Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών

Συγκριτική Μελέτη Αλγορίθμων Βαθιάς Μηχανικής Μάθησης για Ανίχνευση Αντικειμένων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΑΝΑΣΤΑΣΙΟΣ ΦΑΝΤΑΚΗΣ

Επιβλέπων : Ανδρέας-Γεώργιος Σταφυλοπάτης
Καθηγητής Ε.Μ.Π.

Συνεπιβλέπων : Παρασκευή Τζούβελη
Ε.ΔΙ.Π. Ε.Μ.Π.

Αθήνα, Μάιος 2019

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών

Συγκριτική Μελέτη Αλγορίθμων Βαθιάς Μηχανικής Μάθησης για Ανίχνευση Αντικειμένων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΑΝΑΣΤΑΣΙΟΣ ΦΑΝΤΑΚΗΣ

Επιβλέπων : Ανδρέας-Γεώργιος Σταφυλοπάτης
Καθηγητής Ε.Μ.Π.

Συνεπιβλέπων : Παρασκευή Τζούβελη
Ε.ΔΙ.Π. Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 10η Μαΐου 2019.

Ανδρέας-Γεώργιος Σταφυλοπάτης Καθηγητής Ε.Μ.Π.	Γεώργιος Στάμου Αναπληρωτής Καθηγητής Ε.Μ.Π.	Παναγιώτης Τσανάκας Καθηγητής Ε.Μ.Π.

Αθήνα, Μάιος 2019

Αναστάσιος Φαντάκης

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

Περίληψη

Τις τελευταίες δεκαετίες έχει παρατηρηθεί ιδιαίτερη ανάπτυξη στην έρευνα των τεχνικών της όρασης υπολογιστών (computer vision) ενώ και η ραγδαία εξέλιξη στον τομέα της υπολογιστικής ισχύς έχει δώσει μεγάλη ώθηση στον τομέα της βαθιάς μάθησης (deep learning). Η ανάπτυξη αυτή έχει αποδώσει σημαντικό έργο με σκοπό, μεταξύ άλλων, στη διευκόλυνση της καθημερινότητας των πολιτών σε όλο τον κόσμο. Ένα επιμέρους τμήμα της όρασης υπολογιστών που αξιοποιεί τεχνικές βαθιάς μάθησης αποτελεί η ανίχνευση αντικειμένων (object detection), υποχώρος της ανάλυσης εικόνας. Η ανίχνευση αντικειμένων χρησιμοποιείται σε μια σειρά από εφαρμογές, όπως η αναγνώριση προσώπου και τα συστήματα ελέγχου οδικής κυκλοφορίας.

Σκοπός της παρούσας διπλωματικής εργασίας είναι η συνοπτική παρουσίαση και σύγκριση βασικών αλγορίθμων στον χώρο της ανίχνευσης αντικειμένων αλλά και η βελτιστοποίησή τους με χρήση καινοτόμων τεχνικών από το πεδίο της βαθιάς μάθησης και της ανάλυσης εικόνας. Σε πρώτο στάδιο έγινε η ανάλυση των 2 διαφορετικών μεθόδων ανίχνευσης, δηλαδή αλγορίθμων δύο σταδίων και ενός σταδίου. Δύο χαρακτηριστικά παραδείγματα αυτών αποτελούν ο Faster R-CNN και ο You Only Look Once (YOLO). Για την εκπαίδευση χρησιμοποιήθηκε μια μικρή συλλογή δεδομένων, που είναι διαθέσιμη στον ιστότοπου Kaggle. Τα αποτελέσματα έδειξαν μεγάλο χάσμα κυρίως στο χρόνο εκπαίδευσης με τον αλγόριθμο YOLO να κυριαρχεί σε αυτό το κομμάτι.

Σε επόμενο στάδιο, με βάση τον αλγόριθμο YOLO, επιβεβαιώσαμε τη σημασία των residual blocks στη χρήση τεχνικών βαθιάς μάθησης και επιχειρήσαμε την επέκταση τους. Αναλυτικότερα, δοκιμάσαμε το συνδυασμό της αρχιτεκτονικής του αλγορίθμου YOLO με την τεχνική των Residual Networks of Residual Networks (RoR). Επιπροσθέτως, έγινε δοκιμή της εκπαίδευσης του αλγορίθμου YOLO, με χρήση group normalization αντί batch normalization, μιας τεχνικής που στοχεύει στην άρση των περιορισμών εκπαίδευσης σε περιπτώσεις μικρού batch size. Για τους σκοπούς αυτούς, αξιοποιήθηκαν συλλογές δεδομένων που είναι διαθέσιμες από τον εκπαιδευτικό οργανισμό Udacity και πιο συγκεκριμένα που περιέχουν καταγραφή οδικής κυκλοφορίας. Τα αποτελέσματα έδειξαν μια αισθητή βελτίωση στην απόδοση του νέου δικτύου σε κάθε περίπτωση ανοίγοντας νέες προοπτικές για περαιτέρω μελέτη των νέων αυτών τεχνικών στο τομέα της ανάλυσης εικόνας.

Λέξεις κλειδιά

Τεχνητή Νοημοσύνη, Βαθιά Μάθηση, Ανίχνευση Αντικειμένων, Faster R-CNN, You Only Look Once (YOLO), Networks of Residual Networks (RoR), Group Normalization

Abstract

Over the last few decades, research into computer vision techniques has grown considerably, and rapid developments in computing power have boosted the field of deep learning. This development has attributed important work with the aim, among other things, of facilitating the everyday life of citizens around the world. An individual segment of computer vision that utilizes deep learning techniques is object detection, a sector of image analysis. Object detection is used in a number of applications, such as facial recognition and road traffic control systems.

The aim of this diploma thesis is to summarize and compare basic algorithms in the sector of object detection and to optimize them using innovative techniques from the field of deep learning and image analysis. The first step was to analyze the two different detection methods, namely two-stage and one-stage algorithms. Two typical examples include Faster R-CNN and You Only Look Once (YOLO). A small collection of data, available on Kaggle, was used for training. The results showed a large gap mainly in training time with the YOLO algorithm dominating this task.

In the next step, based on the YOLO algorithm, we confirmed the importance of residual blocks in the use of deep learning techniques and attempted to extend them. In more detail, we tested the combination of the YOLO algorithm architecture with Residual Networks of Residual Networks (RoR). In addition, the YOLO algorithm was tested using group normalization instead of batch normalization, a technique aimed at confronting training limitations in small batch sizes. For these purposes, data collections were utilized which are available from the Udacity educational organization and more specifically containing road traffic records. The results showed a marked improvement in the performance of the new network in each case opening new perspectives for further study of these new techniques in image analysis.

Key words

Artificial Intelligence, Deep Learning, Object Detection, Faster R-CNN, You Only Look Once (YOLO), Networks of Residual Networks (RoR), Group Normalization

Ευχαριστίες

Η παρούσα διπλωματική εκπονήθηκε στα πλαίσια του Προπτυχιακού Προγράμματος Σπουδών της Σχολής Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Εθνικού Μετσόβιου Πολυτεχνείου και σηματοδοτεί την ολοκλήρωση των σπουδών μου. Πριν όμως από οποιαδήποτε αναφορά στη διαδικασία που ακολουθήθηκε και στα αποτελέσματα που προέκυψαν, θα ήθελα να ευχαριστήσω θερμά τους ανθρώπους οι οποίοι συνέβαλαν στην ολοκλήρωση της εργασίας αυτής.

Αρχικά θα ήθελα να απευθύνω τις ευχαριστίες μου στον επιβλέποντα κ. Ανδρέα-Γεώργιο Σταφυλοπάτη, Καθηγητή Ε.Μ.Π, ο οποίος μου προσέφερε τη δυνατότητα να εκπονήσω την διπλωματική μου σε ένα αντικείμενο ιδιαίτερα ελκυστικό και ενδιαφέρον για μένα και να διευρύνω τις επιστημονικές μου γνώσεις. Παράλληλα θα ήθελα να ευχαριστήσω τον κ. Γεώργιο Στάμου Αναπληρωτή Καθηγητή Ε.Μ.Π και τον κ. Παναγιώτη Τσανάκα Καθηγητή Ε.Μ.Π για την τιμή που μου έκαναν να είναι μέλη της επιτροπής εξέτασης της διπλωματικής εργασίας.

Ειδική μνεία αξίζει στην κ. Παρασκευή Τζούβελη, Ε.ΔΙ.Π Ε.Μ.Π. για τον χρόνο που αφιέρωσε και την θεμελιώδη της συνεισφορά στην εκπόνηση της συγκεκριμένης εργασίας. Τόσο η επιστημονική όσο και η πνευματική της στήριξη ήταν ιδιαίτερα σημαντική για εμένα.

Τέλος, θα ήθελα να ευχαριστήσω τους φίλους μου και την οικογένειά μου για την συμπαράστασή τους σε όλη την διάρκεια των σπουδών, και ιδιαίτερα τους Άλκη Κούκο και Γιάννη Φίλιππα.

Αναστάσιος Φαντάκης,

Αθήνα, 10η Μαΐου 2019

Περιεχόμενα

Περίληψη . . . 5

Abstract . . . 7

Ευχαριστίες . . . 9

Περιεχόμενα . . . 11

Κατάλογος σχημάτων . . . 13

1. Εισαγωγή . . . 17
1.1 Τεχνητή Νοημοσύνη . . . 17
1.2 Μηχανική Μάθηση . . . 18
1.3 Τεχνητά Νευρωνικά Δίκτυα και Βαθιά Μάθηση . . . 19
1.4 Ανίχνευση Αντικειμένων . . . 20

2. Θεωρητικό Υπόβαθρο . . . 23
2.1 Στατιστική Ταξινόμηση . . . 23
2.1.1 Γραμμική Ταξινόμηση . . . 23
2.1.2 Μη Γραμμική Ταξινόμηση . . . 24
2.2 Εκπαίδευση Συστημάτων . . . 25
2.2.1 Συναρτήσεις Σφάλματος . . . 25
2.2.2 Συναρτήσεις Ενεργοποίησης . . . 26
2.2.3 Συναρτήσεις Βελτιστοποίησης . . . 28
2.2.4 Υπερπροσαρμογή . . . 34
2.3 Συνελικτικά Νευρωνικά Δίκτυα . . . 36
2.3.1 Συνελικτικό Επίπεδο (Convolutional Layer) . . . 37
2.3.2 Επίπεδο Συγκέντρωσης (Pooling Layer) . . . 38
2.3.3 Πλήρως Συνδεδεμένο Επίπεδο (Fully-Connected Layer) . . . 39
2.3.4 Κανονικοποίηση Παρτιδών (Batch Normalization) . . . 39
2.4 Μετρικές Απόδοσης . . . 40

3. Μοντέλα Ανίχνευσης Αντικειμένων . . . 43
3.1 Faster R-CNN . . . 43
3.2 You Only Look Once (YOLO) . . . 45
3.3 Καινοτομίες και Χρήση Νέου Μοντέλου . . . 48
3.3.1 Group-Normalization . . . 48
3.3.2 Residual Networks of Residual Networks . . . 49

4. Πειραματική διαδικασία και αποτελέσματα . . . 53
4.1 Συλλογή Δεδομένων . . . 53
4.2 Διαδικασία Εκπαίδευσης και Εργαλεία . . . 53
4.3 Πειραματικές Διαδικασίες . . . 54

4.3.1 Εξέλιξη του YOLO 54
4.3.2 Σύγκριση με τον αλγόριθμο Faster R-CNN 55
4.3.3 Αξιοποίηση μεγάλων συλλογών δεδομένων και εφαρμογή καινοτομιών . . . 57
4.4 Αναλυτικά Αποτελέσματα 63

5. Συμπεράσματα 67

Βιβλιογραφία 69

Κατάλογος σχημάτων

1.1 Οι σύνεδροι του Dartmouth . . . 17
1.2 Kasparov εναντίον Deep Blue . . . 18
1.3 Ένα απλό νευρωνικό δίκτυο . . . 19
1.4 Βαθύ νευρωνικό δίκτυο . . . 20
1.5 Παράδειγμα Ανίχνευσης Αντικειμένων και Κατάτμησης Εικόνας . . . 20

2.1 Το μοντέλο perceptron . . . 24
2.2 Ένα MLP με 3 κρυφά επίπεδα . . . 24
2.3 Εποπτικά η συνάρτηση σφάλματος hinge loss . . . 25
2.4 Σιγμοειδής Συνάρτηση . . . 26
2.5 Η Συνάρτηση tanh . . . 27
2.6 Η Συνάρτηση ReLU . . . 27
2.7 Η Γραφική Παράσταση της Leaky ReLU . . . 28
2.8 Σχηματική παρουσίαση του αλγορίθμου back propagation σε ένα νευρώνα. Με πράσινες γράμμες παρουσιάζεται η εμπρόσθια διάδοση ενώ με κόκκινες η ανάστροφη. . . . 29
2.9 Σχηματική παρουσίαση της επίδρασης του ρυθμού εκμάθησης σε ένα σύστημα μηχανικής μάθησης. . . . 30
2.10 Το πρόβλημα εντοπισμού ολικού ελαχίστου για τον vanilla gradient descent. . . . 30
2.11 Εποπτική σύγκριση των 3 αλγορίθμων σε **κυρτή** επιφάνεια. . . . 31
2.12 Saddle point. . . . 32
2.13 Σύγκριση βήματος Nesterov Momentum με κλασικό Momentum . . . 32
2.14 Σύγκριση των αλγορίθμων βελτιστοποίησης. Παρατηρούμε την ανωτερότητα των εκδόσεων του Adam αλλά του SGD με ορμή έναντι των άλλων μοντέλων . . . 34
2.15 Παράδειγμα υπερπροσαρμογής και υπόπροσαρμογής . . . 35
2.16 Παράδειγμα early stopping για την αποφυγή υπερπροσαρμογής . . . 35
2.17 Η χρήση dropout σε ένα νευρωνικό δίκτυο . . . 36
2.18 Σύγκριση ενός παραδοσιακού νευρωνικού συστήματος με ένα συνελικτικό δίκτυο . . . 37
2.19 Διαδικασία συνέλιξης: Ο μπλέ πίνακας εκφράζει την είσοδο, ο πράσινος το φίλτρο και δεξιά βλέπουμε να σχηματίζεται ο χάρτης χαρακτηριστικών . . . 37
2.20 Σύγκριση δειγματοληψίας με max και average pooling . . . 39
2.21 Εικονική αναπαράσταση των TP-TF-FP-FN. Ο κύκλος που αναπαρίσταται με συνεχή γραμμή δείχνει τις πραγματικές τιμές των δεδομένων. Ο κύκλος που αναπαρίσταται με διακεκομμένη γραμμή δείχνει τις προβλέψεις του μοντέλου. . . . 40
2.22 Precision και Recall . . . 41
2.23 Intersection over Union σχηματικά . . . 42

3.1 Η διαδικασία που ακολουθείται στο Faster R-CNN. Ο χάρτης χαρακτηριστικών τροφοδοτείται στο υποδίκτυο RPN και έπειτα γίνεται η τελική αξιολόγηση . . . 44
3.2 Η προκαθορισμένη επιλογή είναι η χρήση 9 anchor boxes, 3 σχημάτων σε διαφορετικές διαστάσεις τα οποία εξετάζονται σε κάθε θέση του χάρτη χαρακτηριστικών. Ο αριθμός αυτός μπορεί να αλλάξει ανάλογα με τις ανάγκες και τις απαιτήσεις του κάθε προβλήματος. . . . 44
3.3 Η αρχιτεκτονική του VGG-16. . . . 45

3.4 Χωρισμός εικόνας που κάνει ο YOLO σε πλέγμα 13x13 46
3.5 Η πλήρης αρχιτεκτονική του YOLOv3 48
3.6 Διαφορετικές προσεγγίσεις των κυρίαρχων τεχνικών κανονικοποίησης 49
3.7 Σύγκριση αποτελέσματων για διαφορετικές κανονικοποιήσεις στον τομέα της ταξινόμησης εικόνων. Παρατηρούμε ότι το ποσοστό λάθους με χρήση group normalization μένει σχεδόν ανεπηρεάστο από των αριθμό παρτίδων ενώ με χρήση batch normalization η αύξηση των λαθών είναι αλματώδης 49
3.8 Εικονική απεικόνιση ενός residual block. 50
3.9 Αναλυτική περιγραφή του Darknet-53 50
3.10 Σύγκριση των αρχιτεκτονικών των απλών ResNets με τα RoR 51

4.1 Τα διαγράμματα των συναρτήσεων απώλειας για το training και το validation set για 90 εποχές εκπαίδευσης του tiny YOLOv2 55
4.2 Τα διαγράμματα των συναρτήσεων απώλειας για το training και το validation set για 90 εποχές εκπαίδευσης του full YOLOv2 55
4.3 Συνδυαστικά αποτελέσματα γραφημάτων για τις 2 εκδόσεις του YOLOv2 55
4.4 Η χρήση της νεότερης έκδοσης βελτίωσε την ταχύτητα σύγκλισης και μείωσε τις πιθανές ανωμαλίες της απώλειας 56
4.5 Η πορεία εκπαίδευσης του αλγορίθμου Faster R-CNN με το δίκτυο ResNet-50. Ο οριζόντιος άξονας μετριέται σε iterations αντί για εποχές 56
4.6 Η πορεία εκπαίδευσης του αλγορίθμου Faster R-CNN με το δίκτυο ResNet-101. . . 56
4.7 Η εκπαίδευσης του αλγορίθμου YOLO στο νέο dataset με τις 3 κλάσεις. 57
4.8 Η εκπαίδευσης του αλγορίθμου YOLO στο νέο dataset με τις 4 κλάσεις. 57
4.9 Διαφορά στην αρχιτεκτονική plain nets και residual nets 58
4.10 Σύγκριση της εκπαίδευσης με χρήση plain δικτύου (μπλε γραμμή) και residual δικτύου (πορτοκαλί γραμμή) για την συλλογή με τις 3 κλάσσεις. 58
4.11 Σύγκριση της εκπαίδευσης με χρήση plain δικτύου (μπλε γραμμή) και residual δικτύου (πορτοκαλί γραμμή) για την συλλογή με τις 4 κλάσσεις. 58
4.12 Σύγκριση της εκπαίδευσης με χρήση plain δικτύου (μπλε γραμμή) και residual δικτύου (πορτοκαλί γραμμή) για την συλλογή με τα πλοία. 59
4.13 Σύγκριση της εκπαίδευσης του εξελιγμένου δικτύου (μπλε γραμμή) και του κλασσικού δικτύου (πορτοκαλί γραμμή) για την συλλογή με τα πλοία. 59
4.14 Σύγκριση της εκπαίδευσης του εξελιγμένου δικτύου (μπλε γραμμή) και του κλασσικού δικτύου (πορτοκαλί γραμμή) για την συλλογή με τις 3 κλάσσεις. 60
4.15 Σύγκριση της εκπαίδευσης του εξελιγμένου δικτύου (γκρι γραμμή) και του κλασσικού δικτύου (πορτοκαλί γραμμή) για την συλλογή με τις 4 κλάσσεις. 60
4.16 Σύγκριση της εκπαίδευσης του εξελιγμένου δικτύου (κόκκινη γραμμή) και του κλασσικού δικτύου (πορτοκαλί γραμμή) για τη συλλογή με τα πλοία χωρίς τη χρήση προεκπαιδευμένων βαρών. 60
4.17 Σύγκριση της εκπαίδευσης του εξελιγμένου δικτύου (γαλάζια γραμμή) και του κλασσικού δικτύου (πράσινη γραμμή) για την συλλογή με τις 3 κλάσεις χωρίς την χρήση προεκπαιδευμένων βαρών. 61
4.18 Σύγκριση της εκπαίδευσης του εξελιγμένου δικτύου (πράσινη γραμμή) και του κλασσικού δικτύου (γαλάζια γραμμή) για την συλλογή με τις 4 κλάσεις χωρίς την χρήση προεκπαιδευμένων βαρών. 61
4.19 Σύγκριση της εκπαίδευσης δικτύου με χρήση group normalization (πράσινη γραμμή) και δικτύου με χρήση batch normalization (πορτοκαλί γραμμή) για τη συλλογή με τα πλοία. 62
4.20 Σύγκριση της εκπαίδευσης δικτύου με χρήση group normalization (κόκκινη γραμμή) και δικτύου με χρήση batch normalization (πορτοκαλί γραμμή) για την συλλογή με τις 3 κλασεις. 62

4.21 Σύγκριση της εκπαίδευσης δικτύου με χρήση group normalization (κόκκινη γραμμή) και δικτύου με χρήση batch normalization (πορτοκαλί γραμμή) για τη συλλογή με τις 4 κλασεις. 62
4.22 Σύγκριση της εκπαίδευσης δικτύου με χρήση group normalization (μπλε γραμμή) και δικτύου με χρήση batch normalization (πορτοκαλί γραμμή) from scratch για τη συλλογή με τα πλοία. 63
4.23 Σύγκριση της εκπαίδευσης δικτύου με χρήση group normalization (ροζ γραμμή) και δικτύου με χρήση batch normalization (πράσινη γραμμή) from scratch για τη συλλογή με τις 3 κλάσεις. 63
4.24 Σύγκριση της εκπαίδευσης δικτύου με χρήση group normalization (ροζ γραμμή) και δικτύου με χρήση batch normalization (γαλάζια γραμμή) from scratch για τη συλλογή με τις 4 κλάσεις. 63
4.25 Παραδείγματα ανίχνευσης 1 64
4.26 Παραδείγματα ανίχνευσης 2 65
4.27 Παραδείγματα ανίχνευσης 3 65

Κεφάλαιο 1

Εισαγωγή

1.1 Τεχνητή Νοημοσύνη

Αν και η ”γέννηση” ενός ολόκληρου κλάδου, ειδικά τόσο πολυδιάστατου, δεν είναι εύκολο να προσδιοριστεί με ακρίβεια χρονικά, είναι ευρέως αποδεκτό πως οι απαρχές της Τεχνητής Νοημοσύνης εντοπίζονται στη δεκαετία του 1940. Εξέχοντα ρόλο στη διαδικασία αυτήν διαδραμάτισε ο Βρετανός μαθηματικός Alan Turing. Τόσο με τη δημιουργία του Enigma [Hodg12], του συστήματος που στόχευε στην αποκωδικοποίηση του κώδικα επικοινωνίας του γερμανικού στρατού, και ακόμη περισσότερο με την εισαγωγή του Turing Test [Turi09]. Πιο συγκεκριμένα, στο Turing Test εξετάζονται οι συνθήκες με βάση τις οποίες μπορεί να θεωρηθεί μια μηχανή ευφυής, ελέγχοντας την ικανότητα ή μη να εξαπατήσει έναν παρατηρητή, θεωρώντας την ”άνθρωπο”.

Η πρώτη επίσημη επαφή, όμως, της ακαδημαϊκής κοινότητας με την Τεχνητή Νοημοσύνη έρχεται το 1956, όπου σε ένα συνέδριο στο Darthmouth [McCa06], ο John McCarthy εισήγαγε τον ορισμό της, ως την επιστήμη και μεθοδολογία της δημιουργίας νοούντων μηχανών. Οι σύνεδροί του αποτέλεσαν τον πρώτο πυρήνα ερευνητών της Τεχνητής Νοημοσύνης και μια νέα εποχή μόλις άρχιζε. Τα πρώτα αποτελέσματα ήταν εντυπωσιακά και η πεποίθηση που δημιουργήθηκε ήταν πως σύντομα τα περισσότερα ζητήματα του χώρου θα έχουν λυθεί. Η υπερεκτίμηση, όμως, των τότε δυνατοτήτων έφερε τις πρώτες κωλυσιεργίες και η απογοήτευση μεταφράστηκε σε μείωση χρηματοδότησης, εγκατάλειψη ερευνητικών προγραμμάτων και την αρχή μιας εποχής που ονομάστηκε *”Χειμώνας της Τεχνητής Νοημοσύνης”*.

Σχήμα 1.1: Οι σύνεδροι του Dartmouth

Η δεκαετία του 1980 σηματοδοτεί την ανάκαμψη του κλάδου εξαιτίας της μεγάλης εμπορικής επιτυχία των λεγόμενων *έμπειρων συστημάτων*. Τα συστήματα αυτά σχεδιάστηκαν ώστε να αξιοποιούν τη διαθέσιμη γνώση και να λύνουν βάσει αυτής πολύπλοκα προβλήματα όπως κάνει ένας εμπειρογνώμονας, εξ ου και η ονομασία τους. Ταυτόχρονα όμως, οι ηλεκτρονικοί υπολογιστές αναπτύσσονταν με τέτοιο ρυθμό ώστε το βάρος της ερευνητικής κοινότητας να πέσει σε αυτόν τον τομέα.

Από την δεκαετία του 1990 μέχρι τις μέρες, η Τεχνητή Νοημοσύνη βρίσκεται σε διαρκή άνθιση. Η αυξανόμενη υπολογιστική ισχύς, η εστίαση στην επίλυση συγκεκριμένων προβλημάτων αλλά και η αλληλεπίδραση με άλλους τομείς είναι μερικοί από τους λόγους αυτής της ανάπτυξης. Η Τεχνητή Νοημοσύνη πλέον βρίσκει εφαρμογή σε μια μεγάλη κλίμακα πεδίων. Χαρακτηριστικό της εξέλιξής της είναι η αναμέτρηση και νίκη του υπερυπολογιστή Deep Blue επί του παγκόσμιου πρωταθλητή στο σκάκι Garry Kasparov το 1997 [Hsu04].

Σχήμα 1.2: Kasparov εναντίον Deep Blue

Έκτοτε η Τεχνητή Νοημοσύνη, μέσω της Μηχανικής Μάθησης μεταξύ άλλων, έχει καταμετρήσει μια σειρά από επιτυχίες σε αντίστοιχες αναμετρήσεις, δείχνοντας το δρόμο για ένα ακόμα πιο λαμπρό και προσοδοφόρο μέλλον για το χώρο.

1.2 Μηχανική Μάθηση

Η *Μηχανική Μάθηση (Machine Learning)* αποτελεί ένα παρακλάδι της Τεχνητής Νοημοσύνης. Το όνομα αποδόθηκε από τον Arthur Samuel το 1959 [Samu59] και βασίζεται στην καινοτόμα προσέγγιση με βάση την οποία οι υπολογιστές αντί να προγραμματίζονται ντετερμινιστικά για τον τρόπο εκτέλεσης εντολών, να μπορούν από μόνοι τους να διδαχθούν πως θα πρέπει να δράσουν. Είναι άρρηκτα συνδεδεμένη με την επιστήμη της *Υπολογιστικής Στατιστικής (Computational statistics)* καθώς η διαδικασία εξέλιξής της συχνά βασίζεται σε μαθηματικές βελτιστοποιήσεις. Ο τυπικός ορισμός δόθηκε από τον Tom M. Mitchell και είναι ο εξής: ”Ένα πρόγραμμα υπολογιστή λέγεται ότι μαθαίνει από εμπειρία E ως προς μια κλάση εργασιών T και ένα μέτρο επίδοσης P, αν η επίδοσή του σε εργασίες της κλάσης T, όπως αποτιμάται από το μέτρο P, βελτιώνεται με την εμπειρία E” [Mitc06].

Στόχος της αποτελεί η διεύρυνση του παραδοσιακού προγραμματισμού στη βάση μιας λογικής όπου ένα πρόγραμμα μηχανικής μάθησης να μπορεί αξιοποιώντας τους διαθέσιμους αλγορίθμους να προσαρμόζεται στο εκάστοτε πρόβλημα. Παραδείγματα μηχανικής μάθησης που ήδη έχουμε πρακτικές εφαρμογές είναι η αναγνώριση ομιλίας και γραφικού χαρακτήρα αλλά και η αυτόματη οδήγηση αυτοκινήτων.

Η μηχανική μάθηση ταξινομούνται σε 3 βασικές κατηγορίες ανάλογα με την φύση των δεδομένων εισόδου και την ανατροφοδότηση σε ένα σύστημα εκμάθησης [Russ95]:

- **Επιτηρούμενη μάθηση (Supervised learning)**: Το υπολογιστικό σύστημα δέχεται τα δεδομένα εισόδου μαζί με τις ετικέτες του, δηλαδή τα επιθυμητά χαρακτηριστικά, και ο στόχος είναι να μάθει έναν γενικό κανόνα προκειμένου να αντιστοιχίσει τις εισόδους με τα αποτελέσματα.

- **Μη επιτηρούμενη μάθηση (Unsupervised learning)**: Στο σύστημα δεν παρέχονται μαζί με τα δεδομένα εισόδου ετικέτες, δηλαδή επιπρόσθετες πληροφορίες, και καλείται να βρει τη δομή

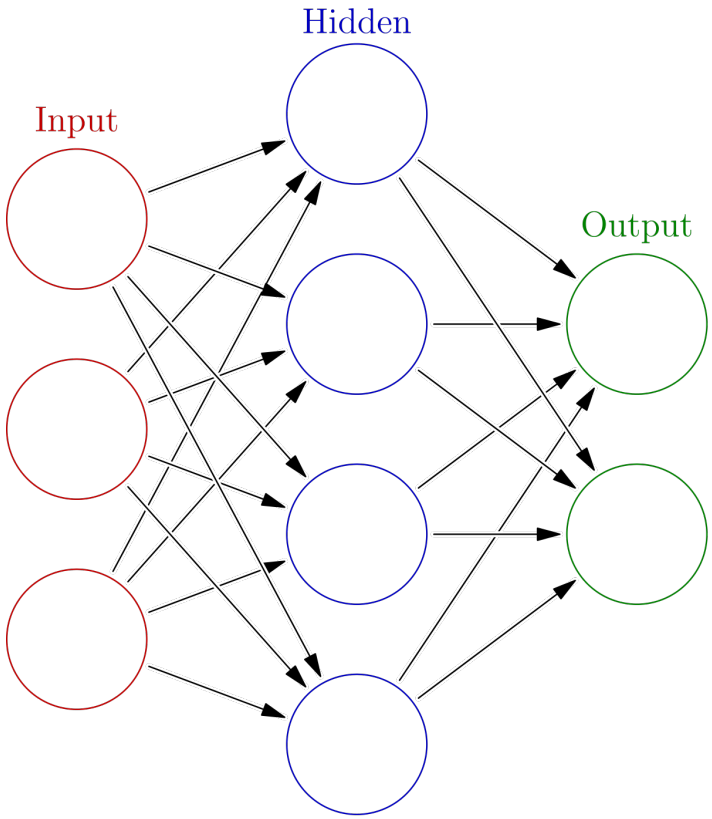
των δεδομένων. Η μη επιτηρούμενη μάθηση μπορεί να χρησιμοποιηθεί για την ανακάλυψη κρυμμένων μοτίβων σε δεδομένα ή ως μέσο για την εύρεση κάποιου χαρακτηριστικού.

- **Ενισχυτική μάθηση (Reinforcement learning)**: Ουσιαστικά πρόκειται για συνδυασμό των προηγούμενων δύο, καθώς το σύστημα λαμβάνει δεδομένα εισόδου χωρίς ετικέτες και προσπαθεί να εξάγει χαρακτηριστικά. Σε περίπτωση που το κάνει με επιτυχία ”επιβραβεύεται” ενώ αντίθετα δέχεται ”τιμωρία”. Στόχος είναι η μεγιστοποίηση της ανταμοιβής.

Στην παρούσα εργασία θα ασχοληθούμε μόνο με συστήματα επιτηρούμενης μάθησης, εξ ου θα γίνει και ανάλυση μόνο αυτού του πεδίου.

1.3 Τεχνητά Νευρωνικά Δίκτυα και Βαθιά Μάθηση

Το *τεχνητό νευρωνικό δίκτυο(Artificial Neural Network)* είναι ένα δίκτυο από απλούς υπολογιστικούς κόμβους (νευρώνες), διασυνδεδεμένους μεταξύ τους. Κάθε νευρώνας δέχεται ένα σύνολο εισόδων από διαφορετικές πηγές και παράγει μία έξοδο. Πολλές αλγοριθμικές στρατηγικές που αφορούν τη μηχανική μάθηση μπορούν να μοντελοποιηθούν με τη χρήση νευρώνων. Οι νευρώνες κατανέμονται σε διάφορα επίπεδα, τα εσωτερικά ονομάζονται και κρυφά (hidden), επιτελώντας ξεχωριστές λειτουργίες. Η απρόσεκτη αύξηση επιπέδων μπορεί μεν να επιφέρει μεγαλύτερη ακρίβεια αλλά παράλληλα να αυξήσει και την πολυπλοκότητα σε τέτοιο βαθμό που να καθιστά το σύστημα μη αποδοτικό στην πράξη.

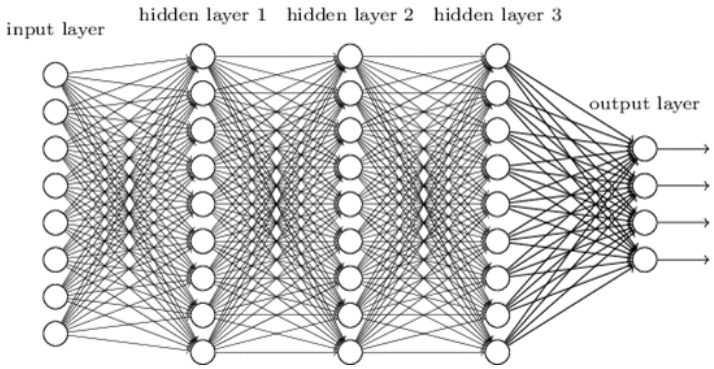


Σχήμα 1.3: Ένα απλό νευρωνικό δίκτυο

Η *Βαθιά Μάθηση (Deep Learning)* αποτελεί εξέλιξη των απλών νευρωνικών δικτύων. Ο όρος εισήχθη από την Rina Dechter το 1986 [Dech86] στην μηχανική μάθηση και από τον Igor Aizenberg το 2000 στα τεχνητά νευρωνικά δίκτυα [Aize13]. Σε αντίθεση με τα απλά νευρωνικά δίκτυα, στη βαθιά

μάθηση έχουμε πολύ περισσότερα κρυφά επίπεδα που αποσκοπούν στην εξαγωγή συμπερασμάτων με μεγαλύτερη ακρίβεια καθώς και εξαγωγή χαρακτηριστικών υψηλού επιπέδου.

Τα δίκτυα συνήθως λαμβάνουν πολυδιάστατα δεδομένα και σε συνδυασμό με την ύπαρξη πολλών επιπέδων, η επεξεργασία αυτών καθίσταται εξαιρετικά χρονοβόρα. Όμως, η ανάπτυξη της τεχνολογίας έχει φέρει ευεργετικές επιπτώσεις στο ζήτημα αυτό με την δημιουργία πολυπύρηνων αρχιτεκτονικών και την αξιοποίηση των νέων μονάδων γραφικής επεξεργασίας (GPUs) που πλέον χρησιμοποιούνται κατά κόρον, μειώνοντας δραστικά το χρόνο εκπαίδευσης τέτοιων πολύπλοκων δικτύων.

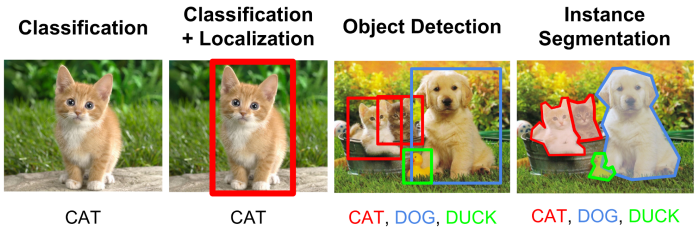


Σχήμα 1.4: Βαθύ νευρωνικό δίκτυο

1.4 Ανίχνευση Αντικειμένων

Η *Ανίχνευση Αντικειμένων (Object Detection)* [Amit14] αποτελεί ένα βασικό πρόβλημα στην Όραση Υπολογιστή (Computer Vision) και στην Επεξεργασία Εικόνας (Image Processing) Ως στόχο έχει την απομόνωση στιγμιότυπων μιας εικόνας ή την ταξινόμησή εικόνων σε σημασιολογικές κλάσεις και την περαιτέρω επεξεργασία τους. Παραπλήσιος όρος είναι και η *Κατάτμηση Εικόνας (Image Segmentation)* [Chen80] όπου κατά την επεξεργασία η εικόνα διαιρείται σε μικρότερες περιοχές ενδιαφέροντος (regions of interest), το οποίο δε θα αποτελέσει αντικείμενο αυτής της διπλωματικής. Τόσο η ανίχνευση αντικειμένων όσο και η τμηματοποίηση εικόνας αποτελούν βασικά στάδια για την ερμηνεία τους και την εξόρυξη πληροφοριών από αυτές.

Αν και μια εικόνα μπορεί να χωριστεί σε λιγότερες ή περισσότερες περιοχές ενδιαφέροντος και να εφαρμοστεί αναλυτική ανίχνευση αντικειμένων, όπου το κάθε επιμέρους αντικείμενο θα πρέπει να χαρακτηρίζεται από τυπική ομοιογένεια η οποία μπορεί να ποικίλλει ανάλογα με τα δεδομένα εισόδου και την φύση του προβλήματος, ενώ αντίστοιχα οι γειτονικές περιοχές να εμφανίζουν αισθητές και ευδιάκριτες διαφορές.



Σχήμα 1.5: Παράδειγμα Ανίχνευσης Αντικειμένων και Κατάτμησης Εικόνας

Σήμερα αυτές οι τεχνικές χρησιμοποιούνται σε μια σειρά από πρακτικές εφαρμογές. Πολύ συνοπτικά αναφέρουμε μερικές:

- Ανίχνευση και αναγνώριση προσώπου

- Αναγνώριση δακτυλικών αποτυπωμάτων
- Συστήματα ελέγχου οδικής κυκλοφορίας
- Διάγνωση ιατρικών ασθενειών

Η εξέχουσα σημασία αυτών των ζητημάτων για τη μηχανική μάθηση επαληθεύεται από την συνεχώς αυξανόμενη προβολή και χρηματοδότηση διαγωνισμών όπως ο ImageNet Large Scale Visual Recognition Competition (ILSVRC) που διεξάγεται σε ετήσια βάση και έχει ως αντικείμενο τα προβλήματα που θέσαμε.

Κεφάλαιο 2

Θεωρητικό Υπόβαθρο

2.1 Στατιστική Ταξινόμηση

Η εξέλιξη των σύγχρονων νευρωνικών δικτύων πέρασε από πολλά στάδια. Η δημιουργία μηχανών με νοημοσύνη που θα μπορούσαν να δράσουν χωρίς συγκεκριμένους κανόνες, αλλά αξιοποιώντας την εκάστοτε υπάρχουσα γνώση και εξετάζοντας παραδείγματα, αποτέλεσε μια σημαντικότατη πρόκληση που απασχόλησε την ερευνητική κοινότητα από τα μέσα του περασμένου αιώνα. Μια πρόκληση η οποία με τον καιρό έγινε επιτακτική ανάγκη εξαιτίας του αυξανόμενου όγκου των πληροφοριών και ως εκ τούτου την αδυναμία του παραδοσιακού προγραμματισμού να ανταπεξέλθει σε ένα πολύ μεγαλύτερο χώρο καταστάσεων.

Η μηχανική μάθηση (machine learning) αποτέλεσε την προσπάθεια της επιστημονικής κοινότητας να κάνει πράξη αυτήν την πρόκληση. Οι μηχανές θα έπρεπε από μόνες τους να εξάγουν κανόνες ύστερα από εκπαίδευση σε υπάρχοντα δεδομένα χωρίς την παρέμβαση ανθρώπινου παράγοντα. Η προσέγγιση αυτή συνέβαλλε τα μέγιστα στην αλματώδη ανάπτυξη της τεχνητής νοημοσύνης που διανύουμε μέχρι την εποχή μας.

Τα νευρωνικά δίκτυα αποτελούνται από ένα σύνολο επιμέρους νευρώνων και των συνδέσεων μεταξύ τους και θεωρούνται η πρακτική εφαρμογή της μηχανικής μάθησης. Αδιαμφισβήτητα, πλέον, αποτελούν βασικό εργαλείο στο πεδίο της αναγνώρισης μοτίβων και πιο συγκεκριμένα στην αναγνώριση εικόνα που θα μας απασχολήσει στα πλαίσια αυτής της εργασίας.

2.1.1 Γραμμική Ταξινόμηση

Σε αυτή την παράγραφο θα ασχοληθούμε με την ανάλυση ενός απλού *Γραμμικού Ταξινομητή (Linear Classifier)* που αποτέλεσε τον προπομπό για τα σύγχρονα νευρωνικά δίκτυα.

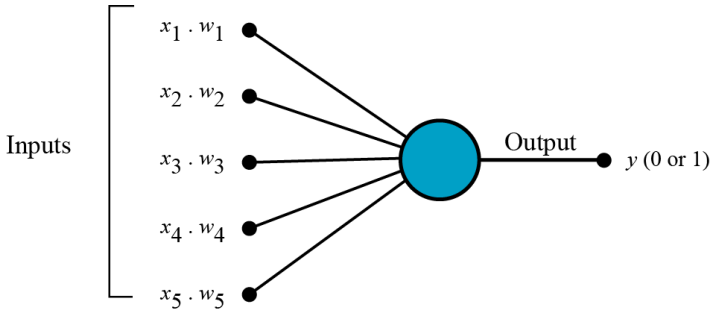
2.1.1.1 Perceptron

Αν και η πρώτη μηχανή νευρωνικών δικτύων δημιουργήθηκε από τους Marvin Minsky και Dean Edmonds το SNARC, το 1951, [Berk97] σημείο σταθμός για την περαιτέρω εξέλιξη του χώρου στις αρχές που το διέπουν μέχρι σήμερα αποτελεί η εφεύρεση του Perceptron. Το Perceptron είναι ένας από τους πρώτους αλγόριθμους επιτηρούμενης μάθησης και εφευρέθηκε το 1957 από τον Frank Rosenblatt [Rose57].

Το νευρωνικό σύστημα αποτελούνταν μόνο από ένα νευρώνα και δεχόταν σαν είσοδο ένα διάνυσμα ύστερα από επεξεργασία έδινε σαν έξοδο έναν δυαδικό αριθμό. Η είσοδος x πολλαπλασιάζεται με έναν πίνακα βαρών w και στο αποτέλεσμα προστίθεται ένας αριθμός b που ονομάζεται πόλωση. Αν το τελικό αποτέλεσμα είναι μεγαλύτερο από ένα κατώφλι c, τότε το σύστημα δίνει έξοδο 1, διαφορετικά 0. Η συνοπτική μαθηματική φόρμουλα δίνεται παρακάτω:

$$f(x) = \begin{cases} 1, & \text{if } w \cdot x + b > c \\ 0, & otherwise \end{cases}$$

Αν και η εφεύρεση του perceptron έφερε επανάσταση στον χώρο, το, όπως είναι λογικό, ακόμα πρώιμο στάδιο του αντιμετώπιζε μερικά μειονεκτήματα τα οποία θα αποτελούσαν τις επόμενες προ-



Σχήμα 2.1: Το μοντέλο perceptron

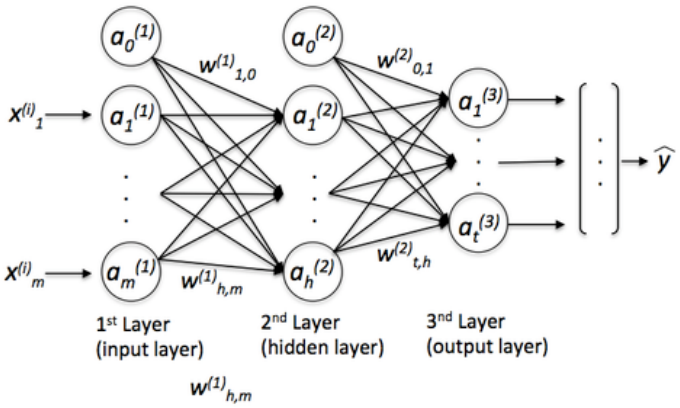
κλήσεις. Η δυαδικότητα του αποτελέσματος, καθώς και η απλότητα του αποτελούσε περιοριστικό παράγοντα στην εξαγωγή πιο σύνθετων συμπερασμάτων ενώ ο αλγόριθμος δεν ανταποκρινόταν κατάλληλα αν το σύνολο δεν ήταν γραμμικά διαχωρίσιμο.

2.1.2 Μη Γραμμική Ταξινόμηση

Από την στιγμή που τα περισσότερα σύνολα δεν είναι γραμμικά διαχωρίσιμα, η κατεύθυνση που όφειλαν να πάρουν τα συστήματα μηχανικής μάθησης ήταν προς την σωστή ταξινόμηση τέτοιων συνόλων. Στην παράγραφο αυτή θα αναφερθεί συνοπτικά η δομή των αλγορίθμων μη γραμμικής ταξινόμησης.

2.1.2.1 Multilayer Perceptron (MLP)

Επόμενο βήμα μετά την εφεύρεση του perceptron ήταν η ανάπτυξη πιο πολύπλοκων αρχιτεκτονικών με στόχο τη βαθύτερη ανάλυση των δεδομένων. Μια τέτοια προσέγγιση ήταν το *Multilayer Perceptron (MLP)*. Σε αντίθεση με το απλό perceptron, η αρχιτεκτονική αυτή επεκτείνει την ιδέα του με περισσότερα από ένα κρυφά στρώματα τα οποία αποτελούνται με περισσότερους από ένα νευρώνες το καθένα. Πρόκειται για ένα πλήρως συνδεδεμένο δίκτυο, με βάση τη λογική ότι κάθε κόμβος ενός επιπέδου συνδέεται με όλους τους κόμβους του επόμενου. Η ύπαρξη πολλών κρυφών επιπέδων μπορεί να αυξήσει εκθετικά το υπολογιστικό κόστος του συστήματος, ως εκ τούτου πρέπει να γίνεται με σύνεση στο πρίσμα της πολυπλοκότητας των χαρακτηριστικών που θέλουμε να εξάγουμε σε κάθε πρόβλημα. Η διαδικασία υπολογισμού της εξόδου κάθε νευρώνα ακολουθεί την ίδια λογική με τον απλό νευρώνα perceptron, με τη διαφορά ότι δεν περιορίζεται στο δυαδικό της χαρακτήρα.



Σχήμα 2.2: Ένα MLP με 3 κρυφά επίπεδα

Η διαδικασία που περιγράφηκε ονομάζεται εμπρόσθια διάδοση (Forward Propagation). Τα σύγχρονα Συνελικτικά Νευρωνικά Δίκτυα ακολουθούν την προσέγγιση αυτή με την ύπαρξη πολλών κρυ-

φών επιπέδων για την εξαγωγή υψηλού επιπέδου χαρακτηριστικών.

2.2 Εκπαίδευση Συστημάτων

2.2.1 Συναρτήσεις Σφάλματος

Η εκπαίδευση ενός συστήματος μηχανικής μάθησης καθιστά απαραίτητη την ύπαρξη μίας *Συνάρτησης Σφάλματος (Loss Function)*. Γενικά με τον όρο σφάλμα εννοούμε τη διαφορά μεταξύ επιθυμητής εξόδου, δηλαδή της πραγματικής, από την έξοδο που προέβλεψε το σύστημά μας. Μια συνάρτηση σφάλματος επεξεργάζεται αυτό το σφάλμα και ουσιαστικά δείχνει πόσο ορθά έχει λειτουργήσει το μοντέλο μας. Διαφορετικές συναρτήσεις μπορούν να οδηγήσουν σε διαφορετικά αποτελέσματα και ως εκ τούτου η επιλογή της πιο κατάλληλης συνάρτησης για το σύστημα αποτελεί κομβικό σκέλος της εκπαίδευσής του. Παρακάτω θα παραθέσουμε συνοπτικά 2 πολύ συνηθισμένες συναρτήσεις σφάλματος για επεξεργασία εικόνας και όχι μόνο.

2.2.1.1 Hinge Loss

Θεωρούμε πως έχουμε ένα σύνολο από εικόνες τις οποίες θέλουμε να κατανείμουμε στις σωστές κλάσεις. Η hinge loss λειτουργεί βάση της λογικής ότι το σύστημά μας θέλει η σωστή κλάση για κάθε εικόνα να λαμβάνει "σκορ" από τους νευρώνες εξόδου μεγαλύτερο από τις υπόλοιπες κατά ένα περιθώριο C. Υποθέτουμε πως για μια i εικόνα η έξοδος από τους νευρώνες είναι ένα διάνυσμα s με αριθμό στοιχείων ίσο με τις υποψήφιες κλάσεις και η επιθυμητή έξοδος είναι y_i. Η hinge loss μοντελοποιείται ως εξής:

$$L_i = \sum_{j \neq y_i} max(0, s_j - s_{y_i} + C)$$

Παρατηρούμε, λοιπόν, πως η συνάρτηση σφάλματος για μία εικόνα μας δίνεται από ένα άθροισμα παραγόντων που πηγάζει από τις λανθασμένες κλάσεις. Οφείλουμε να σημειώσουμε πως στην περίπτωση που το σύστημά μας έχει προβλέψει την σωστή κλάση, δηλαδή έχει "σκορ" μεγαλύτερο από τις άλλες κατά C, ο δεξιός όρος βγαίνει αρνητικός και εξαιτίας της συνάρτησης max μας δίνει 0. Δηλαδή οι εικόνες που έχουν ταξινομηθεί σωστά μας δίνουν μηδενικό σφάλμα και στην συνάρτηση συμβάλλουν μόνο οι λάθος επιλογές, κάτι που είναι το ευκταίο. Τέλος, όμως, πρέπει να σημειώσουμε ότι στο σφάλμα συμμετέχουν και οι σωστές προβλέψεις αν αυτές δεν έχουν αρκετά μεγάλη πιθανότητα οπότε οφείλουμε να κάνουμε σωστή επιλογή του C.

Σχήμα 2.3: Εποπτικά η συνάρτηση σφάλματος hinge loss

2.2.1.2 Cross-Entropy Loss

Η συνάρτηση σφάλματος που είδαμε προηγουμένως αξιοποιεί αυτούσια την έξοδο από τους τελικούς νευρώνες με σκοπό να υπολογίσει το παραγόμενο σφάλμα. Οι αριθμοί όμως που παράγονται από τους νευρώνες δεν σημαίνουν κάτι διαισθητικά για τον ανθρώπινο νου. Αντίθετα, η συνάρτηση σφάλματος της Σταυροειδούς Εντροπίας (Cross Entropy) επεξεργάζεται τα σκορ από τους νευρώνες και τα μετατρέπει σε πιθανότητες ενός στοιχείου να ανήκει σε κάποια κλάση. Προφανώς οι πιθανότητες κυμαίνονται από 0 έως 1 και το άθροισμα τους δίνει μονάδα. Θεωρώντας τις ίδιες υποθέσεις που κάναμε πριν η πιθανότητα ενός στοιχείου να ανήκει στην κλάση i δίνεται από τον εξής μαθηματικό τύπο:

$$P_i = \frac{e^{f_{y_i}}}{\sum_j e^{f_j}}$$

Μετά την κανονικοποίηση των τιμών σε πιθανότητες, το σφάλμα υπολογίζεται ως εξής:

$$L_i = -log(P_i) \text{ όπου i η επιθυμητή κλάση}$$

Παρατηρούμε ξανά ότι όσο μεγαλύτερη είναι η πιθανότητα να επιλέξουμε την σωστή κλάση, τόσο μικραίνει και το παραγόμενο σφάλμα. Σε αντίθεση με πριν, όμως, όλες οι παρατηρήσεις μας συνεισφέρουν σε αυτό. Οφείλουμε να σημειώσουμε ότι σε πιο πολύπλοκα προβλήματα, όπως αυτό της ανίχνευσης αντικειμένων, οι συναρτήσεις σφάλματος λαμβάνουν πιο σύνθετες μορφές για να ικανοποιούν τις επιμέρους απαιτήσεις όπως θα δούμε σε μετέπειτα κεφάλαιο.

2.2.2 Συναρτήσεις Ενεργοποίησης

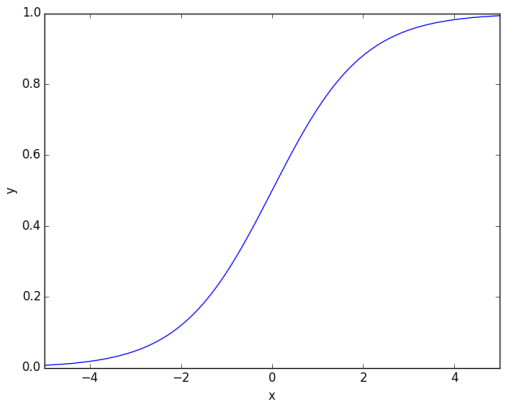
Οι *Συναρτήσεις Ενεργοποίησης (Activation Functions)* αποτελούν αναπόσπαστο κομμάτι των νευρωνικών δικτύων. Πρακτικά οι συναρτήσεις αυτές είναι που εισάγουν μη γραμμικές ιδιότητες στο δίκτυο. Η έξοδος κάθε νευρώνα, ύστερα από την επεξεργασία της εισόδου με τα αντίστοιχα βάρη και την πόλωση που περιγράψαμε νωρίτερα, φιλτράρεται μέσω μιας συνάρτησης ενεργοποίησης και διοχετεύεται στο επόμενο επίπεδο.

Χωρίς την ύπαρξη των συναρτήσεων ενεργοποίησης, το σήμα εξόδου θα ήταν μια γραμμική συνάρτηση, περιορίζοντας το σύστημά μας στα πλαίσια των γραμμικών ταξινομητών, αδυνατώντας να εξάγει συμπεράσματα υψηλού επιπέδου. Οι συναρτήσεις ενεργοποίησης θα πρέπει επίσης να είναι καλώς διαφορίσιμες για λόγους που θα εξηγήσουμε στην αμέσως επόμενη παράγραφο. Παρακάτω παρατίθενται μερικές ευρέως χρησιμοποιούμενες συναρτήσεις ενεργοποίησης.

2.2.2.1 Σιγμοειδής Συνάρτηση (Sigmoid)

Ο μαθηματικό τύπος της σιγμοειδής συνάρτησης καθώς η γραφική της απεικόνιση είναι οι εξής:

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



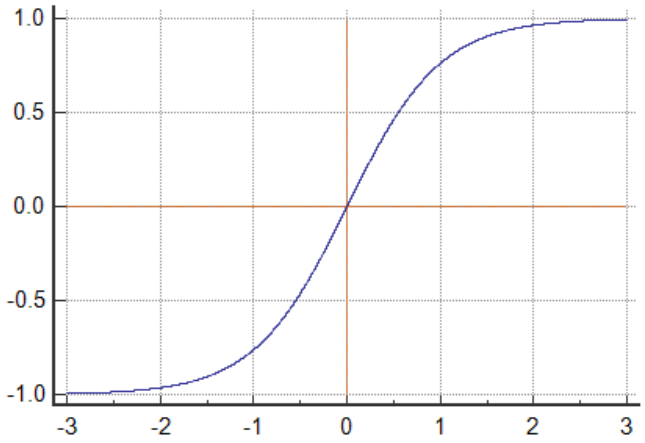
Σχήμα 2.4: Σιγμοειδής Συνάρτηση

Όπως παρατηρούμε πρόκειται για μία μη γραμμική συνάρτηση. Στα πλαίσια ενός νευρωνικού δικτύου, η σιγμοειδής λαμβάνει μία είσοδο και την κανονικοποιεί μεταξύ των τιμών 0 και 1. Αν και χρησιμοποιείται εκτενώς στα δίκτυα [Han95], χάριν της ερμηνεία της ως ρυθμός πυροδότησης ενός νευρώνα, παρουσιάζει μερικά σημαντικά μειονεκτήματα. Η παράγωγός της σε ακραία τιμές τείνει να

μηδενιστεί οπότε περιορίζει τη διαδικασία μάθησης. Ακόμα, δεν είναι συμμετρική ως προς το 0 ενώ και η διαδικασία ύψωσης σε δύναμη έχει μεγάλο υπολογιστικό κόστος.

2.2.2.2 Tanh

Η γραφική της απεικόνιση της tanh είναι η εξής:



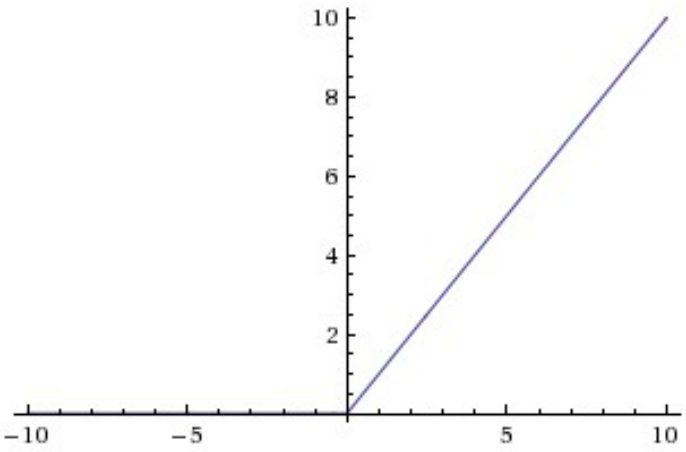
Σχήμα 2.5: Η Συνάρτηση tanh

Η tanh αν και είναι συμμετρική ως προς το 0 [LeCu91] ακόμα αντιμετωπίζει το ίδιο πρόβλημα με την σιγμοειδή στις ακραίες τιμές.

2.2.2.3 Ανορθωμένη Γραμμική Μονάδα (ReLU)

Η συγκεκριμένη συνάρτηση έχει αποκτήσει ιδιαίτερη απήχηση με την πάροδο των χρόνων και δηλώνεται ως εξής:

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



Σχήμα 2.6: Η Συνάρτηση ReLU

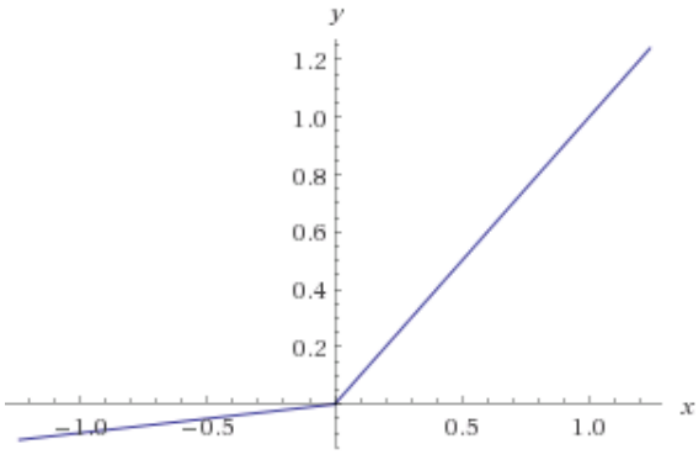
Σε σύγκριση με τις προηγούμενες συναρτήσεις, η ανορθωμένη γραμμική μονάδα έχει σημαντικές αναβαθμίσεις [Nair10]. Αν και δεν είναι συμμετρική, δεν αντιμετωπίζει το πρόβλημα στις θετικές

ακραίες τιμές όπου διατηρεί τις επιθυμητές της ιδιότητες. Σε συνδυασμό με την απλότητά της, διευκολύνει σε μεγάλο βαθμό την εκπαίδευση ενός συστήματος. Παρόλα αυτά, στις αρνητικές τιμές έχει το ίδιο πρόβλημα με τις 2 προηγούμενες. Πιο συγκεκριμένα παρουσιάζεται πιο έντονο καθώς η τιμή για αρνητικές τιμές είναι μόνιμα μηδέν ενώ εν αντιθέσει με πριν, η παράγωγος είναι και αυτή μηδέν. Αυτό έχει σαν αποτέλεσμα αν ένας νευρώνας σε οποιαδήποτε φάση της εκπαίδευσης εμφανίσει αρνητική τιμή, να παραμείνει ανενεργός σε όλη την υπόλοιπη διαδικασία.

2.2.2.4 Διακεκομμένη Ανορθωμένη Γραμμική Μονάδα (Leaky ReLU)

Η Leaky ReLU αποτελεί μετεξέλιξη της κλασικής ReLU που αναφέραμε από πανω. Η μαθηματική της έκφραση είναι η παρακάτω:

$$f(x) = \begin{cases} x, & \text{if } x > 0 \\ 0.01x, & otherwise \end{cases}$$



Σχήμα 2.7: Η Γραφική Παράσταση της Leaky ReLU

Η εξελιγμένη αυτή συνάρτηση ενεργοποίησης διατηρεί όλα τα πλεονεκτήματα της ReLU ενώ ταυτόχρονα επιλύει και το βασικό πρόβλημα που αντιμετώπιζε για αρνητικές τιμές [Maas13]. Αποτελεί πλέον βασικό εργαλείο στα προβλήματα μηχανικής μάθησης.

2.2.3 Συναρτήσεις Βελτιστοποίησης

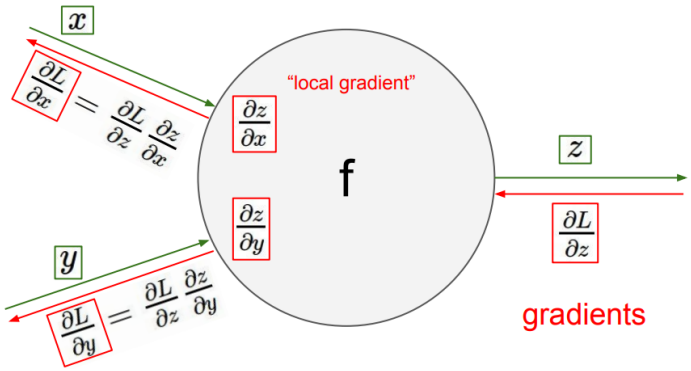
Αφού περιγράψαμε όλα τα δομικά στοιχεία ενός Τεχνητού Νευρωνικού Συστήματος, μπορούμε πλέον να αναλύσουμε σε μεγαλύτερο βάθος τη διαδικασία λειτουργίας του καθώς και τις τεχνικές βελτιστοποίησης των αποτελεσμάτων του.

2.2.3.1 Back Propagation

Ο αλγόριθμος *Ανάστροφης Διάδοσης (Back Propagation)* αποτελεί τη βασική λογική στην οποία εκπαιδεύονται τα συστήματα μηχανικής μάθησης. Η ιστορία του αλγορίθμου ξεκινά στη δεκαετία του 1960 ενώ παρουσιάστηκε επίσημα το 1976 από τον Seppo Linnainmaa [Linn76]. Όμως, αρχικά, δεν αναγνωρίστηκε η τεράστια αξία του για το χώρο. Ο αλγόριθμος ήρθε στην επιφάνεια μόλις το 1986 [Rume86] όπου μια σειρά από πειράματα έδειξαν ότι υπερέβαινε όλους τους τότε υπάρχοντες αλγορίθμους για την εκπαίδευση νευρωνικών δικτύων.

Αναλυτικότερα, ο αλγόριθμος λειτουργεί με τα εξής βήματα. Μόλις ολοκληρωθεί η διαδικασία και εξαχθούν τα αποτελέσματα, το σύστημά μας υπολογίζει το σφάλμα με βάση κάποια συνάρτηση

σφάλματος όπως αυτές που αναφέραμε στις προηγούμενες παραγράφους. Έπειτα, υπολογίζει την μερική παράγωγο της συνάρτησης σφάλματος ως προς την είσοδο. Μετά, με τον κανόνα της αλυσίδας υπολογίζονται τα σφάλματα σε όλα τα προηγούμενα επίπεδα μέχρι την είσοδο, τέλος μιας εποχής. Γνωρίζοντας πια όλες τις μερικές παραγώγους, μπορούμε να ανανεώσουμε τα βάρη και την πόλωση σε όλους τους νευρώνες όλων των επιπέδων προς την κατεύθυνση της μείωσης του σφάλματος εξόδου. Τώρα γίνεται εμφανές γιατί θέλαμε οι συναρτήσεις ενεργοποίησης που αναφέραμε να είναι διαφοροποιήσιμες.



Σχήμα 2.8: Σχηματική παρουσίαση του αλγορίθμου back propagation σε ένα νευρώνα. Με πράσινες γράμμες παρουσιάζεται η εμπρόσθια διάδοση ενώ με κόκκινες η ανάστροφη.

Στις επόμενες παραγράφους θα αναλύσουμε διαφορετικούς τρόπους αξιοποίησης των πληροφοριών που λαμβάνουμε από την ανάστροφη διάδοση.

2.2.3.2 Gradient Descent

Αποτελεί ουσιαστικά την απλή εφαρμογή των όσων περιγράψαμε παραπάνω. Πρόκειται για έναν επαναληπτικό αλγόριθμο βελτιστοποίησης που εγγυάται την εύρεση ελαχίστων. Θεωρείται το θεμέλιο στην εκπαίδευση των νευρωνικών συστημάτων. Είναι μέθοδος πρώτης τάξης, δηλαδή αξιοποιεί για την επεξεργασία μόνο τις πρώτες μερικές παραγώγους της συνάρτησης σφάλματος. Το διάνυσμα των παραγώγων δείχνει την κατεύθυνση που αυξάνεται η τιμή της συνάρτησης σφάλματος οπότε το βήμα κάθε φορά γίνεται προς την αντίθετη κατεύθυνση.

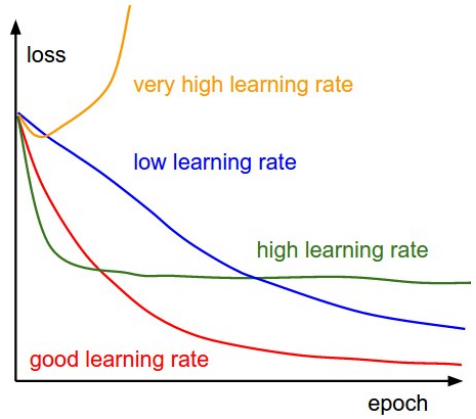
Αφού βρεθεί η κατεύθυνση που πρέπει να κινηθούν οι παράμετροι, είναι απαραίτητο να προσδιοριστεί το βήμα μεταβολής τους. Το ρόλο αυτόν τον διαδραματίζει μια υπερπαράμετρος ο *ρυθμός εκμάθησης (learning rate)*. Η επιλογή του ρυθμού εκμάθησης καθίσταται κομβική για την εκπαίδευση ενός συστήματος. Υψηλός ρυθμός εκμάθησης μπορεί να φέρει πρόσκαιρα σωστά αποτελέσματα αλλά μπορεί να οδηγήσει σε ακατάλληλη συμπεριφορά, με το ελάχιστο να ξεπερνιέται συνεχώς. Αντίθετα, χαμηλός ρυθμός εκμάθησης, αν και προσφέρει ασφάλεια στο σύστημα ότι θα βρει το ελάχιστο, έχει ως αποτέλεσμα την εκθετική αύξηση του υπολογιστικού κόστους του συστήματος και του χρόνου σύγκλισης.

Οι πληροφορίες για τους παρακάτω αλγόριθμους αντλήθηκαν από το [Rude16].

2.2.3.3 Vanilla Gradient Descent

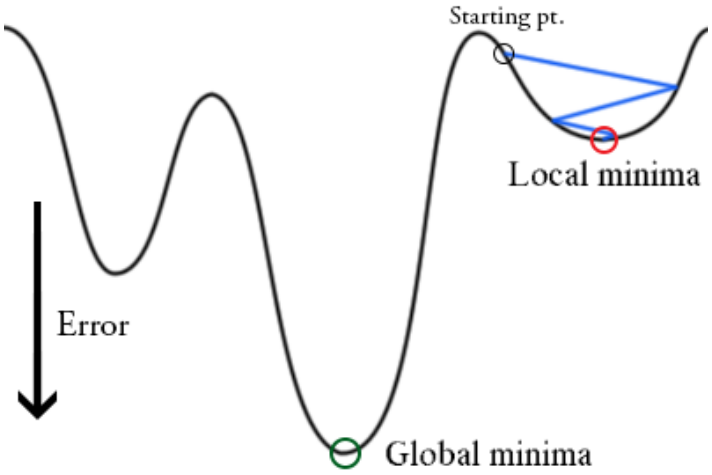
Η πιο απλή υλοποίηση αυτής της λογικής αποτελεί ο αλγόριθμος *Vanilla (Batch) Gradient Descent* όπου υπολογίζει τις μερικές παραγώγους των παραμέτρων για ολόκληρο το σύνολο των δεδομένων. Θεωρώντας τις παραμέτρους W, το ρυθμό εκμάθησης λ και την συνάρτηση σφάλματος J, η ανανέωση των παραμέτρων εκφράζεται μαθηματικά ως εξής:

$$W = W - \lambda \cdot \nabla_W J(W)$$



Σχήμα 2.9: Σχηματική παρουσίαση της επίδρασης του ρυθμού εκμάθησης σε ένα σύστημα μηχανικής μάθησης.

Αν και παράγει σταθερό σφάλμα κλίσης, ο αλγόριθμος εμφανίζει 2 σημαντικά μειονεκτήματα που οδήγησαν στη μετεξέλιξη του. Από την στιγμή που λαμβάνει υπόψη του όλα τα δεδομένα για τον υπολογισμό των μερικών παραγώγων, αυτό έχει σαν άμεση συνέπεια ο αλγόριθμος να είναι δύσχρηστος για μεγάλα σύνολα δεδομένων, σύνολα που πλέον εμφανίζονται κατά κόρον στη μηχανική μάθηση. Επιπροσθέτως, αν και καλύτερος σε κυρτές επιφάνειες, είναι ευάλωτος σε προβλήματα που εμφανίζουν τοπικά ελάχιστα καθώς περιορίζεται σε αυτόν τον χώρο, μη έχοντας τη δυνατότητα να εντοπίσει το ζητούμενο ολικό ελάχιστο.



Σχήμα 2.10: Το πρόβλημα εντοπισμού ολικού ελαχίστου για τον vanilla gradient descent.

2.2.3.4 Stochastic Gradient Descent

Ο αλγόριθμος *Stochastic Gradient Descent (SGD)* έρχεται να λύσει το πρόβλημα με τους περιττούς υπολογισμούς του προηγούμενο αλγορίθμου. Ο SGD ανανεώνει τις παραμέτρους για κάθε ένα από τα παραδείγματα εκπαίδευσης του συνόλου δεδομένων. Η ενημέρωση γίνεται για παράδειγμα εκπαίδευσης $x^{(i)}$ και ετικέτα $y^{(i)}$ ως εξής:

$$W = W - \lambda \cdot \nabla_W J(W, x^{(i)}, y^{(i)})$$

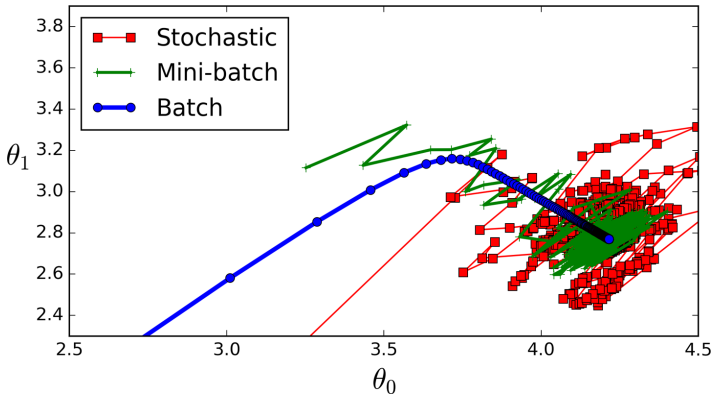
Οι συχνά παρόμοιες παράγωγοι, πριν από κάθε ενημέρωση, εξαλείφονται, αυξάνοντας κατακόρυφα τη χρονική αποδοτικότητα του συστήματος μας. Αυτό μετατρέπει τον SGD σε κατάλληλο αλγόριθμο για μηχανική μάθηση εξαιτίας του μεγάλου όγκου των δεδομένων. Οι συνεχείς ενημερώσεις όμως, αξιοποιώντας λιγότερες πληροφορίες, μπορεί να κάνει τον αλγόριθμο ευάλωτο σε ακραία δεδομένα εισόδου. Οπότε οι ενημερώσεις χαρακτηρίζονται εδώ από έντονο θόρυβο. Αν και φαινομενικά αυτό αποτελεί μειονέκτημα του αλγορίθμου, στην πραγματικότητα αποτελεί ένα ακόμα ατού του. Ο θόρυβος που εντοπίζεται στις ενημερώσεις, βοηθά τη μέθοδο αυτή να επιλύσει και το δεύτερο πρόβλημα του vanilla, καταφέρνοντας να ξεφύγει από τα τοπικά ελάχιστα που παγιδευόταν. Πειράματα, επίσης, απέδειξαν ότι η σταδιακή μείωση του ρυθμού εκμάθησης, οδηγεί το σύστημα σε ακόμα καλύτερα αποτελέσματα.

2.2.3.5 Mini-Batch Gradient Descent

Ο αλγόριθμος *Mini-Batch Gradient Descent* αποτελεί συνδυασμό των 2 παραπάνω. Τα δεδομένα χωρίζονται σε μικρές παρτίδες (batches) από n παραδείγματα εκπαίδευσης και η διαδικασία μάθησης συντελείται για κάθε μία παρτίδα:

$$W = W - \lambda \cdot \nabla_W J(W, x^{(i:i+n)}, y^{(i:i+n)})$$

Με αυτόν τον τρόπο ο θόρυβος περιορίζεται αισθητά, χωρίς όμως να χαθεί η ικανότητα εντοπισμού καλύτερων ελαχίστων. Αποτελεί τη βασική τεχνική μάθηση από τις 3 και χρησιμοποιείται ευρέως. Η εκτενής χρήση του έχει οδηγήσει στη χρήση του όρου SGD ακόμα και όταν εφαρμόζεται η διαδικασία με τις μικρές παρτίδες.

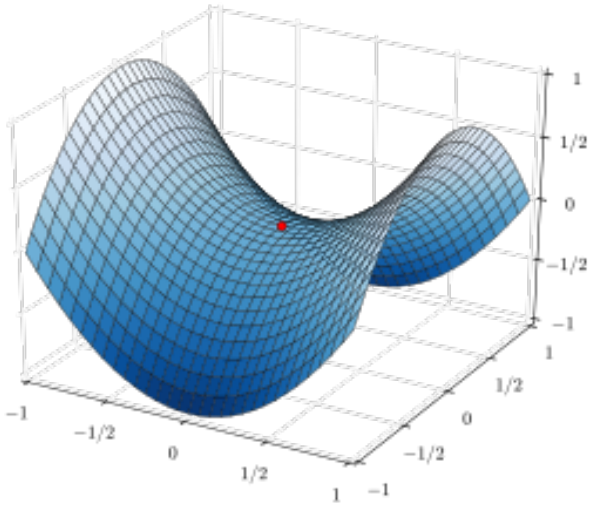


Σχήμα 2.11: Εποπτική σύγκριση των 3 αλγορίθμων σε **κυρτή** επιφάνεια.

2.2.3.6 Momentum

Η εξέλιξη όμως των αλγορίθμων βελτιστοποίησης είναι συνεχής και δεν σταμάτησε στον SGD. Βασικό πρόβλημα του SGD αποτελούν οι περιπτώσεις των λεγόμενων *saddle points*, περιοχών δηλαδή που η μια διάσταση έχει πολύ μεγαλύτερη κλίση από την άλλη. Οι περιοχές αυτές είναι πολύ συνηθισμένες σε προβλήματα πολλών διαστάσεων και ο SGD χρονοτριβεί κάνοντας κινήσεις προς τη λάθος διάσταση.

Η *ορμή-momentum* αποτέλεσε μιας πρώτης τάξης προσέγγιση στην επίλυση αυτού του ζητήματος, κατευθύνοντας την παράγωγο πιο γρήγορα προς της σωστή κατεύθυνση. Αυτό επιτυγχάνεται με



Σχήμα 2.12: Saddle point.

την πρόσθεση ενός νέου όρου ορμής στο διάνυσμα ενημερώσεων v_t, που παίζει το ρόλο της ταχύτητας, χρησιμοποιώντας το προηγούμενο βήμα πολλαπλασιασμένο με ένα παράγοντα α, που συνήθως παίρνει την τιμή 0,9 ή 0,99:

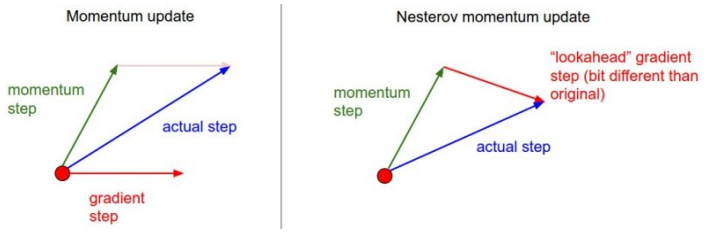
$$v_{t+1} = \alpha \cdot v_t + \lambda \cdot \nabla_{W_t} J(W_t)$$

$$W_{t+1} = W_t - v_{t+1}$$

Παραλλαγμένη εκδοχή του βασικού αλγορίθμου της ορμής-momentum αποτέλεσε ο αλγόριθμος *Nesterov Momentum*. Έχει ακόμα πιο ισχυρό μαθηματικό υπόβαθρο ενώ και στην πράξη έχει ελαφρώς καλύτερη συμπεριφορά. Η βασική ιδέα πίσω από τον αλγόριθμο είναι πως δίνεται η δυνατότητα να προβλέπει την επόμενη κίνηση του, όχι μόνο με βάση τη θέση που βρίσκεται, αλλά χρησιμοποιώντας την επόμενη πιθανή του θέση. Αυτό γίνεται λαμβάνοντας υπόψη στην παράγωγο εκτός από τη θέση των παραμέτρων και την προϋπάρχουσα ταχύτητα. Συνοπτικά:

$$v_{t+1} = \alpha \cdot v_t + \lambda \cdot \nabla_{W_t} J(W_t - \alpha \cdot v_t)$$

$$W_{t+1} = W_t - v_{t+1}$$



Σχήμα 2.13: Σύγκριση βήματος Nesterov Momentum με κλασικό Momentum

2.2.3.7 Adagrad / RMSprop

Μία διαφορετική προσέγγιση στους αλγορίθμους βελτιστοποίησης έχει αποκτήσει απήχηση τα τελευταία χρόνια. Βασικό ζήτημα στους παραπάνω αλγορίθμους είναι η σταδιακή μείωση του ρυθμού εκμάθησης για την επίτευξη καλύτερων αποτελεσμάτων. Έχουν προταθεί στο παρελθόν οι επιλογές του υποδιπλασιασμού του ύστερα από ένα αριθμό εποχών ή της εκθετικής μείωσης με την πάροδο του χρόνου. Οι αλγόριθμοι *Adagrad* και *RMSprop* απορρίπτουν τις παραπάνω λογικές. Πιο συγκεκριμένα, αν μία παράμετρος έχει χαρακτηριστικά με υψηλή συσχέτιση πρέπει ο ρυθμός εκμάθησης να είναι μικρός, ενώ σε χαρακτηριστικά με χαμηλή συσχέτιση ο ρυθμός εκμάθησης οφείλει να είναι μεγάλος. Η ύπαρξη διαφορετικών ρυθμών εκμάθησης επιτυγχάνεται με την ύπαρξη ενός βασικού ρυθμού, ο οποίος διαιρείται με ένα παράγοντα που λαμβάνει υπόψη την παράγωγο της κάθε παραμέτρου. Αν η παράγωγος είναι μεγάλη τότε ο ρυθμός μικραίνει και αντίστροφα:

$$G_{t+1,jj} = G_{t,jj} + \nabla J_W(W_{t+1,j})^2$$

$$W_{t+1}, j = W_{t,j} - \frac{\lambda}{\sqrt{G_{t+1,jj} + \epsilon}} \cdot \nabla J_W(W_{t+1,j})$$

Ο παράγοντας ϵ λαμβάνει τιμή απο 1e-8 εώς 1e-4 και αποτρέπει τη διαίρεση με το 0

Ο αλγόριθμος RMSprop κινείται στην ίδια λογική με τον Adagrad, με τη διαφορά ότι επιλύει ένα πρόβλημα που προέκυψε. Επειδή το πρώτο βήμα του Adagrad είναι συνεχώς αυξητικό, αφού ο προηγούμενος όρος προστίθεται κάθε φορά, ο ρυθμός εκμάθησης συνεχώς μειώνεται, αδυνατώντας να προσαρμοστεί κατάλληλα στις νέες καταστάσεις του χώρου. Στην εκπαίδευση εδώ, ο ρυθμός εκμάθησης επηρεάζεται από τις παραγώγους των τελευταίων n βημάτων. Έτσι ο ρυθμός εκμάθησης μπορεί να προσαρμοστεί πιο εύκολα στις ανάγκες της κάθε στιγμής. Αυτό εκφράζεται μέσο ενός κυλιόμενου μέσου όρου των τελευταίων n βημάτων, ο οποίος πολλαπλασιάζεται με ένα παράγοντα γ που συνήθως παίρνει τιμή 0,9 ή 0,99 ή 0,999:

$$E[g^2]_{t+1} = \gamma \cdot E[g^2]_t + (1 - \gamma) \cdot (\nabla J_W(W_{t+1})^2)$$

$$W_{t+1} = W_t - \frac{\lambda}{\sqrt{E[g^2]_{t+1} + \epsilon}} \cdot \nabla J(W_{t+1})$$

2.2.3.8 Adam

Ο τελευταίος αλγόριθμος βελτιστοποίησης που θα αναφερθούμε είναι ο *Adam*. Ο Adam μοιάζει σαν τον RMSprop που παράλληλα κάνει χρήση της ορμής:

$$m_{t+1} = \beta_1 \cdot m_t + (1 - \beta_1) \cdot (\nabla J_W(W_{t+1}))$$

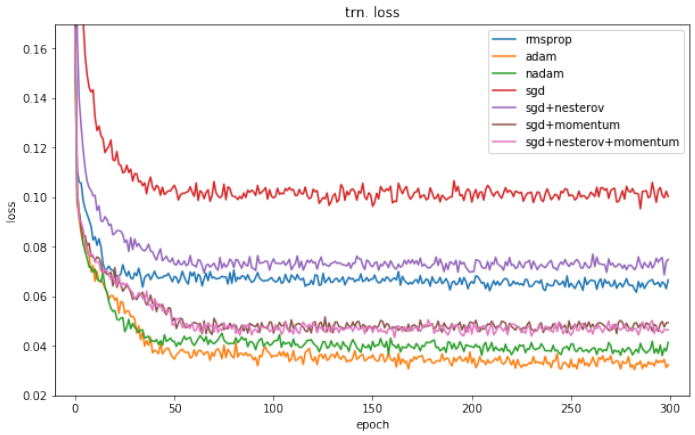
$$v_{t+1} = \beta_2 \cdot v_t + (1 - \beta_2) \cdot (\nabla J_W(W_{t+1})^2)$$

$$W_{t+1} = W_t - \frac{\lambda}{\sqrt{v_{t+1}} + \epsilon} \cdot m_{t+1}$$

Οι παράμετροι β_1 και β_2 παίρνουν τιμές 0,9 και 0,999, ενώ οι μεταβλητές αρχικοποιούνται στο 0. Για αυτό το λόγο οι μεταβλητές m και v στα πρώτα βήματα της εκπαίδευσης δυσκολεύονται να πάρουν τις σωστές τιμές αφού τείνουν στο 0. Το πρόβλημα αυτό επιλύεται με τη χρήση πολώσεων και η ολοκληρωμένη μορφή του Adam είναι η εξής:

$$m_{t+1} = \beta_1 \cdot m_t + (1 - \beta_1) \cdot (\nabla J_W(W_{t+1}))$$
$$v_{t+1} = \beta_2 \cdot v_t + (1 - \beta_2) \cdot (\nabla J_W(W_{t+1})^2)$$
$$\hat{m}_{t+1} = \frac{m_t}{1 - \beta_1^{t+1}}$$
$$\hat{v}_{t+1} = \frac{v_t}{1 - \beta_2^{t+1}}$$
$$W_{t+1} = W_t - \frac{\lambda}{\sqrt{\hat{v}_{t+1}} + \epsilon} \cdot \hat{m}_{t+1}$$

Ο αλγόριθμος Adam, μαζί με τις τροποποιήσεις του (Nadam et.), πετυχαίνει πολύ καλά αποτελέσματα. Πλέον, τα περισσότερα συστήματα χρησιμοποιούν σαν βάση τους τον Adam ή τον SGD+Nesterov Momentum.



Σχήμα 2.14: Σύγκριση των αλγορίθμων βελτιστοποίησης. Παρατηρούμε την ανωτερότητα των εκδόσεων του Adam αλλά του SGD με ορμή έναντι των άλλων μοντέλων

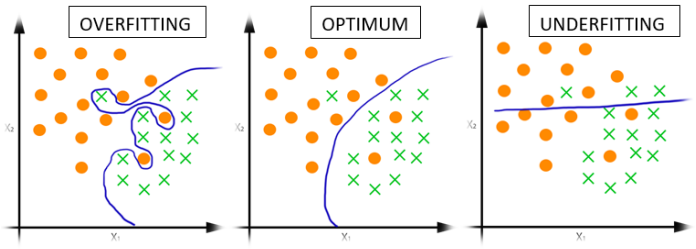
2.2.4 Υπερπροσαρμογή

Με την εξέλιξη των τεχνικών βελτιστοποίησης, ο τομέας της μηχανικής εκμάθησης έχει εκτελέσει άλματα προόδου. Η αξιοποίησή τους όμως τυφλά μπορεί να οδηγήσει σε δυσμενή αποτελέσματα. Η εκπαίδευση ενός μοντέλου μηχανικής εκμάθησης γίνεται με βάση ένα σύνολο δεδομένων (dataset). Για να είναι πετυχημένη μία εκπαίδευση, πρέπει το εκπαιδευμένο πια σύστημα να μπορεί να ανταποκριθεί εξίσου καλά, δεχόμενο νέα δεδομένα εκτός του dataset εκπαίδευσης. Εξαιτίας της μεγάλης διαθέσιμης υπολογιστικής ισχύς, το σύστημα εγκυμονεί να υποπέσει στο φαινόμενο της *Υπερπροσαρμογής (Overfitting)* [Domi12].

Με τον όρο υπερπροσαρμογή, όπως μαρτυράει το όνομα, εννοούμε τη διαδικασία κατά την οποία το σύστημα έχει προσαρμόσει πλήρως τις παραμέτρους στα δεδομένα εκπαίδευσης, με αποτέλεσμα την αδυναμία γενίκευσης του. Για να αξιολογηθεί η ύπαρξη υπερπροσαρμογής σε ένα μοντέλο, τα διαθέσιμα δεδομένα χωρίζονται σε τρία σύνολα. Το πρώτο σύνολο ονομάζεται *train set* το οποίο αποτελεί και το μεγαλύτερο μέρος των δεδομένων. Με βάση αυτό το σύνολο διεξάγεται η εκπαίδευση και ενημερώνονται οι παράμετροι του συστήματος. Το δεύτερο και πολύ μικρότερο σύνολο είναι το *validation set* και χρησιμοποιείται περιστασιακά στην εκμάθηση . Βασική χρήση του είναι ο σωστός συντονισμός των υπερπαραμέτρων του μοντέλου. Μπορεί να αξιοποιηθεί και σαν δείκτης του βαθμού υπερπροσαρμογής καθώς αυξανόμενα ποσοστά σφάλματος στο validation set μας καθοδηγούν σε αύξηση των τεχνικών ομαλοποίησης που θα αναφερθούν στην επόμενη παράγραφο. Το τρίτο σύνολο

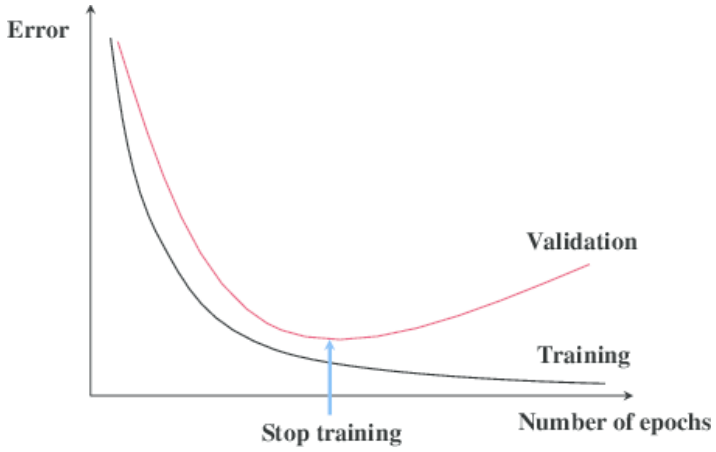
είναι το *test set* το οποίο είναι ο τελικός δείκτης αξιολόγησης του μοντέλου. Χρησιμοποιείται μόνο μετά την ολοκλήρωση της διαδικασίας εκπαίδευσης και κρίνει αμερόληπτα την επιτυχία του συστήματος. Αν το ποσοστό σφαλμάτων στο test set προσεγγίζει το αντίστοιχο ποσοστό στην εκπαίδευση τότε το σύστημα θεωρείται πως απέφυγε την υπερπροσαρμογή ενώ αν αποκλίνει σημαντικά αποτελεί απόδειξη λανθασμένης εκπαίδευσης. Βασικό χαρακτηριστικό των 3 συνόλων πρέπει να είναι η όσο δυνατόν πιο κοινή κατανομή στοιχείων.

Στον αντίποδα, *Υποπροσαρμογή (Underfitting)* έχουμε όταν ένα σύστημα δεν είναι όσο πολύπλοκο χρειάζεται για να ανιχνεύσει τα επιθυμητά χαρακτηριστικά. Το πρόβλημα αυτό είναι αρκετά προφανές και πολύ πιο εύκολα ανιχνεύσιμο.



Σχήμα 2.15: Παράδειγμα υπερπροσαρμογής και υπόπροσαρμογής

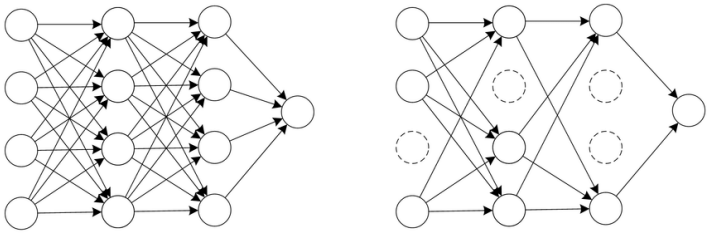
Ο χωρισμός στα 3 παραπάνω υποσύνολα μας βοηθάει να ανιχνεύσουμε την υπερπροσαρμογή. Μια σειρά από τεχνικές όμως έχουν αναπτυχθεί για τη μετέπειτα αντιμετώπισή της. Βασικό εργαλείο αποτελεί η χρήση ομαλοποιήσεων (regularization) στην συνάρτηση σφάλματος, δηλαδή όρων αναλόγων της πολυπλοκότητας των παραμέτρων. Έτσι μια εκτενής αύξηση της πολυπλοκότητας θα επιφέρει και μεγάλη αύξηση στην συνάρτηση σφάλματος και δεν θα επιλεχθεί, οδηγώντας σε πιο γενικευμένες λύσεις. Η συνεχής εναλλαγή μεταξύ train set και validation set, δηλαδή η επιλογή διαφορετικού validation set κάθε φορά, αν και έχει θετική επίδραση εμποδίζοντας το σύστημα να υπερπροσαρμοστεί δεν προτιμάται σε προβλήματα βαθιάς μάθησης εξαιτίας του μεγάλου υπολογιστικού κόστους. Αντίθετα, μια τεχνική που επιλέγεται σε τέτοια προβλήματα είναι η διακοπή της εκπαίδευσης του συστήματος (early stopping) όταν η αύξηση της πολυπλοκότητας προκαλεί μείωση του σφάλματος στο train set αλλά αύξηση στο validation set.



Σχήμα 2.16: Παράδειγμα early stopping για την αποφυγή υπερπροσαρμογής

Μια ακόμα τεχνική για την αποφυγή της υπερπροσαρμογής που δημοσιεύθηκε τη δεκαετία που διανύουμε, αποτελεί το Dropout [Sriv14]. Η λογική της είναι να κρατάει ενεργό μόνο ένα ορισμένο αριθμό νευρώνων, που αλλάζουν κάθε φορά, στη διάρκεια της εκπαίδευσης. Κάθε νευρώνας μένει ενεργός με μια πιθανότητα p, που αποτελεί υπερπαράμετρο του συστήματος. Παρά την απλότητά της μεθόδου και τη φαινομενικά δυσμενής συνέπεια που έχει, καθώς κάποιο μέρος του συστήματος δεν

χρησιμοποιείται κάθε φορά, η προσέγγιση αυτή δεν οδηγεί σε χειρότερα αποτελέσματα ενώ επειδή οι νευρώνες που είναι ενεργοί αλλάζουν, δεν επιτρέπει την υπερπροσαρμογή των παραμέτρων στα δεδομένα εκπαίδευσης, οδηγώντας σε πιο γενικευμένες λύσεις. Μελέτες απέδειξαν ότι η χρήση dropout, επιφέρει θετικότερα αποτελέσματα από άλλων μορφών ομαλοποιήσεις.



(a) Standard Neural Network (b) Network after Dropout

Σχήμα 2.17: Η χρήση dropout σε ένα νευρωνικό δίκτυο

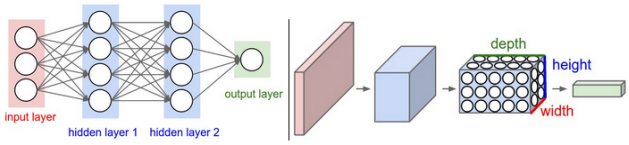
2.3 Συνελικτικά Νευρωνικά Δίκτυα

Η ιστορία των *Συνελικτικών Νευρωνικών Δικτύων (Convolutional Neural Networks CNN)* ξεκινάει το 1987 με το *Time Delay Neural Network (TDNN)* [Waib90] που αποτέλεσε το πρώτο συνελικτικό δίκτυο και χρησιμοποιήθηκε για αναγνώριση φωνής. Το 1989 ο Yann LeCun αξιοποίησε τον αλγόριθμο της ανάστροφης διάδοσης για την αυτόματη εκμάθηση μέσα από εικόνα. Το ολοκληρωμένο δίκτυο ονομάστηκε LeNet-5 και δημοσιεύτηκε από την ομάδα του το 1998 [LeCu98]. Το 2012 σχεδιάστηκε και δημοσιεύτηκε από τον Alex Krizhevksy et al. το δίκτυο AlexNet [Kriz12], το οποίο διαγωνίστηκε και κέρδισε στο διαγωνισμό ILSVRC. Η βασική καινοτομία του ήταν στο μεγάλο βάθος των στρωμάτων και από τότε το deep learning κυριάρχησε στην τεχνητή νοημοσύνη και κατά επέκταση στα συνελικτικά δίκτυα και στην αναγνώριση εικόνων.

Ένα συνελικτικό δίκτυο διατηρεί όλα τα βασικά χαρακτηριστικά των τεχνητών νευρωνικών δικτύων που έχουμε αναλύσει. Η καινοτομία των αρχιτεκτονικών αυτών έγκειται στο γεγονός ότι αντιμετωπίζουν την είσοδο σαν εικόνες ή βίντεο ως εκ τούτου μπορούν να εκμεταλλευτούν κάποιες εφαρμόσιμες σε αυτές ιδιότητες για την αποδοτικότερη εμπρόσθια διάδοση, με βασική την ιδιότητα της συνέλιξης.

Η κλασική μορφή των νευρωνικών δικτύων αποτελείται όπως έχουμε πει από κρυφά στρώματα, τα οποία έχουν νευρώνες και συνδέονται με όλους τους νευρώνες του προηγούμενου επιπέδου. Σε προβλήματα αναγνώρισης και κατάτμησης εικόνας το παραπάνω σύστημα βρίσκεται αντιμέτωπο με το πολύ σημαντικό πρόβλημα της κλιμάκωσης. Το CIFAR-10 είναι ένα μικρό dataset με εικόνες πολύ χαμηλής ανάλυσης (32x32) το οποίο χρησιμοποιείται ευρύτατα για ερευνητικούς σκοπούς στον τομέα της μηχανικής μάθησης. Ένας απλός νευρώνας για την αναγνώριση κλάσεων θα χρειαζόταν $32 \times 32 \times 3$ (για τα 3 κανάλια χρώματος) δηλαδή 3072 παραμέτρους. Αν και ο αριθμός μοιάζει διαχειρίσιμος για την παρούσα υπολογιστική ισχύ, πρέπει να αναλογιστούμε ότι ένας νευρώνας και ένα μόνο κρυφό στρώμα δεν είναι αρκετό για τα προβλήματα ταξινόμησης. Αντίθετα σε σύγχρονα dataset που χρησιμοποιούνται σε μεγάλους διαγωνισμούς όπως ο ILSVRC, το τυπικό μέγεθος μιας εικόνας είναι 482×415 pixels που μας οδηγεί 600.090 παραμέτρους για ένα μόλις νευρώνα. Είναι προφανές ότι σε τέτοια προβλήματα η παραδοσιακή μορφή των νευρωνικών δικτύων καθίσταται άχρηστη.

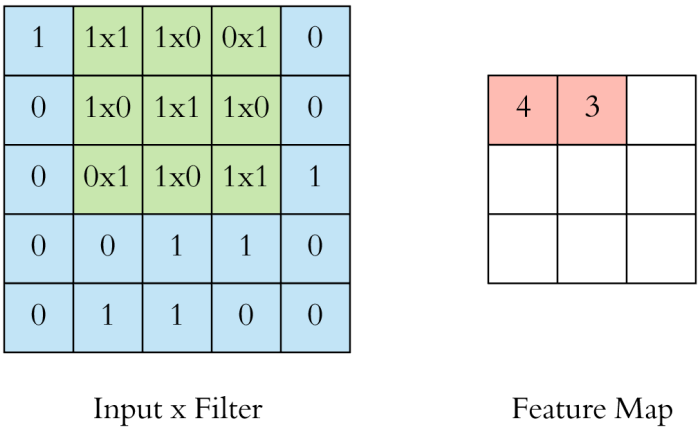
Μέσω της εφαρμογής της συνέλιξης, τα δίκτυα αυτά μπορούν να τροποποιήσουν τα δεδομένα σε πιο βολικές δομές, χωρίς να χαθεί σημαντική πληροφορία. Με αυτό τον τρόπο, το δίκτυο αποδεσμεύεται από την πλήρη συνεκτικότητα των νευρώνων με το προηγούμενο επίπεδο που είναι υπεύθυνη για την εκθετική αύξηση των παραμέτρων του συστήματος. Παρακάτω θα αναλύσουμε βασικά στρώματα/επίπεδα ενός συνελικτικού δικτύου.



Σχήμα 2.18: Σύγκριση ενός παραδοσιακού νευρωνικού συστήματος με ένα συνελικτικό δίκτυο

2.3.1 Συνελικτικό Επίπεδο (Convolutional Layer)

Βασικός στόχος ενός *Συνελικτικού Επιπέδου (Convolutional Layer)* αποτελεί η εξαγωγή χαρακτηριστικών από μια εικόνα εισόδου. Για να γίνει κατανοητή διαδικασία εξαγωγής πρέπει πρώτα να εξηγηθεί η έννοια του συνελικτικού φίλτρου. Το φίλτρο είναι ένας μικρός, σχετικά με την είσοδο, τετραγωνικός πίνακας αλλά με βάθος όσο η αρχική εικόνα, δηλαδή 3 διαστάσεων όσα και τα κανάλια χρώματος σε μια έγχρωμη εικόνα. Το φίλτρο διατρέχει διαδοχικά την αρχική εικόνα φτιάχνοντας σταδιακά ένα νέο πίνακα που ονομάζεται χάρτης χαρακτηριστικών. Ο χάρτης χαρακτηριστικών σχηματίζεται ουσιαστικά με συνεχόμενο πολλαπλασιασμό πινάκων, δηλαδή της συνέλιξης 2 πινάκων.



Σχήμα 2.19: Διαδικασία συνέλιξης: Ο μπλέ πίνακας εκφράζει την είσοδο, ο πράσινος το φίλτρο και δεξιά βλέπουμε να σχηματίζεται ο χάρτης χαρακτηριστικών

Όπως προείπαμε, όταν τα δεδομένα είναι υψηλής ανάλυσης, η πλήρης σύνδεση των νευρώνων του προηγούμενο στρώματος με τους νευρώνες του επόμενου καθιστά το δίκτυο δύσχρηστο. Έτσι κάθε νευρώνας συνδέεται με ένα μόνο τμήμα του προηγούμενου στρώματος. Η έκταση αυτή εκφράζεται με το *πεδίο υποδοχής (respective field)*, δηλαδή του μεγέθους του φίλτρου, και αποτελεί υπερπαράμετρο του δικτύου. Με είσοδο μία εικόνα από το CIFAR-10 (32x32x3), με φίλτρο 5x5 θα είχαμε $5 \times 5 \times 3 = 75$ παραμέτρους, ενώ αν ήταν πλήρως συνδεδεμένο $32 \times 32 \times 3 = 3072$ παραμέτρους για κάθε νευρώνα.

Πέρα από το μέγεθος του φίλτρου, υπάρχουν και άλλοι παράγοντες που καθορίζουν τον όγκο εξόδου από ένα συνελικτικό επίπεδο. Ο *αριθμός των φίλτρων (depth)* είναι ένας από αυτούς. Παραλληλίζοντας με ένα παραδοσιακό νευρωνικό δίκτυο, ο αριθμός των φίλτρων εκφράζει τον αριθμό των νευρώνων σε ένα στρώμα. Η ύπαρξη πολλών διαφορετικών φίλτρων, αν και κοστίζει υπολογιστικά, μεγαλώνει τον όγκο εξόδου και συμβάλλει στην εξαγωγή περισσότερων χαρακτηριστικών. Το *βήμα (stride)* καθορίζει κατά πόσο θα ολισθαίνει κάθε φορά το φίλτρο πάνω στον πίνακα εισόδου. Μικρή ολίσθηση οδηγεί σε μεγαλύτερη έξοδο και αντίστροφα. Επίσης, σε ορισμένες περιπτώσεις, για να μη χαθούν τυχόν χαρακτηριστικά που βρίσκονται κοντά στο πλαίσιο της εικόνας, εφαρμόζεται *γέμισμα (padding)*, συνήθως με μηδενικά (zero padding), στο περίγραμμα της. Τόσο το padding, όσο και το depth και το stride αποτελούν υπερπαραμέτρους του δικτύου.

Συνοπτικά, δοθέντος μιας εισόδου με μέγεθος $W_1 \times H_1 \times D_1$, και εφαρμόζοντας K φίλτρα μεγέθους $F \times F \times D_1$, με βήμα S και padding P, ο όγκος εξόδου θα είναι μεγέθους $W_2 \times H_2 \times D_2$

όπου:

$$W_2 = \frac{W_1 - F + 2P}{S + 1}$$

$$H_2 = \frac{H_1 - F + 2P}{S + 1}$$

$$D_2 = K$$

Αναφέραμε πιο πριν ότι ο αριθμός των φίλτρων σε ένα επίπεδο ισοδυναμεί με τον αριθμό των νευρώνων κάτι που δεν είναι απόλυτα ακριβές. Κάθε φίλτρο από το συνολικό αριθμό των φίλτρων, επιτρέπεται να έχει διαφορετικές παραμέτρους σε κάθε βήμα της συνέλιξη, αυξάνοντας υπερβολικά τον αριθμό των παραμέτρων. Αυτό, όμως, στην πράξη δεν συμβαίνει αφού ένα συνελικτικό δίκτυο χρησιμοποιεί σύστημα διαμοιρασμού παραμέτρων. Ουσιαστικά, ο ρόλος των φίλτρων είναι η ανίχνευση συγκεκριμένων χαρακτηριστικών (ακμές, γωνίες) , με αποτέλεσμα σε περίπτωση που βρεθεί σε μία θέση, να είναι θεμιτό να βρεθεί και σε μία διαφορετική. Έτσι ακολουθείται η τακτική διαμοιρασμού των παραμέτρων με σκοπό την μείωση του υπολογιστικού κόστους.

Μετά τα συνελικτικά επίπεδα, οι χάρτες χαρακτηριστικών περνάνε από τα λεγόμενα επίπεδα ενεργοποίησης. Οι πιο γνωστές συναρτήσεις ενεργοποίησης αναφέρθηκαν σε προηγούμενη παράγραφο μαζί με τα εκάστοτε προβλήματά τους. Στα συνελικτικά νευρωνικά δίκτυα χρησιμοποιούνται οι ReLU ή κυρίως κάποια παραλλαγή της όπως η Leaky ReLU που έχουμε αναλύσει.

2.3.2 Επίπεδο Συγκέντρωσης (Pooling Layer)

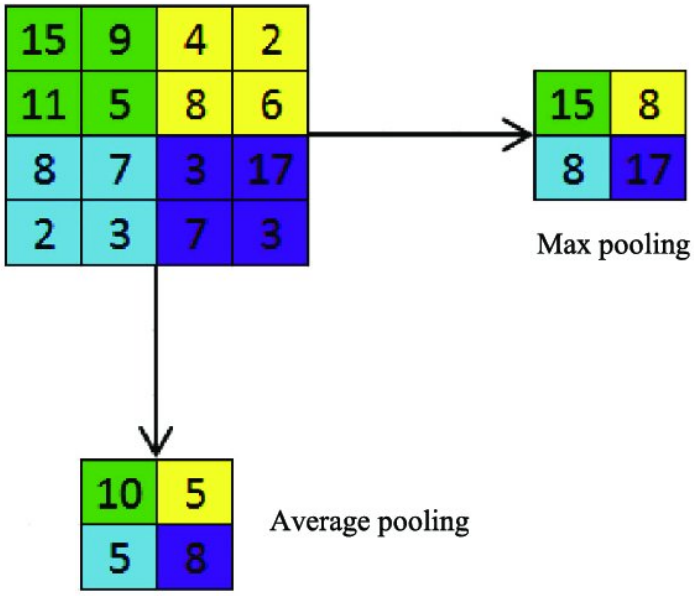
Κοινή τακτική αποτελεί η προσθήκη στρωμάτων δειγματοληψίας ανάμεσα στα συνελικτικά επίπεδα με στόχο τη μείωση των διαστάσεων του όγκου δεδομένων και κατά επέκταση των παραμέτρων και της πολυπλοκότητας του μοντέλου. Ονομάζονται *Επίπεδα Συγκέντρωσης (Pooling Layer)* και βοηθάνε επίσης στον έλεγχο της υπερπροσαρμογής. Το πιο γνωστό είδος δειγματοληψίας είναι το max-pooling όπου σε μια μικρή περιοχή, συνήθως 2x2, στην έξοδο τροφοδοτείται μόνο η μέγιστη τιμή που θεωρείται ότι περιέχει τη βασική πληροφορία. Μια διαφορετική προσέγγιση, που δεν χρησιμοποιείται τόσο, είναι το average pooling που τροφοδοτεί τη μέση τιμή της επιλεγμένης περιοχής. Παρατηρούμε ότι αυτή η μικρή δειγματοληψία είναι ικανή να μειώσει κατά 75% το μέγεθος των δεδομένων. Γενικά δοθέντος μιας εισόδου με μέγεθος $W_1 \times H_1 \times D_1$, και εφαρμόζοντας pooling μεγέθους $F \times F \times D_1$, με βήμα S, ο όγκος εξόδου θα είναι μεγέθους $W_2 \times H_2 \times D_2$ όπου:

$$W_2 = \frac{W_1 - F}{S + 1}$$

$$H_2 = \frac{H_1 - F}{S + 1}$$

$$D_2 = D_1$$

Με την πάροδο των χρόνων το συγκεκριμένο επίπεδο έχει αρχίσει να χάνει τη δημοφιλία του στο βωμό της απλότητας. Οι αρχιτεκτονικές με συνεχόμενα συνελικτικά επίπεδα έχουν αποκτήσει μεγαλύτερη απήχηση, ενώ σε ορισμένα προβλήματα μηχανικής μάθησης έχουν καλύτερα αποτελέσματα. Για τη βασική λειτουργία των επιπέδων που είναι η μείωση του όγκου δεδομένων, αντιπροτείνεται μεγαλύτερο βήμα ανά τακτά επίπεδα συνέλιξης.



Σχήμα 2.20: Σύγκριση δειγματοληψίας με max και average pooling

2.3.3 Πλήρως Συνδεδεμένο Επίπεδο (Fully-Connected Layer)

Οι νευρώνες σε ένα *Πλήρως Συνδεδεμένο Επίπεδο (Fully-Connected Layer)* έχουν συνδέσεις με όλους τους νευρώνες του προηγούμενου στρώματος. Πρόκειται στην ουσία για ένα επίπεδο που υπάρχει στα παραδοσιακά νευρωνικά δίκτυα. Σε συνελικτικά δίκτυα, έπειτα από την επεξεργασία που γίνεται από τα υπόλοιπα επίπεδα, συνήθως στο τέλος υπάρχει ένα πλήρως συνδεδεμένο επίπεδο που χρησιμοποιείται για την αξιοποίηση των χαρακτηριστικών που έχουν εξαχθεί και την ταξινόμηση τους στις διάφορες κλάσεις. Συνηθισμένη δομή ενός συνελικτικού δικτύου ταξινόμησης είναι η ακόλουθη:

$$[(CONV - RELU) \times N - POOL] \times M - (FC - RELU)$$

Με το N να λαμβάνει μικρές τιμές γύρω στο 5 και το M μεγάλες που εκφράζει το επιθυμητό βάθος του συστήματος.

2.3.4 Κανονικοποίηση Παρτιδών (Batch Normalization)

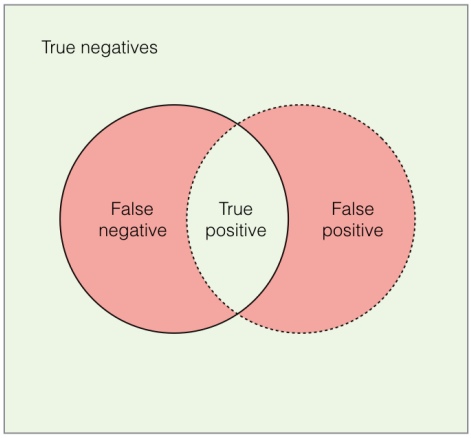
Μια τεχνική που εισήχθη πρόσφατα και χρησιμοποιείται κατά κόρον σε συνελικτικά δίκτυα αποτελεί η *Κανονικοποίηση Παρτιδών (Batch Normalization)* [Ioff15]. Στα νευρωνικά δίκτυα αποτελούσε συνήθης τακτική τα δεδομένα πριν την είσοδό τους στο δίκτυο να προεπεξεργάζονταν με σκοπό την αποφυγή δυσμενών αποτελεσμάτων κυρίως στα πρώτα στάδια της εκπαίδευσης. Αυτό γίνεται μέσω κανονικοποιήσης τους, δηλαδή περιορισμού των χαρακτηριστικών τους σε ορισμένες κλίμακες. Η προσέγγιση αυτή επεκτείνεται μέσω του Batch Normalization και στα ενδιάμεσα στάδια. Έπειτα από τα συνελικτικά επίπεδα και πριν τα επίπεδα ενεργοποίησης, οι χάρτες χαρακτηριστικών υπόκεινται σε κανονικοποίηση ανά παρτίδα. Αυτό επιτυγχάνεται μέσω αφαίρεσης της μέσης τιμής των χαρακτηριστικών και διαίρεση της με την τυπική απόκλιση. Έτσι τα δεδομένα ακολουθούν μια πιο ομαλή κατανομή που διευκολύνει τη διαδικασία εκμάθησης βελτιώνοντας την ακρίβεια των προβλέψεων ενώ η αξιοποίηση αυτών των χαρακτηριστικών ανά παρτίδα φέρει ευεργετικές συνέπειες και στην καταπολέμηση της υπερπροσαρμογής οδηγώντας σε πιο εύκολα γενικευμένες λύσεις.

2.4 Μετρικές Απόδοσης

Κλείνοντας το θεωρητικό υπόβαθρο της διπλωματικής εργασία οφείλουμε να κάνουμε αναφορά στις μετρικές συναρτήσεις. Η μετρική συνάρτηση αποτελεί μια συνάρτηση που χρησιμοποιείται για να αποτιμηθεί η απόδοση του μοντέλου. Όντας σε επιβλεπόμενη μάθηση, γνωρίζουμε τις πραγματικές τιμές των δεδομένων. Το μοντέλο μας, αφού εκπαιδευθεί, μπορεί να υλοποιήσει τις προβλέψεις του. Οι μετρικές συναρτήσεις αξιολογούν κατά πόσο οι προβλέψεις του μοντέλου συμβαδίζουν με τις πραγματικές τιμές. Ουσιαστικά οι μετρικές συναρτήσεις έχουν πολλά κοινά με τις συναρτήσεις σφάλματος, με τη διαφορά ότι δεν χρησιμοποιούνται στη διάρκεια της εκπαίδευση.

Προτού γίνει αναφορά στις συνηθέστερες μετρικές, απαιτείται να προσδιοριστούν 4 βασικές έννοιες που χρησιμοποιούνται σε αυτές:

- **True Positive (TP)**: Οι προβλέψεις του μοντέλου ότι ένα στοιχείο ανήκει σε μια κλάση και αυτό είναι ορθό.
- **True Negative (TF)**: Οι προβλέψεις του μοντέλου ότι ένα στοιχείο δεν ανήκει σε μια κλάση και αυτό είναι ορθό.
- **False Positive (FP)**: Οι προβλέψεις του μοντέλου ότι ένα στοιχείο ανήκει σε μια κλάση και αυτό **δεν** είναι ορθό.
- **False Negative (FN)**: Οι προβλέψεις του μοντέλου ότι ένα στοιχείο **δεν** ανήκει σε μια κλάση και αυτό **δεν** είναι ορθό.



Σχήμα 2.21: Εικονική αναπαράσταση των TP-TF-FP-FN. Ο κύκλος που αναπαρίσταται με συνεχή γραμμή δείχνει τις πραγματικές τιμές των δεδομένων. Ο κύκλος που αναπαρίσταται με διακεκομμένη γραμμή δείχνει τις προβλέψεις του μοντέλου.

Με βάση αυτές τις διευκρινήσεις μπορούμε τώρα να παρουσιάσουμε τις πιο γνωστές μετρικές απόδοσης.

Accuracy

Η accuracy χρησιμοποιείται ως στατιστικό μέτρο για το πόσο καλά ένα μοντέλο εντοπίζει σωστά ή αποκλείει μια κατάσταση. Δηλαδή, η ακρίβεια είναι η αναλογία των πραγματικών αποτελεσμάτων (τόσο TP όσο και TN) μεταξύ του συνολικού αριθμού των περιπτώσεων που εξετάστηκαν. Αν και αρκετά συνηθισμένη μετρική, δεν χρησιμοποιείται ευρέως σε προβλήματα ανίχνευσης αντικειμένων καθώς το πλήθος των TN στοιχείων είναι πολύ μεγαλύτερο και κυριαρχεί. Πρακτική εφαρμογή

βρίσκει σε περιπτώσεις που τα σύνολα είναι πιο ισοσταθμισμένα μεταξύ θετικών και αρνητικών προβλέψεων.

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Precision

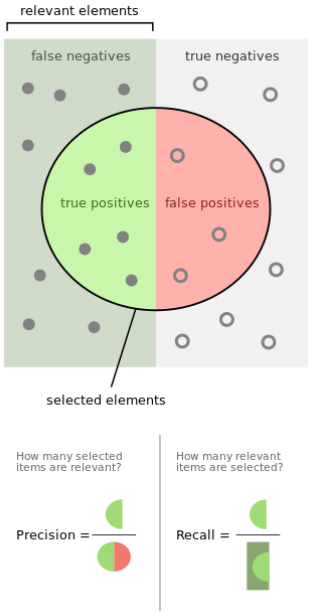
Η ακρίβεια (precision) αντικατοπτρίζει το ποσοστό των επιτυχημένων προβλέψεων μίας κλάσης του μοντέλου ως προς τις συνολικές προβλέψεις. Αποτελεί ένα καλύτερο παράδειγμα μετρικής για την ανίχνευση αντικειμένων καθώς περιορίζεται μόνο στα θετικά δείγματα αλλά από μόνο του δεν αρκεί.

$$precision = \frac{TP}{TP + FP}$$

Recall

Η ανάκληση (recall) αποδίδει το ποσοστό των επιτυχημένων προβλέψεων μίας κλάσης του μοντέλου ως προς τις συνολικές πραγματικές τιμές. Όπως και με την precision, δεν είναι επαρκής αυτόνομα για την αξιολόγηση ενός μοντέλου που κάνει ανίχνευση αντικειμένων.

$$recall = \frac{TP}{TP + FN}$$

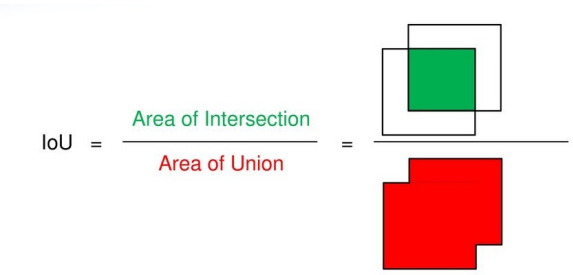


Σχήμα 2.22: Precision και Recall

Τομή προς Ένωση (Intersection over Union)

Η τομή προς ένωση ή Intersection over Union (IoU) αποτελεί μια πιο ευαίσθητη μετρική απόδοσης. Η IoU δείχνει το ποσοστό των ορθώς ανακτημένων στοιχείων (TP) ως προς την ένωση των αληθινών ορίων ενός αντικειμένου με την πρόβλεψη του μοντέλου.

$$IoU = \frac{TP}{TP + FP + FN}$$



Σχήμα 2.23: Intersection over Union σχηματικά

Average Precision

Αφού αναλύσαμε τις επιμέρους μετρικές, μπορούμε να αναφερθούμε στην average precision (AP) η οποία μαζί με τις όποιες παραλλαγές σε εκάστοτε διαγωνισμούς θεωρείται η πλέον κατάλληλη για ανίχνευση αντικειμένων. Ουσιαστικά για κάθε πρόβλεψη του μοντέλου υπολογίζονται precision και recall και ταξινομούνται με αύξουσα ανάκληση. Η AP υπολογίζει την τιμή της precision (p) για κάθε τιμή της ανάκλησης (r) από 0 έως 1 και εκφράζεται ως εξής:

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

Για να θεωρηθεί μια πρόβλεψη σωστή και να συμβάλλει στην AP θα πρέπει να επιτυγχάνει IoU μεγαλύτερη από 0.5. Στην πράξη το ολοκλήρωμα ισοδυναμεί με το έξης άθροισμα της συνάρτησης:

$$AP = \sum_{k=1}^{N} P(k)\Delta r(k)$$

Όπου N ο αριθμός των προβλέψεων και k οι προβλέψεις μέχρι μια δεδομένη ανάκληση. Σε προβλήματα με πολλαπλές κλάσεις υπολογίζεται για κάθε κλάση η average precision και μετά η mean average precision (mAP) ως μέσος όρος των ξεχωριστών AP.

Κεφάλαιο 3

Μοντέλα Ανίχνευσης Αντικειμένων

Αφού αναλύσαμε τα επιμέρους συστατικά των συνελικτικών δικτύων που παραμένουν κοινά ανεξαρτήτως πεδίου δράσης και συνόλου δεδομένων, σε αυτήν την ενότητα θα εξετάσουμε 2 βασικούς αλγορίθμους μηχανικής μάθησης που χρησιμοποιούνται για την ανίχνευση αντικειμένων, με σαφείς διαφορές μεταξύ τους. Το πρώτο δίκτυο είναι το Faster R-CNN [Ren15] και το δεύτερο το YOLO [Redm18].

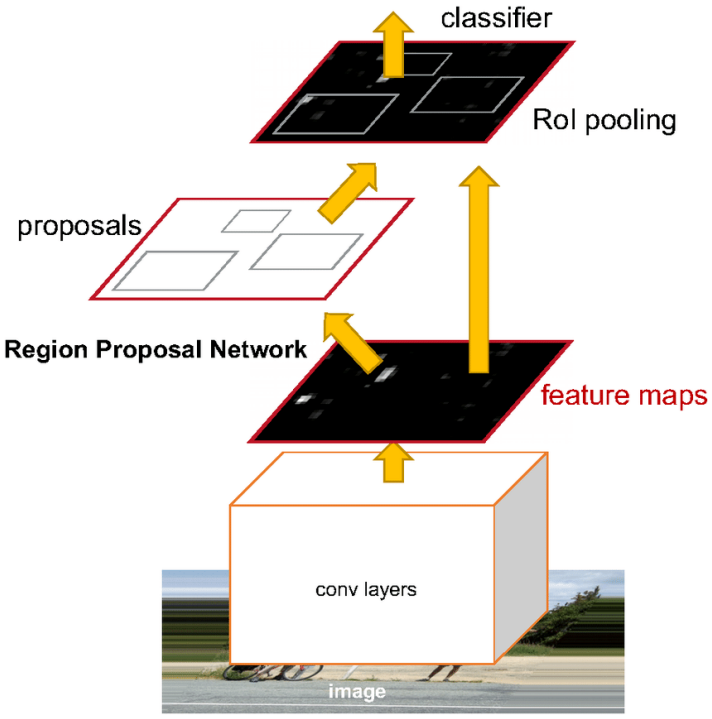
3.1 Faster R-CNN

Το δίκτυο Faster R-CNN δημοσιεύτηκε από μια ομάδα ερευνητών το 2016 και αποτελεί μετεξέλιξη του R-CNN [Girs14] και του Fast R-CNN [Girs15] . Στον τομέα της ανίχνευσης αντικειμένων, σε αντίθεση με την απλή ταξινόμηση, απαιτείται και ο εντοπισμός του εκάστοτε αντικειμένου χωρικά στην εικόνα. Ως εκ τούτου, οι παραδοσιακές αρχιτεκτονικές των συνελικτικών δικτύων που καταλήγουν σε πλήρως συνδεδεμένα επίπεδα με αριθμό ίσο με τις υποψήφιες κλάσεις, δεν μπορούσαν να εφαρμοστούν αυτούσιες αφού ο αριθμός των υποψήφιων περιοχών και αντικειμένων δεν είναι σταθερός και μπορεί να ποικίλλει από εικόνα σε εικόνα.

Στην πρώτη εκδοχή του Faster R-CNN, δηλαδή το R-CNN, η λύση στο πρόβλημα ήταν ο περιορισμός των υποψήφιων περιοχών ενδιαφέροντος (Regions Of Interest – ROI) σε 2000 ανά εικόνα. Η επιλογή των περιοχών αυτών γινόταν μέσω αλγορίθμου επιλεκτικής αναζήτησης (Selective Search) και ήταν εξαιρετικά ακριβή χρονικά, υπολογίζοντας το μεγάλο αριθμό εικόνων που περιλαμβάνει ένα σύνολο δεδομένων. Η προσέγγιση που ακολουθήθηκε στο Fast R-CNN είναι ελαφρώς διαφορετική με το σύστημα αντί να τροφοδοτεί τις υποψήφιες ROI στο δίκτυο, να παρέχει την εικόνα εισόδου με στόχο την κατασκευή ενός χάρτη χαρακτηριστικών (feature map), αξιοποιώντας τις ιδιότητες της συνέλιξης. Χρησιμοποιώντας πάλι τη μέθοδο επιλεκτικής αναζήτησης στο χάρτη χαρακτηριστικών, εξάγονται οι ROI οι οποίες επανασχεδιάζονται σε σταθερό μέγεθος για να τροφοδοτηθούν στα επόμενα επίπεδα της αρχιτεκτονικής. Μέσω αυτής της βελτίωσης ο χρόνος εκπαίδευσης περιορίστηκε αισθητά, όμως ξανά η διαδικασία περιορισμού των περιοχών καταλάμβανε μεγάλο μέρος της.

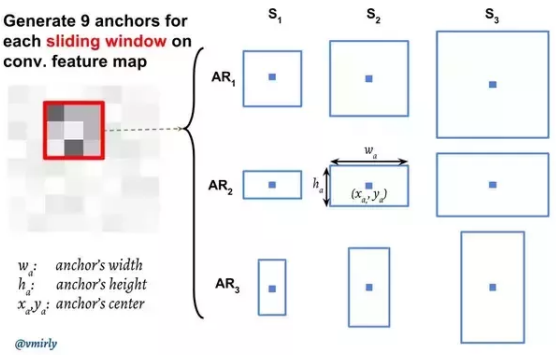
Η δυσκαμψία στην εκπαίδευση του δικτύου οφείλεται σε μεγάλο βαθμό στον αλγόριθμο επιλεκτικής αναζήτησης για την επιλογή των υποψήφιων περιοχών. Η εξάλειψη από αυτό το πρόβλημα γίνεται στον αλγόριθμο Faster R-CNN. Ο ρόλος του αλγορίθμου επιλεκτικής αναζήτησης επιτυγχάνεται με τη χρήση ενός νέου δικτύου, του Region Proposal Network (RPN). Το RPN καθίσταται αρμόδιο για την επιλογή των υποψηφίων περιοχών.

Μετά την εξαγωγή του χάρτη χαρακτηριστικών, το RPN καλείται να εντοπίσει της περιοχές. Σε αντίθεση με την επιλεκτική αναζήτηση που αποτελεί μια ντετερμινιστική διαδικασία που δεν επιδέχεται μάθηση, το RPN είναι ένα υποδίκτυο που και αυτό εκπαιδεύεται με την σειρά του. Βασικό εργαλείο στο έργο αυτό αποτελούν τα λεγόμενα anchor boxes, τα οποία είναι κουτιά διαφορετικών διαστάσεων που εξετάζονται σε κάθε θέση. Το RPN ακολουθεί την τεχνική του κυλιόμενου παραθύρου (sliding window) καθώς σε κάθε περιοχή του χάρτη χαρακτηριστικών εξετάζει όλα τα anchor boxes και τα βαθμολογεί ως υποψήφιες περιοχές. Ουσιαστικά αποτελεί συνδυασμό 2 βασικών τεχνικών για RPN, πρώτον της εξαγωγής διαφορετικών χαρτών χαρακτηριστικών για διαφορετικές διαστάσεις των εικόνων, το οποίο δίνει πιο πλήρεις λύσεις αλλά αργές, και δεύτερο της χρήσης διαφορετικών φίλτρων



Σχήμα 3.1: Η διαδικασία που ακολουθείται στο Faster R-CNN. Ο χάρτης χαρακτηριστικών τροφοδοτείται στο υποδίκτυο RPN και έπειτα γίνεται η τελική αξιολόγηση

που διατρέχουν την εικόνα σαν κυλιόμενο παράθυρο.



Σχήμα 3.2: Η προκαθορισμένη επιλογή είναι η χρήση 9 anchor boxes, 3 σχημάτων σε διαφορετικές διαστάσεις τα οποία εξετάζονται σε κάθε θέση του χάρτη χαρακτηριστικών. Ο αριθμός αυτός μπορεί να αλλάξει ανάλογα με τις ανάγκες και τις απαιτήσεις του κάθε προβλήματος.

Στόχος του RPN είναι να αποφανθεί αν σε ένα υποψήφιο κουτί περιλαμβάνεται αντικείμενο ή όχι. Αντικείμενο θεωρείται ότι υπάρχει όταν η υποψήφια περιοχή εξασφαλίζει ένα όριο IoU σε σχέση με τα αληθινά όρια (IoU > 0.7 θετικά δείγματα, IoU < 0.3 αρνητικά, τα υπόλοιπα δεν συνεισφέρουν). Επειδή αρκετές περιοχές είναι επικαλυπτόμενες, χρησιμοποιείται μη μέγιστη καταστολή (non-maximum suppression - NMS) για να περιοριστούν οι υποψήφιες περιοχές από 6000 σε 300 (default).

Αφού ολοκληρωθεί η διαδικασία επιλογής υποψηφίων περιοχών, το υπόλοιπο δίκτυο παραμένει παρόμοιο με το Fast R-CNN. Δηλαδή, η συνάρτηση σφάλματος μοιάζει με αυτή που χρησιμοποιείται στο Fast R-CNN και οφείλει να λαμβάνει υπόψη τόσο την ύπαρξη αντικειμένου όσο και τις συντεταγμένες των ορίων κάθε κουτιού. Η συνάρτηση σφάλματος συνοψίζεται ως εξής.

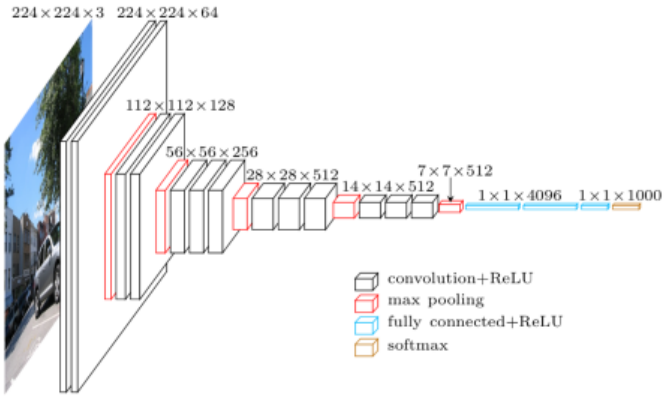
$$L(p_i, t_i) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, {p_i}^*) + \lambda \frac{1}{N_{box}} \sum_i {p_i}^* L_{box}(t_i, {t_i}^*)$$

Με p_i συμβολίζεται η πιθανότητα ύπαρξης αντικειμένου και με t_i οι προβλεπόμενες συντεταγμένες του, ενώ τα πραγματικά δεδομένα με ${p_i}^*$ και ${t_i}^*$. Οι όροι N_{cls}, N_{box} και λ (υπερπαράμετρος) βοηθάνε στην ισορροπία του σφάλματος. Για τις συντεταγμένες χρησιμοποιείται η συνάρτηση σφάλματος smooth L1 ενώ η επιμέρους συνάρτηση σφάλματος για την ταξινόμηση είναι:

$$L_{cls}(p_i, {p_i}^*) = -{p_i}^* log p_i - (1 - {p_i}^*) log(1 - p_i)$$

Η διαδικασία εκπαίδευσης είναι η ακόλουθη. Πρώτα εκπαιδεύεται το δίκτυο RPN και οι προτάσεις τροφοδοτούνται στο κυρίως δίκτυο το οποίο εκπαιδεύεται και αυτό. Στη συνέχεια έχουμε επανάληψη της εκπαίδευσης του RPN δικτύου με βάση το δίκτυο που εκπαιδεύτηκε μόλις, διατηρώντας σταθερά τα συνελικτικά επίπεδα τα οποία πλέον είναι κοινά στα 2 επιμέρους. Τέλος εκπαιδεύεται πάλι το κυρίως δίκτυο χρησιμοποιώντας τα νέα βάρη, διατηρώντας και αυτό με την σειρά του σταθερά τα συνελικτικά επίπεδα.

Αρχικά το Faster R-CNN χρησιμοποίησε ως backbone δίκτυο υποστήριξης το VGG-16 [Simo14] το οποίο έχει 13 συνελικτικά επίπεδα τα οποία είναι κοινά. Πλέον, με την μεγαλύτερη ανάπτυξη των GPUs και δυνατότητα ακόμα πιο βαθιών δικτύων, μπορεί να υλοποιηθεί με χρήση δικτύων ResNet-50 και ResNet-101 [He16] για ακόμα καλύτερα αποτελέσματα.

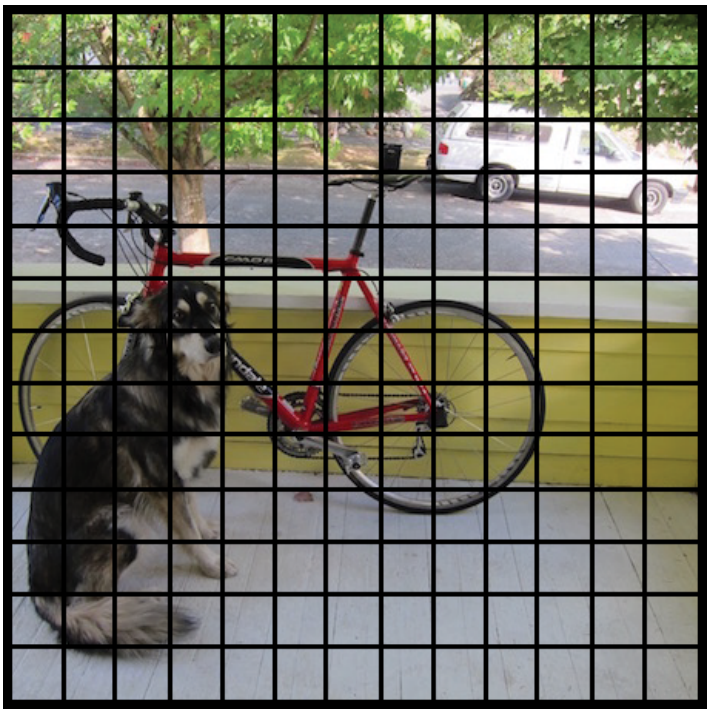


Σχήμα 3.3: Η αρχιτεκτονική του VGG-16.

3.2 You Only Look Once (YOLO)

Συνοπτικά, μελετήσαμε έναν αλγόριθμο ο οποίος λειτουργεί μόνιμα σε 2 στάδια. Πρώτα γίνεται η επιλογή υποψηφίων περιοχών, είτε αρχικά με επιλεκτική αναζήτηση είτε στην συνέχεια μέσω ενός δευτερεύοντος δικτύου RPN, και έπειτα η αξιολόγησή τους από τον κυρίως αλγόριθμο. Αντίθετα ο αλγόριθμος You Only Look Once (YOLO), όπως μαρτυρά και το όνομα του, ακολουθεί τελείως διαφορετική προσέγγιση.

Ο YOLO, πρόκειται για ένα αλγόριθμο μονού σταδίου που κάνει ταυτόχρονα και την επιλογή περιοχών και την αξιολόγησή τους. Για να το επιτύχει αυτό, απορρίπτει την προσέγγιση του κυλιόμενου παραθύρου που ακολουθεί μεταξύ άλλων και ο Faster R-CNN. Σε αντίθεση με αυτόν, χωρίζει την εικόνα σε ένα πλέγμα SxS, όπου το S καθορίζεται ανάλογα με τις απαιτήσεις μας και τις ιδιαιτερότητες του συνόλου δεδομένων, και σε κάθε μέρος του πλέγματος, αν υπάρχει αντικείμενο με βάση τα δεδομένα, προχωρά η ανάλυση για την επιλογή των σωστών περιοχών.



Σχήμα 3.4: Χωρισμός εικόνας που κάνει ο YOLO σε πλέγμα 13x13

Με αυτόν τον τρόπο περιορίζεται δραστικά ο χρόνος εκπαίδευσης αφού περιοχές χωρίς αντικείμενα που συνήθως κυριαρχούν, δεν συμβάλλουν στην επεξεργασία. Τα αντικείμενα όμως δεν βρίσκονται μόνο σε μια περιοχή του πλέγματος. Ο αλγόριθμος θεωρεί πως μια περιοχή είναι αρμόδια για την ανίχνευση ενός αντικειμένου, αν το κέντρο του αντικειμένου με βάση τα δεδομένα, βρίσκεται στην περιοχή αυτή. Για να γίνει καλύτερα κατανοητός ο τρόπος λειτουργίας του αλγορίθμου, παρατίθεται η μορφή της πληροφορίας που κρατείται για σε κάθε περιοχή.

Y	p_c
	b_x
	b_y
	b_h
	b_w
	c_1
	c_2

Θεωρώντας ότι έχουμε 2 κλάσεις, η πληροφορία κωδικοποιείται ως εξής. Το p_c συμβολίζει την πιθανότητα ύπαρξης αντικειμένου, οι επόμενες 4 μεταβλητές τα όρια της προβλεπόμενης περιοχής και οι μεταβλητές c_1 και c_2 παίρνουν τιμή μεταξύ 0 και 1 και συμβολίζουν την πιθανότητα μιας κλάσης να αντιπροσωπεύεται στο αντικείμενο. Η μεταβλητή p_c αποτελεί γινόμενο της πραγματικής δυαδικής τιμής ύπαρξης αντικειμένου και του δείκτη IoU που υπολογίζεται με την προβλεπόμενη περιοχή.

Η αρχική έκδοση του YOLO [Redm16] παρά τα ενθαρρυντικά και προπαντός γρήγορα αποτελέσματα, δυσκολευόταν στην ανίχνευση αντικειμένων που εμφανίζονταν πολλάκις σε κάποιο μπλοκ καθώς ο προηγούμενος μηχανισμός δεν προέβλεπε αυτήν την ύπαρξη, ενώ υπήρχε σχετικά μεγάλος αριθμός σφαλμάτων που προερχόταν από τον σωστό καθορισμό των bounding boxes. Αυτή οι περιορισμοί αίρονται στην επόμενη έκδοση του αλγορίθμου [Redm17] όπου περιέχει μια σειρά από βελτιώσεις. Εκτός από την χρήση batch normalization αντί για dropout για την αποφυγή overfitting αλλά και συνελικτικών επιπέδων στη θέση των χρονοβόρων πλήρων συνδεδεμένων, βασική βελτίωση έφερε η χρήση κάποιας μορφής anchor boxes. Αντί για την κλασική μορφή του διανύσματος που παρουσιάσαμε πριν σε κάθε μέρος του πλέγματος, πλέον σε κάθε επιμέρους κομμάτι εξετάζονται όλα τα διαθέσιμα anchor boxes και αποθηκεύονται βαθμοί εμπιστοσύνης για το καθένα. Με αυτόν

τον τρόπο βελτιώνεται η απόδοση ενώ γίνεται εφικτή και η ανίχνευση παραπάνω αντικείμενων στο πλέγμα αφού κάθε anchor box θα μπορεί να εξειδικεύεται σε διαφορετικό αντικείμενο.

Τα anchor boxes έχουν την δυνατότητα να προσαρμόζονται και αυτά στα εκάστοτε αντικείμενα μέσα από την διαδικασία εκμάθησης. Με τυχαία αρχικοποίηση όμως χάνεται πολύτιμος χρόνος και είναι πιθανό τα αποτελέσματα να μην είναι τα αναμενόμενα μέσα σε ένα περιορισμένο χρονικό διάστημα εκπαίδευσης. Για αυτό το λόγο προτείνεται να τρέξει πρώτα ένας αλγόριθμος συσταδοποίησης (k-means) στο σύνολο δεδομένων ώστε να χρησιμοποιηθούν τα anchor boxes με τον μεγαλύτερο δείκτη IoU. Πειράματα έδειξαν ότι το δίκτυο αποδίδει πολύ καλύτερα με αυτήν την προσέγγιση. Επειδή υπάρχει η πιθανότητα ένα αντικείμενο να φέρει υψηλή πιθανότητα σε διαφορετικά anchor boxes, εφαρμόζεται και εδώ μη μέγιστη καταστολή (non-maximum suppression - NMS) για τον περιορισμό τους.

Επειδή όπως έχουμε τονίσει, η ανίχνευση αντικειμένων δεν είναι ένα απλό πρόβλημα ταξινόμησης, η συνάρτηση σφάλματος, όπως και στον Faster R-CNN, θα πρέπει να αξιολογεί τόσο την σωστή επιλογή κλάσης όσο και τα προβλεπόμενα όρια του αντικειμένου. Πιο συγκεκριμένα, η συνάρτηση σφάλματος στον YOLO έχει 3 επιμέρους συστατικά. Το πρώτο μέρος έχει να κάνει με την πιθανότητα ύπαρξης αντικειμένου μιας κλάσης:

$$L_{prob} = \lambda_{class} \sum_{i}^{S^2} \sum_{j}^{B} \sum_{c}^{C} 1_{ij}(p_{ij}(c) - p^*_{ij}(c))^2$$

Με C συμβολίζουμε των σύνολο των κλάσεων, με S^2 το πλέγμα, με B των αριθμό των anchor boxes και $p_{ij}(c)$ και $p^*_{ij}(c)$ είναι η πιθανότητα να ανήκει στην κλάση και η πραγματική τιμή αντίστοιχα. Ο όρος 1_{ij} ενεργοποιείται και γίνεται 1 όταν υπάρχει αντικείμενο στο κελί του πλέγματος και το box που είναι υπεύθυνο έχει IoU > 0.5 και υψηλότερο IoU, διαφορετικά είναι 0. Το δεύτερο μέρος λαμβάνει υπόψη τις διαστάσεις των περιοχών:

$$L_{box}(r) = \lambda_{coord} \sum_{i}^{S^2} \sum_{j}^{B} 1_{ij}^{respObj} \sum_{r \in (x,y,w,h)} (pred_{ij}^r - truth_{ij}^r)^2$$

$$+\lambda_{coord} \sum_{i}^{S^2} \sum_{j}^{B} 1_{ij}^{noRespObj} \sum_{r \in (x,y,w,h)} (pred_{ij}^r - anchor_{ij}^r)^2$$

Με r συμβολίζονται οι διαστάσεις της περιοχής, με pred οι προβλέψεις, truth τα πραγματικά δεδομένα και anchor οι διαστάσεις του εκάστοτε anchor box. Παρόμοια με πριν, ενεργοποιούνται οι όροι 1_{ij}, ο πρώτος όταν το anchor box είναι αρμόδιο για τον εντοπισμό του αντικειμένου (IoU > 0.5 και υψηλότερο IoU) και ο δεύτερος όταν δεν είναι αρμόδιο. Σε πλήρη αναλογία υπολογίζεται και ο τελευταίος όρος:

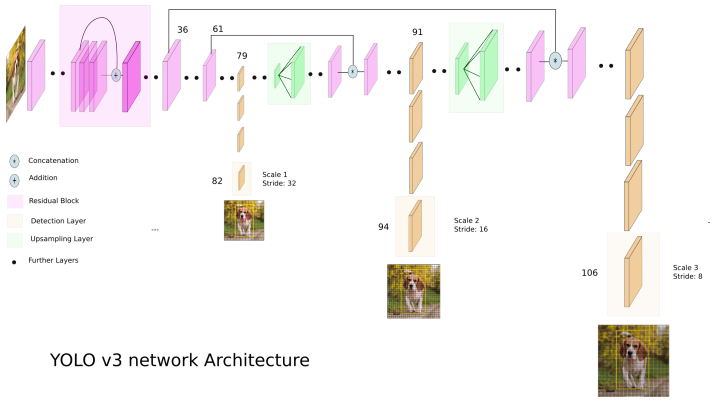
$$L_{conf}(r) = \lambda_{obj}^{conf} \sum_{i}^{S^2} \sum_{j}^{B} 1_{ij}^{respObj} (conf_{ij}^{pred} - IoU(pred_{ij}, truth_{ij}^r)^2$$

$$+\lambda_{noobj}^{conf} \sum_{i}^{S^2} \sum_{j}^{B} 1_{ij}^{noRespObj} (conf_{ij}^{pred} - 0)^2$$

Όλοι οι όροι λ αποτελούν υπερπαραμέτρους και στόχο έχουν την καλύτερη ισορροπία του συστήματος.

Σε αντίθεση με άλλους αλγορίθμους, ως βοηθητικό δίκτυο για την ανάλυση χαρακτηριστικών δεν χρησιμοποιήθηκε κάποιο από τα γνωστά. Στα πλαίσια αυτής της διπλωματικής εργασίας θα αξιοποιήσουμε 3 διαφορετικά δίκτυα που δημιουργήθηκαν για τον YOLO αποκλειστικά. Μια μικρή έκδοση του YOLOv2 με συνελικτικά 9 επίπεδα (tiny-YOLOv2), το full-YOLOv2 με 23 συνελικτικά επίπεδα

και η τελευταία έκδοση (full-YOLOv3) με 76 συνελικτικά επίπεδα και συνολικά 106. Παρά τον μεγάλο αριθμό των επιπέδων, ο αλγόριθμος YOLO πετυχαίνει εφάμιλλα αποτελέσματα και καλύτερα σε σχέση με τον Faster R-CNN σε πολύ μικρότερο χρόνο όμως, καθιστώντας τον αλγόριθμο ικανό για ανίχνευση real-time.



Σχήμα 3.5: Η πλήρης αρχιτεκτονική του YOLOv3

Η εικόνα μαρτυράει ότι η διαδικασία ανίχνευσης ολοκληρώνεται σε 3 φάσεις. Κάθε μία από τις φάσεις χρησιμοποιεί το 1/3 των anchor boxes και διεξάγει ανίχνευση για διαφορετικά μεγέθη αντικειμένων. Πρώτα γίνεται η ανίχνευση αντικειμένων μεγάλων διάστασεων έπειτα από 82 επίπεδα και έχοντας του μικρότερους χάρτες χαρακτηριστικών. Στην συνέχεια με χρήση αντίστροφης δειγματοληψίας, οι χάρτες χαρακτηριστικών μεγαλώνουν και στις επόμενες 2 φάσεις γίνεται η ανίχνευση για αντικείμενα μεσαίων και μικρών διαστάσεων αντίστοιχα. Παρατηρούμε ότι στα πρώτα βήματα, έχουμε προσθέσεις χαρτών για καλύτερη εκμάθηση, χαρακτηριστικό το οποίο στα πλαίσια αυτής της διπλωματικής εργασίας θα επεκτείνουμε, ενώ μετά από κάθε αντίστροφη δειγματοληψία έχουμε συγκόλληση χαρτών από προηγούμενα επίπεδα, προσθέτοντας μεγαλύτερη συνοχή στο δίκτυο.

3.3 Καινοτομίες και Χρήση Νέου Μοντέλου

Στην παράγραφο που ακολουθεί θα αναλυθούν οι καινοτομίες που εφαρμόστηκαν στον υπάρχων αλγόριθμο YOLO και που οδήγησαν υπό ορισμένες συνθήκες σε βελτίωση των αρχικών αποτελεσμάτων.

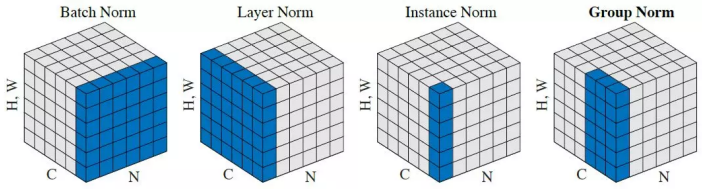
3.3.1 Group-Normalization

Η κανονικοποίηση παρτίδων (Batch Normalization - BN), όπως αναφέραμε και σε προηγούμενο κεφάλαιο, αποτελεί βασικό κομμάτι πλέον στα νευρωνικά δίκτυα και ειδικότερα στην βαθιά μάθηση. Το εκπαιδευόμενο δίκτυο όμως αποδίδει καλύτερα όταν ο αριθμός των εικόνων στις παρτίδες (batch size) είναι σχετικά μεγάλος, συνήθως 16 ή 32. Η διατήρηση τόσο υψηλού αριθμού εικόνων στις παρτίδες αποτελεί τροχοπέδη για προβλήματα που καταναλώνουν πολλή μνήμη. Ειδικά στην ανίχνευση αντικειμένων, που θα ασχοληθούμε, όπως και στην κατάτμηση εικόνας, το μεγάλο υπολογιστικό κόστος περιορίζει το μέγεθος παρτίδας σε μικρούς αριθμούς, μειώνοντας την απόδοση του δικτύου.

Υπό το φως αυτών των ζητημάτων, μόλις το 2018 αναδείχθηκε μια νέα τεχνική κανονικοποίησης που έρχεται να δώσει λύσεις στα προβλήματα που αναφέραμε. Η Κανονικοποίησης Ομάδας (Group Normalization - GN) [Wu18] χρησιμοποιεί διαφορετική προσέγγιση από την BN, μην εκτελώντας κανονικοποίηση στην διάσταση των παρτίδων. Αν και προηγουμένως, είχαν περιγραφεί τεχνικές που ακολουθούν την ίδια τακτική με την GN, όπως η Layer Normalization και η Instance Normalization, καμία από αυτές δεν επιτύγχανε καλύτερα αποτελέσματα στην ανάλυση εικόνας.

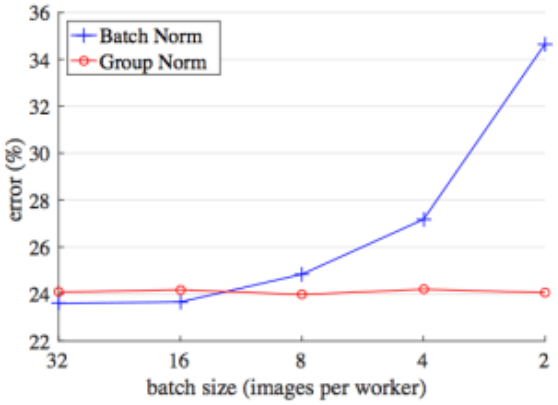
Στην ανάλυση εικόνων 2 διαστάσεων, έπειτα από κάθε συνελικτικό επίπεδο του δικτύου ο όγκος πληροφορίας αποτελείται 4 διαστάσεις NxCxHxW. Οι όροι H και W αντιπροσωπεύουν τους άξονες

των διαστάσεων της εικόνας, ο όρος N τον άξονα των παρτίδων ενώ ο C των καναλιών του επιπέδου. Η BN εκτελεί κανονικοποίηση στον άξονα του N, ενώ η Layer Normalization στον άξονα του C. Αντίθετα, η καινοτομία της GN ενέχει στο γεγονός ότι χωρίζει την διάσταση του C σε ομάδες, συνήθως 32, και εκτελεί κανονικοποίηση σε κάθε ομάδα χωριστά. Με αυτήν την τακτική, η απόδοση του συστήματος αποδεσμεύεται σε μεγάλο βαθμό από τον αριθμό των παρτίδων, διευρύνοντας τους ορίζοντες ακόμα και για υπολογιστικά συστήματα που υστερούν σε hardware.



Σχήμα 3.6: Διαφορετικές προσεγγίσεις των κυρίαρχων τεχνικών κανονικοποίησης

Πειραματικά αποτελέσματα στο πεδίο της ταξινόμησης εικόνων έδειξαν ότι η GN υστερεί ελαφρώς σε περιπτώσεις όπου ο αριθμός των παρτίδων είναι 16 ή 32 ενώ αντίθετα υπερτερεί όταν ο αριθμός μικραίνει. Το δίκτυο στο οποίο δοκιμάστηκε είναι το ResNet-50 και έχει αρκετές ομοιότητες με το δίκτυο στο οποίο πειραματιστήκαμε στα πλαίσια του αλγορίθμου YOLO για ανίχνευση αντικειμένων.

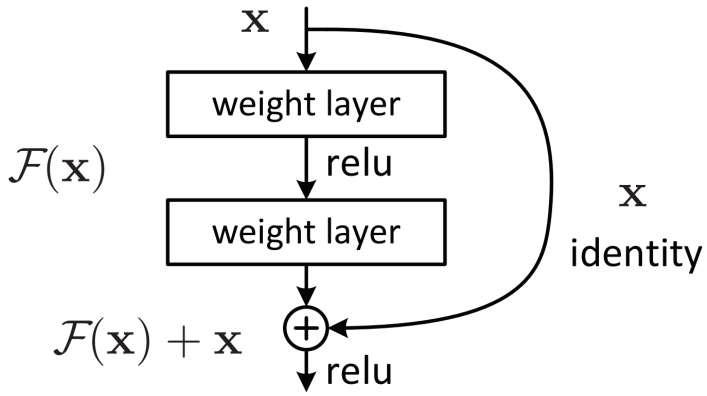


Σχήμα 3.7: Σύγκριση αποτελέσματων για διαφορετικές κανονικοποιήσεις στον τομέα της ταξινόμησης εικόνων. Παρατηρούμε ότι το ποσοστό λάθους με χρήση group normalization μένει σχεδόν ανεπηρεάστο από των αριθμό παρτίδων ενώ με χρήση batch normalization η αύξηση των λαθών είναι αλματώδης

3.3.2 Residual Networks of Residual Networks

Η μεγάλη ανάπτυξη της ισχύς των υπολογιστικών συστημάτων έκανε εφικτή, όπως προείπαμε, την χρήση βαθιών δικτύων που παλιότερα ήταν εξαιρετικά χρονοβόρα. Η χρήση όμως πολύ βαθιών δικτύων έφερε όμως τους ερευνητές αντιμέτωπους με μια νέα πρόκληση. Τα δίκτυα παρατηρήθηκαν ότι είχαν την τάση να «ξεχνάνε» τα χαρακτηριστικά στα οποία θα έπρεπε θεωρητικά να είχαν εκπαιδευτεί στα αρχικά στάδια και ως εκ τούτου η αύξηση των επιπέδων δεν έφερε την προσδοκώμενη αύξηση στην απόδοση. Η απάντηση της επιστημονικής κοινότητας ήταν τα Residual Networks (ResNets) [He16] στα οποία εισήχθη η έννοια της ανατροφοδότησης με την είσοδο.

Πιο συγκεκριμένα, στα ResNets ανά 2 ή ανά 3 συνελικτικά επίπεδα, το αποτέλεσμα ανατροφοδοτείται με αυτό που έλαβε σαν είσοδο το πρώτο από την ομάδα επιπέδων. Με αυτήν την προσέγγιση, τα συστήματα λάβανε μεγάλη ώθηση και αξιοποίησαν στο έπακρο τις δυνατότητες της βαθιάς μάθησης. Ένα τέτοιο δίκτυο είναι και το Darknet-53 που αποτελεί το backbone δίκτυο για το YOLOv3.



Σχήμα 3.8: Εικονική απεικόνιση ενός residual block.

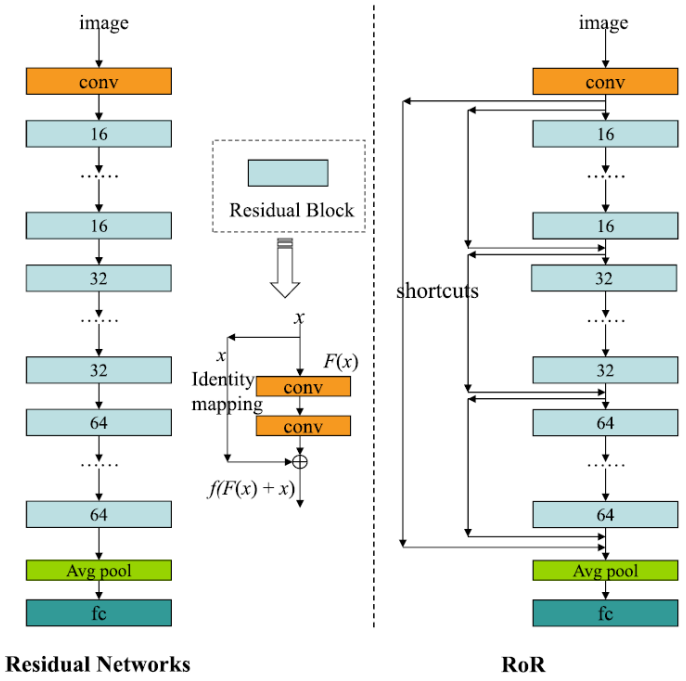
Στο Darknet-53 που αποτελεί τα 74 πρώτα επίπεδα του συνολικού δικτύου των 106 επιπέδων του YOLOv3, κάθε 2 ή 3 συνελικτικά επίπεδα έχουμε ανατροφοδότηση. Με αυτήν την αρχιτεκτονική το δίκτυο περιέχει 21 residual blocks.

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Σχήμα 3.9: Αναλυτική περιγραφή του Darknet-53

Σε συνέχεια αυτή της λογικής προτάθηκε η χρήση των λεγόμενων Residual Networks of Residual Networks (RoR) [Zhan18] που επεκτείνουν την ιδέα των απλών ResNets. Αναλυτικότερα, εκτός από τον χωρισμό του δικτύου σε ομάδες 2 ή 3 επιπέδων που κάνουν τα ResNets, δημιουργούνται και ευρύτερες ομάδες επιπέδων, οι οποίες έχουν τις ίδιες διαστάσεις όγκου πληροφορίας στις οποίες επεκτείνεται η ανατροφοδότηση. Με αυτήν την καινοτομία, οι δυνατότητες των δικτύων επεκτείνονται καθώς ο εκάστοτε όγκος πληροφορίας συνυπολογίζεται από περισσότερα δεδομένα, περιορίζοντας ταυτόχρονα και τις ενδεχόμενες αστοχίες του.

Στα πλαίσια αυτής της διπλωματικής εργασίας αναπτύχθηκε και δοκιμάστηκε ένα πειραματικό δίκτυο, που αποτελεί παραλλαγή τους κλασικού Darknet-53 καθώς αξιοποιεί τις τεχνικές των RoR δικτύων. Παράλληλα, τα residual blocks σε αντίθεση με την κλασική μορφή του YOLOv3, επεκτάθηκαν και πέραν του backbone δικτύου, στο υποδίκτυο που αφορά την ανίχνευση αντικειμένων, προσδίδοντας περαιτέρω ώθηση στα αποτελέσματα.



Σχήμα 3.10: Σύγκριση των αρχιτεκτονικών των απλών ResNets με τα RoR

Κεφάλαιο 4

Πειραματική διαδικασία και αποτελέσματα

4.1 Συλλογή Δεδομένων

Στόχος της διπλωματικής εργασίας είναι η σύγκριση του αλγορίθμου YOLO με άλλους αλγορίθμους στον τομέα της ανίχνευσης αντικειμένων, η παρουσίαση της σταδιακής του εξέλιξης και η βελτιστοποίηση του με χρήση νέων τεχνικών και αρχιτεκτονικών. Για αυτό τον σκοπό χρησιμοποιήθηκαν διαφορετικές συλλογές δεδομένων που να ταιριάζουν στο εκάστοτε κομμάτι της εργασίας.

Στο πεδίο της εξέλιξης του αλγορίθμου και της σύγκρισης με τον Faster R-CNN χρησιμοποιήθηκε μόνο μία μικρή συλλογή δεδομένων με εικόνες πλοίων στην θάλασσα από δορυφόρους. Η συλλογή δεδομένων είναι προσβάσιμη δημόσια μέσω του ιστοτόπου Kaggle[1]. Αποτελείται από 794 εικόνες με μοναδική κλάση "ship" που χωρίζονται σε 694 εικόνες για training και 100 για testing. Μια μικρή μειοψηφία των εικόνων διέφερε σημαντικά σε μέγεθος και ανάλυση από τις υπόλοιπες για αυτό και αποκλείστηκαν από την εκπαίδευση του δικτύου, η οποία έγινε με 607 εικόνες. Η ανάλυση των εκπαιδευόμενων εικόνων κυμαίνεται από 254x208 μέχρι 868x810 , προσδίδοντας μια ευκταία ποικιλία στο σύνολο.

Οι λόγοι επιλογής της συγκεκριμένης συλλογής είναι τόσο θεωρητικοί όσο και πρακτικοί. Η ραγδαία αύξηση της ναυτιλίας έχει οδηγήσει μοιραία σε μεγαλύτερες πιθανότητες εγκλημάτων στην θάλασσα, από περιβαλλοντικά ατυχήματα μέχρι πειρατεία και διακίνηση παράνομου φορτίου. Αυτό το γεγονός έχει καταστήσει αδήριτη ανάγκη την προσεκτική παρακολούθηση και καταγραφή της κίνησης στην ανοιχτή θάλασσα. Όμως και το ιδιαίτερα μικρό μέγεθος αλλά και η απλότητα της, ύπαρξη μίας μόνο κλάσης, έκανε την συλλογή δεδομένων πλέον κατάλληλη για την σύγκριση του YOLO με άλλες εκδόσεις του όσο και με άλλους αλγόριθμους χωρίς να αποτελεί τροχοπέδη η μεγάλη διάρκεια εκπαίδευσης.

Για το τελευταίο κομμάτι της εργασίας που αφορά την βελτιστοποίηση του μοντέλου χρησιμοποιήθηκαν εκτός από την προηγούμενη συλλογή δεδομένων και 2 πολύ μεγαλύτερες συλλογές για την εξαγωγή ασφαλέστερων συμπερασμάτων. Οι συλλογές προορίζονται για την εκπαίδευση συστημάτων αυτοκινούμενων οχημάτων και είναι προσβάσιμες στον ιστότοπο της Udacity[2].

Η πρώτη αποτελείται από 9224 εικόνες εκ των οποίων οι 5119 χρησιμοποιήθηκαν για εκπαίδευση οι 4105 για testing. Οι εικόνες προήρθαν από φωτογραφίες κατά τη διάρκεια οδήγησης και χωρίζονται σε 3 κλάσεις "car", "truck" και "pedestrian" με την πρώτη κλάση να υπερισχύει συντριπτικά στα στιγμιότυπα. Η δεύτερη έχει παρόμοια χαρακτηριστικά με 13071 εικόνες εκ των οποίων οι 7314 αξιοποιήθηκαν για εκπαίδευση και οι 5757 για εξαγωγή αποτελεσμάτων. Μια ακόμα υπαρκτή διαφορά είναι η εισαγωγή μίας ακόμα κλάσης "street lights". Η ανάλυση των εικόνων και στις 2 συλλογές είναι 1920x1200.

4.2 Διαδικασία Εκπαίδευσης και Εργαλεία

Σε πολύ απαιτητικά προβλήματα όπως η ανίχνευση αντικειμένων και η τμηματοποίηση εικόνας, η προσέγγιση που ακολουθείται διαφέρει σε μεγάλο βαθμό στην εκπαίδευση από ότι στην απλή τα-

[1] https://www.kaggle.com/tomluther/ships-in-google-earth
[2] https://github.com/udacity/self-driving-car/tree/master/annotations

ξινόμηση. Στην απλή ταξινόμηση, τα δίκτυα εκπαιδεύονται εκ του μηδενός (from scratch), χωρίς προεκπαιδευμένα βάρη, στο επιθυμητό σύνολο δεδομένων. Στα πρώτα επίπεδα το δίκτυο μαθαίνει χαρακτηριστικά χαμηλού επιπέδου, όπως ακμές και γωνίες, ενώ σε μετέπειτα στάδια πιο εξειδικευμένα χαρακτηριστικά όπως μάτια, ρόδες κ.α. Στο τέλος μέσω ενός πλήρως συνδεδεμένου επιπέδου κατανέμονται οι πιθανότητες της κάθε κατηγορίας.

Αντίθετα σε ζητήματα ανίχνευσης δεδομένων δεν συνίσταται η εκπαίδευση "from scratch" καθώς τόσο η πολυπλοκότητα του προβλήματος όσο και η υπερβολικά μεγάλος χρόνος εκπαίδευσης αποτελούν εμπόδιο. Στην περίπτωση αυτή χρησιμοποιούνται τα βάρη από δίκτυα που έχουν εκπαιδευτεί σε ταξινόμηση χωρίς το τελευταίο επίπεδο που αναφέραμε νωρίτερα. Αυτό τον ρόλο τον διαδραματίζουν τα προαναφερθέντα δίκτυα όπως τα ResNets και το Darknet-53, που έχουν εκπαιδευτεί για μεγάλο αριθμό εποχών στην συλλογή δεδομένων του ImageNet που περιέχει περισσότερες από 14.000.000 εικόνες σε παραπάνω από 20.000 κατηγορίες και όπως γίνεται αντιληπτό μια τέτοια εκπαίδευση δεν μπορεί να γίνει σε συμβατικές GPUs. Τα κομμάτια αυτά του τελικού δικτύου ονομάζονται backbone και ακολουθούνται από το υπόλοιπο δίκτυο που εξειδικεύεται στην ανίχνευση.

Αυτή η προσέγγιση ακολουθήθηκε στα πρώτα στάδια των πειραμάτων που εκτελέστηκαν και αποσκοπούσαν στην σύγκριση των υπαρχόντων αλγορίθμων, ενώ σε κάθε πείραμα θα επισημαίνεται ποιο κομμάτι του αποτελεί το backbone.

Στο πλαίσιο αυτής της εργασίας έχει χρησιμοποιηθεί το Keras, μια σύγχρονη διεπαφή προγραμματισμού εφαρμογών που είναι συμβατό με πολλές βιβλιοθήκες που αξιοποιούνται στην μηχανική μάθηση όπως το Tensorflow. Στις βιβλιοθήκες του Keras, εκτός των άλλων, υπάρχουν πολλά δομικά συστατικά των νευρωνικών δικτύων που έχουμε παρουσιάσει σε προηγούμενα κεφάλαια, όπως επίπεδα, συναρτήσεις σφάλματος, συναρτήσεις ενεργοποιήσεων, βελτιστοποιητές. Ταυτόχρονα ανανεώνεται συνεχώς και είναι διαθέσιμα σε πιλοτικές βιβλιοθήκες καινοτόμα στοιχεία λίγο μετά την παρουσίασή τους. Ο κώδικας γράφτηκε σε Python ενώ τα διαγράμματα προέρχονται από το Tensorboard μια βιβλιοθήκη του Tensorflow για εύκολο έλεγχο της πορείας εκπαίδευσης.

4.3 Πειραματικές Διαδικασίες

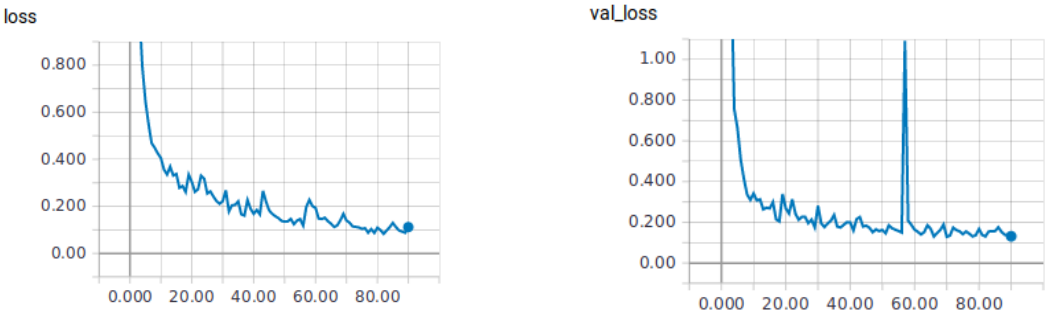
4.3.1 Εξέλιξη του YOLO

Στο πρώτο πειραματικό κομμάτι της εργασίας γίνεται σύγκριση της απόδοσης διαφορετικών εκδόσεων του YOLO (βασίστηκαν σε [Anh18]). Η πρώτη έκδοση κρίθηκε πολύ πρωτόλεια και δεν αξιοποιήθηκε καν για σύγκριση. Αντίθετα χρησιμοποιήθηκε μια μικρή έκδοση του YOLOv2 (tiny YOLOv2), η ολοκληρωμένη έκδοση του YOLOv2 (full YOLOv2) και η νεότερη YOLOv3.

Η συνάρτηση βελτιστοποίησης που χρησιμοποιήθηκε σε αυτά τα πειράματα αλλά και σε όλα τα παρακάτω που αφορούν τον αλγόριθμο του YOLO είναι ο Adam που έχουμε αναλύσει σε προηγούμενο κεφάλαιο καθώς στα πειράματά μας υπερτερούσε έναντι των άλλων συναρτήσεων που έχουμε αναφέρει αλλά και του Padam [Chen18] που αποτελεί μια τροποποιημένη μορφή του. Οι παράμετροι που χρησιμοποιήθηκαν είναι οι εξής: $lr = learningrate, beta_1 = 0.9, beta_2 = 0.999, epsilon = 1e-08, decay = 0.0$. Για συνάρτηση ενεργοποίησης επιλέχθηκε η εξελιγμένη LeakyReLU του Keras και γίνεται αξιοποίηση BatchNormalization. Τα πειράματα διεξήχθησαν στις GPUs του εργαστηρίου, μοντέλου GeForce GTX 1080 8GB ενώ το batch size ορίστηκε στα 4 γιατί οι υψηλές απαιτήσεις καθιστούσαν μεγαλύτερο αριθμό αδύνατο λόγω ανεπάρκειας μνήμης.

Με την χρήση του tiny YOLOv2 ήρθαμε σε επαφή με τα πρώτα αποτελέσματα της εκπαίδευσης του αλγορίθμου. Το πολύ μικρό μέγεθος του backbone δικτύου, μόλις 8 συνελικτικά επίπεδα + 1 ακόμα για την ανίχνευση, καθιστούν το δίκτυο πολύ γρήγορο και ευέλικτο αλλά αδυνατούν να φέρουν τα επιθυμητά αποτελέσματα σε μια απλή συλλογή δεδομένων με μία κλάση και περιορίζονται στο 70,15% AP αλλά μόλις σε 16 λεπτά εκπαίδευσης.

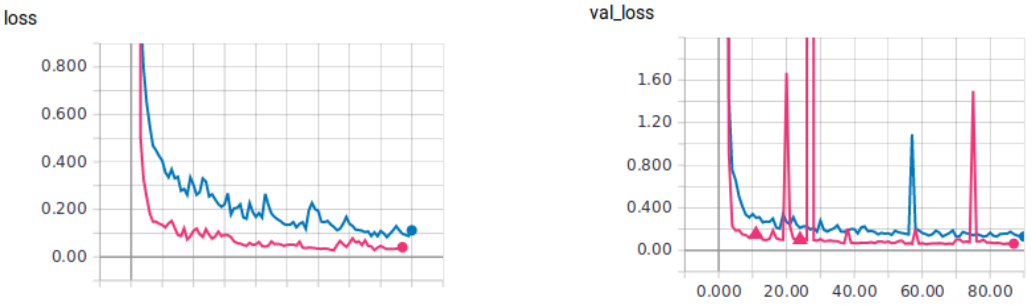
Έπειτα η εκπαίδευση έγινε για την ολοκληρωμένη έκδοση του YOLOv2 που αποτελείται από 22 συνελικτικά επίπεδα για εξαγωγή χαρακτηριστικών συν 1 ακόμα για ανίχνευση ξανά. Τα αποτελέσμα έδειξαν την αναμενόμενη άνοδο στην ακρίβεια η οποία έφτασε στο 86,51% AP σε 42 λεπτά.



Σχήμα 4.1: Τα διαγράμματα των συναρτήσεων απώλειας για το training και το validation set για 90 εποχές εκπαίδευσης του tiny YOLOv2

Σχήμα 4.2: Τα διαγράμματα των συναρτήσεων απώλειας για το training και το validation set για 90 εποχές εκπαίδευσης του full YOLOv2

Παρατηρούμε ότι παρά την ύπαρξη περισσότερων ανωμαλιών στην συνάρτηση απώλειας στο validation set για το μεγαλύτερο δικτύο, η εκπαίδευση καταφέρνει να φτάσει σε καλύτερα επίπεδα κάτι που φαίνεται πιο παραστατικά στο σχήμα 4.3 που αποτελεί συνδυασμό των 2 προηγουμένων.

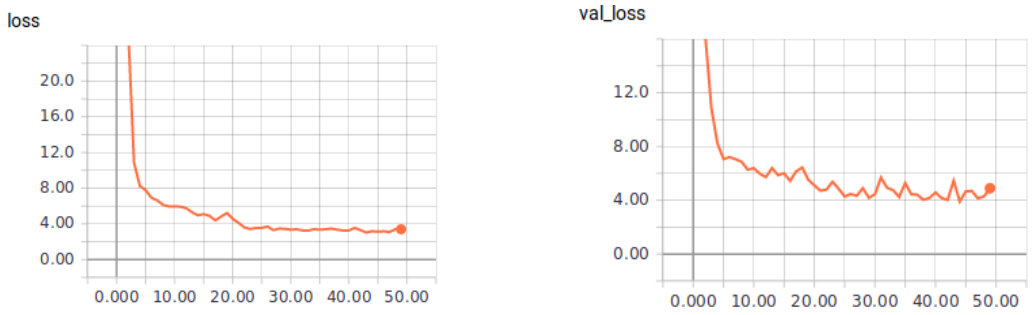


Σχήμα 4.3: Συνδυαστικά αποτελέσματα γραφημάτων για τις 2 εκδόσεις του YOLOv2

Τέλος για το πρώτο μέρος των πειραματικών αποτελεσμάτων, γίνεται η σύγκριση με την νεότερη έκδοση του YOLO (YOLOv3) η οποία όπως έχουμε προαναφέρει εκτός από τις έτερες καινοτομίες έχει πολύ μεγαλύτερο δίκτυο, αφού αποτελείται 53 συνελικτικά επίπεδα στο backbone δίκτυο και συνολικά 76, αποτελώντας ένα χαρακτηριστικό παράδειγμα βαθιάς μάθησης. Η γρήγορη εκπαίδευση περιόρισε τον αριθμό των απαιτούμενων εποχών στις 50 για αυτήν την απλή συλλογή και ύστερα από 47 λεπτά η ακρίβεια φτάνει 94,75% AP.

4.3.2 Σύγκριση με τον αλγόριθμο Faster R-CNN

Σε αυτήν την παράγραφο θα γίνει μια σύντομη σύγκριση με τον αλγόριθμο Faster R-CNN, αξιοποιήθηκε η υλοποίηση [Chen17], που όπως έχουμε αναλύσει κάνει την ανίχνευση αντικειμένων σε

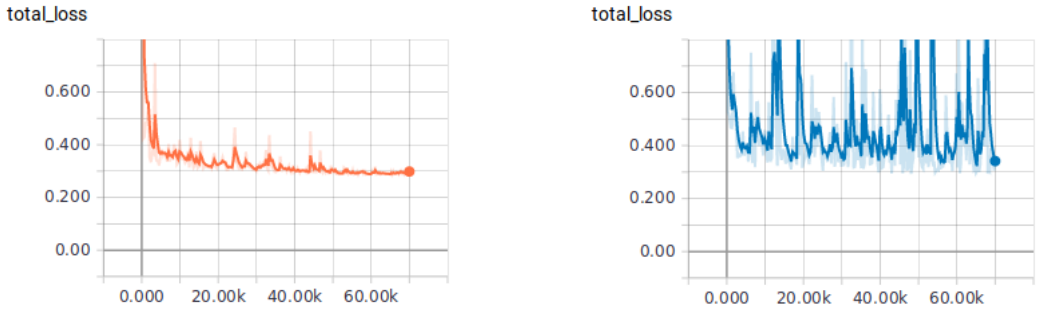


Σχήμα 4.4: Η χρήση της νεότερης έκδοσης βελτίωσε την ταχύτητα σύγκλισης και μείωσε τις πιθανές ανωμαλίες της απώλειας

2 στάδια και θεωρείται από τους πιο σύγχρονους και βάση σύγκρισης για κάθε νέο αλγόριθμο στο πεδίο.

Στα παρακάτω πειράματα χρησιμοποιήθηκαν οι προεπιλεγμένες τιμές παραμέτρων που χρησιμοποιήθηκαν και στο paper με εξαίρεση το backbone δίκτυο. Αρχικά το σύστημα αξιοποιούσε το VGG-16 για την εξαγωγή χαρακτηριστικών όμως η εξέλιξη των δικτύων και των διαθέσιμων πόρων οδήγησαν όπως αναλύσαμε σε χρήση βαθιών δικτύων. Έτσι για λόγους πληρότητας και πιο αμερόληπτης σύγκρισης ο αλγόριθμος εκτελέστηκε τόσο με backbone το ResNet-50 όσο και το ResNet-101.

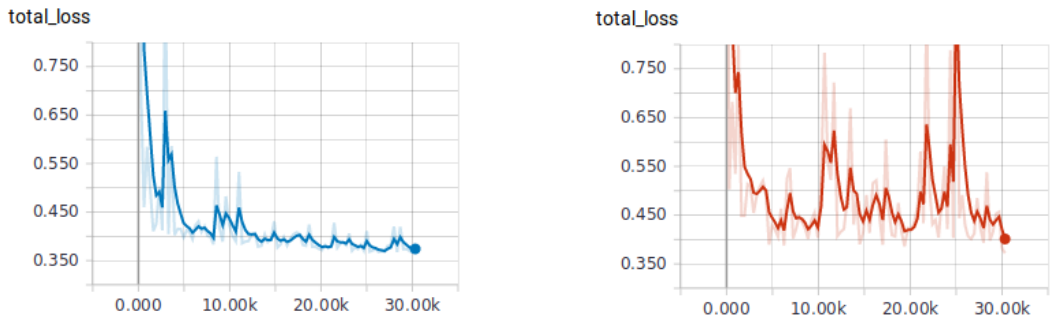
Για το ResNet-50 το δίκτυο εκπαιδεύτηκε για υπερδιπλάσιες εποχές (115 εποχές) απο το YOLOv3. Η εκπαίδευση διήρκησε 11 ώρες και 25 λεπτά και απέφερε ακρίβεια της τάξης του 88,86%.



(a) train loss **(b)** val loss

Σχήμα 4.5: Η πορεία εκπαίδευσης του αλγορίθμου Faster R-CNN με το δίκτυο ResNet-50. Ο οριζόντιος άξονας μετριέται σε iterations αντί για εποχές

Για το ResNet-101 το δίκτυο εκπαιδεύτηκε για 50 εποχές, όσες δηλαδή και το YOLOv3. Η εκπαίδευση διήρκησε 7 ώρες και 30 λεπτά και η ακρίβεια του ήταν 90,7%.



(a) train loss **(b)** val loss

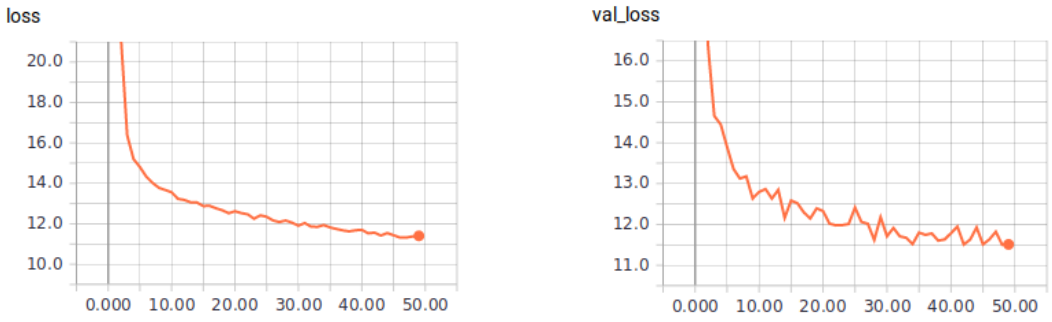
Σχήμα 4.6: Η πορεία εκπαίδευσης του αλγορίθμου Faster R-CNN με το δίκτυο ResNet-101.

Τα αποτελέσματα έδειξαν ότι ο αλγόριθμος Faster R-CNN υπερκερνά και τις 2 εκδόσεις του YOLOv2 αλλά υστερεί έναντι του εξελιγμένου YOLOv3. Η μεγαλύτερη διαφορά, όμως, είναι η διάρκεια εκπαίδευσης των 2 αλγορίθμων καθώς ο Faster R-CNN απαιτεί πολύ περισσότερο χρόνο για να επιτύχει αυτά τα αποτελέσματα. Αυτό εξηγείται από το γεγονός που αναπτύξαμε σε προηγούμενο κεφάλαιο ότι ο Faster R-CNN είναι αλγόριθμος που εκτελεί την ανίχνευση σε 2 στάδια. Σε πιο σύνθετες συλλογές δεδομένων που περιέχουν πιο πολλές κλάσεις μπορεί να επιτύχει καλύτερα αποτελέσματα από τον YOLO αλλά ο χρόνος εκπαίδευσης συνεχίζει να αποτελεί αναγκαίο trade-off. Τέλος, πέρα από την ταχύτητα εκπαίδευσης, η μεγάλη ταχύτητα με την οποία κάνει την ανίχνευση ο YOLO τον καθιστά real-time detector κάτι που επίσης υστερεί ο Faster R-CNN αλλά στα πλαίσια αυτής της διπλωματικής εργασίας δεν θα αναλυθεί αυτό το πεδίο.

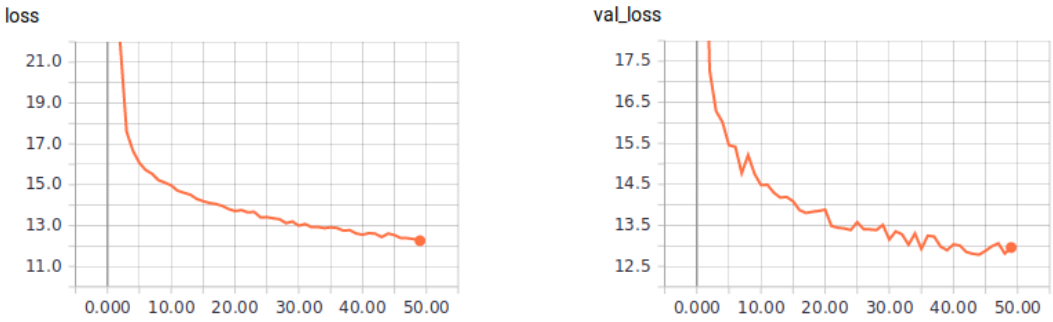
4.3.3 Αξιοποίηση μεγάλων συλλογών δεδομένων και εφαρμογή καινοτομιών

4.3.3.1 Απλό Μοντέλο

Σε αυτήν την παράγραφο τα πειράματά μας επεκτείνονται και στις άλλες 2 συλλογές δεδομένων. Στην πρώτη συλλογή δεδομένων όπου υπάρχουν 3 κλάσεις η εκπαίδευση για 50 εποχές διήρκησε 9 ώρες και 36 λεπτά καταφέρνοντας να επιτύχει ακρίβεια 65,92% mAP. Ομοίως για τη δεύτερη συλλογή, με εκπαίδευση 12 ωρών και 11 λεπτών η ακρίβεια φτάνει στο 60,5% mAP. Και στις περιπτώση η απόδοση θα ήταν πολύ μεγαλύτερη αν στην συλλογή δεδομένων οι κλάσεις είχαν περισσότερη ισορροπία.



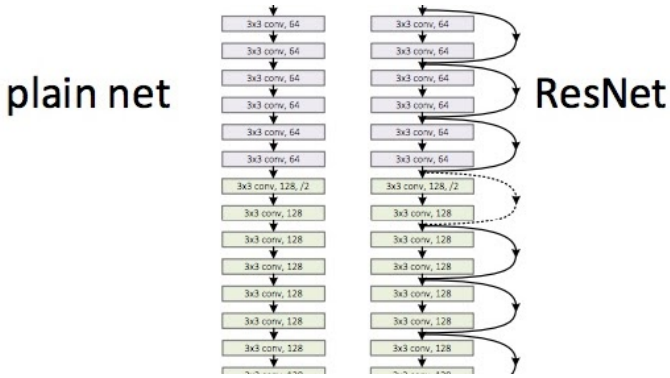
Σχήμα 4.7: Η εκπαίδευσης του αλγορίθμου YOLO στο νέο dataset με τις 3 κλάσεις.



Σχήμα 4.8: Η εκπαίδευσης του αλγορίθμου YOLO στο νέο dataset με τις 4 κλάσεις.

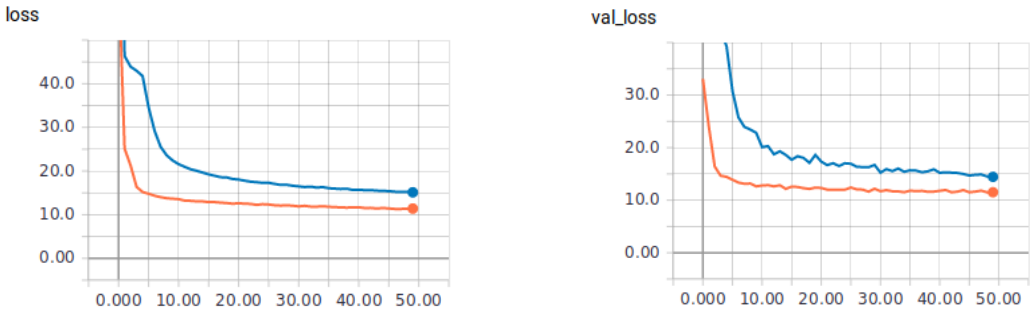
4.3.3.2 Μοντέλο χωρίς την χρήση residual blocks

Έπειτα πριν προχωρήσουμε στις επεκτάσεις που έχουμε αναφέρει, αποφασίσαμε να εξακριβώσουμε την συμβολή των residual blocks στα σύγχρονα νευρωνικά δίκτυα. Στην παράγραφο αυτήν τα πειράματα γίνονται με το ίδιο ακριβώς αριθμό συνελικτικών επιπέδων με τη διαφορά ότι το δίκτυο είναι plain.

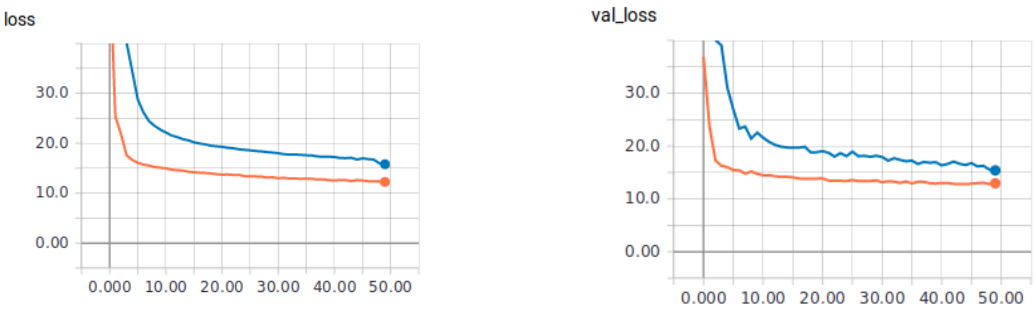


Σχήμα 4.9: Διαφορά στην αρχιτεκτονική plain nets και residual nets

Διαπιστώσαμε σημαντική πτώση στην απόδοση του δικτύου και στις 2 νέες συλλογές δεδομένων. Στην πρώτη περίπτωση, η ακρίβεια έπεσε από το 65,92% mAP στο 45,32% ενώ στη δεύτερη, από το 60,5% στο 43,79% mAP κάτι που φαίνεται παραστατικά στα επόμενα 2 συνδυαστικά διαγράμματα.

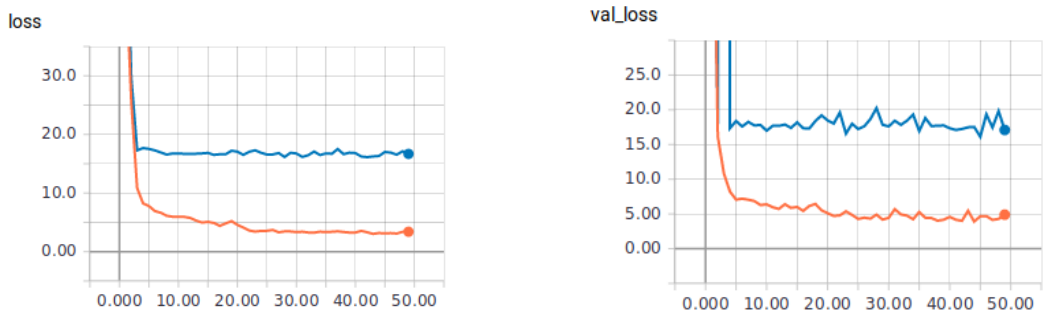


Σχήμα 4.10: Σύγκριση της εκπαίδευσης με χρήση plain δικτύου (μπλε γραμμή) και residual δικτύου (πορτοκαλί γραμμή) για την συλλογή με τις 3 κλάσσεις.



Σχήμα 4.11: Σύγκριση της εκπαίδευσης με χρήση plain δικτύου (μπλε γραμμή) και residual δικτύου (πορτοκαλί γραμμή) για την συλλογή με τις 4 κλάσσεις.

Η σημασία των residual blocks γίνεται ακόμα περισσότερο αντιληπτή στην πρώτη μικρή συλλογή που χρησιμοποιήσαμε στις προηγούμενες παραγράφους. Υπενθυμίζουμε ότι στην απλή αυτή συλλογή, η ακρίβεια έφτανε στα επίπεδα του 94,75% AP. Χωρίς όμως την χρήση residual blocks, το νευρωνικό δίκτυο αδυνατεί να εκπαιδευτεί επαρκώς με αποτέλεσμα να μην μπορεί να εντοπίσει κανένα αντικείμενο και να έχει 0% AP.



Σχήμα 4.12: Σύγκριση της εκπαίδευσης με χρήση plain δικτύου (μπλε γραμμή) και residual δικτύου (πορτοκαλί γραμμή) για την συλλογή με τα πλοία.

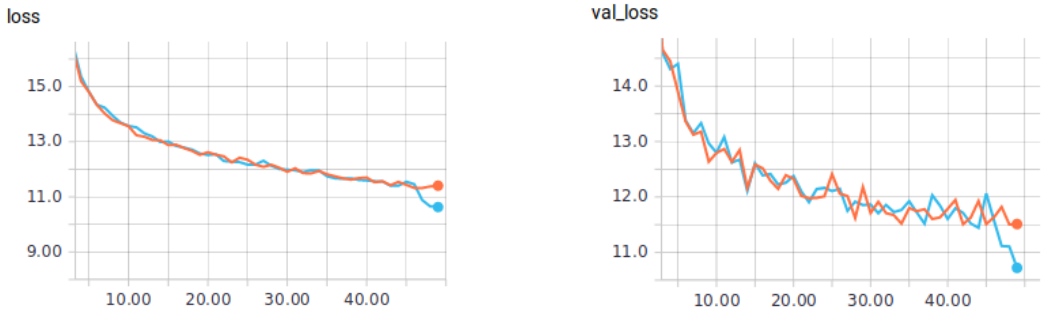
Η πειραματική επιβεβαίωση της σημασίας των residual blocks για την εκπαίδευση βαθιών δικτύων μας έδωσε ώθηση για την περαιτέρω επέκταση που σκοπεύαμε να κάνουμε, ενώ η μηδαμινή μεταβολή στη διάρκεια εκπαίδευσης λόγω αυτού του στοιχείου αποτέλεσε έναν ακόμα σημαντικό παράγοντα που τεκμηρίωνε την επιλογή μας.

4.3.3.3 Μοντέλο με περαιτέρω residual blocks και περισσότερες ανατροφοδοτήσεις

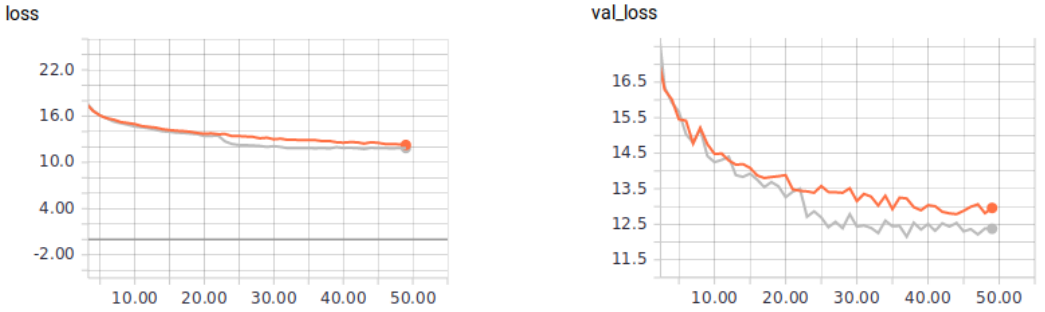
Το επόμενο βήμα ήταν η επέκταση των residual blocks και πέρα από το backbone δίκτυο του YOLO ενώ χρησιμοποιήθηκαν και ευρύτερες ανατροφοδοτήσεις με σκοπό να δώσουν μεγαλύτερη συνοχή στο δίκτυο. Όμως τα αποτελέσματα και στα 2 μεγάλα datasets ήταν χειρότερα ή στα ίδια επίπεδα σε σχέση με την κλασσική μορφή του αλγορίθμου ρίχνοντας την απόδοση απο 65,92% mAP σε 65,2% και στο δεύτερο από 60,5% σε 59,09% αν και δεν αντικατοπτρίζεται στην συνάρτηση απώλειας που μοιάζει να βελτιώνεται ελαφρώς. Μόνο στην συλλογή δεδομένων με τα πλοία υπήρξε μικρή βελτίωση από 94,75% σε 95,91% αλλά η απλότητά της όσο και τα αντικρουόμενα αποτελέσμα των προηγουμένων δεν μπορούν να οδηγήσουν σε ασφαλή συμπεράσματα.

Σχήμα 4.13: Σύγκριση της εκπαίδευσης του εξελιγμένου δικτύου (μπλε γραμμή) και του κλασσικού δικτύου (πορτοκαλί γραμμή) για την συλλογή με τα πλοία.

Παρά την πρόβλεψη μας για καλύτερα αποτελέσματα αυτά δεν επιτέφθηκαν ενώ και στα διαγραμμάτα διαφαίνεται μια ισορροπία μεταξύ των δικτύων. Όμως υπάρχει μια ειδοποιός διαφορά που παίζει καταλυτικό ρόλο. Οι αλλαγές που επιχειρήσαμε αλλάζουν δομικά συστατικά της αρχιτεκτονικής του δικτύου. Για την εκπαίδευση όπως έχουμε αναλύσει χρησιμοποιούνται τα βάρη από το backbone που έχουν εκπαιδευτεί προηγουμένως για μεγάλο χρονικό διάστημα για ταξινόμηση. Έτσι οποιαδήποτε



Σχήμα 4.14: Σύγκριση της εκπαίδευσης του εξελιγμένου δικτύου (μπλε γραμμή) και του κλασσικού δικτύου (πορτοκαλί γραμμή) για την συλλογή με τις 3 κλάσσεις.

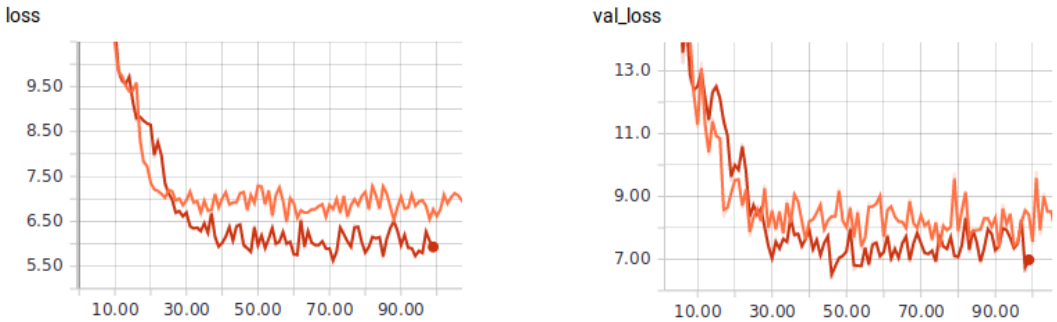


Σχήμα 4.15: Σύγκριση της εκπαίδευσης του εξελιγμένου δικτύου (γκρι γραμμή) και του κλασσικού δικτύου (πορτοκαλί γραμμή) για την συλλογή με τις 4 κλάσσεις.

αλλαγή δοκιμαστεί έχει το μειονέκτημα ότι τα βάρη που χρησιμοποιούνται για αρχικοποίηση έχουν βάση διαφορετικό μοντέλο, καθιστώντας τα μη βέλτιστα και δυσχεραίνοντας την σύγκριση με την κλασσική μορφή του δικτύου.

Για αυτό το λόγο, αποφασίσαμε να κάνουμε την εκπαίδευση του δικτύου χωρίς τα προεκπαιδευμένα βάρη (from scratch), ώστε να συγκρίνουμε τα δίκτυα επί ίσοις όροις. Τα αποτελέσματα σαφώς περιμέναμε να είναι χειρότερα σημαντικά αλλά θα ήταν εφικτή μια σύγκριση.

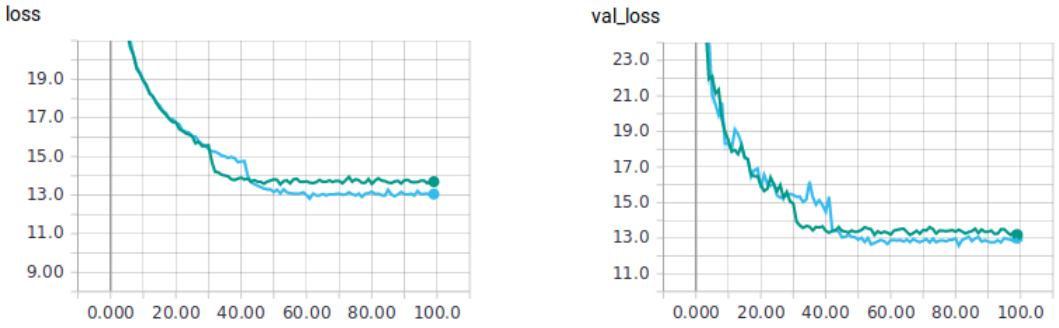
Επειδή οι προϋποθέσεις είναι πιο δυσμενείς, σε κάθε πείραμα αυξήσαμε τον αριθμό των εποχών εκπαίδευσης από 50 στις 100. Στο πρώτο πείραμα με τη συλλογή των πλοίων, το κλασσικό μοντέλο επιτυγχάνει ακρίβεια 80,42% AP ενώ το εξελιγμένο 85,51%. Η εκπαίδευση διήρκησε 1 ώρα και 10 λεπτά περίπου και στις 2 περιπτώσεις.



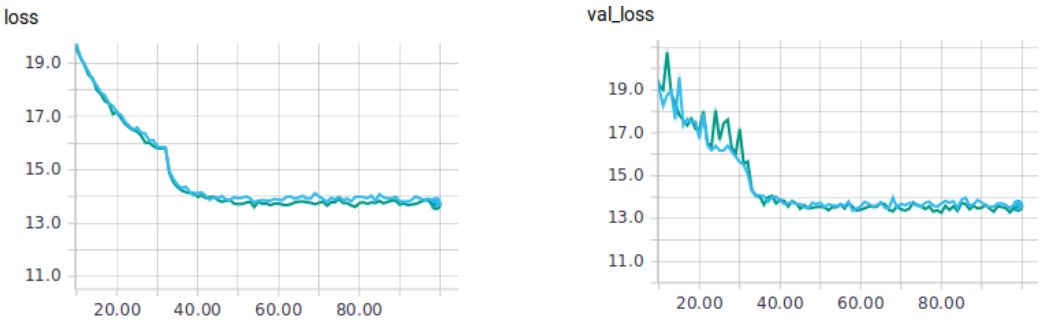
Σχήμα 4.16: Σύγκριση της εκπαίδευσης του εξελιγμένου δικτύου (κόκκινη γραμμή) και του κλασσικού δικτύου (πορτοκαλί γραμμή) για τη συλλογή με τα πλοία χωρίς τη χρήση προεκπαιδευμένων βαρών.

Πέρα από την πρώτη συλλογή δεδομένων, και για τις άλλες 2 μεγάλες τα αποτελέσματα ήταν θετικά. Για τη συλλογή δεδομένων με τις 3 κλάσεις, η κλασική μορφή του YOLO επιτυγχάνει ακρί-

βεια 48,14% mAP ενώ μετέπειτα αυξάνεται 50,69%. Παρόμοια, αν και με μικρότερη βελτίωση είναι τα αποτελέσματα και για την άλλη συλλογή με 51,45% αρχικά και 52,04% ύστερα. Η εκπαίδευση κράτησε 13 ώρες και 30 λεπτά περίπου και 18 ώρες και 20 λεπτά αντίστοιχα.



Σχήμα 4.17: Σύγκριση της εκπαίδευσης του εξελιγμένου δικτύου (γαλάζια γραμμή) και του κλασσικού δικτύου (πράσινη γραμμή) για την συλλογή με τις 3 κλάσεις χωρίς την χρήση προεκπαιδευμένων βαρών.



Σχήμα 4.18: Σύγκριση της εκπαίδευσης του εξελιγμένου δικτύου (πράσινη γραμμή) και του κλασσικού δικτύου (γαλάζια γραμμή) για την συλλογή με τις 4 κλάσεις χωρίς την χρήση προεκπαιδευμένων βαρών.

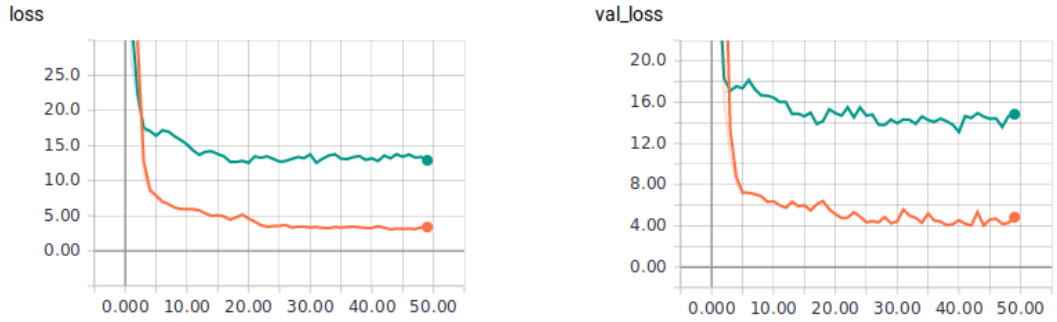
4.3.3.4 Μοντέλο με χρήση group normalization

Απολύτως ανάλογη διαδικασία ακολουθήθηκε και για τη δεύτερη καινοτομία που εφαρμόσαμε. Απο τη στιγμή που οι απαιτήσεις μνήμης μας οδηγούν στο να έχουμε αριθμό batch size 4, περιμέναμε ότι η χρήση group normalization αντί για batch normalization θα είχε αποτέλεσμα καλύτερη απόδοση. Όμως ξανά τα αποτελέσματα αρχικά δεν ήταν τα αναμενόμενα.

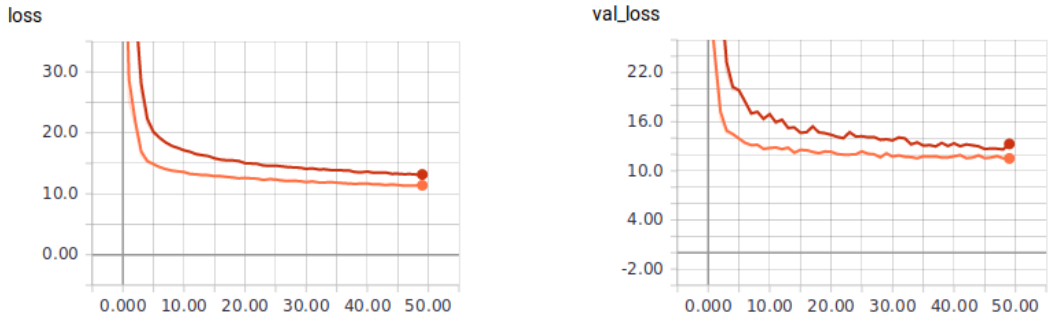
Πιο συγκεκριμένα, στα πλοία η ακρίβεια καταποντίστηκε από 94,75% AP σε 27,26%. Επίσης και στις άλλες 2 συλλογές δεδομένων είχαμε χειρότερη απόδοση με την μετρική mAP να υποχωρεί από 65,92% σε 63,09% και από 60,5% σε 50,25%.

Για ακόμα μια φορά παρατηρούμε πόσο σημαντικό ρόλο διαδραματίζουν τα βάρη που χρησιμοποιούνται για αρχικοποίηση στο backbone δίκτυο. Παρά τις αλλαγές που επιτελέσαμε η απόδοση χειροτέρεψε, είτε λίγο, είτε πολύ, σε όλες τις περιπτώσεις. Όπως θα διαπιστώσουμε, σε αυτό το κομμάτι των πειραμάτων, αυτό συμβαίνει γιατί τα βάρη όπως και πριν δεν είναι βέλτιστα για το εξελιγμένο δίκτυο που χρησιμοποιούμε.

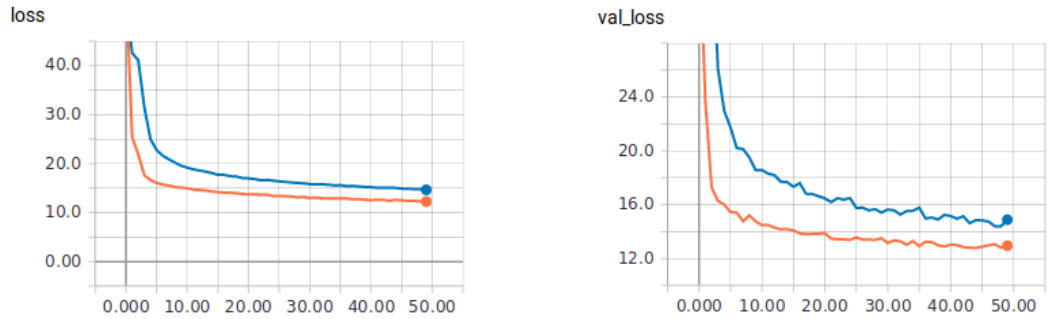
Αναλυτικότερα, με την εκπαίδευση του δικτύου εκ του μηδενός παρατηρήσαμε βελτίωση στην απόδοση σε 2 από τις 3 συλλογές δεδομένων. Μόνο στη μικρή συλλογή με τα πλοία η απόδοση παραμένει χειρότερη σε μεγάλο βαθμό καθώς η ακρίβεια από 80,42% πέφτει σε 53,52%. Αντίθετα στις άλλες 2 περιπτώσεις έχουμε μια μικρή βελτίωση στην μετρική mAP από 48,14% σε 49,76% και από 51,45% σε 52,39%. Το αρνητικό είναι ότι η εκπαίδευση με group normalization επιφέρει μια αύξηση περίπου 50% στο χρόνο εκπαίδευσης.



Σχήμα 4.19: Σύγκριση της εκπαίδευσης δικτύου με χρήση group normalization (πράσινη γραμμή) και δικτύου με χρήση batch normalization (πορτοκαλί γραμμή) για τη συλλογή με τα πλοία.

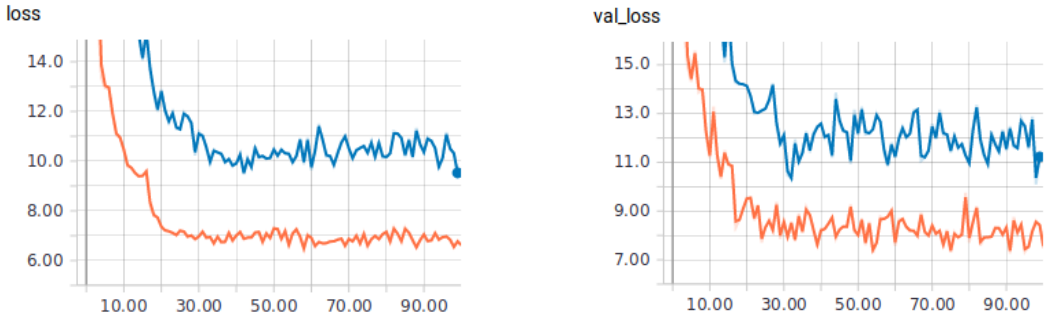


Σχήμα 4.20: Σύγκριση της εκπαίδευσης δικτύου με χρήση group normalization (κόκκινη γραμμή) και δικτύου με χρήση batch normalization (πορτοκαλί γραμμή) για την συλλογή με τις 3 κλασεις.

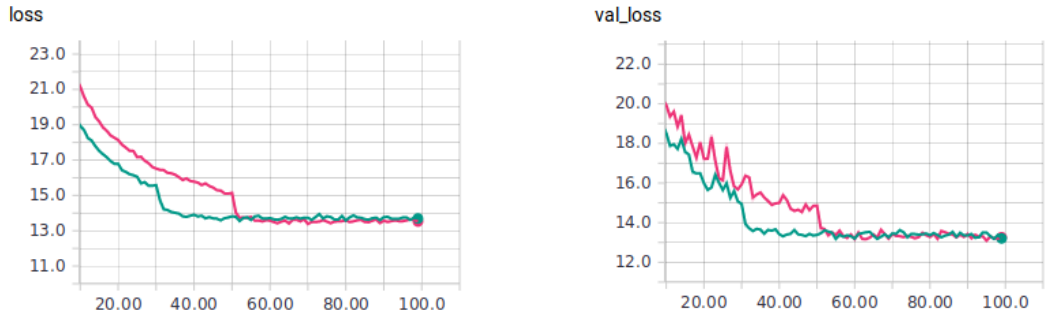


Σχήμα 4.21: Σύγκριση της εκπαίδευσης δικτύου με χρήση group normalization (κόκκινη γραμμή) και δικτύου με χρήση batch normalization (πορτοκαλί γραμμή) για τη συλλογή με τις 4 κλασεις.

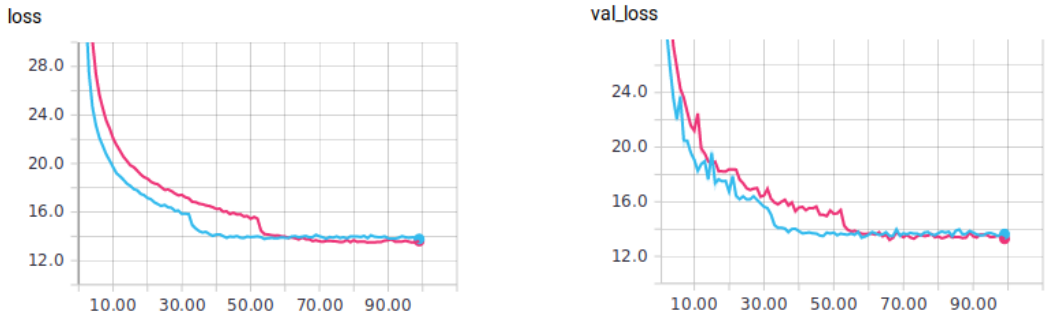
Η εξήγηση αυτών των αποτελεσμάτων θα παρουσιαστεί στο επόμενο κεφάλαιο της εργασίας.



Σχήμα 4.22: Σύγκριση της εκπαίδευσης δικτύου με χρήση group normalization (μπλε γραμμή) και δικτύου με χρήση batch normalization (πορτοκαλί γραμμή) from scratch για τη συλλογή με τα πλοία.



Σχήμα 4.23: Σύγκριση της εκπαίδευσης δικτύου με χρήση group normalization (ροζ γραμμή) και δικτύου με χρήση batch normalization (πράσινη γραμμή) from scratch για τη συλλογή με τις 3 κλάσεις.



Σχήμα 4.24: Σύγκριση της εκπαίδευσης δικτύου με χρήση group normalization (ροζ γραμμή) και δικτύου με χρήση batch normalization (γαλάζια γραμμή) from scratch για τη συλλογή με τις 4 κλάσεις.

4.4 Αναλυτικά Αποτελέσματα

Στην ενότητα που ακολουθεί θα παρουσιαστούν αναλυτικότερα τα αποτελέσματα που εξήχθησαν από τα πειράματα για κάθε κατηγορία όπως και ορισμένες εικόνες στις οποίες έχει διεξαχθεί ανίχνευση αντικείμενων.

Πιο συγκεκριμένα, έχουμε αναφέρει τον όγκο των δεδομένων στην κάθε συλλογή στο πρώτο μέρος αυτού του κεφαλαίου. Πρέπει να επισημάνουμε όμως και τα στιγμιότυπα που έχει κάθε κλάση στο σύνολο δεδομένων πριν προβούμε σε αναλυτική παρουσίαση των αποτελεσμάτων. Στην πρώτη συλλογή με τα πλοία, η ανάλογη ετικέτα εμφανίζεται 777 φορές.

Στη δεύτερη συλλογή οι κλάσεις έχουν την εξής κατανομή: ’car’: 42342, ’pedestrian’: 3062, ’truck’: 2786.

Συλλογή Δεδομένων: Πλοία	
Μοντέλο	AP
Χωρίς residual blocks	0
Κλασικό	94,75
Περισσότερα residual blocks	**95,91**
Group normalization	27,26
Μοντέλο χωρίς αρχικοποίηση	AP
Κλασικό	80,42
Περισσότερα residual blocks	**85,51**
Group normalization	53,52

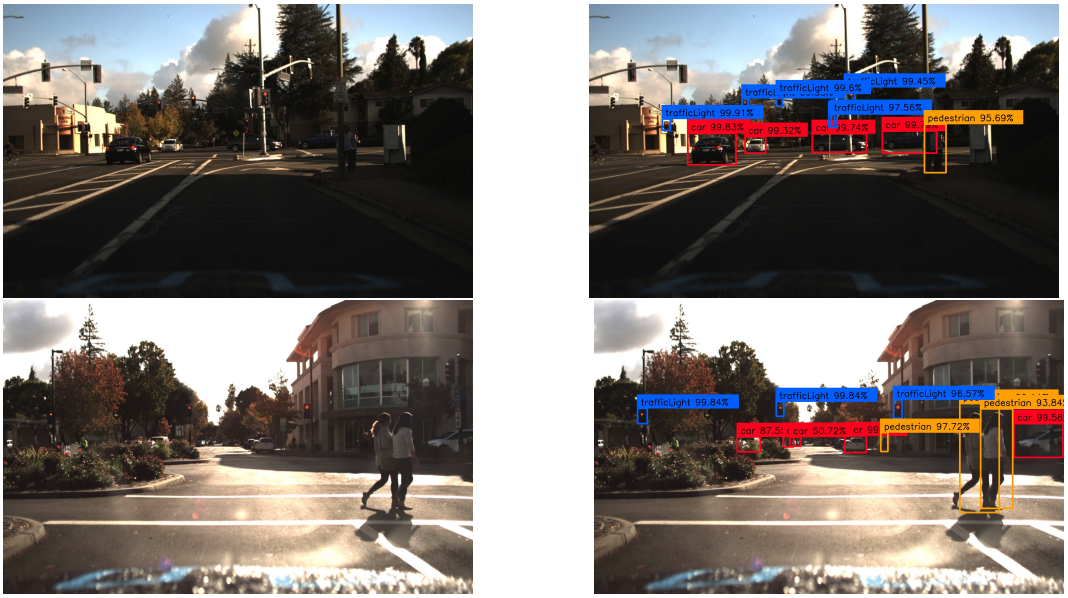
Σχήμα 4.25: Παραδείγματα ανίχνευσης 1

Συλλογή Δεδομένων: Udacity-1				
Μοντέλο	car	pedestrian	truck	mAP
Χωρίς residual blocks	75,22	16,87	43,87	45,32
Κλασικό	85,14	46,49	66,12	**65,92**
Περισσότερα residual blocks	84,83	45,07	65,71	65,20
Group normalization	84,11	40,63	64,53	63,09
Μοντέλο χωρίς αρχικοποίηση	car	pedestrian	truck	mAP
Κλασικό	76,55	16,29	51,59	48,14
Περισσότερα residual blocks	78,34	19,45	54,29	**50,69**
Group normalization	77,65	20,97	50,66	49,76

Σχήμα 4.26: Παραδείγματα ανίχνευσης 2

Ενώ στην τρίτη συλλογή: ’car’: 33934, ’trafficLight’: 10970, ’pedestrian’: 7095, ’truck’: 1870.

Συλλογή Δεδομένων: Udacity-2					
Μοντέλο	car	pedestrian	truck	traffic lights	mAP
Χωρίς residual blocks	64,54	20,03	32,00	58,58	43,79
Κλασικό	72,37	42,20	61,32	66,12	**60,50**
Περισσότερα residual blocks	71,97	42,59	55,91	65,89	59,09
Group normalization	66,00	30,72	43,15	61,16	50,26
Μοντέλο χωρίς αρχικοποίηση	car	pedestrian	truck	traffic lights	mAP
Κλασικό	67,67	25,36	49,35	63,43	51,45
Περισσότερα residual blocks	68,12	23,34	51,19	65,52	52,04
Group normalization	67,03	29,05	51,04	62,44	**52,39**



Σχήμα 4.27: Παραδείγματα ανίχνευσης 3

Γίνεται εύκολα αντιληπτό ότι η κλάση 'car' κυριαρχεί σε μεγάλο βαθμό έναντι των άλλων και ως εκ τούτου εκεί περιμένουμε τα καλύτερα αποτελέσματα όπως θα διαπιστωθεί. Αυτό συμβαίνει επειδή η συλλογή αποτελείται από εικόνες σε πραγματική ροή οδηγικής κυκλοφορίας όπου αναμένουμε τα αυτοκίνητα να αποτελούν την πλειοψηφία. Τα αποτελέσματά μας θα είχαν πολύ μικρότερη διακύμανση αν είχαν εφαρμοστεί τεχνικές εμπλουτισμού δεδομένων ή χρήση βαρών στις κλάσεις αλλά αυτό ξεφεύγει από τους σκοπούς της παρούσας εργασίας.

Κεφάλαιο 5

Συμπεράσματα

Σκοπός της συγκεκριμένης εργασίας ήταν πρωταρχικά η τριβή με το διαρκώς εξελισσόμενο χώρο της μηχανικής μάθησης και πιο συγκεκριμένα στο πεδίο του εντοπισμού αντικειμένων. Για αυτό το σκοπό, σε πρώτο στάδιο, έγινε σύγκριση και αξιολόγηση 2 βασικών αλγορίθμων του χώρου, του Faster R-CNN και του YOLO. Η σύγκρισή μας απέδειξε ότι ιδιαίτερα στο πεδίο του χρόνου εκπαίδευσης, η ανωτερότητα του YOLO ήταν εμφανής κάτι που αναμενόταν καθώς πρόκειται για έναν single shot detector που όμως δεν υστερούσε σε ακρίβεια.

Έπειτα, επιχειρήθηκε λεπτομερέστερη ανάλυση των αποτελεσμάτων του YOLO, προσπαθώντας να διαπιστώσουμε την σημασία των residual blocks. Πράγματι, η βαθιά μάθηση αποδείχτηκε πολύ υποδεέστερη με τη χρήση απλών (plain) δικτύων. Αυτό το συμπέρασμα μας ώθησε σε περαιτέρω επέκταση του αλγορίθμου με residual blocks και πέραν του backbone δικτύου αλλά και στην χρήση τεχνικών RoR.

Τα αρχικά αποτελέσματα δεν ήταν ενθαρρυντικά αλλά αποδείξαμε ότι αυτό οφειλόταν στη χρήση προεκπαιδευμένων βαρών για το backbone δίκτυο, που πλέον είχε διαφορετική αρχιτεκτονική με τις αλλαγές που διεξάγαμε. Με την εκπαίδευση να γίνεται εκ του μηδενός, είχαμε βελτίωση σε όλες τις χρησιμοποιημένες βάσεις δεδομένων.

Επόμενο κομμάτι της εργασίας ήταν η αντικατάσταση του Batch Normalization με την καινοτόμο μέθοδο του Group Normalization που προτείνεται για χρήση, όταν οι απαιτήσεις της μνήμης ή/και το μικρό μέγεθος της GPU δεν επιτρέπει μεγάλο αριθμό στο batch size. Σε πλήρη αντιστοιχία με πριν, τα αποτελέσματα ήταν δυσμενέστερα, αρχικά, αλλά αυτό συνέβαινε, ξανά, εξαιτίας των προεκπαιδευμένων βαρών. Χωρίς τη χρήση βαρών, η ακρίβεια βελτιώθηκε και πάλι στις μεγάλες βάσεις δεδομένων. Μόνη εξαίρεση αποτέλεσε το dataset με τα πλοία, το οποίο εξαιτίας της μεγάλης απλότητας του φάνηκε να μην επηρεάζεται από το μικρό αριθμό του batch size.

Όποτε ένα πεδίο εξέλιξης της εργασίας είναι η ευρύτερη χρήση του νέου δικτύου και για ταξινόμηση, κάτι που κατ' επέκταση θα έχει και ευεργετικές ιδιότητες στη χρήση του backbone δικτύου για ανίχνευση αντικειμένων. Αυτό θα επέτρεπε την άμεση σύγκρισή του με βάση τα datasets που χρησιμοποιούνται για ανίχνευση αντικειμένων σε επίπεδο διεθνών διαγωνισμών όπως το COCO[Lin14]αλλά και την ανάπτυξη εφαρμογών που αξιοποιούν αυτήν την τεχνολογία.

Επιπροσθέτως, η εργασία επιβεβαιώνει τα μέχρι τώρα συμπεράσματα ότι η χρήση του group normalization επιφέρει θετικά αποτελέσματα στην ακρίβεια και ανοίγει το δρόμο για περαιτέρω χρήση της σε ζητήματα που καταναλώνουν υπολογιστική μνήμη. Τέλος, μελλοντικό κομμάτι που αξίζει να διερευνηθεί είναι ο συνδυασμός των μεθόδων που αναλύσαμε, χρήση νέου δικτύου και group normalization, με στόχο ακόμα μεγαλύτερη βελτίωση.

Βιβλιογραφία

[Aize13] Igor Aizenberg, Naum N Aizenberg and Joos PL Vandewalle, *Multi-Valued and Universal Binary Neurons: Theory, Learning and Applications*, Springer Science & Business Media, 2013.

[Amit14] Yali Amit and Pedro Felzenszwalb, "Object Detection", *Computer Vision: A Reference Guide*, pp. 537–542, 2014.

[Anh18] Huynh Ngoc Anh, "keras-yolo2", https://github.com/experiencor/keras-yolo2, 2018.

[Berk97] Istvan SN Berkeley, "A revisionist history of connectionism", *Unpublished manuscript*, 1997.

[Chen80] PC Chen and T Pavlidis, "Image segmentation as an estimation problem", *Computer Graphics and Image Processing*, vol. 12, no. 2, pp. 153–172, 1980.

[Chen17] Xinlei Chen and Abhinav Gupta, "An Implementation of Faster RCNN with Study for Region Sampling", *arXiv preprint arXiv:1702.02138*, 2017.

[Chen18] Jinghui Chen and Quanquan Gu, "Closing the generalization gap of adaptive gradient methods in training deep neural networks", *arXiv preprint arXiv:1806.06763*, 2018.

[Dech86] Rina Dechter, *Learning while searching in constraint-satisfaction problems*, University of California, Computer Science Department, Cognitive Systems Laboratory, 1986.

[Domi12] Pedro Domingos, "A few useful things to know about machine learning", *Communications of the ACM*, vol. 55, no. 10, pp. 78–87, 2012.

[Girs14] Ross Girshick, Jeff Donahue, Trevor Darrell and Jitendra Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014.

[Girs15] Ross Girshick, "Fast R-CNN", in *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[Han95] Jun Han and Claudio Moraga, "The influence of the sigmoid function parameters on the speed of backpropagation learning", in *International Workshop on Artificial Neural Networks*, pp. 195–201, Springer, 1995.

[He16] Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun, "Deep residual learning for image recognition", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[Hodg12] Andrew Hodges, *Alan Turing: The Enigma: The Enigma*, Random House, 2012.

[Hsu04] Feng-Hsiung Hsu, *Behind Deep Blue: Building the computer that defeated the world chess champion*, Princeton University Press, 2004.

[Ioff15] Sergey Ioffe and Christian Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift", *arXiv preprint arXiv:1502.03167*, 2015.

[Kriz12] Alex Krizhevsky, Ilya Sutskever and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks", in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[LeCu91] Yann LeCun, Ido Kanter and Sara A Solla, "Second order properties of error surfaces: Learning time and generalization", in *Advances in neural information processing systems*, pp. 918–924, 1991.

[LeCu98] Yann LeCun, Léon Bottou, Yoshua Bengio and Patrick Haffner, "Gradient-based learning applied to document recognition", *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[Lin14] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár and C Lawrence Zitnick, "Microsoft coco: Common objects in context", in *European conference on computer vision*, pp. 740–755, Springer, 2014.

[Linn76] Seppo Linnainmaa, "Taylor expansion of the accumulated rounding error", *BIT Numerical Mathematics*, vol. 16, no. 2, pp. 146–160, 1976.

[Maas13] Andrew L Maas, Awni Y Hannun and Andrew Y Ng, "Rectifier nonlinearities improve neural network acoustic models", in *Proc. icml*, vol. 30, p. 3, 2013.

[McCa06] John McCarthy, Marvin L Minsky, Nathaniel Rochester and Claude E Shannon, "A proposal for the dartmouth summer research project on artificial intelligence, august 31, 1955", *AI magazine*, vol. 27, no. 4, p. 12, 2006.

[Mitc06] Tom Michael Mitchell, *The discipline of machine learning*, vol. 9, Carnegie Mellon University, School of Computer Science, Machine Learning Department Pittsburgh, PA, 2006.

[Nair10] Vinod Nair and Geoffrey E Hinton, "Rectified linear units improve restricted boltzmann machines", in *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814, 2010.

[Redm16] Joseph Redmon, Santosh Divvala, Ross Girshick and Ali Farhadi, "You only look once: Unified, real-time object detection", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

[Redm17] Joseph Redmon and Ali Farhadi, "YOLO9000: better, faster, stronger", *arXiv preprint*, 2017.

[Redm18] Joseph Redmon and Ali Farhadi, "Yolov3: An incremental improvement", *arXiv preprint arXiv:1804.02767*, 2018.

[Ren15] Shaoqing Ren, Kaiming He, Ross Girshick and Jian Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks", in *Advances in neural information processing systems*, pp. 91–99, 2015.

[Rose57] Frank Rosenblatt, *The perceptron, a perceiving and recognizing automaton Project Para*, Cornell Aeronautical Laboratory, 1957.

[Rude16] Sebastian Ruder, "An overview of gradient descent optimization algorithms", *arXiv preprint arXiv:1609.04747*, 2016.

[Rume86] David E Rumelhart, Geoffrey E Hinton and Ronald J Williams, "Learning representations by back-propagating errors", *nature*, vol. 323, no. 6088, p. 533, 1986.

[Russ95] Stuart Russell, Peter Norvig and Artificial Intelligence, "A modern approach", *Artificial Intelligence. Prentice-Hall, Egnlewood Cliffs*, vol. 25, no. 27, pp. 79–80, 1995.

[Samu59] Arthur L Samuel, "Some studies in machine learning using the game of checkers", *IBM Journal of research and development*, vol. 3, no. 3, pp. 210–229, 1959.

[Simo14] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition", *arXiv preprint arXiv:1409.1556*, 2014.

[Sriv14] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever and Ruslan Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting", *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[Turi09] Alan M Turing, "Computing machinery and intelligence", in *Parsing the Turing Test*, pp. 23–65, Springer, 2009.

[Waib90] Alexander Waibel, Toshiyuki Hanazawa, Geoffrey Hinton, Kiyohiro Shikano and Kevin J Lang, "Phoneme recognition using time-delay neural networks", in *Readings in speech recognition*, pp. 393–404, Elsevier, 1990.

[Wu18] Yuxin Wu and Kaiming He, "Group normalization", in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 3–19, 2018.

[Zhan18] Ke Zhang, Miao Sun, Tony X Han, Xingfang Yuan, Liru Guo and Tao Liu, "Residual networks of residual networks: Multilevel residual networks", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 6, pp. 1303–1314, 2018.