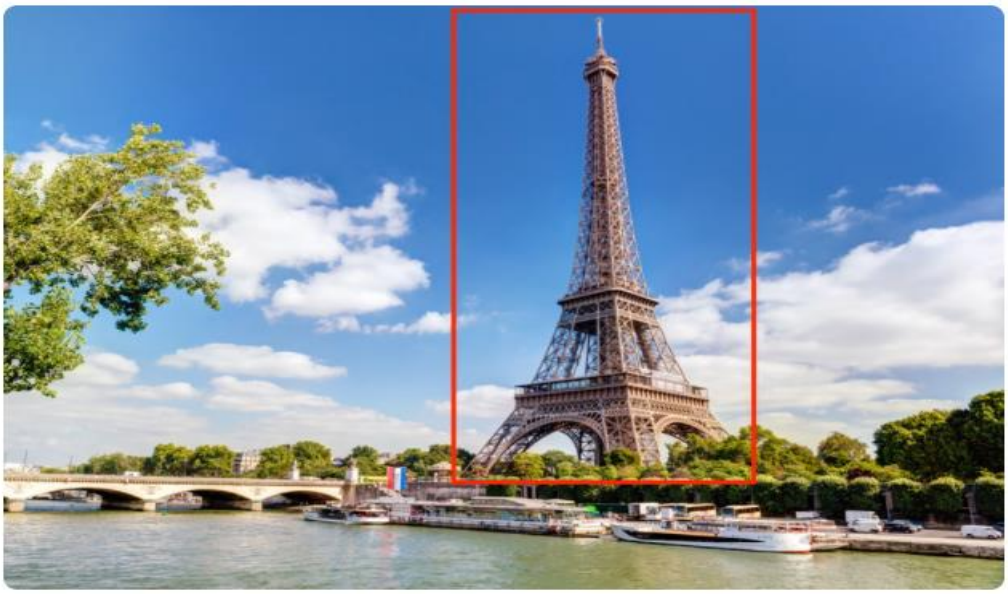


Postcard of tourist attractions in Paris

Εικόνα 1.3

- **Εντοπισμός (Localization)**

  Ας υποθέσουμε τώρα ότι δεν θέλουμε μόνο να γνωρίζουμε ποια τουριστικά αξιοθέατα εμφανίζονται σε μια εικόνα, αλλά επίσης ενδιαφερόμαστε να μάθουμε ακριβώς που βρίσκονται. Ο στόχος του Localization είναι η εύρεση της θέσης ενός αντικειμένου σε μια εικόνα. Για παράδειγμα, στην παρακάτω εικόνα (Εικόνα 1.4), ο Πύργος του Άιφελ έχει εντοπιστεί. Ο τυπικός τρόπος για να εκτελεστεί το Localization είναι να ορίσουμε ένα πλαίσιο οριοθέτησης (bounding box) που περικλείει το αντικείμενο στην εικόνα. Το Localization είναι μια ιδιαίτερα χρήσιμη εργασία. Μπορεί να επιτρέψει την αυτόματη περικοπή αντικειμένων σε ένα σύνολο εικόνων για παράδειγμα. Εάν συνδυαστεί με το Classification, θα μας επιτρέψει να δημιουργήσουμε γρήγορα ένα σύνολο δεδομένων (dataset) από περικομμένες εικόνες από διάσημα τουριστικά αξιοθέατα.

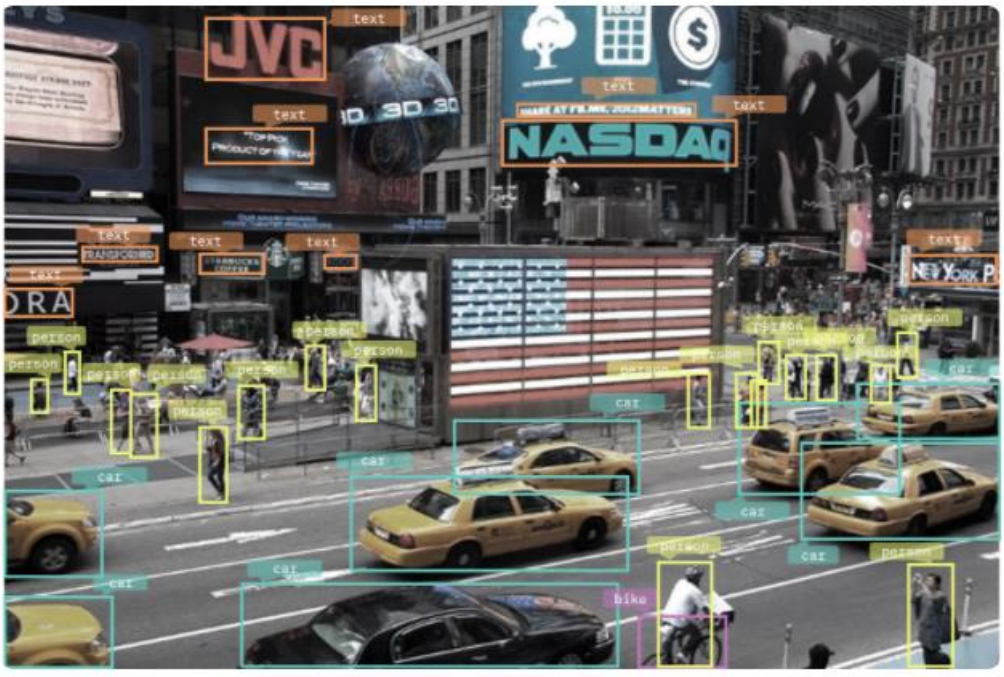


The Eiffel Tower enclosed by a bounding box

Εικόνα 1.4

- **Εντοπισμός Αντικειμένου (Object Detection)**

Εάν φανταστούμε μια ενέργεια που περιλαμβάνει ταυτόχρονα τις έννοιες του Localization και Classification και είναι επαναλαμβανόμενη για όλα τα αντικείμενα ενδιαφέροντος σε μια εικόνα, καταλήγουμε στο Object Detection. Στο Object Detection, ο αριθμός των αντικειμένων που μπορεί να περιέχει μια εικόνα είναι άγνωστος και σε πολλές περιπτώσεις μηδενικός. Ο σκοπός του Object Detection είναι επομένως, η εύρεση και μετά ταξινόμηση ενός μεταβλητού αριθμού αντικειμένων σε μια εικόνα. Στην παραπάνω ιδιαίτερα πυκνή εικόνα (Εικόνα 1.5), βλέπουμε πως ένα Computer Vision σύστημα αναγνωρίζει έναν μεγάλο αριθμό διαφορετικών αντικειμένων, όπως αυτοκίνητα, ανθρώπους, ποδήλατα, ακόμη και πινακίδες που περιέχουν κείμενο. Ορισμένα αντικείμενα είναι μόνο μερικώς ορατά, είτε επειδή βρίσκονται εν μέρει έξω από το πλαίσιο της εικόνας είτε επειδή αλληλεπικαλύπτονται. Επίσης, το μέγεθος παρόμοιων αντικειμένων ποικίλλει σημαντικά. Μια απλή εφαρμογή του Object Detection είναι η αρίθμηση. Οι εφαρμογές της αρίθμησης στην πραγματική ζωή είναι αρκετά διαφορετικές, από τη μέτρηση διαφορετικών τύπων φρούτων που συλλέγονται έως την καταμέτρηση ανθρώπων σε εκδηλώσεις όπως δημόσιες διαδηλώσεις ή ποδοσφαιρικούς αγώνες.



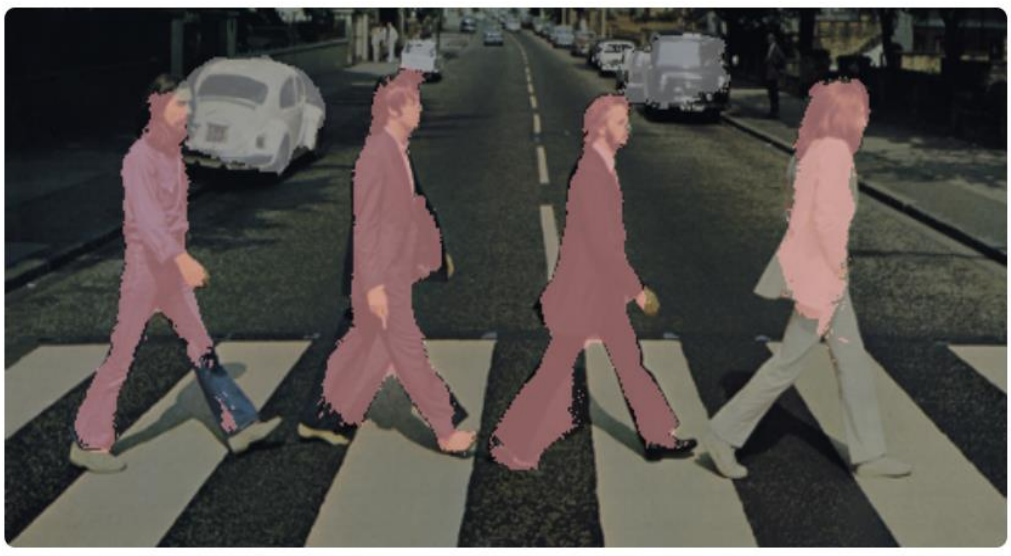
Object detection results
Εικόνα 1.5

- **Ταυτοποίηση Αντικειμένου (Object Identification)**

  Το Object Identification είναι ελαφρώς διαφορετική τεχνική από το Object Detection, αν και παρόμοιες τεχνικές χρησιμοποιούνται συχνά για την επίτευξη και των δύο. Σε αυτήν την περίπτωση, δεδομένου ενός συγκεκριμένου αντικειμένου, ο στόχος είναι η εύρεση παρουσιών του εν λόγω αντικειμένου σε εικόνες. Δεν πρόκειται για την ταξινόμηση μιας εικόνας όπως είδαμε προηγουμένως, αλλά για τον προσδιορισμό εάν το αντικείμενο εμφανίζεται σε μια εικόνα ή όχι και αν όντως εμφανίζεται, να προσδιορίζονται οι τοποθεσίες όπου εμφανίζεται. Ένα παράδειγμα μπορεί να είναι η αναζήτηση εικόνων που περιέχουν το λογότυπο μιας συγκεκριμένης εταιρείας. Ένα άλλο παράδειγμα είναι η παρακολούθηση εικόνων σε πραγματικό χρόνο από κάμερες ασφαλείας για τον προσδιορισμό του προσώπου ενός συγκεκριμένου ατόμου.

- **Τμηματοποίηση Αντικειμένου (Object Segmentation)**

  Το Object Segmentation μπορεί να θεωρηθεί ως ένα επόμενο βήμα μετά το Object Detection. Σε αυτήν την περίπτωση, δεν πρόκειται μόνο για την εύρεση αντικειμένων σε μια εικόνα, αλλά και για τη δημιουργία μίας όσο το δυνατόν ακριβέστερη μάσκας για κάθε ανιχνευόμενο αντικείμενο.



Instance segmentation results

Εικόνα 1.6

Μπορούμε να δούμε στην παραπάνω εικόνα (Εικόνα 6), πως ο Object Segmentation αλγόριθμος βρίσκει μάσκες για τους τέσσερις Beatles και ορισμένα αυτοκίνητα. Τέτοια αποτελέσματα θα ήταν πολύ δαπανηρά αν οι εργασίες για να επιτευχθεί είχαν εκτελεστεί χειροκίνητα, αλλά η τεχνολογία διευκολύνει την επίτευξή τους.

- **Ιχνηλάτηση Αντικειμένου (Object Tracking)**

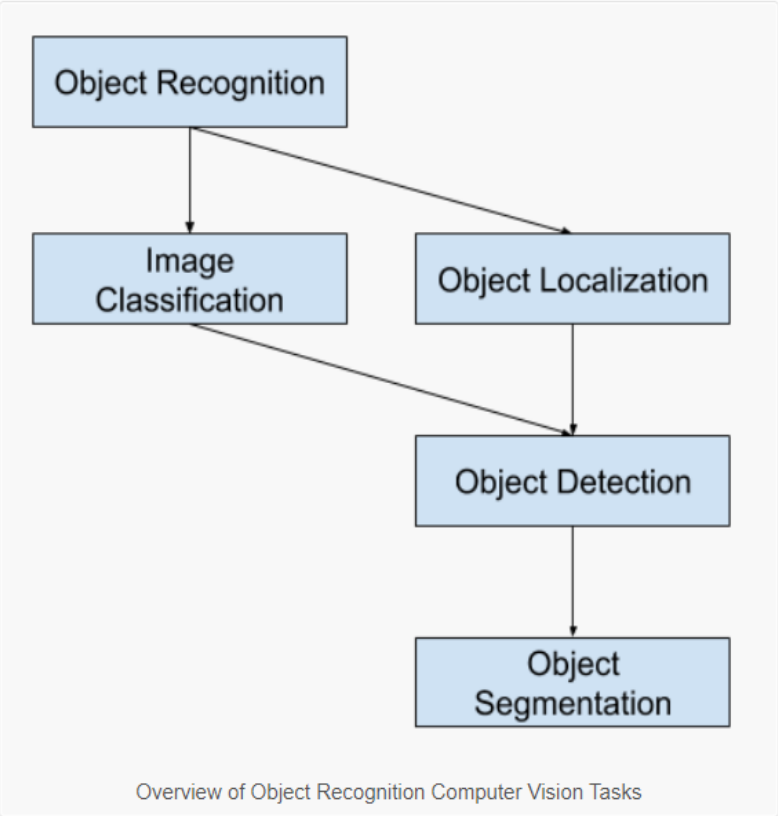
  Ο σκοπός του Object Tracking είναι η παρακολούθηση ενός αντικειμένου που κινείται με την πάροδο του χρόνου, χρησιμοποιώντας διαδοχικά καρέ από βίντεο ως είσοδο. Αυτή η λειτουργικότητα είναι απαραίτητη για τα αυτόνομα οχήματα, ώστε να τους επιτρέπει χωρικούς συλλογισμούς και υψηλού επιπέδου σχεδιασμό διαδρομών. Ομοίως, είναι χρήσιμο σε διάφορα ανθρώπινα συστήματα παρακολούθησης, όπως αυτά που προσπαθούν να κατανοήσουν τη συμπεριφορά των πελατών σε καταστήματα ή εκείνα που παρακολουθούν συνεχώς ποδοσφαιριστές ή καλαθοσφαιριστές κατά τη διάρκεια ενός παιχνιδιού. Μια Object Tracking μέθοδος, δεν χρειάζεται απαραίτητα να είναι ικανή να ανιχνεύει αντικείμενα. Μπορεί απλά να βασίζεται σε κριτήρια κίνησης, χωρίς να γνωρίζει το αντικείμενο ότι παρακολουθείται.

Όσο καλοί και αποτελεσματικοί όμως και αν είναι οι Computer Vision αλγόριθμοι, εξακολουθούν να υποφέρουν από κάποιες μεγάλες προκλήσεις. Το πρώτη μεγάλη πρόκληση είναι η έλλειψη καλής επισήμανσης αντικειμένων στις εικόνες που χρησιμοποιούνται για την εκπαίδευση των αλγορίθμων ώστε να έχουν την βέλτιστη απόδοση. Τα νευρωνικά δίκτυα που χρησιμοποιούνται για Computer Vision εφαρμογές είναι πιο εύκολο να εκπαιδευτούν από ποτέ, αλλά αυτό απαιτεί πολλά υψηλής ποιότητας δεδομένα. Αυτό σημαίνει ότι οι αλγόριθμοι χρειάζονται πολλά δεδομένα που σχετίζονται ειδικά με το έργο που καλούνται να φέρουν εις πέρας, για να έχουν όσο το δυνατόν καλύτερα αποτελέσματα. Παρά το γεγονός ότι οι εικόνες είναι διαθέσιμες στο διαδίκτυο σε τεράστιες ποσότητες περισσότερο από ποτέ, η απόδοση λύσης σε πολλά προβλήματα στον πραγματικό κόσμο απαιτεί υψηλής ποιότητας επισημασμένων δεδομένων εκπαίδευσης (high-quality labeled training data). Μία δεύτερη πολύ μεγάλη πρόκληση, είναι η έλλειψη ακρίβειας όταν εφαρμόζονται οι αλγόριθμοι σε εικόνες του περιβάλλοντος διαφορετικές από το σύνολο δεδομένων πάνω στο οποίο έγινε η

εκπαίδευσης του μοντέλου αναγνώρισης. Ένα καλό παράδειγμα είναι η έννοια ενός βιβλίου. Οι άνθρωποι γνωρίζουμε την έννοια ενός βιβλίου και μπορούμε να διακρίνουμε τις διαφορές μεταξύ ενός βιβλίου, ενός περιοδικού ή ενός κόμικ, ενώ κατανοούμε ότι ανήκουν στην ίδια γενική κατηγορία αντικειμένων. Με άλλα λόγια, η τεχνολογία δεν διαθέτει ακόμη το υψηλό επίπεδο ακρίβειας που απαιτείται για να λειτουργεί με απαράμιλλη αποδοτικότητα στον πραγματικό πολυποίκιλο κόσμο μας. Καθώς η ανάπτυξη αυτής της τεχνολογίας βρίσκεται ακόμη σε εξέλιξη, απαιτείται μεγάλη ανοχή σε λάθη από τις επιστημονικές ομάδες που εργάζονται πάνω της.

1.1.2 Αναγνώριση Αντικειμένου (Object Recognition)

Το Object Recognition (Αναγνώριση Αντικειμένων), είναι ένας γενικός όρος που περιγράφει το συνονθύλευμα πολλών σχετικών Computer Vision εργασιών, που περιλαμβάνουν την αναγνώριση αντικειμένων σε ψηφιακές εικόνες ή βίντεο, όπου είναι και η βασική τεχνική που θα χρησιμοποιήσουμε στο πέρας της εργασίας μας. Είναι ουσιαστικά η τεχνική, κατά την οποία ένα Computer Vision σύστημα χρησιμοποιεί εξειδικευμένους αλγορίθμους για να κατανοήσει το ποια είδη φρούτων υπάρχουν για παράδειγμα, σε μια εικόνα με πολλά διαφορετικά φρούτα. Αυτό το συνονθύλευμα εργασιών λοιπόν, εμπεριέχει τους όρους των Object Classification, Object Localization και Object Detection όπου αναφέρθηκαν και στην προηγούμενη υποενότητα. Μία περαιτέρω επέκταση σε αυτήν την ανάλυση των Computer Vision εργασιών είναι επίσης το Object Segmentation. Με την παρακάτω ανάλυση (Εικόνα 1.7), μπορούμε να έχουμε μια καλύτερη εικόνα της σειράς των Computer Vision εργασιών που απαρτίζουν το Object Recognition.



Εικόνα 1.7

Ο ακριβής εντοπισμός συγκεκριμένων αντικειμένων σε εικόνες και βίντεο έχει γίνει εξαιρετικά επιτυχής λόγω της αύξησης των Machine Learning και Deep Learning αλγορίθμων. Μερικοί από αυτούς είναι οι, SSD, RetinaNet, YOLO, COCO αλλά και η R-CNN οικογένεια μοντέλων, ένα εκ των οποίων θα χρησιμοποιηθεί και στα επόμενα κεφάλαια της συγκεκριμένης πτυχιακής εργασίας.

1.1.3 Επεξεργασία Εικόνας (Image Processing)

Παρά τις μεγάλες διαφορές στους στόχους αυτών των δύο φαινομενικά διαφορετικών τεχνολογιών, θα μπορούσαμε να πούμε πως το Image Processing αποτελεί εν μέρει υποπεδίο του Computer Vision. Ένα Computer Vision σύστημα χρησιμοποιεί Image Processing αλγόριθμους, για να δοκιμάσει και να πραγματοποιήσει την προσομοίωση της όρασης σε ανθρώπινη κλίμακα. Το Image Processing είναι η μελέτη θεωριών, μοντέλων και αλγορίθμων για την τροποποίηση των εικόνων. Περιλαμβάνει μια μεγάλη ποικιλία θεμάτων όπως

ψηφιοποίηση (digitation), χειρισμός ιστογραμμάτων (histogram manipulation), στρέβλωση (warping), φιλτράρισμα (filtering), τμηματοποίηση (segmentation), αποκατάσταση (restoration), συμπίεση (compression) και πολλά άλλα. Το Image Processing πρωτοπόρησε στο εργαστήριο Jet Propulsion της NASA στα τέλη της δεκαετίας του 1960, για τη μετατροπή αναλογικών σημάτων από το διαστημικό σκάφος Ranger σε ψηφιακές εικόνες. Τώρα, το Image Processing έχει ένα ευρύ φάσμα εφαρμογών, με ιδιαίτερη έμφαση στην ιατρική, όπου μπορούμε να δούμε την συνεισφορά του στους τομείς της αξονικής τομογραφίας, της μαγνητικής τομογραφίας και των υπερήχων. Το Image Processing σχετίζεται κυρίως με τη χρήση και την εφαρμογή μαθηματικών συναρτήσεων και μετασχηματισμών σε εικόνες, ανεξάρτητα από την πραγματοποίηση ή όχι ιδιαίτερα εμφανών συμπερασμάτων πάνω στην ίδια την εικόνα. Σημαίνει απλά ότι ένας αλγόριθμος κάνει κάποιους μετασχηματισμούς στην εικόνα, όπως smoothing, sharpening, contrasting, stretching και άλλα.

Το Computer Vision και το Image Processing συνεργάζονται σε πολλές περιπτώσεις. Πολλά Computer Vision συστήματα βασίζονται σε Image Processing αλγόριθμους. Για παράδειγμα, τα Computer Vision συστήματα σπάνια χρησιμοποιούν ακατέργαστα δεδομένα απεικόνισης που προέρχονται απευθείας από έναν αισθητήρα. Αντ 'αυτού, χρησιμοποιούν εικόνες που υποβάλλονται σε επεξεργασία από έναν επεξεργαστή σήματος εικόνας. Το αντίθετο είναι επίσης δυνατό. Δεν ήταν σύνηθες για έναν Image Processing αλγόριθμο να βασίζεται σε Computer Vision συστήματα στο παρελθόν, αλλά όλο και πιο προηγμένες μέθοδοι επεξεργασίας εικόνας έχουν αρχίσει να χρησιμοποιούν το Computer Vision για να βελτιώσουν τις εικόνες. Τα φίλτρα ομορφιάς προσώπου, για παράδειγμα, χρησιμοποιούν Computer Vision τεχνικές για να ανιχνεύουν πρόσωπα και να εφαρμόζουν επιλεκτικά φίλτρα. Μπορούν να κάνουν πιο προηγμένα πράγματα, όπως ενίσχυση της ευκρίνειας των ματιών ή προσομοίωση ενός προβολέα εντοπίζοντας ορόσημα του προσώπου και συντονίζοντας τις εικόνες τοπικά

1.2 Μηχανική Μάθηση (Machine Learning)

Η μηχανική μάθηση (Machine Learning) είναι μια τεχνική ανάλυσης δεδομένων που διδάσκει στους υπολογιστές να κάνουν ότι εμείς οι άνθρωποι αλλά και τα ζώα κάνουμε αυθόρμητα. Να μαθαίνουμε από την εμπειρία. Το Machine Learning βασίζεται, εν μέρει, σε

ένα μοντέλο αλληλεπίδρασης εγκεφαλικών κυττάρων που δημιουργήθηκε το 1949 από τον Donald Hebb σε βιβλίο με τίτλο “The Organization of Behavior”. Το βιβλίο παρουσιάζει τις θεωρίες του Hebb σχετικά με τον “ενθουσιασμό” των νευρώνων και την επικοινωνία μεταξύ τους. Το Machine Learning είναι μια σημαντική πτυχή των σύγχρονων επιχειρήσεων και της έρευνας. Χρησιμοποιεί αλγόριθμους και μοντέλα νευρωνικών δικτύων για να βοηθήσει τα συστήματα υπολογιστών να βελτιώσουν προοδευτικά την απόδοσή τους. Οι Machine Learning αλγόριθμοι δημιουργούν αυτόματα ένα μαθηματικό μοντέλο χρησιμοποιώντας δείγματα δεδομένων επίσης γνωστά ως "training data", για τη λήψη αποφάσεων χωρίς να προγραμματίζονται συγκεκριμένα για τη λήψη αυτών των αποφάσεων. Οι αλγόριθμοι βελτιώνουν προσαρμοστικά την απόδοσή τους καθώς αυξάνεται ο αριθμός των διαθέσιμων training data. Το Deep Learning είναι μια εξειδικευμένη Machine Learning μορφή. Το Machine Learning και το Computer Vision ευελπιστούν να φέρουν στους υπολογιστές, τις ανθρώπινες δυνατότητες για κατανόηση δεδομένων και ανάληψη δράσης πάντα βάση των αποτελεσμάτων του παρελθόντος και του παρόντος.

Οι Machine Learning λύσεις περιστρέφονται γύρω από τη συλλογή δεδομένων, την εκπαίδευση ενός μοντέλου και τη χρήση του εκπαιδευμένου μοντέλου στο να κάνει προβλέψεις. Υπάρχουν μοντέλα και υπηρεσίες που παρέχονται από ιδιωτικές εταιρείες για αναγνώριση ομιλίας, ανάλυση κειμένου και ταξινόμησης εικόνας όπου οποιοσδήποτε μπορεί να χρησιμοποιήσει τα μοντέλα τους μέσω εφαρμογών προγραμματισμού διεπαφής (API). Για παράδειγμα, Amazon Recognition, Polly, Lex, Microsoft Azure Cognitive Services και IBM Watson είναι μερικά από αυτά.

Με την άνοδο του τομέα του Big Data, το Machine Learning έχει γίνει βασική τεχνική επίλυσης προβλημάτων σε τομείς όπως

- **Computational finance**, για αλγοριθμικές συναλλαγές

- **Image Processing και Computer Vision**, για Object Detection και Recognition, motion και face detection κ.τ.λ.

- **Computational biology**, για τον εντοπισμό όγκων, δημιουργία φαρμάκων κ.τ.λ.

- **Automotive, aerospace and manufacturing**, για την πρόβλεψη διατήρησης πορείας

- **Natural language processing**, για εφαρμογές αναγνώρισης φωνής

Ένα άλλο καλό παράδειγμα εφαρμογής Machine Learning αλγορίθμων στην καθημερινότητά μας είναι οι ιστοσελίδες, όπου βασίζονται σε αυτούς για να περιηγηθούν ανάμεσα σε εκατομμύρια επιλογές ώστε να μας δώσουν προτάσεις για τραγούδια ή ταινίες.

Το Machine Learning χρησιμοποιεί τρείς τύπους τεχνικών. Την εποπτευόμενη μάθηση (supervised learning), η οποία εκπαιδεύει ένα μοντέλο σε γνωστά δεδομένα εισόδου και εξόδου, ώστε να μπορεί να προβλέψει μελλοντικά αποτελέσματα. Την μη εποπτευόμενη μάθηση (unsupervised learning), η οποία βρίσκει κρυμμένα μοτίβα ή εγγενείς δομές στα δεδομένα εισόδου. Και τέλος την ημιεποπτευόμενη μάθηση (semi-supervised learning), η οποία συνδυάζει μια μικρή labeled data ποσότητα και μια μεγάλη unlabeled data ποσότητα δεδομένων κατά τη διάρκεια της εκπαίδευσης και σαν τεχνική εμπίπτει μεταξύ των unsupervised learning και supervised learning τεχνικών.

1.2.1 Βαθιά Μάθηση (Deep Learning)

Τα τελευταία χρόνια, οι μέθοδοι βαθιάς μάθησης (Deep Learning) έχουν αποδειχθεί ότι ξεπερνούν τις προηγούμενες προηγμένες Machine Learning τεχνικές σε διάφορους τομείς, με το Computer Vision να είναι μια από τις πιο εξέχουσες περιπτώσεις. Το Deep Learning επιτρέπει σε υπολογιστικά μοντέλα πολλαπλών επιπέδων επεξεργασίας, να μαθαίνουν και να παρουσιάζουν δεδομένα με πολλαπλά επίπεδα αφαίρεσης, που μιμούνται τον τρόπο με τον οποίο ο ανθρώπινος εγκέφαλος αντιλαμβάνεται και κατανοεί τις πολυτροπικές πληροφορίες, καταγράφοντας συνεπώς περίπλοκες δομές δεδομένων μεγάλης κλίμακας. Το Deep Learning είναι μια πλουραλιστική οικογένεια μεθόδων, που περιλαμβάνει νευρωνικά δίκτυα (Neural Networks), ιεραρχικά πιθανοτικά μοντέλα (hierarchical probabilistic methods) και μια ποικιλία

από unsupervised και supervised αλγόριθμους εκμάθησης χαρακτηριστικών. Το ‘Deep’ στο Deep Learning, αναφέρεται στο βάθος των στοιβάδων σε ένα Neural Network. Ένα Neural Network που αποτελείται από περισσότερες από τρείς στοιβάδες οι οποίες θα περιλαμβάνουν τις εισόδους και την έξοδο, μπορεί να θεωρηθεί Deep Learning αλγόριθμος. Η πρόσφατη αύξηση του ενδιαφέροντος για τις Deep Learning μεθόδους, οφείλεται στο γεγονός ότι έχουν αποδειχθεί πως ξεπερνούν τις προηγούμενες προηγμένες τεχνικές σε πολλές εργασίες, καθώς και η προσαρμοστικότητα στην αφθονία σύνθετων δεδομένων από διαφορετικές πηγές (π.χ. οπτικά δεδομένα, δεδομένα ήχου, ιατρικά δεδομένα, κοινωνικά δεδομένα κ.τ.λ.)

Η φιλοδοξία για τη δημιουργία ενός συστήματος που προσομοιώνει τον ανθρώπινο εγκέφαλο, τροφοδότησε την αρχική ανάπτυξη των Neural Networks. Το 1943, οι McCulloch και Pitts προσπάθησαν να καταλάβουν πως ο εγκέφαλος μπορούσε να παράγει πολύπλοκα μοτίβα χρησιμοποιώντας διασυνδεδεμένα βασικά κύτταρα, που ονομάζονται νευρώνες. Το μοντέλο McCulloch και Pitts ενός νευρώνα, που ονομάζεται μοντέλο MCP, συνέβαλε σημαντικά στην ανάπτυξη τεχνητών νευρωνικών δικτύων (Artificial Neural Networks) και σε συνδυασμό με μια από τις σημαντικότερες πρωτοπορίες στη χώρο, αυτή του Hinton το 2006 όπου εισήγαγε μια νέα Deep Neural Network κλάση, την Deep Belief Network (DBN), συνέβαλαν στην εκτόξευση των Deep Architecture και Deep Learning αλγορίθμων. Επίσης, μεταξύ των πιο σημαντικών παραγόντων που συνέβαλαν στην αυτή την τεράστια ώθηση του Deep Learning είναι η εμφάνιση μεγάλων, υψηλής ποιότητας, διαθέσιμων στο κοινό labelled δεδομένων και μαζί με την αύξηση των δυνατοτήτων του parallel GPU computing, η οποία επέτρεψε τη μετάβαση από CPU based σε GPU based εκπαίδευση, επιτρέποντας έτσι σημαντική επιτάχυνση στην εκπαίδευση των Deep Learning μοντέλων.

Πρακτικά, όπως είπαμε και νωρίτερα το Deep Learning είναι μόνο ένα υποσύνολο του Machine Learning με κύρια διαφορά τους, τον τρόπο με τον οποίο ο κάθε αλγόριθμος ‘μαθαίνει’. Ενώ τα βασικά Machine Learning μοντέλα γίνονται σταδιακά καλύτερα σε όποια και αν είναι η λειτουργία τους, χρειάζονται ακόμη κάποια καθοδήγηση. Αυτό σημαίνει πως οι αλγόριθμοι εξαρτόνται από την ανθρώπινη παρέμβαση για να μάθουν, απαιτώντας labelled σύνολα δεδομένων για την κατανόηση των διαφορών μεταξύ των δεδομένων εισόδου. Εάν ένας αλγόριθμος Machine Learning επιστρέψει μια ανακριβή πρόβλεψη, τότε ένας μηχανικός πρέπει να επέμβει και να κάνει τις απαραίτητες προσαρμογές. Το Deep Learning μπορεί επίσης να αξιοποιήσει τα labelled σετ δεδομένων για να ενημερώσει τον αλγόριθμό της, αλλά δεν απαιτεί απαραίτητα να υπάρχει ένα labelled σύνολο δεδομένων. Αντ 'αυτού, μπορεί επίσης να υποστηρίξει unsupervised learning για να εκπαιδεύσει τον εαυτό του. Τέλος ενώ το supervised

learning, η συνηθέστερη Machine Learning τεχνική αξιοποιεί labelled δεδομένα, το unsupervised learning χρησιμοποιεί unlabelled ή unstructured δεδομένα. Σε ένα Deep Learning μοντέλο, ένας αλγόριθμος μπορεί να καθορίσει μόνος του εάν μια πρόβλεψη είναι ακριβής ή όχι, μέσω του δικού του νευρωνικού δικτύου.

1.3 Νευρωνικά Δίκτυα (Neural Networks)

Ένα Neural Network ή πιο στοχευμένα ένα Artificial Neural Networks (ANN) είναι μια σειρά αλγορίθμων που προσπαθούν να αναγνωρίσουν τις υποκείμενες σχέσεις σε ένα σύνολο δεδομένων, μέσω μιας διαδικασίας που μιμείται τον τρόπο λειτουργίας του ανθρώπινου εγκεφάλου. Υπό αυτήν την έννοια, τα Neural Networks αναφέρονται σε συστήματα νευρώνων, οργανικής ή τεχνητής φύσης τα οποία μπορούν να προσαρμοστούν στην αλλαγή εισόδων, έτσι ώστε το δίκτυο να δημιουργήσει το καλύτερο δυνατό αποτέλεσμα χωρίς να χρειάζεται να επανασχεδιάσουμε τα κριτήρια εξόδου. Οι περισσότερες Deep Learning εφαρμογές χρησιμοποιούν Convolutional Neural Networks, στα οποία οι κόμβοι κάθε στοιβάδας συμπλέκονται, οι συμπλεγμένοι κόμβοι κάνουν overlap και κάθε σύμπλεγμα τροφοδοτεί δεδομένα σε πολλούς κόμβους της επόμενης στοιβάδας. Τα Neural Networks ουσιαστικά είναι μια νέα ονομασία προσέγγισης για την τεχνητή νοημοσύνη από το Deep Learning.

Όπως είπαμε και σε προηγούμενο υποκεφάλαιο, τα Neural Networks προτάθηκαν για πρώτη φορά το 1944 από τους Warren McCullough και Walter Pitts, δύο ερευνητές του Πανεπιστημίου του Σικάγο και τα οποία εισέρχονται και εξέρχονται από τη μόδα για περισσότερα από 70 χρόνια. Υπήρξαν σημαντικός τομέας έρευνας τόσο στη νευροεπιστήμη όσο και στην επιστήμη των υπολογιστών μέχρι το 1969, όταν, οι μαθηματικοί του MIT Marvin Minsky και Seymour Papert, κατήργησαν την γενική έννοια των Neural Networks. Λίγο αργότερα, στην δεκαετία του 1980, η τεχνική αναζωπυρώθηκε μόνο για να πέσει ξανά σε έκλειψη την πρώτη δεκαετία του νέου αιώνα και επέστρεψε για τα καλά στη δεύτερη, τροφοδοτούμενη σε μεγάλο βαθμό από την αυξημένη ισχύ επεξεργασίας των graphic chips.

Στα Neural Networks οι νευρώνες κατηγοριοποιούνται σε τρείς διαφορετικούς τύπους στοιβάδων, την στοιβάδα εισαγωγής (input layer), την κρυφή στοιβάδα (hidden layer) και την στοιβάδα εξόδου (output layer).

- Το **input layer** λαμβάνει τα δεδομένα εισόδου και τα μεταπερνά στην επόμενη στοιβάδα.

- Το **hidden layer** εκτελεί μαθηματικούς υπολογισμούς στις εισόδους. Μία από τις προκλήσεις στη δημιουργία Neural Networks είναι η απόφαση για τον αριθμός των hidden layers, καθώς και ο αριθμός των νευρώνων για κάθε στοιβάδα. Το 'Deep' στο Deep Learning, αναφέρεται στο γεγονός ύπαρξης παραπάνω από μία hidden layer στοιβάδων.

- Το **output layer** επιστρέφει τα δεδομένα εξόδου.

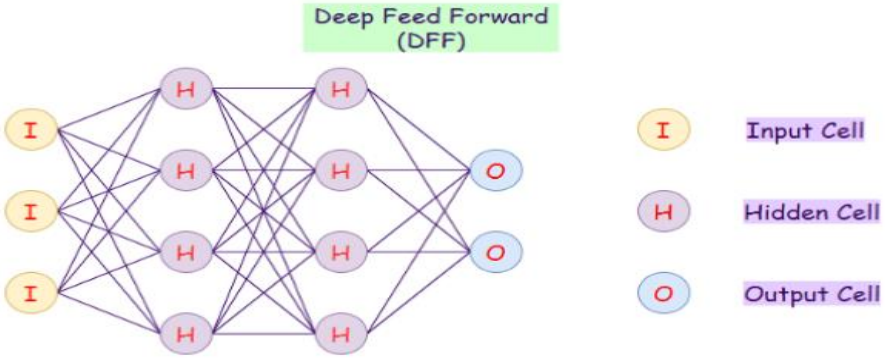
Σε βασικό επίπεδο, ένα Neural Network αποτελείται από τέσσερα κύρια συστατικά: inputs, weights, bias ή threshold, output. Τα weights και bias είναι οι παράμετροι εκμάθησης στο Neural Network. Τα weights ελέγχουν το σήμα μεταξύ δύο νευρώνων ή αλλιώς την δύναμη της σύνδεσής τους. Με άλλα λόγια, ένα weight αποφασίζει πόση επιρροή θα έχει η είσοδος στην έξοδο. Τα biases, τα οποία είναι σταθερά, είναι μια επιπλέον είσοδος στην επόμενη στοιβάδα που θα έχει πάντα την τιμή 1. Τα biases δεν επηρεάζονται από την προηγούμενη στοιβάδα (δεν έχουν εισερχόμενες συνδέσεις) αλλά έχουν εξερχόμενες συνδέσεις με τις δικές τους weights. Τα biases εγγυόνται ότι ακόμη και όταν όλες οι είσοδοι είναι μηδενικά, θα εξακολουθεί να υπάρχει ενεργοποίηση στον νευρώνα.

Όταν ένα Neural Network έχει πολλά επίπεδα, ονομάζεται Deep Neural Network και η διαδικασία εκπαίδευσης και χρήσης των Deep Neural Networks ονομάζεται Deep Learning. Τα Deep Neural Network αναφέρονται γενικά σε ιδιαίτερα σύνθετα Neural Networks όπου έχουν περισσότερες στοιβάδες (έως 1.000) και συνήθως περισσότερους νευρώνες ανά στοιβάδα. Με περισσότερες στοιβάδες και περισσότερους νευρώνες, τα δίκτυα μπορούν να χειριστούν όλο και πιο περίπλοκες εργασίες, αλλά αυτό σημαίνει ότι χρειάζονται και περισσότερο χρόνο για να εκπαιδευτούν. Σήμερα, υπάρχουν πολλοί Neural Networks τύποι στο Deep Learning, που χρησιμοποιούνται για διαφορετικούς σκοπούς.

- **Deep Feed-forward (DFF)**

  Είναι ένα δίκτυο προώθησης τροφοδοσίας (feed-forward network) που χρησιμοποιεί περισσότερα από ένα hidden layer. Το κύριο πρόβλημα με τη χρήση μόνο ενός hidden layer είναι αυτό του overfitting, επομένως με την προσθήκη περισσότερων hidden layers, ενδέχεται να επιτύχουμε (όχι σε όλες τις περιπτώσεις) μειωμένο overfitting και βελτιωμένο generalization.

  Εφαρμογές: Data Compression, Pattern Recognition, Computer Vision, Noise Filtering, Financial Prediction κ.τ.λ.

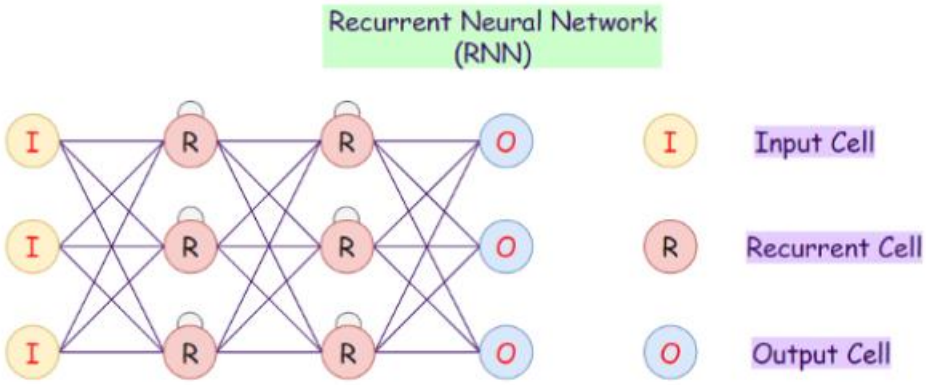


Εικόνα 1.8-DFF Neural Network

- **Recurrent Neural Network (RNN)**

  Στο RNN, καθένας από τους νευρώνες σε hidden layer λαμβάνει μια είσοδο με συγκεκριμένη καθυστέρηση στο χρόνο. Χρησιμοποιούμε αυτόν τον τύπο νευρωνικού δικτύου όπου πρέπει να έχουμε πρόσβαση σε προηγούμενες πληροφορίες σε τρέχουσες επαναλήψεις. Για παράδειγμα, όταν προσπαθούμε να προβλέψουμε την επόμενη λέξη σε μια πρόταση, πρέπει πρώτα να γνωρίζουμε τις λέξεις που χρησιμοποιήθηκαν προηγουμένως. Το μέγεθος του μοντέλου δεν αυξάνεται με το μέγεθος της εισαγωγής και οι υπολογισμοί σε αυτό το μοντέλο λαμβάνουν υπόψη τις προηγηθείσες πληροφορίες. Ωστόσο, το πρόβλημα με αυτό το νευρωνικό δίκτυο είναι η αργή υπολογιστική ταχύτητα.

  Εφαρμογές: Machine Translation, Robot Control, Time Series Prediction, Speech Recognition, Speech Synthesis, Music Composition, Rhythm Learning κ.τ.λ.

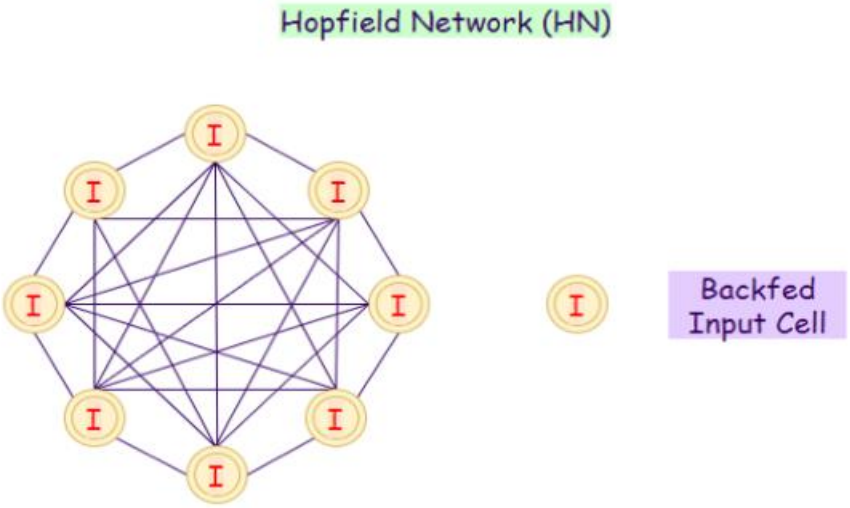


Εικόνα 1.9-Recurrent Neural Network

- **Hopfield Network (HN)**

  Στα HN, κάθε νευρώνας συνδέεται άμεσα με άλλους νευρώνες. Σε αυτό το δίκτυο, ένας νευρώνας είναι είτε ON είτε OFF. Η κατάσταση των νευρώνων μπορεί να αλλάξει λαμβάνοντας inputs από άλλους νευρώνες. Γενικά χρησιμοποιούμε δίκτυα Hopfield για να αποθηκεύσουμε μοτίβα και αναμνήσεις. Όταν εκπαιδεύουμε ένα νευρωνικό δίκτυο σε ένα σύνολο μοτίβων, τότε μπορεί να αναγνωρίσει το μοτίβο ακόμη και αν είναι κάπως παραμορφωμένο ή ελλιπές. Μπορεί να αναγνωρίσει το πλήρες μοτίβο όταν το τροφοδοτούμε με ελλιπή inputs, επιστρέφοντας και την καλύτερη εικασία.

  Εφαρμογές: Optimization Problems, Image Detection and Recognition, Medical Image Recognition, Enhancing X-Ray Images κ.τ.λ.

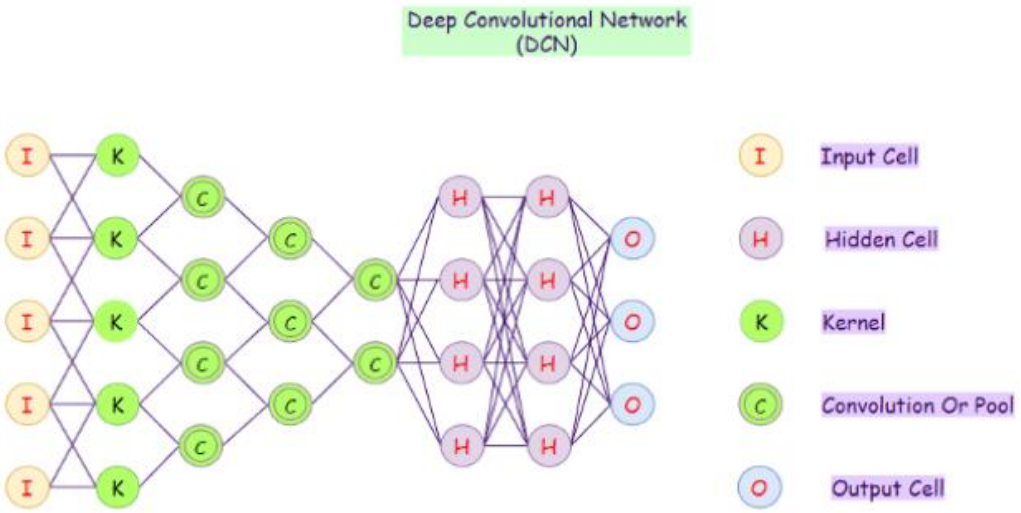


Εικόνα 1.10-HN Neural Network

- **Deep Convolutional Neural Network (Deep CNN)**

  Τα Deep CNN επιτρέπουν την unsupervised κατασκευή ιεραρχικών αναπαραστάσεων εικόνας. Τα DNN χρησιμοποιούνται για να προσθέσουν πολύ πιο περίπλοκα χαρακτηριστικά στην ιεραρχική απεικόνιση εικόνας, ώστε να μπορεί να εκτελεί την εργασία με καλύτερη ακρίβεια.

  Εφαρμογές: Identify Faces, Street Sings, Tumors, Image Recognition, Video Analysis, Anomaly Detection, Drug Discovery, Time Series Forecasting κ.τ.λ.

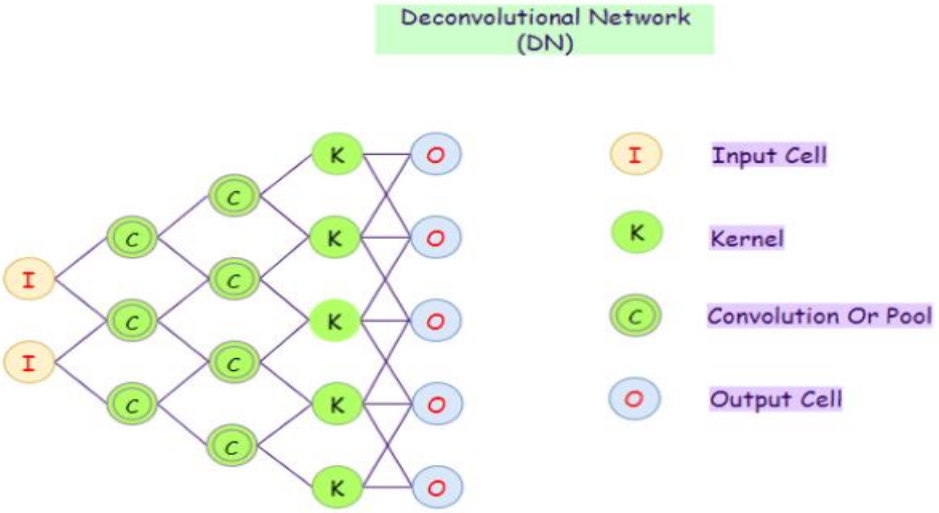


Εικόνα 1.11-Deep CNN

- **Deconvolutional Neural Network (DNN)**

  Τα DNN είναι CNN που λειτουργούν σε μια αντίστροφη διαδικασία. Παρόλο που ένα DNN είναι παρόμοιο με το CNN στη φύση της εργασίας, η εφαρμογή του στην τεχνητή νοημοσύνη είναι πολύ διαφορετική. Τα DNN βοηθούν στην εύρεση χαμένων λειτουργιών ή σημάτων σε δίκτυα που θεωρούνταν χρήσιμα στο παρελθόν. Ένα DNN μπορεί να χάσει ένα σήμα λόγω της πιθανής περιπλοκής του με άλλα σήματα. Ένα DNN μπορεί να πάρει έναν vector και να βγάλει μια εικόνα από αυτόν.

  Εφαρμογές: Image super-resolution, Surface depth estimation for an image, Optical flow estimation κ.τ.λ.

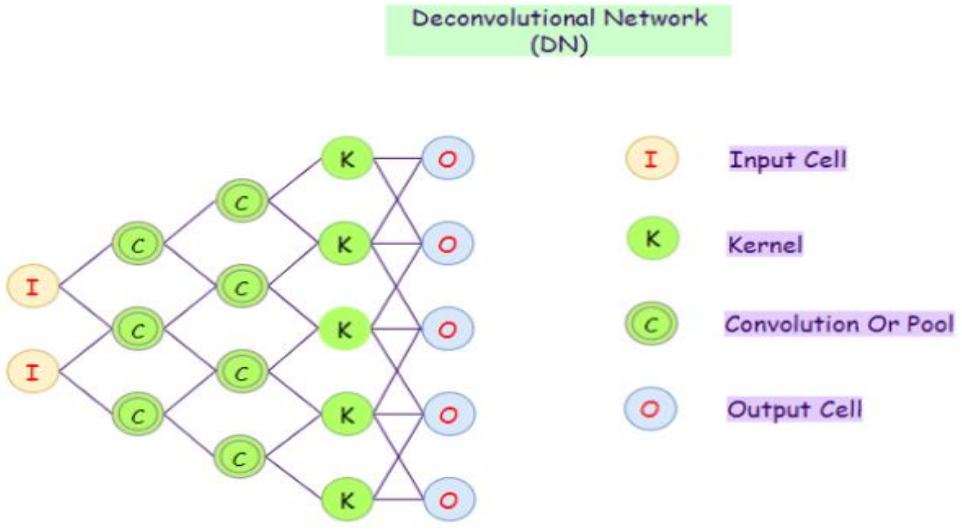


Εικόνα 1.12-Deconvolutional Neural Network

- **Deep Residual Network (DRN)**

  Τα Deep Neural Networks με πολλές στοιβαδες μπορεί να είναι δύσκολο να εκπαιδευτούν και να χρειάζονται πολύ χρόνο, όπως επίσης και να οδηγήσουν στην υποβάθμιση των αποτελεσμάτων. Τα Deep Residual Networks (DRN) αποτρέπουν την υποβάθμιση των αποτελεσμάτων, παρόλο που έχουν πολλές στοιβάδες. Με τα DRN, ορισμένα μέρη των input περνούν επόμενη στοιβάδα. Επομένως, αυτά τα δίκτυα μπορεί να είναι αρκετά βαθιά (Μπορεί να περιέχει περίπου 300 στοιβάδες).

  Εφαρμογές: Image Classification, Object Detection, Semantic Segmentation, Speech Recognition, Language Recognition κ.τ.λ.



Εικόνα 1.13-DR Neural

Η υπόσχεση του Deep Learning και των Neural Networks δεν είναι πως οι υπολογιστές θα αρχίσουν να σκέφτονται σαν τους ανθρώπους. Είναι σαν να ζητάς από ένα μήλο να γίνει πορτοκάλι. Αντίθετα, αποδεικνύει ότι δεδομένου ενός αρκετά μεγάλου συνόλου δεδομένων, αρκετά γρήγορων επεξεργαστών και ενός αρκετά εξελιγμένου αλγορίθμου, οι υπολογιστές μπορούν να αρχίσουν να εκτελούν εργασίες όπου μέχρι πρότινος είχαν αφεθεί αποκλειστικά στο βασίλειο της ανθρώπινης αντίληψης.

Κεφάλαιο 2- Θεωρητικό Υπόβαθρο

2.1 Η Έννοια των Νευρωνικών Δικτύων (Neural Networks Concept)

Εδώ έχουμε ένα γλωσσάριο βασικών όρων με τους οποίους πρέπει να είμαστε εξοικειωμένοι πριν εμβαθύνουμε περισσότερο στις λεπτομέρειες των Neural Networks.

Inputs: Τα δεδομένα εισόδου τροφοδοτούνται στο Neural Networks, με στόχο τη λήψη απόφασης ή πρόβλεψης για τα δεδομένα. Οι είσοδοι σε ένα Neural Networks είναι συνήθως ένα σύνολο πραγματικών τιμών όπου κάθε τιμή τροφοδοτείται σε έναν από τους νευρώνες στη στοιβάδα εισόδου.

Training Set: Ένα σύνολο inputs για τις οποίες οι σωστές έξοδοι είναι γνωστές και χρησιμοποιούνται για την εκπαίδευση του Neural Networks.

Outputs: Τα Neural Networks δημιουργούν τις προβλέψεις τους με τη μορφή ενός συνόλου πραγματικών τιμών ή δυαδικών (boolean) αποφάσεων. Κάθε τιμή εξόδου δημιουργείται από έναν από τους νευρώνες στην στοιβάδα εξόδου.

Neuron/Perceptron: Το βασικό στοιχείο ενός Neural Networks. Δέχεται ένα input και παράγει μια πρόβλεψη.

Activation Function: Κάθε νευρώνας δέχεται μέρος του input και το περνά μέσω της activation function. Τα κοινά activation function είναι τα sigmoid, TanH και ReLu. Τα activation function συμβάλλουν στη δημιουργία output τιμών εντός αποδεκτού εύρους και η μη γραμμική (non-linear)μορφή τους είναι ζωτικής σημασίας για την εκπαίδευση του δικτύου.

Weight Space: Σε κάθε νευρώνα αποδίδεται ένα αριθμητικό βάρος (numeric weight). Τα weights, μαζί με το activation function, καθορίζουν το output κάθε νευρώνα. Τα Neural Networks εκπαιδεύονται με fine-tuning weights, για να ανακαλύψουν το βέλτιστο σύνολο weights που δημιουργεί την πιο ακριβή πρόβλεψη.

Forward Pass: Το forward pass παίρνει τα inputs, τα περνά μέσω του δικτύου και επιτρέπει σε κάθε νευρώνα να αντιδρά σε ένα input κλάσμα. Οι νευρώνες δημιουργούν τα outputs τους και τα μεταδίδουν στην επόμενη στοιβάδα, έως ότου τελικά το δίκτυο παράγει ένα output.

Error Function: Ορίζει πόση απόκλιση έχει το output του τρέχοντος μοντέλου από το σωστό output. Κατά την εκπαίδευση του μοντέλου, ο στόχος είναι να ελαχιστοποιηθεί το error function και να φέρουμε το output όσο το δυνατόν πιο κοντά στη σωστή τιμή.

Backpropagation: Προκειμένου να ανακαλύψουμε τα βέλτιστα weights για τους νευρώνες, εκτελούμε μια οπισθοδρόμηση (backpropagation), πηγαίνοντας προς τα πίσω, από την πρόβλεψη του δικτύου στους νευρώνες που δημιούργησαν αυτήν την πρόβλεψη. Αυτό ονομάζεται backpropagation. Το backpropagation παρακολουθεί τα παράγωγα των activation functions σε κάθε διαδοχικό νευρώνα, για να βρει weights που μειώνουν το error function στο ελάχιστο, το οποίο και θα δημιουργήσει την καλύτερη πρόβλεψη. Αυτή είναι μια μαθηματική διαδικασία που ονομάζεται backpropagation.

Bias and Variance: Κατά την εκπαίδευση των Neural Networks, όπως και σε άλλες Machine Learning τεχνικές, προσπαθούμε να ισορροπήσουμε μεταξύ μεροληψίας (Bias) και διακύμανσης (Variance). Το bias μετρά πόσο καλά το μοντέλο ταιριάζει στο σετ εκπαίδευσης, ώστε να είναι ικανό να προβλέψει σωστά τα γνωστά outputs των παραδειγμάτων εκπαίδευσης.

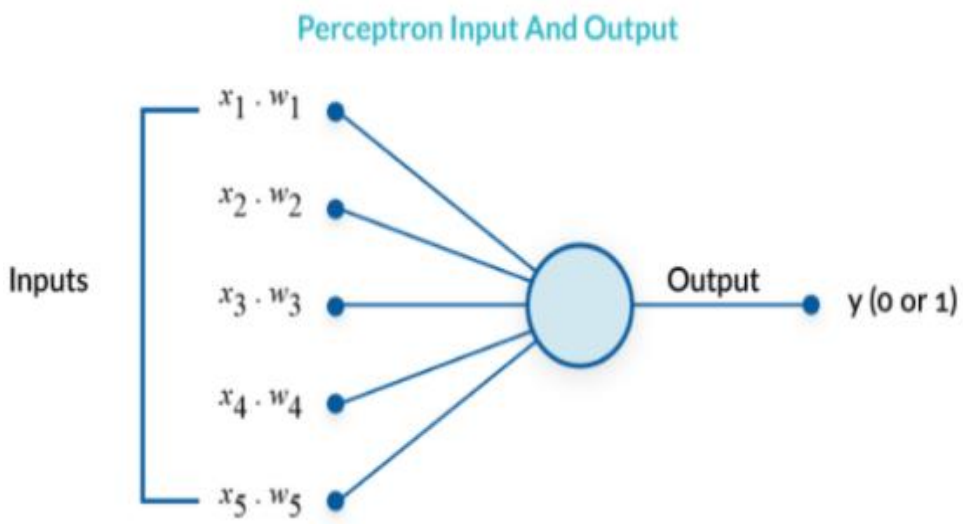
Το variance μετρά πόσο καλά λειτουργεί το μοντέλο με άγνωστα inputs που δεν ήταν διαθέσιμα κατά τη διάρκεια της εκπαίδευσης. Μια άλλη έννοια του variance είναι ένας "bias neuron" που χρησιμοποιείται σε κάθε στοιβάδα του Neural Network. Ο bias neuron κρατά τον αριθμό 1 και καθιστά δυνατή τη μετακίνηση του activation function πάνω, κάτω, αριστερά και δεξιά στο γράφημα αριθμών.

Hyperparameters: Μια υπερπαράμετρος (hyperparameter) είναι μια ρύθμιση που επηρεάζει τη δομή ή τη λειτουργία του Neural Network. Σε πραγματικά Deep Learning projects, ο συντονισμός των hyperparameter είναι ο πρωταρχικός τρόπος δημιουργίας ενός δικτύου που παρέχει ακριβείς προβλέψεις για ένα συγκεκριμένο πρόβλημα. Τα κοινά hyperparameter περιλαμβάνουν τον αριθμό των hidden layers, το activation function και πόσες φορές η εκπαίδευση πρέπει να επαναληφθεί.

2.1.1 Αντίληπτρο και Πολυεπίπεδο Αντίληπτρο (Perceptron Και Multilayer Perceptron)

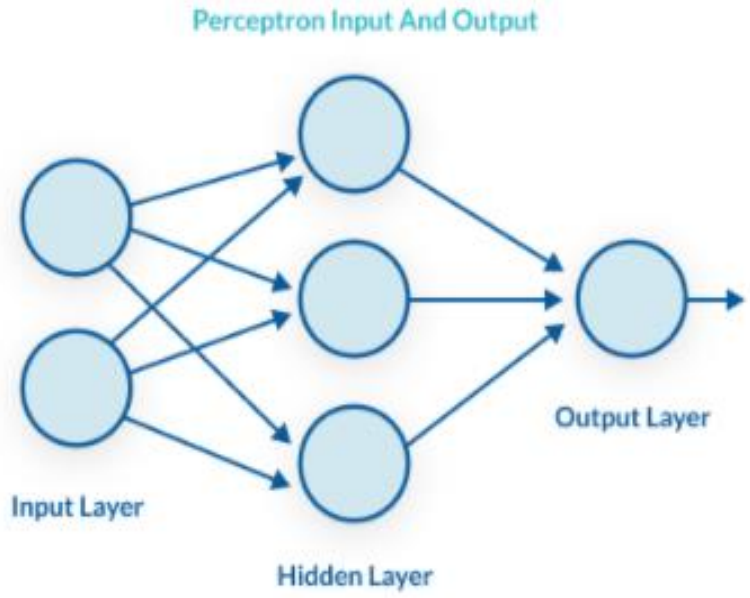
Οι Perceptron και Multilayer Perceptron νευρώνες, αποτελούν τους θεμέλιους λίθους των Neural Networks.

Το Perceptron είναι ένας αλγόριθμος δυαδικής ταξινόμησης που έχει διαμορφωθεί σύμφωνα με τη λειτουργία του ανθρώπινου εγκεφάλου και προοριζόταν να μιμηθεί τον νευρώνα. Το Perceptron, αν και έχει μια απλή δομή, έχει την ικανότητα να μαθαίνει και να επιλύει πολύ περίπλοκα προβλήματα.



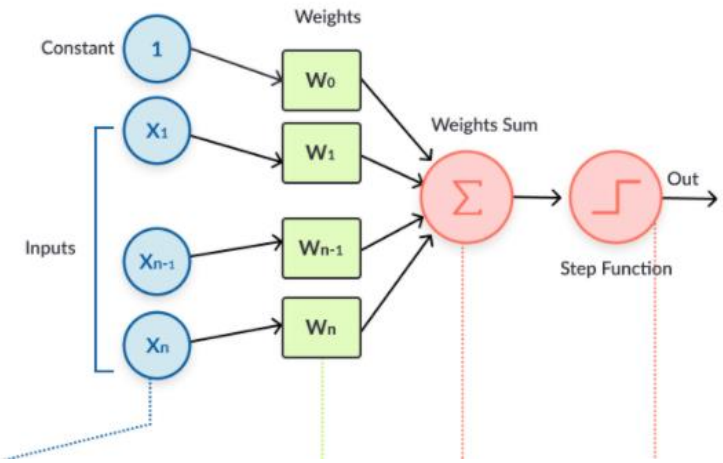
Εικόνα 2.1-Perceptron

Το Multilayer Perceptron (MLP) είναι μια ομάδα Perceptron, οργανωμένη σε πολλαπλές στοιβάδες, που μπορεί να απαντήσει με ακρίβεια σε πολύπλοκες ερωτήσεις. Κάθε Perceptron στην πρώτη στοιβάδα (στα αριστερά) στέλνει σήματα σε όλα τα Perceptron στη δεύτερη στοιβάδα, και ούτω καθεξής. Ένα MLP περιέχει μια input στοιβάδα, τουλάχιστον ένα hidden layer και μια output στοιβάδα.



Εικόνα 2.2 -Multilayer Perceptron

Η διαδικασία εκπαίδευσης ενός Perceptron έχει ως εξής:



Εικόνα 2.3 -Σχεδιάγραμμα Εκπαίδευσης Perceptron

1. Παίρνει τα inputs που τροφοδοτούνται στα Perceptron στη στοιβάδα εισόδου, τα πολλαπλασιάζει με τα weights τους και υπολογίζει το άθροισμα (sum).

2. Προσθέτει το νούμερο ένα, πολλαπλασιαζόμενο με bias weight. Αυτό είναι ένα τεχνικό βήμα που καθιστά δυνατή τη μετακίνηση της λειτουργίας εξόδου κάθε Perceptron (Activation Function) πάνω, κάτω, αριστερά και δεξιά στο γράφημα αριθμών.

3. Τροφοδοτεί το άθροισμα μέσω του Activation Function. Σε ένα απλό Perceptron σύστημα, το Activation Function είναι μια λειτουργία βήμα (step function).

4. Το αποτέλεσμα του step function, είναι το output.

Ένα Multilayer Perceptron είναι αρκετά παρόμοιο με ένα σύγχρονο Neural Network. Προσθέτοντας μερικά συστατικά, η Perceptron αρχιτεκτονική γίνεται ένα ολοκληρωμένο Deep Learning σύστημα:

- **Activation Functions** και άλλες **Hyperparameters.** Ένα πλήρες Neural Network χρησιμοποιεί μια ποικιλία από Activation Function που εξάγουν πραγματικές

τιμές και όχι δυαδικές τιμές όπως στο κλασικό Perceptron. Είναι πιο ευέλικτο όσον αφορά άλλες λεπτομέρειες της εκπαιδευτικής διαδικασίας, όπως ο αριθμός των επαναλήψεων (training iterations), τα σχήματα αρχικοποίησης βάρους (weight initialization schemes), τακτοποίηση (regularization) και ούτω καθεξής. Όλα αυτά μπορούν να οριστούν ως Hyperparameters.

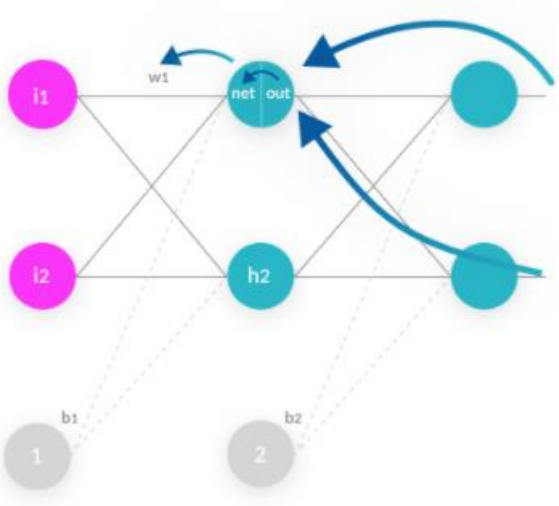
- **Backpropagation.** Ένα πλήρες Neural Network χρησιμοποιεί τον αλγόριθμο Backpropagation, για να εκτελέσει επαναλαμβανόμενες οπίσθιες μεταβάσεις που προσπαθούν να βρουν τις βέλτιστες τιμές των weights των Perceptron, για να δημιουργήσουν την πιο ακριβής πρόβλεψη.

- **Advanced Architectures.** τα πλήρες Neural Network μπορούν να έχουν μια ποικιλία αρχιτεκτονικών που μπορούν να βοηθήσουν στην επίλυση συγκεκριμένων προβλημάτων. Μερικά παραδείγματα είναι τα Recurrent Neural Networks (RNN), τα Convolutional Neural Networks (CNN) και τα Generative Adversarial Networks (GAN).

2.1.2 Οπισθοδιάδοση στα Νευρωνικά Δίκτυα (Backpropagation In Neural Networks)

Τι είναι το Backpropagation και γιατί είναι σημαντικό; Αφού καθοριστεί ένα Neural Network με αρχικά weights και εκτελεστεί μια κίνηση προς τα εμπρός για τη δημιουργία της αρχικής πρόβλεψης, υπάρχει ένα error function που καθορίζει πόσο μακριά είναι το μοντέλο από την πραγματική πρόβλεψη. Υπάρχουν πολλοί πιθανοί αλγόριθμοι που μπορούν να ελαχιστοποιήσουν το error function, για παράδειγμα, κάποιος θα μπορούσε να κάνει μια ωμή αναζήτηση (brute force search) για να βρει τα weights που δημιουργούν το μικρότερο σφάλμα (error). Ωστόσο, για μεγάλα Neural Networks, απαιτείται ένας αλγόριθμος εκπαίδευσης που είναι πολύ υπολογιστικά αποτελεσματικός. Το Backpropagation είναι αυτός ο αλγόριθμος. Μπορεί να ανακαλύψει τα βέλτιστα weights σχετικά γρήγορα, ακόμη και για ένα δίκτυο με εκατομμύρια weights.

Το Backpropagation δουλεύει ως εξής:



Εικόνα 2.4-Τρόπος Λειτουργείας Backpropagation

1. **Forward pass.** Τα weights αρχικοποιούνται και τα inputs από το training set τροφοδοτούνται στο δίκτυο. Η κίνηση προς τα εμπρός πραγματοποιείται και το μοντέλο δημιουργεί την αρχική πρόβλεψη.

2. **Error function.** Το Error function υπολογίζεται ελέγχοντας πόσο μακριά είναι η πρόβλεψη από τη γνωστή πραγματική τιμή.

3. **Backpropagation.** Ο Backpropagation αλγόριθμος υπολογίζει πόσο επηρεάζονται οι output τιμές από κάθε ένα από τα weights του μοντέλου.

4. **Weight update.** Τα weights μπορούν να ενημερώνονται μετά από κάθε δείγμα στο training set, αλλά αυτό συνήθως δεν είναι πρακτικό. Συνήθως, μια παρτίδα δειγμάτων εκτελείται σε ένα μεγάλο πέρασμα προς τα εμπρός (forward pass), και στη συνέχεια εκτελείται Backpropagation στο συνολικό αποτέλεσμα. Το μέγεθος της παρτίδας και ο αριθμός των παρτίδων που χρησιμοποιούνται στην εκπαίδευση, ονομάζονται επαναλήψεις (iterations), είναι σημαντικοί Hyperparameters που έχουν ρυθμιστεί για να έχουν τα καλύτερα αποτελέσματα. Η εκτέλεση ολόκληρου του training set μέσω της διαδικασίας Backpropagation ονομάζεται epoch.

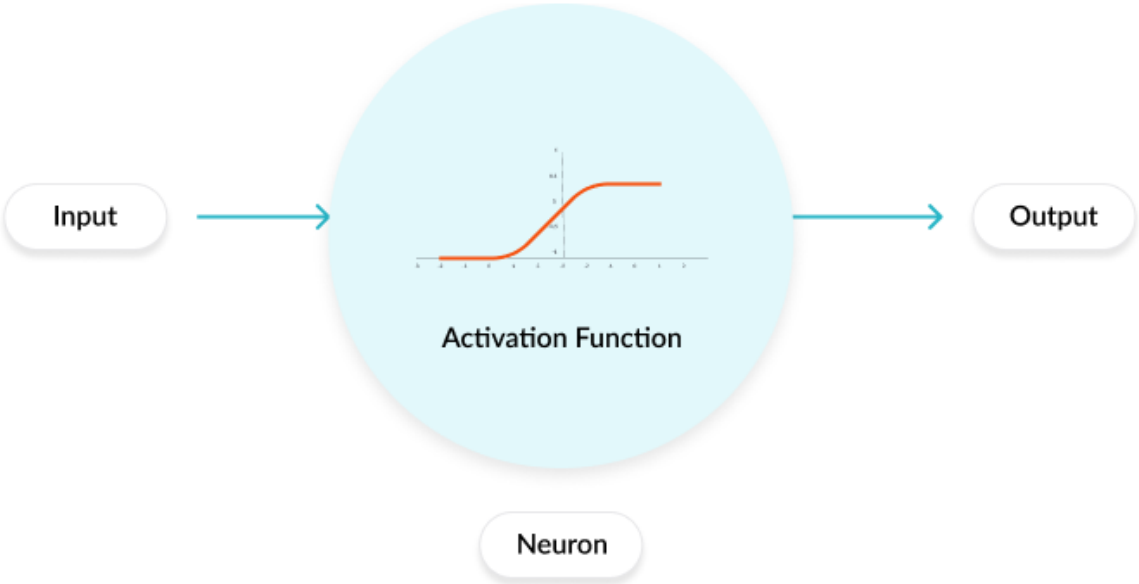
Στον πραγματικό κόσμο, πιθανότατα δεν θα κωδικοποιήσουμε μια εφαρμογή του Backpropagation, επειδή άλλοι το έχουν ήδη κάνει για εμάς. Μπορούμε να εργαστούμε με Deep Learning frameworks όπως το TensorFlow ή το Keras, τα οποία περιέχουν αποτελεσματικές Backpropagation εφαρμογές, τις οποίες μπορούμε να εκτελέσουμε με λίγες μόνο γραμμές κώδικα.

2.1.3 Λειτουργία Ενεργοποίησης στα Νευρωνικά Δίκτυα (Neural Network Activation Function)

Μια Activation Function είναι μια μαθηματική εξίσωση που καθορίζει την έξοδο κάθε στοιχείου (perceptron ή neuron) στο Neural Network. Παίρνει τα inputs από κάθε νευρώνα και το μετατρέπει σε outputs, συνήθως μεταξύ ενός και μηδέν ή μεταξύ -1 και ενός. Οι κλασικές Activation Function που χρησιμοποιούνται σε Neural Network περιλαμβάνουν τη λειτουργία βήματος (step function), το sigmoid και το TanH. Νέες Activation Functions, που αποσκοπούν στη βελτίωση της υπολογιστικής απόδοσης, περιλαμβάνουν ReLu και Swish. Σε ένα Neural Network, τα inputs, τα οποία είναι συνήθως πραγματικές τιμές, τροφοδοτούνται στους νευρώνες του δικτύου. Κάθε νευρώνας έχει weight και τα inputs πολλαπλασιάζονται με το weight και τροφοδοτούνται στην Activation Function.

Το output κάθε νευρώνα είναι το input της επόμενης στοιβάδας νευρώνων του δικτύου, και έτσι τα input καταρρέουν μέσω πολλαπλών activation functions έως ότου τελικά, η output στοιβάδα δημιουργεί μια πρόβλεψη. Τα Neural Networks βασίζονται σε μη γραμμικές activation functions. Το παράγωγο της activation function βοηθά το δίκτυο να “μάθει” μέσω της διαδικασίας Backpropagation.

Η επιλογή μιας activation functions είναι ζωτικής σημασίας για τη δημιουργία και την εκπαίδευση στο δίκτυο. Σε real world Neural Network projects, η activation functions είναι ένα hyperparameter.



Εικόνα 2.5- Activation Function

2.1.4 Πόλωση Νευρωνικών Δικτύων (Neural Network Bias)

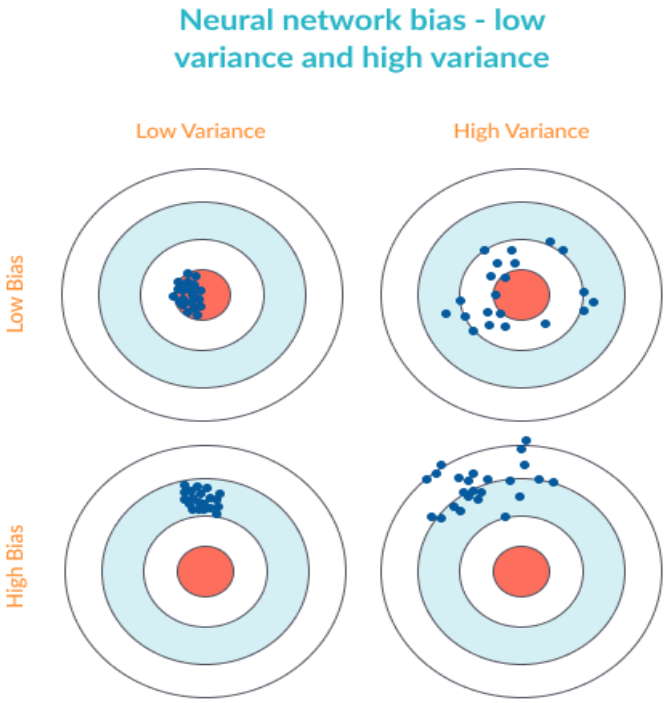
Στα Neural Networks, η λέξη bias έχει δύο ερμηνείες. Μπορεί να σημαίνει ένας νευρώνας bias, που είναι κομμάτι της δομής του Neural Network ή μπορεί να έχει την έννοια ενός στατιστικού concept, το οποίο αντανακλά το πόσο καλά το δίκτυο είναι ικανό να παράγει προβλέψεις, βασισμένο στα δείγματα εκπαίδευσης που του παρέχουμε.

Σε κάθε στοιβάδα του Neural Network, προστίθεται ένας bias νευρώνας, ο οποίος αποθηκεύει απλά την τιμή 1. Ο bias νευρώνας καθιστά δυνατή τη μετακίνηση της activation function αριστερά, δεξιά, πάνω ή κάτω στο γράφημα αριθμών. Χωρίς bias νευρώνα, κάθε νευρώνας παίρνει το input και το πολλαπλασιάζει με το weight του, χωρίς να προσθέτει τίποτα στην εξίσωση ενεργοποίησης. Αυτό σημαίνει πως, για παράδειγμα εάν δεν είναι δυνατόν να εισαγάγουμε μια μηδενική τιμή και να δημιουργήσουμε μια έξοδο δύο. Σε πολλές περιπτώσεις είναι απαραίτητο να μετακινήσετε ολόκληρη την activation function προς τα αριστερά ή προς τα δεξιά, προς τα πάνω ή προς τα κάτω, για να δημιουργήσουμε τις απαιτούμενες output τιμές. Ο bias νευρώνας το καθιστά δυνατό.

Για να κατανοήσουμε καλύτερα τη διαφορά bias και variance, πρέπει πρώτα να παραθέσουμε το concept ενός training set και ενός validation set. Ένα training set είναι ένα

σύνολο παραδειγμάτων που τροφοδοτούμε το Neural Network κατά την εκπαίδευση. Ένα validation set, είναι ένα σύνολο παραδειγμάτων που δεν έχουν τροφοδοτηθεί στο δίκτυο (unseen examples) και τα χρησιμοποιούμε για να δοκιμάσουμε το πως αποδίδει το δίκτυο. Ένα error function υπολογίζει το error, είτε για τα training set είτε για τα validation set. Το error αντικατοπτρίζει το πόσο απέχουν συγκριτικά οι πραγματικές προβλέψεις του δικτύου με τα γνωστά outputs.

Το bias αντικατοπτρίζει πόσο καλά ταιριάζει το μοντέλο στο training set. Ένα υψηλό bias σημαίνει ότι το Neural Network δεν μπορεί να δημιουργήσει σωστές προβλέψεις ακόμη και για τα παραδείγματα στα οποία εκπαιδεύτηκε. Το variance αντικατοπτρίζει πόσο καλά το μοντέλο ταιριάζει σε unseen παραδείγματα στο validation set. Υψηλό bias σημαίνει ότι το Neural Network δεν μπορεί να προβλέψει σωστά για νέα παραδείγματα που δεν έχει ξαναδεί.



Εικόνα 2.6-Bias και Variance διακυμάνσεις

2.1.5 Υπέρ-προσαρμογή και Υπό-προσαρμογή στα Νευρωνικά Δίκτυα (Overfitting And Underfitting In Neural Networks)

Το overfitting συμβαίνει όταν το Neural Network είναι καλό στο να μάθει το training set του, αλλά δεν είναι σε θέση να γενικεύσει τις προβλέψεις του σε επιπλέον, unseen παραδείγματα. Αυτό χαρακτηρίζεται από χαμηλό bias και υψηλό variance. Το underfitting συμβαίνει όταν το Neural Network δεν είναι σε θέση να προβλέψει με ακρίβεια ούτε για το training set, ούτε για το validation set. Αυτό χαρακτηρίζεται από υψηλό bias και υψηλό variance.

Θα παραθέσουμε μερικές βασικές μεθόδους αποφυγής του overfitting.

- **Επανεκπαίδευση του Neural Network.** Τρέχουμε το ίδιο μοντέλο στο ίδιο training set, αλλά με διαφορετικά αρχικά weights και επιλέγοντας το δίκτυο με την καλύτερη απόδοση.

- **Πολλαπλά Neural Networks.** Εκπαίδευση διαφόρων Neural Network μοντέλων παράλληλα, με την ίδια δομή αλλά με διαφορετικά weights, και στρογγυλοποιώντας τα outputs.

- **Early stopping.** Εκπαίδευση του δικτύου, παρακολούθηση του error στο validation set μετά από κάθε επανάληψη και διακοπή της εκπαίδευσης όταν το δίκτυο αρχίζει να κάνει overfit τα δεδομένα**.**

- **Regularization.** Έναν όρος που προσθέτουμε στην εξίσωση συνάρτησης σφάλματος (error function equation), με σκοπό τη μείωση των weights και των biases, την εξομάλυνση των outputs και να κάνει το δίκτυο λιγότερο πιθανό στο overfitting.

- **Συντονισμός του performance ratio.** Παρόμοια με το regularization, αλλά χρησιμοποιώντας μια παράμετρο που καθορίζει κατά πόσον το δίκτυο πρέπει να κανονικοποιηθεί (regularized)

Θα παραθέσουμε επίσης μερικές βασικές μεθόδους αποφυγής του underfitting.

- **Πρόσθεση στοιβάδων νευρώνων ή inputs.** Η προσθήκη στοιβάδων νευρώνων, ή η αύξηση του αριθμού των inputs και των νευρώνων σε κάθε στοιβάδα, μπορεί να δημιουργήσει πιο περίπλοκες προβλέψεις και να βελτιώσει την εφαρμογή του μοντέλου.

- **Πρόσθεση περισσότερων training δειγμάτων ή βελτίωση της ποιότητάς τους.** Όσο περισσότερα δείγματα εκπαίδευσης τροφοδοτείτε στο δίκτυο και όσο καλύτερα αντιπροσωπεύουν το variance στον πραγματικό πληθυσμό τους, τόσο καλύτερη θα είναι η απόδοση του δικτύου.

- **Dropout.** "Σκοτώνοντας" τυχαία ένα ορισμένο ποσοστό νευρώνων σε κάθε επανάληψη της εκπαίδευσης. Αυτό διασφαλίζει ότι ορισμένες πληροφορίες που μαθαίνονται αφαιρούνται τυχαία, μειώνοντας τον κίνδυνο overfitting.

- **Μείωση της regularization παραμέτρου.** Το regularization μπορεί να είναι υπερβολικό. Χρησιμοποιώντας μια παράμετρο απόδοσης κανονικοποίησης (regularization performance parameter), μπορούμε να μάθουμε τον βέλτιστο regularization βαθμό, που μπορεί να βοηθήσει το μοντέλο να ταιριάζει καλύτερα στα δεδομένα.

2.1.6 Υπερπαράμετροι Νευρωνικών Δικτύων (Neural Networks Hyperparameters)

Οι υπερπαράμετροι (hyperparametres) καθορίζουν πώς είναι δομημένο το Neural Network, πώς εκπαιδεύεται και πως λειτουργούν τα διαφορετικά στοιχεία του. Η βελτιστοποίηση των hyperparameters είναι τέχνη. Υπάρχουν διάφοροι τρόποι, από τη χειροκίνητη δοκιμή και το σφάλμα έως τις εξελιγμένες αλγοριθμικές μεθόδους.

Ποια είναι η διαφορά όμως μεταξύ μίας παραμέτρου μοντέλου (model parameter) και μιας υπερπαραμέτρου;

- Ένα model parameter είναι εσωτερικό για την εκπαίδευση του δικτύου μας και χρησιμοποιείται για να κάνει προβλέψεις σε ένα Deep Learning μοντέλο. Ο στόχος της εκπαίδευσης είναι να μάθουμε τις τιμές των παραμέτρων του μοντέλου.

- Ένα hyperparameter είναι μια εξωτερική παράμετρος που ορίζεται από τον χειριστή του Neural Network. Για παράδειγμα, ο αριθμός των επαναλήψεων της εκπαίδευσης, ο αριθμός των hidden layers ή το activation function. Διαφορετικές hyperparameter τιμές μπορούν να έχουν σημαντικό αντίκτυπο στην απόδοση του δικτύου.

Μερικά hyperparameters που σχετίζονται με τη δομή του Neural Network, είναι ο αριθμός των hidden layers, το dropout, το activation function και η αρχικοποίηση των weights. Μερικά hyperparameters που σχετίζονται με τον αλγόριθμο εκπαίδευσης, είναι ο ρυθμός εκμάθησης, το epoch, οι επαναλήψεις και το μέγεθος της παρτίδας, ένας optimizer αλγόριθμος και το training momentum.

Σε ένα Neural Network πείραμα, συνήθως θα δοκιμάσουμε πολλές πιθανές hyperparameter τιμές και θα δούμε τι λειτουργεί καλύτερα. Για να αξιολογήσουμε την επιτυχία διαφορετικών τιμών, επανεκπαιδεύουμε το δίκτυο, χρησιμοποιώντας κάθε hyperparameter σύνολο και δοκιμάζοντάς το έναντι του validation set. Εάν το training set είναι μικρό, μπορούμε να χρησιμοποιήσουμε cross validation διαιρώντας το training set σε πολλές ομάδες, εκπαιδεύοντας το μοντέλο σε κάθε μία από τις ομάδες και στη συνέχεια επικυρώνοντας το στις

άλλες ομάδες. Ακολουθούν οι κοινές μέθοδοι που χρησιμοποιούνται για το συντονισμό των hyperparameters:

1. **Χειροκίνητος συντονισμός των hyperparameters.** Ένας έμπειρος χειριστής, μπορεί προβλέψει τις τιμές των παραμέτρων που θα επιφέρουν μεγάλη ακρίβεια. Αυτό βέβαια απαιτεί πολλές δοκιμές και errors.

2. **Grid search.** Αυτό συνεπάγεται συστηματικό έλεγχο πολλαπλών κάθε hyperparameter τιμών και επανεκπαίδευση του μοντέλου για κάθε συνδυασμό.

3. **Τυχαία αναζήτηση.** Μια ερευνητική μελέτη των Bergsta και Bengio έδειξε ότι η χρήση τυχαίων hyperparameter τιμών είναι στην πραγματικότητα πιο αποτελεσματική από τη χειροκίνητη αναζήτηση ή το grid search.

4. **Βελτιστοποίηση Bayesian.** Μια μέθοδος που προτείνεται από τους Shahriari, η οποία εκπαιδεύει το μοντέλο με διαφορετικές hyperparameters τιμές ξανά και ξανά, και προσπαθεί να παρατηρήσει το σχήμα της συνάρτησης που δημιουργείται από διαφορετικές τιμές παραμέτρων. Στη συνέχεια επεκτείνει αυτήν τη λειτουργία για να προβλέψει τις καλύτερες δυνατές τιμές. Αυτή η μέθοδος παρέχει μεγαλύτερη ακρίβεια από την τυχαία αναζήτηση.

Σε ένα πραγματικό Neural Network έργο, μπορούμε είτε να βελτιστοποιήσουμε χειροκίνητα τις τιμές των hyperparameters χρησιμοποιώντας τεχνικές βελτιστοποίησης στο Deep Learning framework της επιλογής μας, είτε χρησιμοποιώντας ένα από πολλά τρίτα εργαλεία hyperparameter βελτιστοποίησης . Εάν χρησιμοποιήσουμε το Keras, μπορούμε να χρησιμοποιήσουμε τις βιβλιοθήκες: Hyperopt, Kopt και Talos. Εάν χρησιμοποιήσουμε το TensorFlow, μπορούμε να χρησιμοποιήσουμε το GPflowOpt για Bayesian βελτιστοποίηση και εμπορικές λύσεις όπως το Google’s Cloud Machine Learning Engine που παρέχουν πολλαπλές επιλογές βελτιστοποίησης.

2.1.7 Ταξινόμηση με Νευρωνικά Δίκτυα (Classification With Neural Networks)

Τι είναι το Classification στο Machine και Deep Learning; Υπάρχουν πολλοί, πολύ αποτελεσματικοί αλγόριθμοι ταξινόμησης (classification algorithms), τα Neural Networks είναι μόνο ένας από αυτούς. Η μοναδική δύναμη ενός Neural Network είναι η ικανότητά του να δημιουργεί δυναμικά, σύνθετες λειτουργίες πρόβλεψης και να επιλύει προβλήματα ταξινόμησης με τρόπο που μιμείται την ανθρώπινη σκέψη. Για ορισμένα προβλήματα ταξινόμησης, τα Neural Networks μπορούν να παρέχουν βελτιωμένη απόδοση σε σύγκριση με άλλους αλγόριθμους. Ωστόσο, επειδή τα Neural Networks είναι πιο εντατικά υπολογιστικά και πιο περίπλοκα στη δημιουργία τους, μπορεί να είναι υπερβολικά πολλές περιπτώσεις. Για να κατανοήσουμε το classification με Neural Networks ας καλύψουμε κάποιους από τους κοινούς classification αλγορίθμους. Ορισμένοι αλγόριθμοι είναι δυαδικοί (binary), παρέχοντας μια απόφαση ναι / όχι, ενώ άλλοι είναι πολλαπλών τάξεων (multiclass), επιτρέποντάς μας να ταξινομήσουμε ένα input σε διάφορες κατηγορίες.

- **Logistic regression (binary).** Αναλύει ένα σύνολο σημείων δεδομένων (data points set) και βρίσκει το πιο κατάλληλο μοντέλο για να τα περιγράψει. Εύκολο στην εφαρμογή και πολύ αποτελεσματικό για μεταβλητές inputs που είναι πολύ γνωστές και σχετίζονται στενά με το αποτέλεσμα.

- **Decision tree (multiclass).** Ταξινομεί κάνοντας χρήση μιας δομής δέντρου με κανόνες if-then, εκτελώντας τα inputs μέσω μιας σειράς αποφάσεων μέχρι να φτάσει σε μια κατάσταση τερματισμού. Έχει δυνατότητα μοντελοποίησης σύνθετων διαδικασιών λήψης αποφάσεων και είναι πολύ διαισθητικό, αλλά μπορεί εύκολα να κάνει overfit τα δεδομένα.

- **Random forest (multiclass).** Ένα decision trees σύνολο, με αυτόματη επιλογή του δέντρου με την καλύτερη απόδοση. Παρέχει τη δύναμη του decision tree αλγορίθμου χωρίς το πρόβλημα του overfitting.

- **Naive Bayes classifier (multiclass).** Έναν ταξινομητής βάσει πιθανότητας (probability-based classifier). Υπολογίζει την πιθανότητα να υπάρχει κάθε data point σε κάθε κατηγορία στόχου. Απλό στην εφαρμογή και ακριβές για ένα μεγάλο σύνολο προβλημάτων, αλλά ευαίσθητο στο σύνολο των επιλεγμένων κατηγοριών.

- **k-Nearest neighbor (multiclass).** Ταξινομεί κάθε data point αναλύοντας τους πλησιέστερους γείτονές του μεταξύ των παραδειγμάτων εκπαίδευσης. Απλό στην εφαρμογή και στην κατανόηση, αποτελεσματικό για πολλά προβλήματα, ειδικά εκείνα με χαμηλή διάσταση (low dimensionality). Παρέχει χαμηλότερη ακρίβεια σε σύγκριση με supervised αλγόριθμους και είναι υπολογιστικά εντατικός (computationally intensive)

Τα Neural Networks ταξινομούν περνώντας τις input τιμές μέσω μιας σειράς στοιβάδων νευρώνων, τα οποία εκτελούν πολύπλοκους μετασχηματισμούς στα δεδομένα. **Πλεονεκτήματα**: Τα Neural Networks είναι πολύ αποτελεσματικά για high dimensionality προβλήματα ή με πολύπλοκες σχέσεις μεταξύ μεταβλητών. Για παράδειγμα, τα Neural Networks μπορούν να χρησιμοποιηθούν για την ταξινόμηση και την επισήμανση εικόνων, ήχου και βίντεο, για την ανάλυση συναισθημάτων σε κείμενο και για την ταξινόμηση περιστατικών ασφαλείας σε κατηγορίες κινδύνου. **Αδυναμίες**: Τα Neural Networks θεωρητικά είναι πολύπλοκα, δύσκολα στην εφαρμογή τους, απαιτούν προσεκτική ρύθμιση και είναι computationally intensive. Σε ένα real world Machine Learning έργο, πιθανότατα θα πειραματιστούμε με διάφορους classification αλγορίθμους για να δούμε ποιος παρέχει το καλύτερο αποτέλεσμα. Εάν περιορίσουμε τον εαυτό μας σε σύνηθες classifiers και όχι Neural Networks, μπορούμε να χρησιμοποιήσουμε open source libraries όπως το scikit-learn, το οποίο παρέχει έτοιμες υλοποιήσεις δημοφιλών αλγορίθμων και είναι εύκολο να ξεκινήσουμε με αυτές. Εάν πάλι θέλουμε να δοκιμάσουμε Neural Network classification θα πρέπει να χρησιμοποιήσουμε Deep Learning frameworks όπως TensorFlow, Keras και PyTorch. Αυτά τα frameworks είναι πολύ ισχυρά, υποστηρίζοντας τόσο Neural Networks όσο και παραδοσιακούς classifiers όπως Naive Bayers, αλλά έχουν μια πιο απότομη καμπύλη μάθησης.

2.1.8 Χρήση Νευρωνικών Δικτύων για Παλινδρόμηση (Using Neural Networks For Regression)

Για δεκαετίες, τα μοντέλα παλινδρόμησης (regression models) έχουν αποδειχθεί χρήσιμα στη μοντελοποίηση προβλημάτων και στην παροχή προβλέψεων. Αυτή είναι η κλασική συνάρτηση γραμμικής παλινδρόμησης (linear regression):

$$\gamma = \beta_1 + \beta_2 X_2 + \beta_3 X_3 + \cdots + \beta_k X_k + \varepsilon$$

Σε ένα regression model, τα inputs ονομάζονται ανεξάρτητες τιμές (X1..K στην παραπάνω εξίσωση). Το output ονομάζεται εξαρτημένη τιμή (y στην παραπάνω εξίσωση). Υπάρχουν weights που ονομάζονται συντελεστές, τα οποία καθορίζουν πόσο συνεισφέρει κάθε input τιμή στο αποτέλεσμα ή πόσο σημαντική είναι (β1..K στην παραπάνω εξίσωση). Τα Neural Networks μπορούν να μοντελοποιήσουν πολύπλοκα προβλήματα, χρησιμοποιώντας μια μαθησιακή διαδικασία που μιμείται τον ανθρώπινο εγκέφαλο. Μπορούμε να χρησιμοποιήσουμε ένα Neural Network για να εκτελέσουμε ένα regression; Η σύντομη απάντηση είναι ναι. Τα Neural Networks μπορούν να δημιουργήσουν ένα μοντέλο που προσεγγίζει οποιαδήποτε regression λειτουργία. Επιπλέον, τα περισσότερα regression μοντέλα δεν ταιριάζουν απόλυτα στα δεδομένα και τα Neural Networks μπορούν να δημιουργήσουν ένα πιο περίπλοκο μοντέλο που θα παρέχει μεγαλύτερη ακρίβεια.

Τύποι Regression analysis:

- **Linear regression.** Κατάλληλο για εξαρτημένες τιμές που μπορούν να εφαρμοστούν με ευθεία γραμμή

- **Polynomial regression.** Κατάλληλο για εξαρτημένες μεταβλητές που μπορούν να εφαρμοστούν με καμπύλη ή σειρά καμπυλών

- **Logistic regression.** Κατάλληλο για εξαρτημένες μεταβλητές που είναι δυαδικές και επομένως δεν είναι κανονικά κατανεμημένες.

- **Stepwise regression.** Μια αυτοματοποιημένη τεχνική που μπορεί να αντιμετωπίσει το high dimensionality των ανεξάρτητων μεταβλητών.

- **Ridge regression.** Μια regression τεχνική που βοηθά στην πολυγραμμικότητα (multicollinearity), ανεξάρτητες μεταβλητές που σχετίζονται πολύ. Προσθέτει μια προκατάληψη στις εκτιμήσεις παλινδρόμησης, τιμωρώντας τους συντελεστές χρησιμοποιώντας μια παράμετρο συρρίκνωσης.

- **Lasso regression.** Όπως το Ridge regression, συρρικνώνει συντελεστές για την επίλυση του multicollinearity, ωστόσο, συρρικνώνει επίσης τις απόλυτες τιμές, που σημαίνει ότι ορισμένοι από τους συντελεστές μπορούν να γίνουν μηδέν. Αυτό εκτελεί "επιλογή χαρακτηριστικών" (feature selection), αφαιρώντας μερικές μεταβλητές από την εξίσωση.

- **ElasticNet regression.** Συνδυάζει το Ridge regression και Lasso regression και εκπαιδεύεται με κανονικοποίηση L1 και L2, ανταλλάσσοντας μεταξύ των δύο τεχνικών.

Τα Neural Networks είναι μια πολύ πιο περίπλοκη μαθηματική δομή από τα regression μοντέλα, αλλά μπορούν να μειωθούν σε regression εξισώσεις. Ουσιαστικά, οποιαδήποτε regression εξίσωση, μπορεί να μοντελοποιηθεί από ένα Neural Network. Για παράδειγμα, αυτό το πολύ απλό Neural Network, το οποίο τους παίρνει αρκετά inputs, τα πολλαπλασιάζει με weights και τα περνά μέσω μιας λειτουργίας βήματος (step function), ισοδυναμεί με ένα logistic regression. Ένα ελαφρώς πιο σύνθετο Neural Network μπορεί να κατασκευαστεί για να μοντελοποιήσει μια ταξινόμηση παλινδρόμησης πολλαπλών τάξεων (multiclass regression classification), χρησιμοποιώντας τη Softmax activation function για να δημιουργήσει πιθανότητες για κάθε τάξη, η οποία μπορεί να ομαλοποιηθεί έως και το 1.

Τα Neural Networks μπορούν να χρησιμοποιηθούν για τη δημιουργία regression μοντέλων. Αλλά αξίζει να τα χρησιμοποιήσουμε για αυτό το σκοπό; Η απάντηση εξαρτάται από τη διαίσθησή μας σχετικά με την αποτελεσματικότητα της regression λειτουργίας. Για ορισμένα σύνολα δεδομένων και προβλήματα, οι regression λειτουργίες μπορούν να παρέχουν

πολύ ακριβείς απαντήσεις. Σε αυτές τις περιπτώσεις, πιθανώς δεν απαιτείται ένα Neural Network. Εάν τα δεδομένα είναι πολύπλοκα και η regression συνάρτηση δεν μπορεί να τα μοντελοποιήσει με ακρίβεια, ένα Neural Network μπορεί να δημιουργήσει μια πιο πολύπλοκη μαθηματική δομή που θα αντιπροσωπεύει με ακρίβεια τα δεδομένα. Σε αυτές τις πιο περίπλοκες περιπτώσεις, τα Neural Networks μπορούν να έχουν πολύ περισσότερη προγνωστική ισχύ και μπορούν να αποδώσουν πολύ καλύτερα από μια regression λειτουργία.

Για να εκτελέσουμε μια παραδοσιακή regression συνάρτηση, θα χρησιμοποιούσαμε συνήθως R ή άλλη βιβλιοθήκη μαθηματικών ή στατιστικής. Για να εκτελέσουμε ένα Neural Network ισοδύναμο με ένα regression μοντέλο, θα πρέπει να χρησιμοποιήσουμε Deep Learning frameworks, όπως το TensorFlow, το Keras ή το PyTorch, στα οποία είναι πιο δύσκολο να τελειοποιηθούμε. Ενώ τα Neural Networks έχουν τα γενικά τους μελανά σημεία και είναι θεωρητικά πιο περίπλοκα, παρέχουν ασύγκριτη ισχύ πρόβλεψης συγκριτικά με τα πιο εξελιγμένα regression μοντέλα. Οι regression εξισώσεις είναι περιορισμένες και δεν μπορούν να χωρέσουν απόλυτα σε όλα τα αναμενόμενα σύνολα δεδομένων και όσο πιο περίπλοκο είναι το σενάριό μας, τόσο περισσότερο θα επωφεληθούμε από την είσοδο στον κόσμο του Deep Learning.

2.2 Συνελικτικά Νευρωνικά Δίκτυα (Convolutional Neural Networks (CNN))

Καλύψαμε την παραδοσιακή ή “plain vanilla” αρχιτεκτονική των Artificial Neural Networks στην προηγούμενη ενότητα. Multilayer Perceptron και κατανόηση του Backpropagation. Πάνω σε αυτήν τη βασική δομή, οι ερευνητές έχουν προτείνει αρκετές προηγμένες αρχιτεκτονικές. Παρακάτω καλύπτουμε την CNN αρχιτεκτονική που αναπτύσσεται ευρέως και βοηθά στην παροχή απαντήσεων σε ερωτήσεις που είναι δύσκολο να επιλυθούν με μια παραδοσιακή Neural Network δομή.

Τα Convolutional Neural Networks ή αλλιώς CNN, έχουν αποδειχθεί πολύ καλά στην επεξεργασία δεδομένων που είναι στενά συνδεδεμένα μεταξύ τους. Ένα CNN χρησιμοποιεί μια τρισδιάστατη δομή, με τρία εξειδικευμένα Neural Networks που αναλύουν τα κόκκινα, πράσινα και μπλε στρώματα μιας έγχρωμης εικόνας. Το CNN σαρώνει σε μια εικόνα μία περιοχή κάθε φορά, προσδιορίζει και εξάγει σημαντικά χαρακτηριστικά και τα χρησιμοποιεί για την ταξινόμηση (classification) της εικόνας. Τα CNN χρησιμοποιούνται κυρίως για το

Computer Vision, για εργασίες τροφοδοσίας όπως Image Classification, Face Recognition, Object Detection και Identification και Image Processing σε ρομπότ και αυτόνομα οχήματα. Χρησιμοποιούνται επίσης για video analysis και classification, σημασιολογική ανάλυση (semantic parsing), αυτόματη δημιουργία υπότιτλων (automatic caption generation), ανάκτηση λίστας αναζήτησης (search query retrieval), ταξινόμηση προτάσεων (sentence classification) και άλλα. Ενώ το τυπικό CNN χρησιμοποιεί 2-διαστάσεων ή 3-διαστάσεων νευρωνικές στοιβάδες για την ανάλυση εικόνων με 2 ή 3 έγχρωμα κανάλια, τα CNN με μονοδιάστατες στοιβάδες είναι επίσης πολύ χρήσιμα. Ένα 1D CNN μπορεί να αντλήσει σημαντικά χαρακτηριστικά από μικρά τμήματα ενός dataset συνόλου όταν η θέση κάθε τμήματος δεν είναι τόσο σημαντική. Για παράδειγμα, δεδομένα αισθητήρων, ηχητικά σήματα και επεξεργασία φυσικής γλώσσας.

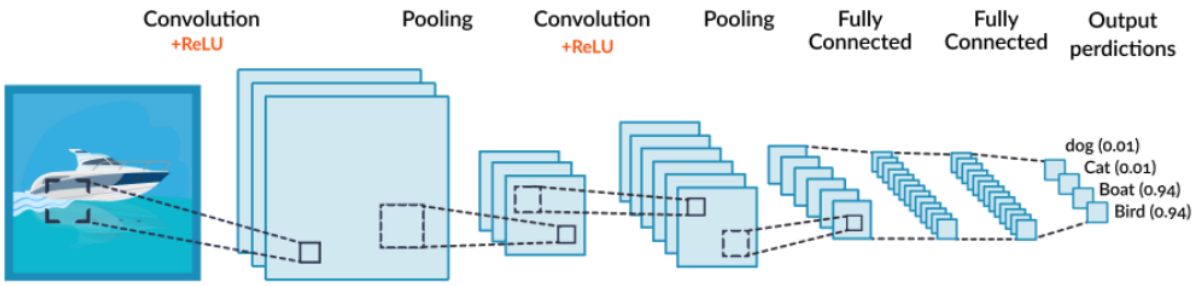
Η CNN αρχιτεκτονική καθιστά δυνατό το Object και Face Detection σε εικόνες χρησιμοποιώντας σύνολα δεδομένων συγκριτικής αξιολόγησης (industry benchmark dataset) με ακρίβεια έως και 95%, μεγαλύτερη από τις ανθρώπινες ικανότητες με ακρίβεια 94%. Ακόμα κι έτσι όμως, τα CNN έχουν τους περιορισμούς τους. Απαιτείται υψηλή επεξεργαστική ισχύς, διότι τα μοντέλα συνήθως εκπαιδεύονται σε μηχανήματα υψηλού κόστους με εξειδικευμένες μονάδες επεξεργασίας γραφικών (GPU). Τέλος ένα CNN μπορεί να αποτύχει όταν οι εικόνες περιστρέφονται ή γέρνουν ή όταν μια εικόνα έχει τα χαρακτηριστικά του επιθυμητού αντικειμένου, αλλά όχι στη σωστή σειρά ή θέση. Για παράδειγμα, ένα πρόσωπο με τη μύτη στη θέση του στόματος και το στόμα στη θέση της μύτης. Μια νέα αρχιτεκτονική που ονομάζεται CAPSNet προέκυψε για την αντιμετώπιση αυτού του περιορισμού.

2.2.1 CNN Αρχιτεκτονική (CNN Architecture)

Ένα απλό vanilla Neural Network, στο οποίο όλοι οι νευρώνες σε μια στοιβάδα επικοινωνούν με όλους τους νευρώνες στην επόμενη στοιβάδα ονομάζεται "πλήρως συνδεδεμένο" και είναι αναποτελεσματικό όσον αφορά την ανάλυση μεγάλων εικόνων και βίντεο. Για μια εικόνα μέσου μεγέθους με εκατοντάδες pixels και τρία κανάλια χρώματος (κόκκινο, πράσινο, μπλε), ο αριθμός των παραμέτρων που χρησιμοποιεί ένα παραδοσιακό Neural Network θα εκατομμύρια, κάτι που μπορεί να οδηγήσει σε overfitting. Για να περιορίσει τον αριθμό των παραμέτρων και να εστιάσει το Neural Network σε σημαντικά μέρη

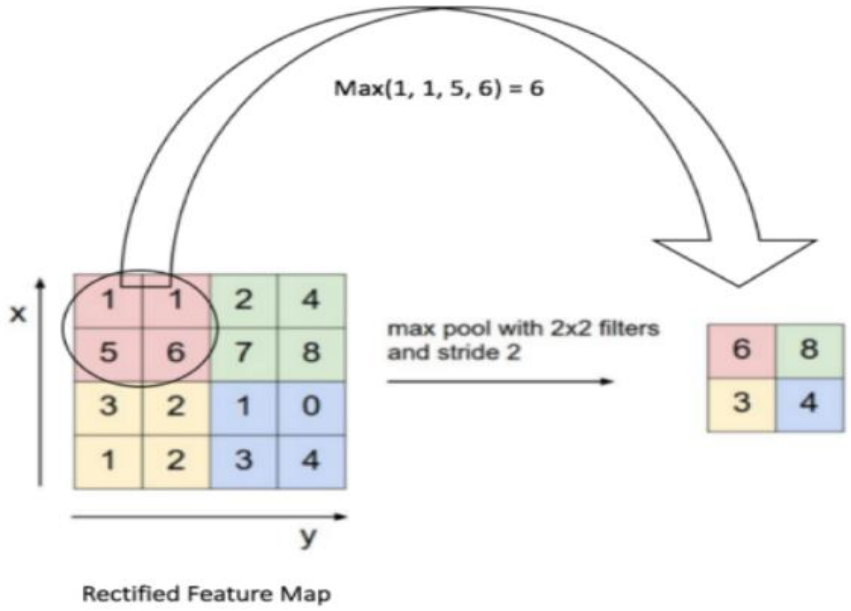
της εικόνας, ένα CNN χρησιμοποιεί μια τρισδιάστατη δομή στην οποία κάθε σύνολο νευρώνων αναλύει μια μικρή περιοχή ή "χαρακτηριστικό" (feature) της εικόνας. Αντί όλοι οι νευρώνες να έχουν να περάσουν τη λήψη των αποφάσεών τους στην επόμενη νευρωνική στοιβάδα, κάθε ομάδα νευρώνων ειδικεύεται στην αναγνώριση ενός μέρους της εικόνας. Για παράδειγμα, της μύτης, του αριστερού αυτιού, του στόματος ή των μαλλιών. Το τελικό output είναι ένα probability scores διάνυσμα, που αντιπροσωπεύει το πόσο πιθανό είναι ένα από τα χαρακτηριστικά, να είναι κομμάτι μιας κατηγορίας (class).

Ένα CNN λειτουργεί σε τρία στάδια. Σε ένα απλό classification παράδειγμα, το πρώτο στάδιο είναι μια συνέλιξη (convolution), στην οποία η εικόνα "σαρώνεται" μερικά pixels κάθε φορά, και δημιουργείται ένας χάρτης χαρακτηριστικών (feature map) με πιθανότητες ότι κάθε χαρακτηριστικό ανήκει στο απαιτούμενο class. Το δεύτερο στάδιο είναι ομαδοποίηση (pooling), η οποία μειώνει το dimensionality κάθε feature διατηρώντας παράλληλα τις πιο σημαντικές πληροφορίες του. Το pooling στάδιο δημιουργεί μια "περίληψη" των πιο σημαντικών χαρακτηριστικών της εικόνας.



Εικόνα 2.7-CNN λειτουργία

Τα περισσότερα CNN χρησιμοποιούν τη μέγιστη συγκέντρωση (max pooling), στην οποία η υψηλότερη τιμή λαμβάνεται από κάθε pixel area που σαρώνεται από το CNN, όπως φαίνεται και παρακάτω στην Εικόνα 2.8.



Εικόνα 2.85-CNN max pooling

Ένα CNN μπορεί να πραγματοποιήσει αρκετούς convolution rounds και μετά να κάνει pooling. Για παράδειγμα, στον πρώτο γύρο, μια εικόνα μπορεί να χωριστεί σε αντικείμενα, όπως μια βάρκα, ένα άτομο, ένα αεροσκάφος ή γρασίδι. Στον δεύτερο γύρο, το CNN μπορεί να αναγνωρίσει features σε κάθε αντικείμενο όπως, πρόσωπο, κορμό, χέρια, πόδια. Σε έναν τρίτο γύρο, το CNN θα μπορούσε να πάει βαθύτερα και να αναλύσει τα χαρακτηριστικά μέσα στο πρόσωπο, κ.λπ. Τέλος, όταν τα features βρίσκονται στον σωστό βαθμό ανάλυσης, το CNN εισέρχεται στο τρίτο στάδιο, το οποίο είναι ένα πλήρως συνδεδεμένο νευρωνικό δίκτυο (Fully Connected Neural Network) που αναλύει τα τελικά probabilities και αποφασίζει σε ποια κατηγορία ανήκει η εικόνα. Το τελευταίο βήμα μπορεί επίσης να χρησιμοποιηθεί και για άλλες εργασίες, όπως η δημιουργία κειμένου, μια συνηθισμένη χρήση των CNN, όπως η αυτόματη δημιουργία λεζάντας για εικόνες.

2.2.2 Πλήρως Συνδεδεμένο CNN (Fully Connected CNN)

Οι Fully Connected στοιβάδες αποτελούν βασικό συστατικό των Convolutional Neural Networks (CNN), τα οποία έχουν αποδειχθεί πολύ επιτυχημένα για Image Recognition και Classification στο Computer Vision. Η CNN διαδικασία ξεκινά με convolution και pooling, χωρίζοντας την εικόνα σε features και αναλύοντας τα ανεξάρτητα. Το αποτέλεσμα αυτής της διαδικασίας τροφοδοτείται σε ένα Fully Connected Neural Network που οδηγεί την τελική classification απόφαση.

Οι Fully Connected στοιβάδες σε ένα CNN δεν πρέπει να συγχέονται με Fully Connected Neural Networks, την κλασική Neural Network αρχιτεκτονική δηλαδή, στην οποία όλοι οι νευρώνες συνδέονται με όλους τους νευρώνες της επόμενης στοιβάδας. Τα CNN επιτρέπουν το Deep Learning για Computer Vision.

Η CNN αρχιτεκτονική αποτελείται από πολλά είδη στοιβάδων (layers).

- **Convolutional layer.** Ένα "φίλτρο" περνά πάνω από την εικόνα, σαρώνοντας μερικά pixel κάθε φορά και δημιουργώντας ένα feature map που προβλέπει το class στο οποίο ανήκει κάθε feature.

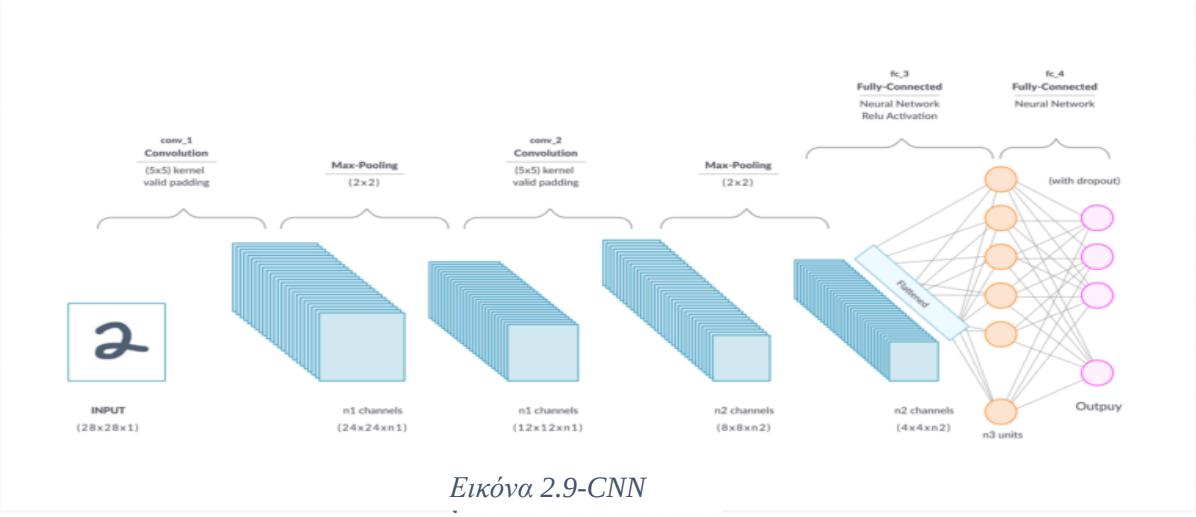
- **Pooling layer.** Μειώνει την ποσότητα πληροφοριών σε κάθε feature που λαμβάνεται στο convolutional layer, διατηρώντας παράλληλα τις πιο σημαντικές πληροφορίες. Συνήθως υπάρχουν αρκετά convolutional και pooling rounds.

- **Fully connected input layer.** παίρνει το output των προηγούμενων στοιβάδων, τα "ισοπεδώνει" και τα μετατρέπει σε ένα μονό διάνυσμα που μπορεί να αποτελέσει input για την επόμενη στοιβάδα.

- **The first fully connected layer.** Παίρνει τα inputs από το feature analysis και εφαρμόζει weights για να προβλέψει το σωστό label.

- **Fully connected output layer.** Δίνει τα τελικά probabilities για κάθε label

Ακολουθεί ένα παράδειγμα (Εικόνα 2.9) που δείχνει τις στοιβάδες που απαιτούνται για την επεξεργασία μιας εικόνας ενός γραπτού ψηφίου, με τον αριθμό των pixels να υποβάλλονται σε επεξεργασία σε κάθε στοιβάδα. Εδώ έχουμε μία πολύ απλή εικόνα.

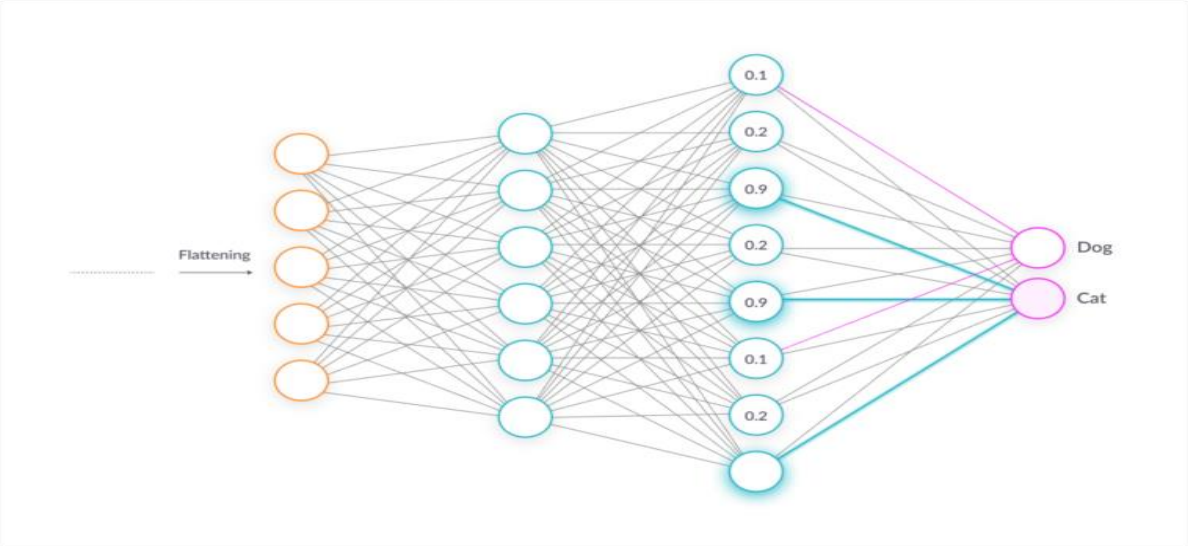


Εικόνα 2.9-CNN

Μεγαλύτερες και πιο περίπλοκες εικόνες θα απαιτούσαν περισσότερες convolutional και pooling στοιβάδες.

Ο στόχος ενός fully connected layer είναι να λάβει τα αποτελέσματα της convolutional και pooling διαδικασίας και να τα χρησιμοποιήσει για να ταξινομήσει την εικόνα σε ένα label. Το output των convolution και pooling “ισοπεδώνεται” σε έναν μόνο vector of values, καθένα από τα οποία αντιπροσωπεύει μια πιθανότητα ότι ένα συγκεκριμένο feature ανήκει σε ένα label. Για παράδειγμα, εάν η εικόνα μιας γάτας, παραθέτει features όπως μουστάκια ή γούνα θα πρέπει να έχουν μεγάλες πιθανότητες για το label “γάτα”. Η παρακάτω εικόνα (Εικόνα 2.10) δείχνει πως οι input τιμές, ρέουν στην πρώτη στοιβάδα νευρώνων. Πολλαπλασιάζονται με weights και περνούν από ένα activation Function (συνήθως ReLu), όπως ακριβώς σε ένα κλασικό Artificial Neural Network. Στη συνέχεια περνούν προς τα εμπρός, στην output στοιβάδα, στη οποία κάθε νευρώνας αντιπροσωπεύει ένα classification label.

Το fully connected τμήμα του δικτύου CNN περνά από τη δική του διαδικασία backpropagation για να προσδιορίσει τα πιο ακριβή weights. Κάθε νευρώνας λαμβάνει weights που δίνουν προτεραιότητα στο πιο κατάλληλο label. Τέλος, οι νευρώνες "ψηφίζουν" σε κάθε ένα από τα labels, και ο νικητής αυτής της ψηφοφορίας είναι η απόφαση ταξινόμησης (classification decision).



Εικόνα 6.10-Classification decision διαδικασία

2.2.3 Μοντέλα Οικογενείας R-CNN (R-CNN Models Family)

Το Computer Vision είναι ένα διεπιστημονικό πεδίο που έχει κερδίσει τεράστια έλξη τα τελευταία χρόνια ιδιαίτερα μετά από τα CNN. Ένα αναπόσπαστο μέρος του Computer Vision είναι το Object Detection. Το Object Detectionστην όπως είπαμε, συνδράμει στην εκτίμηση θέσης, ανίχνευση οχήματος, επιτήρηση κ.λπ. Η διαφορά μεταξύ Object Detection αλγορίθμων και Classification αλγορίθμων είναι ότι στους Detection αλγόριθμους, προσπαθούμε να σχεδιάσουμε ένα bounding box γύρω από το αντικείμενο ενδιαφέροντος για να το εντοπίσουμε μέσα στην εικόνα. Επίσης, ενδέχεται να μην σχεδιάζετε απαραίτητα μόνο ένα bounding box σε μια Object Detection περίπτωση. Θα μπορούσαν να υπάρχουν πολλά bounding boxes που αντιπροσωπεύουν διαφορετικά αντικείμενα ενδιαφέροντος εντός της εικόνας και πιθανότατα δεν θα γνωρίζουμε εκ των προτέρων πόσα θα είναι. Ο κύριος λόγος για τον οποίο δεν μπορούμε να αντιμετωπίσουμε αυτό το πρόβλημα δημιουργώντας ένα τυπικό

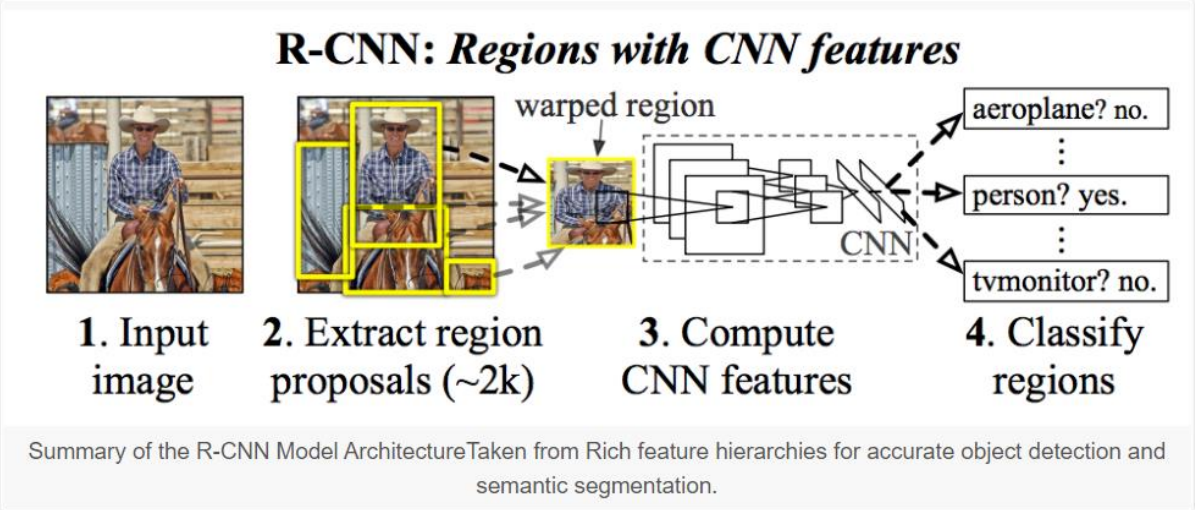
CNN που ακολουθείται από ένα fully connected layer είναι ότι, το μήκος του output layer είναι μεταβλητό και όχι σταθερό. Αυτό συμβαίνει επειδή ο αριθμός των εμφανίσεων των αντικειμένων ενδιαφέροντος είναι δεν σταθερός ή γνωστός. Μια αφελής προσέγγιση για την επίλυση αυτού του προβλήματος θα ήταν να ληφθούν διαφορετικές περιοχές ενδιαφέροντος (Region of Interest- RoI) από την εικόνα και να χρησιμοποιηθεί ένα CNN για να ταξινομηθεί η παρουσία του αντικειμένου εντός αυτής της περιοχής. Το πρόβλημα με αυτήν την προσέγγιση είναι ότι τα αντικείμενα ενδιαφέροντος μπορεί να έχουν διαφορετικές χωρικές τοποθεσίες (spatial locations) εντός της εικόνας και διαφορετικούς λόγους διαστάσεων (aspect ratios). Ως εκ τούτου, θα πρέπει να επιλέξουμε έναν τεράστιο αριθμό RoI's και αυτό θα μπορούσε υπολογιστικά να καταρρεύσει. Επομένως, αλγόριθμοι όπως η R-CNN οικογένεια μοντέλων έχουν αναπτυχθεί για να βρουν αυτά τα περιστατικά και να τα βρουν γρήγορα. Το ακρωνύμιο R-CNN, σημαίνει 'Region-Based Convolutional Neural Networks' και αναπτύχθηκε από τον Ross Girshick. Η οικογένεια αυτή περιλαμβάνει τις τεχνικές των R-CNN, Fast R-CNN και Faster R-CNN οι οποίες σχεδιάστηκαν και επιδείχθηκαν και τους σκοπούς του Object Localization, Object Detection και Object Recognition.

R-CNN

Το R-CNN περιγράφεται για πρώτη φορά το 2014 από τον Ross Girshick, στην εργασία του με τίτλο 'Rich feature hierarchies for accurate object detection and semantic segmentation'. Μπορεί να ήταν μια από τις πρώτες μεγάλες και επιτυχημένες εφαρμογές των CNN στο πρόβλημα Object Localization, Object Detection και Object Segmentation. Το προτεινόμενο μοντέλο R-CNN αποτελείται από τρεις ενότητες.

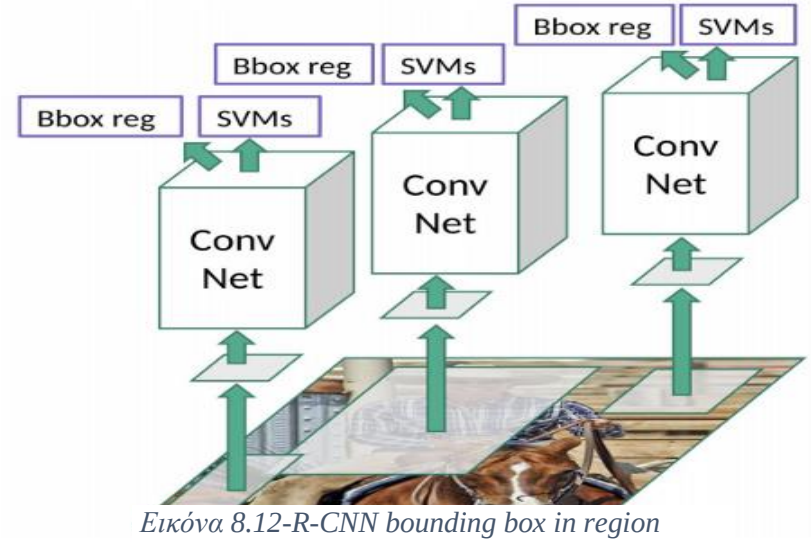
- **Region Proposal (Πρόταση Περιοχής)**
- **Feature Extraction (Εξαγωγή Χαρακτηριστικών)**
- **Classifier (Ταξινομητής)**

Η αρχιτεκτονική του μοντέλου συνοψίζεται στην παρακάτω εικόνα (Εικόνα 2.11).



Εικόνα 2.117-R-CNN Regions

Μια Computer Vision τεχνική χρησιμοποιείται για να προτείνει υποψήφιες περιοχές (Region Proposal) ή πλαίσια οριοθέτησης (bounding boxes) πιθανών αντικειμένων στην εικόνα που ονομάζεται ‘επιλεκτική αναζήτηση’, αν και η ευελιξία του σχεδιασμού επιτρέπει τη χρήση και άλλων region proposal αλγορίθμων. Η εξαγωγή χαρακτηριστικών (Feature Extraction) που χρησιμοποιήθηκε από το μοντέλο, έγινε από το Deep CNN AlexNet. Η έξοδος του CNN ήταν ένας φορέας 4.096 στοιχείων που περιγράφει το περιεχόμενο της εικόνας που τροφοδοτείται σε ένα γραμμικό SVM για ταξινόμηση της ύπαρξης του αντικειμένου μέσα στο υποψήφιο region proposal. Τα SVM είναι εποπτευόμενα μοντέλα μάθησης με συσχετιζόμενους αλγόριθμους μάθησης που αναλύουν δεδομένα για ταξινόμηση και ανάλυση παλινδρόμησης (regression analysis). Ένα SVM εκπαιδεύεται για κάθε γνωστή τάξη. Εκτός από την πρόβλεψη της παρουσίας ενός αντικειμένου εντός των region proposals, ο αλγόριθμος



Εικόνα 8.12-R-CNN bounding box in region

προβλέπει επίσης τέσσερις τιμές που είναι τιμές μετατόπισης (offset values) για την αύξηση της ακρίβειας του bounding box. Για παράδειγμα, δεδομένου ενός region proposal, ο αλγόριθμος θα είχε προβλέψει την παρουσία ενός ατόμου, αλλά το πρόσωπο αυτού του ατόμου μέσα σε αυτό το region proposal θα μπορούσε να είχε κοπεί στα μισά. Επομένως, τα offset values βοηθούν στην προσαρμογή του bounding box στο region proposal.

Είναι μια σχετικά απλή εφαρμογή των CNN στο πρόβλημα του Object Localization και Recognition. Βασικό μειονέκτημα της προσέγγισης είναι ότι είναι αργή, απαιτώντας ένα feature extraction πέρασμα με βάση το CNN σε καθεμία από τις region proposal περιοχές, που δημιουργείται από τον αλγόριθμο. Αυτό είναι πρόβλημα καθώς η εργασία του Ross Girshick περιγράφει μοντέλο που λειτουργεί σε περίπου 2.000 προτεινόμενες περιοχές ανά εικόνα κατά το χρόνο δοκιμής.

Fast R-CNN

Δεδομένης της μεγάλης επιτυχίας του R-CNN, ο Ross Girshick, τότε στη Microsoft Research, πρότεινε μια επέκταση για να αντιμετωπίσει τα ζητήματα ταχύτητας του R-CNN σε μια νέα εργασία του το 2015 με τίτλο ‘Fast R-CNN’. Η εργασία ανοίγει με μια ανασκόπηση των περιορισμών του R-CNN, οι οποίοι μπορούν να συνοψιστούν ως εξής:

- **Η εκπαίδευση είναι ένας αγωγός πολλαπλών σταδίων**

  Περιλαμβάνει την προετοιμασία και λειτουργία τριών ξεχωριστών μοντέλων.

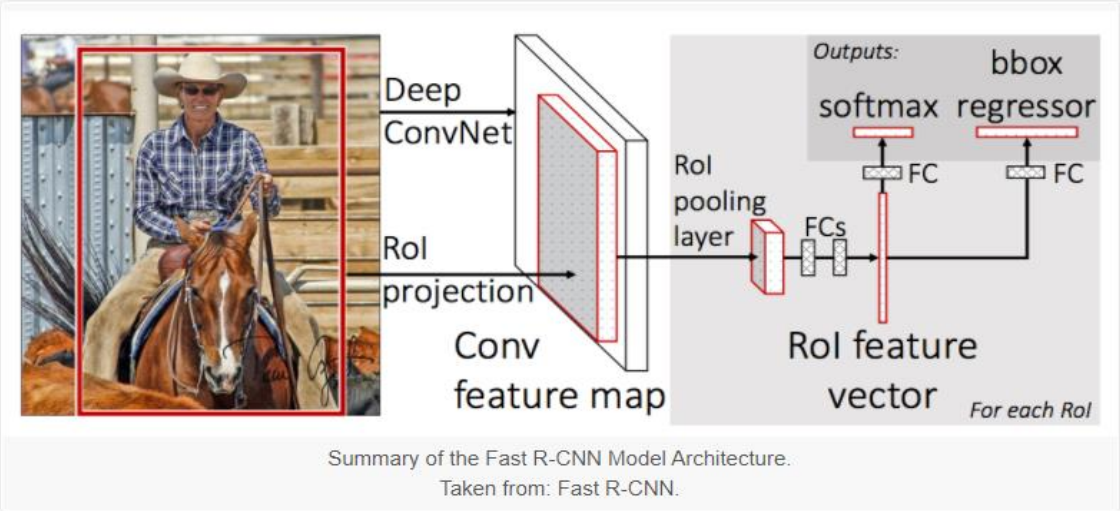
- **Η εκπαίδευση είναι ασύμφορη για μνήμη και χρόνο**

  Η εκπαίδευση ενός Deep CNN σε τόσες πολλές προτάσεις ανά περιοχή είναι πολύ αργή.

- **Το Object Detection είναι αργό**

  Χρησιμοποιώντας ένα Deep CNN, έκανε τις προβλέψεις σε τόσες πολλά Region Proposal αργή.

Μια προηγούμενη εργασία προτάθηκε για να επιταχυνθεί η τεχνική που ονομάζεται SPPnets, σε μια εργασία του 2014 ‘Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition’. Αυτό επιτάχυνε την εξαγωγή χαρακτηριστικών, αλλά ουσιαστικά χρησιμοποίησε έναν τύπο αλγορίθμου προσωρινής αποθήκευσης. Το Fast R-CNN προτείνεται ως ένα μόνο μοντέλο αντί ενός pipeline (αγωγού), για άμεση εκμάθηση και παραγωγή περιοχών και ταξινομήσεων. Η αρχιτεκτονική του μοντέλου παίρνει τη φωτογραφία σαν ένα σύνολο από region proposals ως inputs που περνούν μέσα από ένα Deep CNN.

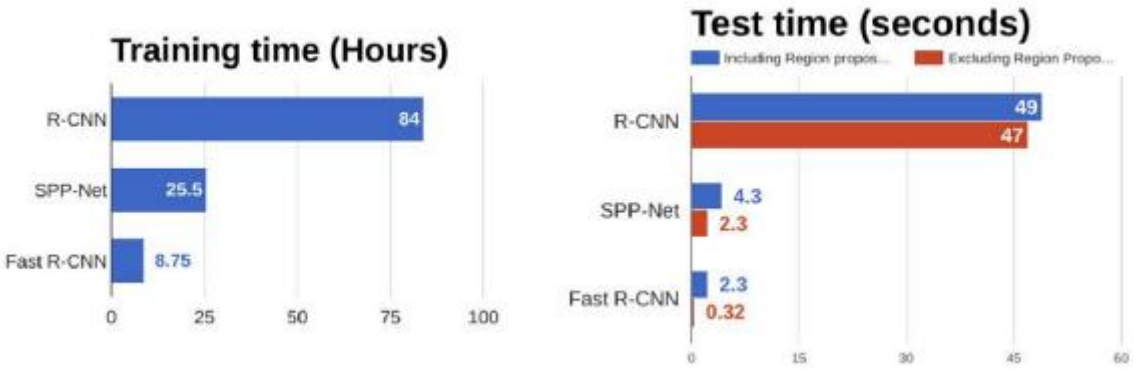


Summary of the Fast R-CNN Model Architecture.
Taken from: Fast R-CNN.

Εικόνα 9.13-Fast R-CNN

Το τέλος του Deep CNN, είναι ένα προσαρμοσμένο επίπεδο (custom layer) που ονομάζεται Region of Interest Pooling Layers, ή RoI το οποίο εξάγει χαρακτηριστικά ειδικά για μία δεδομένη Region Proposal εισαγωγή. Στη συνέχεια, η έξοδος του CNN ερμηνεύεται από ένα πλήρως συνδεδεμένο layer και μετά το μοντέλο διαιρείται σε δύο outputs, μία για την πρόβλεψη κλάσης μέσω ενός softmax layer και μια άλλη με γραμμική έξοδο για τα bounding boxes. Αυτή η διαδικασία επαναλαμβάνεται στη συνέχεια πολλές φορές για κάθε RoI σε μια δεδομένη εικόνα. Η αρχιτεκτονική του μοντέλου συνοψίζεται στην παρακάτω εικόνα (Εικόνα 2.13). Το μοντέλο είναι πολύ πιο γρήγορο στην εκπαιδεύσει και στο να κάνει προβλέψεις, αλλά απαιτεί ακόμη να προταθεί ένα σύνολο Region Proposals μαζί με κάθε είσοδο εικόνας. Ο λόγος για τον οποίο το Fast R-CNN είναι γρηγορότερο από το R-CNN είναι επειδή δεν χρειάζεται να τροφοδοτείτε 2000 region proposals στο CNN κάθε φορά. Αντ 'αυτού, η convolution λειτουργία γίνεται μόνο μία φορά ανά εικόνα και δημιουργείται ένα feature map από αυτήν.

Από τα παραπάνω γραφήματα στην εικόνα 2.14 μπορούμε να συμπεράνουμε ότι το Fast R-CNN είναι πολύ πιο γρήγορο σε εκπαιδεύσεις και δοκιμές σε σχέση με το R-CNN. Όταν εξετάζουμε την απόδοση του Fast R-CNN κατά τη διάρκεια του χρόνου δοκιμής, συμπεριλαμβανομένων των region proposals, επιβραδύνεται σημαντικά ο αλγόριθμος σε σύγκριση με τη μη χρήση region proposals. Επομένως, τα region proposals γίνονται εμπόδια στον Fast R-CNN αλγόριθμο που επηρεάζει την απόδοσή του.



Εικόνα 10.14-R-CNN vs Fast R-CNN Time comparison

Faster R-CNN

Η αρχιτεκτονική του μοντέλου βελτιώθηκε περαιτέρω τόσο ως προς την ταχύτητα εκπαίδευσης όσο και ως προς τον εντοπισμό από τον Shaoqing Ren, στη Microsoft Research σε εργασία του το 2016 με τίτλο 'Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks'. Η αρχιτεκτονική σχεδιάστηκε για να προτείνει και να τελειοποιεί Region Proposals ως μέρος της εκπαιδευτικής διαδικασίας, που αναφέρεται ως Region Proposal Network ή RPN. Αυτές οι περιοχές στη συνέχεια χρησιμοποιούνται σε συνδυασμό με ένα μοντέλο Fast R-CNN, σε ένα σχέδιο. Αυτές οι βελτιώσεις μειώνουν τόσο τον αριθμό των Region Proposal όσο και επιταχύνουν το χρόνο δοκιμής του μοντέλου σε σχεδόν πραγματικό χρόνο με την τότε τελευταία λέξη της απόδοσης. Αν και είναι ένα ενιαίο μοντέλο, η αρχιτεκτονική αποτελείται από δύο ενότητες:

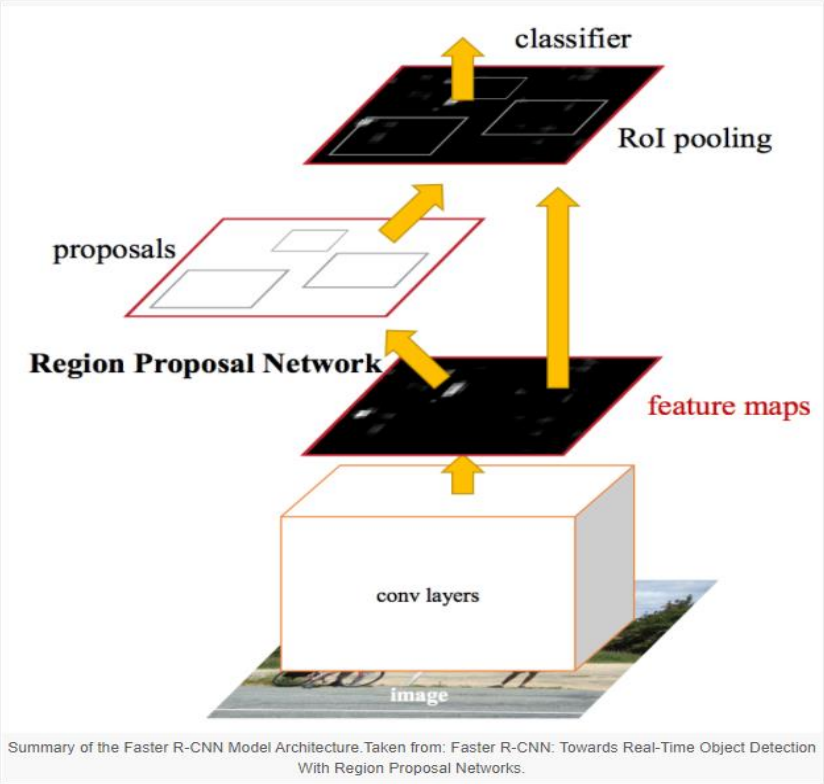
- **Region Proposal Network**

  CNN για το Region Proposal και το είδος του αντικειμένου που πρέπει να ληφθεί υπόψη στην περιοχή.

- **Fast R-CNN**

  CNN για την εξαγωγή χαρακτηριστικών από τα Region Proposals και την έξοδο των bounding boxes και των κατηγοριών.

Και οι δύο ενότητες λειτουργούν στο ίδιο output ενός Deep CNN. Το Region Proposal Network λειτουργεί ως μηχανισμός παρακολούθησης για το Fast R-CNN δίκτυο, ενημερώνοντας το δεύτερο δίκτυο για το που να κοιτάξει ή να δώσει προσοχή. Η αρχιτεκτονική του μοντέλου συνοψίζεται στην εικόνα 2.15.

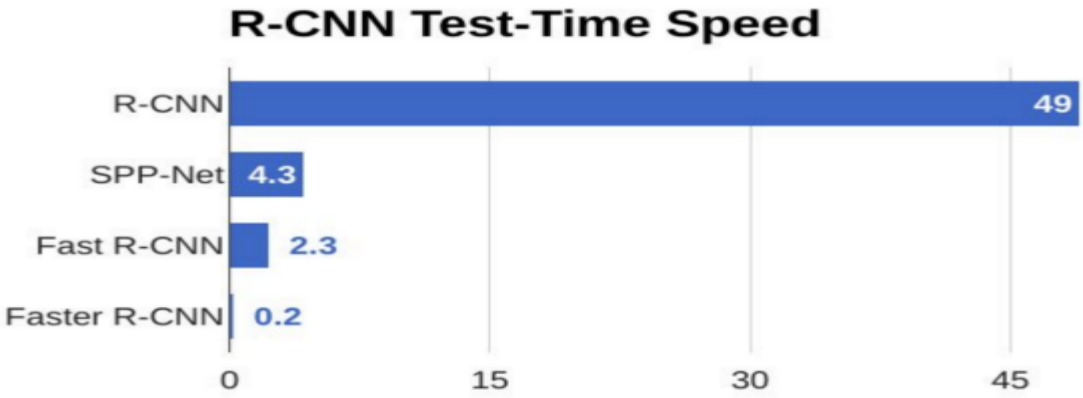


Εικόνα 11.15-Faster R-CNN Αρχιτεκτονική

Το RPN λειτουργεί λαμβάνοντας το output ενός προ-εκπαιδευμένου (pre-trained) Deep CNN, όπως το VGG-16, και περνώντας ένα μικρό δίκτυο μέσω του feature map και εξάγοντας πολλά proposal regions και μια πρόβλεψη κατηγορίας (class prediction) για κάθε μία πρόβλεψη. Τα region proposals είναι bounding boxes, βασισμένα σε προκαθορισμένα σχήματα, σχεδιασμένα για να επιταχύνουν και να βελτιώσουν το region proposal. Η πρόβλεψη της κατηγορίας είναι

δυαδική, υποδεικνύοντας την παρουσία ενός αντικειμένου ή όχι, της λεγόμενης "αντικειμενικότητας" (objectness) της προτεινόμενης περιοχής.

Χρησιμοποιείται μια διαδικασία εναλλακτικής εκπαίδευσης όπου και τα δύο υποδίκτυα εκπαιδεύονται ταυτόχρονα. Αυτό επιτρέπει στις παραμέτρους στον feature detector (ανιχνευτή χαρακτηριστικών) του Deep CNN να προσαρμόζεται ή να ρυθμίζονται και για τις δύο εργασίες ταυτόχρονα. Τη στιγμή της γραφής, η Faster R-CNN αρχιτεκτονική είναι το αποκορύφωμα της οικογένειας των μοντέλων και συνεχίζει να επιτυγχάνει τα σχεδόν βέλτιστα αποτελέσματα σε Object Recognition εργασίες. Μια περαιτέρω επέκταση υποστηρίζει υποστήριξη για Image Segmentation, που περιγράφεται στην εργασία του 2017 'Mask R-CNN'.



Εικόνα2.15

Από το παραπάνω γράφημα στην εικόνα 2.15, μπορούμε να δούμε ότι το Faster R-CNN είναι πολύ πιο γρήγορο από τους προκατόχους του. Επομένως, μπορεί ακόμη και να χρησιμοποιηθεί για την ανίχνευση αντικειμένων σε πραγματικό χρόνο.

2.3 TensorFlow Framework

Το Machine Learning είναι μια περίπλοκη πειθαρχία. Ωστόσο η εφαρμογή Machine Learning μοντέλων είναι λιγότερο τρομακτική και δύσκολη από ότι στο παρελθόν, χάρη στα Machine Learning frameworks όπως το TensorFlow της Google που διευκολύνουν τη διαδικασία απόκτησης δεδομένων, εκπαιδευτικών μοντέλων, προβολής προβλέψεων και βελτίωσης των μελλοντικών αποτελεσμάτων. Πριν από την ανάπτυξη των libraries, ο μηχανισμός κωδικοποίησης για το Machine Learning και Deep Learning ήταν πολύ πιο

περίπλοκος. Αυτό το library παρέχει high level API και δεν απαιτείται σύνθετη κωδικοποίηση για την προετοιμασία ενός Neural Network, τη διαμόρφωση ενός νευρώνα ή τον προγραμματισμό ενός νευρώνα διότι το ίδιο το library ολοκληρώνει όλες αυτές τις εργασίες. Δημιουργήθηκε από την ομάδα της Google Brains, το TensorFlow είναι μια end-to-end open source library για αριθμητικούς υπολογισμούς και μεγάλης κλίμακας Machine Learning. Διαθέτει ένα ολοκληρωμένο, ευέλικτο οικοσύστημα εργαλείων, βιβλιοθηκών και community resources που επιτρέπει στους ερευνητές να χρησιμοποιήσουν την τελευταία λέξη της τεχνολογίας στο Machine Learning και οι προγραμματιστές να κατασκευάζουν και να αναπτύσσουν εύκολα εφαρμογές που υποστηρίζονται από Machine Learning. Το TensorFlow συνδυάζει μια σειρά Machine Learning και Deep Learning αλγορίθμων και μοντέλων (Neural Networking) και τα καθιστά χρήσιμα μέσω μιας κοινής μεταφοράς. Χρησιμοποιεί την Python γλώσσα προγραμματισμού για να παρέχει ένα βολικό front-end API (Application Programming Interface) για την κατασκευή εφαρμογών με το framework, ενώ εκτελεί αυτές τις εφαρμογές σε υψηλής απόδοση C ++ γλώσσα προγραμματισμού. Το TensorFlow μπορεί να εκπαιδεύσει και να εκτελέσει Deep Neural Networks για ταξινόμηση χειρόγραφων ψηφίων (handwritten digit classification), Image Recognition, ενσωματώσεις λέξεων (word embedding), επαναλαμβανόμενα νευρωνικά δίκτυα (Recurrent Neural Networks), sequence to sequence μοντέλα για machine translation, natural language processing και προσομοιώσεις βασισμένες σε PDE (partial differential simulations). Το καλύτερο από όλα, είναι πως το TensorFlow υποστηρίζει την πρόβλεψη παραγωγής σε κλίμακα (production predicting at scale), με τα ίδια μοντέλα που χρησιμοποιούνται για την εκπαίδευση.

Το TensorFlow επιτρέπει στους προγραμματιστές να δημιουργούν γραφήματα ροής δεδομένων (data flow graphs), δομές δηλαδή που περιγράφουν τον τρόπο με τον οποίο τα δεδομένα κινούνται μέσω γραφήματος ή μιας σειράς κόμβων επεξεργασίας (processing nodes). Κάθε node στο graph αντιπροσωπεύει μια μαθηματική λειτουργία και κάθε σύνδεση ή άκρη μεταξύ των nodes είναι ένας πολυδιάστατος πίνακας δεδομένων (multidimensional data array) ή tensor. Το TensorFlow παρέχει όλα αυτά για τον προγραμματιστή μέσω της Python. Η Python είναι εύκολη στην εκμάθηση και στη λειτουργία και παρέχει βολικούς τρόπους για να εκφράσει πως μπορούν να συνδυαστούν οι υψηλού επιπέδου αφαιρέσεις (high level abstractions). Τα nodes και οι tensors στο TensorFlow είναι Python objects και οι TensorFlow εφαρμογές είναι και οι ίδιες ουσιαστικά Python εφαρμογές. Οι πραγματικές μαθηματικές λειτουργίες ωστόσο, δεν εκτελούνται με Python. Οι βιβλιοθήκες μετασχηματισμών (transformation libraries) που είναι διαθέσιμες μέσω του TensorFlow γράφονται ως υψηλής

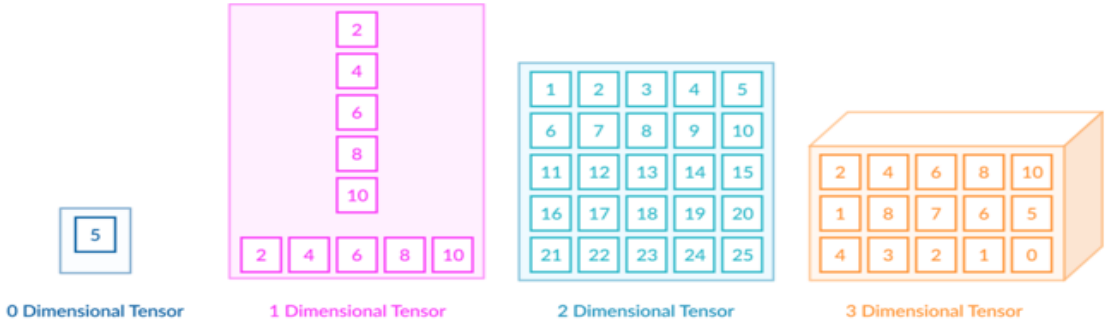
απόδοσης C ++ binaries. Η Python κατευθύνει την κίνηση μεταξύ των κομματιών και παρέχει high level abstractions για να τα συνδέσει.

Οι TensorFlow εφαρμογές μπορούν να εκτελεστούν σχεδόν οπουδήποτε. Ένα local machine, σε ένα cluster στο Cloud, συσκευές iOS και Android, CPU's ή GPU's. Οι GPU's είχαν αρχικά σχεδιαστεί για video games. Στα τέλη του 2010 όμως, οι ερευνητές του πανεπιστημίου του Stanford διαπίστωσαν ότι οι GPU's ήταν επίσης πολύ καλές σε matrix functions και στην άλγεβρα, πράγμα που τις καθιστούσε πολύ γρήγορες στο να κάνουν τέτοιου είδους υπολογισμούς. Το Deep Learning βασίζεται σε πολλούς matrix multiplications. Το TensorFlow είναι πολύ γρήγορο στον υπολογισμό των matrix multiplications επειδή είναι γραμμένο σε C ++. Αν και εφαρμόζεται σε C ++, το TensorFlow μπορεί να προσεγγιστεί και να ελεγχθεί και από άλλες γλώσσες, κυρίως την Python. Εάν χρησιμοποιήσουμε το Cloud της Google, μπορούμε να τρέξουμε το TensorFlow στο προσαρμοσμένο πυρίτιο της Google, TensorFlow Processing Unit (TPU), για περαιτέρω επιτάχυνση. Τα προκείμενα μοντέλα που δημιουργήθηκαν από την TensorFlow ωστόσο, μπορούν να χρησιμοποιηθούν στις περισσότερες συσκευές για την προβολή προβλέψεων.

2.3.1 Σύσταση του TensorFlow

1. **Τανυστές (Tensor)**

   Το όνομα του TensorFlow προέρχεται άμεσα από το βασικό του framework: Tensor. Στο TensorFlow, όλοι οι υπολογισμοί περιλαμβάνουν tensors. Ένας tensor είναι ένας vector ή n-dimensional matrix που αντιπροσωπεύει όλους τους τύπους δεδομένων. Ένας tensor είναι ένας τρόπος να παρουσιάζουμε Deep Learning data. Πρόκειται για έναν multidimensional array, που χρησιμοποιείται για την αποθήκευση δεδομένων για



Εικόνα 2.16-Tensor

πολλά features ενός dataset, όπου κάθε feature αντιπροσωπεύει ένα πρόσθετο dimension. Για παράδειγμα, ένας τρισδιάστατος tensor είναι ένας "κύβος" που αποθηκεύει τιμές κατά μήκος τριών αξόνων. Όλες οι τιμές σε έναν tensor διατηρούν τον ίδιο τύπο δεδομένων με ένα γνωστό ή μερικώς γνωστό σχήμα (shape). Το shape των δεδομένων είναι το dimensionality του matrix ή του array . Ένας tensor μπορεί να προέρχεται από τα input data ή το αποτέλεσμα ενός υπολογισμού. Στο TensorFlow, όλες οι λειτουργίες εκτελούνται μέσα σε ένα graph. Το graph είναι ένα σύνολο υπολογισμών που λαμβάνει χώρα διαδοχικά. Κάθε λειτουργία ονομάζεται op node (node operation) και συνδέονται μεταξύ τους. Το graph περιγράφει τα ops και τις συνδέσεις μεταξύ των nodes. Ωστόσο, δεν εμφανίζει τις τιμές. Το άκρο των nodes είναι ένας tensor, δηλαδή ένας τρόπος για να εμπλουτίσουμε τα ops με δεδομένα. Στο Machine Learning, τα μοντέλα τροφοδοτούνται με μια λίστα αντικειμένων που ονομάζονται feature vectors. Ένα feature vector μπορεί να είναι οποιουδήποτε τύπου δεδομένων. Το feature vector θα είναι συνήθως το main input για να εμπλουτίσουμε έναν tensor. Αυτές οι τιμές θα ρέουν σε έναν op node μέσω του tensor και το αποτέλεσμα αυτής της λειτουργίας/υπολογισμού θα δημιουργήσει έναν νέο tensor ο οποίος με τη σειρά του θα χρησιμοποιηθεί σε μια νέα λειτουργία. Όλες αυτές οι λειτουργίες μπορούν να προβληθούν στο graph.

Στο TensorFlow, όλοι οι υπολογισμοί περνούν από έναν ή περισσότερους tensors. Ο tensor είναι ένα αντικείμενο με τρεις ιδιότητες:

- Unique label (name)
- Dimension (shape)
- Data type (dtype)

Κάθε λειτουργία που κάνουμε με το TensorFlow περιλαμβάνει τον χειρισμό ενός tensor. Υπάρχουν τέσσερις βασικοί tensors που μπορούμε να δημιουργήσουμε:

- Tf.Variable
- Tf.constant
- Tf.placeholder
- Tf.SpaceTensor

2. **Γράφος (Graph)**

Το Machine Learning μπορεί να γίνει πολύπλοκο γρήγορα και τα Deep Learning μοντέλα μπορούν να γίνουν μεγάλα. Για πολλά model graphs, χρειαζόμαστε κατανεμημένη εκπαίδευση (distributed training) για να μπορούν να επαναλαμβάνονται εντός εύλογου χρονικού πλαισίου. Και συνήθως, θέλουμε τα μοντέλα που αναπτύσσουμε να αξιοποιούνται σε πολλές πλατφόρμες. Με την τρέχουσα έκδοση του TensorFlow, γράφουμε κώδικα για να δημιουργήσουμε ένα computation graph και στη συνέχεια να το εκτελέσουμε. Το graph είναι ένα data structure που περιγράφει πλήρως τον υπολογισμό που θέλετε να εκτελέσουμε. Το computational graph στο TensorFlow αντιπροσωπεύει τη ροή των λειτουργιών που συμβαίνουν κατά την εκπαίδευση ενός Deep Learning μοντέλου. Αυτό έχει πολλά πλεονεκτήματα:

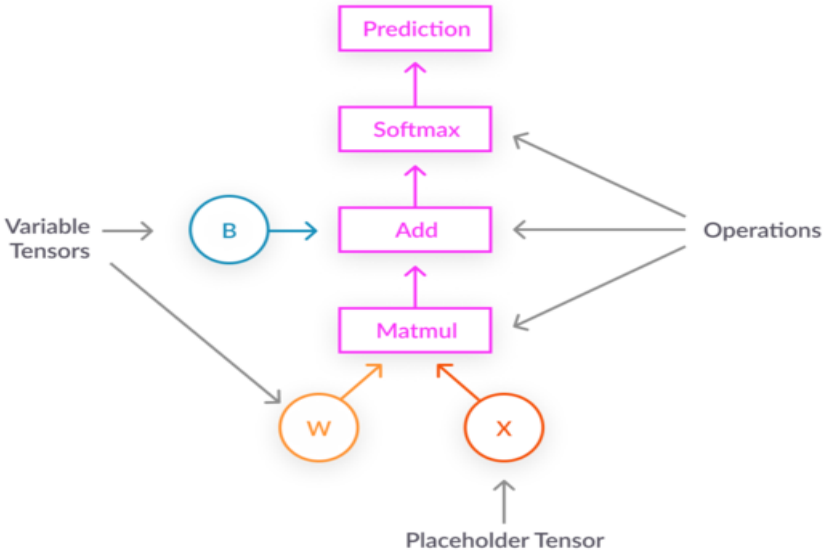
- Είναι φορητό, καθώς το graph μπορεί να εκτελεστεί αμέσως ή να αποθηκευτεί για αργότερη χρήση και μπορεί να εκτελεστεί σε πολλές πλατφόρμες: CPU's, GPU's, TPU's, κινητά, embedded. Επίσης, μπορεί να αξιοποιηθεί στην παραγωγή χωρίς να χρειάζεται να εξαρτάται από οποιονδήποτε από τους κώδικες που δημιούργησαν το graph, παρά μόνο τον χρόνο εκτέλεσης που απαιτείται για την εκτέλεση του.

- Είναι μεταμορφώσιμο και βελτιστοποιήσιμο, καθώς το graph μπορεί να μετατραπεί για να παράγει μια πιο βέλτιστη έκδοση για μια δεδομένη πλατφόρμα. Επίσης, μπορεί να πραγματοποιήσει βελτιστοποιήσεις μνήμης ή υπολογισμού και ανταλλαγές μεταξύ τους. Αυτό είναι χρήσιμο, για παράδειγμα, στην υποστήριξη ταχύτερων κινητών συμπερασμάτων (faster mobile inference) μετά την εκπαίδευση σε μεγαλύτερα μηχανήματα

- Υποστηρίζει το distributed training

Τα high level API του TensorFlow, σε συνδυασμό με computation graphs, επιτρέπουν ένα πλούσιο και ευέλικτο περιβάλλον ανάπτυξης και ισχυρές δυνατότητες παραγωγής

στο ίδιο framework. Για τα CNN μοντέλα, το computational graph μπορεί να είναι αρκετά περίπλοκο. Στην εικόνα 2.17 ακολουθεί ένα παράδειγμα ενός απλού graph. Μπορούμε να οπτικοποιήσουμε το computational graph του μοντέλου μας χρησιμοποιώντας το TensorBoard.



Εικόνα 12.17-CNN computational

3. Σταθερά (Constant)

Χρησιμοποιείται στο TensorFlow για την αποθήκευση σταθερών τιμών που δεν αλλάζουν. Χρησιμοποιείται για κόμβους που απαιτείται να παραμείνουν οι ίδιοι καθ 'όλη τη διάρκεια της εκπαίδευσης του μοντέλου Μια constant δεν λαμβάνει inputs.

4. Σύνοδος (Session)

Ένα session επιτρέπει την εκτέλεση των graphs ή μέρους των graphs. Διανέμει πόρους σε έναν ή περισσότερους υπολογιστές για αυτό και διατηρεί τις πραγματικές τιμές των ενδιάμεσων αποτελεσμάτων και μεταβλητών. Ένα session είναι βασικά η ραχοκοκαλιά ενός TensorFlow προγράμματος. Ένα session ενεργοποιεί το πρόγραμμα για να προετοιμάσει τις σταθερές και να εκτελέσει την επιθυμητή λειτουργία.

5. **Σύμβολο Υποκατάστασης (Placeholder)**

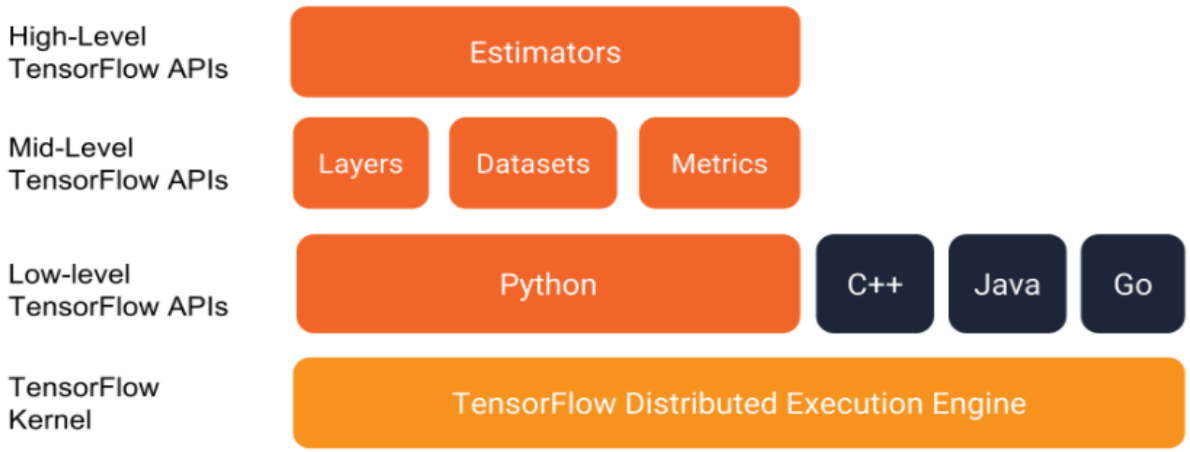
Χρησιμοποιείται για την input τροφοδοσία κατά την εκτέλεση ενός μοντέλου. Ένα placeholder μπορεί να λάβει παραμέτρους και έτσι μπορεί να τροποποιηθεί κατά το χρόνο εκτέλεσης, ενώ εκτελείτε το computational graph.

6. **Μεταβλητή (Variable)**

Χρησιμοποιείται για την τροποποίηση του computational graph, προσθέτοντας παραμέτρους ή nodes που μπορούν να εκπαιδευτούν στο graph.

7. **TensorFlow API levels**

Το TensorFlow μας επιτρέπει να εργαζόμαστε απευθείας με τα tensors για να δημιουργήσουμε ένα Neural Network από το μηδέν. Ωστόσο, αντί να χρησιμοποιούν αυτά τα low level API's που μπορεί να είναι αρκετά περίπλοκα, το TensorFlow συνιστά να εργαζόμαστε με τα high level Estimators API. Αυτό το API επιτρέπει το Object Detection στο TensorFlow, επιτρέποντάς μας να ορίσουμε ένα object, σε higher abstraction level , το οποίο δημιουργεί και εκπαιδεύει Deep Learning structures. Επίσης το TensorFlow παρέχει το Object Detection API, ένα open source framework που μας επιτρέπει να εκτελούμε Image Recognition και Image Segmentation εργασίες.



Εικόνα 2.1813-TensorFlow API levels

2.3.2 Μοντέλα Ανοιχτού Κώδικα στο TensorFlow (TensorFlow Open Source Models)

Η ομάδα του TensorFlow έχει κάνει open source έναν μεγάλο αριθμό μοντέλων. Πολλά από αυτά, ο κωδικός που κυκλοφόρησε περιλαμβάνει όχι μόνο το graph του μοντέλου, αλλά και τα trained weights του μοντέλου. Αυτό σημαίνει ότι μπορούμε να δοκιμάσουμε τέτοια μοντέλα σύμφωνα με τις δικές μας ανάγκες και μπορούμε να συντονίσουμε πολλά από αυτά περαιτέρω, χρησιμοποιώντας μια διαδικασία που ονομάζεται μεταφορά εκμάθησης (transfer learning). Εδώ είναι μερικά από τα μοντέλα που κυκλοφόρησαν πρόσφατα.

- **Object Detection API**

  Εξακολουθεί να αποτελεί βασική Machine Learning πρόκληση για τη δημιουργία ακριβών Machine Learning μοντέλων ικανών για localizing identifying πολλαπλών objects. Το πρόσφατο open source TensorFlow Object Detection API έχει παράγει άριστα αποτελέσματα.

- **Tf-seq2seq**

  Η Google ανακοίνωσε προηγουμένως το Google Neural Machine Translation (GNMT), ένα sequence to sequence μοντέλο (seq2seq) που χρησιμοποιείται τώρα σε συστήματα παραγωγής της Μετάφρασης Google. Το tf-seq2seq είναι ένα open source seq2seq framework στο TensorFlow που το καθιστά εύκολο να πειραματιστούμε με μοντέλα seq2seq και να επιτύχουμε άριστα αποτελέσματα.

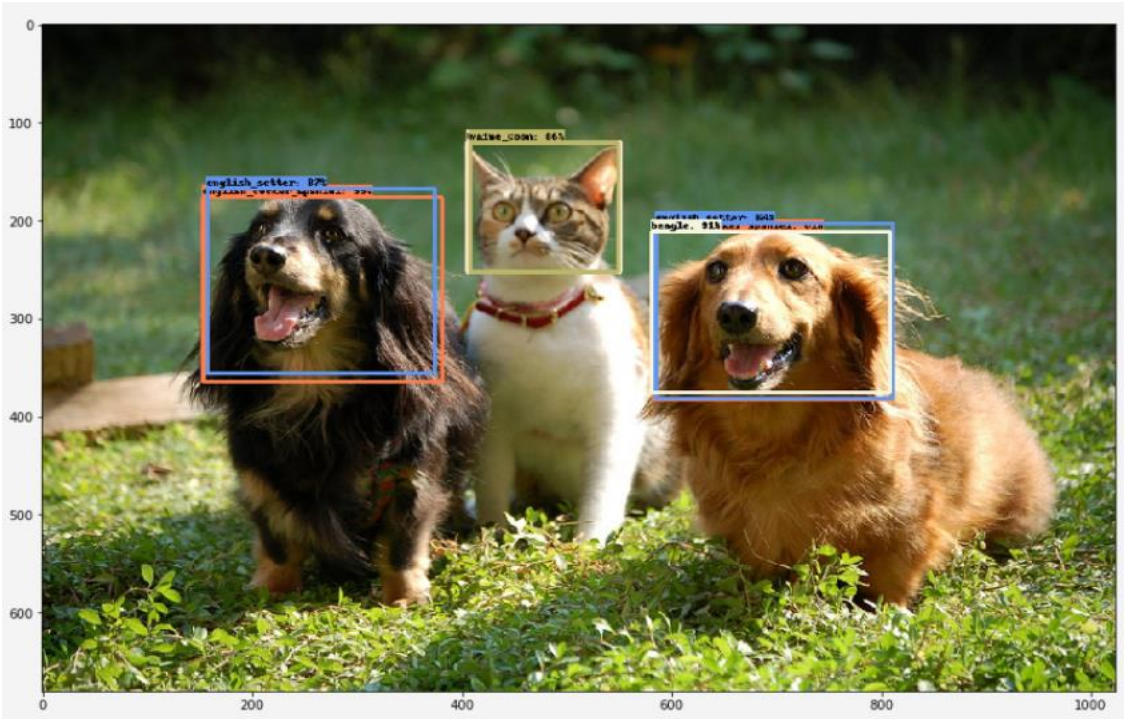
- **ParseySaurus**

  Είναι ένα σύνολο προκαθορισμένων μοντέλων που αντικατοπτρίζουν μια αναβάθμιση στο SyntaxNet. Τα νέα μοντέλα χρησιμοποιούν μια αναπαράσταση εισαγωγής βάσει χαρακτήρων (character based input representation) και είναι πολύ καλύτερα στην πρόβλεψη της σημασίας των νέων λέξεων που βασίζονται τόσο στην ορθογραφία τους όσο και στον τρόπο με τον οποίο χρησιμοποιούνται στο περιεχόμενο.

2.3.3 Μεταφορά Μάθησης (Transfer Learning)

Πολλά από τα TensorFlow μοντέλα περιλαμβάνουν trained weights και παραδείγματα που μας δείχνουν πως μπορούμε να τα χρησιμοποιήσουμε το Transfer Learning, για να εκπαιδεύσουμε τα δικά μας classifications. Αυτό το γίνεται αντλώντας πληροφορίες σχετικά με τα input data μας από το προτελευταίο επίπεδο ενός trained model, το οποίο κωδικοποιεί χρήσιμα abstractions και στη συνέχεια χρησιμοποιούμε αυτές τις πληροφορίες ως input για να εκπαιδεύσουμε το δικό μας πολύ μικρότερο Neural Network για να προβλέψουμε τα δικά μας classes. Λόγω της δύναμης των learned abstractions, η πρόσθετη εκπαίδευση συνήθως δεν απαιτεί μεγάλα data sets. Για παράδειγμα, μπορούμε να χρησιμοποιήσουμε το Transfer Learning με το Inception Image Classifier μοντέλο για να εκπαιδεύσουμε έναν Image Classifier που χρησιμοποιεί τα εξειδικευμένα image data μας.

Ο Object Detection API κώδικας έχει σχεδιαστεί για να υποστηρίζει το Transfer Learning. Στο tensorflow/models repository, υπάρχει ένα παράδειγμα για το πώς μπορούμε να χρησιμοποιήσουμε το Transfer Learning για να κάνουμε bootstrap αυτό το εκπαιδευμένο μοντέλο για να δημιουργήσουμε ένα pet detector, χρησιμοποιώντας ένα (κάπως περιορισμένο) σύνολο δεδομένων παραδειγμάτων φυλών σκύλου και γάτας.



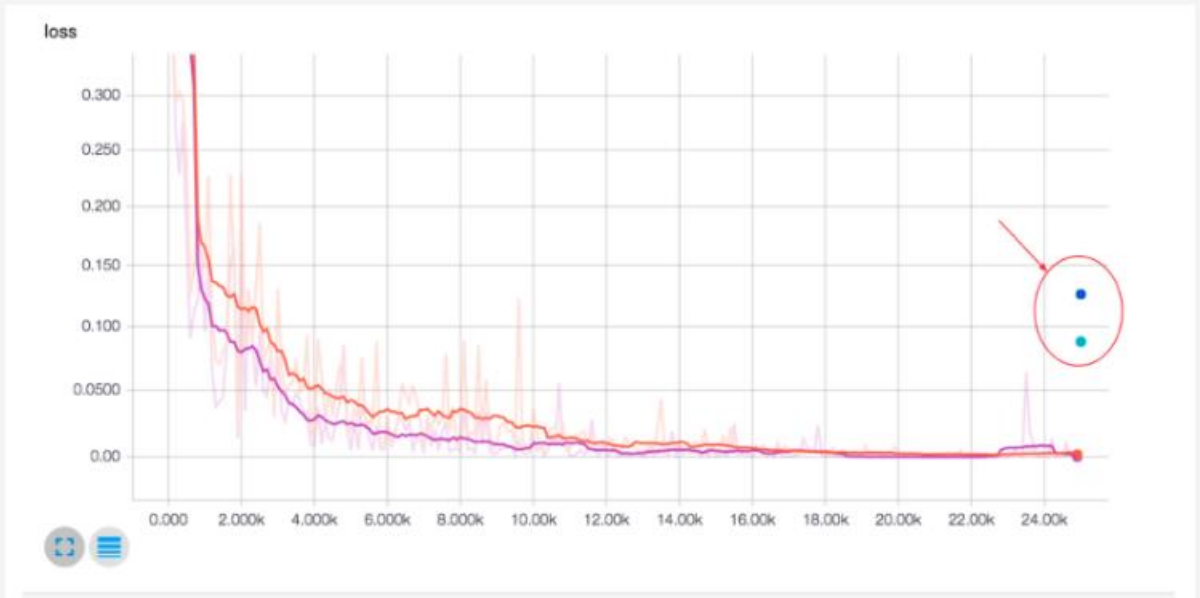
Εικόνα 2.1914-Εκπαίδευση pet detector μέσω Transfer

Στη συνέχεια της εργασία και για την εκπαίδευση του δικού μας classifier, θα κάνουμε bootstrap στο Faster-R-CNN-inception-v2-pets pre trained model μέσω Transfer Learning.

2.3.4 Οικοσύστημα του TensorFlow (TensorFlow Ecosystem)

TensorBoard

Το TensorBoard είναι ένα web application suite για έλεγχο, οπτικοποίηση και κατανόηση των TensorFlow runs και graphs. Μπορούμε να χρησιμοποιήσουμε το TensorBoard για να δούμε τα TensorFlow model graphs και να μεγεθύνουμε τις λεπτομέρειες των υποτμημάτων του graph. Μπορούμε να σχεδιάσουμε μετρήσεις όπως loss και accuracy κατά τη διάρκεια μιας εκπαίδευσης. Μπορούμε επίσης να έχουμε οπτικοποιήσεις ιστογράμματος (histogram visualizations) για το πώς ένας tensor αλλάζει με την πάροδο του χρόνου. Μπορούμε να έχουμε την εμφάνιση πρόσθετων δεδομένων όπως εικόνες και την συλλογή runtime metadata μιας εκτέλεσης όπως συνολική χρήση μνήμης και tensor shapes για nodes και άλλα. Το TensorBoard λειτουργεί διαβάζοντας αρχεία TensorFlow που περιέχουν συνοπτικές πληροφορίες σχετικά με τη διαδικασία εκπαίδευσης. Μπορούμε να ενεργοποιήσουμε αυτά τα αρχεία όταν εκτελούμε TensorFlow εργασίες. Μπορούμε επίσης να χρησιμοποιήσουμε το TensorBoard για να συγκρίνουμε τις εκπαιδεύσεις, να συλλέξουμε στατιστικά χρόνου εκτέλεσης και να δημιουργήσετε histograms.



Εικόνα 2.20-Σύγκριση διαφορετικών optimizers για DNN Classifiers

Ένα ιδιαίτερο χαρακτηριστικό του TensorBoard είναι το embeddings visualizer. Τα embeddings είναι πανταχού παρόντα στο Machine Learning και στο περιεχόμενο του TensorFlow, είναι συχνά φυσικό να βλέπουμε τους tensors ως σημεία στο διάστημα, οπότε σχεδόν κάθε μοντέλο TensorFlow θα δημιουργήσει διάφορες ενσωματώσεις.

Datalab

Τα Jupyter notebooks είναι ένας εύκολος τρόπος για να εξερευνήσουμε διαδραστικά τα δεδομένα μας, να ορίσουμε TensorFlow models και να ξεκινήσουμε εκπαιδεύσεις. Εάν χρησιμοποιούμε Google Cloud Platform εργαλεία ως μέρος της ροής εργασίας μας, ενδεχομένως το Google Cloud Storage ή το BigQuery για τα datasets μας ή το Apache Beam για προεπεξεργασία δεδομένων, τότε το Google Cloud Datalab παρέχει ένα περιβάλλον με βάση το Jupyter με όλα αυτά τα εργαλεία και πολλά ακόμη όπως NumPy, pandas, scikit-learn και Matplotlib, μαζί με το TensorFlow, προεγκατεστημένα και ομαδοποιημένα. Το Datalab είναι open source, οπότε αν θέλουμε να τροποποιήσουμε περαιτέρω το περιβάλλον του notebook μας, είναι εύκολο να το κάνουμε.

Facets

Η δύναμη του Machine Learning προέρχεται από την ικανότητά του να μαθαίνει patterns από μεγάλες ποσότητες δεδομένων, οπότε η κατανόηση των δεδομένων μας είναι κρίσιμη για τη δημιουργία ενός ισχυρού Machine Learning συστήματος. Το Facets είναι ένα open source data visualization εργαλείο που κυκλοφόρησε πρόσφατα και μας βοηθά να κατανοήσουμε τα Machine Learning datasets και να αποκτήσουμε μια αίσθηση του shape και των χαρακτηριστικών κάθε δυνατότητας και να δούμε με μια ματιά πώς αλληλεπιδρούν οι λειτουργίες μεταξύ τους. Για παράδειγμα, μπορούμε να προβάλουμε τα training και test datasets, να συγκρίνουμε τα χαρακτηριστικά κάθε δυνατότητας και να ταξινομήσουμε τις λειτουργίες ανά απόσταση κατανομής (distribution distance).

2.3.5 TensorFlow Εναντίων Ανταγωνισμού

Το TensorFlow ανταγωνίζεται ένα πλήθος άλλων Machine Learning frameworks. Τα PyTorch, CNTK και MXNet είναι τρία μεγάλα frameworks που καλύπτουν πολλές από τις ίδιες ανάγκες.

- **PyTorch.** Εκτός από το ότι έχει κατασκευαστεί με Python, έχει και πολλές άλλες ομοιότητες με το TensorFlow όπως, hardware accelerated components, ένα πολύ διαδραστικό μοντέλο ανάπτυξης και πολλά χρήσιμα components. Το PyTorch είναι γενικά μια καλύτερη επιλογή για γρήγορη ανάπτυξη έργων που πρέπει να τεθούν σε λειτουργία σε σύντομο χρονικό διάστημα, αλλά το TensorFlow κερδίζει για μεγαλύτερα έργα και πιο περίπλοκα workflows.

- **CNTK.** Το Microsoft Cognitive Toolkit, όπως το TensorFlow χρησιμοποιεί ένα graph structure για να περιγράψει το dataflow, αλλά επικεντρώνεται περισσότερο στη δημιουργία Deep Learning Neural Networks. Το CNTK χειρίζεται πολλές Neural Networks εργασίες γρηγορότερα και διαθέτει ένα ευρύτερο σύνολο API (Python, C ++, C #, Java). Ωστόσο, το CNTK δεν είναι τόσο εύκολο στην εκμάθηση όσο το TensorFlow.

- **Apache MXNet**. Υιοθετήθηκε από την Amazon ως το κορυφαίο Deep Learning framework στο AWS, μπορώντας να κλιμακώσει σχεδόν γραμμικά σε πολλές GPU και πολλαπλούς υπολογιστές. Υποστηρίζει επίσης ένα ευρύ φάσμα API γλωσσών όπως, Python, C ++, Scala, R, JavaScript, Julia, Perl και Go, αν και τα native API του δεν είναι τόσο ευχάριστα στη χρήση όσο του TensorFlow.

2.3.6 Σύνοψη

Το TensorFlow είναι η πιο διάσημη Deep Learning βιβλιοθήκη τα τελευταία χρόνια. Ένας επαγγελματίας που χρησιμοποιεί το TensorFlow μπορεί να δημιουργήσει οποιαδήποτε Deep Learning δομή, όπως CNN, RNN ή απλό Artificial Neural Network. Το TensorFlow χρησιμοποιείται κυρίως από ακαδημαϊκούς, νεοσύστατες επιχειρήσεις και μεγάλες εταιρείες. Η Google χρησιμοποιεί το TensorFlow σε σχεδόν όλα τα καθημερινά προϊόντα της, συμπεριλαμβανομένων των Gmail, Photo και Google Search Engine. Η ομάδα του Google Brain, ανέπτυξε το TensorFlow για να καλύψει το κενό μεταξύ ερευνητών και προγραμματιστικών προϊόντων. Το 2015 δημοσιοποίησαν το TensorFlow, αυξάνοντας ραγδαία τη δημοτικότητά του καθημερινά. Σήμερα, το TensorFlow είναι το Deep Learning library με τα περισσότερα repositories στο GitHub. Οι επαγγελματίας χρησιμοποιούν το TensorFlow επειδή είναι εύκολο να αναπτυχθεί σε κλίμακα και να λειτουργεί στο cloud ή σε κινητές συσκευές όπως iOs και Android.

2.4 OpenCV

Το OpenCV (Open Source Computer Vision Library) είναι μια open source Computer Vision και Machine Learning software library ξεκίνησε από μερικούς ενθουσιώδεις προγραμματιστές το 1999 για να ενσωματώσει Image Processing σε μια μεγάλη ποικιλία γλωσσών προγραμματισμού. Το OpenCV δημιουργήθηκε για να παρέχει μια κοινή υποδομή για Computer Vision εφαρμογές και για να επιταχύνει τη χρήση της αντίληψης του υπολογιστή (machine perception) στα εμπορικά προϊόντα. Όντας προϊόν με άδεια BSD, το OpenCV διευκολύνει τις επιχειρήσεις να χρησιμοποιούν και να τροποποιούν τον κώδικα.

Η βιβλιοθήκη διαθέτει περισσότερους από 2500 optimized αλγόριθμους, οι οποίοι περιλαμβάνουν ένα ολοκληρωμένο σύνολο κλασικών και προηγμένων Computer Vision και Machine Learning αλγορίθμων. Αυτοί οι αλγόριθμοι μπορούν να χρησιμοποιηθούν για Face Detection and Recognition, Object Identification, Human Action Classification σε βίντεο, Track Camera Movement, Track Moving Objects, 3D Object Model Extraction, την παραγωγή

3D points cloud από stereo cameras, τη συρραφή εικόνων για την παραγωγή υψηλής ανάλυσης εικόνα ολόκληρης της σκηνής, εύρεση παρόμοιων εικόνων από μια image database, αφαίρεση κόκκινων ματιών από φωτογραφίες που λαμβάνονται χρησιμοποιώντας φλας, παρακολούθηση κινήσεων των ματιών, αναγνώριση τοπίου και δημιουργία δεικτών για επικάλυψη με επαυξημένη πραγματικότητα κ.λπ. Το OpenCV έχει περισσότερα από 47 χιλιάδες άτομα user community και εκτιμάται ότι ο αριθμός των λήψεων υπερβαίνει τα 18 εκατομμύρια. Η βιβλιοθήκη χρησιμοποιείται εκτενώς σε εταιρείες, ερευνητικές ομάδες και από κυβερνητικούς φορείς.

Μαζί με τις καταξιωμένες εταιρείες όπως, Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota που χρησιμοποιούν τη βιβλιοθήκη, υπάρχουν πολλές νεοσύστατες εταιρείες όπως η Applied Minds, η VideoSurf και η Zeitera, που κάνουν εκτενή χρήση του OpenCV. Οι χρήσεις του OpenCV κυμαίνονται από τη συρραφή εικόνων streetview, την ανίχνευση εισβολών σε βίντεο παρακολούθησης στο Ισραήλ, την παρακολούθηση εξοπλισμού ορυχείων στην Κίνα, τη βοήθεια ρομπότ στην πλοήγηση και την παραλαβή αντικειμένων στο Willow Garage, τον εντοπισμό ατυχημάτων πνιγμού σε πισίνες στην Ευρώπη, την εκτέλεση διαδραστικής τέχνης στην Ισπανία και Νέα Υόρκη, έλεγχος αεροδιαδρόμων για σκουπίδια στην Τουρκία, επιθεώρηση ετικετών σε προϊόντα σε εργοστάσια σε όλο τον κόσμο και Rapid Face Detection στην Ιαπωνία.

Διαθέτει C ++, Python, Java και MATLAB interfaces και υποστηρίζει Windows, Linux, Android και Mac OS. Το OpenCV προσανατολίζεται κυρίως σε real time vision applications και εκμεταλλεύεται τις οδηγίες MMX και SSE όταν είναι διαθέσιμες. Τα full features CUDA και OpenCL interfaces αναπτύσσονται ενεργά αυτή τη στιγμή. Υπάρχουν πάνω από 500 αλγόριθμοι και περίπου 10 φορές περισσότερες συναρτήσεις που συνθέτουν ή υποστηρίζουν αυτούς τους αλγόριθμους. Το OpenCV γράφεται natively σε C ++ και έχει ένα templated interface που λειτουργεί απρόσκοπτα με κοντέινερ STL.

2.4.1 Τμήματα Βιβλιοθήκης του OpenCV (OpenCV Library Modules)

Τα βασικά library modules του OpenCV είναι:

- **Core Functionality**
  Οι βασικές λειτουργίες της βιβλιοθήκης OpenCV καλύπτουν τις βασικές δομές δεδομένων, όπως Scalar, Point, Range, κ.λπ. Για την αποθήκευση εικόνων, έχει το multidimensional array Mat.

- **Image Processing**
  Αυτό το module καλύπτει διάφορες λειτουργίες επεξεργασίας εικόνας, όπως image filtering, geometrical image transformation, color space conversion , histograms κ.λπ.

- **Video**
  Αυτό το module καλύπτει video analysis έννοιες όπως motion estimation, background subtraction και object tracking.

- **Video I/O**
  Αυτό το module εξηγεί το video capturing και τους video coders χρησιμοποιώντας τη βιβλιοθήκη OpenCV.

- **Calib3d**
  Αυτό το module περιλαμβάνει αλγόριθμους σχετικά με βασικούς multiple view geometry algorithms, single και stereo camera calibration, object pose estimation, stereo correspondence και 3D elements restoration.

- **Features2d**
  Αυτό το module περιλαμβάνει το σκεπτικό των feature detection και description

- **Objdetect**

  Αυτό το module περιλαμβάνει το detection των objects και instances προκαθορισμένων τάξεων, όπως πρόσωπα, μάτια, φανάρια, άτομα, αυτοκίνητα κ.λπ.

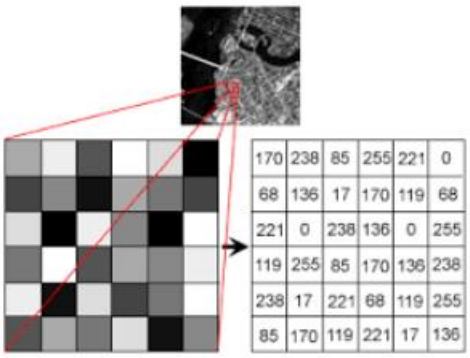
- **Highgui**

  Είναι ένα εύκολο στη χρήση interface, με απλές UI ικανότητες

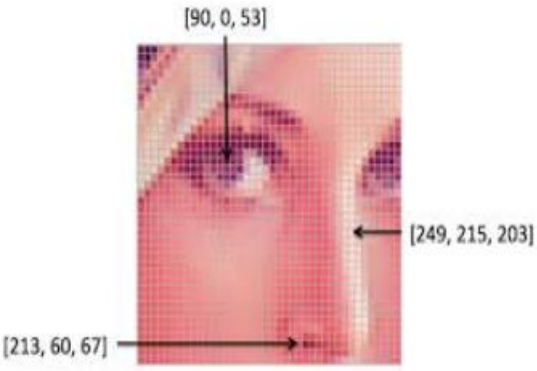
2.4.2 Βασικές Λειτουργίες του OpenCV (Basic OpenCV Functions)

- **Ανάγνωση, Γραφή και Παρουσίαση Εικόνων (Reading, Writing and Displaying Images)**

  Οι υπολογιστές βλέπουν και επεξεργάζονται τα πάντα χρησιμοποιώντας αριθμούς, συμπεριλαμβανομένων εικόνων και κειμένου. Πώς μετατρέπουμε εικόνες σε αριθμούς όμως; Η απάντηση είναι τιμές pixel. Κάθε αριθμός αντιπροσωπεύει την ένταση των pixel στη συγκεκριμένη τοποθεσία. Στην παρακάτω εικόνα (Εικόνα 2.20) μπορούμε να δούμε τις τιμές pixel για μια εικόνα σε γκρι κλίμακα (grayscale) όπου κάθε pixel περιέχει μόνο μία τιμή, δηλαδή την ένταση του μαύρου χρώματος σε αυτήν τη θέση. Σημειώστε ότι οι έγχρωμες εικόνες θα έχουν πολλές τιμές για ένα pixel (Εικόνα 2.21). Αυτές οι τιμές αντιπροσωπεύουν την ένταση των αντίστοιχων καναλιών, για παράδειγμα, κόκκινα, πράσινα και μπλε κανάλια για εικόνες RGB. Η ανάγνωση και η γραφή εικόνων είναι απαραίτητη για κάθε Computer Vision project και η βιβλιοθήκη OpenCV κάνει αυτή τη λειτουργία πολύ πιο εύκολη. Από προεπιλογή, η imread function διαβάζει εικόνες σε μορφή BGR (Μπλε-Πράσινο-Κόκκινο). Μπορούμε να διαβάσουμε εικόνες σε διαφορετικές μορφές χρησιμοποιώντας επιπλέον flags στη λειτουργία imread όπως, cv2.IMREAD_GRAYSCALE, cv2.IMREAD_UNCHANGED



Εικόνα 2.20-Grayscale image pixel values



Εικόνα 2.21-RGB image pixel values

- **Αλλαγή χρωματικών χώρων (Changing color spaces)**

  Το color space είναι ένα πρωτόκολλο για την αναπαράσταση των χρωμάτων με τρόπο που τα καθιστά εύκολα αναπαραγώγιμα. Γνωρίζουμε ότι οι εικόνες κλίμακας του γκρι έχουν τιμές ενός pixel και οι έγχρωμες εικόνες περιέχουν 3 τιμές για κάθε pixel, τις εντάσεις των κόκκινων, πράσινων και μπλε καναλιών. Οι περισσότερες Computer Vision περιπτώσεις χρήσης επεξεργάζονται εικόνες σε μορφή RGB. Ωστόσο, εφαρμογές όπως η συμπίεση βίντεο (video compression) και η ανεξάρτητη αποθήκευση συσκευών (device independent storage), εξαρτώνται σε μεγάλο βαθμό από άλλα color spaces, όπως Hue-Saturation-Value ή HSV. Όπως καταλαβαίνουμε, μια εικόνα RGB αποτελείται από την ένταση των χρωμάτων διαφορετικών καναλιών χρώματος. Δηλαδή η ένταση και οι πληροφορίες χρώματος αναμιγνύονται στο RGB color space, αλλά στο HSV color space οι πληροφορίες χρώματος και έντασης διαχωρίζονται μεταξύ τους. Αυτό καθιστά το HSV color space πιο ανθεκτικό στις αλλαγές φωτισμού. Το OpenCV διαβάζει μια δεδομένη εικόνα σε μορφή BGR από προεπιλογή. Επομένως, θα πρέπει να αλλάξουμε εμείς το color space της εικόνας μας από BGR σε RGB κατά την ανάγνωση εικόνων χρησιμοποιώντας το OpenCV.

- **Αλλαγή Μεγέθους Εικόνων (Resizing Images)**

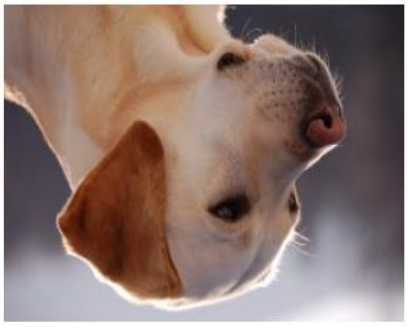
  Τα Machine Learning μοντέλα λειτουργούν με σταθερό μέγεθος εισόδου. Η ίδια ιδέα ισχύει και για τα Computer Vision μοντέλα. Οι εικόνες που χρησιμοποιούμε για την εκπαίδευση του μοντέλου μας πρέπει να έχουν το ίδιο μέγεθος. Τώρα αυτό μπορεί να γίνει προβληματικό εάν δημιουργούμε το δικό

μας σύνολο δεδομένων αποκόπτοντας εικόνες από διάφορες πηγές. Εκεί έρχεται στο προσκήνιο η λειτουργία αλλαγής μεγέθους των εικόνων. Οι εικόνες μπορούν εύκολα να κλιμακωθούν προς τα πάνω και προς τα κάτω χρησιμοποιώντας το OpenCV. Αυτή η λειτουργία είναι χρήσιμη για την εκπαίδευση Deep Learning μοντέλων όταν πρέπει να μετατρέψουμε εικόνες στο input shape του μοντέλου. Διαφορετικές μέθοδοι παρεμβολής (interpolation) και δειγματοληψίας (downsampling) υποστηρίζονται από το OpenCV, οι οποίες μπορούν να χρησιμοποιηθούν από τις ακόλουθες παραμέτρους.

1. **INTER_NEAREST:** Nearest neighbor interpolation
2. **INTER_LINEAR:** Bilinear interpolation
3. **INTER_AREA:** Επαναδειγματοληψία χρησιμοποιώντας τις σχέσεις των pixel area
4. **INTER_CUBIC:** Bicubic interpolation για άνω των 4x4 pixel neighborhood
5. **INTER_LANCZOS4:** Lanczos interpolation για άνω των 8x8 pixel neighborhood

- **Περιστροφή Εικόνας (Image rotation)**

Χρειαζόμαστε μεγάλο αριθμό δεδομένων για να εκπαιδεύσουμε ένα Deep Learning μοντέλο. Οι περισσότεροι Deep Learning αλγόριθμοι εξαρτώνται σε μεγάλο βαθμό από την ποιότητα και την ποσότητα των δεδομένων. Τι γίνεται όμως αν δεν έχουμε αρκετά μεγάλο σύνολο δεδομένων; Δεν είναι δυνατόν όλοι να συλλέγουν και να επισημαίνουν με μη αυτόματο τρόπο εικόνες. Ας υποθέσουμε ότι χτίζουμε ένα Image Classifier μοντέλο για τον προσδιορισμό



Εικόνα 2.22

του ζώου που υπάρχει σε μια εικόνα. Έτσι, και οι δύο εικόνες που εμφανίζονται παρακάτω (Εικόνα 2.22) πρέπει να ταξινομηθούν ως «σκύλος». Αλλά το μοντέλο μπορεί να δυσκολεύεται να ταξινομήσει τη δεύτερη εικόνα ως σκύλο εάν δεν είχε εκπαιδευτεί σε τέτοιες εικόνες. Οπότε τι θα έπρεπε να κάνουμε? Θα χρησιμοποιήσουμε την data augmentation τεχνική. Αυτή η μέθοδος μας επιτρέπει να δημιουργήσουμε περισσότερα δείγματα για την εκπαίδευση του Deep Learning μοντέλου. Το data augmentation χρησιμοποιεί τα διαθέσιμα δείγματα δεδομένων για την παραγωγή των νέων, εφαρμόζοντας λειτουργίες εικόνας όπως περιστροφή, κλιμάκωση, μετάφραση κ.λπ. Αυτό καθιστά το μοντέλο μας ισχυρό στις αλλαγές εισόδου και οδηγεί σε καλύτερη γενίκευση. Η περιστροφή είναι μια από τις πιο χρησιμοποιούμενες και εύκολες στην εφαρμογή data augmentation τεχνικές. Όπως υποδηλώνει το όνομα, περιλαμβάνει περιστροφή της εικόνας σε αυθαίρετη γωνία και παροχή της ίδιας ετικέτας με την αρχική εικόνα.

- **Μετάφραση Εικόνας (Image Translation)**

  Η μετάφραση εικόνας είναι ένας γεωμετρικός μετασχηματισμός που χαρτογραφεί τη θέση κάθε αντικειμένου της εικόνας σε μια νέα θέση στην τελική εικόνα εξόδου. Μετά τη λειτουργία μετάφρασης, ένα αντικείμενο που βρίσκεται στη θέση (x, y) στην εικόνα εισαγωγής μετατοπίζεται σε νέα θέση (X, Y).

$$X = x + dx$$

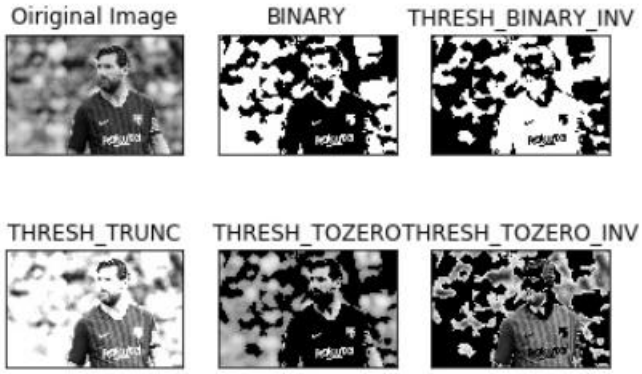
$$Y = y + dy$$

  Εδώ, dx και dy είναι οι αντίστοιχες μεταφράσεις σε διαφορετικές διαστάσεις. Η μετάφραση εικόνας μπορεί να χρησιμοποιηθεί για την προσθήκη μετατόπισης στο μοντέλο, καθώς με τη μετάφραση μπορούμε να αλλάξουμε τη θέση του αντικειμένου στην εικόνα δίνοντας περισσότερη ποικιλία στο μοντέλο που οδηγεί σε καλύτερη γενικευσιμότητα (generalizability) που λειτουργεί σε δύσκολες συνθήκες δηλαδή όταν το αντικείμενο δεν είναι τέλεια ευθυγραμμισμένο με το κέντρο της εικόνας. Αυτή η augmentation τεχνική

μπορεί επίσης να βοηθήσει το μοντέλο να ταξινομήσει σωστά τις εικόνες με μερικά ορατά αντικείμενα.

- **Κατώφλι (Thresholding)**

  Το thresholding είναι μια image segmentation μέθοδος. Συγκρίνει τιμές των pixel με μια threshold τιμή και τις ενημερώνει αναλόγως. Το OpenCV υποστηρίζει πολλαπλές threshold παραλλαγές. Το threshold μπορεί να

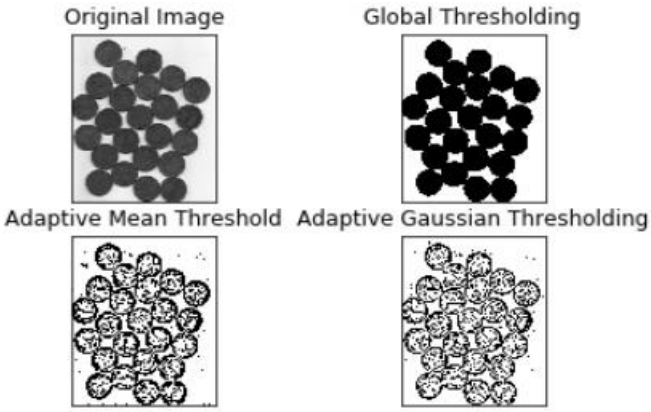


Εικόνα 2.2315-

εφαρμοστεί μόνο σε grayscale εικόνες. Μια απλή threshold εφαρμογή εικόνας θα μπορούσε να χωρίσει την εικόνα σε προσκήνιο και παρασκήνιο.

- **Προσαρμοστικό Κατώφλι (Adaptive Threshold)**

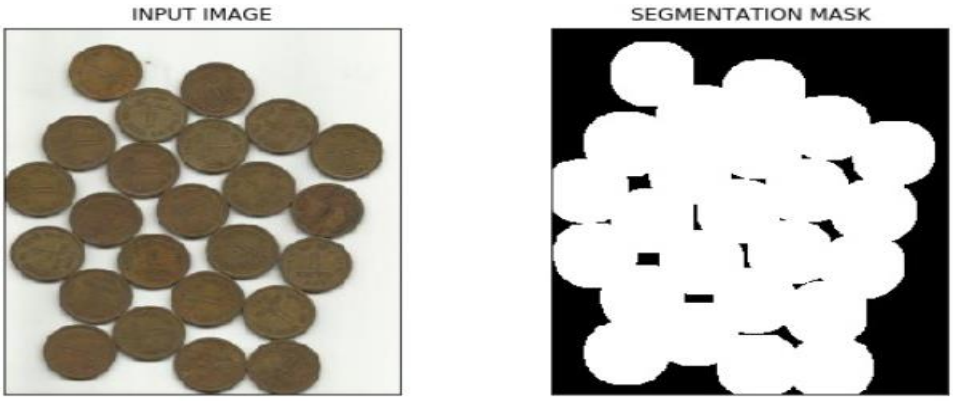
  Στην περίπτωση του adaptive threshold, χρησιμοποιούνται διαφορετικές threshold τιμές για διαφορετικά μέρη της εικόνας. Αυτή η λειτουργία δίνει καλύτερα αποτελέσματα για εικόνες με διαφορετικές συνθήκες φωτισμού - εξ ου και ο όρος adaptive.



Εικόνα 2.24- Adaptive threshold

- **Τμηματοποίηση Εικόνας (Image segmentation)**

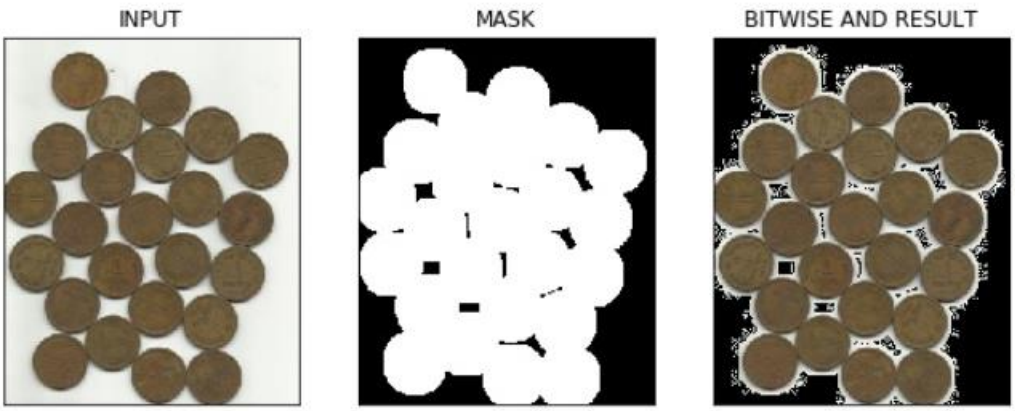
  Το image segmentation είναι το καθήκον της ταξινόμησης κάθε pixel της εικόνας σε κάποια κλάση. Για παράδειγμα, ταξινόμηση κάθε pixel ως προσκηνίου ή παρασκηνίου. Το image segmentation είναι σημαντικ;ο για την εξαγωγή των σχετικών τμημάτων από μια εικόνα. Ο watershed αλγόριθμος είναι ένας κλασικός image segmentation αλγόριθμος. Θεωρεί τις τιμές των pixel σε μια εικόνα ως τοπογραφία. Για την εύρεση των ορίων του αντικειμένου, απαιτείται η εισαγωγή των αρχικών δεικτών.



Εικόνα 2.2516-Image segmentation

- **Δυαδικές Λειτουργίες (Bitwise Operations)**

  Οι λειτουργίες Bitwise περιλαμβάνουν AND, OR, NOT και XOR. Στο Computer Vision, αυτές οι λειτουργίες είναι πολύ χρήσιμες όταν έχουμε μια εικόνα μάσκας και θέλουμε να εφαρμόσουμε αυτήν τη μάσκα σε μια άλλη εικόνα για να εξαγάγουμε την περιοχή ενδιαφέροντος. Στην παρακάτω εικόνα (Εικόνα 41), μπορούμε να δούμε ένα input image και το segmentation mask που υπολογίζεται χρησιμοποιώντας τον αλγόριθμο Watershed. Επιπλέον, έχουμε εφαρμόσει την AND λειτουργία για να αφαιρέσουμε το φόντο από την εικόνα και να εξαγάγουμε σχετικά τμήματα από την εικόνα.



Εικόνα 2.26-Bitwise operations

- **Αναγνώριση Άκρων (Edge Detection)**

  Τα άκρα είναι τα σημεία μιας εικόνας όπου η φωτεινότητα της εικόνας αλλάζει απότομα ή έχει ασυνέχειες. Τέτοιες ασυνέχειες γενικά αντιστοιχούν σε, ασυνέχειες βάθους, ασυνέχειες στον επιφανειακό προσανατολισμό, αλλαγές στις ιδιότητες του υλικού ή σε παραλλαγές στον φωτισμό της σκηνής. Τα άκρα είναι πολύ χρήσιμα χαρακτηριστικά μιας εικόνας που μπορούν να χρησιμοποιηθούν για διαφορετικές εφαρμογές όπως Object Classificaton στην εικόνα και Localization. Ακόμα και τα Deep Learning μοντέλα υπολογίζουν τα edge features για την εξαγωγή πληροφοριών σχετικά με τα αντικείμενα που υπάρχουν στην εικόνα. Οι άκρες διαφέρουν από τα περιγράμματα, καθώς δεν σχετίζονται με αντικείμενα, αλλά υποδηλώνουν τις αλλαγές στις τιμές των pixel μιας εικόνας. Το edge detection μπορεί να χρησιμοποιηθεί για image segmentation και ακόμη και για image sharpening

Εικόνα 2.27-Edge detection

2.4.3 OpenCV Εναντίων Ανταγωνισμού

Το OpenCV ανταγωνίζεται και αυτό ένα πλήθος άλλων Computer Vision frameworks και libraries, που καλύπτουν πολλές από τις ίδιες ανάγκες.

- **IPSDK.** Το IPSDK είναι μια Image Processing βιβλιοθήκη σε C ++ και Python. Η βιβλιοθήκη προσφέρει ένα πλήρες φάσμα Image Processing λειτουργιών για την επεξεργασία συνόλων δεδομένων, καθώς και μια ολοκληρωμένη και βελτιστοποιημένη σειρά λειτουργιών για επεξεργασία 2D και 3D. Το IPSDK προσαρμόζεται αυτόματα στην αρχιτεκτονική και τις δυνατότητες του επεξεργαστή. Τα χαρακτηριστικά αυτής της βιβλιοθήκης περιλαμβάνουν πλήρη υποστήριξη συμπλέγματος υπολογιστών (full PC cluster support), υψηλή απόδοση, high availability computing κ.λπ.

- **Imutils.** Το Imutils είναι ένα Computer Vision πακέτο που περιλαμβάνει μια σειρά από OpenCV και convenience functions για να κάνουν βασικές λειτουργίες επεξεργασίας εικόνας, όπως translation, rotation, resizing, skeletonisation, sorting contours, edge detection κ.τ.λ.

- **Matplotlib.** Το Matplotlib είναι μια ολοκληρωμένη βιβλιοθήκη οπτικοποίησης για τη δημιουργία στατικών, κινούμενων και διαδραστικών απεικονίσεων στην Python. Η βιβλιοθήκη μπορεί να εκτελέσει διάφορες λειτουργίες όπως ανάπτυξη γραφημάτων ποιότητας έκδοσης, εξαγωγή και ενσωμάτωση σε διάφορες μορφές αρχείων και διαδραστικά περιβάλλοντα και πολλά άλλα.

- **Scikit-Image.** Το Scikit-Image είναι μια δημοφιλής open source Python βιβλιοθήκη που περιλαμβάνει μια συλλογή αλγορίθμων για την επεξεργασία εικόνων. Είναι βασικά μια εργαλειοθήκη επεξεργασίας εικόνας για το SciPy. Η βιβλιοθήκη είναι βασισμένη στο scipy.ndimage για να παρέχει ένα ευέλικτο σύνολο Image Processing routines σε Python. Αυτή η Image Processing βιβλιοθήκη παρέχει ένα καλά τεκμηριωμένο API σε Python και εφαρμόζει

αλγόριθμους και βοηθητικά προγράμματα για χρήση σε εφαρμογές έρευνας, εκπαίδευσης και βιομηχανίας.

- **SimpleCV.** Το SimpleCV είναι ένα από τα δημοφιλή Machine Vision frameworks για την κατασκευή Computer Vision εφαρμογών. Γραμμένη σε Python, αυτή η βιβλιοθήκη βοηθάει στην ένταξη σε πολλές high powerd Computer Vision βιβλιοθήκες, όπως το OpenCV. Το framework είναι μια συλλογή βιβλιοθηκών και λογισμικού που μπορούν να χρησιμοποιηθούν για την ανάπτυξη εφαρμογών όρασης. Παρέχει ένα συνοπτικό και ευανάγνωστο interface για κάμερες, image manipulation, feature extraction και format conversion. Επιτρέπει επίσης στο χρήστη να εργάζεται σε τις εικόνες ή τις ροές βίντεο που προέρχονται από κάμερες web, Kinect, FireWire και κάμερες IP ή κινητά τηλέφωνα.

Κεφάλαιο 3-Σχεδίαση Και Υλοποίηση Λύσης

3.1 Περιγραφή Σεναρίου Εργασίας

Σκοπός της εν λόγω πτυχιακής εργασίας είναι η συγκριτική μελέτη, σχεδίαση αλλά και ανάπτυξη τεχνικών για την αποδοτικότερη αναγνώριση ύπαρξης ή μη αντικειμένων, την αναγνώριση της χωρικής τοποθέτησης σε περιπτώσεις πολλαπλής αναγνώρισης, όπως επίσης και η επίλυση προβλημάτων τοποθέτησης αντικειμένων από μια σταθερή θέση ταυτοποίησης προς μια άλλη, με ίδια χαρακτηριστικά του αντικειμένου. Για την επίτευξη των παραπάνω στόχων, χρησιμοποιούμαι δυο από τις πιο σύγχρονες μεθόδους για την υλοποίηση Computer Vision και Machine Learning projects. Αυτή του TensorFlow framework και της βιβλιοθήκης του OpenCV. Για να μελετήσουμε σωστά λοιπόν την απόδοση των αποτελεσμάτων αυτών των δύο προσεγγίσεων, δημιουργήσαμε ένα custom demo πάνω στο οποίο θα εξετάσουμε και τις δύο.

Το demo αποτελείται από παραμέτρους όπως, τρία είδη γεωμετρικών σχημάτων (τετράγωνο, κύκλο και τρίγωνο) τα οποία κατά την αρχή του σεναρίου θα θεωρούνται κενές

θέσεις, συγκεκριμένους περιμετρικούς χρωματισμούς για τα γεωμετρικά σχήματα (κόκκινο για το τετράγωνο, κίτρινο για τον κύκλο, μπλε για το τρίγωνο) και τυχαία τοποθέτηση των σχημάτων στο καρτεσιανό επίπεδο. Επίσης εμπεριέχονται 'πούλια' ιδίων σχημάτων και περιμετρικών αποχρώσεων τα οποία θα αποτελούν τα αντικείμενα προς τοποθέτηση στις κενές θέσεις των γεωμετρικών σχημάτων, αλλά και κάμερα λήψης εικόνας υπό αμετάβλητη γωνία. Πιο συγκεκριμένα, θα έχουμε εννέα γεωμετρικά σχήματα (τρία τετράγωνα, τρείς κύκλους και τρία τρίγωνα) μέσα στο frame της κάμερας, τοποθετημένα σε τυχαίες θέσεις στο επίπεδο και δύο Ranges of Interest (RoI), μέσα στα οποία, στο ένα θα τοποθετούμε τα πούλια και στο άλλο τα σχήματα/κενές θέσεις. Κάθε φορά θα τοποθετείται ένα πούλι στο πρώτο RoI και τα εννέα σχήματα στο δεύτερο.

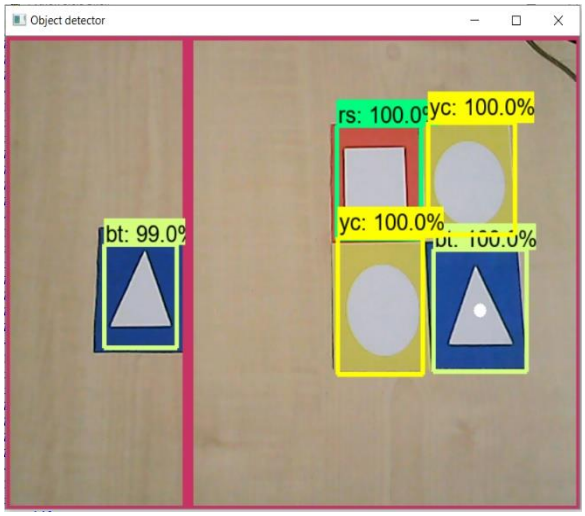
Σκοπός του αλγορίθμου, είναι η αναγνώριση του είδους σχήματος για το κάθε αντικείμενο που βρίσκεται εντός του frame της κάμερας, η αναγνώριση του χρωματισμού που περιβάλλει το κάθε σχήμα, αλλά και η εύρεση της ακριβής τοποθεσία του κάθε σχήματος στους άξονες X,Y του καρτεσιανού επιπέδου. Σκοπός του πρώτου RoI είναι η αναγνώριση του αντικειμένου (πούλι) που θα τοποθετήσουμε στο εσωτερικό του, αν δηλαδή θα είναι ένα κόκκινο τετράγωνο, ένας κίτρινος κύκλος ή ένα μπλε τρίγωνο. Ανάλογα λοιπόν με το τι θα αντιληφθεί ο αλγόριθμός μας πως είναι το αντικείμενο που τοποθετήσαμε στο εσωτερικό του RoI, θα τοποθετήσει έναν διακριτικό κύκλο στο κέντρο του πρώτου διαθέσιμου κενού αντικειμένου του ιδίου σχήματος και περιμετρικής απόχρωσης, στο εσωτερικό του δεύτερου RoI, εκεί δηλαδή που θα βρίσκονται τα σχήματα/κενές θέσεις. Έτσι εμείς σαν χρήστης θα μπορούμε να πάρουμε το πούλι από το πρώτο RoI και να το τοποθετήσουμε στην κενή θέση που μας έχει υποδείξει ο αλγόριθμός μας. Το συγκεκριμένο demo είναι μεγάλου τεχνολογικού ενδιαφέροντος, καθώς ανά πάσα στιγμή μπορούμε να γνωρίζουμε τις ακριβείς τοποθεσίες όλων των σχημάτων μέσα στο frame της κάμερας, γεγονός που καθιστά πολύ εύκολη την απομάκρυνση του ανθρώπινου παράγοντα και την προσαρμογή ενός πλήρως αυτοματοποιημένου μηχανικού βραχίονα για να εκτελεί την μεταφορά του πουλιού προς την διαθέσιμη κενή θέση.

Μέσω του TensorFlow, σκοπός μας είναι να δημιουργήσουμε το δικό μας μοντέλο αναγνώρισης αντικειμένων έτσι ώστε καθ' όλη τη διάρκεια του demo, το framework να είναι ικανό να αναγνωρίζει τα σχήματα που θα τοποθετούμε στο εσωτερικό των δύο RoIs. Για να γίνει αυτό, θα πρέπει πρώτα να συλλέξουμε τις φωτογραφίες με τα γεωμετρικά σχήματα που θα χρησιμοποιήσουμε για την εκπαίδευση του μοντέλου και μέσω του LabelImage, ένα open source labeling tool, να ορίσουμε bounding boxes στις περιοχές των φωτογραφιών όπου

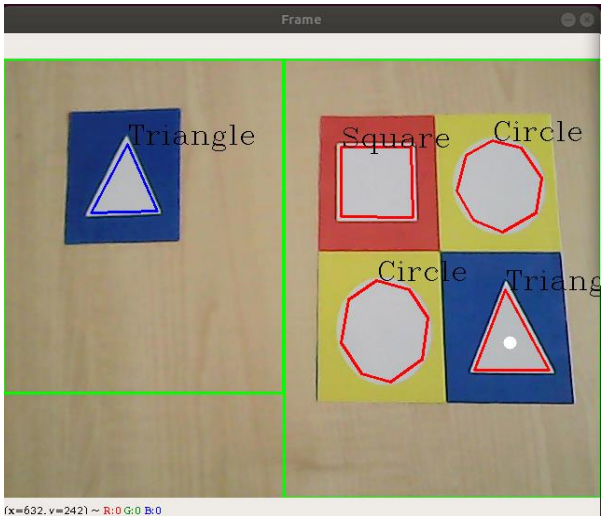
βρίσκονται τα γεωμετρικά σχήματα, αλλά και να τα ονομάσουμε. Έπειτα μέσω του Faster-R-CNN μοντέλου και της Transfer Learning τεχνικής, το TensorFlow θα εκπαιδεύσει νέο μοντέλο σύμφωνα με τις δικές μας ανάγκες.

Μέσω της βιβλιοθήκης του OpenCV, σκοπός μας είναι να έχουμε μια αλγοριθμική προσέγγιση για τα ζητούμενα του demo. Για να γίνει αυτό θα εφαρμόσουμε αρχικά μάσκα συγκεκριμένου εύρους χρώματος, ώστε όλες οι λειτουργίες που θα χρησιμοποιήσουμε, να λάβουν χώρα μόνο στις περιοχές του frame της κάμερας που τα pixel βρίσκονται εντός αυτού του εύρους τιμών. Για την αναγνώριση του είδους των γεωμετρικών σχημάτων, θα μετρήσουμε τις πλευρές των περιγραμμάτων του κάθε σχήματος. Για την αναγνώριση των περιμετρικών αποχρώσεων του κάθε σχήματος, θα ορίσουμε τρία lower και higher values για το εύρος τιμών των pixels της εκάστοτε απόχρωσης των σχημάτων. Τέλος θα δημιουργήσουμε δύο RoIs μέσα στα οποία θα εφαρμοστούν οι παραπάνω λειτουργίες για την αναγνώριση των αντικειμένων.

Ακολουθούν δύο ενδεικτικές φωτογραφίες λειτουργίας του demo και με τις προσεγγίσεις.



Εικόνα 3.0- TensorFlow προσέγγιση



Εικόνα 3.1- OpenCV προσέγγιση

3.2 Python

Η Python είναι μια ερμηνευτική, αντικειμενοστραφής γλώσσα προγραμματισμού υψηλού επιπέδου με δυναμική σημασιολογία. Οι ενσωματωμένες δομές δεδομένων υψηλού επιπέδου, σε συνδυασμό με τη δυναμική πληκτρολόγηση και τη δυναμική δέσμευση, την καθιστούν πολύ ελκυστική για την ταχεία ανάπτυξη εφαρμογών, καθώς και για χρήση ως γλώσσα ανάπτυξης ή τον συνδετικό κρίκο για σύνδεση των υπαρχόντων στοιχείων. Η απλή, εύχρηστη σύνταξη της Python δίνει έμφαση στην αναγνωσιμότητα και συνεπώς μειώνει το κόστος συντήρησης του προγράμματος. Η Python υποστηρίζει λειτουργικές μονάδες και πακέτα, τα οποία ενθαρρύνουν τη διαμόρφωση του προγράμματος και την επαναχρησιμοποίηση κώδικα. Ο διερμηνέας της Python και η εκτεταμένη βασική βιβλιοθήκη διατίθενται σε source ή binary μορφή χωρίς χρέωση για όλες τις μεγάλες πλατφόρμες και μπορούν να διανεμηθούν ελεύθερα. Συχνά, οι προγραμματιστές ερωτεύονται την Python λόγω της αυξημένης παραγωγικότητας που παρέχει. Επειδή δεν υπάρχει compilation βήμα, ο κύκλος edit-test-debug είναι απίστευτα γρήγορος. Ο εντοπισμός σφαλμάτων των προγραμμάτων Python είναι εύκολος: ένα σφάλμα ή κακή είσοδος δεν θα προκαλέσει ποτέ σφάλμα τμηματοποίησης. Αντ 'αυτού, όταν ο διερμηνέας ανακαλύπτει ένα σφάλμα, δημιουργεί μια εξαίρεση. Όταν το πρόγραμμα δεν έχει την εξαίρεση, ο διερμηνέας εκτυπώνει ένα ίχνος στοίβας. Ένα source level debugger επιτρέπει την επιθεώρηση local και global μεταβλητών, την αξιολόγηση αυθαίρετων εκφράσεων, τον καθορισμό σημείων διακοπής, τον έλεγχο του κώδικα μια γραμμή κάθε φορά και ούτω καθεξής. Το πρόγραμμα εντοπισμού σφαλμάτων είναι γραμμένο σε Python και αυτό, μαρτυρώντας την ενδοσκοπική ισχύ της Python. Από την άλλη πλευρά, συχνά ο γρηγορότερος τρόπος για την αποσφαλμάτωση ενός προγράμματος είναι να προσθέσουμε μερικές print δηλώσεις στο source. Ο γρήγορος κύκλος edit-test-debug καθιστά αυτήν την απλή προσέγγιση πολύ αποτελεσματική.

Η Python συχνά συγκρίνεται με άλλες ερμηνευτικές γλώσσες όπως Java, JavaScript, Perl, Tcl ή Smalltalk. Στην πράξη, η επιλογή μιας γλώσσας προγραμματισμού συχνά υπαγορεύεται από άλλους πραγματικούς περιορισμούς όπως το κόστος, η διαθεσιμότητα, η εκπαίδευση και οι προηγούμενες επενδύσεις ή ακόμη και η συναισθηματική προσκόλληση.

Υπάρχουν διάφοροι λόγοι που η Python είναι μια καλή επιλογή ως γλώσσα προγραμματισμού, ανάλογα με την προοπτική και το ιστορικό μας. Αυτοί που είναι νέοι στον προγραμματισμό μπορούν να επωφεληθούν από το υψηλό επίπεδο αφαίρεσης της Python.

Είναι πολύ διαδραστική και είναι γνωστή για τις «ισχυρές απόψεις» της σχετικά με συγκεκριμένη σύνταξη (συμπεριλαμβανομένου του κενού χώρου). Η Python, όπως και άλλες γλώσσες υψηλού επιπέδου, έχει μια διαδικασία συλλογής απορριμμάτων για τη διαχείριση της μνήμης ή τη διαγραφή αχρησιμοποίητων πόρων. Ένας χρήστης μπορεί να λάβει άμεσα σχόλια από τον διερμηνέα πληκτρολογώντας python στο command line ή χρησιμοποιώντας projects όπως το JupyterLab αν θέλει μια εμπειρία ανάπτυξης που βασίζεται σε πρόγραμμα περιήγησης. Πολλοί χρήστες εκτιμούν επίσης ότι η Python έχει μια αυστηρή σύνταξη που επιβάλλεται από τον compiler, καθιστώντας εύκολο να έχεις έναν «σωστό τρόπο» για να γράψεις ένα πρόγραμμα.

Ανεξάρτητα από το επίπεδο εμπειρίας, οι προγραμματιστές από πολλά διαφορετικά υπόβαθρα συνεισφέρουν στη γλώσσα με σημαντικούς τρόπους. Η Python διαθέτει ένα ώριμο οικοσύστημα τόσο δωρεάν όσο και ιδιόκτητων εργαλείων, συμπεριλαμβανομένων ολοκληρωμένων περιβαλλόντων ανάπτυξης (IDE), linters και πλαισίων. Οι ενότητες που μοιράζονται μέσω PyPI και Conda έχουν βάθος και εύρος που θα καλύπτει σχεδόν κάθε θέμα. Μερικά από τα πιο δημοφιλή περιλαμβάνουν:

- Web frameworks όπως Django, Pyramid, Flask και Bottle
- Internet protocol support στην standard βιβλιοθήκη για JSON, HTML, XML, FTP, IMAP και sockets
- Data science και Machine Learning με SciPy, Pandas, IPython, NumPy και πολλά άλλα

Ίσως το πιο σημαντικό χαρακτηριστικό της Python, είναι πως έχει μια τεράστια κοινότητα χρηστών. Η δημοτικότητα της Python είναι αιτία και αποτέλεσμα της κοινότητάς της. Ήταν η νούμερο ένα γλώσσα προγραμματισμού το 2018, σύμφωνα με την κατάταξη του IEEE Spectrum και είναι η νούμερο ένα "Most Wanted" και νούμερο δύο "Most Loved", σύμφωνα με το StackOverflow 2019 Developer Survey. Οι Pythonistas όπως αποκαλούνται, ως μέλη της κοινότητας συναντώνται σε όλο τον κόσμο σε χιλιάδες σε συνέδρια της PyCon. Αυτό σημαίνει ότι ανεξάρτητα από το πρόβλημα που προσπαθούμε να λύσουμε, υπάρχουν πιθανότητες να υπάρχουν ήδη ισχυροί άνθρωποι που εργάζονται σε μια λύση. Οι πιθανότητες είναι επίσης καλές που έχουν κοινόχρηστο κώδικα, τεκμηρίωση, μαθήματα και παραδείγματα

για να βοηθήσουν στον προγραμματισμό μιας λύσης στην Python. Υπάρχουν πολλά IDE και άλλα εργαλεία ανάπτυξης για να διαλέξουμε χιλιάδες πακέτα ανοιχτού κώδικα διαθέσιμα για να επεκτείνουμε την Python για να κάνουμε οτιδήποτε μπορούμε να σκεφτούμε.

3.2.1 Πλαίσια, Τμήματα και Βιβλιοθήκες της Python (Python Frameworks, Modules And Libraries)

Για τη δομή του project μας χρησιμοποιήσαμε ένα μικρό κομμάτι από το τεράστιο εύρος διαθέσιμων βιβλιοθηκών που μας προσφέρει η Python και στη συνέχεια θα αναλύσουμε μερικές από αυτές

Αρχικά χρησιμοποιήσαμε την NumPy βιβλιοθήκη για την παρουσίαση των τριών χρωματικών καναλιών ενός pixel, σαν ένα πίνακα τριών διαστάσεων στο OpenCV. Η NumPy είναι μια βιβλιοθήκη για της Python, η οποία προσθέτει υποστήριξη για μεγάλες πολυδιάστατες συστοιχίες και πίνακες, μαζί με μια μεγάλη συλλογή μαθηματικών συναρτήσεων υψηλού επιπέδου για λειτουργία σε αυτές τις συστοιχίες. Η γλώσσα προγραμματισμού Python δεν είχε αρχικά σχεδιαστεί για αριθμητικούς υπολογισμούς, αλλά προσέλκυσε την προσοχή της επιστημονικής και μηχανικής κοινότητας από νωρίς. Τα Python bindings της ευρέως χρησιμοποιούμενης Computer Vision βιβλιοθήκης OpenCV, χρησιμοποιούν NumPy arrays για αποθήκευση και λειτουργία δεδομένων. Δεδομένου ότι οι εικόνες με πολλά κανάλια αντιπροσωπεύονται απλώς ως τρισδιάστατα arrays, το indexing, το slicing ή το masking με άλλα arrays είναι πολύ αποτελεσματικοί τρόποι πρόσβασης σε συγκεκριμένα pixels μιας εικόνας. Η έγχρωμη εικόνα είναι μια συλλογή τριών παράλληλων επιπέδων και κάθε επίπεδο αντιπροσωπεύει την ένταση χρώματος ενός μόνο καναλιού. Οι περισσότερες από τις έγχρωμες εικόνες αποθηκεύονται σε μορφή RGB, R για κόκκινο, G για πράσινο και B για μπλε. Εάν πάρουμε ένα μόνο στοιχείο ενός πίνακα, αυτό θα αντιπροσωπεύει την ένταση του κόκκινου, του πράσινου και του μπλε αυτού του pixel. Ο πίνακας NumPy ως καθολική δομή δεδομένων στο OpenCV για εικόνες, extracted feature points, filter kernels και πολλά άλλα, απλοποιεί ευρέως τη ροή εργασιών προγραμματισμού και τον εντοπισμό σφαλμάτων.

Επίσης χρησιμοποιήσαμε το os Python module για την απαραίτητη πλοήγηση μεταξύ των πολλαπλών directories και folders τουTensorFlow. Η Python έχει διάφορες os-dependent

εργασίες και επιτρέπει στον προγραμματιστή να χρησιμοποιεί πολλαπλές os-dependent functionalities με os Python module. Αυτό το πακέτο συνοψίζει τις λειτουργίες της πλατφόρμας και παρέχει στη python λειτουργίες για πλοήγηση, δημιουργία, διαγραφή και τροποποίηση αρχείων και φακέλων. Συγκεκριμένα χρησιμοποιήσαμε τις:

- **os.getcwd()**

  Η μέθοδος αυτή στην Python επιστρέφει το τρέχων directory εργασίας μιας διαδικασίας. Κάθε διαδικασία που εκτελείται κάτω από ένα operating system έχει έναν συσχετιζόμενο working directory, το οποίο καλείται ως το τρέχων working directory της διαδικασίας.

- **os.path.join**

  Η μέθοδος αυτή στην Python, συνδυάζει ένα ή περισσότερα path names σε path. Αυτή η μέθοδος χρησιμοποιείται συχνά με μεθόδους os όπως os.walk() για να δημιουργήσουμε το τελικό path για ένα αρχείο ή φάκελο

Τέλος χρησιμοποιήσαμε το sys Python module ώστε να δώσουμε συγκεκριμένο path στον interpreter για να ψάξει. Το sys Python module παρέχει συναρτήσεις και μεταβλητές που χρησιμοποιούνται για τον χειρισμό διαφορετικών τμημάτων του Python Runtime Enviroment. Μας επιτρέπει να έχουμε πρόσβαση σε συγκεκριμένες παραμέτρους και λειτουργίες του συστήματος. Συγκεκριμένα χρησιμοποιούμε το **sys.path.append** το οποίο είναι μια λίστα από strings που καθορίζει τη διαδρομή αναζήτησης για τα modules. Βασικά αυτό λέει στην Python ποιες τοποθεσίες να κοιτάξει όταν προσπαθεί να εισαγάγει ένα module. Σύμφωνα με το Python documentation, το sys.path αρχικοποιείται από μια μεταβλητή περιβάλλοντος που ονομάζεται PYTHONPATH, συν μια προεπιλεγμένη εξαρτώμενη από την εγκατάσταση. Η PYTHONPATH είναι μια environmental variable την οποία μπορούμε να ορίσουμε για να προσθέσουμε επιπλέον directories όπου η Python θα ψάξει για modules και packages

3.3 TensorFlow Προσέγγιση

Σε αυτό το υποκεφάλαιο θα αναλύσουμε την προσέγγιση μέσω TensorFlow για την διεκπεραίωση της συγκεκριμένης πτυχιακής εργασίας και σκοπός είναι να εξηγήσουμε τι θα κάνουμε, για να εκπαιδεύσουμε το δικό μας CNN Object Detection Classifier.

3.3.1 Πλατφόρμα Anaconda (Anaconda Platform)

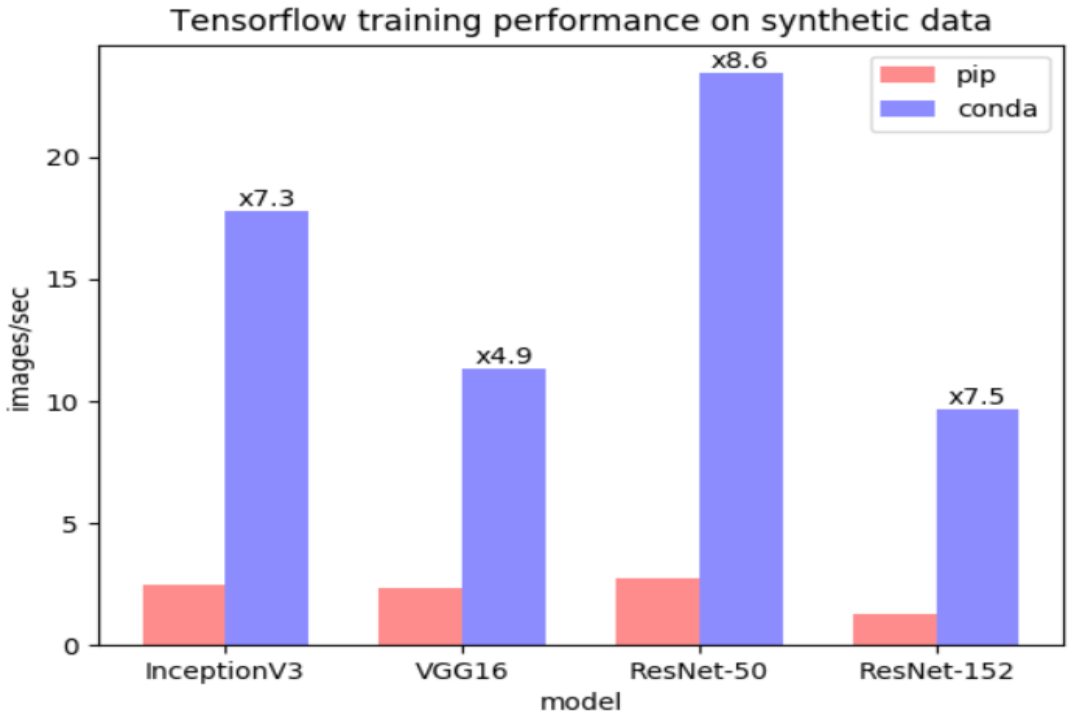
Το Anaconda είναι ένα distribution των Python και R γλωσσών προγραμματισμού για επιστημονικούς υπολογισμούς όπως data science, machine learning applications, large-scale data processing, predictive analytics κ.τ.λ. που στοχεύει στην απλούστευση της διαχείρισης και ανάπτυξης πακέτων. Το distribution περιλαμβάνει data science πακέτα κατάλληλα για Windows, Linux και macOS. Αναπτύχθηκε και συντηρείται από την Anaconda, Inc., η οποία ιδρύθηκε από τους Peter Wang και Travis Oliphant το 2012.

Υπάρχουν πολλές μέθοδοι που μπορούν να χρησιμοποιηθούν για την εγκατάσταση του TensorFlow, όπως η χρήση pip για την εγκατάσταση των wheels που είναι διαθέσιμοι στο PyPI. Η εγκατάσταση του TensorFlow με χρήση πακέτων conda προσφέρει πολλά οφέλη, όπως ένα πλήρες σύστημα διαχείρισης πακέτων, ευρύτερη υποστήριξη πλατφόρμας, μια πιο βελτιωμένη εμπειρία GPU και καλύτερη απόδοση της CPU. Αυτά τα πακέτα είναι διαθέσιμα μέσω του Anaconda Repository και η εγκατάσταση τους είναι τόσο εύκολη όσο η εκτέλεση του "conda install tensorflow" ή "conda install tensorflow-gpu" από ένα command line interface. Ένα βασικό πλεονέκτημα της εγκατάστασης του TensorFlow χρησιμοποιώντας conda και όχι pip είναι το αποτέλεσμα του συστήματος διαχείρισης πακέτων conda. Όταν το TensorFlow εγκαθίσταται χρησιμοποιώντας conda, το conda εγκαθιστά όλα τα απαραίτητα και συμβατά dependencies και για τα πακέτα αυτόματα. Οι χρήστες δεν χρειάζεται να εγκαταστήσουν επιπλέον λογισμικό μέσω system package manager ή με άλλα μέσα. Επιπλέον, οποιοδήποτε από τα 1.400+ επαγγελματικά κατασκευασμένα πακέτα στο Anaconda Repository μπορεί να εγκατασταθεί παράλληλα με το TensorFlow για να παρέχει ένα ολοκληρωμένο data science περιβάλλον. Αυτά τα πακέτα εγκαθίστανται σε ένα απομονωμένο conda environment του οποίου τα περιεχόμενα δεν επηρεάζουν άλλα environments.

Πολλές από τις λειτουργίες του TensorFlow μπορούν να επιταχυνθούν χρησιμοποιώντας GPU NVIDIA. Το κέρδος επιτάχυνσης μπορεί να είναι ιδιαίτερα μεγάλο όταν εκτελεί υπολογιστικά απαιτητικές Deep Learningεφαρμογές. Κατά την εγκατάσταση του

TensorFlow χρησιμοποιώντας pip, οι βιβλιοθήκες CUDA και CuDNN που απαιτούνται για υποστήριξη GPU πρέπει να εγκατασταθούν ξεχωριστά, προσθέτοντας ένα επιπλέον φορτίο στην έναρξη της εγκατάστασης. Όταν η GPU accelerated έκδοση του TensorFlow εγκαθίσταται χρησιμοποιώντας conda, με την εντολή "conda install tensorflow-gpu", αυτές οι βιβλιοθήκες εγκαθίστανται αυτόματα, με εκδόσεις που είναι γνωστό ότι είναι συμβατές με το πακέτο tensorflow-gpu. Επιπλέον, το conda εγκαθιστά αυτές τις βιβλιοθήκες σε μια τοποθεσία όπου δεν θα παρεμβαίνουν σε άλλα instances αυτών των βιβλιοθηκών που ενδέχεται να έχουν εγκατασταθεί μέσω άλλης μεθόδου. Ανεξάρτητα από τη χρήση του tensorflow-gpu που είναι εγκατεστημένο σε pip ή conda, ο NVIDIA driver πρέπει να εγκατασταθεί ξεχωριστά.

Τα πακέτα conda TensorFlow έχουν επίσης σχεδιαστεί για καλύτερη απόδοση σε CPU μέσω της χρήσης της Intel® Math Kernel Library για Deep Neural Networks (Intel® MKL-DNN). Ξεκινώντας με την έκδοση 1.9.0, τα πακέτα conda TensorFlow κατασκευάζονται χρησιμοποιώντας τη βιβλιοθήκη Intel® MKL-DNN, η οποία δείχνει σημαντικές βελτιώσεις απόδοσης. Για παράδειγμα, στην εικόνα 43 συγκρίνεται η απόδοση της εκπαίδευσης και τα συμπεράσματα σε δύο διαφορετικά image classification models, χρησιμοποιώντας το



Εικόνα 3.2-Απόδοση εκπαίδευσης του TensorFlow σε ορισμένα κοινά Deep Learning μοντέλα χρησιμοποιώντας συνθετικά δεδομένα. Οι δείκτες αναφοράς πραγματοποιήθηκαν σε ένα Intel® Xeon® Gold 6130.

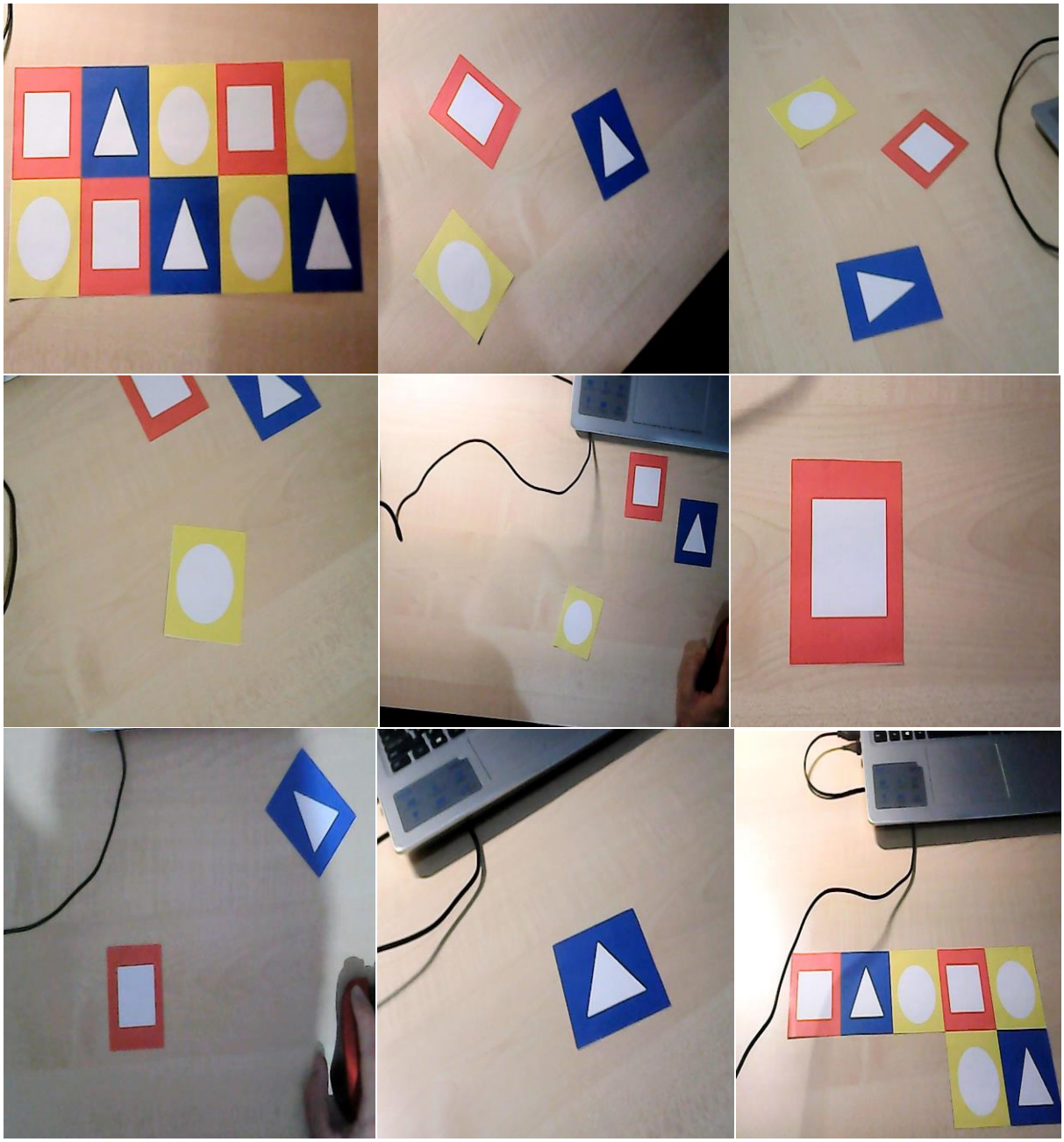
TensorFlow εγκατεστημένο χρησιμοποιώντας conda και την ίδια έκδοση εγκατεστημένη χρησιμοποιώντας pip. Η απόδοση της εγκατεστημένης conda έκδοσης έχει πάνω από οκτώ φορές την ταχύτητα του pip εγκατεστημένου πακέτου σε πολλά από τα σημεία αναφοράς.

Για την εκπαίδευση και του δικού μας CNN Object Detection Classifier, χρησιμοποιήσαμε pip install και TensorFlow CPU version.

3.3.2 Συγκέντρωση και Ονοματοδοσία Φωτογραφιών

Το TensorFlow χρειάζεται εκατοντάδες εικόνες ενός αντικειμένου για να εκπαιδεύσει έναν καλό Detection Classifier. Για να εκπαιδεύσουμε έναν ισχυρό classifier, οι εικόνες εκπαίδευσης θα πρέπει να έχουν τυχαία αντικείμενα στην εικόνα μαζί με τα επιθυμητά αντικείμενα και θα πρέπει να έχουν ποικίλα υπόβαθρα και συνθήκες φωτισμού. Πρέπει να υπάρχουν κάποιες εικόνες όπου το επιθυμητό αντικείμενο είναι μερικώς σκοτεινό, επικαλυπτόμενο με κάτι άλλο ή μόνο στη μέση της εικόνας. Για τον δικό μας classifier, έχουμε τρία διαφορετικά αντικείμενα που θέλουμε να ανιχνεύσουμε (κόκκινο τετράγωνο, κίτρινο κύκλο και μπλε τρίγωνο) και πήραμε 50 φωτογραφίες του κάθε αντικειμένου, είτε μόνο του στη λήψη, είτε με άλλα πολλαπλά αντικείμενα μαζί.

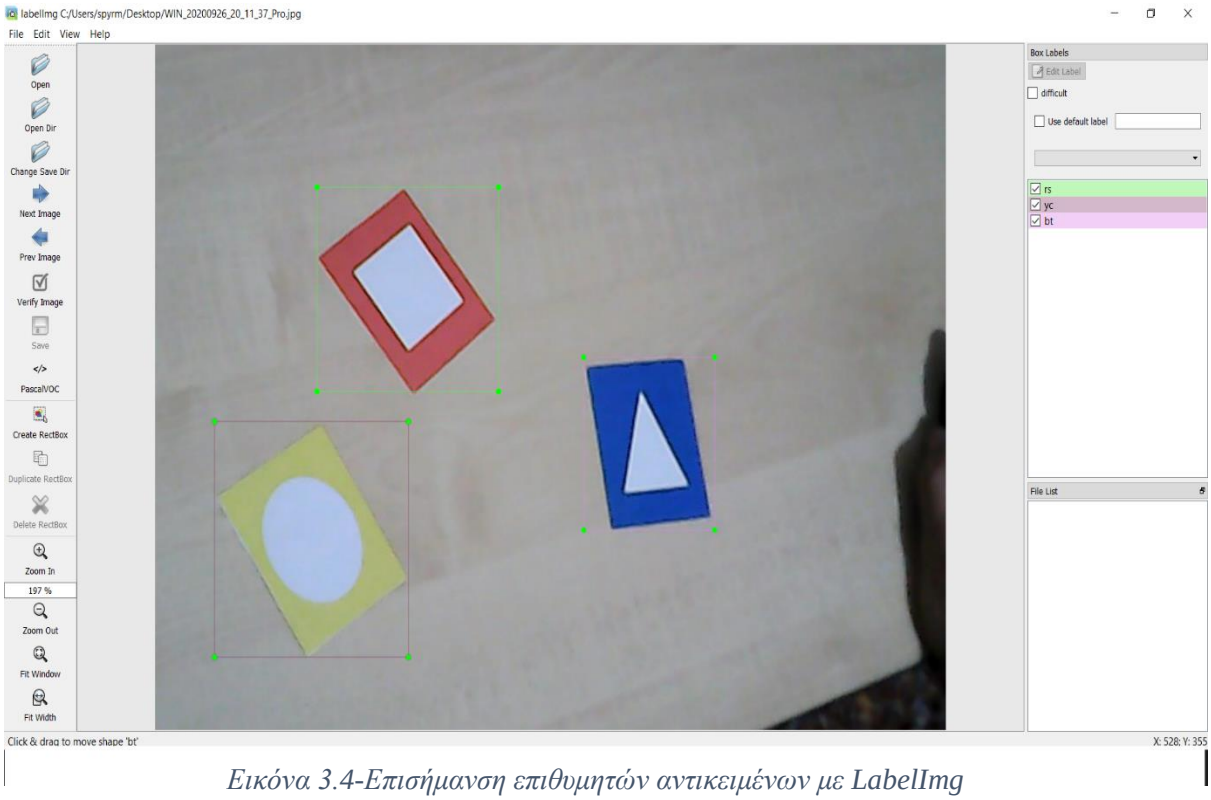
Για την λήψη των φωτογραφιών χρησιμοποιήσαμε μια απλή webcam διότι δεν πρέπει να είναι πολύ μεγάλες. Πρέπει να καταναλώνουν λιγότερο από 200kb η μια και η ανάλυσή τους δεν θα πρέπει να ξεπερνά τα 720x1280 pixels. Όσο πιο μεγάλες είναι οι εικόνες, τόσο περισσότερο θα διαρκέσει η εκπαίδευση του classifier.



Εικόνα 3.3 - *Συγκεντρωμένες φωτογραφίες για εκπαίδευση του μοντέλου*

Έχοντας συγκεντρώσει όλες τις εικόνες, ήρθε η ώρα να επισημάνουμε τα επιθυμητά αντικείμενα σε κάθε εικόνα. Το LabelImg είναι ένα εξαιρετικό εργαλείο για την επισήμανση εικόνων και η σελίδα του στο GitHub έχει πολύ σαφείς οδηγίες σχετικά με τον τρόπο εγκατάστασης και χρήσης του. Στην εικόνα 3.3 βλέπουμε ένα παράδειγμα της επισήμανσης των επιθυμητών αντικειμένων με το LabelImg εργαλείο. Επισημαίνοντας τα επιθυμητά αντικείμενα σε κάθε εικόνα λοιπόν, χρησιμοποιούμε την supervised τεχνική του Machine Learning. Μια τεχνική εκμάθησης μιας function που χαρτογραφεί μια είσοδο σε μια έξοδο με

βάση παραδείγματα από ζεύγη εισόδου-εξόδου. Παραπέμπει μια function από labelled training data που αποτελούνται από ένα σύνολο παραδειγμάτων εκπαίδευσης.



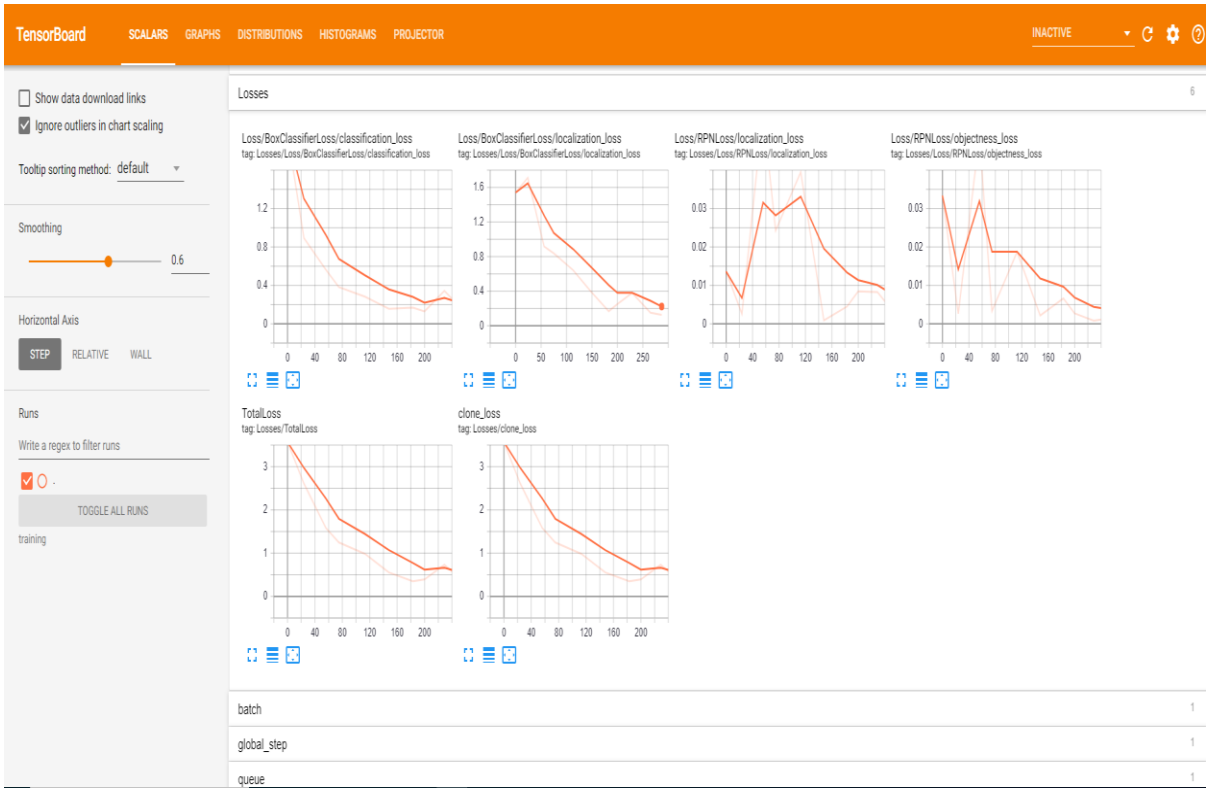
Εικόνα 3.4-Επισήμανση επιθυμητών αντικειμένων με LabelImg

3.3.3 Εκπαίδευση Μοντέλου (Model Training)

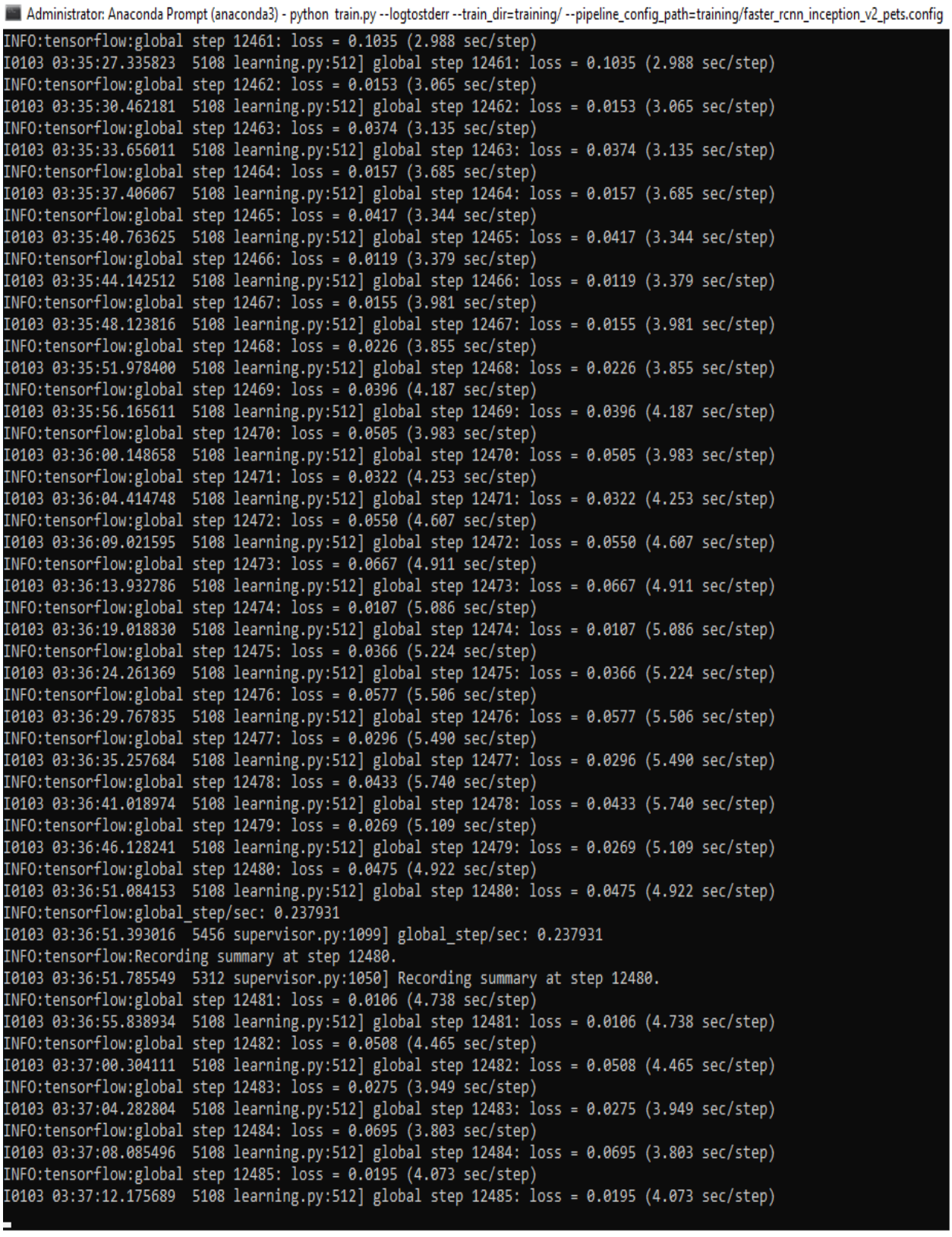
Έχοντας επισημάνει όλες τις απαραίτητες φωτογραφίες και έχοντας εκτελέσει όλες τις προαπαιτούμενες ενέργειες, μπορούμε πλέον να ξεκινήσουμε την εκπαίδευση του Object Detection Classifier. Η εκπαίδευση θα γίνει με τη χρήση της Transfer Learning τεχνικής από το Faster-R-CNN-inception-v2-pets pre trained model.

Το TensorFlow λοιπόν ξεκινάει την εκπαίδευση και αρχίζει να περνά τα training batches και να αναφέρει το loss σε κάθε βήμα. Το loss ξεκινάει από πολύ ψηλά και με το πέρας του χρόνου εκπαίδευσης του μοντέλου, πέφτει σταδιακά. Μπορούμε να πούμε πως το μοντέλο είναι επαρκώς εκπαιδευμένο, όταν το TotalLoss έχει πλέον πέσει σταθερά κάτω από το 0.05 ή όταν το γράφημα του TotalLoss έχει σταθεροποιηθεί και τείνει να μοιάζει με ευθεία γραμμή.

Μπορούμε να παρακολουθούμε την πορεία του TotalLoss και γενικά της εκπαίδευσης του μοντέλου μέσω του TensorBoard. Στην παραπάνω εικόνα (Εικόνα 46), μπορούμε να διακρίνουμε την καμπύλη του TotalLoss κατά τα πρώτα στάδια της εκπαίδευσης.



Εικόνα 3.5-TensorBoard

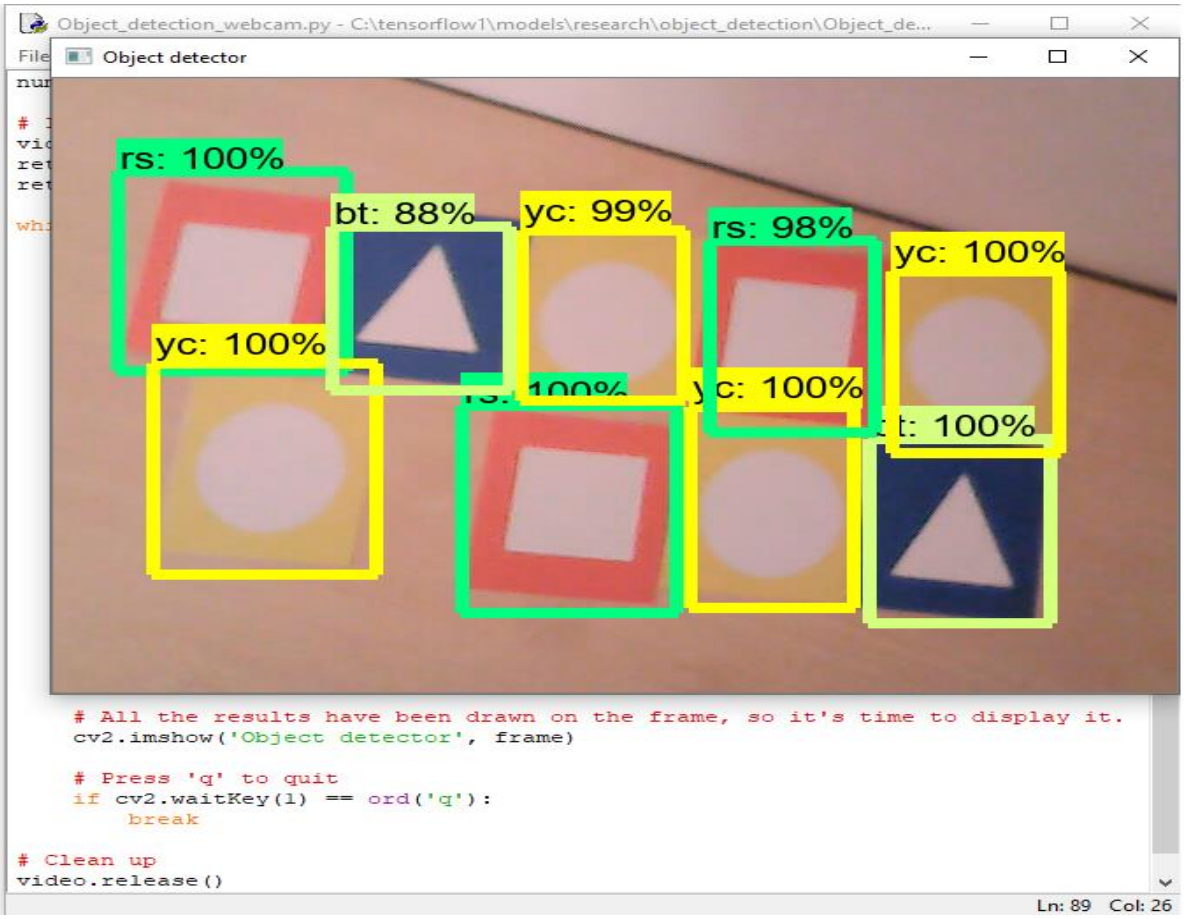


Εικόνα 3.6-Ολοκλήρωση εκπαίδευσης

Έχοντας εκτελέσει 12480 βήματα και ενώ το TensorFlow έχει γράψει 15 ώρες συνεχόμενης εκπαίδευσης, τερματίζουμε την διαδικασία και παρατηρούμε από το Anaconda Command Prompt πως το TotalLoss είναι πλέον σχετικά σταθερά κάτω από το 0.05, πράγμα που μαρτυρά και το γράφημα του TensorBoard στην εικόνα 3.6.

Εικόνα 3.7-Τελικό TotalLoss

Τέλος πρέπει να εξάγουμε το frozen inference graph από το τελευταίο checkpoint στα 12480 βήματα. Το frozen inference graph είναι ένα protobuf αρχείο το οποίο μετατρέπει αριθμητικές μεταβλητές όπως τα weights από το τελευταίο checkpoint και εμπεριέχει το graph definition. Είναι ουσιαστικά το αρχείο το οποίο εμπεριέχει το Object Detection Classifier μας. Στην εικόνα 3.7 βλέπουμε τα αποτελέσματα που έχει ο classifier μέσω webcam feed, εμφανίζοντας bounding boxes στα επιθυμητά αντικείμενα.



Εικόνα 3.8-Object Detection Classifier

3.4 OpenCV Προσέγγιση

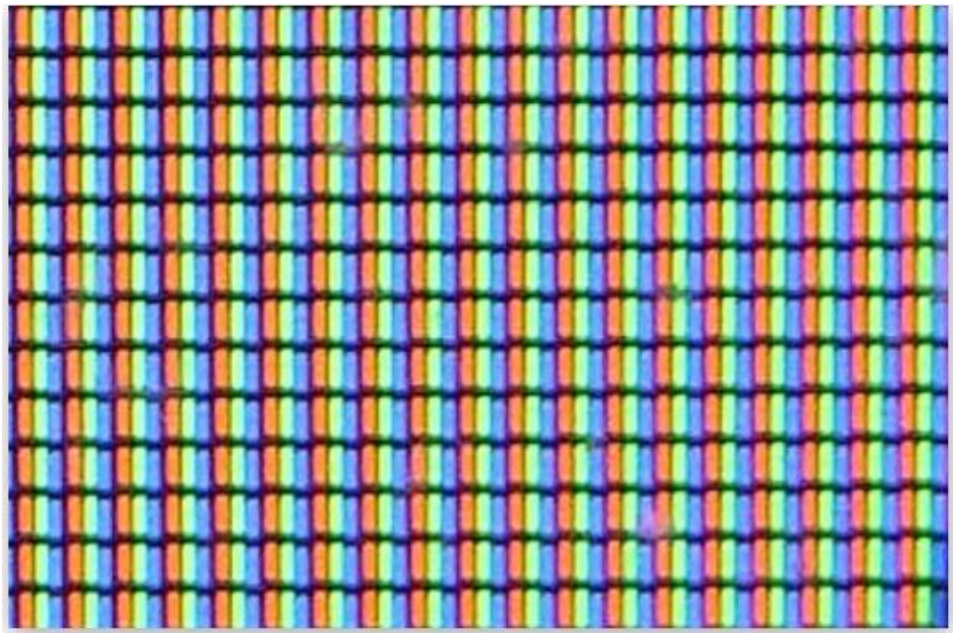
Σε αυτό το υποκεφάλαιο θα αναλύσουμε την προσέγγιση μέσω OpenCV για την διεκπεραίωση της συγκεκριμένης πτυχιακής εργασίας με αποκλειστικά αλγοριθμικό τρόπο. Θα δομήσουμε από την αρχή ένα πρόγραμμα σε Python και θα χρησιμοποιήσουμε ένα μεγάλο εύρος των λειτουργιών που προσφέρει η βιβλιοθήκη του OpenCV για να επιτεύξουμε το έργο μας.

3.4.1 Χρωματικοί Χώροι και Μάσκα (Color Spaces And Masking)

Υπάρχουν περισσότεροι από 150 μέθοδοι μετατροπής color spaces στο OpenCV. Εμείς θα χρησιμοποιήσουμε έναν, ο οποίος χρησιμοποιείται και ευρύτερα σε πολλά Computer Vision έργα: BGR ↔ HSV.

Στο πιο κοινό color space, αυτό του RGB (Red, Green, Blue) ή BGR όπως συνηθίζεται να αναφέρεται στο OpenCV, τα χρώματα αντιπροσωπεύονται με βάση τα κόκκινα, πράσινα και μπλε στοιχεία τους. Σε πιο τεχνικούς όρους, το RGB περιγράφει ένα χρώμα ως tuple τριών στοιχείων. Κάθε στοιχείο μπορεί να πάρει μια τιμή μεταξύ 0 και 255, όπου το tuple (0, 0, 0) αντιπροσωπεύει το μαύρο και το (255, 255, 255) αντιπροσωπεύει το λευκό. Το RGB θεωρείται 'πρόσθετο' color space και τα χρώματα μπορούμε να φανταστούμε ότι παράγονται από λαμπρές ποσότητες κόκκινου, μπλε και πράσινου φωτός σε μαύρο φόντο. Για παράδειγμα,

είναι σύνηθες οι έγχρωμες εικόνες να αντιπροσωπεύονται από 8-bit 3-channel images. Σε αυτήν την περίπτωση, κάθε pixel συγκεντρώνει πληροφορίες αξίας 3 bytes ένα byte το καθένα για εντάσεις κόκκινου, πράσινου και μπλε. Ρίχνοντας μια πολύ προσεκτική ματιά σε μια LCD οθόνη θα δούμε αυτόν τον τρόπο αποθήκευσης των δεδομένων.



Εικόνα 3.9-Μεγεθυμένη φωτογραφία pixels μιας LCD οθόνης

Το RGB είναι ένα από τα πέντε μεγάλα color spaces μοντέλα, καθένα από τα οποία έχει πολλά offshoots. Υπάρχουν πάρα πολλά color spaces επειδή διαφορετικά color spaces είναι χρήσιμα για διαφορετικούς σκοπούς.

Το HSV σημαίνει Hue (Απόχρωση), Saturation (Κορεσμός) και Value (Τιμή ή φωτεινότητα) και είναι ένα κυλινδρικό color space. Το HSV είναι πιο κοντά στο πως οι άνθρωποι αντιλαμβάνονται το χρώμα. Αυτό το color space περιγράφει τα χρώματα ως προς τη σκιά τους και την τιμή φωτεινότητας.

- **Απόχρωση (Hue)**

  Το Hue είναι το χρωματικό τμήμα του μοντέλου, εκφραζόμενο ως αριθμός από το 0 έως το 360

- **Κορεσμός (Saturation)**

  Το Saturation περιγράφει την ποσότητα του γκρι σε ένα συγκεκριμένο χρώμα, από 0 % έως 100 %. Η μείωση αυτού του στοιχείου προς το μηδέν εισάγει περισσότερο γκρι και παράγει ένα ξεθωριασμένο αποτέλεσμα. Μερικές φορές, το Saturation εμφανίζεται ως εύρος από 0 έως 1, όπου το 0 είναι γκρι και το 1 είναι το κύριο χρώμα.

- **Τιμή (Value)**

  Το Value λειτουργεί σε συνδυασμό με το Saturation και περιγράφει τη φωτεινότητα ή την ένταση του χρώματος, από 0 % έως 100 % όπου το 0 είναι εντελώς μαύρο και το 100 είναι το πιο φωτεινό και αποκαλύπτει το μεγαλύτερο χρώμα.

Για να μετατρέψουμε μια εικόνα από BGR σε HSV color space, χρησιμοποιούμε το **cvtColor()** function του OpenCV. Μια function που χρησιμοποιούμε και εμείς στην εργασία μας για τον ίδιο ακριβώς λόγο.

```
hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HDV)
```

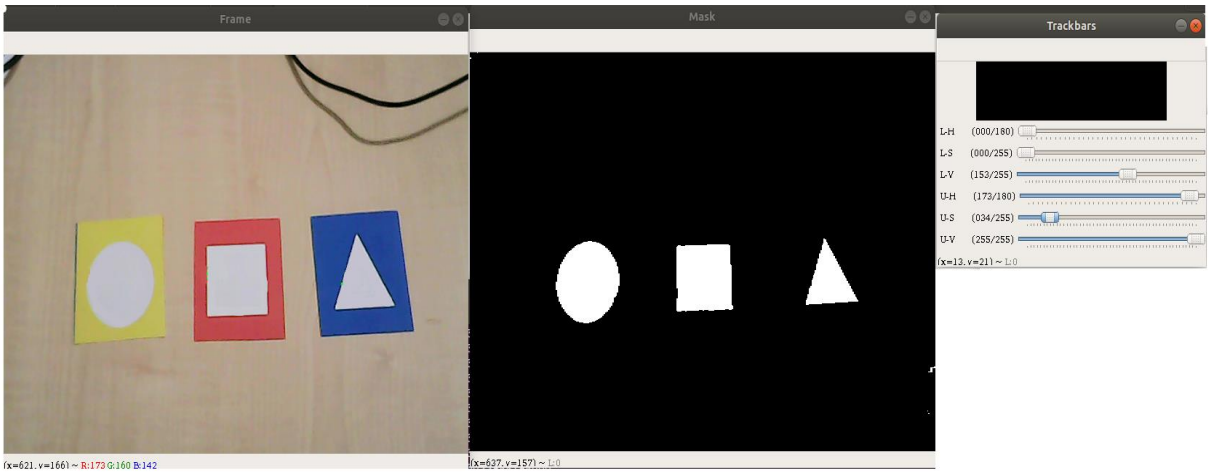
Γιατί όμως είναι απαραίτητο να γίνει αυτή η μετατροπή από το ένα color space στο άλλο. Σε περιπτώσεις όπου η περιγραφή χρώματος διαδραματίζει αναπόσπαστο ρόλο, το HSV color space προτιμάται πιο συχνά από το BGR. Το HSV περιγράφει χρώματα παρόμοια με το πως το ανθρώπινο μάτι τείνει να αντιλαμβάνεται το χρώμα όπως είπαμε, ενώ το BGR ορίζει το χρώμα σε συνδυασμό με τα πρωτεύοντα χρώματα. Το HSV περιγράφει το χρώμα χρησιμοποιώντας πιο γνωστές συγκρίσεις όπως το χρώμα, η ζωντάνια και φωτεινότητα.

Μετά την μετατροπή του color space, πρέπει να επιλέξουμε ένα color range για την εφαρμογή του Masking. Τα masks χρησιμοποιούνται για τον αποκλεισμό ορισμένων pixel από

την επεξεργασία εικόνας και όταν εφαρμόζεται, τα masked pixels δεν είναι ορατά. Το mask είναι ένα δυαδικό raster που περιέχει τιμές pixel 0 και 1. Ο λόγος εφαρμογής της μάσκας είναι η εξαίρεση ορισμένων pixels από την εικόνα, ώστε να μην επηρεάζουν το αποτέλεσμα όταν θέλουμε να κάνουμε ανάλυση ή επεξεργασία εικόνας σε συγκεκριμένα μέρη της εικόνας. Εμείς εφαρμόζουμε mask στην εργασία μας σε συγκεκριμένο HSV color range, ώστε να απομονώσουμε μόνο το λευκό χρώμα που βρίσκεται στο εσωτερικό των γεωμετρικών σχημάτων. Με αυτό το αποτέλεσμα μπορούμε μετέπειτα να εκτελέσουμε διάφορες εργασίες που μας είναι απαραίτητες για την ολοκλήρωση της εργασίας, όπως ανίχνευση τύπου σχήματος, ανίχνευση ακριβής τοποθεσίας του σχήματος στο επίπεδο και άλλα. Για την εφαρμογή του mask, χρησιμοποιούμε την cv2.inRange function του OpenCV που επιστρέφει ένα binary mask, όπου τα λευκά pixels (255) αντιπροσωπεύουν τα pixels της εικόνας που εμπίπτουν στο lower και upper HSV values που έχουμε ορίσει και τα μαύρα pixels (0) δεν εμφανίζονται.

```
mask = cv2.inRange(hsv, lower_colorvalue, upper_colorvalue)
```

Τα lower και upper HSV values, για την αποτελεσματικότερη εφαρμογή του mask μπορούμε να τα ρυθμίζουμε real time χρησιμοποιώντας την cv2.getTrackbarPos function, όπου επιστρέφει τα lower και upper values από τα trackbars που έχουμε δημιουργήσει. Στην εικόνα 3.10 βλέπουμε το αποτέλεσμα του masking.



Εικόνα 3.10-Masking

Με τον ίδιο τρόπο χωρίς τη χρήση των trackbars όμως, επιτυγχάνουμε και το Color Detection για την αναγνώριση της περιμετρικής απόχρωσης των σχημάτων μας. Εδώ ορίζουμε

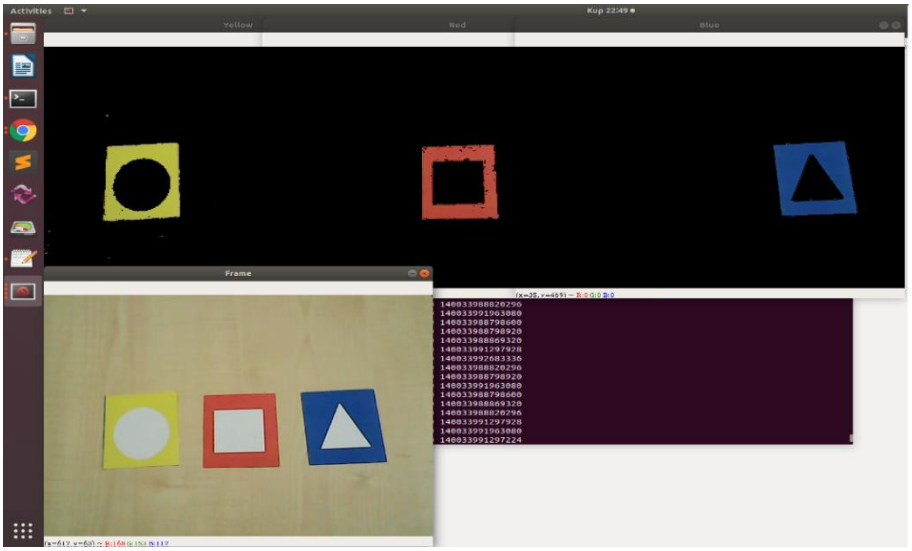
από μία σταθερή upper value και από μια σταθερή lower value για το κάθε ένα από τα τρία χρώματα που θέλουμε να αναγνωρίζει ο αλγόριθμός μας.

```
lower_red = np.array([170,50,50])
upper_red = np.array([180,255,255])

lower_blue = np.array([85,50,50])
upper_blue = np.array([132,255,255])

lower_yellow = np.array([25,50,50])
upper_yellow = np.array([32,255,255])
```

Στην εικόνα 3.10 βλέπουμε το αποτέλεσμα του Color Detection με τα lower και upper HSV values



Εικόνα 3.1117-Color

3.4.2 Εντοπισμός Περιγραμμάτων και Εντοπισμός Κέντρου (Contour Detection And Moments Detection)

Τα contours (περιγράμματα) είναι μαθηματική έννοια. Είναι μια συνδεδεμένη γραμμή input σημείων, για την οποία μια συνάρτηση έχει την ίδια output τιμή. Τώρα, αν υιοθετήσουμε αυτήν την έννοια στο Image Processing και εφαρμόσουμε τα contours στις εικόνες, στην πραγματικότητα είναι η επιλογή γραμμής pixels που έχουν την ίδια τιμή και αυτή η γραμμή περικλείει μια ομάδα τιμών pixel σχεδόν παρόμοια με τα pixel της γραμμής. Με απλά λόγια, τα contours ορίζονται ως η γραμμή που ενώνει όλα τα σημεία κατά μήκος του ορίου μιας εικόνας που έχουν την ίδια ένταση. Τα contours είναι ευρέως χρησιμοποιούμενα για shape analysis, στην εύρεση του μεγέθους του αντικειμένου ενδιαφέροντος και στο Object Detection. Το OpenCV διαθέτει την cv2.findContour function που βοηθά στην εξαγωγή των contours από την εικόνα και την cv2.drawContours για την αποτύπωση των contours στην εικόνα. Η cv2.findContours function, αποτελείται από τρία arguments.

```
contours, _ = cv2.findContours(mask, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
```

Το πρώτο argument είναι το mask που έχουμε εφαρμόσει και για το οποίο μιλήσαμε στο προηγούμενο υποκεφάλαιο, το δεύτερο είναι το contour retrieval mode και το τρίτο είναι το contour approximation method.

- **Contour Retrieval Mode:** Αποτελείται από τέσσερα βασικά contour hierarchy modes

1. **CV_RETR_EXTERNAL**

   Επιστρέφει μόνο το εξωτερικό contour

2. **CV_RETR_LIST**

   Επιστρέφει όλα τα contours χωρίς να εφαρμόζει κάποια ιεραρχική σχέση μεταξύ τους

3. **CV_RETR_CCOMP**

   Επιστρέφει όλα τα contours και τα οργανώνει σε ιεραρχία δύο επιπέδων. Στο ανώτερο επίπεδο, υπάρχουν εξωτερικά όρια των στοιχείων. Στο δεύτερο επίπεδο, υπάρχουν όρια των οπών των στοιχείων. Εάν υπάρχει ένα άλλο contour μέσα σε μια οπή ενός συνδεδεμένου στοιχείου, εξακολουθεί να βρίσκεται στο ανώτερο επίπεδο.

4. **CV_RETR_TREE**

   Επιστρέφει όλα τα contours και ανακατασκευάζει μια πλήρη ιεραρχία ένθετων contours

- **Contour Approximation Method:** Υπάρχουν τρείς βασικές μέθοδοι αποθήκευσης των (x,y) συντεταγμένων των contours

1. **CV_CHAIN_APPROX_NONE**

   Αποθηκεύει όλα τα contours point

2. **CV_CHAIN_APPROX_SIMPLE**

   Συμπιέζει τα οριζόντια, κάθετα και διαγώνια τμήματα και αφήνει μόνο τα τελικά σημεία τους. Για παράδειγμα, ένα ορθογώνιο contour κωδικοποιείται με 4 σημεία

3. **CV_CHAIN_APPROX_TC89_L1,CV_CHAIN_APPROX_TC89_KCOS**

   Εφαρμόζει ένα από τα flavors του Teh-Chin chain approximation algorithm

Τα moment είναι ένα ποσοτικό μέτρο, που χρησιμοποιείται ευρέως στη μηχανική και την στατιστική, για να περιγράψει τη χωρική κατανομή των συνόλων σημείων. Με τους περισσότερους απλοϊκούς όρους, τα moments είναι σύνολο βαθμών που παρέχουν ένα

συγκεντρωτικό μέτρο ενός συνόλου διανυσμάτων. Στο Image Processing, στο Computer Vision και σε συναφή πεδία, ένα image moment είναι ένας συγκεκριμένος μέσος όρος των εντάσεων των pixels της εικόνας, που συνήθως επιλέγονται για να έχουν κάποια ελκυστική ιδιότητα ή ερμηνεία. Τα image moments είναι χρήσιμα για την περιγραφή αντικειμένων μετά το segmentation. Οι απλές ιδιότητες των στοιχείων που εντοπίζονται μέσω image moments περιλαμβάνουν, την περιοχή που βρίσκονται (ή τη συνολική ένταση), το centroid (κεντροειδές) της και πληροφορίες σχετικά με τον προσανατολισμό τους. Το OpenCV διαθέτει την cv2.Momets function για την εύρεση του κέντρου βάρους των στοιχείων μιας εικόνας και διατυπώνεται ως εξής

```
M = cv2.moments(cnt)
cX = int(M["m10"] / M["m00"])
cY = int(M["m01"] / M["m00"])
```

Τα κεντρικά moments υπολογίζονται από τον τύπο

$$\mathtt{mu}_{ji} = \sum_{x,y} \left(\mathtt{array}(x,y) \cdot (x - \bar{x})^j \cdot (y - \bar{y})^i \right)$$

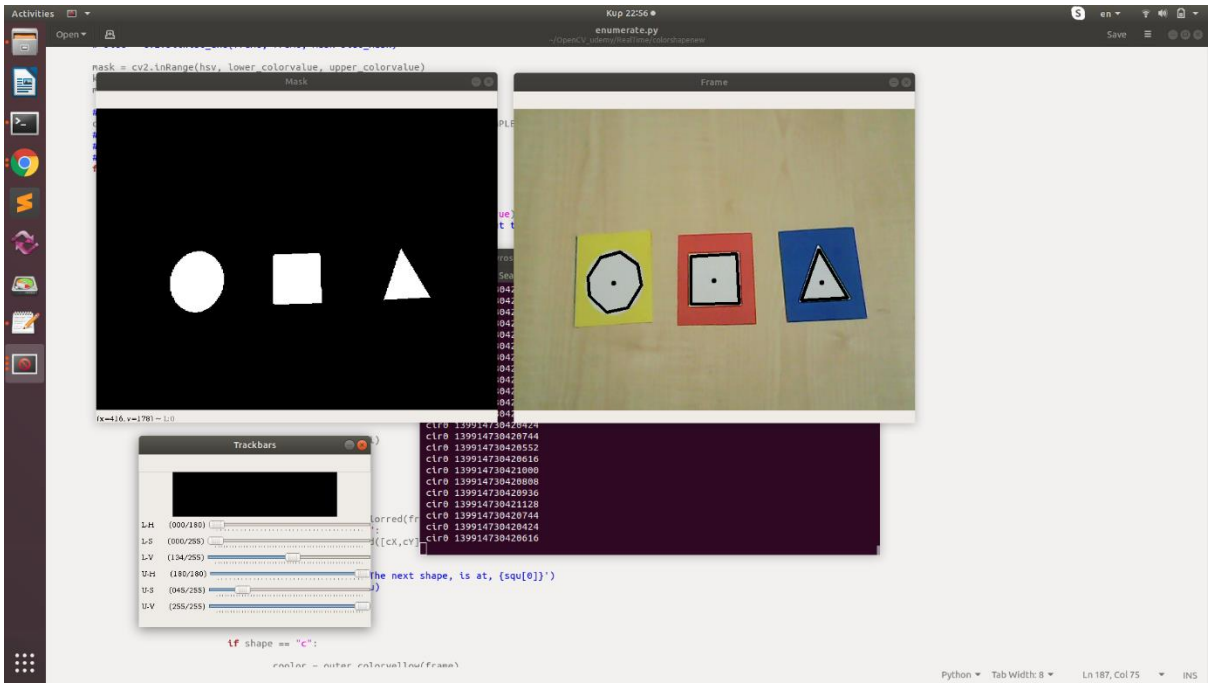
Όπου τα $(\bar{x},\bar{y})$ είναι το κέντρο βάρους

$$\bar{x} = \frac{\mathtt{m}_{10}}{\mathtt{m}_{00}},\ \bar{y} = \frac{\mathtt{m}_{01}}{\mathtt{m}_{00}}$$

Τα moments ενός contour υπολογίζονται χρησιμοποιώντας τη φόρμουλα του Green, δηλαδή για περιοχές που περικλείονται από καμπύλες, χωρίς γεωμετρικό σχήμα. Έτσι, λόγω της περιορισμένης raster ανάλυσης, τα moments που υπολογίζονται για ένα contour είναι ελαφρώς διαφορετικά από τα moments που υπολογίζονται για το ίδιο rasterized contour. Δεδομένου ότι τα contour moments υπολογίζονται χρησιμοποιώντας τον τύπο Green, μπορεί να έχουν φαινομενικά περίεργα αποτελέσματα για contours με ακανόνιστο σχήμα.

Εμείς χρησιμοποιούμε τα συγκεκριμένη functions στην εργασία μας, για τον σχεδιασμό των contours των σχημάτων και την επιστροφή της εξαγωγής του κέντρου βάρους των contours, για την ακριβή τοποθεσία των σχημάτων μας μέσα στο frame της κάμερας, ώστε να

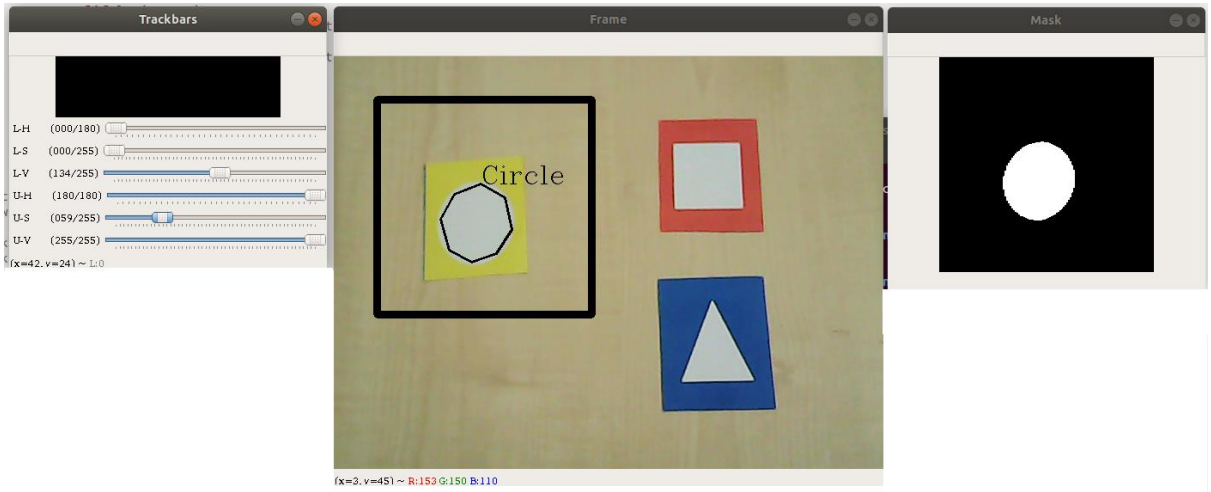
γνωρίζουμε που να τοποθετήσουμε το ταυτοποιημένο από το ROI σχήμα, στο πρώτο διαθέσιμο κενό σχήμα ιδίων χαρακτηριστικών. Στην παρακάτω εικόνα (Εικόνα 3.11) βλέπουμε τα αποτελέσματα των παραπάνω functions



Εικόνα 3.1218-Contours και Moments

3.4.3 Περιοχή Ενδιαφέροντος (Region of Interest)

Μερικές φορές, ένα processing function πρέπει να εφαρμόζεται μόνο σε ένα τμήμα μιας εικόνας. Το OpenCV ενσωματώνει έναν κομψό και απλό μηχανισμό για τον καθορισμό μιας υποπεριοχής σε μια εικόνα και τον χειρισμό ως κανονική εικόνα και το ονομάζει Range of Interest (RoI). Στην περίπτωσή μας, ένα ROI καθορίζεται δημιουργώντας ένα τετράφωνο subsection του frame της κάμερας και εφαρμόζοντας δικές μας functions μέσα σε αυτό το subsection. Συγκεκριμένα χρησιμοποιούμε την cv2.rectangle function για την δημιουργία του τετραγώνου και έπειτα εφαρμόζουμε τις απαραίτητες λειτουργείες σε αυτή την περιοχή του frame. Σκοπός του RoI στην εργασία μας όπως έχουμε πει, είναι η αναγνώριση των σχημάτων που θα εισάγουμε στην συγκεκριμένη περιοχή από τον αλγόριθμό μας, ώστε να μας επισημάνει την διαθέσιμη κενή θέση του σχήματος με τα ίδια χαρακτηριστικά. Στην εικόνα 3.12 βλέπουμε την εφαρμογή του RoI στο project μας, μαζί με contour και shape detection.



Εικόνα 3.13-Region of Interest

3.5 Δυσκολίες Εφαρμογής

TensorFlow

Τα τελευταία χρόνια, έχει σημειωθεί σημαντική πρόοδος στον τομέα του Machine Learning. Μεγάλο μέρος αυτής της προόδου μπορεί να αποδοθεί στην αυξανόμενη χρήση των GPU's για την επιτάχυνση της εκπαίδευσης των Machine Learning μοντέλων. Συγκεκριμένα, η επιπλέον υπολογιστική ισχύς οδήγησε στην αύξηση της δημοσιότητας του Deep Learning, τη χρήση δηλαδή σύνθετων, πολυεπίπεδων νευρωνικών δικτύων για τη δημιουργία μοντέλων ικανών να εντοπίζουν χαρακτηριστικά από μεγάλες ποσότητες μη επισημασμένων εκπαιδευτικών δεδομένων. Οι GPU's είναι κατάλληλες για το Machine Learning και το Deep Learning, επειδή ο τύπος των υπολογισμών που είχαν σχεδιαστεί για την εκτελούν τυχαίνει να είναι ο ίδιος με αυτών που συναντώνται στο Machine Learning και στο Deep Learning. Οι εικόνες, τα βίντεο και άλλα γραφικά παρουσιάζονται ως πίνακες (matricies), έτσι ώστε όταν εκτελείτε οποιαδήποτε λειτουργία, όπως zoom in εφέ ή περιστροφή κάμερας, το μόνο που κάνουμε είναι να εφαρμόζουμε κάποια μαθηματική μετατροπή σε matrix.

Στην πράξη, αυτό σημαίνει ότι οι GPU's, σε σύγκριση με τις CPU's, είναι πιο εξειδικευμένες στην εκτέλεση matrix λειτουργιών και πολλών άλλων τύπων προηγμένων

μαθηματικών μετασχηματισμών. Αυτό κάνει τους Machine Learning και Deep Learning αλγορίθμους να εκτελούνται πολλές φορές γρηγορότερα σε μια GPU, παρά σε CPU. Οι χρόνοι εκπαίδευσης μπορούν συχνά να μειωθούν από ημέρες σε μόλις ώρες σε μεγάλα project.

Έτσι λοιπόν και το TensorFlow, ως ένα Machine Learning εργαλείο, για την αποδοτικότερη και γρηγορότερη εκπαίδευση των εξαγόμενων μοντέλων του λειτουργεί συνεργατικά με τις NVIDIA GPU's, χωρίς αυτό να σημαίνει πως χωρίς αυτές δεν λειτουργεί. Στην δική μας περίπτωση στην οποία δεν έγινε χρήση GPU, η εκπαίδευση διήρκησε αρκετές ώρες και με cores της CPU να δουλεύουν σχεδόν στο 100 % των δυνατοτήτων τους, με αποτέλεσμα η θερμοκρασία στους πυρήνες της να κυμαίνεται μονίμως μεταξύ 87 και 95 βαθμών κελσίου. Πιο συγκεκριμένα, κάθε training step διαρκούσε μεταξύ 4 έως 8 δευτερόλεπτα και για να εκτελεστούν συνολικά 12480 βήματα ώστε να φτάσει το error του TotalLoss graph κάτω από το 0,05 και να λάβει τέλος η εκπαίδευση, πέρασαν 15 ώρες.

Για να γίνει μία μικρή σύγκριση αριθμών, ένα σχετικά πανομοιότυπο project εκπαίδευσης classifier σε CNN με NVIDIA GeForce GTX 1060 6Gb GPU και 16Gb RAM, χρειάστηκε συνολικά μόλις τρείς ώρες για να ολοκληρώσει την εκπαίδευση. Συγκεκριμένα είχε έξι objects για classification έναντι των τριών δικών μας, 311 φωτογραφίες έναντι των 150 δικών μας και εκτέλεσε συνολικά 56160 βήματα με το κάθε βήμα να διαρκεί μεταξύ 0.190 έως 0.230 δευτερόλεπτα. Να αναφερθεί πως ο δικός μας υπολογιστής έκανε την εκπαίδευση με CPU Intel Core i5-8250U 1.60GHz και 8Gb RAM

OpenCV

Όσων αφορά τις δυσκολίες υλοποίησης στην προσέγγιση του OpenCV, αυτό που αντιμετωπίσαμε είναι δυσκολία εφαρμογής της shape detection τεχνικής μας σε γεωμετρικά σχήματα με στρογγυλεμένες άκρες. Πιο συγκεκριμένα, στην αρχή του project μας είχαμε χρησιμοποιήσει γεωμετρικά σχήματα με στρογγυλεμένες άκρες και όχι γωνίες. Αυτό είχε σαν αποτέλεσμα την μη σωστή εφαρμογή των contours πάνω στα σχήματα και εν συνεχεία τη μη σωστή επιστροφή του αποτελέσματος του shape detection από το αλγόριθμο. Το shape detection στο OpenCV το έχουμε ορίσει να γίνεται μέσω της καταμέτρησης των πλευρών των contours, έτσι το τετράγωνο θα αποτελείται από τέσσερις πλευρές, το τρίγωνο από τρείς και ο κύκλος από 7 έως 20 πλευρές. Στο OpenCV μπορούμε να εφαρμόσουμε contours ακόμα και

στο πιο ιδιαίτερο σχήμα με την cv2.approxPolyDP function η οποία εφαρμόζει προσεγγιστικά το contour πάνω σε ένα σχήμα με μικρότερο αριθμό κορυφών ανάλογα με την ακρίβεια που καθορίζουμε.

```
epsilon = 0.1*cv2.arcLength(cnt, True)
approx. = cv2.approxPolyDP(cnt, epsilon, True)
```

Μπορούμε να χρησιμοποιήσουμε αυτήν την function για να προσεγγίσουμε ένα σχήμα. Στη functionαυτή, η δεύτερη παράμετρος ονομάζεται epsilon, η οποία είναι η μέγιστη απόσταση από το contour έως το κατά προσέγγιση contour. Είναι μια παράμετρος ακρίβειας. Χρειάζεται μια συνετή επιλογή του epsilon για να έχουμε τη σωστή έξοδο. Έτσι με αυτό τον τρόπο αυτό μπορούμε να εφαρμόσουμε contours σε σχεδόν οποιοδήποτε σχήμα.

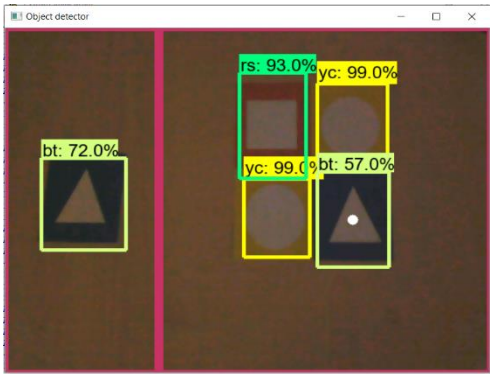
Κεφάλαιο 4-Ανάλυση Απόδοσης

4.1 Φωτισμός και Υπόβαθρο (Lighting And Background)

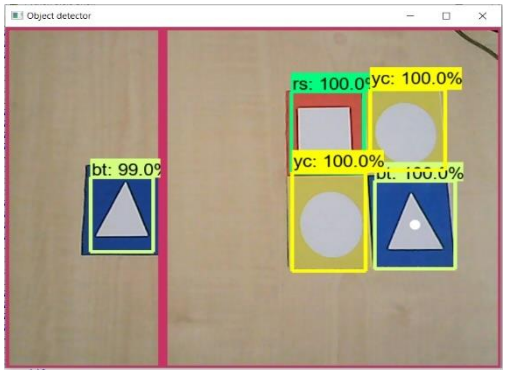
TensorFlow

Το θέμα της εναλλαγής του φωτισμού και των πιθανών διαφορετικών υποβάθρων, δεν δείχνουν να επηρεάζουν ιδιαίτερα την απόδοση του TensorFlow στην εφαρμογή του στο demo. Αυτό γίνεται διότι οι εικόνες που δόθηκαν σαν input training data, λήφθηκαν σε συνθήκες διαφορετικών φωτισμών και διαφορετικών υποβάθρων, με απώτερο σκοπό την αναγνώριση των στοιχείων μας σε ένα σχετικά ευρύ φάσμα υποβάθρων και φωτισμών. Αυτό δεν σημαίνει βέβαια πως εάν εφαρμόσουμε στο demo ακραίες συνθήκες φωτισμού, το TensorFlow θα ανταποκριθεί άψογα, διότι δεν εκπαιδεύτηκε το μοντέλο πάνω σε τέτοιες συνθήκες.

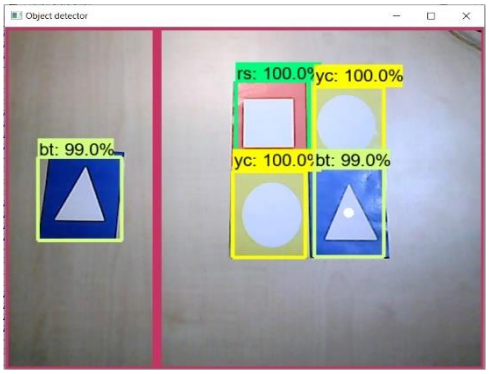
Παρακάτω θα παραθέσουμε μερικές ενδεικτικές φωτογραφίες λειτουργίας του TensorFlow υπό συνθήκες διαφορετικών φωτισμών αλλά και αλληλεπικάλυψης (overlapping) των σχημάτων.



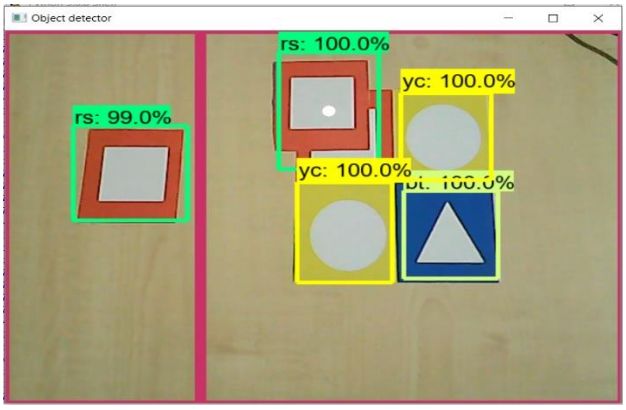
Εικόνα 4.1- Χαμηλός Φωτισμός



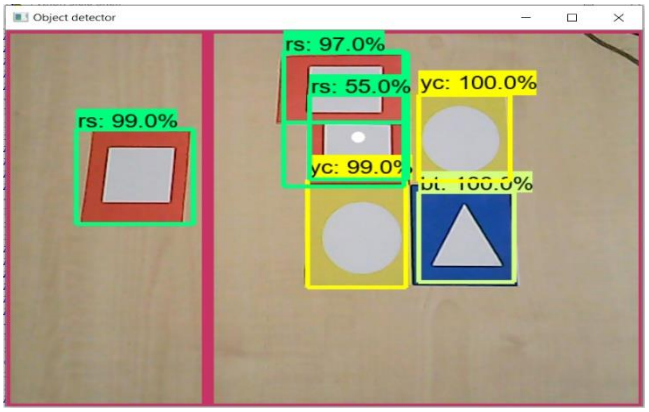
Εικόνα 4.2- Μέτριος Φωτισμός



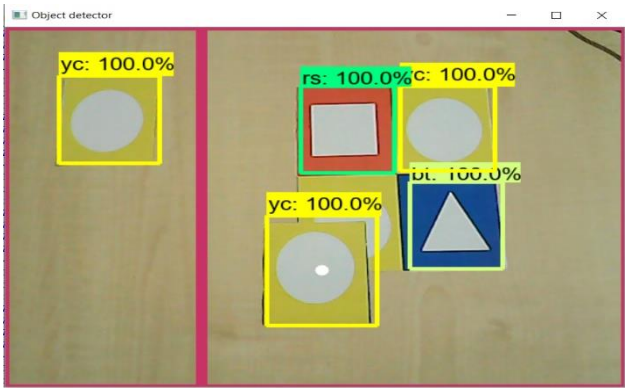
Εικόνα 4.3- Υψηλός Φωτισμός



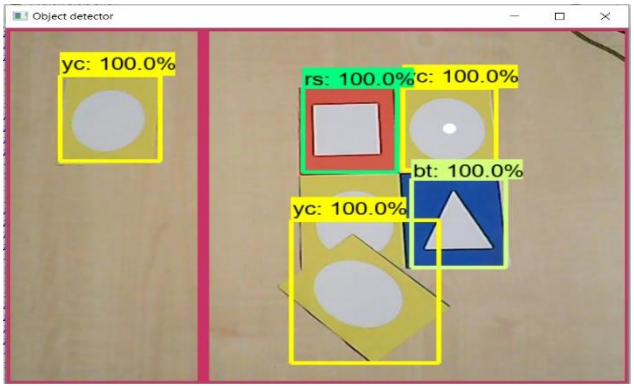
Εικόνα 4.4- Overlapping τετράγωνο



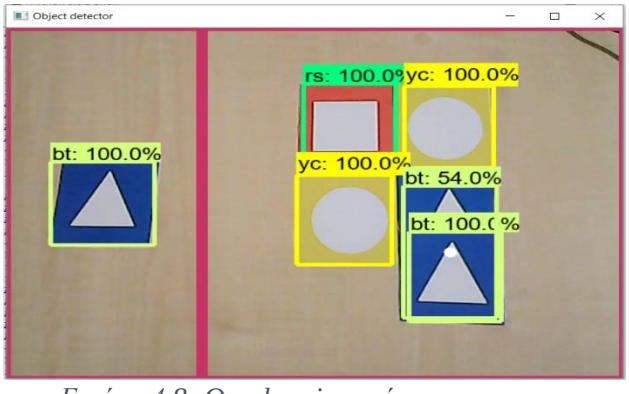
Εικόνα 4.5- Overlapping τετράγωνο



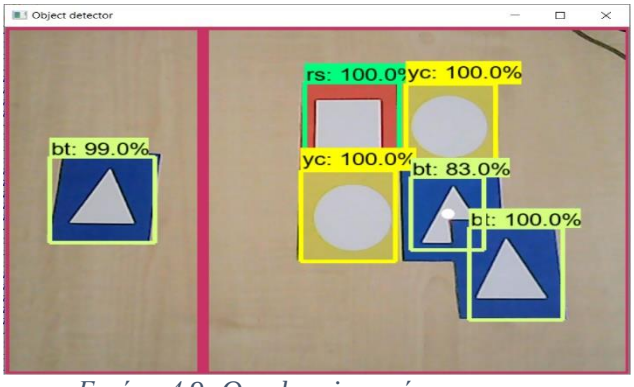
Εικόνα 4.6- Overlapping κύκλος



Εικόνα 4.7- Overlapping κύκλος



Εικόνα 4.8- Overlapping τρίγωνο



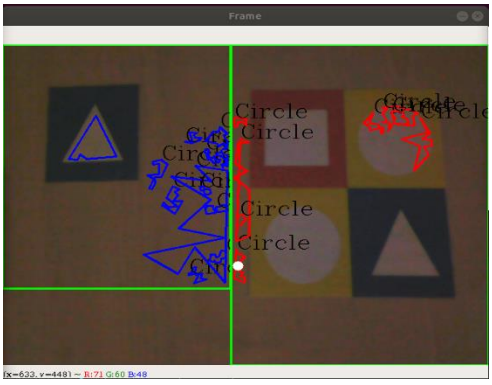
Εικόνα 4.9- Overlapping τρίγωνο

OpenCV

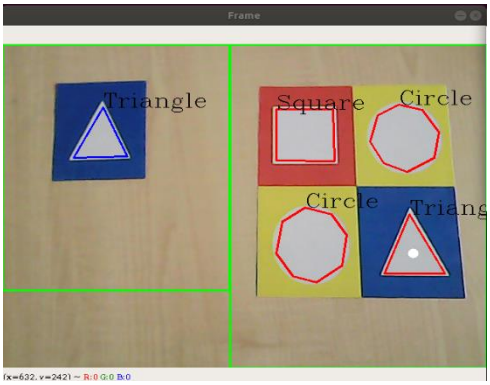
Το θέμα της εναλλαγής του φωτισμού και των πιθανών διαφορετικών υποβάθρων, φαίνεται να επηρεάζουν αρκετά την απόδοση του αλγορίθμου στην εφαρμογή του στο demo. Πιο συγκεκριμένα, υπό συνθήκες σταθερού φωτισμού και ενώ έχουμε κάνει τις απαραίτητες ρυθμίσεις στα lower και higher HSV values από τα trackbars, ο αλγόριθμος φαίνεται να λειτουργεί κανονικά, χωρίς προβλήματα. Όταν όμως οι συνθήκες φωτισμού τροποποιηθούν έστω και στο ελάχιστο, θα πρέπει να γίνει εκ νέου προσαρμογή των lower και higher HSV values από τα trackbars διότι άλλαξαν οι συνθήκες του περιβάλλοντος που λαμβάνει χώρα το πείραμα. Αυτό έχει σαν αποτέλεσμα να διαστρεβλωθεί το σωστό masking, να μην αποδίδονται τα σωστά contours και εν τέλει να μην γίνεται σωστό shape detection.

Τώρα αναφορικά με το υπόβαθρο, φαίνεται να παίζει και αυτό σημαντικό ρόλο στην απόδοση του αλγορίθμου. Συγκεκριμένα για το δικό ας demo, δεν πρέπει να υπάρχουν λευκά στοιχεία στο υπόβαθρο, διότι θα δημιουργηθούν shape detection προβλήματα μετά την εφαρμογή του masking. Αυτό θα συμβεί διότι το εσωτερικό των σχημάτων μας είναι λευκό και επειδή το masking εφαρμόζεται για να απομονώσει μόνο το λευκό χρώμα και να γίνει shape detection από τα contours, εάν μείνουν και άλλα λευκά στοιχεία στο mask, ο αλγόριθμος θα εφαρμόσει και σε αυτά contours και συνάμα shape detection. Αυτή η περίπτωση θα μας επέστρεφε μη σωστά αποτελέσματα για στοιχεία που δεν θα έπρεπε να εμπλέκονται στο demo μας.

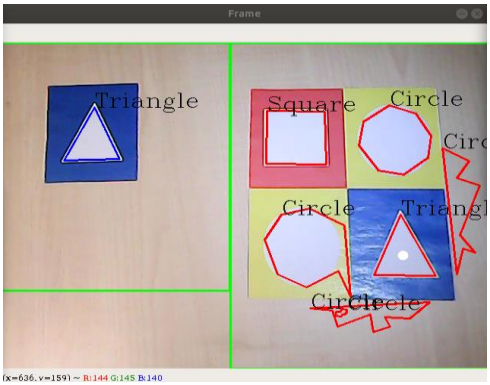
Παρακάτω θα παραθέσουμε μερικές ενδεικτικές φωτογραφίες λειτουργίας του αλγορίθμου μας στο OpenCV, υπό συνθήκες διαφορετικών φωτισμών αλλά και overlapping των σχημάτων.



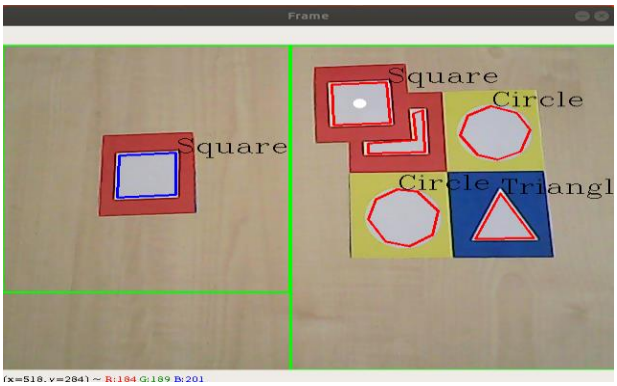
Εικόνα 4.10- Χαμηλός Φωτισμός



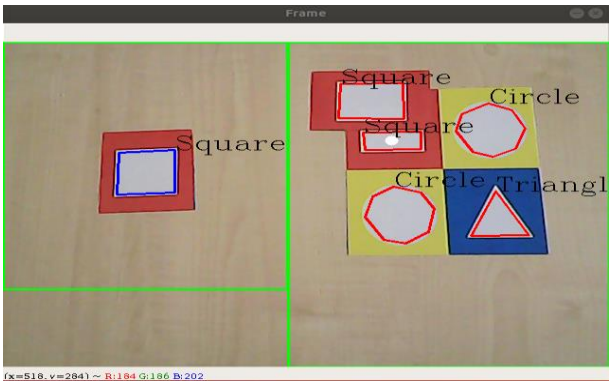
Εικόνα 4.11- Μέτριος Φωτισμός



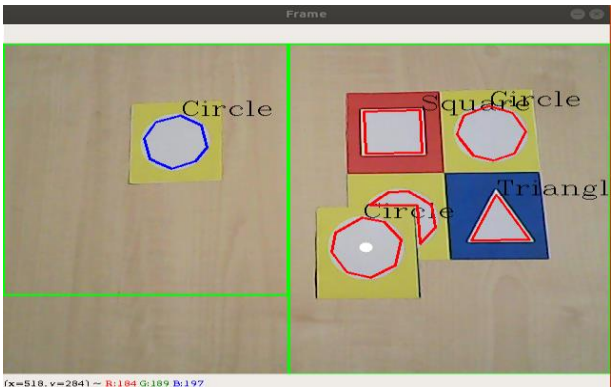
Εικόνα 4.12- Υψηλός Φωτισμός



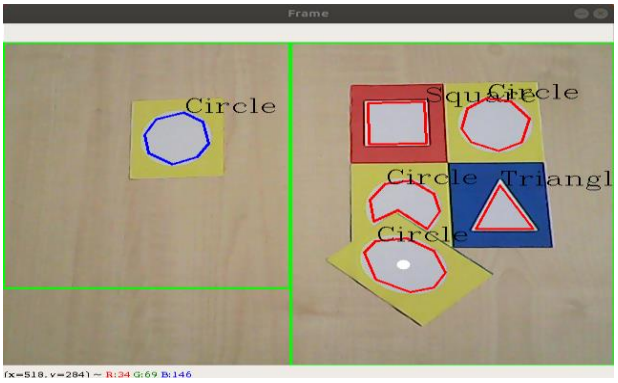
Εικόνα 4.13- Overlapping τετράγωνο



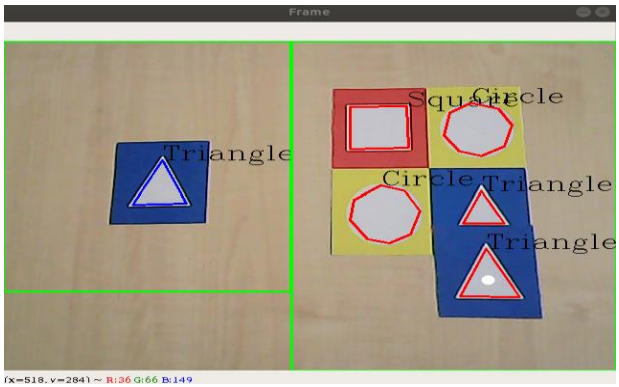
Εικόνα 4.14- Overlapping τετράγωνο



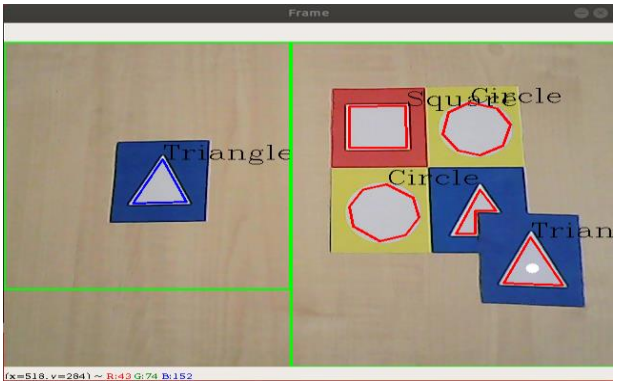
Εικόνα 4.15- Overlapping κύκλος



Εικόνα 4.16- Overlapping κύκλος



Εικόνα 4.17- Overlapping τρίγωνο



Εικόνα 4.18- Overlapping τρίγωνο

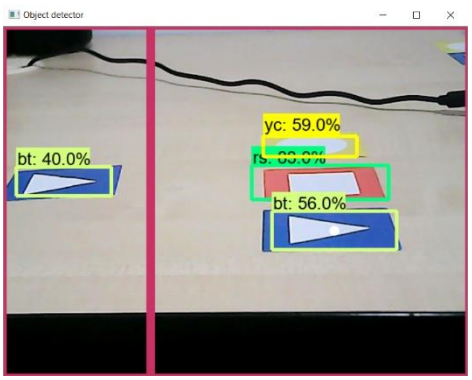
4.2 Γωνία Κάμερας και Απόσταση (Camera Angle And Distance)

TensorFlow

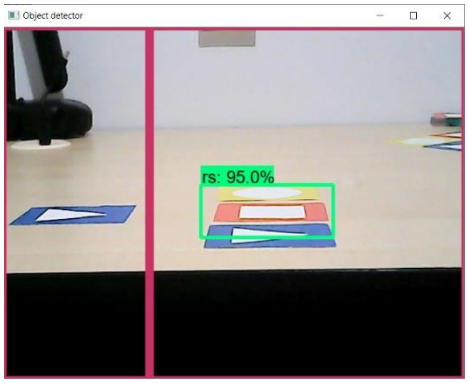
Η γωνία λήψης εικόνας της κάμερας αλλά και η απόστασή της από το επίπεδο που βρίσκονται τα σχήματα, φαίνεται να επηρεάζει το αποτέλεσμα που φέρει το TensorFlow στο demo όσων αφορά ‘ακραίες’ γωνίες και αποστάσεις λήψεις. Αυτό συμβαίνει ξανά, επειδή τα input training data, οι φωτογραφίες δηλαδή που εισήχθησαν για την εκπαίδευση του μοντέλου, ναι μεν λήφθηκαν από πολλαπλές γωνίες και από πολλαπλές αποστάσεις ώστε να καλύψουμε ξανά ένα σχετικά μεγάλο φάσμα τοποθεσιών για την τοποθέτηση της κάμερας, αλλά λήφθηκαν με γνώμονα μια σχετική απόσταση κάμερας και αντικειμένων. Αυτό όμως δεν σημαίνει πως στις περιπτώσεις που τοποθετούμε την κάμερα σε μια σχετικά κοντινή ή μακρινή απόσταση και υπό σχετικά μικρές γωνίες λήψεις, το TensorFlow δεν θα λειτουργεί αψεγάδιαστα. Και αυτό θα συμβαίνει πολύ απλά επειδή το μοντέλο εκπαιδεύτηκε να πραγματοποιεί αναγνωρίσεις υπό τέτοιες συνθήκες.

Παρατηρήσαμε πως το TensorFlow ξεκινά να μην λειτουργεί αποδοτικά από τις 20 μοίρες και κάτω, ξεκινώντας να μην αναγνωρίζει τα σχήματα και να μην μπορεί να εναποθέσει σωστά τα bounding boxes στα σχήματα που καταφέρνει έστω και σε μικρό ποσοστό να αναγνωρίζει. Παρατηρήσαμε επίσης πως το TensorFlow ξεκινά να μην λειτουργεί αποδοτικά από την απόσταση των 160 εκατοστών και άνω, ξεκινώντας και σε αυτή την περίπτωση να μην μπορεί να κάνει σωστή αναγνώριση των σχημάτων.

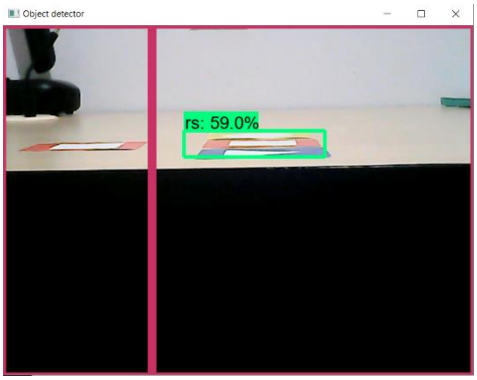
Παρακάτω θα παραθέσουμε μερικές ενδεικτικές φωτογραφίες λειτουργίας του TensorFlow υπό συνθήκες λήψεις εικόνας 35 μοιρών και μικρότερων, αλλά και απόσταση μεταξύ κάμερας και σχημάτων, 150 και 200 εκατοστών.



Εικόνα 4.19- 35 μοίρες γωνία λήψης

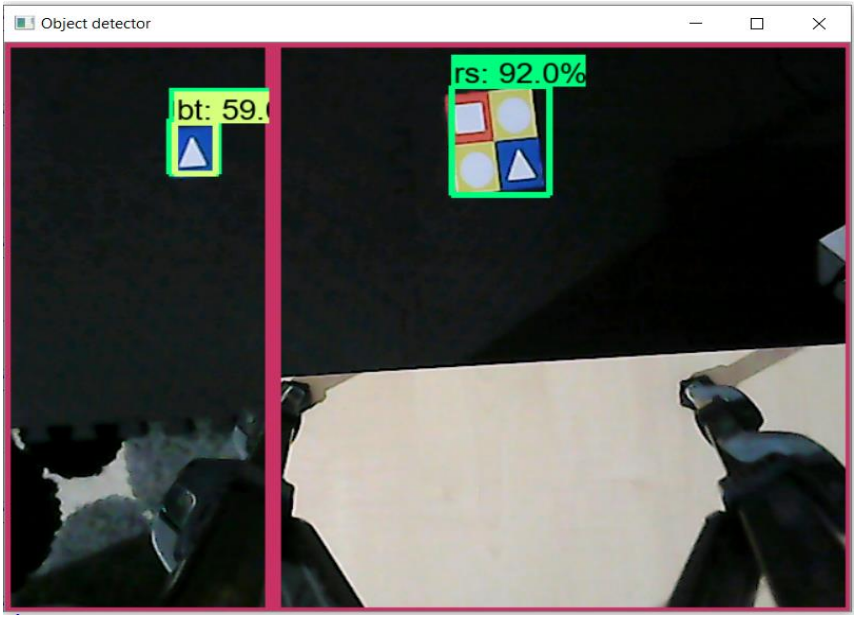


Εικόνα 4.20- 20 μοίρες γωνία λήψης



Εικόνα 4.21- 15 μοίρες γωνία λήψης

Εικόνα 4.22- 150 εκατοστά απόσταση λήψης



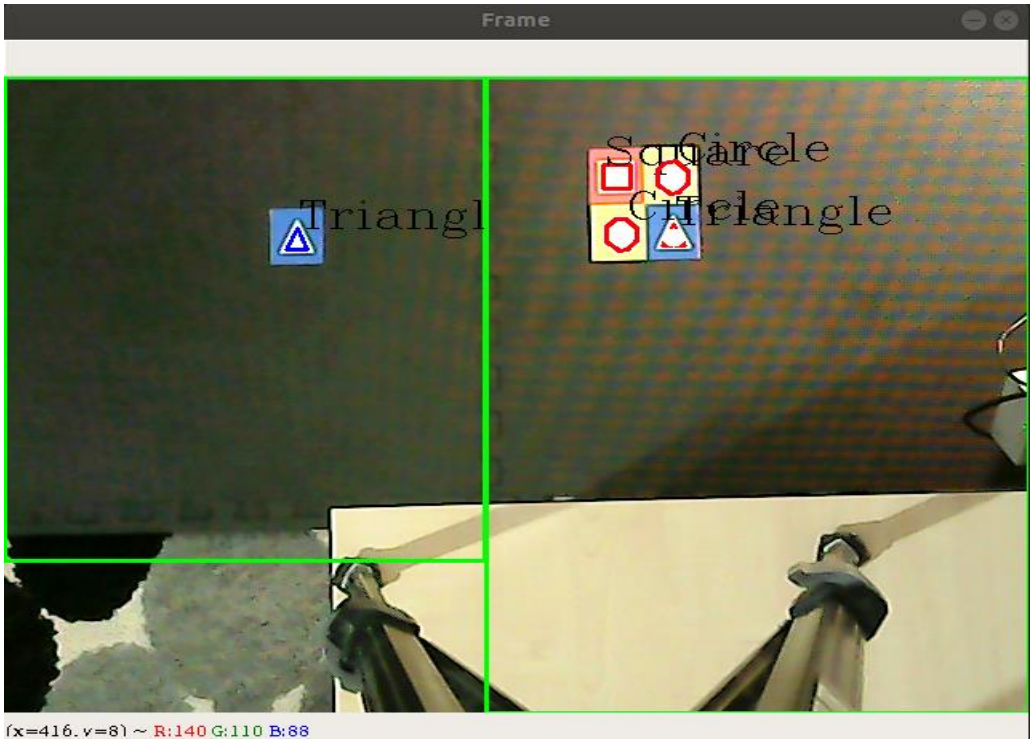
Εικόνα 4.23- 200 εκατοστά απόσταση λήψης

OpenCV

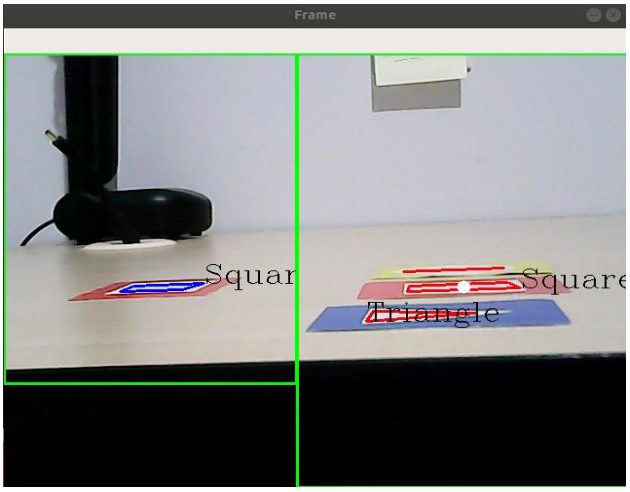
Στο OpenCV, η απόσταση της κάμερας από το επίπεδο που βρίσκονται τα σχήματα επηρεάζει τις cv2.findContours και cv2.drawContours functions. Την πρώτη, στην ευκολότερη και αποδοτικότερη εύρεση των contours και την δεύτερη στην παράμετρο του μεγέθους του contour. Όσο απομακρύνεται η κάμερα και το σχήμα μας όλο και μικραίνει, η παράμετρος μεγέθους παραμένει σταθερή και αν είναι μεγάλη η απόσταση, μπορεί να διαστρεβλώσει το αποτέλεσμα του shape detection και συνάμα την απόδοση του αλγορίθμου. Για να αποφευχθεί αυτό θα πρέπει είτε η παράμετρος της function να οριστεί σε μικρή κλίμακα, να είναι λιγότερο ευδιάκριτη δηλαδή στον χρήστη, είτε να μην τοποθετηθεί τόσο μακριά η κάμερα ώστε να επιφέρει τέτοια προβλήματα αναγνώρισης. Εμείς έχουμε εξαρχής θέσει την παράμετρο αυτή σε μικρή κλίμακα, οπότε δεν δημιουργείτε κάποιο πρόβλημα αναγνώρισης, ακόμη και στην απόσταση των 200 εκατοστών.

Όσων αφορά την γωνία λήψης εικόνας της κάμερας, ισχύουν τα ίδια προβλήματα με τις cv2.findContours και cv2.drawContours functions. Όσο η γωνία λήψης βρίσκεται μεταξύ 15 και 165 μοίρες, δεν φαίνεται να υπάρχει κάποιο θέμα στα contour και shape detection πέραν της δυσκολίας αναγνώρισης του κύκλου, όσο κατεβαίνουμε όμως σε μοίρες και πλησιάζει η κάμερα το επίπεδο, το contour detection ξεκινά να μην αποδίδει σωστά με συνέπεια και το μη σωστό shape detection οπότε και τη μη ορθή απόδοση του αλγορίθμου.

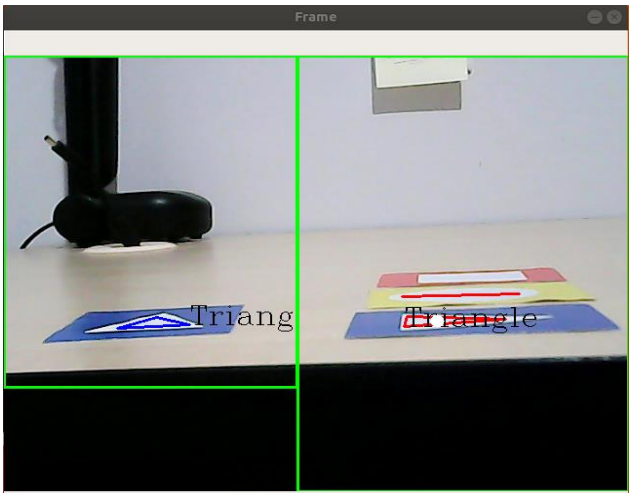
Παρακάτω θα παραθέσουμε μερικές ενδεικτικές φωτογραφίες λειτουργίας του αλγορίθμου μας στο OpenCV υπό συνθήκες λήψεις εικόνας 15 μοιρών και μικρότερων, αλλά και απόσταση 200 εκατοστών μεταξύ κάμερας και σχημάτων.



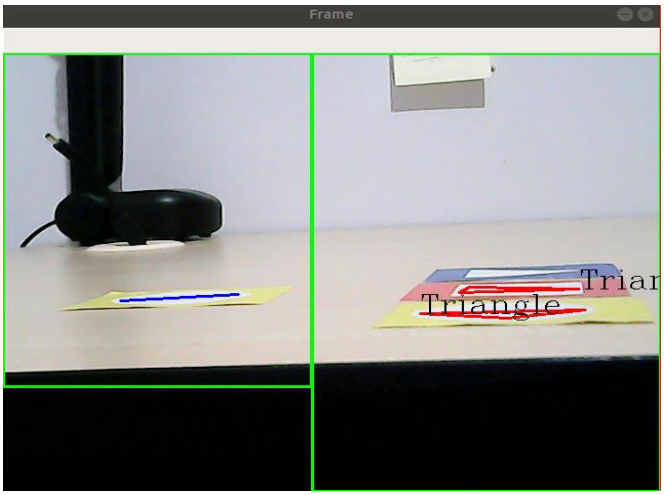
Εικόνα 4.24- 200 εκατοστά απόσταση λήψης



Εικόνα 4.25- 15 μοίρες γωνία λήψης τετράγωνο



Εικόνα 4.26- 15 μοίρες γωνία λήψης τρίγωνο



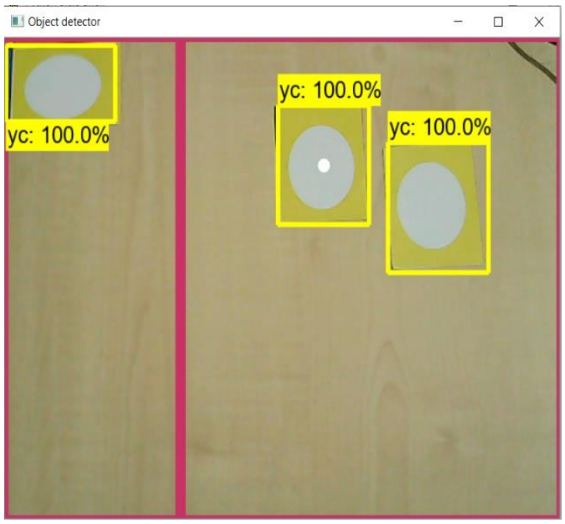
Εικόνα 4.27- 15 μοίρες γωνία λήψης κύκλος

4.3 Ακολουθία Εντοπισμού Αντικειμένου (Object Detection Sequence)

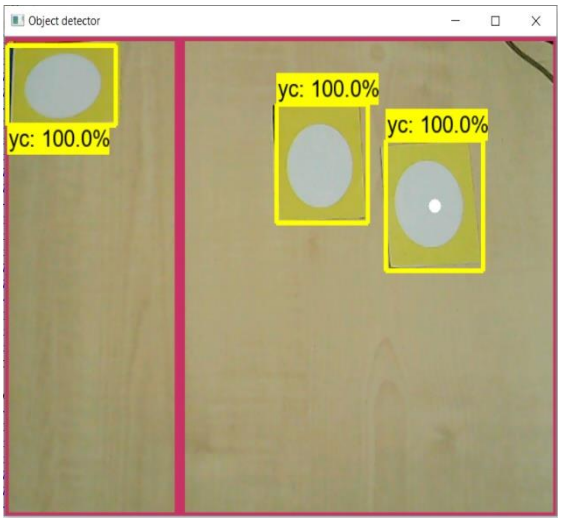
TensorFlow

Στο TensorFlow, τα detected objects είναι ταξινομημένα βάση την ποσοστιαία πρόβλεψη που τους προσδίδεται που τους αποδίδεται από το TensorFlow. Αυτό σημαίνει πως η object detection ακολουθία δεν βασίζεται στην χωρική τοποθέτηση των γεωμετρικών σχημάτων στο επίπεδο, παρά μόνο στην ποσοστιαία πρόβλεψη, μια παράμετρος που συνεχώς μεταβάλλεται. Μπορεί φαινομενικά τις περισσότερες στιγμές όλα τα ποσοστά πρόβλεψης να δείχνουν 100% αλλά στην πραγματικότητα το ποσοστό είναι 99.9999999 με πολλά από τα δεκαδικά ψηφία να αλλάζουν συνεχώς. Έτσι τη μια στιγμή μπορεί να είναι πρώτο στη στοίβα της αναγνώρισης ένα στοιχείο και την επόμενη να είναι ένα άλλο. Αυτό μπορεί να επηρεάσει ένα project, στην περίπτωση που το ζητούμενο θα ήταν να γεμίσει κενές θέσεις με τη σειρά.

Ακολουθούν δύο φωτογραφίες απεικόνισης της αναγνώρισης του πρώτου αντικειμένου από το TensorFlow.



Εικόνα 4.28- Αναγνώριση αριστερού κύκλου σαν πρώτο στοιχείο

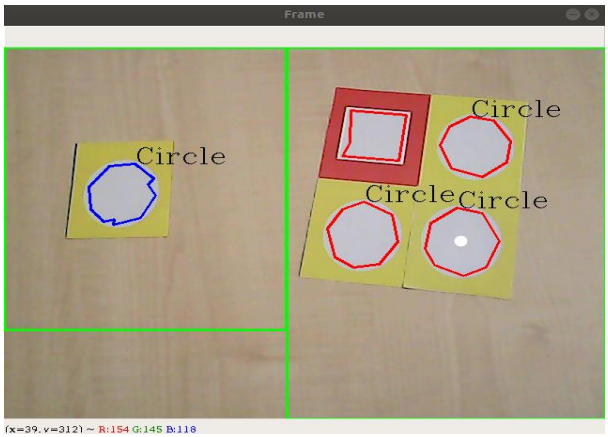


Εικόνα 4.29- Αναγνώριση δεξιού κύκλου σαν πρώτο στοιχείο

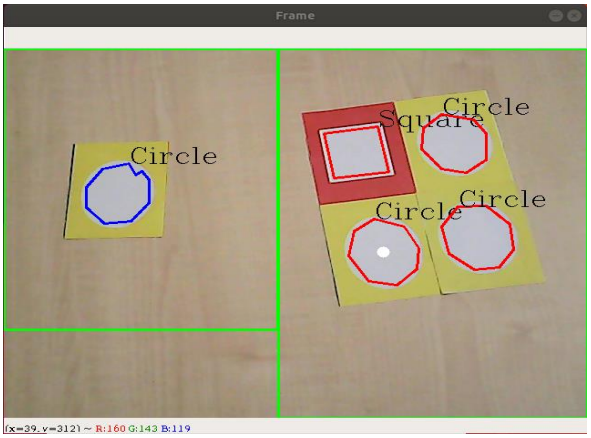
OpenCV

Στο OpenCV, τα detected objects ταξινομούνται αποκλειστικά βάση της χωρικής τοποθέτησης στο επίπεδο, πάντα το χαμηλότερο στο frame αντικείμενο αναγνωρίζεται ως πρώτο. Ακόμα και αν φαινομενικά τα σχήματα βρίσκονται στην ίδια ευθεία, θα καταχωρηθεί ως πρώτο στοιχείο αυτό του οποίου τα pixels του contour, είναι πιο χαμηλά στο frame. Σε αυτή την περίπτωση θα μπορούσαμε πολύ εύκολα να γεμίσουμε τις κενές θέσεις με τη σειρά αν αυτό ήταν το ζητούμενο ενός project.

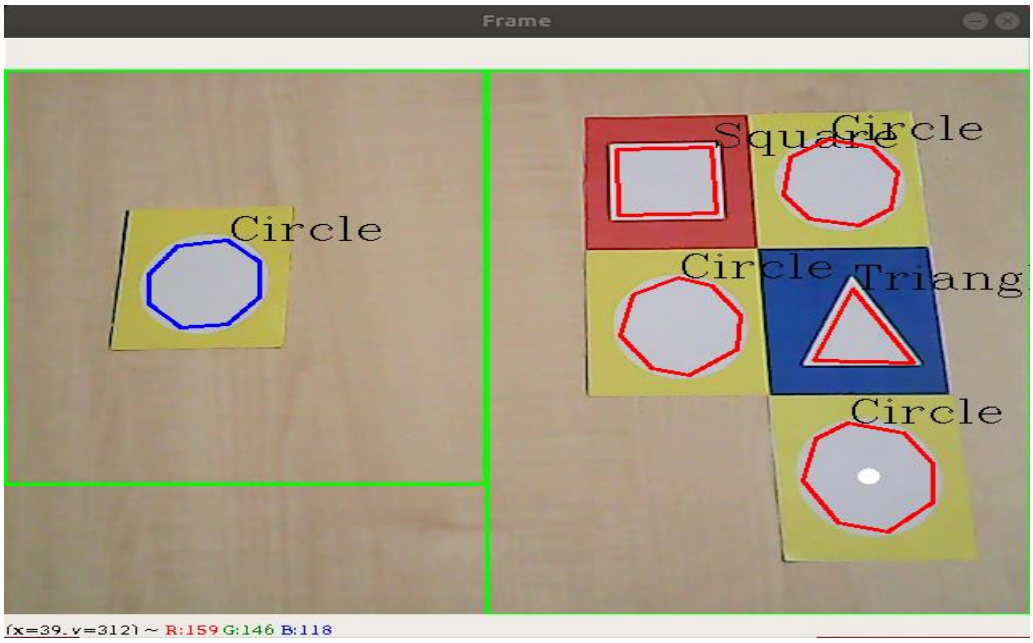
Ακολουθούν δύο φωτογραφίες απεικόνισης της αναγνώρισης του πρώτου αντικειμένου από το OpenCV



Εικόνα 4.30- Αναγνώριση χαμηλότερου κύκλου



Εικόνα 4.31- Αναγνώριση χαμηλότερου κύκλου



Εικόνα 4.32- Αναγνώριση χαμηλότερου κύκλου

4.4 Επεξεργαστική Ταχύτητα σε Πραγματικό Χρόνο και Χρήση CPU (Real Time Processing Speed and CPU Utilization)

TensorFlow

Το TensorFlow λόγο περιορισμένης επεξεργαστικής ισχύεις, μας αποδίδει 1 fps. Τα I/O tasks, σε αντίθεση με τις λειτουργίες που συνδέονται με την CPU, τείνουν να είναι αρκετά αργές. Ενώ οι Computer Vision και οι video processing εφαρμογές είναι σίγουρα αρκετά CPU heavy, ειδικά εάν προορίζονται να εκτελεστούν σε πραγματικό χρόνο, αποδεικνύεται ότι η κάμερα I / O μπορεί επίσης να προκαλέσει μεγάλο bottleneck. Έτσι, μαζί και με τη μη χρήση GPU τα fps είναι πολύ χαμηλά, χωρίς να μειώνεται βέβαια η ποιότητα αναγνώρισης.

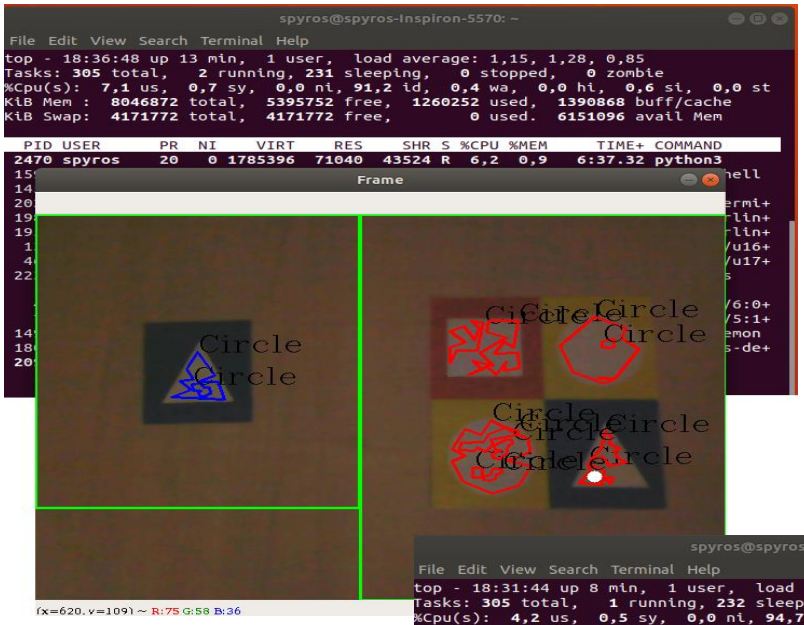
Επίσης πειραματιστήκαμε σχετικά με το αν ο χαμηλός, μέσος ή υψηλός φωτισμός επηρεάζει το utilization της CPU και είδαμε πως παρά την ύπαρξη πολλαπλών συνθηκών φωτισμού, το utilization βρίσκεται σταθερά στο 100 %.

OpenCV

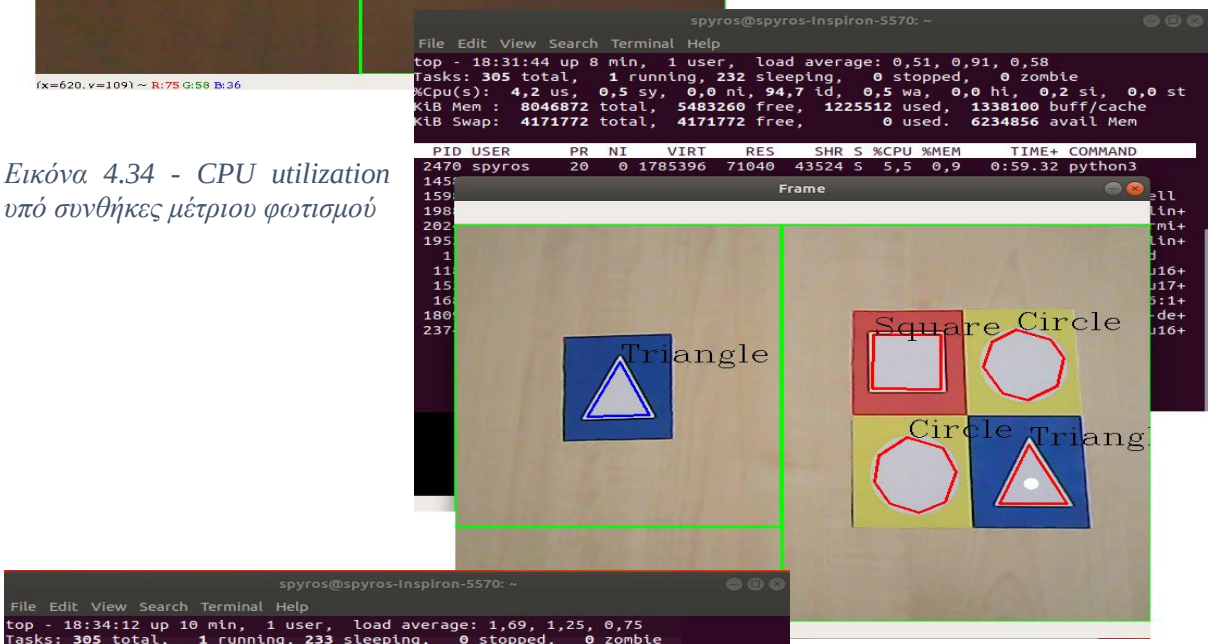
Το OpenCV φαίνεται να έχει μια φυσιολογική επεξεργαστική ταχύτητα με 30 fps. Το OpenCV δεν χρειάζεται ιδιαίτερη επεξεργαστική ισχύ, οπότε δεν αντιμετωπίζουμε lagging προβλήματα.

Επίσης πειραματιστήκαμε και εδώ σχετικά με το αν ο χαμηλός, μέσος ή υψηλός φωτισμός επηρεάζει το utilization της CPU και είδαμε πως η ύπαρξη πολλαπλών συνθηκών φωτισμού επηρεάζει το utilization, ξεκινώντας από τη χρήση του 6.2 % σε συνθήκες χαμηλού φωτισμού, του 5.5 % σε συνθήκες μέτριου φωτισμού και του 30.3 % σε συνθήκες υψηλού φωτισμού.

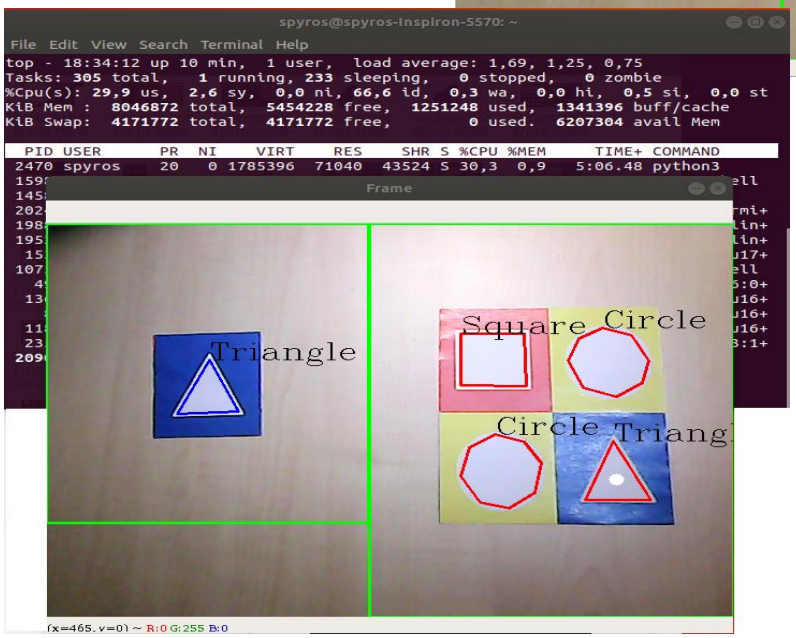
Ακολουθούν φωτογραφίες υπό τις διαφορετικές συνθήκες φωτισμού, μαζί με το CPU utilization.



Εικόνα 4.33 - CPU utilization υπό συνθήκες χαμηλού φωτισμού



Εικόνα 4.34 - CPU utilization υπό συνθήκες μέτριου φωτισμού



Εικόνα 4.35 - CPU utilization υπό συνθήκες υψηλού φωτισμού

Κεφάλαιο 5-Συμπέρασμα Και Προτάσεις Περαιτέρω Ανάπτυξης

Στόχος της εργασίας μας ήταν η συγκριτική μελέτη απόδοσης της αναγνώρισης αντικειμένων μεταξύ του framework του TensorFlow και της βιβλιοθήκης του OpenCV. Τόσο η δημιουργία των αλγορίθμων, όσο και η σύγκριση των δύο προσεγγίσεων, έγινε στα πλαίσια μακρόχρονης επιστημονικής μελέτης και με το κατά το δυνατότερο αντικειμενικό τρόπο.

Το TensorFlow ως μια ολοκληρωμένη Machine Learning πλατφόρμα ανοιχτού κώδικα, έχει εφαρμογές σε αναρίθμητα project ανά τον κόσμο, από ερασιτέχνες προγραμματιστές, μέχρι τις μεγαλύτερες εταιρείες. Στην δική μας εργασία αγγίξαμε μόλις ένα πολύ μικρό κομμάτι των δυνατοτήτων του TensorFlow, αφήνοντας πολλές πτυχές του ανεκμετάλλευτες αλλά αξιοποιώντας στο μέγιστο αυτές που χρειαστήκαμε. Το OpenCV, ως μια open source Computer Vision και Machine Learning software library, μας παρείχε μια πληθώρα λειτουργειών που μας βοήθησαν στο πέρας της εργασίας μας. Το OpenCV αποτελεί και αυτό ένα πολύ δυνατό εργαλείο για Computer Vision εφαρμογές και όχι μόνο, έχοντας δυνατότητα εφαρμογής σε οποιοδήποτε έργο έχει να κάνει με επεξεργασία και ερμηνεία εικόνας. Χρησιμοποιώντας και εδώ ένα μικρό κομμάτι των πραγματικά αναρίθμητων functions που προσφέρει, καταφέραμε να έχουμε αποτελέσματα εξαιρετικής ακρίβειας, έχοντας περιθώρια μεγάλης βελτίωσης και συνολικής τροποποίησης του τελικού αλγορίθμου. Επίσης, μέσα από την απλότητα, τον γρήγορο κύκλο edit-test-debug, αλλά και των επιπρόσθετων βιβλιοθηκών που προσφέρει η Python γλώσσα προγραμματισμού, καταφέραμε να έχουμε ένα άρτιο προγραμματιστικό αποτέλεσμα και στις προσεγγίσεις. Χρησιμοποιώντας αυτή την υψηλού επιπέδου γλώσσα προγραμματισμού, κάναμε το έργο μας πιο απλό στον προγραμματισμό, εύκολα κατανοήσιμο και με επαγγελματικό αποτέλεσμα.

Και οι δύο προσεγγίσεις έχουν τόσο τα φωτεινά, όσο και τα μελανά τους σημεία, τα οποία έρχονταν στην επιφάνεια σε όλη την πορεία της εργασίας. Όσων αφορά την επεξεργαστική ταχύτητα σε μηχάνημα με σχετικά χαμηλή επεξεργαστική ισχύ, το OpenCV υπερτερεί έναντι του TensorFlow. Σε περιπτώσεις με εναλλαγές φωτισμού, το TensorFlow έχει ένα ξεκάθαρο προβάδισμα έναντι του OpenCV. Είναι τέτοιοι παράγοντες που θα πρέπει να λάβει υπόψιν, κάποιος που θα αναλάβει ένα Computer Vision project για την ορθή επιλογή εργαλείου ώστε να επιφέρει τα βέλτιστα επιθυμητά αποτελέσματα.

Εν κατακλείδι, πιστεύουμε πως η συγκεκριμένη εργασία μπορεί να αποτελέσει τον ακρογωνιαίο λίθο για την ανάπτυξη εφαρμογών που μπορούν να έχουν ευρεία χρήση σε πολλούς τομείς. Η μεθοδολογία αναγνώρισης συγκεκριμένου μοτίβου σχήματος, για την

τοποθέτησή του σε συγκεκριμένο σημείο των ιδίων προδιαγραφών, μπορεί να αποτελέσει το έναυσμα για την δημιουργία βιομηχανικών εφαρμογών ταξινόμησης πρώτων υλών ή για την περισυλλογή και το στοίβαγμα του παραγωγικού αποτελέσματος. Μπορεί επίσης να αφαιρεθεί ο ανθρώπινος παράγοντας και να εναρμονιστεί μηχανικός βραχίονας όπου θα λαμβάνει τα αποτελέσματα της χωρικής τοποθεσίας των επιθυμητών αντικειμένων και θα εκτελεί τις αυτός τις απαραίτητες μετακινήσεις στις ανάλογες θέσεις. Οι επεκτάσεις είναι πραγματικά τόσο μεγάλες, όσο και η φαντασία μας και όσο υπάρχει όρεξη για γνώση και εργασία, θα γίνονται ακόμα και περισσότερες.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Bernard Marr. Tech Trends in Practice: The 25Technologies that are Driving the 4th Industrial Revolution. 2020

[2] Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, Kevin Murphy. Speed/Accuracy trade-offs for modern convolutional objects detectors. 2017

[3] James C. Bezberk, James Keller, Raghu Krisnapuram, Nikhil P. Pal. Fuzzy Models And Algorithms for Pattern Recognition and Image Processing. 1999

[4] James J. DiCarlo, Davide Zoccolan, Nicole C. Rust. How Does The Brain Solve Visual Object Recognition. 2012

[5] TensorFlow (https://www.tensorflow.org/)

[6] OpenCV (https://staging.opencv.org/)

[7] Examples Of Applications For Computer Vision (http://www.idconline.com/technical_references/pdfs/electronic_engineering/Examples_of_applications_for_computer_vision.pdf)

[8] Image Processing And Computer Vision (https://openframeworks.cc/ofBook/chapters/image_processing_computer_vision.html)

[9] Neural Networks for Image Recognition: Methods, Best Practices, Applications (https://missinglink.ai/guides/computer-vision/neural-networks-image-recognition-methods-best-practices-applications/)

[10] Building Convolutional Neural Networks on TensorFlow: Three Examples (https://missinglink.ai/guides/tensorflow/building-convolutional-neural-networks-tensorflow-three-examples/)

[11] AI vs Machine Learning vs Deep Learning vs Neural Networks: What's the Difference (https://www.ibm.com/cloud/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks)

[12] Explained: Neural networks. Ballyhooed artificial-intelligence technique known as "deep learning" revives 70-year-old idea. (https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414)

[13] The HSV Color Model in Graphic Design (https://www.lifewire.com/what-is-hsv-in-design-1078068)

[14] Main Types of Neural Networks and its Applications (https://medium.com/towards-artificial-intelligence/main-types-of-neural-networks-and-its-applications-tutorial-734480d7ec8e)

[15] A Brief History of Machine Learning (https://www.dataversity.net/a-brief-history-of-machine-learning/#)

[16] Object Recognition. 3 things you need to know. (https://www.mathworks.com/solutions/image-video-processing/object-recognition.html)

[17] Computer Vision Applications Examples Across Different Industries (https://indatalabs.com/blog/applications-computer-vision-across-industries)

[18] Neural Networks (https://www.investopedia.com/terms/n/neuralnetwork.asp)

[19] Deep learning vs machine learning: a simple way to understand the difference (https://www.zendesk.com/blog/machine-learning-and-deep-learning/)

[20] Computer Vision & Image Processing (https://www.isikdogan.com/blog/computer-vision-image-processing.html)

[21] OpenCV center of contour (https://www.pyimagesearch.com/2016/02/01/opencv-center-of-contour/)

[22] OpenCV shape detection (https://www.pyimagesearch.com/2016/02/08/opencv-shape-detection/)

[23] Determining object color with OpenCV (https://www.pyimagesearch.com/2016/02/15/determining-object-color-with-opencv/)

[24] A Gentle Introduction to Object Recognition With Deep Learning (https://machinelearningmastery.com/object-recognition-with-deep-learning/)

[25] Weights and Biases (https://docs.paperspace.com/machine-learning/wiki/weights-and-biases)

[26] Images are just NumPy Arrays (https://rcvaram.medium.com/images-are-just-numpy-arrays-ceb7b0307fcf)

[27] Introduction to Python OS Module (https://stackabuse.com/introduction-to-python-os-module/)

[28] 16 OpenCV Functions to Start your Computer Vision journey (with Python code) (https://www.analyticsvidhya.com/blog/2019/03/opencv-functions-computer-vision-python/)

[29] Image Segmentation Using Color Spaces in OpenCV + Python (https://realpython.com/python-opencv-color-spaces/)

[30] GPU-Accelerated TensorFlow (https://www.nvidia.com/en-sg/data-center/gpu-accelerated-applications/tensorflow/)

[31] Building a Real-Time Object Recognition App with TensorFlow and OpenCV (https://towardsdatascience.com/building-a-real-time-object-recognition-app-with-tensorflow-and-opencv-b7a2b4ebdc32)

[32] Eager Execution in TensorFlow : A more Pythonic way of building models (https://medium.com/coding-blocks/eager-execution-in-tensorflow-a-more-pythonic-way-of-building-models-e461810618c8)

[33] Getting Started With TensorFlow : Constants, Variables, Placeholders and Sessions (https://medium.com/themlblog/getting-started-with-tensorflow-constants-variables-placeholders-and-sessions-80900727b489)

[34] Real-Time Face Recognition: An End-To-End Project (https://towardsdatascience.com/real-time-face-recognition-an-end-to-end-project-b738bb0f7348)

[35] R-CNN (Object Detection) (https://medium.com/@selfouly/r-cnn-3a9beddfd55a)

[36] Faster R-CNN (Object Detection) (https://medium.com/@selfouly/part-2-fast-r-cnn-object-detection-7303e1988464)

[37] Structuring Your TensorFlow Models (https://danijar.com/structuring-your-tensorflow-models/)

[38] How to Classify Photos of Dogs and Cats (with 97% accuracy) (https://machinelearningmastery.com/how-to-develop-a-convolutional-neural-network-to-classify-photos-of-dogs-and-cats/)