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

Ανάλυση ιατρικών εικόνων με χρήση τεχνικών βαθιάς μάθησης

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΧΑΡΑΛΑΜΠΟΣ ΚΩΣΤΟΠΟΥΛΟΣ

Επιβλέπων : Παναγιώτης Τσανάκας
Καθηγητής Ε.Μ.Π.

Αθήνα, Νοέμβριος 2019

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

Ανάλυση ιατρικών εικόνων με χρήση τεχνικών βαθιάς μάθησης

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΧΑΡΑΛΑΜΠΟΣ ΚΩΣΤΟΠΟΥΛΟΣ

Επιβλέπων : Παναγιώτης Τσανάκας
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 20η Νοεμβρίου 2019.

Παναγιώτης Τσανάκας Καθηγητής Ε.Μ.Π.	Δημήτριος-Διονύσιος Κουτσούρης Καθηγητής Ε.Μ.Π.	Γιώργος Ματσόπουλος Καθηγητής Ε.Μ.Π.

Αθήνα, Νοέμβριος 2019

Χαράλαμπος Κωστόπουλος

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

Περίληψη

Ο τομέας της τεχνητής νοημοσύνης έχει παρουσιάσει μια ραγδαία ανάπτυξη τα τελευταία χρόνια και αυτό οφείλεται αρκετά στην μηχανική μάθηση και ειδικότερα στην εξέλιξη των νευρωνικών δικτύων και τη βαθιά μηχανική μάθηση. Με τον όρο deep learning εννοούμε τη χρήση νευρωνικών δικτύων πολλών στρωμάτων για τη ανάλυση και εύρεση μοτίβων σε πολλά δεδομένα. Η εφαρμογή αυτής της τεχνολογίας ποικίλλει ενώ κάθε μέρα παρουσιάζονται καινούριοι δρόμοι για την χρησιμοποίηση της.

Σκοπός αυτής της εργασίας είναι αρχικά η παρουσίαση του τομέα της βαθιάς μηχανικής μάθησης ξεκινώντας από βασικές έννοιες και έπειτα εξειδικεύοντας στην ανάλυση και επεξεργασία εικόνας. Παρουσιάζονται και αναλύονται όλες οι πτυχές που απαρτούν ένα νευρωνικό δίκτυο, με σκοπό την εφαρμογή στην υπολογιστική όραση, όπως το κλασσικό νευρωνικό δίκτυο,το συνελικτικό δίκτυο και η διαδικασία εκπαίδευσης τους μέσω συναρτήσεων σφάλματος, αλγόριθμο απότομης καθόδου κλπ. Επίσης παρουσιάζονται μέθοδοι για την αξιολόγηση τέτοιων μοντέλων αλλά και έννοιες που χρησιμεύουν στην ανίχνευση αντικειμένων σε εικόνες και την κατάτμηση τους.
Έπειτα γίνεται μια παρουσίαση από κάποιες συγκεκριμένες τεχνολογίες και αρχιτεκτονικές μοντέλων βασισμένες στα συνελικτικά δίκτυα που θεωρούνται state of the art μέχρι και την ολοκλήρωση της παρούσας εργασίας. Αυτά τα μοντέλα θα χρησιμοποιηθούν και στο πειραματικό μέρος της εργασίας.

Αφότου ολοκληρωθεί η παραπάνω περιγραφή και έχουν τεθεί τα θεμέλια για την κατανόηση των τεχνολογιών προχωράμε σε μια εφαρμογή τους στην επίλυση ενός προβλήματος ιατρικής σημασίας. Συγκεκριμένα επικεντρωνόμαστε στην ανάλυση και επεξεργασία δερματικών σπίλων με σκοπό πρώτον τον προσδιορισμό των ορίων μέσω segmentation και έπειτα την ταξινόμηση τους σε διάφορες κλάσεις. Μέσα σε αυτές τις κλάσεις ανήκει και το γνωστό μελάνωμα που αποτελεί καρκίνο του δέρματος και είναι ιδιαίτερα επιθετικό καθώς και άλλες μορφές καρκινώματος. Ο σκοπός μας είναι να εφαρμόσουμε όσες περισσότερες τεχνικές προ επεξεργασίας και εκπαίδευσης διαθέτουμε για να μεγιστοποιήσουμε την απόδοση του μοντέλου μας στην είσοδο νέων δεδομένων. Για αυτό δοκιμάζουμε διάφορες παραμέτρους και εκδόσεις αρχιτεκτονικών και τις συγκρίνουμε επιλέγοντας τις καλύτερες.

Τέλος συνδυάζουμε τα δύο προβλήματα της κατάτμησης και της ταξινόμησης για να δημιουργήσουμε ένα τελικό αποτέλεσμα ενός δίαυλου επεξεργασίας της εικόνας όπως τη λαμβάνουμε από το στάδιο της δερμοσκόπησης. Από την απόδοση που πετύχαμε καταλήγουμε σε διάφορα συμπεράσματα για την χρησιμότητα ενός τέτοιου εργαλείου σε πραγματικές εφαρμογές και αναφερόμαστε σε τρόπους περαιτέρω βελτίωσης με τεχνικές που χρησιμοποιούνται κατά κόρον σήμερα σε τέτοια προβλήματα.

Λέξεις κλειδιά

Τεχνητή Νοημοσύνη, Νευρωνικά Δίκτυα, Συνελικτικά Νευρωνικά Δίκτυα, Faster R-CNN, Mask R-CNN,DeepLab, Residual Networks, Κατάτμηση ,Ταξινόμηση , Δερματικές Αλλοιώσεις

Abstract

Artificial intelligence is rapidly advancing the last few years and a big part of this is the advancement in machine learning and specifically the use of neural networks and deep learning. With the term deep learning we mean those neural networks that consist of many layers of neurons forming a deep pipeline of computation. This particular technology can be applied in a diverse set of problems and each day there are more ways that are discovered in which we can use and apply these techniques.

The goal of this study is initially the presentation of deep learning, starting off at the very basics and then moving on to the architectures that are used in computer vision and image processing. We present and analyze the componets of a neural network, a convolutional network and the ways we currently have to train them and test them in real data through different loss functions,optimization algorithms and evaluation metrics. We then present some of the models that are considered state of the art in the department of object detection, segmentation and image classification. It needs to be reminded though that the advancements are so rapid that todays ”state of the art” might be obsolete in a few years. In any case these models are going to be used in the last part of this study as we try to apply them in a real world scenario.

So after the initial theoretical background is thoroughly presented we will apply these concepts to the problem of dermatic lesions. In particular we are interested initally in segmenting and defining the lesion from the background image. After that we will try to classify each lesion to one of eight different categories. These categories also contain the infamous melanoma a very aggressive type of skin cancer as well as other malignant tumours. It is very important to choose the appropriate metric and to focus more on finding the bad cases in the lesions since those are the ones that have the higher cost for the well being of the patient. For this we try different models and architectures as well as parameterization and data preprocessing to achieve the best performance that we can in the context of the means we have.

Finally we combine the segmentation part and the classficication part to create the final end to end pipeline and we discuss some issues, its general usefullness and further work to be done.

Key words

Artificial Intelligence, Neural Networks, Convolutional Neural Networks, Faster R-CNN, Mask R-CNN, DeepLab, Residual Networks, Segmentation, Classification, Dermatic Lesions

Ευχαριστίες

Ευχαριστώ θερμά τον κύριο Παναγιώτη Τσανάκα καθηγητή Ε.Μ.Π. που μου εμπιστεύθηκε το παρακάτω θέμα διπλωματικής σε ένα τομέα που αναπτύσσεται ραγδαία και βρίσκεται διαρκώς στην επικαιρότητα. Επίσης ευχαριστώ θερμά τον κύριο Ηλιά Μαγκλογιάννη Αναπληρωτή Καθηγητή στο Πανεπιστήμιο Πειραιώς που με καθοδήγησε στα διάφορα στάδια της διπλωματικής και με έφερε σε πρωταρχική επαφή με τις τεχνολογίες που χρησιμοποιήθηκαν. Τέλος ευχαριστώ τον φίλο μου Άγγελο και την αδερφή μου Καλλιόπη για την πολύτιμη βοήθεια τους σε διάφορα στάδια της διπλωματικής και γενικά.

Χαράλαμπος Κωστόπουλος,

Αθήνα, 20η Νοεμβρίου 2019

Περιεχόμενα

Κατάλογος πινάκων

Κατάλογος σχημάτων

Κεφάλαιο 1

Εισαγωγή

1.1 Η μηχανική μάθηση

Ζούμε σε μια εποχή που η επιστήμη και η τεχνολογία προοδεύει με τόσο γρήγορους ρυθμούς που η πραγματικότητα τείνει να συγκλίνει με την επιστημονική φαντασία. Μπορεί ακόμα να μην έχουμε πετύχει την λεγόμενη δημιουργία συνείδησης σε αυτόνομες μηχανές αλλά , όλο και περισσότερες λειτουργίες τείνουν να αυτοματοποιηθούν από συστήματα που ”μαθαίνουν” αντί να προγραμματίζονται.

Η Μηχανική Μάθηση λοιπόν είναι ένας τομέας που ανήκει στην Τεχνητή Νοημοσύνη και όπου αντί για αλγορίθμους που λύνουν άμεσα ένα πρόβλημα έχουμε αλγορίθμους που μαθαίνουν να λύνουν ένα πρόβλημα μέσα από τα ίδια τα δεδομένα και την πληροφορία τους. Τέτοιοι αλγόριθμοι επιτρέπουν ήδη από σήμερα σε μηχανές να οδηγούν αυτοκίνητα να γράφουν αναφορές περιστατικών ακόμα και να καταγράφουν ύποπτη δραστηριότητα σε βίντεο από κάμερες ασφαλείας ,ότι επιπτώσεις και αν έχει αυτό ως προς την ηθική του πλευρά.

Στην παρούσα εργασία θα αξιοποιηθεί ένα υποσύνολο των παραπάνω αλγορίθμων, με τις όποιες εξελίξεις τους έως σήμερα, πάνω σε ένα πρόβλημα αναγνώρισης και κατηγοριοποίησης αντικειμένων με στόχο την ανάλυση ιατρικής πληροφορίας. Συγκεκριμένα θα εστιαστεί η προσοχή μας στις δερματικές αλλοιώσεις/σπίλοι (κοινώς δερματικές ελιές) και θα γίνει προσπάθεια για τη δημιουργία ενός συστήματος για διάγνωση πιθανής κακοήθειας σε αυτές. Πριν προχωρήσουμε όμως αξίζει να παρουσιάσουμε κάποιες έννοιες της μηχανικής μάθησης που είναι χρήσιμες γενικά.

1.1.1 Κατηγορίες μηχανικής μάθησης

Η μηχανική μάθηση χωρίζεται στις εξής 3 μεγάλες κατηγορίες. Ο διαχωρισμός γίνεται με βάση τη σύσταση των δεδομένων και τον τρόπο της εκμάθησης. Γενικά οι ορισμοί αυτών των κατηγοριών συναντιόνται στη βιβλιογραφία διατυπωμένοι όπως παρακάτω.

Επιβλεπόμενη μάθηση

Η επιβλεπόμενη μάθηση (supervised learning) είναι η κατηγορία στην οποία τα δεδομένα μας έχουν την εξής μορφή. Κάθε παράδειγμα αποτελείται από ένα σύνολο εισόδου (συνήθως ένα διάνυσμα από χαρακτηριστικά) και μια επιθυμητή τιμή εξόδου (είτε από ένα διακριτό σύνολο τιμών είτε από ένα συνεχές όπως μια αριθμητική τιμή) . Οι Αλγόριθμοι επιβλεπόμενης μάθησης αναλύουν τα δεδομένα εκπαίδευσης και παράγουν ένα μοντέλο το οποίο μπορεί να χρησιμοποιηθεί για να χαρακτηρίσει νέα παραδείγματα. Το βέλτιστο σενάριο επιτρέπει στον αλγόριθμο να καθορίσει σωστά την ετικέτα της κατηγορίας για άγνωστα μέχρι τώρα παραδείγματα. Για να επιτευχθεί αυτό, απαιτείται ο αλγόριθμος μάθησης να γενικεύει από τα δεδομένα εκπαίδευσης σε αθέατες καταστάσεις με ένα ”λογικό” τρόπο.

Μη επιβλεπόμενη μάθηση

Η μη-επιβλεπόμενη μάθηση (unsupervised learning)αποτελεί κατηγορία της μηχανικής μάθησης, στην οποία επιθυμούμε να ανακαλύψουμε κάποια πιθανής δομής που μπορεί να κρύβεται πίσω από δεδομένα που δεν έχουν κάποια ετικέτα που τα χαρακτηρίζει. Εφόσον τα παραδείγματα τα οποία χρησιμοποιούνται δεν έχουν χαρακτηριστεί, δεν υπάρχει σφάλμα με το οποίο αξιολογούμε πιθανες λύσεις.
Αυτό είναι που διακρίνει την μη-επιβλεπόμενη μάθηση από την επιβλεπόμενη μάθηση και την ενισχυτική (ήμι-επιβλεπόμενη) μάθηση.

Ενισχυτική (ήμι-επιβλεπόμενη)

Η ενισχυτική μάθηση (reinforcement learning) στην επιστήμη των υπολογιστών είναι ένας γενικός όρος που έχει δοθεί σε μια οικογένεια τεχνικών στις οποίες το σύστημα μάθησης προσπαθεί να μάθει μέσα από αλληλεπίδραση με το περιβάλλον. Η έννοια της ενισχυτικής μάθησης είναι εμπνευσμένη από τα αντίστοιχα ανάλογα της μάθησης με επιβράβευση και τιμωρία που συναντώνται ως μοντέλα μάθησης των έμβιων όντων. Σκοπός του συστήματος μάθησης είναι να μεγιστοποιήσει μια συνάρτηση του αριθμητικού σήματος ενίσχυσης (ανταμοιβή), για παράδειγμα την αναμενόμενη τιμή του σήματος ενίσχυσης στο επόμενο βήμα. Το σύστημα δεν καθοδηγείται από κάποιον εξωτερικό επιβλέποντα για το ποια ενέργεια θα πρέπει να ακολουθήσει αλλά πρέπει να ανακαλύψει μόνο του ποιες ενέργειες είναι αυτές που θα του αποφέρουν το μεγαλύτερο κέρδος.

Για την παρούσα εργασία θα γίνει μελέτη μόνο της πρώτης κατηγορίας αφού τα δεδομένα εκπαίδευσης μας θα αποτελούνται από την εικόνα και τις θέσεις των αλλοιώσεων καθώς και την ταμπέλα με την πληροφορία αν είναι κακοήθεις ή όχι.

1.2 Το πρόβλημα της ανάλυσης και επεξεργασίας εικόνας από μηχανές

1.2.1 Τα διαφορετικά είδη της ανάλυσης μιας εικόνας

Έστω ότι έχουμε στα χέρια μας μια εικόνα και επιδιώκουμε να εξάγουμε πληροφορία από αυτήν. Ποια είναι αυτή η πληροφορία που επιθυμούμε και πως θα δημιουργήσουμε την επιθυμητή έξοδο ; Πρέπει να ορίσουμε πρώτα το πρόβλημα που θέλουμε να επιλύσουμε.

Στο Σχήμα 1.1 παρουσιάζονται τα διάφορα είδη προβλήματος που καλείται ένα πρόγραμμα μηχανικής μάθησης να λύσει όσον αφορά την μηχανική όραση. Ανάλογα το είδος της επιθυμητής εξόδου έχουμε διαφορετικές προσεγγίσεις στην ανάλυση της εικόνας και την αρχιτεκτονική του αλγορίθμου όπως θα περιγραφεί αργότερα.

Ταξινόμηση εικόνας Θέλουμε να ταξινομήσουμε μια εικόνα βάσει το κυρίαρχο αντικείμενο το οποίο περιέχεται μέσα σε αυτήν (για παράδειγμα μια γάτα ή ένας σκύλος). Στην ουσία έχουμε έναν ταξινομητή με είσοδο τα εικονοστοιχεία της εικόνας.

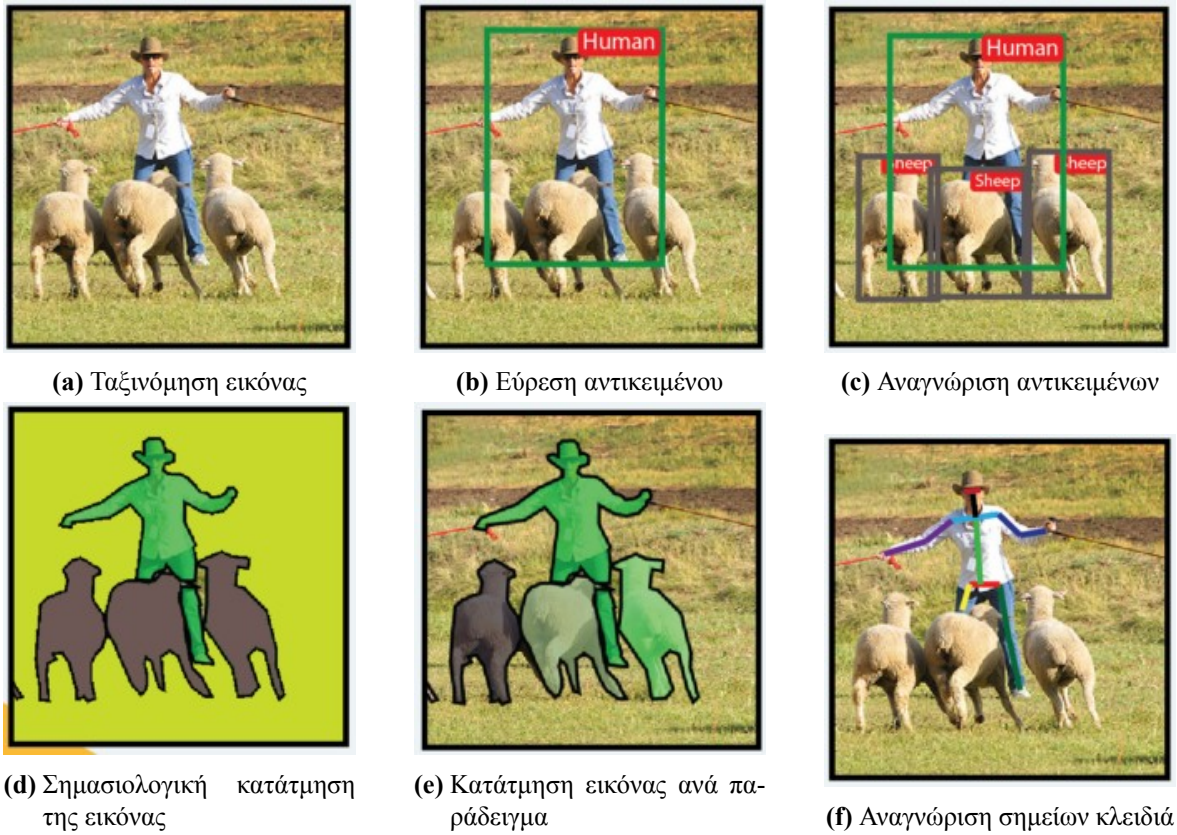
Εύρεση αντικειμένου Θέλουμε να προβλέψουμε την περιοχή της εικόνας η οποία περιέχει το κυρίαρχο αντικείμενό.

Αναγνώριση αντικειμένων Θέλουμε να αναγνωρίσουμε τις θέσεις όλων των αντικειμένων στην εικόνα καθώς και να προσδιορίσουμε το είδος τους.

Σημασιολογική κατάτμηση της εικόνας Θέλουμε να κατηγοριοποιήσουμε κάθε σημείο της εικόνας ξεχωριστά ανάλογα σε ποια κλάση ανήκει. Για παράδειγμα με πράσινο για κάθε σημείο που ανήκει σε αντικείμενο -άνθρωπο.

Κατάτμηση εικόνας ανά παράδειγμα Θέλουμε όχι μόνο να χρωματίσουμε τα διαφορετικά σημεία της εικόνας ανά κλάση αντικειμένου αλλά και ανά αντικείμενο. Δηλαδή να μπορούμε να ξεχωρίσουμε διαφορετικά αντικείμενα που αντιστοιχούν στην ίδια κλάση, όπως δύο διαφορετικό άνθρωποι.

Αναγνώριση σημείων κλειδιά Θέλουμε να βρούμε μέσα στην εικόνα την τοποθεσία συγκεκριμένων προδιαγραφόμενων χαρακτηριστικών όπως ο σκελετός ενός ανθρώπου, τα ζυγωματικά του προσώπου κ.τ.λ.π.



(a) Ταξινόμηση εικόνας **(b)** Εύρεση αντικειμένου **(c)** Αναγνώριση αντικειμένων

(d) Σημασιολογική κατάτμηση της εικόνας **(e)** Κατάτμηση εικόνας ανά παράδειγμα **(f)** Αναγνώριση σημείων κλειδιά

Σχήμα 1.1: Είδη ανάλυσης μιας εικόνας

Προσπάθειες για την επίλυση κάποιων από αυτών των ζητημάτων έχουν γίνει εδώ και πολλά χρόνια πριν την άνθηση της μηχανικής μάθησης και των βαθιών νευρωνικών δικτύων. Για παράδειγμα ο πρώτος αποδοτικός αλγόριθμος για την αναγνώριση προσώπου σε καρέ εφευρέθηκε το 2001 (Viola-Jones Algorithm, 2001) [Viol04] ο οποίος μάλιστα δούλευε σε πραγματικό χρόνο. Αργότερα υπήρξαν και άλλες εξελίξεις όπως το 2005 με τη χρήση ιστογράμματος από προσανατολισμένες παραγώγους για την αναγνώριση πεζών (Navneet Dalal and Bill Triggs ”HOG”) [Dala05] . Το πρόβλημα με αυτές τις προσεγγίσεις όμως ήταν ότι χρειαζόταν σαφής προγραμματισμός από ανθρώπινο χέρι όλων των χαρακτηριστικών που θέλαμε να ανιχνευθούν όπως τα μάτια και η μύτη σε ένα πρόσωπο. Κάτι τέτοιο από μόνο του είναι αρκετά πολύπλοκο πριν συνυπολογίσουμε ότι θέλουμε να υπάρχει αμεταβλητότητα ως προς την θέση και τον προσανατολισμό των παραπάνω χαρακτηριστικών.

1.3 Το πρόβλημα του εντοπισμού δερματικών αλλοιώσεων και της κατηγοριοποίησης τους

1.3.1 Καρκίνοι δέρματος και διάγνωση

Ο καρκίνος του δέρματος είναι από το πιο συνήθεις καρκίνους στον κόσμο. Το μελάνωμα αποτελεί μόνο το 1% των δερματικών καρκίνων. Όμως αποτελεί αιτία για ένα πολύ μεγάλο ποσοστό θανάτων από αυτού του τύπου τους καρκίνους. Τα επιθετικό μελάνωμα συγκεκριμένα αποτελεί έναν από τους πιο ραγδαία διαδεδομένους καρκίνους στον κόσμο. Μάλιστα υπολογίζεται ότι έχει εμφανιστεί 62480 φορές με 8420 θανάτους στην Αμερική το 2008 [JEME08]. Συνήθως επηρεάζει άτομα με λευκό χρώμα ενώ είναι 20 φορές πιο σπάνιο σε άτομα με μελαψό χρώμα. Επίσης έχει βρεθεί άμεση συσχέτιση με την ηλικία καθώς ο μέσος όρος των ανθρώπων που διαγιγνώσκονται με αυτόν τον τύπο καρκίνου είναι άνω των 60 χρόνων. Η έγκαιρη διάγνωση είναι εξαιρετικά σημαντική καθώς το μελάνωμα μπορεί να γιατρευτεί με μια απλή τομή της προβληματικής περιοχής αν είναι νωρίς.

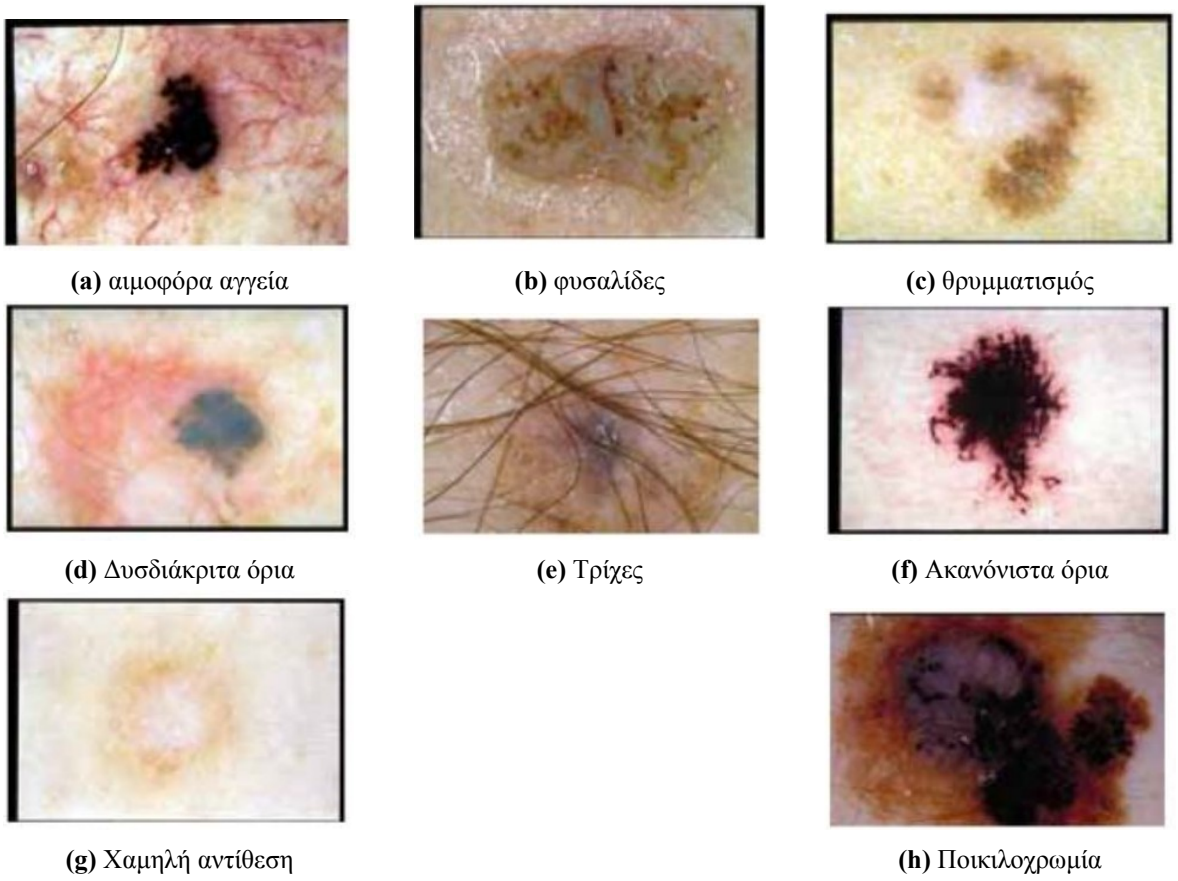
Η διάγνωση γίνεται με την χρήση μιας διαδικασίας που ονομάζεται δερματοσκόπηση. Η δερμοσκόποηση είναι μη επιθετική τεχνική που χρησιμοποιεί οπτική μεγέθυνσή και είτε οπτική βύθιση με υγρό και χαμηλή γωνία ανάκλασης φωτισμού είτε διασταυρωμένου πολωμένου φωτισμού, καθιστώντας τις υποεπιφανειακές δομές πιο ορατές σε σύγκριση με τις συμβατικές κλινικές εικόνες [Cele10]. Παρόλα αυτά έχει δειχτεί σε μελέτες [M95] ότι η δερμοσκόπηση μπορεί να μειώσει την διαγνωστική ακρίβεια στα χέρια μη έμπειρων δερματολόγων. Προκύπτει λοιπόν η ανάγκη για την μείωση της υποκειμενικότητας που έχει η ανθρώπινη ερμηνεία των εικόνων. Κάτι τέτοιο γίνεται εφικτό με την χρήση αυτοματοποιημένων μεθόδων για τον διαχωρισμό της αλλοίωσης από το υπόλοιπο κομμάτι. Τέτοιες μέθοδοι έχουν υπάρξει αρκετοί με διάφορα επίπεδα επιτυχίας. Στην παρούσα εργασία θα εφαρμοστούν μοντέλα μηχανικής μάθησης για να επιτευχθεί αυτό .

1.3.2 Προβλήματα στην ανάλυση

Η αυτοματοποίηση της διαδικασίας διαχωρισμού της αλλοίωσης από την υπόλοιπη περιοχή είναι εξαιρετικά σημαντική όμως και για την περαιτέρω ανάλυση της εικόνας για πολλούς λόγους. Αρχικά τα όρια της αλλοίωσης προσδίδουν σημαντική πληροφορία για την διάγνωση αφού πολλά χαρακτηριστικά όπως η ασυμμετρία και το απότομο απόκομμα της υπολογίζονται απευθείας από την περιοχή των ορίων. Έπειτα η εξαγωγή αρκετών κλινικών χαρακτηριστικών που δεν φαίνεται να σχετίζονται άμεσα με την περιοχή των ορίων έχουν έμμεση συσχέτιση με αυτή ή καλύτερα την ακρίβεια με την οποία έχει υπολογιστεί .

Δυστυχώς η αυτοματοποίηση της διαδικασίας βρίσκει αρκετά εμπόδια. Χαμηλή αντίθεση στο χρώμα της αλλοίωσης, αρκετά περίπλοκα όρια ,τρίχες και αιμοφόρα αγγεία είναι κάποια από αυτά που καλείται ένας αλγόριθμος κατάτμησης της εικόνας να υπερβεί . Στο Σχήμα 1.2 παρουσιάζονται κάποια από αυτά.

Εντέλει ο σκοπός μας είναι εδώ είναι ο διαχωρισμός των σπίλων που παρουσιάζουν κακοήθεια από τις υπόλοιπες. Ένας ταξινομητής είναι πολύ εύλογο να υποθέσουμε ότι θα παρουσιάζει αρκετά καλύτερη επίδοση, αν έχει να επεξεργαστεί την δερματική αλλοίωση κυρίως και όσον το δυνατό γίνεται λιγότερο από το παρασκήνιο της εικόνας. Η ανάλυση των δερματικών αλλοιώσεων εκτείνεται σε πολλά επίπεδα και σίγουρα θα είχε επίσης μεγάλο ενδιαφέρον η ενσωμάτωση πεδίου γνώσης στον αυτοματισμό της διαδικασίας. Κάτι τέτοιο όμως δεν θα γίνει εδώ αφού ο σκοπός είναι η χρήση γενικών μοντέλων για ταξινόμηση και αναγνώριση αντικειμένων σε αυτό το πρόβλημα.



(a) αιμοφόρα αγγεία **(b)** φυσαλίδες **(c)** θρυμματισμός

(d) Δυσδιάκριτα όρια **(e)** Τρίχες **(f)** Ακανόνιστα όρια

(g) Χαμηλή αντίθεση **(h)** Ποικιλοχρωμία

Σχήμα 1.2: Προβλήματα στην αναγνώριση δερματικής αλλοίωσης

Κεφάλαιο 2

Θεωρητικό υπόβαθρο

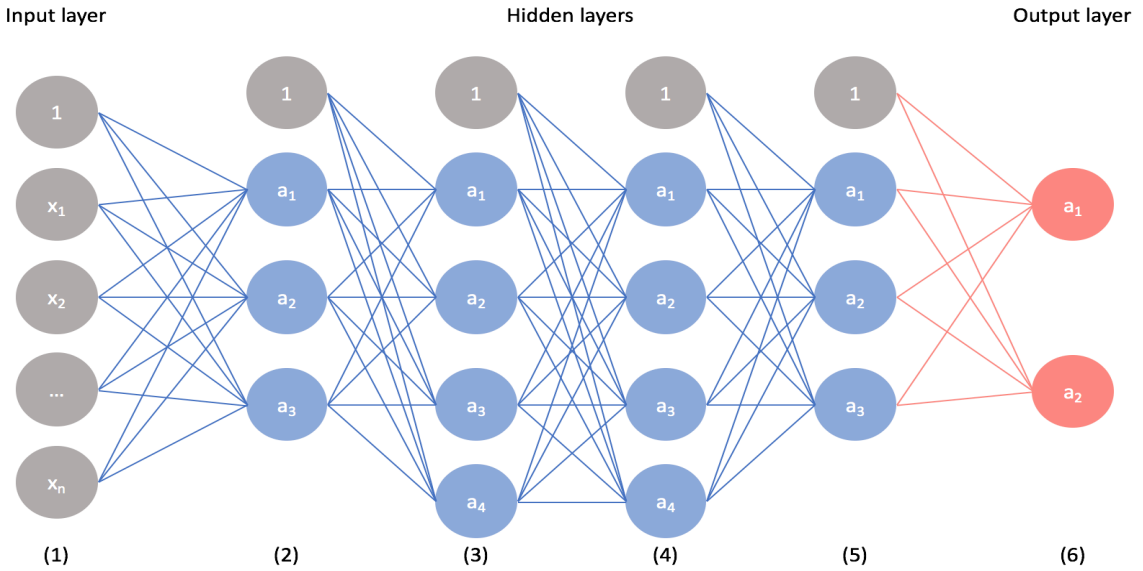
2.1 Εισαγωγή

Σε αυτό το σημείο θα γίνει μια επισκόπηση στην τεχνολογία που θα χρησιμοποιήσουμε στα πλαίσια αυτής της εργασίας. Θα αρχίσουμε με την παρουσίαση των απλών νευρωνικών δικτύων όπου θα περιγραφούν τα συστατικά που τα απαρτίζουν και ο τρόπος που αυτά συνδέονται. Επίσης θα αναφερθούμε στον τρόπο που αυτά τα δίκτυα μαθαίνουν από τα δεδομένα και τέλος σε κάποιες παραλλαγές και εξελίξεις σε αυτά που χρησιμοποιούνται ευρέως σήμερα. Έπειτα θα προχωρήσουμε στα συνελικτικά νευρωνικά δίκτυα όπου θα περιγραφούν αναλυτικά και σε σύγκριση πάντα με τα απλά νευρωνικά δίκτυα όλες οι αντίστοιχες πτυχές τους. Θεωρείται πως η παραπάνω γνώση είναι απαραίτητη ώστε έπειτα να προχωρήσουμε στην ανάλυση των ίδιων των δικτύων που θα χρησιμοποιηθούν για ανίχνευση αντικειμένων και για ταξινόμηση.

2.2 Τα νευρωνικά δίκτυα

2.2.1 Η αρχιτεκτονική ενός πλήρως συνδεδεμένου νευρωνικού δικτύου

Τα νευρωνικά δίκτυα άρχισαν ως μια απόπειρα για αναπαράσταση του ανθρώπινου εγκεφάλου και των νευρώνων που τον απαρτίζουν. Είναι δίκτυα που αποτελούνται από κόμβους που ονομάζονται νευρώνες και σκοπό έχουν την επεξεργασία της πληροφορίας από την είσοδο τους σε στρώματα ως την έξοδο. Η αρχιτεκτονική ενός κοινότυπου νευρωνικού δικτύου παρουσιάζεται στο Σχήμα 2.1.



Σχήμα 2.1: Ένα απλό πλήρως συνδεδεμένο νευρωνικό δίκτυο

Φαίνεται από το σχήμα ότι μπορούμε να διατάξουμε τους κόμβους του δικτύου σε στήλες και να διαβάσουμε το νευρωνικό από αριστερά προς τα δεξιά. Η πληροφορία της εισόδου αποτελεί το πρώτο στρώμα. Έπειτα κάθε νευρώνας έχει ακμές με όλους τους νευρώνες του προηγούμενου στρώματος δηλαδή δέχεται πληροφορία από όλους. Αυτή η πληροφορία χρησιμοποιείται από τον νευρώνα για να παραχθεί η έξοδος του. Οι έξτρα κόμβοι οι οποίοι αναγνωρίζονται με τον αριθμό 1 και που δεν παίρνουν είσοδο από κάπου στο σχήμα είναι ένας τρόπος να παρουσιάσουμε την λεγόμενη προκατάληψη (bias) που θα έχει ένας νευρώνας όταν θα υπολογίσει την έξοδο του (θα εξηγηθεί λίγο αργότερα).

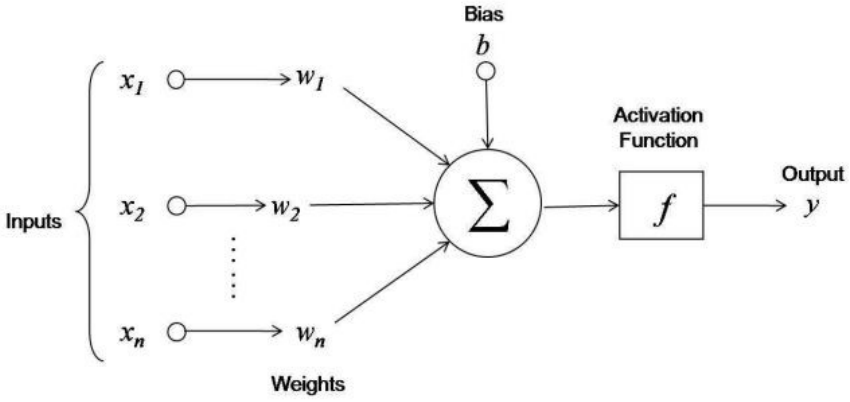
Έτσι ενεργοποιείται το επόμενο στρώμα και τώρα αυτό είναι σε θέση να δώσει είσοδο στο μεθεπόμενο στρώμα και ότου καθεξής. Στο τέλος υπάρχει το στρώμα εξόδου από όπου θα πάρουμε το αποτέλεσμα. Τα ενδιάμεσα στρώματα ονομάζονται κρυφά και όσον αφορά τη βαθιά μηχανική μάθηση έχουμε σήμερα δίκτυα που απαρτίζονται μέχρι και πάνω από εκατό κρυφά στρώματα.

Το ιδιαίτερα ενδιαφέρον στο παραπάνω δίκτυο είναι ότι ενώ για παράδειγμα η είσοδος θα μπορούσε να είναι τα εικονοστοιχεία μιας εικόνας ή ένα σύνολο από τιμές χαρακτηριστικών ενός αντικειμένου ή έξοδος μπορεί να έχει ότι μορφή θέλουμε. Έτσι η έξοδος των νευρώνων του τελευταίου στρώματος θα μπορούσε να αντιπροσωπεύει τις πιθανότητες να ανήκει το αντικείμενο με τα χαρακτηριστικά της εισόδου σε κάποια κλάση όπως σκύλος ή γάτα. Με την ίδια λογική θα μπορούσαν οι ίδιες τιμές να κωδικοποιούν την θέση ενός αντικειμένου! Το πως μαθαίνει το δίκτυο τι έξοδο πρέπει να παράγει είναι μια διαδικασία που θα αναλυθεί παρακάτω.

2.2.2 Η δομή ενός νευρώνα μεμονωμένα

Παραπάνω είδαμε τη γενική μορφή ενός νευρωνικού δικτύου. Για να ολοκληρωθεί η αρχική εικόνα πρέπει να δούμε πως λειτουργεί ο ίδιος ο νευρώνας. Πώς παράγει την έξοδο του από την είσοδο. Όπως φαίνεται και στο Σχήμα 2.2 οι διαδικασίες που λαμβάνουν χώρα είναι ιδιαίτερα απλές. Αρχικά υπολογίζεται το σταθμισμένο άθροισμα των εισόδων με βάση κάποια βάρη και προστίθεται μια τιμή που λέγεται και προκατάληψη. Έπειτα αυτό το αποτέλεσμα διοχετεύεται σε μια συνάρτηση ενεργοποίησης η οποία προσθέτει τη λεγόμενη μη γραμμικότητα στο δίκτυο. Η τελική τιμή της συνάρτησης ενεργοποίησης είναι η έξοδος του νευρώνα. Όσο πιο μεγάλη είναι αυτή η τιμή τόσο πιο δυνατά θεωρούμε ότι ”φωτίζεται” ο συγκεκριμένος νευρώνας . Τα βάρη είναι διαφορετικά για κάθε νευρώνα για την γενική περίπτωση εκτός αν έχουμε μοίρασμα παραμέτρων όπως συμβαίνει στα συνελικτικά νευρωνικά δίκτυα που θα δούμε παρακάτω. Αυτά τα βάρη αποτελούν τις παραμέτρους που θα λάβουν τιμές με τη διαδικασία της εκπαίδευσης και είναι όλη η ουσία του δικτύου.

Αυτή είναι περίπου η λογική σε κάθε δίκτυο που θα δούμε παρακάτω. Αυτό που αλλάζει στην αρχιτεκτονική συνήθως είναι το πλήθος των στρωμάτων και το είδος των συνδέσεων. .

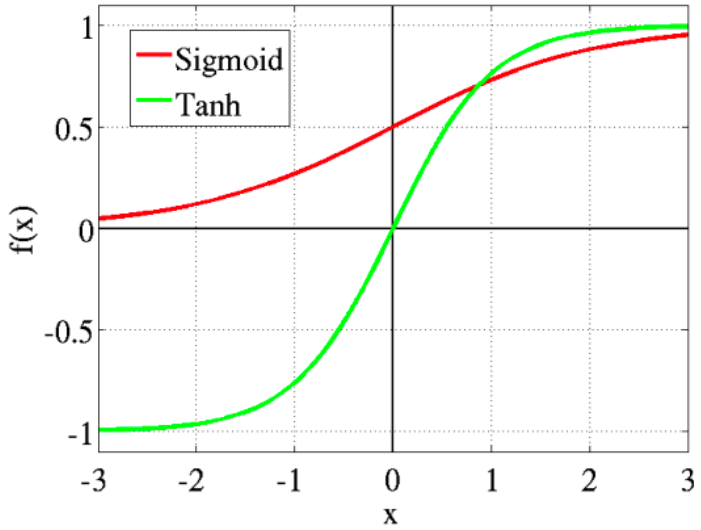


Σχήμα 2.2: Ένας νευρώνας

2.2.3 Οι διάφορες συναρτήσεις ενεργοποίησης των νευρώνων

Αξίζει να αναφερθούμε σύντομα σε μερικές γνωστές και ευρέως χρησιμοποιούμενες συναρτήσεις ενεργοποίησης του νευρώνα. Όπως αναφέρθηκε η σημασία αυτών των συναρτήσεων είναι ιδιαίτερη. Συγκεκριμένα χωρίς αυτές όλο το δίκτυο όσα στρώματα και αν περιέχει θα αναπαριστά μια γραμμική συνάρτηση. Στις περισσότερες περιπτώσεις τα δεδομένα και οι σχέσεις που τα διέπουν δεν μπορούν να περιγραφούν από γραμμικές συναρτήσεις. Επίσης ένα άλλο σημαντικό χαρακτηριστικό αυτών των συναρτήσεων για την διαδικασία της εκπαίδευσης είναι ότι πρέπει να είναι διαφορίσιμες.

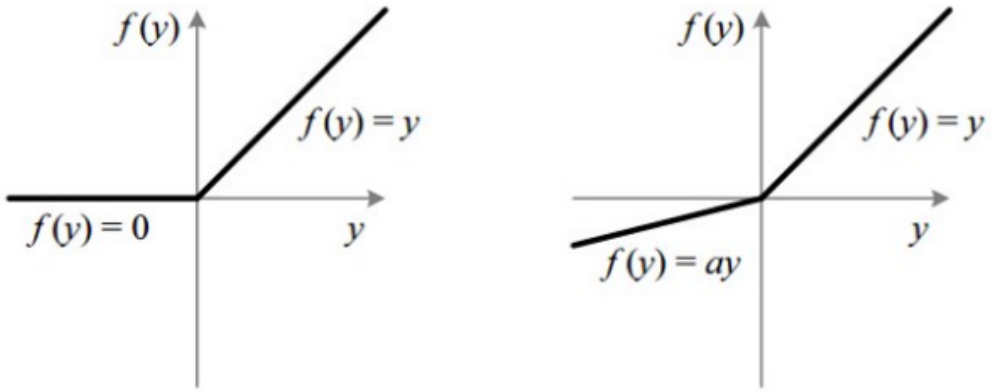
Αρχικά χρησιμοποιούνταν συναρτήσεις όπως η σιγμοειδή και η υπερβολική εφαπτομένη οι οποίες περιόριζαν τη έξοδο σε τιμές στο [-1,1] και [0,1] αντίστοιχα (Σχήμα 2.3). Οι συναρτήσεις αυτές αν και διαφορίσιμες έχουν ένα σοβαρό μειονέκτημα. Οι παράγωγοι τους έχουν επίσης τιμές στα ίδια διαστήματα και μάλιστα τείνουν γρήγορα στο μηδέν για πολύ μεγάλες και πολύ μικρές τιμές.



Σχήμα 2.3: Σιγμοειδή και υπερβολική εφαπτομένη

Το πρόβλημα που προκύπτει εδώ είναι αυτό των εξαφανιζομένων παραγώγων στο οποίο θα αναφερθούμε περισσότερο στην ενότητα για τον τρόπο εκπαίδευσης ενός τέτοιου δικτύου. Συνοπτικά επειδή κατά την εκπαίδευση ,οι παράγωγοι αυτών των συναρτήσεων πολλαπλασιάζονται μεταξύ τους καθώς πηγαίνουμε προς τα πίσω στα στρώματα του δικτύου, στην προσπάθεια μας να ανανεώσουμε τα βάρη, το γινόμενο τους τείνει να γίνεται πολύ μικρό. Πρακτικά μηδενίζεται αφού και η αναπαράσταση των αριθμών στον υπολογιστή μας έχει πεπερασμένη ακρίβεια. Έτσι τείνει να μηδενίζεται και το μέγεθος κατά το οποίο ανανεώνουμε τις παραμέτρους των αρχικών στρωμάτων κάθε φορά που περνάμε ένα στάδιο της εκπαίδευσης.

Αυτό το πρόβλημα λύθηκε μερικώς με τη χρήση μιας πολύ διαδεδομένης συνάρτησης ενεργοποίησης ,της λεγόμενης Rectified Linear Unit (ReLU) και την παραλλαγή της , την leaky ReLU που παρουσιάζονται στο Σχήμα 2.4. Οι συναρτήσεις αυτές είναι γραμμικές στα θετικά και στα αρνητικά με σταθερή παράγωγο αλλά μη γραμμικές στο σύνολο των πραγματικών. Πολλά περαιτέρω μπορούν να ειπωθούν για συναρτήσεις ενεργοποίησης αλλά τα παραπάνω είναι αρκετά για τον σκοπό αυτής της εργασίας. Τέλος λύση στο πρόβλημα των εξαφανιζόμενων παραγώγω δίνει και η χρήση residual networks τα οποία είναι δίκτυα που έχουν απευθείας συνδέσεις μη διαδοχικών στρωμάτων και που θα αναφερθούν και στην ενότητα των μοντέλων που θα χρησιμοποιήσουμε.



Σχήμα 2.4: ReLU και leaky ReLu

2.2.4 Ταξινόμηση σε κλάσεις και η συνάρτηση Softmax

Μέχρι στιγμής περιγράψαμε την αρχιτεκτονική ενός κοινού νευρωνικού δικτύου χωρίς να αναφερθούμε στο πως και γιατί το χρησιμοποιούμε. Μια πολύ διαδεδομένη χρήση τους που θα χρησιμοποιήσουμε ακόμα και για την ανίχνευση αντικειμένων είναι αυτή της ταξινόμησης της εισόδου σε μια από διάφορες προδιαγεγραμμένες κλάσεις. Σε αυτήν την περίπτωση η έξοδος του δικτύου δηλαδή το τελευταίο στρώμα συνηθίζεται να αποτελείται από N νευρώνες όπου N είναι το πλήθος των κλάσεων του προβλήματος μας. Αυτοί οι νευρώνες συνδέονται πλήρως με το προηγούμενο στρώμα αλλά για την έξοδο τους δεν χρησιμοποιούν κάποια συνάρτηση ενεργοποίησης που φράζει τις τιμές όπως είδαμε. Μπορούμε να πούμε ότι παίρνουμε την καθαρή (raw) έξοδο τους. Βέβαια η καθαρή έξοδος από μόνη της δεν είναι και η πιο κατάλληλη μετρική για το πόσο κοντά στην σωστή πρόβλεψη φτάσαμε. Ιδανικά θέλουμε ο νευρώνας της σωστής κλάσης να έχει μεγάλη τιμή ενώ οι υπόλοιποι να έχουν μικρή ή μηδενική. Για να δούμε την τελική απόδοση συνηθίζεται να μετατρέπουμε τις εξόδους αυτές σε πιθανότητες μέσω της συνάρτησης Softmax. Η συνάρτηση αυτή μοιάζει με συνάρτηση ενεργοποίησης μόνο που γίνεται κάποια κλιμάκωση που χρησιμοποιεί όλες τις εξόδους. Ο τύπος της είναι ο παρακάτω.

$$S(y_i) = \frac{e^{y_i}}{\sum_{1 \leq j \leq N} e^{y_j}}$$

Με y_i συμβολίζουμε την καθαρή έξοδο του i-οστού νευρώνα ενώ με $S(y_i)$ την εξόδο της συνάρτησης Softmax που θα αντιπροσωπεύει πιθανότητα όπως φαίνεται και από την κλιμάκωση που κάνουμε με τον παρονομαστή. Υπάρχει αρκετή θεωρία πίσω από την χρήση εκθετικής συνάρτησης και δεν θα αναφερθούμε παραπάνω στο πλαίσιο αυτής της ενότητας. Συνήθως απλά θα λέμε ότι περνάμε την έξοδο από ένα Softmax layer το οποίο θα εκτελεί την παραπάνω διαδικασία.

2.2.5 Η διαδικασία εκπαίδευσης του δικτύου

Σε αυτό το σημείο θα αναφερθούμε συνοπτικά στον τρόπο που εκπαιδεύεται ένα νευρωνικό δίκτυο. Με τον όρο εκπαίδευση του δικτύου εννοούμε την εκτέλεση κάποιου επαναληπτικού αλγορίθμου με σκοπό την επίτευξη μιας καλής παραμετροποίησης για το δίκτυο. Οι παράμετροι του δικτύου είναι τα βάρη των συνδέσεων μεταξύ των νευρώνων καθώς και οι προκαταλήψεις. Καλή παραμετροποίηση θα θεωρούμε ότι πετύχαμε εάν στα δεδομένα αξιολόγησης για τη δοσμένη αυτή παραμετροποίηση πετυχαίνουμε χαμηλό σφάλμα χρησιμοποιώντας κάποια από τις μετρικές απόδοσης που θα εξηγήσουμε καλύτερα παρακάτω. Οι έννοιες που είναι απαραίτητες για την διαδικασία της εκπαίδευσης θα εξηγηθούν στο υπόλοιπο αυτού του κεφαλαίου αρχίζωντας από τις συναρτήσεις σφάλματος.

2.2.6 Συναρτήσεις σφάλματος

Αρχικά πρέπει να ορίσουμε έναν τρόπο με τον οποίο το δίκτυο θα μπορεί συγκρίνει την έξοδο του με την επιθυμητή. Αυτό επιτυγχάνεται με την συνάρτηση σφάλματος. Η συνάρτηση αυτή ορίζει ένα μέτρο για το πόσο μακριά είναι η έξοδος του δικτύου σε σχέση με αυτήν που θα έπρεπε να είχε για την συγκεκριμένη είσοδο. Έτσι μπορούμε να επιδιώξουμε να αλλάξουμε τις παραμέτρους του δικτύου με απώτερο σκοπό την ελαχιστοποίηση αυτής της συνάρτησης σφάλματος.

Υπάρχουν διάφορες τέτοιες συναρτήσεις και η επιλογή της κατάλληλης για το πρόβλημα είναι ζήτημα υψίστης σημασίας καθώς επηρεάζει όλη την διαδικασία της εκπαίδευσης. Γενικά η διαδικασία εκπαίδευσης του δικτύου είναι η εξής.
-Χωρίζουμε τα δεδομένα μας σε δεδομένα εκπαίδευσης και δεδομένα τελικής αξιολόγησης.
-Δεδομένης μιας συνάρτησής σφάλματος ακολουθούμε έναν αλγόριθμο για την αλλαγή των παραμέτρων του δικτύου, μέχρι για τα δεδομένα εκπαίδευσης να παίρνουμε μικρές τιμές σφάλματος. Ιδανικά οι ίδιες παράμετροι θα πρέπει να δίνουν μικρό σφάλμα και πέρα από τα δεδομένα εκπαίδευσης εκτός αν έχουμε το φαινόμενο της υπερπροσαρμογής που θα εξηγηθεί παρακάτω. Μερικές συναρτήσεις σφάλματος είναι οι εξής.

Μέσο τετραγωνικό σφάλμα Αποτελεί έναν απλό τρόπο για τον υπολογισμό σφαλμάτων που χρησιμοποιείται συχνά και πέρα από το πεδίο των νευρωνικών δικτύων. Ουσιαστικά υπολογίζουμε το σφάλμα παίρνοντας την διαφορά της τιμής κάθε νευρώνα της εξόδου με την επιθυμητή ανάλογα προφανώς την κωδικοποίηση του προβλήματος , και υψώνουμε στο τετράγωνο.

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

Οπού n είναι το πλήθος των εξωτερικών νευρώνων , Y_i είναι η έξοδος του i-οστού και Y_i^* η έξοδος που θα επιθυμούσαμε να είχε .

Σφάλμα διασταυρωμένης εντροπίας (Cross-entropy loss) Μια πολύ διαδεδομένη συνάρτηση σφάλματος είναι και η cross-entropy .Για την χρήση αυτής της συνάρτησης το αποτέλεσμα των νευρώνων εξόδου πρέπει να είναι πιθανότητα. Μπορούμε να μετατρέψουμε σε πιθανότητα την έξοδο μέσα από την προσθήκη ενός έξτρα στρώματος Softmax.

Δεδομένου ότι η έξοδοι του τελευταίου στρώματος ακολουθούν τις προδιαγραφές για να είναι πιθανότητες ,δηλαδή τιμές από το 0 μέχρι 1 και άθροισμα ίσο με την μονάδα, τότε για n νευρώνες εξόδου το cross-entropy loss υπολογίζεται ως.

$$Cross - entropy - loss = \sum_{i=1}^{n} (-t_i * log(s_i))$$

Όπου n είναι το πλήθος των νευρώνων εξόδου, t_i είναι η πιθανότητα της πραγματικότητας ground truth , και τέλος s_i η πιθανότητα όπως υπολογίζεται απο το ίδιο το δίκτυο στην έξοδο του νευρώνα.

Γενικά υπάρχουν διάφορες συναρτήσεις και η χρησιμότητα τους εξαρτάται άμεσα από το πρόβλημα που θέλουμε να λύσουμε. Για αυτό και δεν θα αναφερθούμε σε περισσότερες ενώ αν χρειαστεί θα παρουσιαστούν οι συναρτήσεις ανά μοντέλο στην αντίστοιχη ενότητα τους.

2.2.7 Ο αλγόριθμος ανάστροφης διάδοσης

Αφού έχουμε ορίσει την συνάρτηση σφάλματος είμαστε σε θέση να δούμε πως μπορούμε να την χρησιμοποιήσουμε για να μεταβάλλουμε τις παραμέτρους του δικτύου ,δηλαδή τα βάρη των συνδέσεων και τις προκαταλήψεις , ώστε να ελαχιστοποιήσουμε το σφάλμα αυτό. Η ουσία του αλγορίθμου ανάστροφης διάδοσης είναι ο λεγόμενος κανόνας της αλυσίδας στον υπολογισμό παραγώγου.

Ο υπολογισμός της εξόδου του δικτύου μπορεί να θεωρηθεί ως μια σειρά από εφαρμογή διαδοχικών πράξεων στην είσοδο. Ένας τυχαίος εσωτερικός νευρώνας παίρνει τις εξόδους του προηγουμένου στρώματος τις αθροίζει σταθμισμένες κατά τα βάρη και εφαρμόζει τη συνάρτηση ενεργοποίησης για να υπολογίσει την έξοδο του. Δηλαδή έχουμε μια σειρά εφαρμογών από πράξεις που μπορούν να αναπαρασταθούν με κεφαλαία γράμματα όπως οι παρακάτω.

$$f(x) = A(B(C(D(E(x)))))$$

Στο παραπάνω θα μπορούσε για παράδειγμα $E(x) = p1 * x + p2 * x^2 + p3$ με $p1, p2, p3$ παράμετροι του τριωνύμου.

Μπορούμε τώρα να φιξάρουμε την είσοδο και να δούμε πως αλλάζει η συνάρτηση f σε σχέση με την μεταβολή κάποιας παραμέτρου της.

$$f(p) = A(B(C(p)))$$

όπου f θα είναι στην περίπτωση μας η συνάρτησης σφάλματος και p μια παράμετρος του δικτύου η οποία θα παίζει τώρα το ρόλο της μεταβλητής. Επισημαίνεται ότι ο απώτερος σκοπός είναι η μεταβολή όλων των παραμέτρων όπως βάρη κτλπ ώστε να μικρύνει η f. Με το σύμβολο ΄ θα εννοούμε παραγώγιση ως προς τη μεταβλητή μέσα στην παρένθεση. Δηλαδή $F(g)'$ είναι η παράγωγος της F ως προς g. Για να βρούμε την την παράγωγο της f ως προς p αρκεί να υπολογίσουμε το εξής.

$$f^{'}(p) = f^{'}(A) * A^{'}(B) * B^{'}(C) * C^{'}(p)$$

όπου C είναι η εφαρμογή της αντίστοιχης πράξης όπου εμφανίζεται αυτή η παράμετρος. Εντελώς όμοια αν θέλουμε να υπολογίσουμε την παράγωγο της f ως προς κάποια άλλης παραμέτρου, ας την πούμε p΄ ,που εμφανίζεται στην εφαρμογή B, αρκεί να φτάσουμε τον κανόνα μέχρι εκείνο το σημείο.

$$f^{'}(p') = f^{'}(A) * A^{'}(B) * B^{'}(p')$$

Αυτή η τεχνική μας επιτρέπει να υπολογίσουμε την παράγωγο της συνάρτησης σφάλματος ως προς οποιαδήποτε παράμετρο του δικτύου.

Για παράδειγμα πάνω σε ένα νευρώνα του δικτύου ,αν C ήταν η εφαρμογή του υπολογισμού του αθροίσματος των εισόδων σταθμισμένων επί τα βάρη σε κάποιο νευρώνα και B ήταν η εφαρμογή της πρόσθεση της προκατάληψης bias στο πάνω άθροισμα

$$C = \mathbf{w} * \mathbf{x}$$

όπου με x είναι το διάνυσμα εισόδου επί τα βάρη

$$B = a * C + b$$

όπου με b είναι η προκατάληψη. Τότε η παράγωγος ως προς b (bias) της συνάρτησης σφάλματος μπορεί να υπολογιστεί ως εξής

$$f^{'}(b) = f^{'}(A) * A^{'}(B) * B^{'}(b)$$

με

$$B^{'}(b) = (a * C + b)^{'} = 1$$

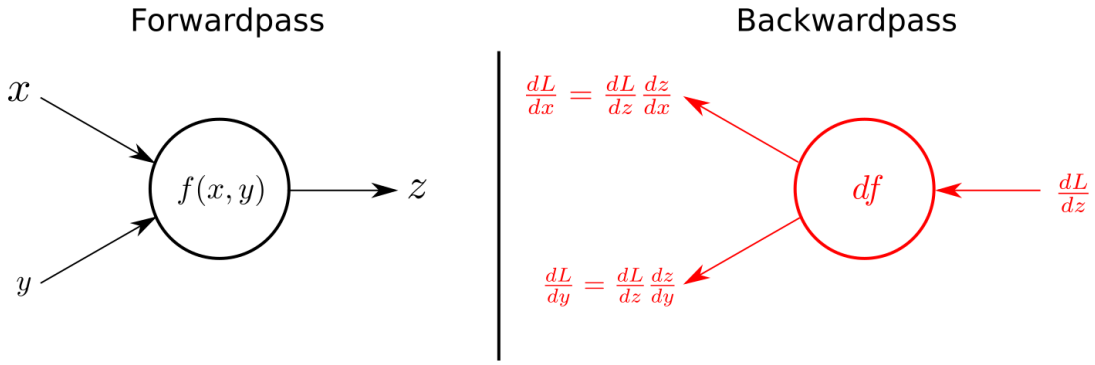
έχουμε

$$f^{'}(b) = f^{'}(A) * A^{'}(B) * 1$$

όπου και το μέγεθος

$$f^{'}(A) * A^{'}(B)$$

υπολογίζεται εύκολα. Για παράδειγμα Α΄(Β) είναι η παράγωγος της εφαρμογής της πράξης Α ως προς τη θεωρούμενη ως μεταβλητή τώρα Β. Γενικά επειδή αυτά τα αποτελέσματα υπολογίζονται ξανά και ξανά μπορούμε να τα αποθηκεύουμε κάνοντας χρήση μιας τεχνικής που ονομάζεται memoization κερδίζοντας υπολογιστικό κόστος.



Σχήμα 2.5: αντίστροφη διάδοση

Πρακτικά για μας το παραπάνω σημαίνει ότι έχουμε μια μέθοδο να βρίσκουμε την μεταβολή της συνάρτησης λάθους ως προς οποιαδήποτε παράμετρο του δικτύου. Αυτό είναι ιδιαίτερα σημαντικό αφού τώρα είμαστε έτοιμοι να εφαρμόσουμε τον αλγόριθμο απότομης καθόδου για να εκπαιδεύσουμε το δίκτυο. Τέλος φαίνεται γιατί θέλαμε τη διαφορισιμότητα στην επιλογή συναρτήσεων ενεργοποίησης. Οποιαδήποτε παράμετρος βρίσκεται ”νωρίτερα” δηλαδή πίσω από μια συνάρτηση ενεργοποίησης (και υπάρχει μονοπάτι προς αυτή) τότε χρειάζεται τη παράγωγο εκείνης της συνάρτησης ενεργοποίησης για να υπολογίσει την δικιά της μερική παράγωγο του σφάλματος.

2.2.8 Ο αλγόριθμος απότομης καθόδου (Gradient Descent)

Το σύνολο των n παραμέτρων ενός δικτύου αποτελεί ένα χώρο n διαστάσεων .Για κάθε σετ από παραμέτρους έχουμε λοιπόν και ένα σφάλμα που υπολογίζεται από όλα τα δεδομένα αθροιστικά και μέσω της συνάρτησης σφάλματος. Αυτό σημαίνει ότι έχουμε μια συνάρτηση n μεταβλητών εισόδου και μιας εξόδου η οποία είναι παραγωγίσιμη και έχουμε τρόπο να υπολογίσουμε τις μερικές παραγώγους. Άρα έχουμε τρόπο να υπολογίσουμε την κλίση(gradient). Η κλίση της συνάρτησης αυτής, υπολογισμένη για ένα σημείο του χώρου δηλαδή για ένα σύνολο τιμών των παραμέτρων του δικτύου, θα μας δίνει τότε την κατεύθυνση μέγιστης αύξησης του συνάρτησης - σφάλματος . Αυτή είναι μια ιδιότητα της κλίσης οποιασδήποτε διαφορίσιμης βαθμωτής συνάρτησης πολλών μεταβλητών. Αντίστοιχα η αντίθετη κατεύθυνση από την κλίση θα μας δίνει την κατεύθυνση μέγιστης πτώσης της συνάρτησης. Με άλλα λόγια προς τα που πρέπει να κινηθούμε στον χώρο των παραμέτρων για να έχουμε την μεγαλύτερη πτώση δεδομένου της απόστασης που θα διανύσουμε.

Μετά την παραπάνω ανάλυση προκύπτει ένας πολύ απλός επαναληπτικός αλγόριθμος βελτιστοποίησης. Πρακτικά μπορούμε αρχίζοντας από ένα τυχαίο σημείο στον χώρο παραμέτρων να υπολογίζουμε κάθε φορά την κλίση και να κάνουμε ένα μικρό βήμα προς την αντίθετη κατεύθυνση. Κάθε φορά μειώνουμε την τιμή της συνάρτησης σφάλματος οπότε και βρισκόμαστε όλο και πιο κοντά στον αρχικό στόχο μας, να πετύχουμε μικρό σφάλμα στα δεδομένα εκπαίδευσης γενικεύωντας καλά σε όλα τα δεδομένα.

Βέβαια εδώ τίθεται το ερώτημα του κατά πόσο μεγάλο βήμα θα μετακινηθούμε στο χώρο ανά υπολογισμό της κλίσης. Αυτό μας το δίνει ο ρυθμός εκμάθησης l (θετικός πραγματικός αριθμός) με τον οποίο πολλαπλασιάζουμε το διάνυσμα της κλίσης πριν το προσθέσουμε στις συντεταγμένες του τωρινού σημείου για να βρούμε το νέο σημείο στο χώρο παραμέτρων. Συγκεκριμένα παίρνουμε τις νέες τιμές του διανύσματος παραμέτρων ως εξής.

$$\mathbf{p}' = \mathbf{p} - l * \frac{\Delta f(\mathbf{p})}{\Delta \mathbf{p}}$$

με f ως την συνάρτηση σφάλματος της οποίας υπολογίζουμε την κλίση ως προς το διάνυσμα p. Το άρθρο [Rude16] περιέχει μια αρκετά επεξηγηματική ανάλυση του αλγορίθμου της απότομης καθόδου και των παραλλαγών του.

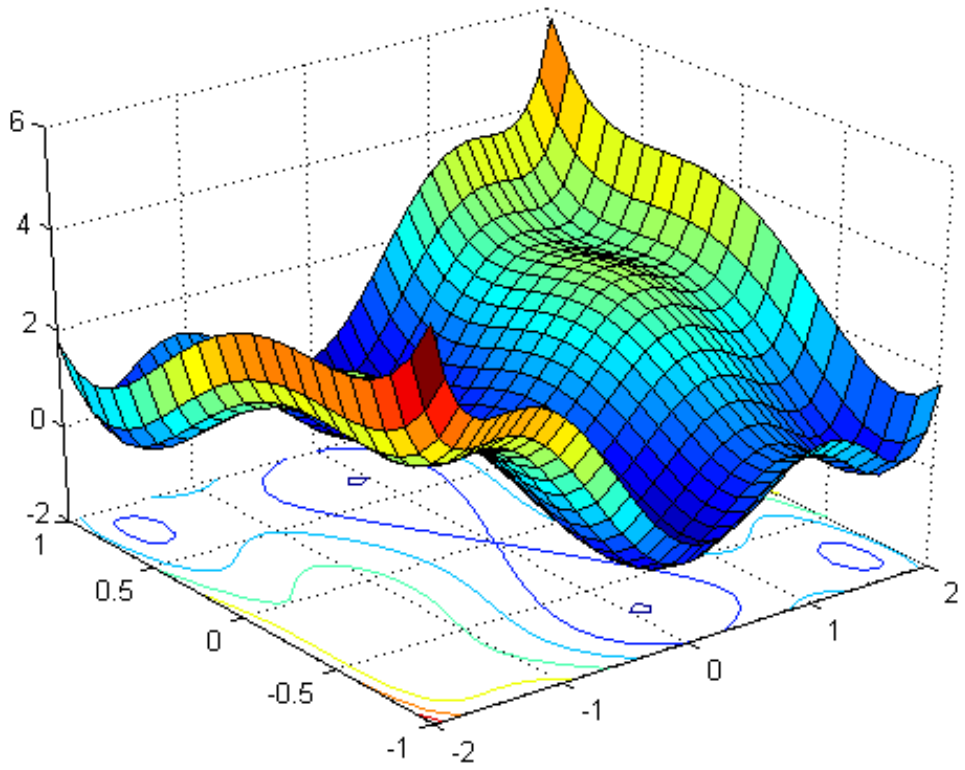
Ένα πρόβλημα που παρουσιάζεται στη παραπάνω μέθοδο και που αγνοήσαμε μέχρι τώρα είναι ότι η κλίση υπολογίζεται ως προς ολόκληρο το σύνολο των δεδομένων εκπαίδευσης. Κάθε βήμα που παίρνουμε είναι προς τη βελτιστοποίηση του σφάλματος προς όλο το σύνολο. Αυτό είναι υπολογιστικά ασύμφορο μιας και μιλάμε για σύνολα δεδομένων που αποτελούνται από χιλιάδες στοιχεία. Ένας τρόπος είναι να προσαρμόζουμε τις παραμέτρους του δικτύου ανά ξεχωριστό παράδειγμα. Αυτή η παραλλαγή ονομάζεται ως ο αλγόριθμος της στοχαστικής απότομης καθόδου.

Άλλο ένα πρόβλημα είναι τα λεγόμενα τοπικά ελάχιστα. Έστω ότι στον χώρο των παραμέτρων ακολουθώντας την παραπάνω διαδικασία καταλήγουμε σε ένα από αυτά. Τότε εγκλωβιζόμαστε εκεί και χάνουμε τη δυνατότητα να προσεγγίσουμε το ολικό ελάχιστο ή τουλάχιστον κάποιο άλλο τοπικό ελάχιστο με λιγότερη τιμή σφάλματος όπως φαίνεται στο Σχήμα 2.6. Για αυτό πολλές φορές χρησιμοποιούνται οι παρακάτω παραλλαγές.

Stohastic Gradient Descent Διαλέγουμε τυχαία ένα στοιχείο και προσαρμόζουμε τις παραμέτρους υπολογίζοντας την κλίση της συνάρτησης σφάλματος για το μεμονωμένο αυτό στοιχείο. Είναι αρκετά πιο γρήγορη μέθοδος από την κλασσική μέθοδο αλλά έχει διάφορα ελαττώματα. Ένα από αυτά είναι ο θόρυβος που εισάγουν ακραία δεδομένα. Είναι λογικό να υπάρχουν τέτοια δεδομένα στο σύνολο εκπαίδευσης μας και ο υπολογισμός της κλίσης ως προς αυτά θα δίνει λαθεμένη κατεύθυνση. Βέβαια ο επιπλέον θόρυβος ,βοηθάει εν μέρη στο να μην κολλήσουμε σε τοπικά ελάχιστα

Batch Gradient Descent Μια μέση λύση ανάμεσα στην στοχαστική εκδοχή και τον κλασσικό αλγόριθμο. Υπολογίζουμε την κλίση και προσαρμόζουμε τα βάρη ανά ένα υποσύνολο (batch) του συνόλου δεδομένων. Τα batches πρεπεί είναι διακριτά υποσύνολα και να απαρτούν όλα τα δεδομένα εκπαίδευσης. Επίσης συνηθίζεται για τεχνικούς λόγους το μέγεθος τους να είναι δύναμη του δύο. Το μέγεθος του τμήματος επίσης ποικίλλει ανάλογα το πρόβλημα και την επεξεργαστική ισχύ μας. Για παράδειγμα στα μοντέλα που θα χρησιμοποιηθούν παρακάτω η χρήση μεγάλου μεγέθους batch ήταν αδύνατη λόγω χαμηλής μνήμης του υπολογιστή που τρέχει την εκπαίδευση.

Τέλος αξίζει να αναφερθεί εδώ ότι η μέθοδος της απότομης καθόδου κατά την κλίση δεν χρησιμοποιείται σχεδόν ποτέ αυτούσια στις μέρες μας. Υπάρχουν πολλές παραλλαγές που εισάγουν έννοιες όπως ορμή και είναι αρκετά πιο αποδοτικές στο να εξαλείφουν τα προβλήματα που συναντάμε. Συγκεκριμένα τα μοντέλα ανίχνευσης τρέχουν τον αλγόριθμο βελτιστοποίησης Adam που επίσης χρησιμοποιεί την ορμή καθώς και άλλες βελτιώσεις. Στα πλαίσια αυτής της εργασίας δεν θα αναφερθούμε σε περισσότερες λεπτομέρειες μιας και μας αρκεί να εστιάσουμε στην αρχιτεκτονική των μοντέλων που θα χρησιμοποιήσουμε.



Σχήμα 2.6: Τα πολλαπλά ελάχιστα στη συνάρτηση σφάλματος- Ο χώρος για 2 παραμέτρους

2.2.9 Μετρικές απόδοσης για επιβλεπόμενη μάθηση

Σε αυτή την ενότητα και έχοντας μιλήσει συνοπτικά για τον τρόπο εκπαίδευσης του δικτύου θα δούμε κάποιες από τις πιο γνωστές μετρικές απόδοσης, δηλαδή τρόπους να μετρήσουμε την επίδοση του δικτύου μας στα δεδομένα αξιολόγησης. Θα δούμε συγκεκριμένα μετρικές για το πρόβλημα της ταξινόμησης σε κλάσεις και αργότερα θα τις επεκτείνουμε όσο χρειάζεται για να καλύψουμε το πρόβλημα της ανίχνευσης αντικειμένων.

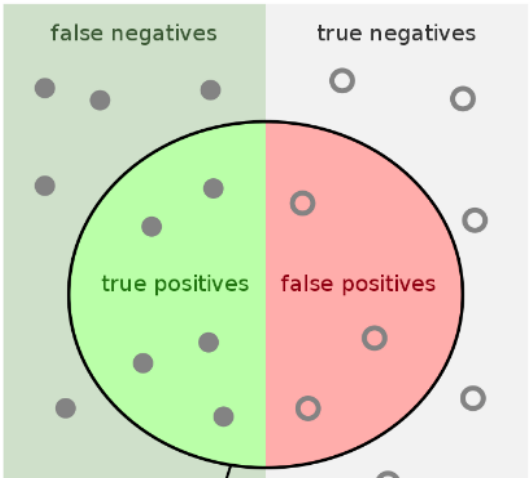
Για κάθε κλάση μπορούμε να ορίσουμε τις παρακάτω βασικές έννοιες για την μέτρηση της απόδοσης. Οι έννοιες αυτές ορίζονται πάντα ως προς μια συγκεκριμένη κλάση αναφοράς

True Positive Με True Positive ονομάζουμε τις προβλέψεις του δικτύου μας που ανήκουν στην κλάση αναφοράς και είναι σωστές. Δηλαδή η ground truth πραγματικότητα που έχει αντιστοιχηθεί με τις εισόδους που δίνουν αυτές τις προβλέψεις μας λέει ότι όντως ανήκουν σε αυτή τη κλάση.

True Negative Αντίστοιχα με True Negative ονομάζουμε τις προβλέψεις που δεν ανήκουν στην συγκεκριμένη κλάση και που όντως βάση του ground truth τους αυτό είναι ορθό.

False Positive Με Flase Positive ονομάζουμε τις προβλέψεις που το δίκτυο μας έχει αντιστοιχήσει στη συγκεκριμένη κλάση αλλά αυτό είναι λάθος σύμφωνα με την αλήθεια.

False Negative Τέλος με False Negative ονομάζουμε τις προβλέψεις που έχουν αντιστοιχηθεί σε άλλη κλάση ενώ το ορθό θα ήταν να είχαν αντιστοιχηθεί στη συγκεκριμένη κλάση αναφοράς.
Τα παραπάνω συνηθίζεται να τα οπτικοποιούμε με ένα διάγραμμα όπως του Σχήματος 2.7.



Σχήμα 2.7: Διάγραμμα για TF,TN,FP,FN

Με τις παραπάνω έννοιες μπορούμε να ορίσουμε τώρα διάφορες μετρικές.

Precision και Recall

Με τον όρο ακρίβεια (Precision) μετράμε την ικανότητα του μοντέλου να είναι ακριβές στην πρόβλεψη των θετικών παραδειγμάτων. Πρακτικά μας λέει ποσά από τα στοιχεία που το μοντέλο βρήκε ότι ανήκουν στην κλάση αναφοράς όντως ανήκουν σε αυτή.

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive} = \frac{TruePositive}{TotalPredictedPositive}$$

Η μετρική Precision είναι καλή όταν το κόστος των False Positive είναι υψηλό. Για παράδειγμα αν το μοντέλο εκτελεί ταξινόμηση μηνυμάτων ηλεκτρονικού ταχυδρομίου σε κανονικά ή spam τότε False Positive σημαίνει ότι θα μάρκαρε ένα κανονικό μήνυμα ως spam. Ο χρήστης τότε θα μπορούσε να αγνοήσει το μήνυμα και να χάσει σημαντική πληροφορία. Σε αυτή την περίπτωση δεν μας νοιάζει τόσο να έχουμε False Negative αφού είναι λιγότερο επιβλαβές να αφήσουμε spam μηνύματα να περάσουν το φιλτράρισμα από το αντίθετο.

Με τον όρο ανάκληση (Recall) μετράμε την ικανότητα του μοντέλου να μπορεί να βρει όλα τα θετικά παραδείγματα δηλαδή όλα τα στοιχεία της κλάσης αναφοράς. Ο τύπος υπολογισμού είναι ο παρακάτω.

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative} = \frac{TruePositive}{TotalActualPositive}$$

Η μετρική αυτή είναι καλή όταν το κόστος για False Negatives είναι υψήλο. Ένα τέτοιο παράδειγμα θα ήταν κάποια ιατρική εφαρμογή για την αναγνώριση κάποιας σοβαρής ασθένειας. Τότε το κόστος να μην αναγνωρίσουμε ότι κάποιος ασθενής όντως πάσχει από αυτήν είναι αρκετά υψηλό για την υγεία του ασθενή αλλά και του περίγυρου του αν η ασθένεια είναι μολυσματική.

Οι δύο αυτές μετρικές δεν αποτελούν από μόνες τους πάντα καλό κριτήριο απόδοσης. Πολλές φορές χρησιμοποιείται η ακρίβεια (όχι τόσο στις μέρες μας).

$$Accuracy = \frac{TruePositive + TrueNegative}{TruePositive + TrueNegative + FalsePositive + FalseNegative}$$

ή

$$Accuracy = \frac{TotalCorrectPredictions}{TotalPredictions}$$

Ο δεύτερος τύπος μπορεί να εφαρμοστεί υπολογίζωντας τις σωστές προβλέψεις για όλες τις κλάσεις δίνοντας την γενική ακρίβεια του μοντέλου. Αποτυγχάνει όμως όταν οι κλάσεις δεν εμφανίζονται με την ίδια συχνότητα στα δεδομένα μας.

Όταν θέλουμε μια καλή ισορροπία μεταξύ precision,recall επίσης συνήθίζεται να χρησιμοποιούμε την μετρικη F1 η οποία είναι ο παρακάτω συνδυασμός τους.

$$F1 = 2\frac{Precision * Recall}{Preicision + Recall}$$

Average Precision και mean Average Precision

Οι παραπάνω μετρικές δυστυχώς δεν είναι τόσο καλές ειδικά στο πρόβλημα του εντοπισμού αντικειμένων σε εικόνες. Αυτή που χρησιμοποιείται κατά κόρων είναι η mean Average Precision αυτούσια ή με διάφορες παραλλαγές την οποία θα εξηγήσουμε εδώ.

Αρχικά είδαμε ότι το μοντέλο μας προβλέπει δίνοντας μια πιθανότητα ξεχωριστά ανα κλάση που αντιπροσωπεύει πόσο πιθανό είναι να ανήκει η είσοδος σε αυτήν την κλάση. Έτσι λοιπόν αν πάρουμε μια κλάση αναφοράς τότε για κάθε στοιχείο του συνόλου αξιολόγησης της απόδοσης έχουμε και μια πιθανότητα-σκορ που το μοντέλο μας δίνει στο στοιχείο αυτό για την κλάση αυτή. Μπορούμε τώρα να ταξινομήσουμε τα στοιχεία σε φθίνουσα σειρά βάση του σκορ τους. Τα στοιχεία αυτά έχουν και μια ground truth ετικέτα που μας λέει αν όντως ανήκουν στην κλάση αυτή ή όχι.
Average Precision είναι λοιπόν το άθροισμα για κάθε στοιχείο (στο ταξινομημένο σύνολο), του precision όπως υπολογίζεται με την προσθήκη αυτού του στοιχείου επί τη διαφορά του recall που έδωσε η προσθήκη αυτού του στοιχείου.

$$AveragePrecision = \sum_{1 \leq i \leq N} P(i) * \Delta R(i)$$

με

$$\Delta R(i) = R(i) - R(i-1)$$

Τα P,R συμβολίζουν τις αντίστοιχες μετρικές Precision και Recall ενώ R(0)=0. Με N συμβολίζουμε το πλήθος των προβλέψεων και μπορούμε να το πάρουμε για όλο το σύνολο των προβλέψεων ή για τις πρώτες N δηλαδή αυτές από μια πιθανότητα-σκορ και πάνω. Αυτές αντιστοιχούν στις θετικές προβλέψεις.

Για να καταλάβουμε καλύτερα αυτή τη μετρική μπορούμε να τη δούμε μέσα από ένα παράδειγμα. Στον παρακάτω πίνακα για παρουσιάζονται ταξινομημένες 6 προβλέψεις ως προς σκορ-πιθανότητα. Στη στήλη ground truth έχουμε την πραγματικότητα για το αν το στοιχείο ανήκει στην κλάση. Στη στήλη Precision και στη στήλη Recall έχουμε τις μετρικές όπως αυτές υπολογίζονται μέχρι και το στοιχείο αυτό. Τέλος στη στήλη Recall Difference την διαφορά στο Recall που έδωσε αυτό το στοιχείο. Το πλήθος των ground truth positive είναι 3 εξού και ο παρονομαστής στις μετρήσεις της ανάκλησης.

Score-probability	Ground Truth	Precision	Recall	Recall Difference
0.9	True	1/1	1/3	1/3
0.8	False	1/2	1/3	0
0.6	True	2/3	2/3	1/3
0.55	True	3/4	3/3	1/3
0.3	False	3/5	3/3	0
0.2	False	3/6	3/3	0

Πίνακας 2.1: Ένα παράδειγμα υπολογισμού Average Precision

Η AP τότε είναι

$$AvgPrec = \frac{1}{1} * \frac{1}{3} + \frac{1}{2} * 0 + \frac{2}{3} * \frac{1}{3} + \frac{3}{4} * \frac{1}{3} + \frac{3}{5} * 0 + \frac{3}{6} * 0 = 0.81$$

Μπορούμε να παρατηρήσουμε τα εξής για την AP. Η μετρική αυτή τιμωρεί μοντέλα που δεν μπορούν να ταξινομήσουν σωστά το σύνολο ή ισοδύναμα μοντέλα που να μην δίνουν τις μεγαλύτερες πιθανότητες σε θετικά παραδείγματα και τις μικρότερες στα αρνητικά. Επισημαίνεται ότι με θετικά και αρνητικά ονομάζουμε τα στοιχεία σύμφωνα με τη ground truth ετικέτα τους σε αντίθεση με τις θετικές και αρνητικές προβλέψεις με τις οποίες περγράφουμε την ταξινόμηση του μοντέλου.

Στο παραπάνω παράδειγμα το μοντέλο πήγε πολύ καλά μιας και τα θετικά παραδείγματα είναι στην αρχή της ταξινόμησης με εξαίρεση το δεύτερο στοιχείο που είχε πιθανότητα 0.8 να ανήκει στη κλάση ενώ δεν άνηκε. Επίσης μπορούμε να παρατηρήσουμε ότι δεν χρειάζεται να πάρουμε την μέτρηση για όλα τα στοιχεία του συνόλου. Θα μπορούσαμε παραπάνω να σταματούσαμε στο 5ο στοιχείο και να θεωρούσαμε τα πρώτα πέντε ως τις θετικές προβλέψεις.
Ένα καλό μοντέλο για την κλάση δεν επηρεάζεται από τα τελευταία στοιχεία του συνόλου. Όπως φαίνεται από το παράδειγμα αυτά δίνανε μηδενικές τιμές στο άθροισμα αφού το recall είχε γίνει ίσο με ένα, αφού βρήκαμε όλα τα θετικά στοιχεία του συνόλου. Ένα κακό μοντέλο θέλει μεγάλο όριο στο πόσα θα πάρουμε για την μέτρηση το όποιο είναι ισοδύναμο με το να έχουμε χαμηλή πιθανότητα-όριο ως το ποια παραδείγματα θα ταξινομούμε ως θετικά αν όντως θέλουμε να προβλέψουμε ως θετικά τα περισσότερα από τα θετικά δείγματα.

Μπορούμε τώρα να ορίσουμε και την **mean Average Precision**. Αυτή δεν είναι τίποτα άλλο από τον μέσο όρο της average precision για όλες τις κλάσεις του προβλήματος και μας δίνει μια συνολική εικόνα για την απόδοση του μοντέλου.

$$mAP = \frac{1}{N_{classes}} \sum_{1 \leq i \leq N_{classes}} AP(i)$$

Για την αναγνώριση αντικειμένων σε εικόνα και όχι απλή ταξινόμηση τα πράματα είναι ελαφρώς πιο περίπλοκα και χρειαζόμαστε την έννοια της IoU για να δούμε πως υπολογίζεται η Average Precision στο αντίστοιχο πρόβλημα. Θα το δούμε στην αντίστοιχη ενότητα λίγο παρακάτω.

2.2.10 Το πρόβλημα της υπέρ-προσαρμογής

Όπως περιγράφηκε παραπάνω η διαδικασία της εκπαίδευσης είναι ουσιαστικά η εύρεση ενός σετ από παραμέτρους για το δίκτυο οι οποίες δίνουν μικρό σφάλμα πάνω στα δεδομένα εκπαίδευσης. Ο σκοπός μας όμως είναι να φτιάξουμε έναν αλγόριθμο με μικρό σφάλμα σε όλο το σύνολο των στοιχείων μας συμπεριλαμβανομένου και των νέων δεδομένων που θα μας ενδιαφέρει να επεξεργαστούμε. Εδώ εμφανίζεται το πρόβλημα της υπερπροσαρμογής. Στο φαινόμενο αυτό η εκπαίδευση έχει δώσει παραμέτρους που έχουν προσαρμοστεί πλήρως στο σύνολο εκπαίδευσης αλλά δεν είναι ιδανικές για να δώσουν μικρό σφάλμα στο ευρύτερο σύνολο των στοιχείων. Με άλλα λόγια το δίκτυο αδυνατεί να γενικεύσει σε νέα στοιχεία ενώ η λειτουργία του έχει συρρικνωθεί σε αυτή της ”μνήμης” αφού τείνει να θυμάται και να ”παπαγαλίζει” τα δεδομένα εκπαίδευσης!

Πολλή έρευνα γίνεται με σκοπό την δημιουργία τεχνικών για αποφυγή της υπέρ-προσαρμογής. Μια συνήθης τεχνική είναι να χωρίζουμε τα δεδομένα που έχουμε στη διάθεση μας για το training σε δεδομένα εκπαίδευσης και δεδομένα εξακρίβωσης (validation set) .
Τα δεδομένα εξακρίβωσης χρησιμοποιούνται ως εξής. Δεν τα χρησιμοποιούμε για την μεταβολή των παραμέτρων αλλά σε κάθε εποχή μετριέται η απόδοση του δικτύου στο validation set. Αν βρεθεί ότι η απόδοση αυτή πέφτει καθώς το δίκτυο συνεχίζει να αυξάνει την απόδοση του στα δεδομένα εκπαίδευσης τότε έχουμε μια ένδειξη ότι είμαστε σε φάση υπέρ-προσαρμογής οπότε και θα σταματήσουμε την εκπαίδευση ή θα πάρουμε άλλα μέτρα πρόληψης. Μπορούμε επίσης να προσαρμόζουμε τις υπερ παραμέτρους (μη εκπαιδεύσιμες παραμέτρους) του δικτύου μας σύμφωνα με τις ενδείξεις που παίρνουμε από την απόδοση στο σύνολο εξακρίβωσης. Επισημαίνεται ότι δεν μπορούμε να χρησιμοποιήσουμε

Σχήμα 2.8: Η υπέρ-προσαρμογή στα δεδομένα εκπαίδευσης

τα δεδομένα τελικού ελέγχου (test data) για να σταματήσουμε την υπέρ προσαρμογή ή γενικά στην εκπαίδευση με οποιοδήποτε τρόπο. Αν επιχειρούσαμε κάτι τέτοιο τότε τίποτα δεν θα μπορούσε να μας εγγυηθεί ότι δεν θα έχουμε πλήρη προσαρμογή σε όλο το σύνολο δεδομένων μας , και πως όταν χρησιμοποιήσουμε το δίκτυο για ανάλυση νέων δεδομένων ,αυτό δεν θα αποτύχει παταγωδώς λόγω αδυναμίας γενίκευσης.

Άλλη μια τεχνική που χρησιμοποιείται και στα μοντέλα που θα δούμε για ανίχνευση είναι το λεγόμενο **dropout**. Ουσιαστικά κατά τη διάρκεια της εκπαίδευσης κάθε νευρώνας έχει επίσης μια πιθανότητα να είναι ενεργός η οποία είναι γενική παράμετρος του δικτύου. Επειδή μόνο μερικοί νευρώνες είναι ανοικτοί κάθε φορά είναι πιο δύσκολο να υπάρξει απομνημόνευση στο δίκτυο και εμπειρικά φαίνεται ότι οδηγούμαστε σε παραμετροποιήσεις του δικτύου που γενικεύουν καλύτερα.

2.3 Τα συνελικτικά νευρωνικά δίκτυα

Εδώ θα παρουσιαστεί ο βασικός λίθος δόμησης των μοντέλων για ανάλυση εικόνας στην βαθιά μηχανική μάθηση ,το συνελικτικό δίκτυο. Τα δίκτυα αυτά έχουν επινοηθέι εδώ και καιρό με αρχική μελέτη του Yann Lecun [LeCu99]. Τα συνελικτικά δίκτυα είναι δίκτυα φτιαγμένα και προσαρμοσμένα για το πρόβλημα της ανάλυσης εικόνας και για αυτό και πετυχαίνουν τρομερά καλύτερη απόδοση σε αυτές τις εφαρμογές. Συγκεκριμένα λύνουν δύο βασικά προβλήματα που εμφανίζουν τα απλά πλήρως συνδεδεμένα δίκτυα.

Πρώτον όταν χρησιμοποιούμε πλήρως συνδεδεμένα δίκτυα για ανάλυση εικόνων χάνουμε τις χωρικές συσχετίσεις τις εικόνας αφού περνάμε όλη την εικόνα ως ένα μονοδιάστατο διάνυσμα στο δίκτυο. Αυτό γιατί όπως και στην ανθρώπινη όραση δεν μας ενδιαφέρουν τα χαρακτηριστικά από μόνα τους αλλά σε σχέση με την περιοχή στην οποία βρίσκονται. Το γεγονός για παράδειγμα ότι ένα εικονοστοιχείο είναι έντονα μπλε από μόνο του δίνει λιγότερη πληροφορία από το γεγονός ότι είναι μπλε και βρίσκεται σε μια περιοχή από άλλα μπλε εικονοστοιχεία , που δηλώνει έντονα την παρουσία ίσως ουρανού στην εικόνα. Τα λεγόμενα συνελικτικά στρώματα που θα παρουσιάσουμε παρακάτω εφαρμόζουν τεχνικές κυλιόμενου παραθύρου οπότε και σέβονται αυτές τις χωρικές συσχετίσεις.

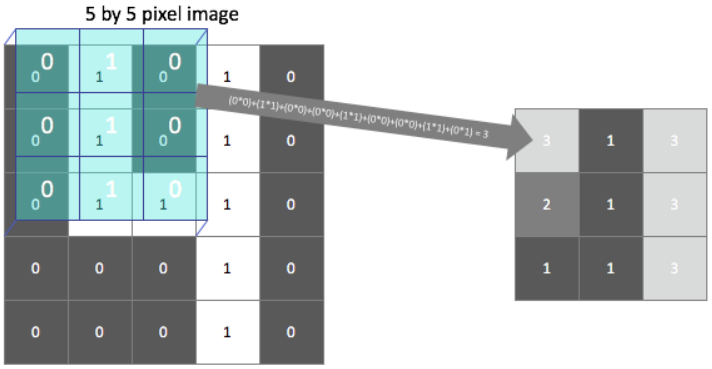
Δεύτερο θέμα είναι ο όγκος των παραμέτρων που για τα πλήρως συνδεδεμένα δίκτυα είναι ασύμφορος και υπολογιστικά και ως προς το πλήθος των δεδομένων εκπαίδευσης που χρειάζονται για να συγκλίνουν σε καλή παραμετροποίηση. Αντίθετα στα συνελικτικά έχουμε μερική σύνδεση κάθε νευρώνα με το προηγούμενο στρώμα. Επιπλέον έχουμε κάτι που ονομάζεται διαμοιρασμός παραμέτρων. Κάθε νευρώνας συνδέεται με το προηγούμενο στρώμα με βάρη τα οποία είναι ολόιδια και για τις συνδέσεις κάθε άλλου νευρώνα του ίδιου στρώματος που αντιστοιχεί στο ίδιο χαρακτηριστικό (βλέπε παρακάτω) αλλά βρίσκεται σε διαφορετικές χωρικές συντεταγμένες πάνω στο στρώμα. Αυτό μειώνει περαιτέρω το πλήθος των παραμέτρων και είναι ιδιαίτερα σημαντικό μιας και θέλουμε το δίκτυο μας να αναγνωρίζει χαρακτηριστικά στην εικόνα σε διάφορες θέσεις που αυτά μπορεί να βρίσκονται. Τα διαφορετικά στρώματα ενός συνελικτικού δικτύου θα αναλυθούν με λεπτομέρεια σε αυτήν την ενότητα.

2.3.1 Το συνελικτικό στρώμα

Για να μελετήσουμε την αρχιτεκτονική αυτού του στρώματος είναι προτιμότερο να ορίσουμε πρώτα μερικές έννοιες και έπειτα να δούμε πως γίνεται ο υπολογισμός της εξόδου ενός νευρώνα ,παρά να πάμε απευθείας στις συνδέσεις του δικτύου. Όπως φανερώνει και το όνομα αυτά τα δίκτυα εφαρμόζουν την πράξη της συνέλιξης πάνω στην εικόνα. Επισημαίνεται ότι η εικόνα έχει διαστάσεις $(height, width, channels)$ όπου channels είναι το πλήθος των καναλιών χρώματος για κάθε εικονοστοιχείο και είναι συνήθως ίσο με 3 , δηλαδή μια τιμή για κάθε ένα από τα τρία βασικά χρώματα (πράσινο,κόκκινο,μπλε).

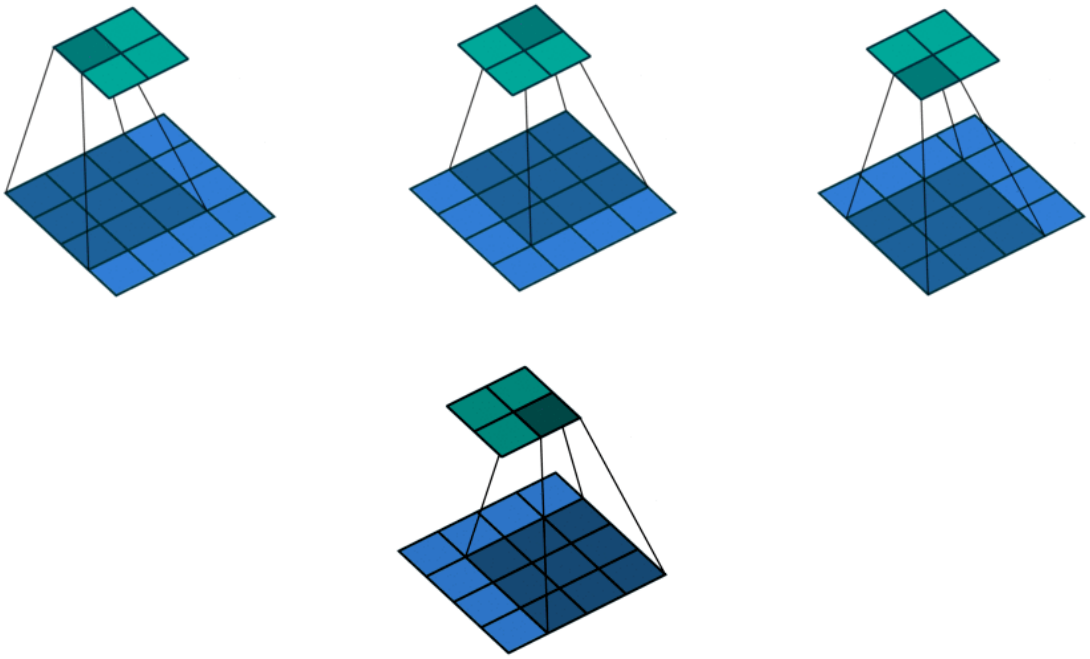
Αρχικά πρέπει να ορίσουμε την έννοια του φίλτρου ενός συνελικτικού στρώματος . Το φίλτρο μπορεί να θεωρηθεί ως ένα παράθυρο με διαστάσεις $h * w * c$. Οι δύο πρώτες διαστάσεις είναι σχετικά αυθαίρετες και συνήθως παίρνουν μικρές τιμές. Αποτελούν τις χωρικές διαστάσεις του φίλτρου. Η τελευταία διάσταση αποτελεί το βάθος του φίλτρου και είναι ίση πάντα με το βάθος του προηγούμενου στρώματος. Αν το προηγούμενο στρώμα είναι τα εικονοστοιχεία της ίδιας της εικόνας τότε το βάθος του φίλτρου θα πρέπει να είναι ίσο με 3 ,όσα και τα κανάλια χρώματος.

Για να υπολογίσουμε το συνελικτικό στρώμα παίρνουμε το παράθυρο και το ολισθαίνουμε πάνω στην εικόνα (ή προηγούμενο στρώμα).Σε κάθε θέση που βρίσκεται το παράθυρο πολλαπλασιάζουμε τις τιμές του φίλτρου με τις αντίστοιχες τιμές της εικόνας ή στρώματος με τις οποίες το παράθυρο έρχεται σε επαφή και έπειτα αθροίζουμε. Τέλος περνάμε αυτή την τιμή μέσα από μια συνάρτηση ενεργοποίησης αντίστοιχη με αυτές που αναφέρθηκαν στην προηγούμενη ενότητα. Η διαδικασία είναι ευκολότερο να φανεί μέσα από ένα παράδειγμα. Στο Σχήμα 2.9 βλέπουμε πως ένα παράθυρο με διαστάσεις 3x3x1 κεντράρεται πάνω σε μια γκρίζα εικόνα με διαστάσεις 5x5x1 και πως εκτελούμε τις πράξεις για τον υπολογισμό της τιμής του επόμενου στρώματος.



Σχήμα 2.9: εφαρμογή ενός φίλτρου σε γκρίζα εικόνα

Εδώ δεν έχουμε την εφαρμογή κάποιας συνάρτησης ενεργοποίησης για λόγους απλότητας. Στο επόμενο βήμα μετακινούμε το παράθυρο κατά μια θέση στα δεξιά και παίρνουμε έτσι την επόμενη τιμή εκτελώντας ξανά την ίδια διαδικασία. Πρακτικά εκτελούμε συνέλιξη του παραθύρου με την εικόνα. Ολοκληρώνοντας την διαδικασία καταλήγουμε με ένα νέο στρώμα όπως φαίνεται και στο Σχήμα 2.10

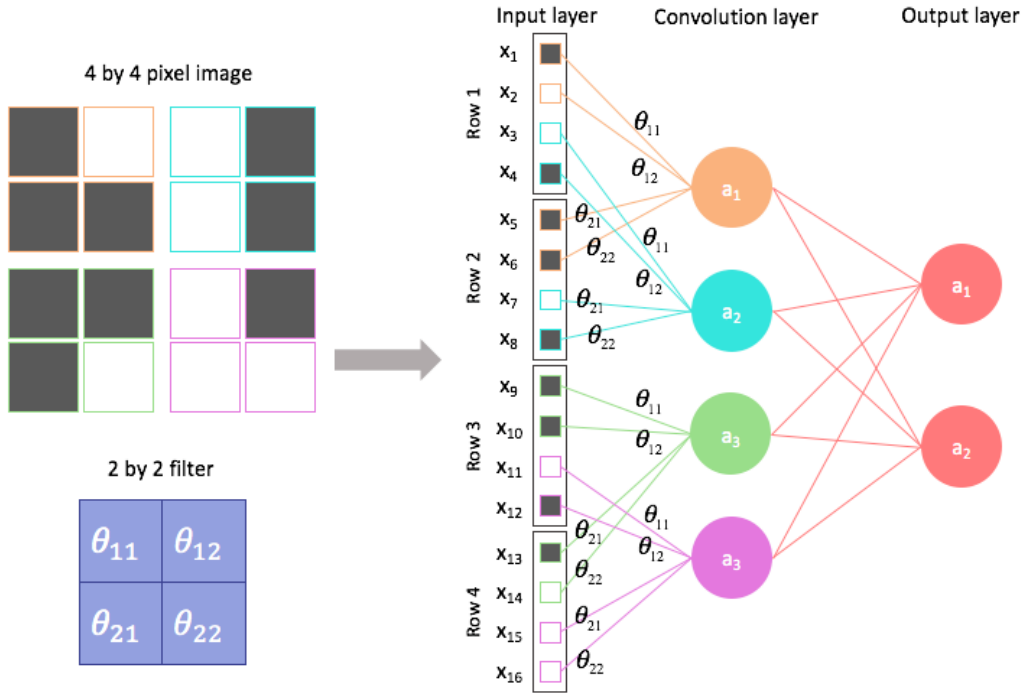


Σχήμα 2.10: Εφαρμογή κυλιόμενου παραθύρου για τον υπολογισμό του επόμενου στρώματος

Ίσως στην παραπάνω περιγραφή να φαίνεται ότι έχουμε ξεφύγει από την δομή ενός νευρωνικού δικτύου. Όμως η αναπαράσταση της παραπάνω διαδικασίας σε δίκτυο προκύπτει εντελώς φυσικά. Κάθε νευρώνας ισοδυναμεί με ένα στοιχείο του νέου στρώματος και συνδέεται με το προηγούμενο στρώμα στο παράδειγμα της εικόνας μας **μόνο** με 3x3x1 νευρώνες, όσο και οι διαστάσεις του φίλτρου και με βάρη τις αντίστοιχες τιμές του φίλτρου. Μάλιστα τώρα γίνεται φανερός ο διαμοιρασμός των παραμέτρων που αναφερθήκαμε προηγουμένως. Κάθε νευρώνας - κελί στο στρώμα που προκύπτει μέσω της πάνω διαδικασίας μοιράζεται τα βάρη των συνδέσεών του με κάθε άλλο νευρώνα σε οποιοδήποτε άλλη θέση στο ίδιο στρώμα. Στο Σχήμα 2.11 έχουμε την αναπαράσταση ενός συνελικτικού στρώματος σε δίκτυο.

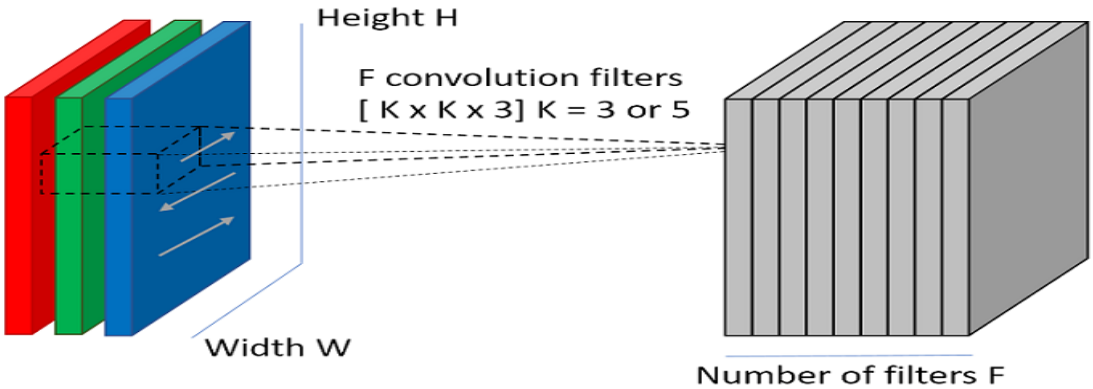
Το βήμα με το οποίο ολισθαίνουμε το παράθυρο λέγεται **stride** και δεν είναι απαραίτητα ίσο με μονάδα. Για παράδειγμα αν το stride είναι 2 στο επόμενο βήμα το παράθυρο θα κεντράρει στο μεθεπόμενο εικονοστοιχείο. Με strides μεγαλύτερα του 1 μειώνουμε τις χωρικές διαστάσεις του επόμενου στρώματος που προκύπτει από την συνέλιξη. Στο σχήμα 2.11 το συνελικτικό στρώμα έχει stride ίσο με 2 καθέτως αλλά και οριζοντίως. Στην προκειμένη περίπτωση μπορούμε να πούμε ότι το stride είναι $(2, 2)$. Βλέπουμε τον διαμοιρασμό παραμέτρων αλλά και το μικρό πλήθος των συνδέσεων σε σχέση με αυτές που θα είχε ένα πλήρες συνδεδεμένο δίκτυο. Τέλος για τον υπολογισμό των παραθύρων που φεύγουν εκτός από τα όρια της εικόνας συνήθως εφαρμόζεται η τεχνική του γεμίσματος με μηδενικά (padding). Δηλαδή επεκτείνουμε όσο χρειάζεται τα όρια της εικόνας δίνοντας μηδενικές (συνήθως) τιμές σε εκείνα τα στοιχεία που προστέθηκαν.

Για να ολοκληρωθεί η περιγραφή του συνελικτικού στρώματος όπως εμφανίζεται στα σύγχρονα δίκτυα μένει να παρατηρήσουμε ότι δεν χρειάζεται να περιοριστούμε σε ένα φίλτρο μόνο. Μπορούμε να έχουμε όσα θέλουμε από αυτά και μάλιστα συνηθίζεται να έχουμε πάρα πολλά.



Σχήμα 2.11: Αναπαράσταση συνελικτικού στρώματος σε μορφή δικτύου

Έστω ότι θέλουμε να προσθέσουμε άλλον ένα φίλτρο στα παραδείγματα μας στο ίδιο συνελικτικό στρώμα. Τότε αυτό που θα συμβεί είναι να προσθέσουμε άλλο ένα επίπεδο δηλαδή να αυξήσουμε το βάθος μας όπως φαίνεται στο Σχήμα 2.12 . Το αποτέλεσμα πολλών φίλτρων δηλαδή η έξοδος αυτή των τριών διαστάσεων την ονομάζουμε και όγκο (volume).



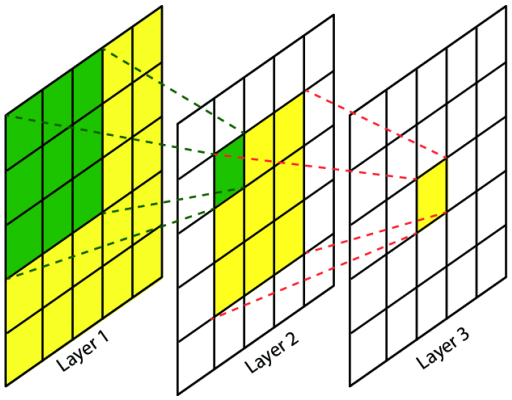
Σχήμα 2.12: Η επίδραση πολλαπλών φίλτρων για την δημιουργία του επόμενου layer

Χρησιμοποιώντας συνελικτικά επίπεδα με stride μεγαλύτερο της μονάδας και με μεγάλο πλήθος φίλτρων μειώνουμε σταδιακά τις χωρικές διαστάσεις της εισόδου $(width, height)$ και αυξάνουμε το βάθος. Για περισσότερες λεπτομέρειες των διάφορων εκδοχών της συνέλιξης με φίλτρα το [Dumo16] παρουσίαζει με λεπτομέρεια την πράξη και τις παραλλαγές της.

2.3.2 Χάρτης χαρακτηριστικών συνελικτικών στρωμάτων και άλλες βασικές έννοιες

Παραπάνω παρουσιάστηκε ο τρόπος υπολογισμού του συνελικτικού στρώματος. Μπορούμε όμως να προχωρήσουμε την ανάλυση μας λίγο βαθύτερα για να καταλάβουμε πως πραγματικά δουλευούν αυτά τα στρώματα.

Οπτικό πεδίο νευρώνα(receptive field) Ας πάρουμε ένα νευρώνα τυχαία σε ένα συνελικτικό στρώμα. Ακολουθώντας τις συνδέσεις του προς τα πίσω στρώματα μπορούμε να γυρίσουμε στην αρχική εικόνα και να δούμε ακριβώς ποια περιοχή είναι υπεύθυνη για την ενεργοποίηση του (Σχήμα 2.13). Αυτή η περιοχή ονομάζεται **receptive field**.
Τα receptive fields νευρώνων σε αρχικά στρώματα είναι μικρά σε έκταση ενώ όσο προχωράμε βαθύτερα αυτά μεγαλώνουν. Ιδανικά στο τελευταίο στρώμα σε μια περίπτωση συνελικτικού δικτύου για ταξινόμηση, οι χωρικές συντεταγμένες έχουν συρρικνωθεί σε 1x1 ,το βάθος είναι όσο οι διαφορετικές κλάσεις προς ταξινόμηση και τέλος το receptive field κάθε νευρώνα είναι όλη η εικόνα.

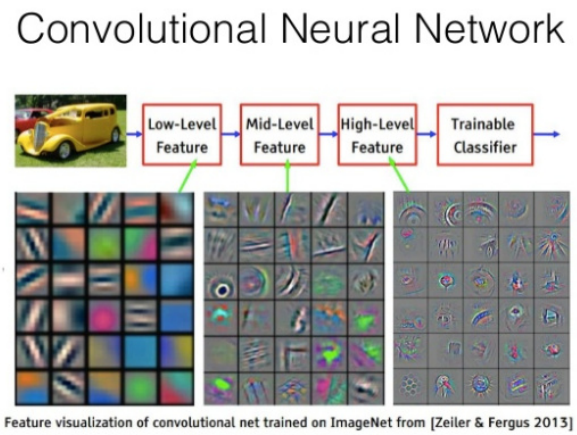


Σχήμα 2.13: Η περιοχή της εικόνας που βλέπει ένας νευρώνας

Χαρακτηριστικό(feature) που αναγνωρίζει ένας νευρώνας Είδαμε ότι κάθε νευρώνας βλέπει ένα μέρος της εικόνας. Χαρακτηριστικό αναγνώρισης του νευρώνα ονομάζουμε το μοτίβο το οποίο πρέπει να υπάρχει στο πεδίο οράσεως του ώστε αυτός να ενεργοποιείται ,δηλαδή να δίνει μεγάλη τιμή στην έξοδο. Οι νευρώνες αρχικών στρωμάτων βλέπουν μικρό τμήμα και αναγνωρίζουν απλά χαρακτηριστικά όπως ακμές ,γωνίες. Οι νευρώνες μετέπειτα στρωμάτων βλέπουν περισσότερο και αναγνωρίζουν πιο σύνθετα χαρακτηριστικά όπως για παράδειγμα ύπαρξη η όχι προσώπου,αντικειμένου κτλ.

Ένας τρόπος να δούμε τι αναγνωρίζει ο νευρώνας αφού εκπαιδεύσουμε το δίκτυο είναι να γυρίσουμε πίσω στην εικόνα ,και κρατώντας σταθερές τις παραμέτρους του δικτύου να εκτελούσε αλγόριθμο απότομης ανόδου πάνω στις τιμές του παραθύρου του receptive field ,βελτιστοποιώντας ως προς την τιμή ενεργοποίησης εκείνου του νευρώνα. Αυτή η διαδικασία λέγεται και **deep dream** και μας δίνει πολύτιμη γνώση για το πως τελικά δουλεύει το δίκτυο μας. Στο Σχήμα 2.14 φαίνεται ένα παράδειγμα του ποια είναι τα χαρακτηριστικά που αναγνωρίζονται στα διάφορα στρώματα.

Ο χάρτης χαρακτηριστικών (feature map) Ας πάρουμε τώρα ένα συγκεκριμένο layer με διαστάσεις έστω WxHxD. Όλοι οι νευρώνες με συντεταγμένες (x,y,c) όπου το c μια σταθερά και x,y μεταβλητές χωρικές συντεταγμένες, έχουν το ίδιο μέγεθος receptive field και ανιχνεύουν το ίδιο χαρακτηριστικό σε διαφορετικά τμήματα της εικόνας. Αυτό είναι εύκολα κατανοητό αν σκεφτούμε το εξής. Όλοι αυτοί οι νευρώνες συνδέονται με τα ίδια βάρη (ή δυικά υπολογίζονται με το ίδιο φίλτρο) σε νευρώνες του προηγούμενου στρώματος ,οι οποίοι αντίστοιχα συνδέονται με το δικό τους προηγούμενο στρώμα με ίδιες παραμέτρους.
Κάθε ένας από αυτούς λοιπόν υπολογίζει την ίδια συνάρτηση επί της εισόδου ,η οποία είσοδος είναι



Σχήμα 2.14: Χαρακτηριστικά που ανακαλύπτουν οι νευρώνες των κρυφών στρωμάτων

κάποιο τμήμα της εικόνας. Αυτή είναι η λεγόμενη χωρική αμεταβλητότητα (spatial invariance) των συνελικτικών νευρωνικών δικτύων!
Αν τώρα αλλάξουμε το c σε c' παίρνουμε ένα άλλο σετ W επί H νευρώνες οι οποίοι προκύπτουν απο διαφορετικό φίλτρο και αναγνωρίζουν ένα άλλο χαρακτηριστικό πάλι σε διαφορετικές θέσεις της εικόνας.

Ο όγκος χαρακτηριστικών (feature volume) Ένα συνελικτικό επίπεδο προκύπτει από πολλά φίλτρα . Όλοι αυτοί οι ξεχωριστοί χάρτες χαρακτηριστικών μαζί λέγονται και feature volume. Κάθε χάρτης ψάχνει και ένα χαρακτηριστικό σε διαφορετικά τμήματα της εικόνας και όλοι οι χάρτες έχουν το ίδιο μέγεθος receptive field. Οι παραπάνω έννοιες φαίνονται καλύτερα στο Σχήμα 2.15

Επίδραση του stride και padding στις διαστάσεις του όγκου χαρακτηριστικών της εξόδου Τέλος ενώ έχουμε αναφερθεί στις έννοιες του βήματος μετακίνησης και στο γέμισμα με μηδενικά δεν έχουμε δει πως ακριβώς μεταβάλλουν τις διαστάσεις του παραγόμενου στρώματος. Προκύπτει ότι μπορούμε να πάρουμε τις νέες διαστάσεις πολύ απλά από τους παρακάτω τύπους.

$$W' = \frac{W - F}{S + 1}$$

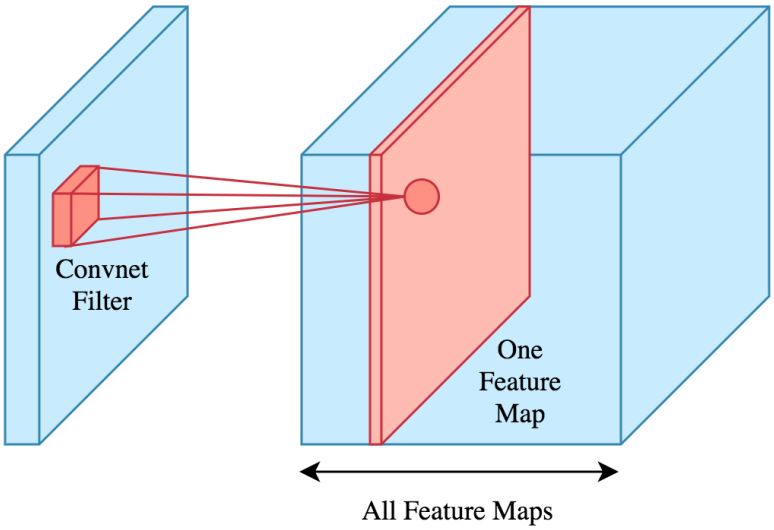
$$H' = \frac{H - F}{S + 1}$$

$$D' = K$$

Όπου S είναι η τιμή του stride (θεωρείται ίσο και για τις δύο διαστάσεις), F είναι ο αριθμός των έξτρα μηδενικών που προσθέσαμε στην άκρη της εικόνας. Ακόμα τα W',W είναι το νέο και το παλιό πλάτος αντίστοιχα , H',H το νέο ύψος και το παλιό ενώ D' είναι το νέο βάθος και K το πλήθος των φίλτρων που χρησιμοποιήθηκαν.

2.3.3 Πλήρως συνδεδεμένα στρώματα ως συνελικτικά

Είδαμε πως επιλέγοντας stride μεγαλύτερο της μονάδας μειώνουμε τις χωρικές διαστάσεις και πως το βάθος ισούται με τον αριθμό των φίλτρων που εφαρμόζουμε. Τα τελευταία στρώματα ενός



Σχήμα 2.15: Πολλοί χάρτες χαρακτηριστικών απαρτούν ένα convolutional layer

συνελικτικού δικτύου είναι στην περίπτωση ενός ταξινομητή πλήρως συνδεδεμένα(fully connected ή εν συντομία fc) .

Μπορούμε να δούμε ένα τέτοιο fc στρώμα ως ένα συνελικτικό στρώμα με διαστάσεις 1x1xK. Αυτό το στρώμα προκύπτει από την εφαρμογή k το πλήθος WxHxD φίλτρων πάνω σε ένα WxHxD συνελικτικό στρώμα. Το δίκτυο μας λοιπόν μπορεί σε κάποιο στάδιο να εφαρμόζει ένα τέτοιο φίλτρο για να μετατρέψει την έξοδο σε ένα τέτοιο διάνυσμα . Μάλιστα η ίδια η αρχιτεκτονική του δικτύου μπορεί να ορίζει ένα τέτοιο φίλτρο με μεταβλητές χωρικές διαστάσεις, έτσι ώστε να είμαστε σε θέση να επεξεργαστούμε εικόνες με μεταβλητές διαστάσεις αν μεσολαβεί στρώμα χωρις εκπαιδεύσιμες παραμέτρους(πχ average pooling) .

Μετέπειτα fc στρώματα μπορούν να αναπαρασταθούν ως εφαρμογές k' φλιτρων διαστάσεων 1x1xK που δίνουν αντίστοιχη έξοδο 1x1xK' . Έτσι δεν χρειάζεται να αλλάξουμε τον τρόπο που βλέπουμε και κατ' επέκταση υπολογίζουμε τα συνελικτικά στρώματα. Μάλιστα σε αντιστοίχηση της παραπάνω μας περιγραφής του χάρτη χαρακτηριστικών μπορούμε να δούμε το εξής.
Ένα στρώμα 1x1xK θα αποτελείται ουσιαστικά από K χάρτες χαρακτηριστικών, όπου κάθε χάρτης ανακαλύπτει διαφορετικό χαρακτηριστικό και μάλιστα αποτελείται από έναν και μόνο νευρώνα , ο οποίος έχει ως receptive field ολόκληρη την αρχική εικόνα! Ουσιαστικά κάθε επόμενο layer συνδυάζει τα χαρακτηριστικά που ανακαλύπτει το προηγούμενο layer για να δημιουργήσει γενικότερα χαρακτηριστικά με τελικό αποτέλεσμα ,να ανακαλύψουμε ένα χαρακτηριστικό ολόκληρης της εικόνας. Αυτό θα μπορούσε να ήταν η παρουσία για παράδειγμα κάποιου αντικειμένου.

2.3.4 Η ανάστροφη διάδοσή σε συνελικτικά στρώματα

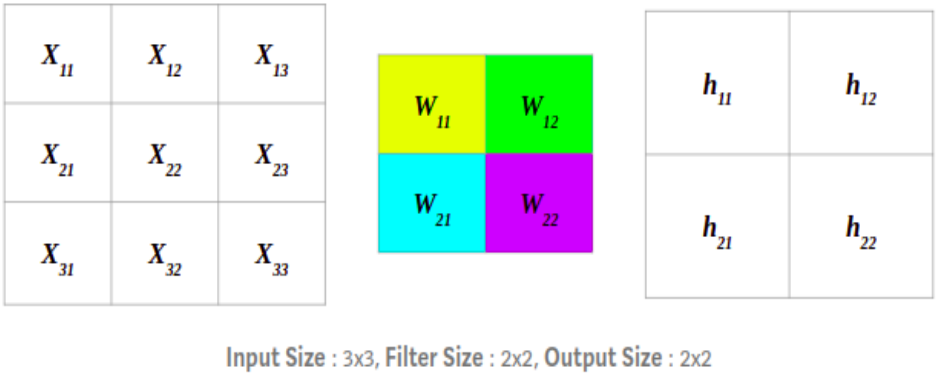
Σε προηγούμενο κεφάλαιο περιγράψαμε το πως υπολογίζονται οι μερικές παράγωγοι της συνάρτησης σφάλματος ως προς οποιαδήποτε παράμετρο ενός πλήρως συνδεδεμένου δικτύου. Στην περίπτωση των συνελικτικών δικτύων τα πράματα είναι ελαφρώς διαφορετικά λόγω του διαμοιρασμού των παραμέτρων. Οι νευρώνες του ίδιου καναλιού του ίδιου στρώματος μοιράζονται τα ίδια βάρη για τις συνδέσεις τους οπότε και η απλή διάδοση προς τα πίσω ίσως φαίνεται ότι δεν δουλεύει. Κάτι τέτοιο είναι λάθος αφού μπορούμε πάλι να χρησιμοποιήσουμε τον κανόνα της αλυσίδας.

Ας δούμε ένα παράδειγμα συνέλιξης όπου το φίλτρο και η είσοδος έχουν βάθος ίσο με 1 για λόγους απλότητας. Η συνέλιξη παρουσιάζεται στο Σχήμα 2.16.

Έχουμε τις παρακάτω εξισώσεις

$$h_{11} = W_{11}X_{11} + W_{12}X_{12} + W_{21}X_{21} + W_{22}X_{22}$$

$$h_{12} = W_{11}X_{12} + W_{12}X_{13} + W_{21}X_{22} + W_{22}X_{23}$$



Σχήμα 2.16: Συνέλιξη εισόδου X με φίλτρο W

$$h_{21} = W_{11}X_{21} + W_{12}X_{22} + W_{21}X_{31} + W_{22}X_{32}$$
$$h_{22} = W_{11}X_{22} + W_{12}X_{23} + W_{21}X_{32} + W_{22}X_{33}$$

Έστω ότι γνωρίζουμε τις μερικές παραγώγους της συνάρτησης σφάλματος F ως προς την έξοδο h (επαγωγικό βήμα του αλγόριθμου ανάστροφης διάδοσης) .Τότε πάλι από τον κανόνα της αλυσίδας έχουμε

$$\frac{\partial F}{\partial W_{11}} = \frac{\partial F}{\partial h_{11}}\frac{\partial h_{11}}{\partial W_{11}} + \frac{\partial F}{\partial h_{12}}\frac{\partial h_{12}}{\partial W_{11}} + \frac{\partial F}{\partial h_{21}}\frac{\partial h_{21}}{\partial W_{11}} + \frac{\partial F}{\partial h_{22}}\frac{\partial h_{22}}{\partial W_{11}}$$

Αντικαθιστώντας τις $\frac{\partial h_{ij}}{\partial W_{11}}$ με αυτές από το προηγούμενο σετ εξισώσεων έχουμε

$$\frac{\partial F}{\partial W_{11}} = \frac{\partial F}{\partial h_{11}}X_{11} + \frac{\partial F}{\partial h_{12}}X_{12} + \frac{\partial F}{\partial h_{21}}X_{21} + \frac{\partial F}{\partial h_{22}}X_{22}$$

Εντελώς όμοια μπορούμε να βρούμε και τις μερικές παραγώγους των υπόλοιπων βαρών του φίλτρου τα οποία είναι.

$$\frac{\partial F}{\partial W_{12}} = \frac{\partial F}{\partial h_{11}}X_{12} + \frac{\partial F}{\partial h_{12}}X_{13} + \frac{\partial F}{\partial h_{21}}X_{22} + \frac{\partial F}{\partial h_{22}}X_{23}$$

$$\frac{\partial F}{\partial W_{21}} = \frac{\partial F}{\partial h_{11}}X_{21} + \frac{\partial F}{\partial h_{12}}X_{22} + \frac{\partial F}{\partial h_{31}}X_{22} + \frac{\partial F}{\partial h_{22}}X_{32}$$

$$\frac{\partial F}{\partial W_{22}} = \frac{\partial F}{\partial h_{11}}X_{22} + \frac{\partial F}{\partial h_{12}}X_{23} + \frac{\partial F}{\partial h_{21}}X_{32} + \frac{\partial F}{\partial h_{22}}X_{33}$$

Οι παραπάνω εξισώσεις δεν είναι τίποτα άλλο από την παρακάτω συνέλιξη

$$\frac{\partial F}{\partial \mathbf{W}} = \begin{pmatrix} X_{11} & X_{12} & X_{13} \\ X_{21} & X_{22} & X_{23} \\ X_{31} & X_{32} & X_{33} \end{pmatrix} * \begin{pmatrix} \frac{\partial F}{\partial h_{11}} & \frac{\partial F}{\partial h_{12}} \\ \\ \frac{\partial F}{\partial h_{21}} & \frac{\partial F}{\partial h_{22}} \end{pmatrix}$$

Όμως για να μπορέσουμε να υπολογίσουμε τώρα τα βάρη του ακριβώς προηγούμενου layer από αυτό στο οποίο έγινε η ανάλυση δηλαδή δρώντας αναδρομικά, θα θέλουμε τις μερικές παραγώγους ως προς την είσοδο στο πάνω παράδειγμα, μιας και αυτή θα είναι η έξοδος για το προηγούμενο στρώμα. Με παρόμοια ανάλυση προκύπτει ότι μας το δίνει η παρακάτω συνέλιξη.

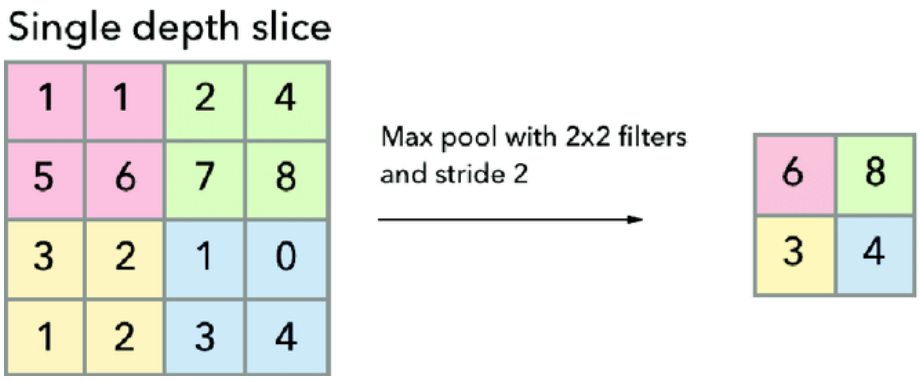
$$\frac{\partial F}{\partial \mathbf{X}} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & \frac{\partial F}{\partial h_{11}} & \frac{\partial F}{\partial h_{12}} & 0 \\ 0 & \frac{\partial F}{\partial h_{21}} & \frac{\partial F}{\partial h_{22}} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} * \begin{pmatrix} W_{22} & W_{21} \\ W_{12} & W_{11} \end{pmatrix}$$

Ο δεξιός πίνακας στην παραπάνω σχέση είναι ο W αφού γίνει transposed και προς τις δύο διαγώνιους. Προκύπτει ότι μπορούμε με παρόμοια διαδικασία να υπολογίσουμε την οπισθόδρομη μεταφορά των παραγώγων σε συνελικτικά στρώματα για οποιαδήποτε τιμες stride , padding και διαστάσεις φίλτρου ακόμα και με βάθος μεγαλύτερο του ένα. Για παράδειγμα για s>1 τοποθετούμε μηδενικά ανάμεσα στις τιμές του πίνακα των μερικών παραγώγων του στρώματος εξόδου.

Συμπερασματικά διαπιστώνουμε ότι ο αλγόριθμος back propagation δεν έχει κανένα πρόβλημα να εφαρμοστεί και σε συνελικτικά επίπεδα. Αυτό είναι ιδιαίτερα σημαντικό γιατί τώρα τα φίλτρα μας επιδέχονται εκπαίδευση.

2.3.5 Στρώματα συγκράτησης μεγίστης τιμής (max pooling)

Μέχρι στιγμής είδαμε πως επιλέγοντας stride μεγαλύτερο της μονάδας μειώνουμε τις δύο διαστάσεις που αντιστοιχούν σε χώρο της εικόνας , και πως επιλέγοντας όλο και περισσότερα φίλτρα αυξάνουμε το βάθος.
Υπάρχει και άλλο ένα είδος στρώματος που μειώνει της χωρικές διαστάσεις και που ήταν αρκετά διαδεδομένο ,αν και η δημοτικότητα του βρίσκεται σε πτώση τελευταία. Τα στρώματα αυτά ονομάζονται στρώματα max pooling. Στην ουσία εφαρμόζουν ένα συγκεκριμένο (μη εκπαιδεύσιμο) φίλτρο που έχει ως σκοπό τη συγκράτηση της μεγαλύτερης τιμής σε μια περιοχή όση και το παράθυρο του φίλτρου. Το στρώμα αυτό έχει επίσης τις χαρακτηριστικές υπερπαραμέτρους των συνελικτικών όπως το stride,padding κτλ. Στο Σχήμα 2.17 φαίνεται η εφαρμογή ενός τέτοιου στρώματος.



Σχήμα 2.17: Max pooling στρώμα με stride 2,2

2.3.6 Στρώματα εισαγωγής μη γραμμικότητας

Έως τώρα αγνοήσαμε τις συναρτήσεις ενεργοποίησης που εισάγουν τις μη γραμμικότητες. Η αλήθεια είναι ότι χωρίς αυτές μπορεί να δειχθεί ότι ένα ολόκληρο δίκτυο από συνελικτικά στρώματα γίνεται ισοδύναμο με ένα μεγάλο συνελικτικό στρώμα. Φαίνεται λοιπόν ότι η συνέλίξη από μόνη της δεν είναι αρκετή. Υπάρχουν πολλές συναρτήσεις ενεργοποίησης που χρησιμοποιούνται με κυρίαρχες τις ReLU και LeakyReLU που αναφέρθηκαν παραπάνω. Μας βολεύει επίσης όταν χτίζουμε τα μοντέλα μας να συμπεριφερόμαστε σε αυτές τις συναρτήσεις ως ξεχωριστά στρώματα του δικτύου και όχι να τις θεωρούμε ενσωματωμένες στα ίδια τα συνελικτικά στρώματα. Αυτή η περιγραφή ακολουθείτε συχνά στις διάφορες μελέτες και θα φανεί χρήσιμη όταν παρουσιάσουμε τα διάφορα μοντέλα ανίχνευσης και ταξινίμησης αντικειμένων που θα χρησιμοποιηθούν.

2.3.7 Στρώματα κανονικοποίησης συνόλων παραγωγής (batch normalization)

Τα στρώματα που εκτελούν batch normalization χρησιμοποιούνται σε όλα τα μοντέλα που θα δούμε και επιταχύνουν την διαδικασία της εκπαίδευσης αρκετά.
Η προεπεξεργασία των δεδομένων είναι ιδιαίτερα συνήθης τεχνική στην μηχανική μάθηση. Ένα συχνό παράδειγμα είναι να έχουμε χαρακτηριστκά με διαφορετικής τάξης μεγέθη. Δεν μας ενδιαφέρουν οι τιμές διάφορων χαρακτηριστικών ως απόλυτες αλλά πως οι διακυμάνσεις σε αυτές τις τιμές επηρεάζουν την πρόβλεψη. Αρκετές φορές κάποιο χαρακτηριστικό έχει μάλιστα τρομέρα μεγάλη ή μικρή τιμή σε σχέση με τα υπόλοιπα και αυτό μπορεί να οδηγήσει σε αδυναμία του μοντέλου να εκπαιδευτεί. Έτσι συνήθως εφαρμόζεται κάποια στρατηγική κανονικοποίησης των τιμών σε κοινό διάστημα.

Την παραπάνω διαδικασία εκτελούν και τα batch normalization στρώματα στα συνελικτικά δίκτυα. Συγκεκριμένα πριν τα επίπεδα ενεργοποίησης οι χάρτες χαρακτηριστικών που προέκυψαν περνάνε μια κανονικοποίηση που μπορεί να γίνει με αφαίρεση μέσης τιμής και διαίρεση της τυπικής απόκλισης όπως αυτές προκύπτουν από το σύνολο των τιμών για το χάρτη σε όλο το batch. Έχει δειχθεί πειραματικά αρκετά σοβαρή βελτίωση στην απόδοση και στον χρόνο σύγκλισης από την κανονικοποίηση.

2.4 Έννοιες και ορισμοί για τη μελέτη αναγνώρισης και ανίχνευσης αντικειμένων

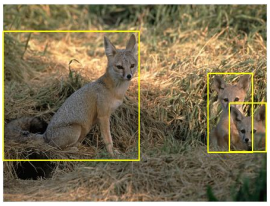
Εδώ θα αναφερθούμε σε βασικές έννοιες που χρησιμοποιούνται στον τομέα της ανίχνευσης αντικειμένων, μέσω της βαθιάς μηχανικής μάθησης αλλά και γενικότερα. Αυτές οι έννοιες είναι σημαντικές για την κατανόηση της λειτουργίας των διάφορων μοντέλων που θα χρησιμοποιηθούν στην παρούσα εργασία και θα αναλυθούν στο επόμενο κεφάλαιο.

2.4.1 Ορθογώνιο οριοθέτησης (bounding box)

Το ορθογώνιο οριοθέτησης (bounding box) χρησιμοποιείται για την αναπαράσταση της περιοχής που ανήκει ένα αντικείμενο. Είναι ένα ορθογώνιο το οποίο περικλείει το αντικείμενο όσο πιο αντιπροσωπευτικά γίνεται. Μπορούμε να θεωρήσουμε ότι το βέλτιστο ορθογώνιο οριοθέτησης ενός αντικειμένου είναι το μικρότερο ορθογώνιο από όλα τα ορθογώνια που το περιέχουν.
Για την αναπαράσταση του χρειαζόμαστε 4 τιμές και αυτές είναι είτε οι συντεταγμένες των δύο αντικριστών γωνιακών του σημείων ,είτε πιο συχνά οι δύο συντεταγμένες του κέντρου του καθώς και άλλες δύο τιμές που αντιπροσωπεύουν το πλάτος και το ύψος του.

Τα μοντέλα ανίχνευσης θα δίνουν διάφορες προτάσεις από bounding boxes αντικειμένων μέσω μιας ευριστικής αναζήτησης ή και άλλων αλγορίθμων , ή όπως γίνεται σήμερα μέσω ειδικών συνελικτικών δικτύων όπως για παράδειγμα το RPN (region proposal network).
Ο τρόπος με τον οποίο θα συγκρίνουμε δύο τέτοια bounding boxes (x, y, w, h) και (x', y', w', h') μεταξύ τους θα είναι συνήθως η L2 απόσταση (ευκλείδια νόρμα) των διανυσματικών αναπαραστάσεων

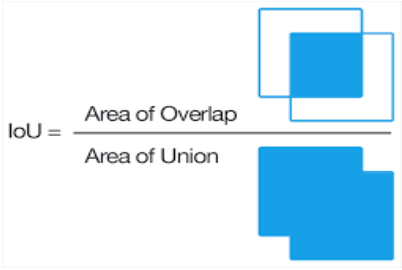
τους. Μάλιστα οι τιμές πλάτους w και ύψους h μπορούν να μετασχηματιστούν μέσω κάποιας λογαριθμικής συνάρτησης . Η τεχνική της λογαρίθμησης των τιμών είναι ευρέως διαδεδομένη στη βαθιά μηχανική μάθηση.



Σχήμα 2.18: Bounding boxes (εικόνα από imagenet)

2.4.2 Λόγος τομής προς ένωση (Intersection over Union)

Το Intersesction over Union ή εν συντομία IoU είναι μια μετρική απόδοσης που χρησιμοποιείται κατα κόρον στην ανίχνευση αντικειμένων. Για την περίπτωση δύο ορθογωνίων οριοθέτησης μπορούμε να υπολογίσουμε το IoU τους ως εξής. Το IoU θα είναι το ο λόγος του ”εμβαδόν” της περιοχής που είναι κοινή και για τα 2 ορθογώνια προς το εμβαδόν όλης της περιοχής που τα δύο ορθογώνια καλύπτουν. Μπορούμε έτσι να μετρήσουμε πόσο καλή ήταν η πρόβλεψη ορθογωνίου μας έναντι του ground truth ορθογωνίου αλλά και να μετρήσουμε την ομοιότητα ορθογωνίων προβλέψεων με σκοπό να αφαιρέσουμε προτάσεις που μοιάζουν με άλλες αλλά υστερούν σε πιθανότητα ύπαρξης αντικειμένου.



Σχήμα 2.19: Λόγος τομής προς ένωση

Η έννοια του IoU δεν είναι κάτι που περιορίζεται σε εικόνες και ορθογώνια οριοθέτησης όμως .Σαν μετρική δεν είναι τίποτα άλλο από το πλήθος των true positives δια το πλήθος των false positives συν true positives συν false negatives.
Συγκεκριμένα μπορούμε στην ανίχνευση να θεωρήσουμε δύο κλάσεις που περιγράφουν τα διάφορα σημεία της εικόνας. Η θετική αντιστοιχεί σε σημεία που ανήκουν στο ground truth (πραγματικό) ορθογώνιο οριοθέτησης και η αρνητική αντιστοιχεί σε σημεία που δεν ανήκουν σε αυτό.

$$IoU = \frac{TruePositives}{TruePositives + FalsePositives + FalseNegatives}$$

True positives λοιπον για μας θα είναι τα σημεία που το δίκτυο μας βρήκε ότι ανήκουν στο ορθογώνιο οριοθέτησης του αντικειμένου και αυτά, ανήκαν και στην περιοχή του ground truth ορθογωνίου. Δηλαδή σχηματικά είναι η τομή των δύο ορθογωνίων. False positives είναι τα σημεία που ανήκουν στο ορθογώνιο πρόβλεψης αλλά όχι στο αληθινό ορθογώνιο. Τέλος false negatives είναι τα σημεία που δεν ανήκουν στο ορθογώνιο πρόβλεψης αλλά ανήκουν στο αληθινό.

Η IoU ως μετρική για την περίπτωσή μας είναι ιδιαίτερα βολική γιατί δίνει μεγάλες τιμές μόνο για τις προβλέψεις ορθογωνίων που όντως θεωρούμε επιτυχείς. Αυτό φαίνεται καλύτερα αν αναλογιστούμε τι τιμές παίρνει στις παρακάτω περιπτώσεις.
Στην περίπτωση που η πρόβλεψη μας δίνει αρκετά μικρότερο σε έκταση ορθογώνιο από το σωστό

τότε το IoU θα είναι μικρό ακόμα και αν το ορθογώνιο μας προβλέπει σωστά τις τιμές εντός του. Παρόμοια αν το ορθογώνιο της πρόβλεψης μας είναι τεράστιο ,και ως αποτέλεσμα έχουμε μεγάλη πιθανότητα να περιέχεται το αληθινό ορθογώνιο εντός του, το IoU είναι μικρό ακόμα και έχουμε βρει όλα τα θετικά σημεία.
Μια άλλη πιο αφελής μετρική όπως η τομή (true positives) θα έκανε τα δίκτυα biased και θα τα οδηγούσε στην επιλογή τεράστιων ορθογωνίων οριοθέτησης για τις προβλέψεις τους.

2.4.3 Average Precision και mean Average Precision για αναγνώριση αντικειμένων

Με χηση της IoU μπορούμε ορίσουμε την Average Precision για την αναγνώριση αντικειμένων σε εικόνες. Η βασική μέθοδος καθώς και άλλες παραλλαγές μπορεί να δει κανείς στο Pascal VOC Challenge [Ever10]. Αρχικά πρέπει να προσδιορίσουμε πότε μια πρόβλεψη ορθογωνίου αντικειμένου θα θεωρείται True Positive, False Positive και False Negative. Μόλις ορίσουμε αυτά τα κριτήρια μετά μπορούμε να υπολογίσουμε και την Average Precision.

- **True Positive** πρόβλεψη θεωρούμε τις προβλέψεις με bounding box που δίνει IoU > 0.5 με το Ground Truth bounding box και ταυτόχρονα το μοντέλο προβλέπει σωστά την κλάση του αντικειμένου.

- **False Positive** πρόβλεψη θεωρούμε τις προβλέψεις που είτε έχουν bounding box που δίνει IoU < 0.5 με οποιοδήποτε Ground Truth box, είτε έχουν IoU > 0.5 με κάποιο Ground Truth box αλλά υπάρχει ήδη καλύτερη πρόβλεψη που ανιχνέυει το ίδιο ορθογώνιο.

- **False Negative** πρόβλεψη έχουμε όταν το ορθογώνιο έχει IoU > 0.5 με κάποιο αλλά η ταξινόμηση της κλάσης για το αντικείμενο εντός του είναι λάθος.

Μπορούμε να πάρουμε όποιο όριο μας βολεύει για την IoU και θα λέμε τότε ότι υπολογίζουμε την AP σε εκείνη την τιμή IoU. Αφού λοιπόν χαρακτηρίσουμε τις προβλέψεις μας με τα παραπάνω μπορούμε να τις ταξινομήσουμε πάλι σύμφωνα με το σκορ τους. Το σκορ σε αυτήν την περίπτωση θα είναι ο βαθμός εμπιστοσύνης που μας δίνει κάποιο Softmax layer στην ταξινόμηση του αντικειμένου εντός του ορθογωνίου οριοθέτησης. Παραπάνω πληροφορία για το πως γίνεται αυτό θα δούμε στο Κεφάλαιο 3. Αφού ταξινομήσουμε τις προβλέψεις θα υπολογίζουμε ακριβώς όπως και στην περίπτωση απλής ταξινόμησης την Precision για κάθε Recall με τον ίδιο τύπο (παράδειγμα πίνακα 2.1). Τα θετικά παραδείγματα μιας κλάσης θα είναι όλα τα αντικείμενα της κλάσης που υπάρχουν στο σύνολο των εικόνων της αξιολόγησης τα όποια παίρνουμε με τα αντίστοιχα ορθογώνια τους. Θα μπορούσε μια εικόνα να περιλαμβάνει πάνω από μια υπόσταση αυτού του αντικειμένου.

Η Average Precision εδώ τιμώρεί τα μοντέλα που προβλέπουν πολλές φορές το ίδιο πράγμα ή προβλέπουν πράγματα που δεν υπάρχουν. Επίσης τιμωρεί το μοντέλο όταν αυτό δεν βρίσκει κάποιο ορθογώνιο αντικειμένου μιας και όσο περισσότερα αντικειμένα ”χάσουμε” τόσο χαμηλότερο άνω όριο recall θα έχουμε στο άθροισμα. Στο Pascal VOC Challenge εισάγεται και η έννοια της interpolated precision για να μειωθεί η επίδραση από τις διακυμάνσεις που δίνει η μικρή εναλλαγή της σειράς ταξινόμησης των προβλέψεων και να εξομαλύνουν την καμπύλη precision-recall της οποιάς πρακτικά μετράμε το εμβαδόν όταν υπολογίζουμε την Average Precision;

Τέλος όσον αφορα την mean Average Precision αυτή είναι και πάλι ο μέσορς όρος της Average Precision για όλες τις κλάσεις δηλαδή ακριβώς όπως και στην περίπτωση της απλής ταξινόμησης.

2.4.4 Καταστολή μη μεγίστων (Non Maximum Suppression)

Αυτή είναι μια τεχνική που λύνει το πρόβλημα των πολλαπλών προβλέψεων που αντιστοιχούν στο ίδιο αντικείμενο. Ο τρόπος με τον οποίο τα μοντέλα ανίχνευσης προβλέπουν ορθογώνια οδηγεί σε πολλαπλές προβλέψεις που υπερκαλύπτουν η μια την άλλη . Έτσι αρχικά χρειάζεται να απορρίπτουμε προβλέψεις που έχουν χαμηλή πιθανότητα να αντιστοιχούν σε αντικείμενο . Όμως αυτό από μόνο του δεν είναι αρκετό οπότε χρειάζεται να απορρίπτουμε προβλέψεις που έχουν σοβαρή επικάλυψη μεταξύ τους, την οποία μετράμε με IoU ,κρατώντας μόνο αυτή με την μεγαλύτερη πιθανότητα. Τα όρια της πιθανότητας και της υπερκάλυψης που κάνουν αυτήν την επιλογή στις προβλέψεις είναι υπερπαράμετροι των μοντέλων μας.

2.4.5 Βελτίωση Ορθογωνίου Οριοθέτησης (Bounding Box Refinement)

Πολλά από τα μοντέλα ανίχνευσης αντικειμένων χρησιμοποιούν ένα στάδιο όπου πραγματοποιείται μετασχημάτισμος του ορθογωνίου της πρόβλεψης από ξεχωριστό νευρωνικό δίκτυο ή άλλου είδος αλγόριθμο όπως για παράδειγμα SVM regressor. Το τμήμα του δικτύου που επιτελεί αυτή τη διαδικασία λέγεται bounding box regressor. Ο μετασχηματισμός αυτός είναι απαραίτητος για την επίδοση του μοντέλου στο να βρίσκει με ακρίβεια τον στόχο. Η σημασία αυτού του σταδίου εξηγείται σε μελέτες όπως [Girs13].

Σε αυτό το στάδιο έχουμε την επιλογή αν θα έχουμε ένα bounding box regressor για κάθε ξεχωριστή κλάση αντικειμένων που θα προβλέψουμε, ή έναν ενιαίο που εκπαιδεύεται για όλες τις κλάσεις.Γενικά όλοι οι αυτοι οι regressors συνδυάζονται με ταξινομητές που δίνουν πιθανότητες για την πρόβλεψη μιας κλάσης εντός του ορθογωνίου οριοθέτησης καθώς αυτό μεταβάλλεται.Έχουμε και πάλι την επιλογή να έχουμε έναν ταξινομητή ανά κλάση ή έναν ενιαίο που δίνει πιθανότητα γενικά για την ύπαρξη κάποιου αντικειμένου εντός του ορθογωνίου.

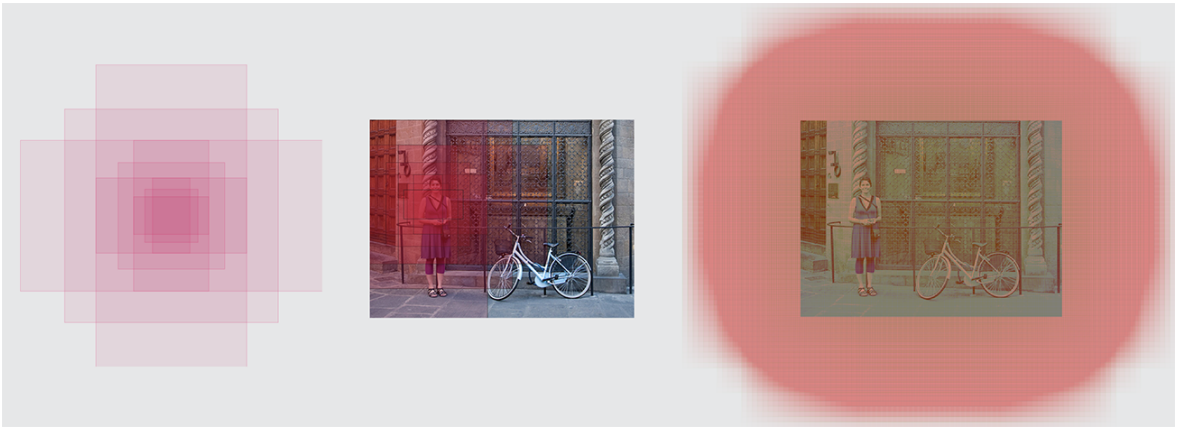
2.4.6 Κουτία Άγκυρες (Anchor Boxes)

Τα anchor boxes είναι επίσης μια πολύ σημαντική έννοια που χρειάζεται να εξηγηθεί για να καταλάβει κανείς πως δουλεύουν μοντέλα ανίχνευσης που είναι state of the art αυτή τη στιγμή (mask r-cnn , yolo v2/v3 και άλλα).

Είπαμε πως ένα μοντέλο ανίχνευσης αντικειμένων οφείλει να δημιουργεί προτάσεις από bounding boxes στα οποία υπάρχουν πιθανά τέτοια αντικείμενα. Όμως είναι εύλογο να αναρωτηθεί κανείς πως κωδικοποιούνται αυτές οι προτάσεις και ίσως πιο βασικά πως υπερνικάμε το πρόβλημα ότι θέλουμε να έχουμε μεταβλητό πλήθος από αυτές. Τα δίκτυα που είδαμε μέχρι τώρα δέχονταν όλα σταθερό μήκος από διανύσματα εισόδου. Το πρόβλημα αυτό λύνουν τα anchor boxes ή αλλιώς prior boxes.

Τα anchors λοιπόν είναι σταθερού μεγέθους και σχήματος κουτιά οριοθέτησης που τοποθετούμε ομοιόμορφα σε διάφορα σημεία της εικόνας. Μπορούμε να δούμε τη διαδικασία ως εξής.
Αρχικά χωρίζουμε την εικόνα σε n επι n κελιά και για κάθε ένα από αυτά στο κέντρο του σχεδιάζουμε k κουτιά οριοθέτησης με προκαθορισμένες διαστάσεις.Κάποια anchor boxes θα μπορούσαν να είναι τετράγωνα μικρά και μεγάλα,άλλα ορθογώνια με μεγάλο μήκος,άλλα ορθογώνια με μεγάλο ύψος και ότου καθ' εξής. Βέβαια το να δημιουργούμε anchors για κάθε ξεχωριστό εικονοστοιχείο γενικά είναι ασύμφορο υπολογιστικά ενώ επιπλέον ,θα δούμε στην συνέχεια ότι δουλεύουμε πάνω στον χάρτη χαρακτηριστικών που εξάγει κάποιο προ-εκπαιδευμένο συνελικτικό νευρωνικό δίκτυο και όχι ακριβώς στην αρχική εικόνα.
Στο Σχήμα 2.20 στο οποίο φαίνεται κάποιο στάδιο από την ανάλυση του μοντέλου Faster R-CNN ,έχουν σχεδιαστεί κάποια από τα anchor boxes πάνω στην αρχική εικόνα.

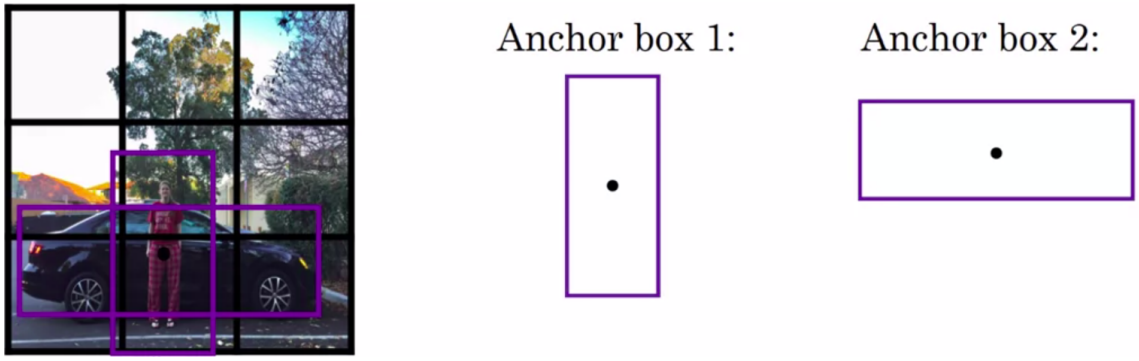
Τα anchor boxes δεν είναι αναγκαία για την ανίχνευση αντικειμένων. Για παράδειγμα ο αλγόριθμος yolo v1 δεν τα χρησιμοποιούσε. Όμως η χρησιμότητά τους είναι υψίστης σημασίας. Αρχικά μπορούμε να σκεφτούμε την περίπτωση όπου έχουμε δύο αντικείμενα τα οποία έχουν το ίδιο κέντρο στην εικόνα. Χωρίς anchor boxes μπορούμε να έχουμε ακριβώς μια πρόβλεψη που θα γίνει generated από αυτό το κέντρο. Έτσι θα ανιχνεύσουμε μόνο ένα από τα δύο αντικείμενα.



Σχήμα 2.20: Τα anchor boxes για ένα σημείο (αριστερά),για ένα σημείο πάνω στην εικόνα (μέση) ,όλα μαζί στην εικόνα (δεξιά)

Αντίθετα αν έχουμε πολλαπλά anchor boxes τότε διαφορετικά anchors μπορούν να αναλάβουν την ανίχνευση διαφορετικών αντικειμένων. Στο παράδειγμα του Σχήματος 2.21 το ένα οριζόντιο ορθογώνιο anchor έχει αναλάβει να βρει το αμάξι ενώ το κάθετο βρίσκει τον άνθρωπο.

Ακόμα πιο σημαντική όμως είναι και η επιπλέον εξειδίκευσή που μπορούμε να πετύχουμε με τα πολλαπλά anchors κατά την διάρκεια της εκπαίδευσης. Αν το πλήθος είναι k anchor boxes τότε έχουμε επί k αύξηση του μήκους του διανύσματος των προτάσεων που θα προωθήσουμε στα επόμενα στάδια. Το δίκτυο μπορεί τώρα ξεχωρίσει το πως μαθαίνει ανά anchor box. Πρακτικά κάθε anchor 'εκπαιδεύεται' διαφορετικά και μαθαίνει να ανιχνεύει συγκεκριμένου είδους αντικείμενα. Στο πάνω παράδειγμα θα μπορούσαμε να καταλήξουμε σε ένα δίκτυο όπου τα οριζόντια anchor boxes να έχουν εξειδικευτεί στην αναγνώριση αυτοκινήτων όπου και αυτά αν βρίσκονται. Μπορούμε τέλος να έχουμε εξειδικευμένα anchor boxes ανα είδος και ανά περιοχή. Για παράδειγμα τα κάθετα μακρόστενα anchors στο κάτω μέρος της εικόνας να εκπαιδευτούν εξειδικεύοντας στην αναγνώριση ανθρώπινης σιλουέτας.



Σχήμα 2.21: Διαφορετικά anchor boxes μαθαίνουν να αναγνωρίζουν διαφορετικά αντικείμενα

Τώρα που έχουμε ένα τρόπο να παράγουμε βασικές προβλέψεις μπορούμε να χρησιμοποιήσουμε τις παραπάνω τεχνικές της καταστολής μη μεγίστων και τη βελτίωση των bounding box περνώντας τα anchor boxes μέσα από τους bounding box regressors για να πάρουμε τα τελικά ορθογώνια.

Βέβαια εδώ δημιουργείται το εξής πρόβλημα. Δεν μπορούμε να απαιτούμε από κάποιον bounding box regressor να βελτιώσει ένα anchor box για να προβλέψει σωστά την ύπαρξη αντικειμένου που βρίσκεται πολύ μακριά από το ίδιο το κέντρο του anchor box. Μάλιστα κατα την εκπαίδευση, χρειάζεται να αποφύγουμε εντελώς την επίδραση ground truth box αντικειμένων σε anchors boxes που βρίσκονται πολύ μακριά από αυτά! Αν δεν το κάνουμε αυτό τότε το πιο πιθανό είναι το μοντέλο μας να μην συγκλίνει ποτέ.

Το παραπάνω πρόβλημα λύνεται με το να υιοθετήσουμε κάποια **στρατηγική ταιριάσματος των ground truth boxes** που χρησιμοποιούμε στην εκπαίδευση με τα anchor boxes. Ένας εύλογος τρόπος είναι μέσω της χρήσης της μετρικής IoU με τρόπο όπως ο παρακάτω (η μέθοδος στο Fast R-CNN μοντέλο). Επιτρέπουμε σε ένα anchor box να μάθει από ένα ground truth box μόνο αν το IoU είναι άνω μιας τιμής έστω θ_1 .Αν το anchor box έχει IoU με τιμή ανάμεσα σε $\theta_1 < U < \theta_2$ όπου θ_1 μια κατώτερη τιμή,τότε αγνοούμε τελείως την εκπαίδευση για τις παραμέτρους του anchor box για αυτό το παράδειγμα. Τέλος αν $IoU < \theta_1$ τότε αφήνουμε να γίνει εκπαίδευση θεωρώντας το αντικείμενο πρόβλεψης να ανήκει στην κλάση background.

Η παραπάνω προσέγγιση μας αφήνει συνήθως με ένα υπερβολικά μεγάλο πλήθος από αρνητικά παραδείγματα, δηλαδή με κλάση να είναι background και ένα μικρό πλήθος από θετικά (οποιαδήποτε άλλη κλάση αντικειμένου εκτός background) . Δυστυχώς αυτό είναι καταστροφικό γιατί πρόκειται σε επόμενο στάδιο να προωθήσουμε αυτά τα παραδείγματα/anchor boxes που περάσαν το παραπάνω φιλτράρισμα σε trainable classifiers , για να βρούνε τις κλάσεις ανά anchor box. Αν σε κάθε δείγμα από τα δεδομένα εκπαίδευσης έχουμε αυτήν την αναλογία οι classifiers αυτοί θα εκπαιδεύονται κυρίως με instances της κλάσης background οπότε και θα είναι προκατειλημμένοι στο να προβλέπουν αυτή.

Η λύση στο πρόβλημα είναι να χρησιμοποιήσουμε άλλη μια στρατηγική που την ονομάζουμε δειγματοληψία αρνητικών (**Negative Sampling**). Μπορούμε κατά την εκπαίδευση να προωθούμε στο επόμενο στάδιο όλα τα θετικά συν κάποια, επιλεγμένα τυχαία, αρνητικά παραδείγματα ώστε η αναλογία τους να είναι σταθερή (πχ 1/3 για Mask R-CNN). Υπάρχουν και άλλες στρατηγικές όπως η προώθηση όλων των αρνητικών παραδειγμάτων αλλά η χρήση για την εκπαίδευση ,μόνο αυτών που δίνουν το μεγαλύτερο σφάλμα στους ταξινομητές .

2.5 Επίλογος Θεωρητικού Υποβάθρου

Οι παραπάνω ενότητες αποτελούν τα αναγκαία σκαλοπάτια που πρέπει να περάσει κάποιος για να καταλάβει την επόμενη ενότητα. Είναι φυσικό ότι αρκετές από τις έννοιες ίσως περιγράφηκαν βιαστικά ή και ελλιπώς καθώς το βάθος της ανάλυσης αυτών αποτελεί αντικείμενο πολλών εργασιών και ανοιχτής έρευνας μέχρι και σήμερα. Παρά αυτό το γεγονός σημαντικός σκοπός της διπλωματικής αυτής είναι να δώσει εκτός της πειραματικής συνεισφοράς , μια καλή και πλήρης εισαγωγή -μελέτη στον κλάδο της ανάλυσης εικόνων με τεχνικές βαθιάς μηχανικής για αυτό και θεωρήθηκε αναγκαία η συμπερίληψη των ιδεών από τη βάση τους.

Κεφάλαιο 3

Μοντέλα βαθιάς μάθησης για ανίχνευση αντικειμένων και ταξινόμηση εικόνας

Σε αυτό το σημείο θα περιγράψουμε τα διάφορα μοντέλα νευρωνικών δικτύων που θα χρησιμοποιηθούν πειραματικά για την ανίχνευση και ταξινόμηση δερματικών σπίλων (ελιών). Θα περιγραφεί η αρχιτεκτονική ο τρόπος μάθησης από τα δεδομένα και τυχόν καινοτομίες,παραλλαγές και προσθήκες στο αρχικό δίκτυο. Θα αρχίσουμε με την περιγραφή του **Faster R-CNN** με σκοπό να μπορέσουμε να εξηγήσουμε έπειτα το μοντέλο **Mask R-CNN** που αποτελεί την επέκταση του. Αυτό το μοντέλο θα το χρησιμοποιήσουμε για την ανίχνευση ενώ έπειτα θα εξηγήσουμε αναλυτικα τα μοντέλα **ResNet** τα οποία θα χρησιμοποιήσουμε τώρα για καθαρή ταξινόμηση. Τα ResNet φέρνουν νέες ιδέες στον τρόπο που χτίζουμε συνελικτικά δίκτυα. Τέλος θα αναφερθούμε συνοπτικά σε ένα νέο (2019) μοντέλο για segmentation, το **DeepLab** από τη Google με σκοπό να το συγκρίνουμε με το Mask R-CNN.

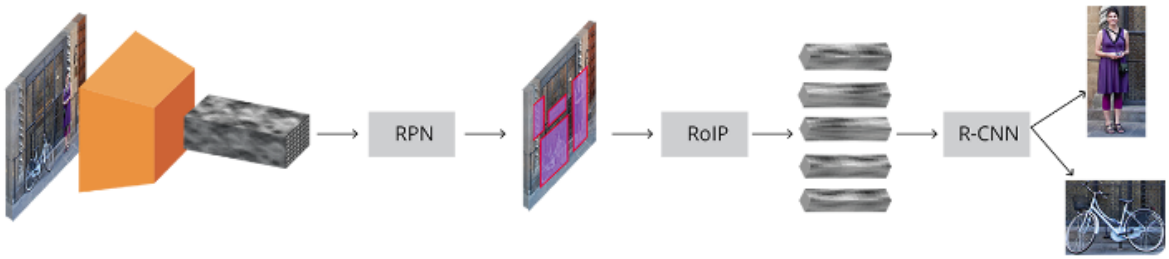
3.1 Faster R-CNN

Αυτό το μοντέλο δέχεται στην είσοδο μια εικόνα και στην έξοδο βγάζει πιθανές προβλέψεις αντικειμένων που ανήκουν σε συγκεκριμένες κλάσεις καθώς και τη θέση τους στην εικόνα δίνοντάς το αντίστοιχο ορθογώνιο οριοθέτησης (bounding box). Μέχρι και σήμερα αποτελεί μια πολύ καλή επιλογή για ανίχνευση αντικειμένων αν όχι state of the art με κάποιες βελτιώσεις και επεκτάσεις και χρησιμοποιείται αρκετά. Επίσης κυκλοφορούν διάφορες υλοποιήσεις του, ανοιχτού κώδικα με πρποσφερόμενες αρχικοποιήσεις για να μην αναγκαστεί κάποιος να το εκπαιδεύσει από την αρχή μιας και είναι υπολογιστικά βαρύ να γίνει αυτό.

Το Faster R-CNN [Ren15] είναι προϊόν εξέλιξης του Fast R-CNN [Girs15] που προέκυψε από το R-CNN [Girs13] στην προσπάθεια να αντικατασταθούν τα στάδια που έφεραν την μεγαλύτερη καθυστέρηση στο pipeline. Η περιγραφή του θεωρήθηκε σκόπιμη μιας και τα συστατικά του εμπεριέχονται ίδια ή με μετεξελίξεις στο δίκτυο Mask R-CNN που μας ενδιαφέρει.

3.1.1 Η αρχιτεκτονική του Faster R-CNN

Το Faster R-CNN είναι ένα πολύπλοκο μοντέλο γιατί αποτελείται από πολλά στάδια. Η αρχιτεκτονική του φαίνεται στο Σχήμα 3.1. Τα επιμέρους τμήματα αποτελούνται από ένα βασικό δίκτυο **Backbone CNN** το οποίο χρησιμοποιείται για εξαγωγή χαρακτηριστικών ενώ ακολουθεί το δίκτυο **RPN** που δημιουργεί τις προτάσεις για πιθανά αντικείμενα χρησιμοποιώντας anchor boxes. Έπειτα έχουμε το στάδιο **Region of Interest Pooling** όπου εξάγουμε για κάθε πρόταση το τμήμα που της αντιστοιχεί από το feature volume του CNN και τέλος το **R-CNN** όπου γίνεται η ταξινόμιση για την εύρεση της κλάσης του αντικειμένου και η τελική βελτίωση του bounding box.



Σχήμα 3.1: Το δίκτυο Faster R-CNN

3.1.2 Το βασικό συνελικτικό δίκτυο (Backbone CNN)

Το πρώτο βήμα στην σειρά επεξεργασίας είναι ένα συνελικτικό δίκτυο το οποίο έχει αρχικά εκπαιδευτεί για ταξινόμηση. Συγκεκριμένα αυτό που θέλουμε είναι η έξοδος κάποιο ενδιάμεσου συνελικτικού στρώματος του δικτύου αυτού. Δηλαδή η έξοδος πρίν περάσουμε τα συνελικτικά και πάμε στα πλήρως συνδεδεμένα στρώματα οπού κωδικοποιείται η ίδια η ταξινόμηση. Αυτό που μας ενδιαφέρει εδώ είναι ο χάρτης χαρακτηριστικών ή καλύτερα ο όγκος χαρακτηριστικών (feature volume) που δημιούργησε το συνελικτικό δίκτυο από την αρχική εικόνα σε εκείνο το στρώμα.

Για να γίνει καλύτερα κατανοητό αυτό το στάδιο πρέπει να θυμηθούμε τι είναι ο όγκος χαρακτηριστικών. Όπως εξηγήθηκε στο προηγούμενο κεφάλαιο καθώς η εικόνα περνάει από τα συνελικτικά στρώματα έχουμε μείωση των διαστάσεων του μήκους και πλάτος και αύξηση του βάθους. Στο βάθος αυτό κωδικοποιούνται τα διάφορα χαρακτηριστικά που ανιχνεύονται ενώ οι χωρικές διαστάσεις κρατάνε την σχετική θέση αυτων των χαρακτηριστικών πάνω στην αρχική εικόνα. Αν δηλαδή η έξοδος από το ενδιάμεσο στάδιο είναι διαστάσεων AxBxΓ τότε οι νευρώνες στις θέσεις $(,,)$ και $(',',)$ "φωτίζουν" όταν ανιχνευθεί το ίδιο χαρακτηριστικό απλά βλέπουν διαφορετικό μέρος της αρχικής εικόνας. Το stride και τα στρώματα pooling μειώνουν τις χωρικές διαστάσεις ενώ το πλήθος των φίλτρων ορίζει το βάθος της εξόδου.

Η ιδέα αυτή της προ επεξεργασίας της εικόνας μέσω των συνελικτικών στρωμάτων είναι σπουδαίας σημασίας και ο βασικός λόγος που μπορούμε να εκπαιδεύουμε δίκτυα όπως το Mask R-CNN. Αρχικά παρατηρούμε ότι το προ-εκπαιδευμένο συνελικτικό δίκτυο έχει εκπαιδευτεί πάνω στις ίδιες (ή σε κοντινές) κλάσεις που ανήκουν τα αντικείμενα που θέλουμε να ανιχνεύσουμε. Εφόσον αυτό ισχύει έχουμε τότε ήδη μια παραμετροποίηση που είναι ικανή να ανιχνεύει χαρακτηριστικά πάνω στην εικόνα που υποδεικνύουν επομένως την ύπαρξη τέτοιων αντικειμένων. Αυτή η αρχικοποίηση του βασικού συνελικτικού δικτύου οδηγεί σε γρηγορότερη σύγκλιση στα βάρη του ολικού δικτύου. Βέβαια εδώ τίθεται το ερώτημα ποιο στρώμα θα διαλέξουμε για πάρουμε τον όγκο χαρακτηριστικών του. Έχουμε ήδη δει ότι όσο πιο βαθιά πάμε τόσο πιο μεγάλο είναι το πεδίο οράσης (receptive field) των νευρώνων του στρώματος και τόσο πιο περίπλοκα είναι τα χαρακτηριστικά που τους ενεργοποιούν. Επίσης όσο πιο βαθιά τόσο μικρότερες είναι οι χωρικές διαστάσεις και αυτό μας βολεύει υπολογιστικά μιας και στο επόμενο βήμα εξάγουμε προδιαγεγραμμένες προτάσεις για κάθε διαφορετικό σημείο. Από την άλλη όσο πιο βαθιά πάμε κινδυνεύουμε να χάσουμε την γενικότητα που αναγνωρίζουνε τα πρώτα στρώματα και να μπλέξουμε με αρκετά εξειδικευμένα χαρακτηριστικά τα οποία ίσως δεν ανταποκρίνονται στις απαιτήσεις μας ως προς τι είδους αντικείμενα θέλουμε να ανιχνεύσουμε.

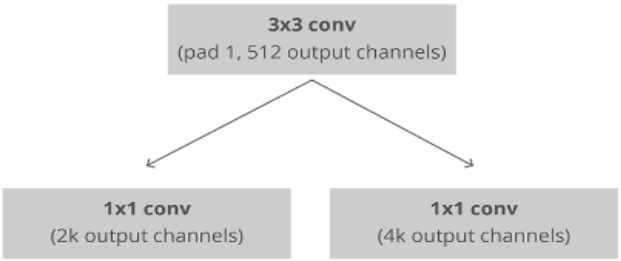
Δεν υπάρχει κάποιος απλός κανόνας για να επιλέγουμε σε ποιο στρώμα θα πάρουμε την έξοδο. Τουλάχιστον όχι κάποιος που προκύπτει από κάποια θεωρητική μελέτη. Στην αρχική μελέτη που παρουσιάστηκε το μοντέλο χρησιμοποιήθηκε ένα δίκτυο που λέγεται VGG ενώ από τότε συνήθως χρησιμοποιούνται οι διάφορες εκδοχές του ResNet το οποίο θα παρουσιαστεί σε δική του ενότητα.

Τελικά λοιπόν από αυτό το στάδιο θα πάρουμε μια επεξεργασμένη μορφή της εικόνας την οποία θα προωθήσουμε στο επόμενο στάδιο για τη δημιουργία προτάσεων. Πρέπει να σημειωθεί εδώ ότι εφόσον πέρνουμε την έξοδο από κάποιο ενδιάμεσο συνελικτικό στρώμα δεν είμαστε υποχρεωμένοι να δίνουμε στην είσοδο του δικτύου εικόνες σταθερών διαστάσεων. Αυτό που θα συμβεί από την παραπάνω ανάλυση είναι απλά να μετασχηματιστούν οι διαστάσεις της εικόνας από $(w, h, 3)$ σε $(w/c, h/c, depth)$ όπου το c εξαρτάται από το stride των συνελικτικών στρωμάτων και τα pooling layers. Βέβαια όπως θα δούμε επειδή οι προτάσεις παράγονται με τη χρήση anchor boxes τα οποία είναι σταθερού μεγέθους, μια εικόνα πολυ μεγάλων διαστάσεων θα οδηγήσει σε αδυναμία εύρεσης μεγάλων αντικειμένων ενώ μικρών διαστάσεων σε αδυναμία εύρεσης μικρών αντικειμένων. Φυσικά μπορούμε να διορθώσουμε αυτό το πρόβλημα είτε προσαρμόζοντας τις υπέρ-παραμέτρους των διαστάσεων των anchor boxes αναλόγως το μέγεθος των εικόνων του προβλήματος μας ,είτε προκαλώντας κάποια κλιμάκωση στις ίδιες τις εικόνες πριν τις προωθήσουμε στο δίκτυο.

3.1.3 Το δίκτυο προτάσεων περιοχών (Regional Proposal Network)

Αφού πάρουμε την έξοδο από το backbone CNN δίκτυο θέλουμε για κάθε σημείο των χωρικών διαστάσεων (w, h) του όγκου διάστασεων WxHxD να δημιουργήσουμε προτάσεις μέσα από anchor boxes, τα οποία εξηγήθηκαν πιο αναλυτικά στο Κεφάλαιο 2. Το Region Proposal Network ή εν συντομία RPN κάνει ακριβώς αυτό και μάλιστα το πετυχαίνει μέσω ενός πλήρους συνελικτικού τρόπου.

Το RPN αποτελείται αρχικά από ένα συνελικτικό στρώμα με φίλτα 3x3xd όπου d το βάθος της εξόδου του backbone cnn ενώ χρησιμοποιείται stride ίσο με 1 και padding ίσο με 1.Τέλος χρησιμοποιούντε 512 τέτοια φίλτρα για να προκύψει μια έξοδος με χωρικές διαστάσεις ίσες με την είσοδο και βάθος ίσο με 512. Αυτή η έξοδος ,που στην ουσία αποτελεί μια προ-επεξεργασία του όγκου χαρακτηριστικών ,περνάει έπειτα από δύο διαφορετικά παράλληλα δίκτυα.



Σχήμα 3.2: Regional Proposal Network

Το πρώτο δίκτυο αποτελείται από ένα συνελικτικό στρώμα με 2k φίλτρα διαστάσεων 1x1x512 ,όπου k είναι το πλήθος των anchor boxes που θέλουμε να παράγουμε ανά σημείο. Στην αρχική μελέτη οι δημιουργοί του μοντέλου δώσανε στο k την τιμή 9. Για να κατανοήσουμε τη λειτουργία αυτού του στρώματος πρέπει να παρατηρήσουμε ότι θέλουμε για κάθε ξεχωριστό σημείο να παράγουμε k anchor boxes και να δώσουμε σε αυτά τα anchor boxes δύο διαφορετικές πιθανότητες. Μια πιθανότητα για το ενδεχόμενο το anchor box να περιέχει κάποιο αντικείμενο και μια να περιέχει παρασκήνιο. Έτσι προκύπτει ο παράγοντας δύο. Το αποτέλεσμα αυτού του δικτύου λοιπόν μπορούμε να το δούμε ως πολλαπλα διανύσματα $(p_1, q_1, p_2, q_2, .., p_k, q_k)$ ένα για κάθε σημείο (x, y) και όπου p_i η πιθανότητα να είναι αντικείμενο και q_i η πιθανότητα να είναι παρασκήνιο για το i-οστό anchor. Αυτα τα anchor boxes μπορεί να υπολογίζονται από τον χάρτη αλλά έχουν άμεση αντιστοίχηση με την αρχική εικόνα. Συγκεκριμένα αν μετά το backbone έχουμε μείωση των χωρικών διαστάσεων από width και height σε $\frac{width}{c}$ και $\frac{height}{c}$ μπορούμε να θεωρήσουμε ότι χωρίζουμε την εικόνα ομοιόμορφα σε $\frac{width*height}{c^2}$ κελιά στα οποία στο κέντρο τους σχεδιάζουμε k κουτιά διαφόρων διαστάσεων και σχημάτων.

Το δεύτερο δίκτυο αντίστοιχα αποτελείται από ένα πάλι συνελικτικό στρώμα στο οποίο έχουμε 4k φίλτρα διαστάσεων 1x1x512. Όμοια με πριν θέλουμε για κάθε χωρικό σημείο και για κάθε anchor box του να πάρουμε τώρα τέσσερις τιμές που αντιπροσωπεύουν αυτή τη φορά τις αποκλίσεις στο κέντρο του achor box $(\Delta X_{center}, \Delta Y_{center})$ και τις αποκλίσεις στο πλάτος και ύψος του $(\Delta Width, \Delta Height)$. Πρακτικά το δίκτυο αυτό είναι ένας bounding box regressor οπως είχαμε αναφέρει στο προηγούμενο κεφάλαιο, ο οποίος δεν κοιτάζει την κλάση του αντικειμένου (class agnostic). Στο Σχήμα 3.2 παρουσιάζεται ένα διάγραμμα των στοιχείων του RPN.

Η έξοδος των παραπάνω δικτύων αποτελεί το σύνολο των προτάσεων για την ανίχνευση αντικειμένων. Όμως χρειάζεται επιπλέον επεξεργασία πριν την διοχετεύσουμε στο επόμενο στάδιο. Όπως αναφέρθηκε στο προηγούμενο κεφάλαιο πρόκειται να έχουμε υπερκαλύψεις από τα anchor boxes καθώς αυτά προσπαθούν να βρούνε το ίδιο αντικείμενο. Το πρόβλημα αυτό το λύνουμε εφαρμόζοντας **Non Maximum Suppression**. Ο αλγόριθμος που χρησιμοποιείται παίρνει μια λίστα με τις προτάσεις ταξινομημένες κατά πιθανότητα (την μεγαλύτερη από τις δύο που αντιστοιχούν στο να είναι αντικείμενο ή παρασκήνιο) και ακολουθιακά διατρέχει τη λίστα ξανά και ξανά αφαιρώντας τις προτάσεις που έχουν IoU μεγαλύτερο μια τιμή (συνήθως 0.6).

Επιπλέον μετά από την παραπάνω διαδικασία συνηθίζεται να έχουμε άλλο ένα φιλτράρισμα των προτάσεων από bounding boxes με το να τις ταξινομήσουμε ξανά ανά πιθανότητα και να κρατήσουμε από αυτές έναν αριθμό N από αυτές με το καλύτερο σκορ. Το N παίρνει τιμές από 2000 μέχρι και τόσο χαμηλά όσο 50.Επισημαίνεται εδώ ότι ξανά ότι στο σύνολο των προτάσεων θα υπάρχουν αρκετές η οποίες θα είναι τύπου - παρασκήνιο. Αυτές θα αφαιρεθούν σε επόμενο στάδιο.

Λίγα λόγια για την εκπαίδευση του RPN Το RPN κάνει δύο διαφορετικών ειδών προβλέψεις από τα δύο παράλληλα δίκτυα όπως παρουσιάσαμε. Η μία είναι ουσιαστικά δυαδική ταξινόμιση για να βρει αν το anchor box περιέχει αντικείμενο ή όχι και η άλλη είναι bounding box regression για να βρει πως το anchor box πρέπει να αλλάξει για να καλύπτει μεγαλύτερο μέρος του αντικειμένου. Για αυτό και χρειαζόμαστε δυο διαφορετικές συναρτήσεις σφάλματος για την εκπαίδευση τις οποίες μπορούμε να αθροίσουμε απλώς ή σταθμισμένα και να εφαρμόσουμε τον αλγόριθμο ανάστροφης διάδοσης.Τα δύο παράλληλα δίκτυα θα έχουν παραγώγους από την αντίστοιχη συνάρτηση σφάλματος τους ενώ το κοινό δίκτυο από το άθροισμα τους.

Αρχικά πρέπει να παρατηρήσουμε ότι δεν μπορούμε να αφήσουμε κάθε anchor box να μάθει από κάθε ground truth box. Για να εκπαιδεύσουμε το RPN χρειαζόμαστε μια στρατηγική ταιριάσματος των anchor boxes με τα ground trudth boxes των δεδομένων εκπαίδευσης. Στην διαδικασία αυτή έχουμε αναφερθεί και σε προηγούμενο κεφάλαιο και είναι απαραίτητη για κάθε μοντέλο που χρησιμοποιεί anchor boxes για να παράγει προτάσεις. Η στρατηγική που υιοθέτησαν οι δημιουργοί του μοντέλου είναι η εξής. Χρησιμοποιούμε την μετρική IoU για να δούμε αν τα anchors υπερκαλύπτουν κάποιο ground truth box. Αν η κάλυψη με κάποιο τέτοιο ground truth box δίνει IoU μεγαλύτερο του 0.5 θεωρούμε ότι έχουμε αντικείμενο και αφήνουμε αυτό το anchor box να μάθει από αυτό το box. Αν 0.1< IoU <0.5 θεωρούμε ότι είναι απροσδιόριστο και δεν αφήνουμε το anchor box να μάθει από αυτό το ορθογώνιο απορρίπτοντας αυτήν την πρόταση. Τέλος αν IoU <0.1 αφήνουμε το anchor box να μάθει θεωρώντας ότι περιέχει την κλάση - παρασκήνιο.

Η παραπάνω διαδικασία μας αφήνει με μεγάλο αριθμό από προτάσεις παρασκήνιο σε σχέση με προτάσεις αντικειμένων και όπως εξηγήσαμε στο Κεφάλαιο 2 αυτό είναι κακό για τα επόμενα στάδια και τους ταξινομητές. Το θέμα λύνεται κάνοντας μια δειγματοληψία (**negative sampling**) και κρατώντας μόνο 256 προτάσεις (ένα mini batch) με ισορροπημένη αναλογία των δύο ειδών. Το RPN έπειτα χρησιμοποιεί όλες αυτές τις προτάσεις για να υπολογίσει το λάθος ταξινόμησης με συνάρτηση σφάλματος την διασταυρωμένη εντροπία για δύο κλάσεις.
Για το τμήμα που κάνει το box regression είχαμε αναφέρει ότι συνήθως χρησιμοποιείται η L2 νόρμα που υπολογίζεται από τα διανύσματα της πρόβλεψης του ορθογωνίου και του πραγματικού ορθογωνίου. Για το RPN επιλέγεται όμως η smooth L1 η οποία είναι η L1 μόνο που για τιμές κάτω από μια σταθερά c δίνουμε μικρότερο λάθος από την καθαρή L1.

Συγκεκριμένα η συνάρτηση σφάλματος είναι η παρακάτω.

$$L(p_i, t_i) = \frac{\sum_i L_{cls}(p_i, p_i^*)}{N_{cls}} + l * \frac{\sum_i p_i^* L_{reg}(t_i, t_i^*)}{N_{reg}}$$

Εδώ i είναι ο δείκτης του anchor box που περιέχεται στο mini batch των 256 ενώ p_i η πιθανότητα να περιέχει αντικείμενο. Το p_i^* είναι 1 αν το anchor box ανήκει στις θετικές προτάσεις ή 0 αν είναι αρνητικό δηλαδή στις προτάσεις παρασκηνίου. Το t_i συμβολίζει τις 4 τιμές του ορθογωνίου που προβλέπεται από το anchor box ενώ t_i^* είναι οι τιμές του ορθογωνίου με το οποίο σχετίστηκε η συγκεκριμένη πρόταση. Το λάθος ταξινόμησης L_{cls} (διασταυρωμένη εντροπία 2 κλάσεων) είναι αναλυτικά.

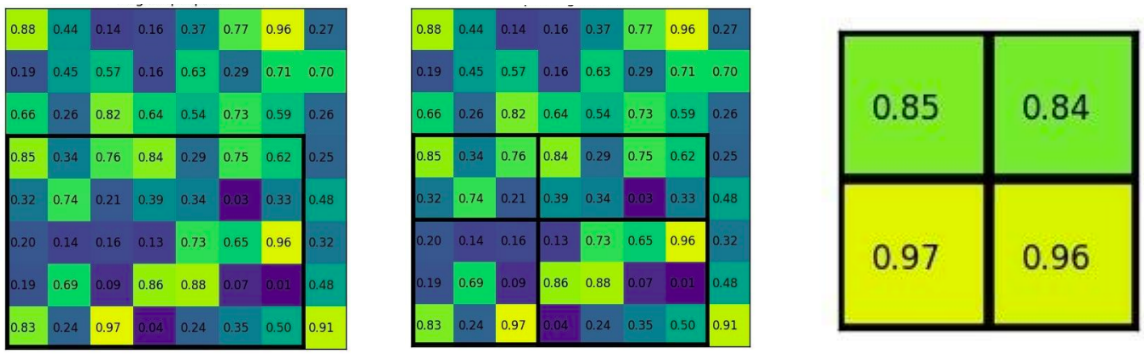
$$L_{cls}(p_i, p_i^*) = -p_i^* log(p_i) - (1 - p_i^*) log(1 - p_i)$$

Ακόμα με L_{reg} συμβολίζουμε το λάθος από τις αποκλίσεις των διαστάσεων δηλαδή την εφαρμογή της smooth L1 νόρμας. Τέλος με N_{cls} συμβολίζουμε τον παράγοντα κανονικοποίησης για την ταξινόμηση και είναι ίσος με το mini batch δηλαδή 256, ενώ με N_{reg} τον παράγοντα κανονικοποίησης για το box regression και συνήθως είναι ίσο με τον αριθμό των διαφορετικών κέντρων των anchor boxes στην εικόνα. Το l είναι ένας έξτρα παράγοντας που μας επιτρέπει να δίνουμε διαφορετική αναλογικά αξία στο μέρος της ταξινόμησης και στο μέρος της πρόβλεψης ορθογωνίου ανάλογα με το τι επιθυμούμε για το πρόβλημά μας. Τέλος υπενθυμίζουμε ότι για τις 4 τιμές του ορθογωνίου $t = (t_x, t_y, t_{width}, t_{height})$ οι οποίες μπορούν να υπολογιστούν εύκολα μέσα από την κλασική κωδικοποίηση των δύο ακριανών γωνιών τα t_{width} και t_{height} είναι οι λογάριθμοι των πραγματικών τιμών του πλάτους και του ύψους.

Η παρουσίαση του RPN τελειώνει εδώ. Μάλιστα άμα θέλουμε να αναγνωρίσουμε αντικείμενα αποκλειστικά μιας κλάσης θα μπορούσαμε να τελειώσουμε εδώ τη μελέτη του δικτύου. Το RPN μπορεί να εξειδικευτεί πλήρως στην αναγνώριση μιας κλάσης και στην βελτίωση των ορθογωνίων και έτσι να πάρουμε έξοδο αποκλειστικά από αυτό για το πρόβλημα μας.
Τα επόμενα στάδια αφιερώνονται στην εύρεση της κλάσης του αντικειμένου και στην περαιτέρω βελτίωση του ορθογωνίου οριοθέτησης.

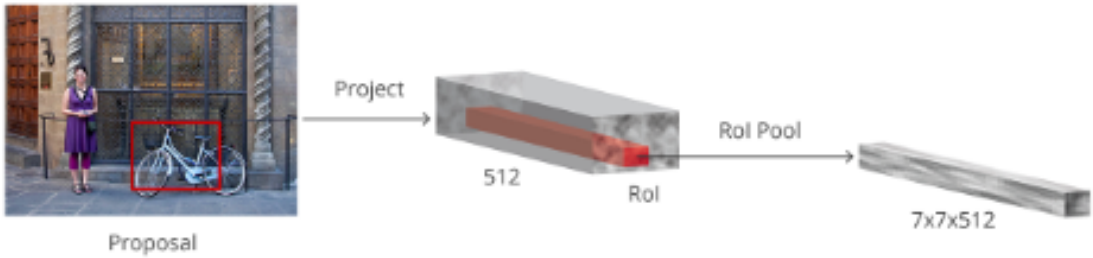
3.1.4 Συγκέντρωση περιοχών ενδιαφέροντος (Region of Interest Pooling)

Από τα παραπάνω στάδια έχουμε τώρα συγκεντρώσει διάφορες προτάσεις για τις οποίες θέλουμε να βρούμε την κλάση που ανήκουν καθώς και να βελτιώσουμε περαιτέρω τα ορθογώνια οριοθέτησης τους. Μια απλοϊκή προσέγγιση θα να πάρουμε αυτές τις προτάσεις να κόψουμε αναλόγως τα τμήματα στα οποία αντιστοιχούν από την αρχική εικόνα και να τα περάσουμε από το αρχικό βασικό δίκτυο ή κάποιο άλλο για ταξινόμηση και βελτίωση ορθογωνίων. Κάτι τέτοιο δεν χρειάζεται αφού μπορούμε να χρησιμοποιήσουμε τον όγκο χαρακτηριστικών που έχει ήδη υπολογιστεί από το backbone δίκτυο. Το Faster R-CNN χρησιμοποιεί την έξοδο του backbone για να εξάγει για κάθε πρόταση σταθερού μεγέθους χάρτες χαρακτηριστικών χρησιμοποιώντας Region of Interest Pooling. Συγκεκριμένα θέλουμε για κάθε πρόταση να πάρουμε σταθερό όγκο χαρακτηριστικών συνήθως μεγέθους 7x7xd όπου d το βάθος της εξόδου του backbone δικτύου. Μπορούμε να δούμε το RoIP ως ένα έξτρα στρώμα που εκτελεί τις παρακάτω λειτουργίες.
1) Δέχεται την έξοδο του backbone δικτύου μας.
2) Δέχεται για κάθε πρόταση ένα διάνυσμα 5 τιμών, μια για να κρατήσει δείκτη για την πρόταση και τέσσερις για να προσδιοριστεί το bounding box της.
Χρησιμοποιώντας την παραπάνω πληροφορία παίρνει ένα τμήμα του χάρτη χαρακτηριστικών της εξόδου του backbone που αναλογεί στις διαστάσεις του ορθογωνίου, χωρίζει αυτό το τμήμα ομοιόμορφα σε 7x7 κελιά/υπό-περιοχές ή όσα θέλουμε ο νέος όγκος χαρακτηριστικών να έχει, και έπειτα επιλέγει τις μέγιστες τιμές μέσα σε αυτές τις υποπεριοχές για να φτιάξει τον 7x7xd όγκο χαρακτηριστικών για κάθε πρόταση που θα διοχετευτεί στο R-CNN. Στο Σχήμα 3.3 φαίνεται η διαδικασία στην περίπτωση που έχουμε d=1 και τελικές χωρικές διαστάσεις 2x2 για λόγους απλότητας.



Σχήμα 3.3: Υπολογισμός RoIP για 2x2x1

Από την πρώτη εμφάνιση του μοντέλου μέχρι τώρα, έχουν αρχίσει να χρησιμοποιούνται άλλες μέθοδοι για να εξάγουμε τους χάρτες χαρακτηριστικών για τα επόμενα επίπεδα. Ένας άλλος τρόπος είναι με το να κόψουμε πάλι την περιοχή της εξόδου του backone δικτύου που αντιστοιχεί στην πρόταση που επεξεργαζόμαστε, αλλά αυτή τη φορά να την μετατρέψουμε σε 14x14xdepth με τη χρήση παρεμβολής. Έπειτα μπορούμε να περάσουμε από κάποιο άλλο στρώμα συνελικτικό με stride =2 η max pooling πάλι με stride = 2 για να πάρουμε την επιθυμητή 7x7xdepth που χρειάζεται το επόμενο στάδιο. Υπάρχουν και άλλοι τέτοιοι τρόποι και αυτό το στάδιο γενικά ποικίλλει ανάλογα την υλοποίηση που θα χρησιμοποιήσουμε για το μοντέλο. Σημειώνουμε ότι το depth του όγκου χαρακτηριστικών είναι όσο ορίζει το backbone δίκτυο και για μας είναι 512. Στο Σχήμα 3.4 φαίνεται η όλη διαδικασία.



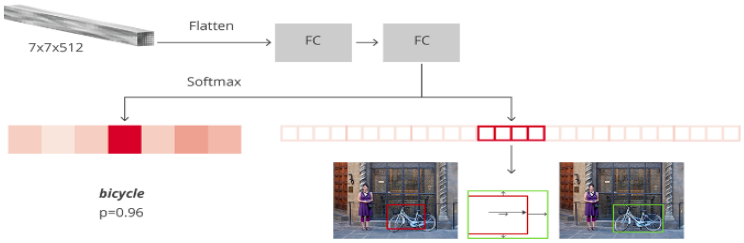
Σχήμα 3.4: Region of Interest Pooling

3.1.5 Το δίκτυο R-CNN (Region Based Convolutional Neural Network)

Αυτό είναι το τελευταίο υπό δίκτυο του Faster R-CNN. Έχοντάς από την παραπάνω διαδικασία πάρει το τμήμα του όγκου χαρακτηριστικών που αντιστοιχεί σε κάθε πρόταση μπορούμε να το χρησιμοποιήσουμε για να να εκτελέσουμε ταξινόμηση και να βρούμε την κλάση του αντικειμένου. Επίσης δεδομένης της κλάσης μπορούμε να βελτιώσουμε ακόμα περισσότερο το bounding box. Αυτή τη λειτουργία επιτελεί το R-CNN.

Το R-CNN δέχεται κάθε όγκο χαρακτηριστικών για κάθε πρόταση ξεχωριστά δηλαδή στην περίπτωση μας είσοδο διαστάσεων 7x7x512. Έπειτα την χρησιμοποιεί δύο πλήρως συνδεδεμένα στρώματα μεγέθους 4096 νευρώνων το καθένα και με ReLU για συνάρτηση ενεργοποίησης. Μετά όπως και στο RPN έχουμε δύο παράλληλα δίκτυα για την ταξινόμηση και τη βελτίωση ορθογωνίου. Συγκεκριμένα έχουμε τα εξής.

Ένα πλήρως συνδεδεμένο δίκτυο με n+1 νευρώνες, δηλαδή έναν νευρώνα για κάθε κλάση αντικειμένου για n αντικείμενα και έξτρα μια κλάση για το παρασκήνιο. Το δίκτυο αυτό κωδικοποιεί τις πιθανότητες των κλάσεων για κάθε πρόταση.
Ένα πλήρως συνδεδεμένο δίκτυο με 4n νευρώνες οπού n οι διαφορετικές κλάσεις που αναγνωρίζουμε τώρα χωρίς να συμπεριλάβουμε το παρασκήνιο. Την έξοδο μπορούμε να την δούμε ως n διανύσματα τετράδων στα οποία όπως και στο regression του RPN κωδικοποιούνται οι αλλαγές στις τιμές του ορθογωνίου οριοθέτησης. Δεν έχουμε bounding box regression για το παρασκήνιο μιας και δεν έχει κάποιο νόημα. Όπως είναι φανερό και από το διάνυσμα εξόδου τώρα κάνουμε βελτιστοποίηση ορθογωνίου λαμβάνοντας υπόψιν και την κλάση του αντικειμένου μιας και παίρνουμε τετράδες για **κάθε** κλάση. Στην τελική μας πρόβλεψη θα χρησιμοποιήσουμε τελικά **μόνο** την τετράδα που αντιστοιχεί στην κλάση της πρόβλεψης για να υπολογίσουμε το τελικό ορθογώνιο και θα αγνοήσουμε τις άλλες. Αυτό το γεγονός πρέπει να το σεβαστούμε και κατά την διάρκεια της εκπαίδευσης. Στο Σχήμα 3.5 έχουμε την αρχιτεκτονική όλου του δικτύου.



Σχήμα 3.5: Region Based Convolutional Neural Network

Αφού περάσουμε όλες τις προτάσεις από το R-CNN μένουν μόνο μερικά βήματα πριν την τελική έξοδο του δικτύου. Πρώτα πρέπει να απορρίψουμε εντελώς τις προτάσεις που αντιστοιχούν σε παρασκήνιο μιας και δεν μας ενδιαφέρουν. Έπειτα πρέπει να αφαιρέσουμε πολλαπλές προτάσεις που αντιστοιχούν στο ίδιο αντικείμενα. Αυτό το πετυχαίνουμε εκτελώντας Non Maximum Suppression αυτή τη φορά ανά κλάση όμως. Ομαδοποιούμε δηλαδή της προτάσεις ανα κλάση πριν εκτελέσουμε τον αλγόριθμο ξεχωριστά για κάθε ομάδα.

Λίγα λόγια για την εκπαίδευση του R-CNN Το R-CNN εκπαιδεύεται παρόμοια με το RPN κάποιες μικρές αλλά σοβαρές όμως διαφοροποιήσεις. Πάλι θέλουμε κάποια στρατηγική ταιριάσματος bounding box και χρησιμοποιούμε την μετρική IoU για αυτό. Από τις προτάσεις που έφτασαν σε αυτό το στάδιο θα χρησιμοποιηθούν για εκπαίδευση μόνο αυτές που έχουν IoU μεγαλύτερο από 0.5 με κάποιο ground truth box οι οποίες και γίνονται assign σε αυτό box. Σε αντίθεση με το RPN όσες προτάσεις έχουν IoU μετξαύ 0.1 και 0.5 θεωρούνται παρασκήνιο ενώ οι υπόλοιπες αγνοούνται εντελώς. Αυτή η διαφορά υπάρχει γιατί σε αυτό το στάδιο θέλουμε ακόμα πιο αυστηρά κριτήρια για τις επιλογές μας μιας και είναι το τέλος του δικτύου οπότε και χρειαζόμαστε μέγιστη ακρίβεια. Αφού ταιριάξουμε προτάσεις με ground truth boxes κάνουμε πάλι μια δειγματοληψία ώστε να έχουμε 3/4 περίπου από προτάσεις παρασκήνιο και 1/4 προτάσεις αντικειμένων.

Οι συναρτήσεις σφάλματος είναι και πάλι διασταυρωμένη εντροπία πολλών όμως κλάσεων για τους ταξινομητές που υπολογίζεται για όλες τις προτάσεις και Smooth L1 για το bounding box regression που υπολογίζεται μόνο για τις προτάσεις που αντιστοιχούν σε αντικείμενα. Η διαφορά εδώ είναι η εξής. Όταν υπολογίζουμε το λάθος για το bounding box regression, που γίνεται από τις διανυσματικές αναπαραστάσεις του ορθογωνίου οριοθέτησης και του ground truth ,πρέπει να λαμβάνουμε υπόψιν την πρόβλεψη μόνο για την αντίστοιχη κλάση αντικειμένου που προβλέπει ο ταξινομητής, δηλαδή τις αντίστοιχες 4 τιμές μόνο του διανύσματος 4n στοιχείων και όχι όλες.

3.1.6 Η εκπαίδευση του Faster R-CNN

Αφού περιγράψαμε τα συστατικά του μοντέλου μπορούμε τώρα να αναφερθούμε στην στρατηγική εκπαίδευσης. Είδαμε πως μπορεί το RPN και το R-CNN να εκπαιδευτούν ανεξάρτητα. Αρχικά αυτή ήταν και η προσέγγιση που ακολουθήθηκε. Τα δύο αυτά υπο δίκτυα εκπαιδεύονταν ανεξάρτητα και μετά ενώνονταν για να γίνει η αξιολόγηση του δικτύου. Έπειτα πειραματικά φάνηκε πως η εκπαίδευση όλου του μοντέλου ταυτόχρονα από άκρη σε άκρη είναι αρκετά πιο γρήγορη στο να φέρει μια επιθυμητή παραμετροποίηση.

Για να εκπαιδεύσουμε ταυτόχρονα όλο το δίκτυο απλά ενώνουμε τις δύο προαναφερθέντες συναρτήσεις λαθών για τα RPN και R-CNN (συνολικά άθροισμα 2+2=4 όρων) και αφήνουμε τον αλγόριθμο ανάστροφης διάδοσης να βρει τις παραγώγους αναδρομικά προς τα πίσω. Γενικά αντί για απλό άθροισμα προτιμάται κάποιο σταθμισμένο άθροισμα των 2 συναρτήσεων σφάλματος αναλόγως σε ποιο στάδιο θέλουμε να δώσουμε περισσότερη σημασία στην εκπαίδευση. Τέλος μπορούμε να αφήσουμε την διάδοση των παραγώγων να φτάσει μέχρι και το backbone δίκτυο οπότε να το εκπαιδεύσουμε και αυτό ,εξειδικεύοντας το περισσότερο στο πρόβλημα που θέλουμε να λύσουμε. Βέβαια αυτό δεν είναι πάντα απαραίτητο ειδικά αν αναγνωρίζουμε στις εικόνες τα ίδια αντικείμενα που το backbone εκπαιδεύτηκε για να ταξινομεί. Μάλιστα μερικές φορές είναι και υπολογιστικά ασύμφορο να εκπαιδεύσουμε και το βασικό δίκτυο.

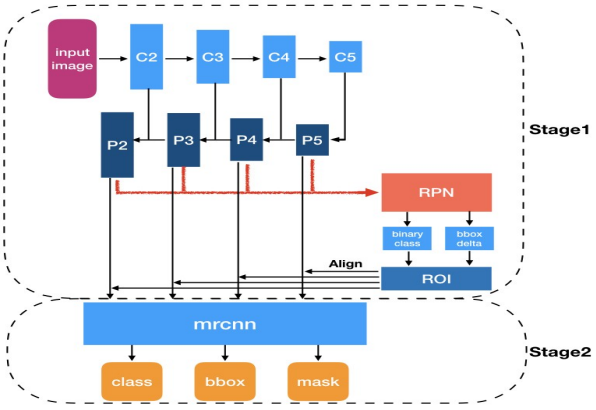
3.2 Mask R-CNN

Αυτό το μοντέλο δέχεται στην είσοδο μια εικόνα και στην έξοδο βγάζει πιθανές προβλέψεις αντικειμένων που ανήκουν σε συγκεκριμένες κλάσεις καθώς και τη θέση τους στην εικόνα δίνοντάς **όχι μόνο** το αντίστοιχο bounding box αλλά και μια **εικόνα-μάσκα**. Τα εικονοστοιχεία της μάσκας με 1 αντιστοιχούν σε εικονοστοιχεία που ανήκουν στο αντικείμενο ενώ αυτά με 0 όχι. Έτσι μπορούμε τοποθετώντας την μάσκα πάνω στην αρχική εικόνα να πάρουμε τη σιλουέτα του αντικειμένου που ανιχνεύθηκε πετυχαίνοντας καλύτερο διαχωρισμό από το background σε σχέση με το να έχουμε μόνο το bounding box του.

Το μοντέλο Mask R-CNN πρώτο-εμφανίστηκε με τη μελέτη [He17] και είναι στην ουσία το μοντέλο Faster R-CNN με την προσθήκη ενός παράλληλου υπό δικτύου στο τελευταίο στάδιο ,στο οποίο γίνεται η πρόβλεψη της μάσκας για το αντικείμενο που ανιχνεύεται από κάθε ορθογώνιο οριοθέτησης καθώς και κάποιες άλλες αλλαγές.

3.2.1 Η αρχιτεκτονική του Mask R-CNN

Η αρχιτεκτονική ολόκληρου το μοντέλου παρουσιάζεται στο Σχήμα 3.6. Τα επιμέρους υποσυστήματα θα εξηγηθούν παρακάτω.

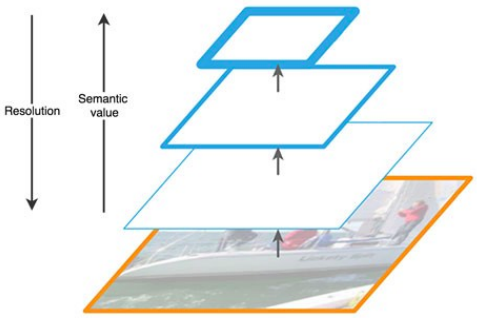


Σχήμα 3.6: Mask R-CNN overview, Ci τα στρώματα του BackBone,Pi τα στρώματα του FPN

3.2.2 Το δίκτυο πυραμίδας χαρακτηριστικών (Feature Pyramid Network)

Μέχρι στιγμής είδαμε πως το backbone δίκτυο μας αποτελείται από ένα μόνο συνελικτικό δίκτυο το οποίο κάνει feature extraction από την εικόνα. Είδαμε πως η αρχική είσοδος μετασχηματίζεται μέσα από τα συνελικτικά στρώματα, μειώνοντας σε κάθε στρώμα τις χωρικές της διαστάσεις και αυξάνοντας το βάθος μέχρι να πάρουμε ένα τελικό σύνολο από χάρτες χαρακτηριστικών στο τέλος του τελευταίου στρώματος του backbone. Αυτή τη μεταβολή των διαστάσεων μπορούμε να την δούμε και ως μια πυραμίδα με μεγάλη βάση με διαστάσεις όσο η αρχική εικόνα και καθώς αυξάνουμε το ύψος όλο και μικρότερες διαστάσεις όπως στο Σχήμα 3.7.

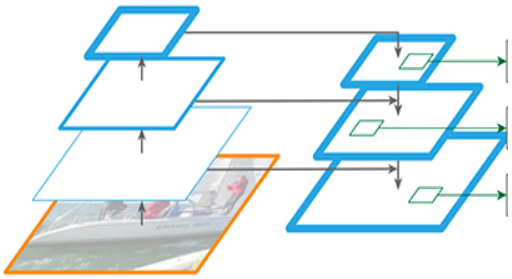
Το πρόβλημα με αυτήν την προσέγγιση ενός cnn μόνο για backbone είναι το εξής. Το δίκτυο κάνει την αναγνώριση αντικειμένων χρησιμοποιώντας την έξοδο του τελευταίου στρώματος του backbone. Κάθε νευρώνας αυτού του στρώματος έχει receptive field με κάποιο σχετικά μεγάλο και προφανώς σταθερό μέγεθος. Αν τα αντικείμενα μας έχουν διαστάσεις αρκετά μικρές τότε δεν θα μπορέσει ποτέ το δίκτυο να μάθει να τα αναγνωρίζει αφού ως μοτίβα είναι αρκετά μικρά ώστε να πιάνονται από το τελευταίο συνελικτικό στρώμα. Η επιλογή να εκπαιδεύουμε το δίκτυο χρησιμοποιώντας τις ίδιες εικόνες σε διαφορετικές κλίμακες θα βοηθούσε στο παραπάνω αλλά είναι μια ασύμφορη υπολογιστικά διαδικασία. Η επιλογή να στέλνουμε όχι μόνο το τελευταίο στρώμα στα επόμενα στάδια για αναγνώριση, αλλά όλα τα στρώματα του backbone πάλι δεν βοηθάει. Ο λόγος είναι ότι τα χαμηλά στρώματα ενός συνελικτικού δικτύου αναγνωρίζουν πολύ βασικά και απλοϊκά χαρακτηριστικά. Δεν μπορούν δηλαδή να προβλέψουν πολύπλοκα αλλά μικρά αντικείμενα γιατί δεν έχουν την ίδια σημασιολογική αξία με τα ανώτερα στρώματα. Πάλι στο Σχήμα 3.7 μπορούμε να δούμε πως όσο ανεβαίνουμε αυτήν την πυραμίδα αυξάνεται η σημασιολογική ικανότητα αλλά πέφτει η ανάλυση.



Σχήμα 3.7: Τα στρώματα του συνελικτικού βασικού δικτύου σαν πυραμίδα

Μια καλή λύση θα δόθεί πάλι από τους δημιουργους του Faster R-CNN στη μελέτη τους [Lin16]. Η ιδέα είναι η εξής. Εκτός από το κλασσικό backbone cnn που είχαμε και που προσφέρει bottom-up ανάλυση έχουμε ένα επιπλέον δίκτυο το Feature Pyramid Network ή FPN που παίρνει την έξοδο του cnn εκτελεί την αντίστροφη διαδικασία κάνωντας top-down ανάλυση. Συγκεκριμένα στο δεύτερο δίκτυο η έξοδος χάνει το βάθος της σταδιακά ενώ κάνουμε upsample για να αυξήσουμε την ανάλυση σε κάθε βήμα. Έτσι κάθε επόμενο στρώμα στο FPN έχει όλο και μεγαλύτερες χωρικές διαστάσεις ενώ λόγω της ροής δεδομένων που περνάει από τα υψηλής σημασιολογίας στρώματα του βασικού δικτύου έχει και υψηλή σημασιολογία. Στο Σχήμα 3.8 φαίνεται η διαδικασία αρκετά πιο κατανοητά.

Βέβαια τα πράματα δεν είναι τόσο απλά. Κάνοντας upsample δεν αυξάνουμε πραγματικά την ανάλυση μιας και η χωρική πληροφορία έχει ήδη χαθεί μιας και η ροή δεδομένων προέρχεται αποκλειστικά από το τελευταίο στρώμα του backbone. Για αυτό και εντέλει προσθέτουμε έξτρα συνδέσεις μεταξύ των στρωμάτων του βασικού cnn και των αντίστοιχων στρωμάτων του FPN. Αυτες οι συνδέσεις είναι οι παράλληλες γραμμές στο Σχήμα 3.8. Οι συνδέσεις αυτές επίσης βοηθάνε και στην επιτάχυνση της διαδικασίας της εκπαίδευσης με παρόμοιο τρόπο όπως συμβαίνει στα ResNet δίκτυα που επίσης τις χρησιμοποιούν και θα εξηγηθούν αργότερα.



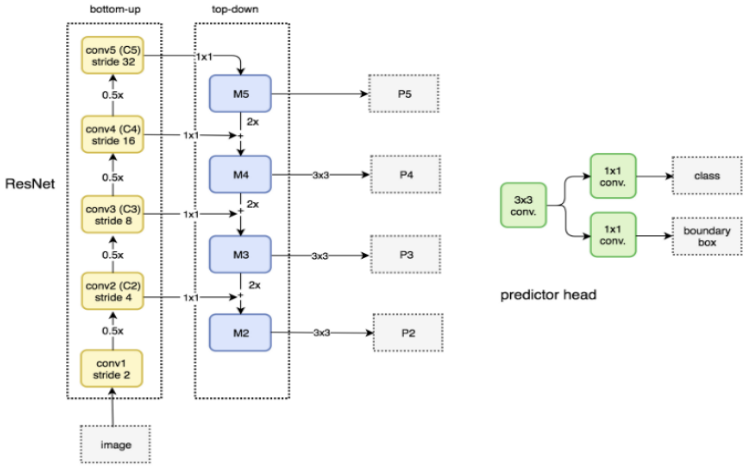
Σχήμα 3.8: Η δεύτερη πυραμίδα του FPN και οι παράλληλες συνδέσεις με την πρώτη πυραμίδα

Το πως γίνεται ακριβώς το upsample και η μετατροπή του βάθους είναι αρκετά τεχνική πληροφορία, για αυτό και δεν θα μπούμε σε περαιτέρω λεπτομέρειες. Γενικά όμως η υπερδειγματοληψία πρέπει να αναιρεί την μείωση των χωρικών διαστάσεων που γίνεται στο βασικό backbone cnn δίκτυο. Μετά από αυτό προωθούμε στο στάδιο του RPN όλα τα στρώματα της δεύτερης top-down πυραμιδας αντί για το ανώτερο που θα συναίβενε αν δεν είχαμε το FPN. Αυτό οδηγεί σε κάποιες αλλαγές στη λειτουργία του RPN.

3.2.3 Αλλαγές του RPN λόγω χρήσης FPN

Το RPN ήταν το δίκτυο που δημιουργούσε προτάσεις μέσω anchor boxes για κάθε χωρικό σημείο στην έξοδο του backbone. Τώρα το RPN χρησιμοποιεί όλα τα στρώματα του FPN δικτύου για να δημιουργεί προτάσεις από κουτιά άγκυρες διαφόρων διαστάσεων. Δεν χρησιμοποιούμε αυθαίρετα τις ίδιες διαστάσεις για την δημιουργία προτάσεων σε κάθε στρώμα. Για παράδειγμα τα anchor boxes που είναι μικρά παράγονται από τους χάρτες χαρακτηριστικών των στρωμάτων υψηλής ανάλυσης του FPN δηλαδή τα κατώτερα στην πυραμίδα. Το RPN λειτουργεί ακριβώς όπως και πριν με πλήρως συνελικτικό τρόπο απλά περνάμε μέσα από αυτό όλους τους χάρτες, παίρωντας στην έξοδο του πιθανότητα αν το anchor περιέχει αντικείμενο και βελτίωση του ορθογωνίου. Τα επόμενα βήματα αυτού του σταδίου είναι παρόμοια με το Faster R-CNN. Γίνεται λοιπόν Non Maximum Suppression και Negative Sampling και την δημιουργία του συνόλου των τελικών προτάσεων.

Στο Σχήμα 3.9 βλέπουμε με P1,..,P5 τα πολλαπλά στρώματα-χάρτες χαρακτηριστικών διαφορετικής ανάλυσης που παράγει το FPN. Κάθε ένα από αυτά θα περάσει από το RPN ξεχωριστά σε αντίθεση με πριν όπου μόνο το top θα πέρναγε.



Σχήμα 3.9: Η εικόνα περνάει από το resnet backbone δίκτυο στο FPN και έπειτα όλοι οι χάρτες διοχετεύονται στο RPN

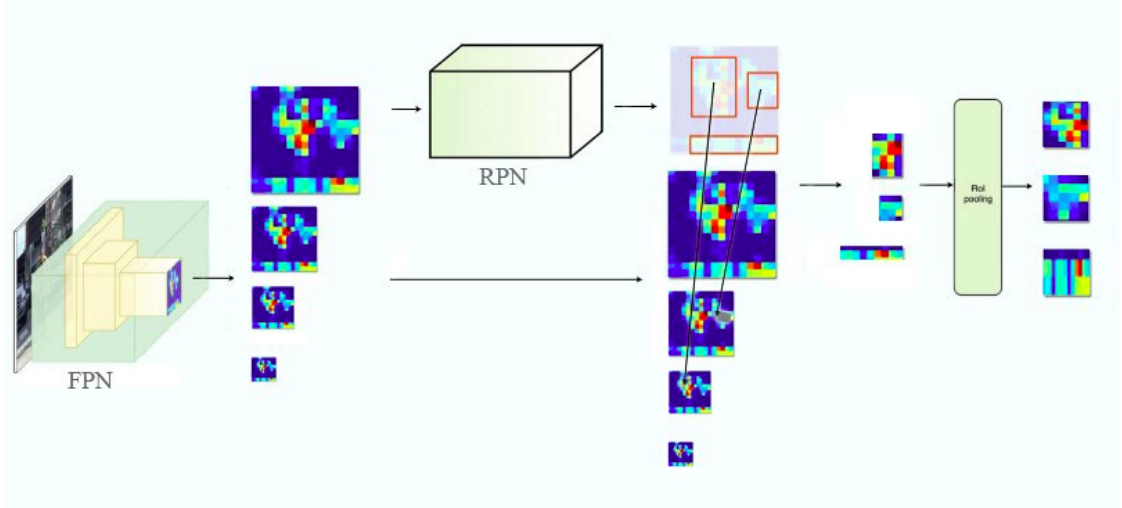
3.2.4 RoiPooling και RoiAlign

Σε αυτό το στάδιο στο faster R-CNN χρησιμοποιούσαμε τα ορθογώνια οριοθέτησης των προτάσεων που πήραμε και βάση αυτών κόβαμε τον όγκο χαρακτηριστικών της εξόδου του βασικού δικτύου στις διαστάσεις που αντιστοιχεί το ορθογώνιο. Μέσω pooling σε αυτό το κομμάτι παίρναμε το τελικό 7x7x512 όγκο που διοχευτεύουμε στο στάδιο του R-CNN. Το ίδιο θα κάνουμε και τώρα με τη διαφορά ότι για κάθε πρόταση πρέπει πρώτα να διαλέξουμε ποιο από τους P1,...,Pn όγκους χαρακτηριστικών που δίνει το FPN θα χρησιμοποιήσουμε για να πάρουμε το τμήμα που θα γίνει το pooling. Η συγκεκριμένη πρόταση βέβαια στο προηγούμενο στάδιο προέκυψε από την διοχέτευση ενός συγκεκριμένου από αυτούς τους όγκους στο RPN και θα μπορούσαμε να θυμόμαστε ποιόν για να χρησιμοποιήσουμε τον ίδιο. Όμως το ορθογώνιο οριοθέτησης έχει αλλάξει στο στάδιο του RPN και έτσι προτιμούμε να διαλέξουμε κάποιο από αυτούς ξανά βάσει του μεγέθους της επιφάνειας του τελικού ορθογωνίου.

Ο τύπος που χρησιμοποιείται για την επιλογή είναι ο παρακάτω.

$$k = \left\lfloor k_0 + log_2(\frac{\sqrt{width * height}}{224}) \right\rfloor$$

Στην παραπάνω εξίσωση με k παίρνουμε τον δείκτη του στρώματος του FPN που θα χρησιμοποιήσουμε, δηλαδή είναι P_k ενώ $k_0 = 4$ και τέλος $width * height = bbox_{area}$. Όσο πιο μεγάλο είναι το ορθογώνιο της πρόβλεψης τόσο πιο μεγάλο θα είναι το k με αποτέλεσμα να χρησιμοποιούμε στρώματα ψηλά στην πυραμίδα δηλαδή στρώματα με χαμηλή ανάλυση. Αντίθετα όσο πιο μικρό τόσο πιο χαμηλά στρώματα στο FPN ανταλλάσσοντας κάποια σημασιολογία για περισσότερη ανάλυση.
Στο Σχήμα 3.10φαίνεται αυτή η λειτουργία. Τέλος μια σημαντική αλλαγή του Mask R-CNN είναι η αντικατάσταση του RoI Pooling με RoI Align. Το αποτέλεσμα των δύο αυτών λειτουργιών είναι ακριβώς το ίδιο δηλαδή 7x7x512 έξοδοι απλά αντί για δειγματοληψία εφαρμόζεται κάποιου είδους παρεμβολή για να μην χαθεί καμία πληροφορία. Η δειγματοληψία ενώ λειτουργεί μια χαρά για την αναγνώριση αντικειμένων αποτυγχάνει όταν θέλουμε να κάνουμε και πρόβλεψη μάσκας. Στην προκειμένη περίπτωση χρειάζεται να σώσουμε όσο περισσότερη πληροφορία μπορούμε.



Σχήμα 3.10: Η επιλογή των κατάλληλων στρωμάτων για RoI pooling ή align

3.2.5 Το R-CNN

Δεν υπάρχουν πολλά παραπάνω να πούμε για αυτό το μέρος του δικτύου μιας και λειτουργεί ακριβώς όπως και στο Faster R-CNN. Δέχεται τα fixed size feature maps και εκτελεί ταξινόμηση στην τελική κλάση και ταυτόχρονα την τελική βελτίωση του ορθογωνίου.

3.2.6 Το υποδίκτυο πρόβλεψης μάσκας

Αυτό το δίκτυο είναι η τελευταία προσθήκη του Mask R-CNN και είναι υπεύθυνο για την πρόβλεψη μάσκας. Η διαδικασία είναι παρόμοια με αυτήν για την εύρεση του bounding box. Δουλεύουμε με συνελικτικό τρόπο. Αρχικά παίρνουμε όλες τις προτάσεις ξανά και τις περνάμε από την διαδικασία του RoI Align για να πάρουμε για κάθε μια από αυτές τα αντίστοιχα feture volume με τα οποία θα δουλέψουμε. Η διαφορά είναι ότι τώρα έχουμε ανάλυση 14x14 αντί για 7x7 για τους χάρτες χαρακτηριστικών. Τέλος τα περνάμε και από μια σειρά από στρώματα με 3x3 συνελικτικά φίλτρα και RelU τα οποία δίνουν κάποιο νέο βάθος αλλά δεν αλλάζουν τις χωρικές διαστάσεις.

Στο Σχήμα 3.11 φάινεται η διαδικασία. Με D είναι το πλήθος των προβλέψεων. Με C είναι το βάθος στον όγκο χαρακτηριστικών (512 ή 256) ενώ με P έχουμε τις χωρικές διαστάσεις μετά το RoIAlign και είναι ίσο με 14 όπως αναφέραμε. C' είναι το νέο βάθος μετά τα πολλαπλά 3x3 στρώματα. Πόσα 3x3 στρώματα και πόσο θα είναι το τελικό βάθος είναι υπέρ-παράμετροι που αλλάζουν λόγω της υλοποίησης ή και λόγω της επιλογής backbone.

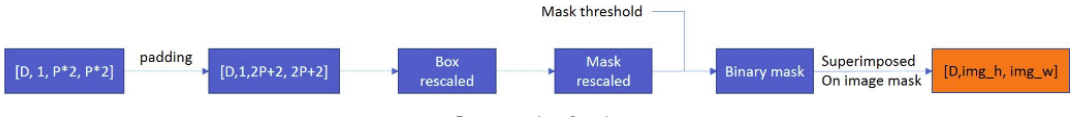
Σχήμα 3.11: Το πρώτο βήμα για την δημιουργία της μάσκας

Τα χαρακτηριστικά των προτάσεων μετά από αυτήν την επεξεργασία περνάνε έπειτα από ένα στρώμα που εκτελεί αποσυνέλιξη για να αυξήσει την ανάλυση σε 28x28 και μετά από έναν συνελικτικό στρώμα με N φίλτρα 1x1xC', με N το πλήθος των διαφορετικών κλάσεων του προβλήματος. Αυτό το δίκτυο παράγει ένα τρισδιάστατο πίνακα [N,28,28] που περιέχει προβλέψεις μάσκας για κάθε κλάση. Όπως και το R-CNN κάνει N προβλέψεις για βελτίωση του ορθογωνίου και διαλέγουμε μόνο αυτή για την οποία συμφωνεί η ταξινόμηση της πρόβλεψης, έτσι και εδώ θα διαλέξουμε μόνο μια από τις μάσκες. Στο Σχήμα 3.12 φαίνεται οπτικά η διαδικασία.

Σχήμα 3.12: Δημιουργία προβλέψεων μάσκας για κάθε κλάση

Μετά και αφού επιλέξουμε την μάσκα που αντιστοιχεί στην κλάση έχουμε έξοδο [1,28,28]. Μπορεί να φαίνεται πολύ χαμηλή η ανάλυση για να πετύχουμε σωστή πρόβλεψη μάσκας αλλά οφείλουμε να παρατηρήσουμε το εξής. Οι τιμές στο παραπάνω δεν είναι 0 ή 1 αλλά συνεχείς οπότε και κωδικοποιούν αρκετά περισσότερη πληροφορία από μια δυαδική μάσκα ανάλυσης 28x28. Αυτό που μένει είναι να επεξεργαστούμε αυτή την πληροφορία για να δημιουργήσουμε την δυαδική τελική μάσκα.

Οι μάσκες λοιπόν κλιμακώνονται στις διαστάσεις της αρχικής εικόνας όπως και το ορθογώνιο οριοθέτησης. Αυτό γίνεται πάλι με παρεμβολή (bi-linear) και έπειτα εφαρμόζεται ένα όριο ως κριτήριο για να διαλέξουμε ποιες θέσεις θα είναι ένα και ποιες μηδέν. Συνήθως όσα εικονοστοιχεία είναι μεγαλύτερα του 0.5 γίνονται 1 και θεωρούνται ότι ανήκουν στο αντικείμενο. Στο Σχήμα 3.13 έχουμε την διαδικασία της επεξεργασίας.



Σχήμα 3.13: Τελική επεξεργασία του συνόλου των προβλέψεων μάσκας

3.2.7 Η εκπαίδευση του Mask R-CNN

Η εκπαίδευση του RPN και του R-CNN γίνεται με παρόμοιο τρόπο όπως και στο Faster R-CNN. Εφαρμόζεται πάλι στρατηγική ταιριάσματος των anchor boxes με τα ground truth boxes και με χρήση της IoU αποφασίζουμε από πού θα μάθει και τι κάθε anchor box. Επίσης με αρνητική δειγματοληψία προστατεύουμε τους ταξινομητές από την αρνητική επίδραση των πολλαπλάσιων αρνητικών παραδειγμάτων παρασκηνίου.

Η διαφορά του μοντέλου είναι πως τώρα για κάθε Region of Interest παίρνουμε έχουμε συνάρτηση σφάλματος τριών όρων.

$$L = L_{cls} + L_{box} + L_{mask}$$

Οι δύο πρώτοι όροι δηλαδή η ταξινόμηση σε κλάση αντικειμένου και το regression του ορθογωνίου οριοθέτησης είναι ίδιες με το μοντέλο Faster R-CNN. Ο τρίτος όρος είναι το σφάλμα της μάσκας και για τον υπολογισμό του λαμβάνουμε υπόψιν μόνο την πρόβλεψη μάσκας για την κλάση αντικειμένου. Όπως αναφέραμε παραπάνω το μοντέλο κάνει πάντα N προβλέψεις μάσκας για κάθε RoI όπου N ο αριθμός των κλάσεων. Έτσι δεν έχουμε ανταγωνισμό των κλάσεων κατά την εκπαίδευση για αναγνώριση μάσκας. Οι δημιουργοί του μοντέλου έδειξαν ποσό σημαντικό είναι το παραπάνω λεγόμενο decoupling των μασκών σε πειραματικές μετρήσεις δοκιμάζοντας και τις δύο προσεγγίσεις.

Τέλος για επιλεγμένη μάσκα της κλάσης-αντικειμένου που αντιστοιχήθηκε στην πρόβλεψη, για το σφάλμα της μάσκας εφαρμόζουμε διασταυρωμένη εντροπία (cross-binary-entropy) πάνω σε όλο το σύνολο των εικονοστοιχείων όπου με 1 είναι η κλάση - εικονοστοιχείου που ανήκει στην μάσκα και 0 που δεν ανήκει. Συγκεκριμένα ο τύπος είναι ο παρακάτω.

$$L_{mask} = -\frac{1}{m^2} \sum_{1 \leq i,j < m} [y_{ij} log y^*_{ij} + (1 - y^*_{ij}) log(1 - y_{ij})]$$

Όπου y_{ij} είναι η πραγματική τιμή στη θέση ij για το ανήκει ή όχι στο αντικείμενο από την ground truth μάσκα, y^*_{ij} είναι η πρόβλεψη του μοντέλου την οποία απομονώνουμε από το σύνολο των προβλέψεων μασκών για κάθε κλάση ανάλογα με το ποιά κλάση προβλέψαμε.

3.3 Τα δίκτυα ResNet (Residual Networks)

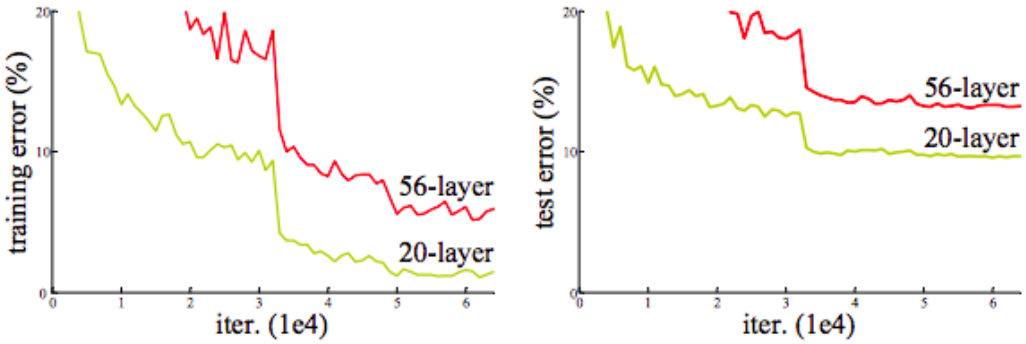
Σε αυτήν την ενότητα θα περιγράψουμε την αρχιτεκτονική των συνελικτικών δικτύων που θα χρησιμοποιήσουμε για το πρόβλημα της ταξινόμησης των εικόνων. Τα δίκτυα αυτά ονομάζονται Resnet από το Residual Networks και τα χαρακτηρίζουν οι απευθείας συνδέσεις των στρωμάτων με μεταγενέστερα. Τα ίδια δίκτυα θα χρησιμοποιηθούν και για την ανίχνευση αντικειμένων ως εξαγωγείς χαρακτηριστικών όπως αναφέραμε ήδη. Ο λόγος είναι η αρκετά καλύτερη απόδοση τους από τα κλασσικά συνελικτικά. Μάλιστα μέχρι και σήμερα η απόδοση τους τα έχει καθιερώσει ως state of the art στον τομέα της ανάλυσης εικόνας.

3.3.1 Το βάθος του δικτύου και η απόδοση

Από την πρώτη στιγμή που τα βαθιά συνελικτικά νευρωνικά δίκτυα κατάφεραν να ξεπεράσουν συμβατικούς αλγορίθμους σε απόδοση [Kriz12] υπήρξε η πεποίθηση ότι όσο πιο βαθιά πάμε δηλαδή όσο πιο πολλά συνελικτικά στρώματα τόσο καλύτερη απόδοση μπορούμε να πετύχουμε. Αυτή η ιδέα είναι αρκετά βάσιμη μιας και τα επιπλέον στρώματα θεωρητικά επιτρέπουν στο μοντέλο μεγαλύτερη ικανότητα να μαθαίνει περίπλοκα χαρακτηριστικά.

Αυτή η πεποίθηση διαψεύστηκε γρήγορα στην πράξη μιας και από ένα ορισμένο βάθος και έπειτα η απόδοση έπεφτε αντί να ανεβαίνει. Ακόμα έτσι η πτώση αυτή αποδόθηκε στο πρόβλημα της υπέρπροσαρμογής, στην αρχικοποίηση των δικτύων και στο πρόβλημα των εξαφανιζόμενων παραγώγων. Για τα δύο πρώτα θέματα, αλγόριθμοι όπως dropout και χρήση L2 νόρμας καθώς και επιπλέον παράμετροι για κανονικοποίηση χρησιμοποιήθηκαν χωρίς κάποιο κέρδος. Για το πρόβλημα εξαφανιζόμενων παραγώγων η ομαδοποίηση ομάδας που αναφέραμε βοηθάει στο να κρατήσει τις τιμές τους σε υγιή επίπεδα καθώς προχωράμε προς τα πίσω με την ανάστροφη διάδοση. Παρόλα αυτά ούτε αυτό άλλαξε την χαμηλή απόδοση των πολύ βαθιών δικτύων.

Μια λύση θα δωθεί από τους He et al. [He15]. Αρχικά δείξανε εμπειρικά πως η αύξηση των στρωμάτων ρίχνει την απόδοση από ένα όριο και μετά. Στο Σχήμα 3.14παρουσιάζονται από την μελέτη τους γραφικές του σφάλματος στα σύνολα εκπαίδευσης και αξιολόγησης δεδομένων όπου φαίνεται πως το δίκτυο 56 στρωμάτων αποτυγχάνει να πετύχει καλύτερη απόδοση ακόμα και στο training set. Κάτι τέτοιο δεν θα ίσχυε εάν η χαμηλή απόδοση ήταν προϊόν απλά της υπέρ προσαρμογής. Στο φαινόμενο της υπερ προσαρμογής το μοντέλο πετυχαίνει πάρα πολύ καλή απόδοση στα δεδομένα εκπαίδευσης και αποτυγχάνει να γενικεύσει με καλή απόδοση στα νέα δεδομένα του συνόλου αξιολόγησης.

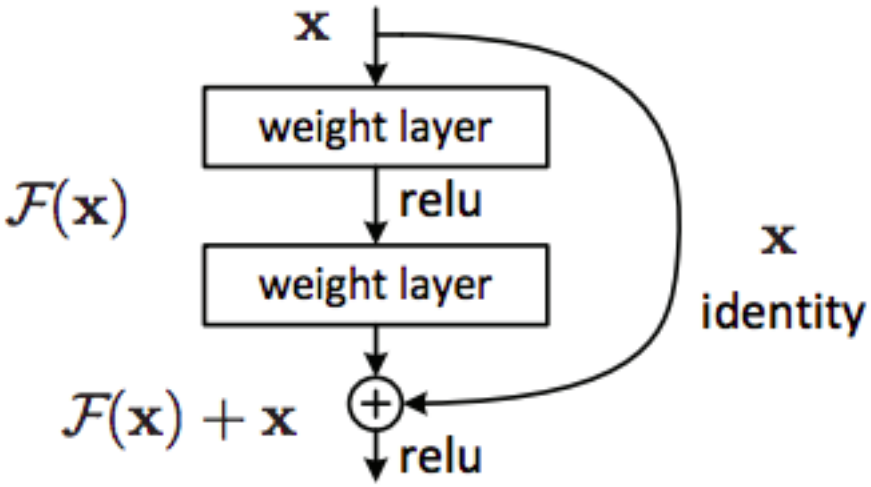


Σχήμα 3.14: Σύγκριση απόδοσης δικτύων με διαφορετικό βάθος

Δείξανε επίσης ότι ένα καλό βάθος είναι γενικά ανάμεσα σε 16 και 30 στρώματα. Για να αυξήσουν το πλήθος των στρωμάτων εισήγαγαν μια νέα έννοια που την ονόμασαν Residual Block.

3.3.2 To Residual Block

Αυτό το κομμάτι του δικτύου αποτελεί και τη καινοτομία των resnets. Ουσιαστικά αντί να προωθούμε απλά την είσοδο σε κάθε στρώμα διαδοχικά και με τη σειρά της τοπολογίας του δικτύου έχουμε και συνδέσεις που προωθούν την έξοδο σε μεταγενέστερα στρώματα αυτούσια (skip connections). Στο Σχήμα 3.15 φαίνεται το block αυτό.



Σχήμα 3.15: Residual block

Η ιδέα είναι η εξής. Εφόσον τα νευρωνικά δίκτυα μπορούνε να αναπαριστούν πολύπλοκες συναρτήσεις τότε θα πρέπει να είναι σε θέση να αναπαριστούν και την ταυτοτική συνάρτηση.

$$f(x) = x$$

Έτσι ένα πολύ βαθύ δίκτυο θα μπορούσε να αναιρέσει αρκετά αχρείαστα στρώματα του παραμετροποιώντας τα βάρη τους ώστε να περνάνε αυτούσια την είσοδο. Πρακτικά ένα πολύ βαθύ δίκτυο θα μπορούσε να ισοδυναμεί με ένα πιο ρηχό πράγμα που σημαίνει ότι δεν υπάρχει κάποιος λόγος το πιο ρηχό δίκτυο να είναι καλύτερο.
Περνώντας την είσοδο αυτούσια στην έξοδο η συνάρτηση που μαθαίνει τώρα το δίκτυο είναι η παρακάτω.

$$f(x) + x = h(x)$$

Τώρα για να προσεγγίσουμε την ταυτοτική συνάρτηση το δίκτυο αρκεί να μάθει ότι $f(x) = 0$ δηλαδή να μηδενίσει τις τιμές των βαρών για αυτό το τμήμα και να έχουμε ροή των δεδομένων μόνο από την απευθείας σύνδεση. Προκύπτει ότι για το δίκτυο αυτό είναι αρκετά πιο εύκολο για αυτό και πειραματικά έχουμε αρκετά καλύτερη απόδοση βαθιών δικτύων Resnet.

Επιπλέον έχουμε ροή των παραγώγων μέσα από τις skip connections οπότε και αντισταθμίζεται το πρόβλημα των εξαφανιζόμενων παραγώγων αφού τα φίλτρα των στρωμάτων δέχονται gradients άμεσα ακόμα και από την τελική έξοδο. Βέβαια αυτές οι συνδέσεις προσθέτουν επιπλέον πολυπλοκότητα στον υπολογισμό της ανάστροφης διάδοσης. Επίσης αυτές οι συνδέσεις απαιτούν η έξοδος από το block να είναι ίδιων διαστάσεων με την είσοδο. Αυτό δεν είναι πάντα εφικτό γιατί όπως είδαμε τα κλασσικά συνελικτικά στρώματα αλλάζουν τον όγκο δεδομένων σύμφωνα με το stride και padding ενώ επίσης μπορούμε να έχουμε και pooling layers μέσα στο block αν και αυτό αποφεύγεται πια. Στην περίπτωση αυτή οι εμπνευστές του μοντέλου μετασχηματίζουν την είσοδο μέσω γραμμικής προβολής με ένα τένσορα W.

$$y = F(\mathbf{x}, W_i) + W_s\mathbf{x}$$

Με τον όρο W_s αναπαριστούμε την προβολή η οποία μάλιστα μπορεί να υλοποιηθεί ως συνέλιξη προσθέτωντας όμως έξτρα παραμέτρους στο δίκτυο.

3.3.3 Η γενική αρχιτεκτονική των δικτύων Resnet στην πράξη

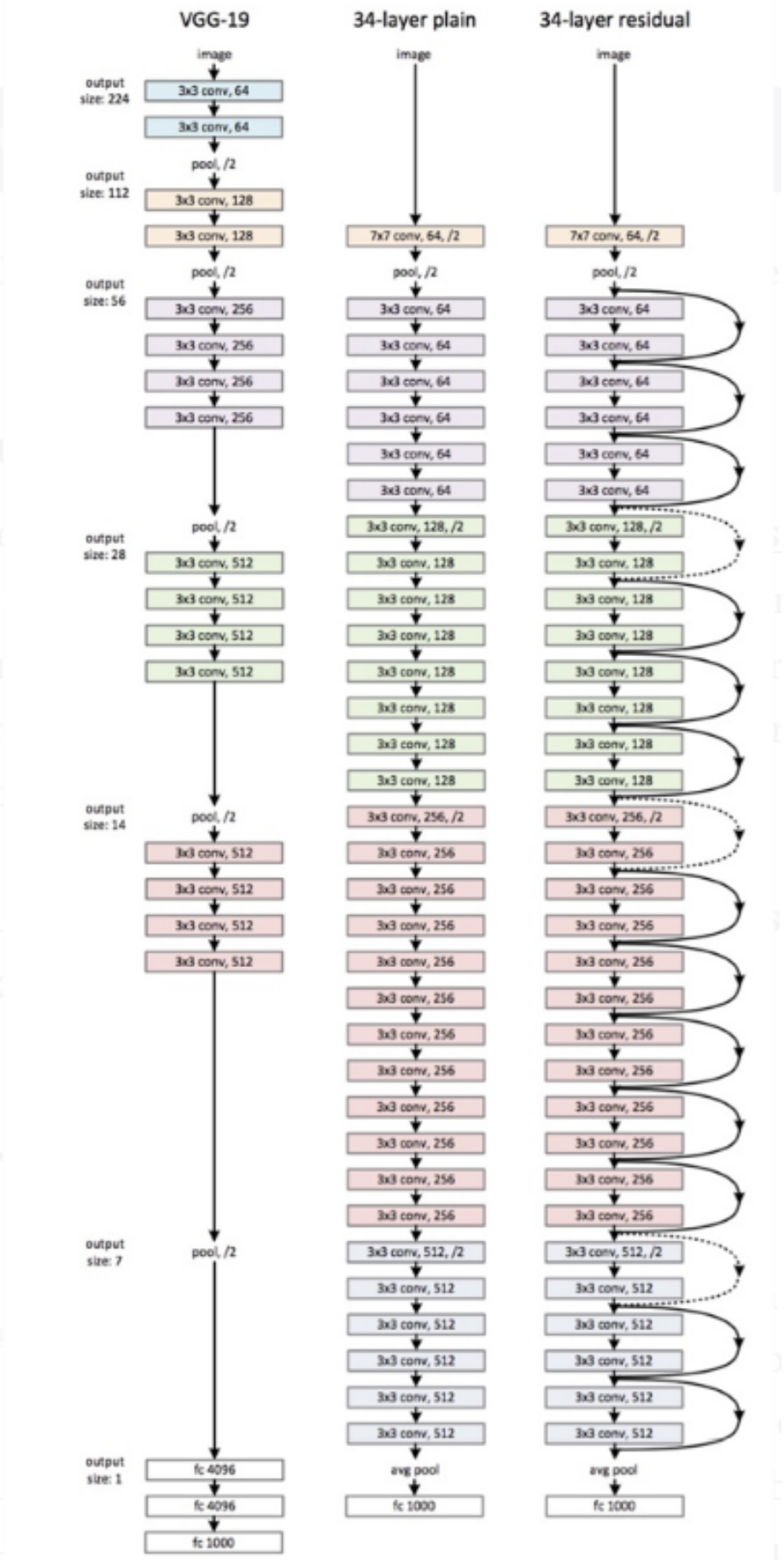
Μπορούμε να δούμε καλύτερα το πως δομούνται αυτού του είδους τα δίκτυα περιγράφοντας την εκδοχή του. Στο Σχήμα 3.17 παρατίθενται οι εκδοχές του διάσημου μοντέλου VGG-19 , ενός απλού κλασσικού συνελικτικού δικτύου 34 στρωμάτων και τέλος η αντίστοιχη εκδόχή με skip connections σε κάθε δύο συνελικτικά στρώματα. Τα δίκτυα αυτά έχουν δημιουργηθεί για να εκπαιδευτούν πάνω στο ImageNet στο οποίο οι εικόνες έχουν ανάλυση 224x2224. Με κατάλληλες τροποποιήσεις μπορούμε εύκολα να αλλάξουμε τις διαστάσεις της εισόδου που περιμένει το δίκτυο. Βέβαια εφόσον μιλάμε για ταξινόμηση αναγκαστικά οι εικόνες θα πρέπει να είναι όλες σταθερού και ίδιου μεγέθους. Γενικά μπορούμε να παρατηρήσουμε τα παρακάτω για τα στρώματα του Resnet. Αρχικά η είσοδος περνάει από ένα συνελικτικό στρώμα με 64 φίλτρα 7x7x3 που με τις κατάλληλες τιμές stride και adding μετασχηματίζουν τον όγκο εισόδου από (224,224,3) σε (112,112,64). Έπειτα ένα max pooling στρώμα με φίλτρο 3x3 και stride ίσο με 2 μειώνει περαιτέρω την ανάλυση σε (56,56,64). Μετά έχουμε τα λεγόμενα στρώματα Resnet.

Τα στρώματα στο Resnet block είναι πολύ απλά και έχουν ένα επαναλαμβανόμενο μοτίβο. Ένα ResNet block αποτελείτα μόνο από συνελικτικά στρώματα και είτε κρατάει σταθερό τον όγκο που περνάει από μέσα του με κατάλληλο padding,stride και αριθμό φίλτρων, είτε μειώνει την ανάλυση στο μισό διπλασιάζοντας ταυτόχρονα όμως το βάθος. Το εσωτερικό του Resnet block αποτελείται από στρώματα 1x1 ή/και 3x3 συνελίξεων. Το πρώτο στρώμα χαμηλώνει την ανάλυση στην περίπτωση όπου έχουμε ”convolutional block” και διπλασιαζει τον όγκο. Η γενική αρχιτεκτονικη είναι ένα convolutional block ακολουθούμενο από πολλαπλά ”identity blocks δηλαδή blocks που η έξοδος τους έχει ίδιες διαστάσεις με την είσοδο. Στο Σχήμα 3.17 στο δίκτυο Resnet κάθε φορά που πέφτει η ανάλυση λόγω ενός στρώματος, έχουμε το ” /2 ” δίπλα από τα στοιχεία αυτού του στρώματος ενώ φαίνεται πως έχει διπλασιαστεί και το βάθος. Επιπλέον έχουμε τις απευθείας συνδέσεις. Κάθε block αθροίζει την είσοδο του στη έξοδο αυτούσια αν είναι identiy block ή μετά από κατάλληλη συνέλιξη αν είναι convolutional block. Με συνεχείς γραμμές στο Σχήμα 3.17 δηλώνεται η απευθείας σύνδεση η οποία είναι εφικτή αφού δεν έχουμε αλλαγές στις διαστάσεις ενώ με διακεκομμένες γραμμές υπονοείτε ότι χρειάζεται να γίνει προβολή στις νέες διαστάσεις.

Τέλος έχουμε το average pooling και τροφοδοτήσει σε ένα πλήρως συνδεδεμένο στρώματα το οποίο θα δώσει στην έξοδο τις πιθανότητες για τις κλάσεις της ταξινόμησης, ακριβώς όπως και στα κλασσικά συνελικτικά δίκτυα. Στο Σχήμα 3.16 παρουσιάζονται διάφορες εκδόσεις των Resnet.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

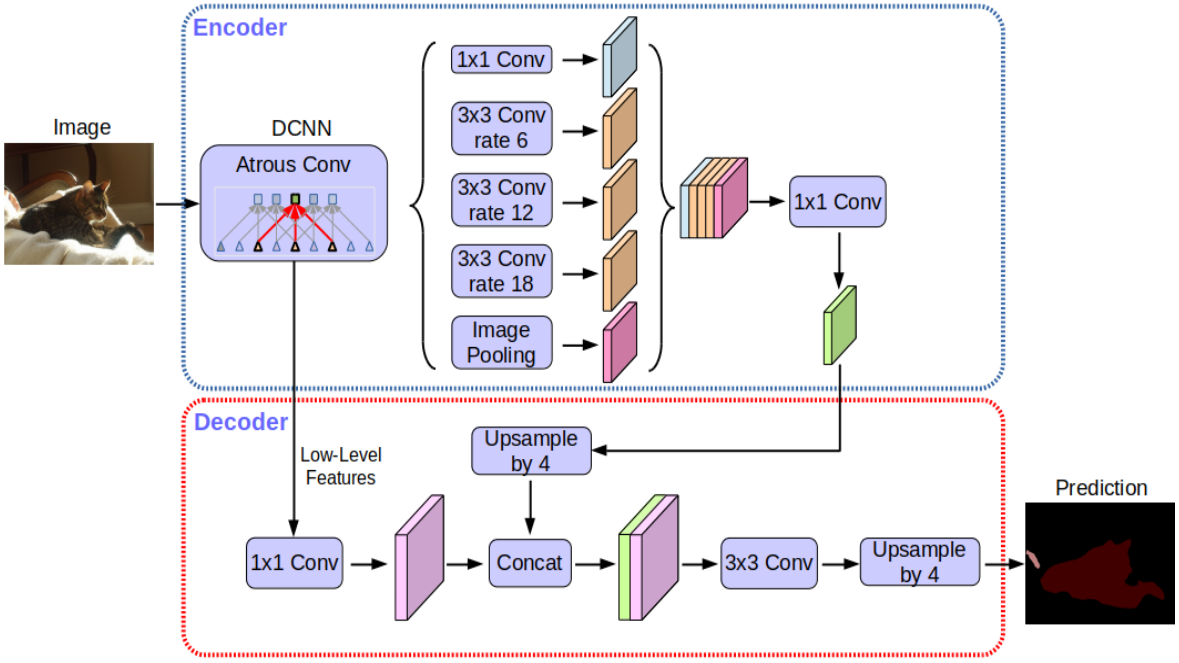
Σχήμα 3.16: Διάφορες εκδοχές των Resnet



Σχήμα 3.17: vgg19,plain 34 και resnet 34 architecture

3.3.4 DeepLab v3+

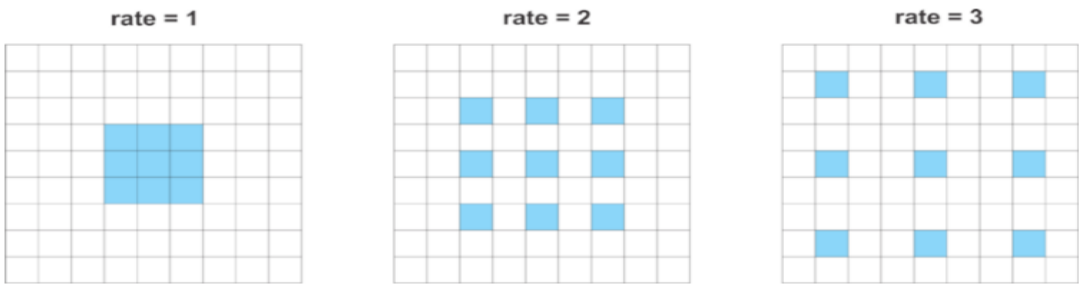
Εδώ θα παρουσιάσουμε πολύ συνοπτικά λίγα από τα χαρακτηριστικά της νέας έκδοσης του μοντέλου της Google για segmentation εικόνας [Chen18b]. Η αρχιτεκτονική του παρουσιάζεται στο παρακάτω Σχήμα.



Σχήμα 3.18: Residual block

Δυο καινούριες έννοιες είναι αυτές του encoder/decoder τις οποίες το deeplab υιοθετεί για να δομήσει την αρχιτεκτονική του. Στην φάση encoding χρησιμοποιείται ένα συνελικτικό δίκτυο για backbone όπως και στα προηγούμενα μοντέλα για αποδόμηση της εικόνας και συμπύκνωση της σε χώρο μικρότερης διάστασης. Η λειτουργία ενός encoder είναι ακριβώς αυτή ενώ ιδανικά θέλουμε η αντίστροφη διαδικασία να μπορεί να δημιουργήσει την είσοδο όσο πιο πιστά γίνεται. Έτσι εκπαιδεύονται γενικά δυάδες encoder/decoder.

Εδώ αξιοσημείωτη είναι η επίσης η χρήση στο δίκτυο ενός καινούριου τρόπου συνέλιξης που λέγεται atrous convolution. Συγκεκριμένα το κυλιόμενο παράθυρο επιδέχεται μια καινούρια υπερ παράμετρο (rate) σύμφωνα με την οποία διαστέλλεται διατηρώντας το ίδιο πλήθος κελιών αλλά διατάσσοντας τα σε απόσταση μεταξύ τους. Το αποτέλεσμα είναι ο νευρώνας να βλέπει πιο μεγάλη περιοχή του προηγούμενου στρώματος χωρίς να έχουμε τον όγκο πράξεων που προσθέτει μια κανόνική συνέλιξη με το ίδιο πεδίο οράσεως.



Σχήμα 3.19: atrous συνέλιξη

Για το decoding στάδιο όπως φαίνεται από το σχήμα γίνεται μερικό upsample και το αποτέλεσμα συνενώνεται με τους χάρτες χαρακτηριστικών χαμηλών επιπέδων όπως αυτά λαμβάνονται από το στάδιο του encode. Έπειτα ακολουθεί μια σειρά από επιπλέον συνελικτικά στρώματα και upsampling για να αποκτήσει η έξοδος της διαστάσεις της αρχικής εικόνας.

Γενικά το μοντέλο αυτό είναι εφάμιλλης πολυπλοκότητας με μοντέλα όπως το Mask R-CNN και μια περίληψη δεν είναι αρκετή για να μελετήσει κανείς τη λειτουργία του. Ύπαρχουν επιπλέον καινοτομίες όπως ένας διαφορετικός και υπολογιστικά καλύτερος τρόπος για αύξηση του όγκου του συνελικτικού στρώματος καθώς και λεπτομέρειες της αρχιτεκτονικής και του τρόπου εκμάθησης του δικτύου. Έμεις θα χρησιμοποιήσουμε την εκδοχή v3+ με backbone το βαθύ συνελικτικό δίκτυο Xception (modified Inception) για να το συγκρίνουμε με το πιο γενικό Mask R-CNN.

Κεφάλαιο 4

Εκπαίδευση και αξιολόγηση των μοντέλων για τα προβλήματα της ανίχνευσης,κατάτμησης και ταξινόμησης

4.1 Εισαγωγή

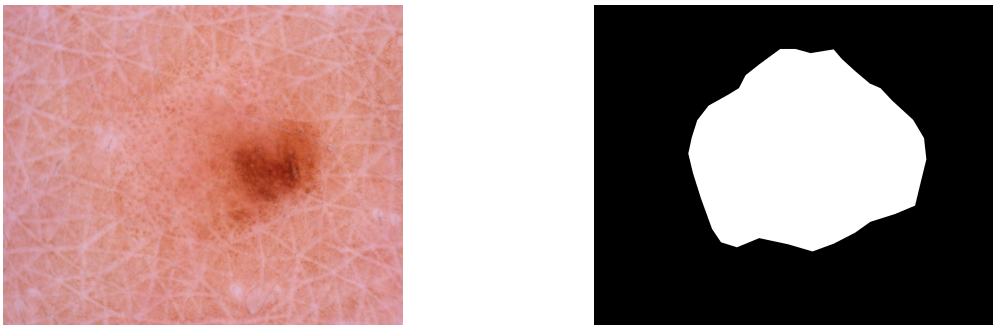
Σε αυτήν την ενότητα θα παρουσιαστούν τα αποτελέσματα της απόδοσης που καταφέραμε να πετύχουμε στα προβλήματα της αναγνώρισης δερματικών ελιών και στην ταξινόμηση τους σε διάφορες κλάσεις παθογένειας. Για την επίτευξη των παραπάνω θα γίνει χρήση των μοντέλων που περιγράφηκαν στην προηγούμενη ενότητα για διάφορες εκδόσεις της αρχιτεκτονικής τους και υπέρπαραμέτρους καθώς και συνδυαστικά μεταξύ τους.

Για την εκπαίδευση και την αξιολόγηση για instance segmentation των δερματικών σπίλων θα χρησιμοποιηθούν τα σύνολα δεδομένων από τον διαγωνισμό ISIC2018 ([Code19] και [Tsch18]) . Για την ταξινόμηση σε 8 κλάσεις θα χρησιμοποιηθούν τα σύνολα δεδομένων από τον διαγωνισμό ISIC2019 ([Tsch18] και [Code17] και [Comb19]). Τέλος για την υλοποίηση και χρήση των μοντέλων χρησιμοποιήθηκε η γλώσσα Python3 καθώς και τα Keras [Chol15] και Tensorflow [Abad15].

4.2 Segmentation των δερματικών σπίλων

4.2.1 Τα δεδομένα

Για αυτό το πρόβλημα θα χρησιμοποιηθούν τα δεδομένα του ISIC2018. Συγκεκριμένα και επειδή δεν δίνονται ground truth μάσκες για το test data του διαγωνισμού θα χρησιμοποιήσουμε μόνο τα προσφερόμενα δεδομένα εκπαίδευσης. Συνολικά έχουμε στη διάθεση μας 2594 εγχρωμες εικόνες με τις αντίστοιχες μάσκες τους που περιέχουν ένα ακριβώς instance δερματικού σπίλου. Αυτές τις χωρίζουμε σε **1794** εικόνες για training set, **300** εικόνες για validation set και τέλος **500** εικόνες στις οποίες θα γίνει η τελική αξιολόγηση. Η ανάλυση των εικόνων ποικίλλει και φτάνει μεχρι 1920x1080 εικονοστοιχεία.



Σχήμα 4.1: Εικόνες από το dataset με ground truth mask

Σχήμα 4.2: Εικόνες από το dataset με ground truth mask

4.2.2 Mask R-CNN με FPN και backbone ResNet50

Θα χρησιμοποιήσουμε την υλοποίηση της Matterport [Abdu17] για το μοντέλο Mask R-CNN. Αρχικά θα χρησιμοποιήσουμε ως backbone ένα Resnet με 50 στρώματα και FPN. Οι ρυθμίσεις των υπέρ παραμέτρων του δικτύου που διαφοροποιούνται από την default παραμετροποίηση παρουσιάζονται στον Πίνακα 4.1

Backbone	Resnet50
Detection minimum confidence	0.01
Image minimmum/maximum dimension	512/512
Lenght of square anchor side	16,32,64,128,256
NMS threshold for RPN proposals	0.5
Anchors / image for RPN training	512
Images per GPU (batch size)	2
Use of mini-masks	True
Use of augmentation	True
mini batch	2
Maximum number of ground truth instances per image	1

Πίνακας 4.1: Mask R-CNN configuration

Για το augmentation του dataset χρησιμοποιούμε σε κάθε εικόνα πριν τη διοχέτευση της στο pipeline του μοντέλου δύο από τις επιλογές του Πίνακα στην τύχη 4.2.

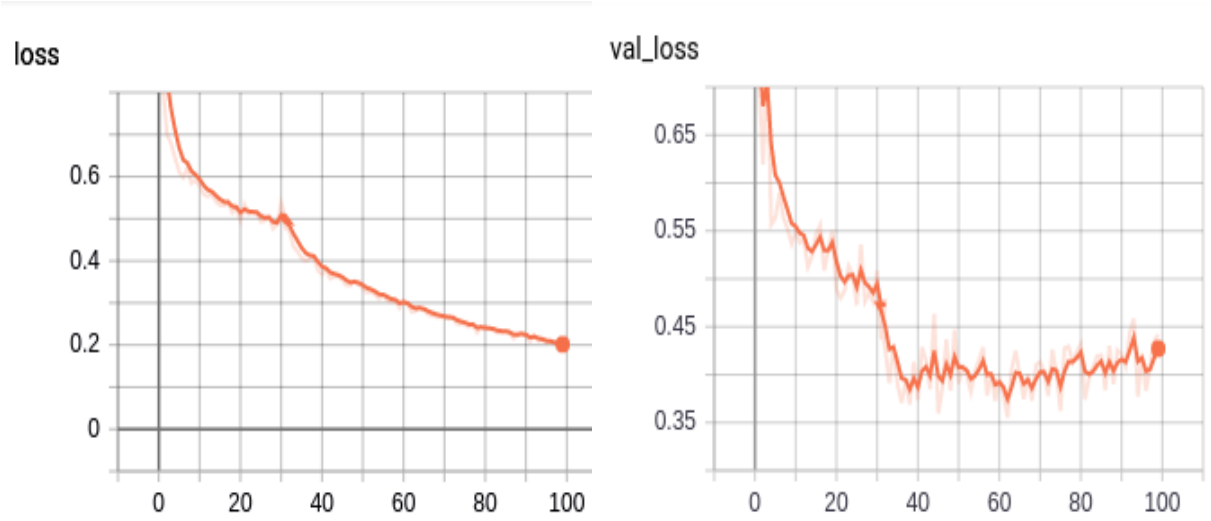
flips	horizontal or vertical
rotations	90,180,270
scaling	0.8,1.5
Gaussian blurring	s=0.0 ή s=1.5

Πίνακας 4.2: Data augmentation

Τέλος δεν αρχίζουμε την εκπαίδευση εκ του μηδενός ”from scratch” αλλά επιλέγουμε αρχικά βάρη τα βάρη το μοντέλου που προκύπτουν από την εκπαίδευση του στο σύνολο δεδομένων COCO [Lin14].

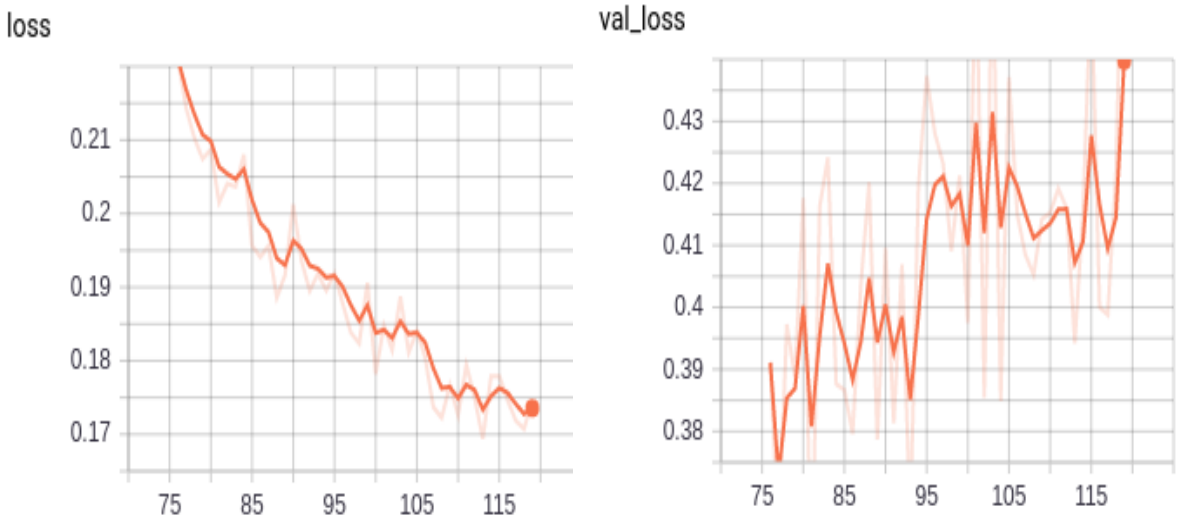
Εκπαίδευση

Εκπαιδεύουμε το δίκτυο για 30 εποχές όπου αλλάζουμε τα βάρη μόνο της κεφαλής του μοντέλου δηλαδή το RPN, R-CNN και το Mask branch με ρυθμό εκμάθησης l=0.001 και momentum=0.9. Έπειτα εκπαιδεύουμε για 70 εποχές ολόκληρο το δίκτυο από άκρη σε άκρη. Στα Σχήματα έχουμε τις συναρτήσεις σφάλματος για αυτές τις πρώτες 100 εποχές στα training και validation σύνολα δεδομένων.



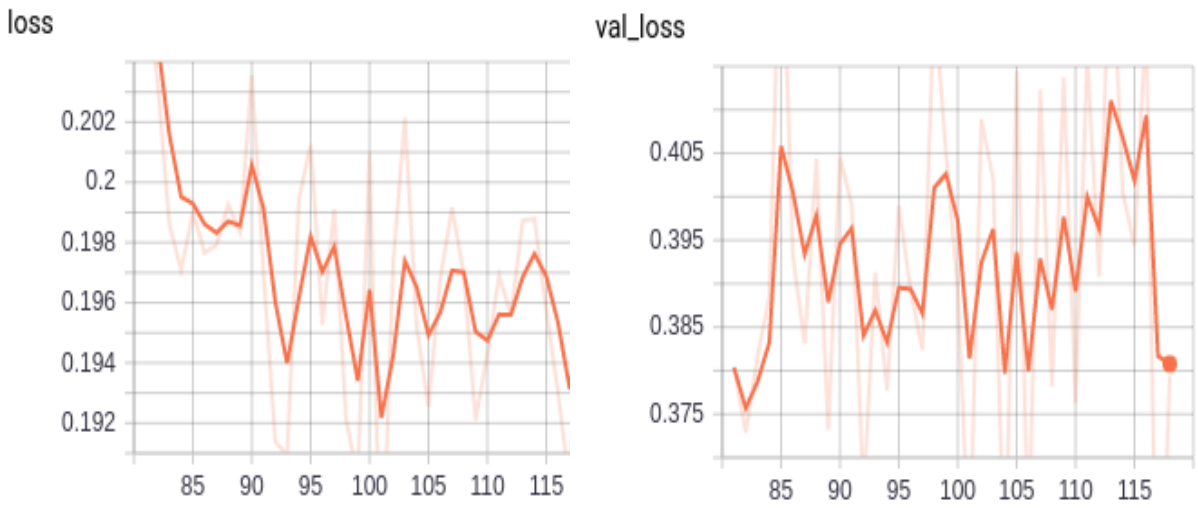
Σχήμα 4.3: training και validation loss για 1-100 εποχές

Φαίνεται ότι έχουμε από κάποιο σημείο και μετά αύξηση στο σφάλμα του validation set για αυτό και για να αποφύγουμε την υπερ προσαρμογή σταματάμε την εκπαίδευσή στην εποχή 75. Από την 76 εποχή και μετά συνεχίζουμε με ρυθμό εκμάθησης το 1/10 του αρχικού. Στα παρακάτω σχήματα έχουμε ξανά τα σφάλματα.



Σχήμα 4.4: training και validation loss για 76-120 εποχές **(overfitting)**

Βλέπουμε οτι ξανά το σφάλμα στο validation set αρχίζει να μεγαλώνει αρκετά οπότε σταματάμε ξανά στην 81 εποχή και από εκεί εκπαιδεύουμε με 1/100 ρυθμό εκμάθησης μέχρι να βρούμε μια παραμετροποίηση που δίνει όσο πιο μικρό λάθος μπορούμε και στα 2 σύνολα.

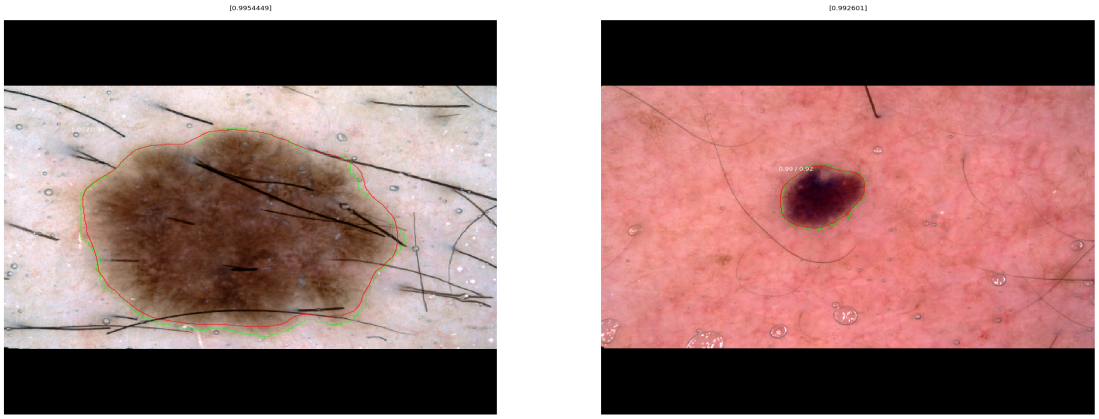


Σχήμα 4.5: training και validation loss για 82-118 εποχές (τελική επιλογή εποχή 118)

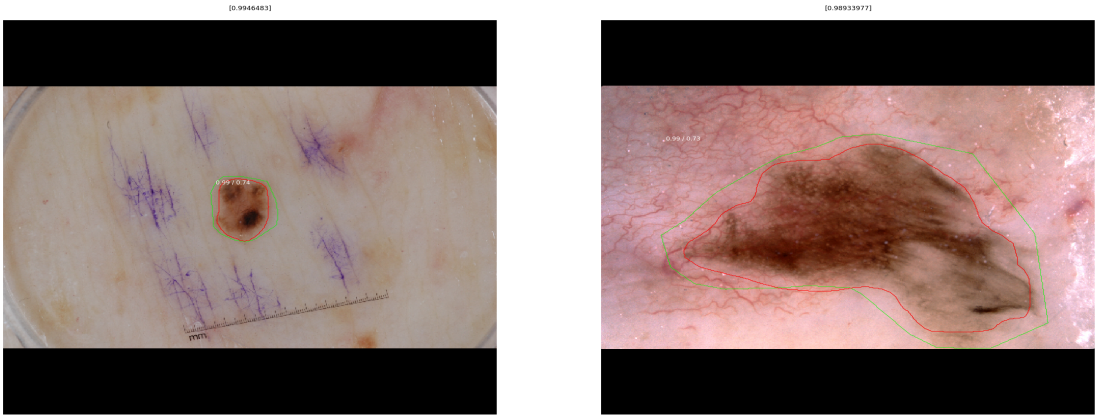
Χρησιμοποιούμε την εποχή 118 για να εκτελέσουμε την αξιολόγηση.

Αξιολόγηση

Για την αξιολόγηση του μοντέλου χρησιμοποιούμε το Jaccard Index που δεν είναι άλλο από την τομή προς ένωση (Intersection over Union) της πρόβλεψης μάσκας με την ground truth μάσκα στο μέσο όρο για όλες τις εικόνες. Σε αυτό το πρόβλημα έχουμε μόνο μια κλάση και μας ενδιαφέρει ιδιαίτερα η ακρίβεια της μάσκας μιας και περιμένουμε το μοντέλο να πετυχαίνει σχεδόν τέλεια απόδοση στην ανίχνευση σπίλου (IoU > 0.5 του ορθογωνίου οριοθέτησης και average precision κοντά στο 1). **Το μέσο IoU στο σύνολο αξιολόγησης είναι 0,8156615**. Το αποτέλεσμα αυτό είναι ανέλπιστα καλό για τα δεδομένα του υλικού που έγινε η εκπαίδευση (χαμηλό batch size, μείωση της ανάλυσης κτλπ).Μερικές εικόνες με τις αντίστοιχες προβλεπόμενες μάσκες και τις πραγματικές τους παρουσιάζονται παρακάτω.



Σχήμα 4.6: κόκκινο η πρόβλεψη,πράσινο τα πραγματικά όρια, labels τα score/IoU



Σχήμα 4.7: κόκκινο η πρόβλεψη,πράσινο τα πραγματικά όρια, labels τα score/IoU

4.2.3 Mask R-CNN με FPN και backbone ResNet101

Εκτελούμε ξανά την διαδικασία της εκπαίδευσης χρησιμοποιώντας ως backbone το δίκτυο Resnet101 που έχει περίπου διπλάσια στρώματα από το Resnet50. Περιμένουμε καλύτερη απόδοση μιας και έχουμε καλύτερη σημασιολογική ικανότητα στην εξαγωγή χαρακτηριστικών. Το configuration των υπερ παραμέτρων παρουσιάζεται στον Πίνακα ενώ στον Πίνακα 4.3 η προ επεξεργασία των εικόνων πριν την αποστολή τους στο pipeline του μοντέλου.

Backbone	Resnet101
Detection minimum confidence	0.01
Image minimmum/maximum dimension	640/896
Lenght of square anchor side	16,32,64,128,256
NMS threshold for RPN proposals	0.5
Anchors / image for RPN training	256
Images per GPU (batch size)	2 during head training / 1 during end to end
Use of mini-masks	True
Use of augmentation	True
mini batch	2 και 1 ***
Maximum number of ground truth instances per image	1

Πίνακας 4.3: Mask R-CNN configuration

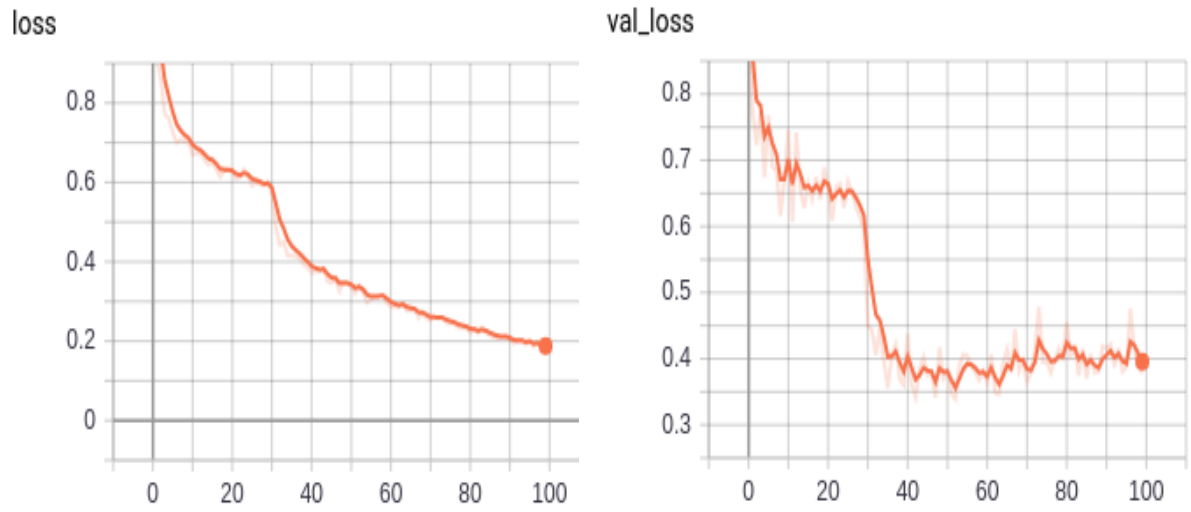
***Δυστυχώς λόγω ελλειπής μνήμης της gpu (1060gtx 6gb) στην εκπαίδευση του μοντέλου από άκρη σε άκρη αναγκαστικά το batch size μειώθηκε σε 1 από 2 λόγω out of memory error.

flips	horizontal or vertical
rotations	90,180,270
scaling	0.8,1.5
Gaussian blurring	s=0.0 ή s=1.5

Πίνακας 4.4: Data augmentation

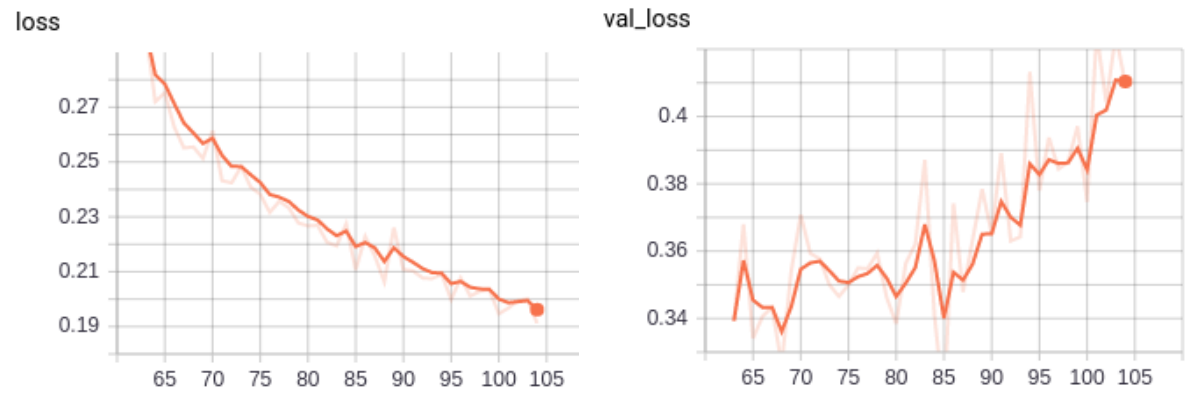
Εκπαίδευση

Παρόμοια με πριν εκπαιδεύουμε το δίκτυο για 30 εποχές όπου αλλάζουμε τα βάρη μόνο της κεφαλής του μοντέλου δηλαδή το RPN, R-CNN και το Mask branch με ρυθμό εκμάθησης l=0.001 και momentum=0.9. Έπειτα εκπαιδεύουμε για 70 (30 μέχρι 100) εποχές ολόκληρο το δίκτυο από άκρη σε άκρη όπου παρατηρούμε ότι το από την εποχή 63 και μετά έχουμε αύξηση αντί για μείωση του σφάλματος στο validation set. Αυτό είναι μια καλή ένδειξη ότι πρέπει να λάβουμε μέτρα για την περίπτωση του overfitting.



Σχήμα 4.8: training και validation loss για 1-100 εποχές

Απορρίπτουμε λοιπόν τις εποχές 64-100 και ξανα-εκπαιδεύουμε από την 63 με ρυθμό εκμάθησης 1/10 του αρχικού.

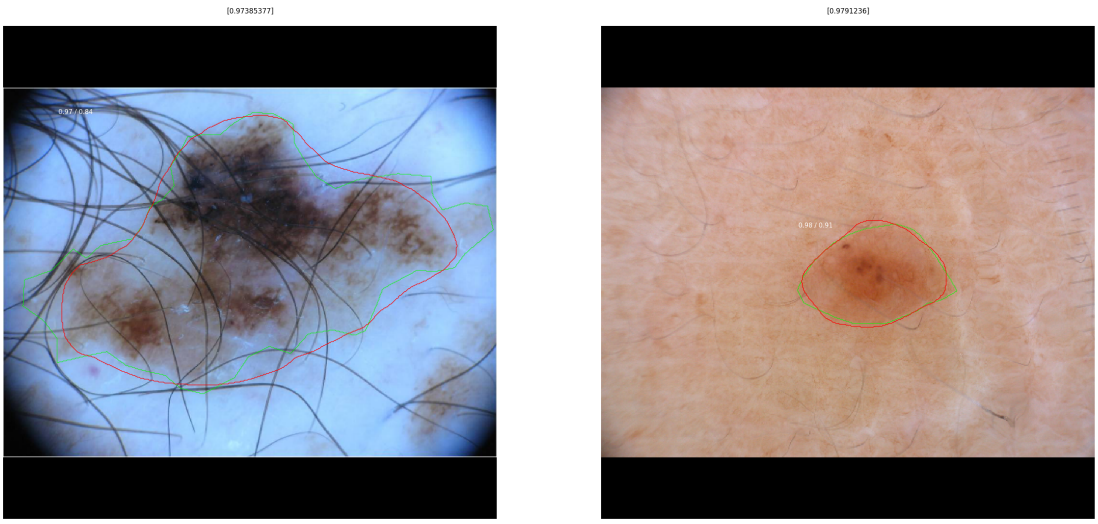


Σχήμα 4.9: training και validation loss για 64-100+ εποχές **(overfitting)**

Πάλι για τον ίδιο λόγο μειώνουμε τον ρυθμό εκμάθησης στην εποχή 86 και μετά στο 1/100 του αρχικού . Στην εποχή 93 καταλήγουμε με μια παραμετροποίηση που δίνει το λιγότερο σφάλμα που έχουμε δει μέχρι στιμγής και στο validation και στο training οπότε και επιλέγουμε αυτή για την φάση της αξιολόγησης. Επισημαίνεται ότι για αρκετό μέρος της εκπαίδευσης τρέχαμε πρακτικά stohastic gradient descent λόγω μνήμης οπότε η σύγκλιση του μοντέλου θα μπορούσε να ήταν καλύτερη υπο άλλες συνθήκες.

Αξιολόγηση

Με τα νέα δεδομένα καταφέρνουμε **να πετύχουμε μέσο IoU ίσο με 0.8076356** . Δηλάδη έχουμε παρόμοια με ελάχιστα χειρότερη απόδοση από το μοντέλο με ResNet50. Αποδίδουμε αυτό το φαινόμενο σε αδυναμία σωστής εκπαίδευσης του μοντέλου με ResNet101 παρά σε κάποιο εγγενές χαρακτηριστικό στην αρχιτεκτονική.



Σχήμα 4.10: κόκκινο η πρόβλεψη,πράσινο τα πραγματικά όρια, labels τα score/IoU

4.2.4 Segmentation με DeepLab v3+

Εδώ θα χρησιμοποιήσουμε την καινούρια έκδοση του μοντέλου της Google για να απομωνόσουμε τον σπίλο από το παρασκήνιο. Το Deeplab εκτελεί απλά σημασιολογική κατάτμηση της εικόνας αλλά για την περίπτωση μας αρκεί. Μάλιστα στον τομέα αυτό πετυχαίνει καλύτερα αποτελέσματα σε μικροτερους χρόνους σύγκλισης από το Mask R-CNN το οποίο βέβαια είναι πιο δυνατό μοντέλο αφού μπορεί και εκτελεί object detection , που για μας ισοδυναμεί με την ικανότητα να βρίσκει πολλαπλούς διαφορετικούς σπίλους κάθε ένας με διαφορετικό ορθογώνιο οριοθέτησης κάτι που δεν εμφανίζεται στα δεδομένα μας.

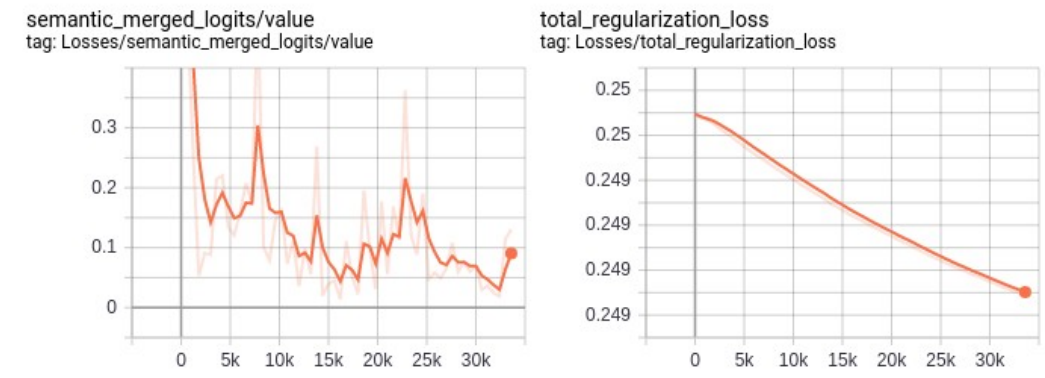
Πάλι επιλέγουμε να μην εκπαιδεύσουμε το δίκτυο από το μηδέν αλλά να χρησιμοποιήσουμε τα έτοιμα βάρη που προέκυψαν από εκπαίδευση της εκδοχής που χρησιμοποιούμε στο MS-COCO dataset. Παρακάτω παρουσιάζονται διάφορες παράμετροι του δικτύου που επιλέχθηκαν για την διαδικασία της εκπαίδευσης.

Backbone	Xception65
Atrous rates	6,12,18,16
Decoder output stride	4
Train crop size	513,513
Train batch size=4	2
Fine tune batch norm	false
Last layers contain logits only	false
Total number of steps	89750
Use of augmentation	True

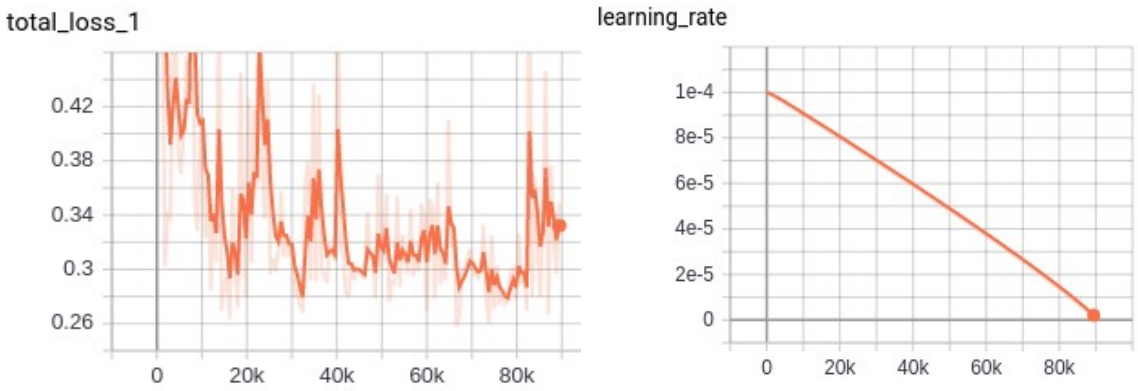
Πίνακας 4.5: Deep Lab configuration

Εκπαίδευση

Αφήνουμε την εκπαίδευση για περίπου 90000 βήματα. Το Deeplab δεν δίνει εύκολα την δυνατότητα για χρήση validation set στην τωρινή του υλοποίηση αλλά χρησιμοποιούνται διάφορες τεχνικές για αποφυγή της υπερπροσαρμογής. Για παράδειγμα όπως παρουσιάζεται και παρακάτω υπάρχει ένας επιπλέον όρος στην συνάρτηση σφάλματος που ονομάζεται regularization loss και στην ουσία τιμωρεί παραμετροποιήσεις με βάρη όπου έχουν μεγάλες τιμές. Κάτι τέτοιο έχει δειχθεί ότι βοηθάει στην αποφυγή της υπερπροσαρμογής. Τέλος είναι αξιοσημείωτο ότι το Deeplab σε σχέση με το Mask R-CNN χρειάστηκε αρκετά λιγότερο χρόνο για να συγκλίνει σε καλή παραμετροποίηση.



Σχήμα 4.11: Losses για Deeplab



Σχήμα 4.12: Total training loss και learning rate για Deeplab

Αξιολόγηση

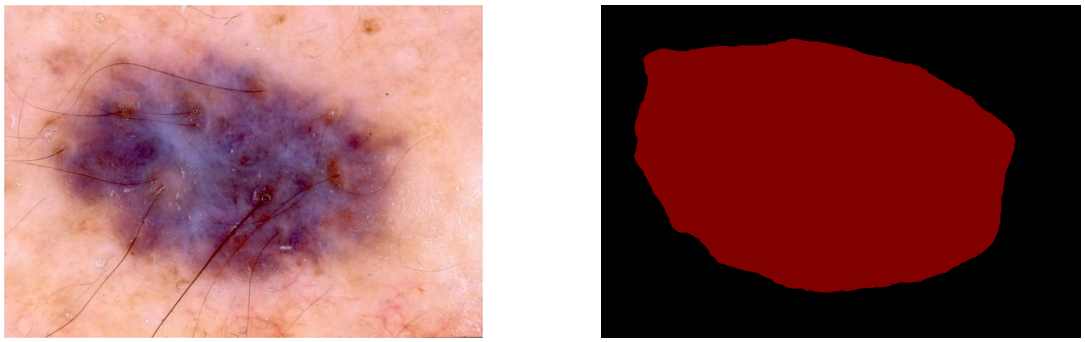
Το μέσο IoU στο σύνολο αξιολόγησης είναι 0,9007. Έχουμε δηλαδή μια σαφή βελτίωση σε σχέση και με τις δύο εκδοχές του Mask R-CNN κάτι που περιμέναμε. Παρακάτω παρουσιάζονται συγκεντρωτικά τα αποτελέσματα για όλα τα μοντέλα στο πρόβλημα του segmentation. Σημειώνεται όμως ότι για τεχνικούς λόγους το inference οπως και το training γίνεται πάνω σε εικόνες ανάλυσης 512x512. Το IoU μετρήθηκε έναντι της resized μάσκας με τη βιβλιοθήκη Pillow της python. Ενδέχεται να είναι ελαφρώς χειρότερο αν εκτελέσουμε upsample της πρόβλεψης στις αρχικές διαστάσεις και μετά συγκρίνουμε με την ground truth μάσκα χωρίς όμως να αλλάζει κάτι ουσιαστικό στο αποτέλεσμα της απόδοσης.

Μοντέλο	**Μέσο IoU**
Mask R-CNN w/ Resnet50	0,8156615
Mask R-CNN w/ Resnet101	0.8076356
DeepLab w/ Xception	0,9007

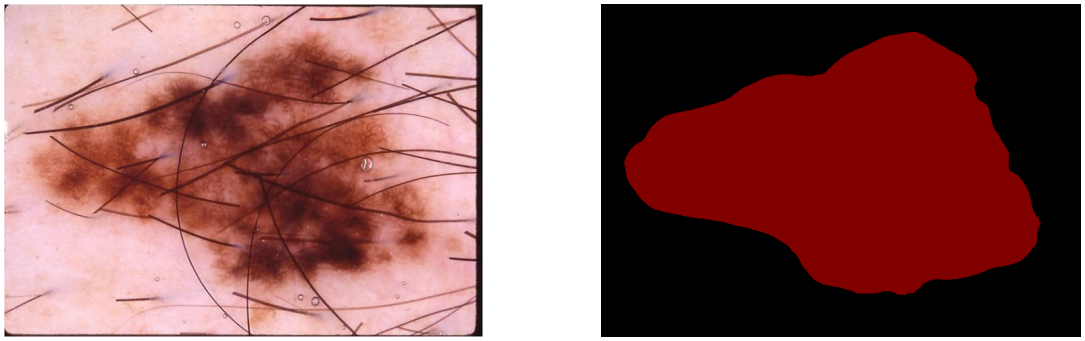
Πίνακας 4.6: Συγκεντρωτικά αποτελέσματα για segmentation

Τέλος δίνονται και μερικές εικόνες με τις μάσκες της πρόβλεψης από τα αποτελέσματα του Deep Lab στο test set.

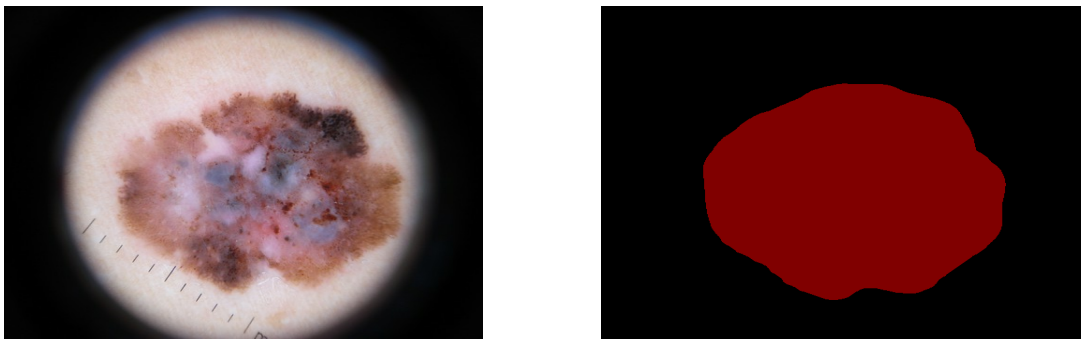
Σχήμα 4.13: Πρόβλεψη του Deep Lab (1)



Σχήμα 4.14: Πρόβλεψη του Deep Lab (2)



Σχήμα 4.15: Πρόβλεψη του Deep Lab (3)



Σχήμα 4.16: Πρόβλεψη του Deep Lab (4)

4.3 Ταξινόμηση δερματικών σπίλων σε 8 κλάσεις

Σε αυτό το σημείο θα χρησιμοποιήσουμε την τεχνολογία των ResNet δικτύων για να επιχειρήσουμε να λύσουμε το πρόβλημα της ταξινόμησης των δερματικών σπίλων σε διάφορες κλάσεις ανάλογα το είδος τους. Η ταξινόμηση αυτή είναι πολύ σημαντική για την σωστή πρόβλεψη και μπορεί να λειτουργήσει ως υποβοήθεια για τους γιατρούς και τους δερματολόγους αλλά και τους απλούς χρήστες που μπορούν να ενημερωθούν από αυτή για το αν πρέπει να υποβληθούν σε ιατρικές εξετάσεις.

Γενικά σε τέτοιες εφαρμογές όπως στην ιατρική πρέπει να είμαστε πολύ προσεκτικοί ποια μετρική θα χρησιμοποιήσουμε για την αξιολόγηση του μοντέλου. Συνήθως οι κλάσεις που δηλώνουν κάποια παθογένεια περιέχουν πολύ λιγότερα στοιχεία από τις κλάσεις που δηλώνουν καλοήθεια. Έτσι είναι εύκολο να φτιάξουμε biased ταξινομητές με πολύ καλή ακρίβεια που όμως είναι ανίκανοι να εντοπίσουν τα στοιχεία της παθογενούς κλάσης. Μάλιστα επειδή το κόστος τέτοιας αστοχίας είναι πολύ μεγάλο προτιμούμε την αστοχία στις καλές κλάσεις μιας και αυτό σημαίνει ότι ένας υγιής ασθενής απλά θα υποβληθεί σε περαιτέρω εξετάσεις. Αντίθετα αστοχία στις κακές κλάσεις σημαίνει ότι αποτύχαμε να εντοπίσουμε μια ασθένεια το οποίο έχει καταστροφικές συνέπειες για τον ασθενή.

4.3.1 Τα δεδομένα

Έχουμε στη διάθεση μας 23332 εικόνες από τις παρακάτω 8 κατηγορίες.

1) **AKIEC: “Actinic keratosis / Bowen’s disease (intraepithelial carcinoma)”** (867 εικόνες)

2) **BCC: “Basal cell carcinoma”** (3323 εικόνες)

3) **BKL: “Benign keratosis ”** (2714 εικόνες)

4) **DF: “Dermatofibroma”**(239 εικόνες)

5) **MEL: “Melanoma”** (4522 εικόνες)

6) **NV: “Melanocytic nevus”**(10875 εικόνες)

7) **VASC: “Vascular lesion”** (253 εικόνες)

8) **SCC: “Squamous cell carcinoma”** (628 εικόνες)

Από το σύνολο των εικόνων χωρίζουμε σε 80% (για training και validation sets) και 20% για test set. Έπειτα από αυτό το 80% χωρίζουμε σε αναλογία 80% για το training set και 20% για το validation set. Στους διαχωρισμούς αυτούς σεβόμαστε την αναλογία των κλάσεων και τη διατηρούμε σε όλα τα σετ. Όμως βλέπουμε ότι οι κλάσεις είναι εξαιρετικά μη ισορροπημένες που σημαίνει ότι θα οδηγηθούμε σε biased model αν προχωρήσουμε στην εκπαίδευση έτσι.
Αποφασίζουμε να εκτελέσουμε upsampling στις κλάσεις με τα λίγα στοιχεία πολλαπλασιάζοντας τις εμφανίσεις κάθε εικόνας αντιγράφοντας την έπειτα από κάποιο augmentation όπως αυτό που θα περνάνε όλες οι εικόνες στο batch πριν την εκπαίδευση.

Για να δημιουργήσουμε τα μοντέλα χρησιμοποιούμε το Keras με backend το Tensorflow καθώς και την αρχιτεκτονική των δικτύων που αναφέρθηκε στην αντίστοιχη ενότητα. Χρησιμοποιούμε την κλάση image generator για να φορτώνουμε τις εικόνες σταδιακά σε batches μιας και είναι αδύνατο να φορτωθούν όλες μαζί στην μνήμη. Τέλος επιλέγουμε να τις κάνουμε όλες resize σε ανάλυση 448x448 και για λόγους μνήμης αλλά και γιατί αυτή η ανάλυση είναι κοντά στην μικρότερη διαστάση που εμφανίζεται στο σύνολο μας.

4.3.2 Χρήση Resnet50

Δημιουργούμε το μοντέλο σύμφωνα με την καθιερωμένη αρχιτεκτονική και ρυθμίζουμε την διαδικασία της εκπαίδευσης με τις παραμέτρους στον παρακάτω πίνακα.

Network	Resnet50
input size	(448,448,3)
batch size	32
optimizer	adam
learning rate	0.001
Reduce Learning Rate on Plateu	True
RL RoP value monitored	validation loss
Rl RoP patience	6 epochs
Rl RoP factor	0.5
Use of augmentation	True

Πίνακας 4.7: ResNet50 training configuration

Όπως φαίνεται από τον πίνακα επιλέγουμε να μειώνουμε τον ρυθμό εκμάθησης κάθε φορά που το λάθος στο validation set σταματά να βελτιώνεται για τις επόμενες 6 εποχές. Για την επεξεργασία των εικόνων πριν από την διοχέτευση τους στο δίκτυο εκτελούμε τυχαία συνδυασμούς των παρακάτω ενεργειών εκτος απο centering και normalization που εφαρμόζεται πάντα.

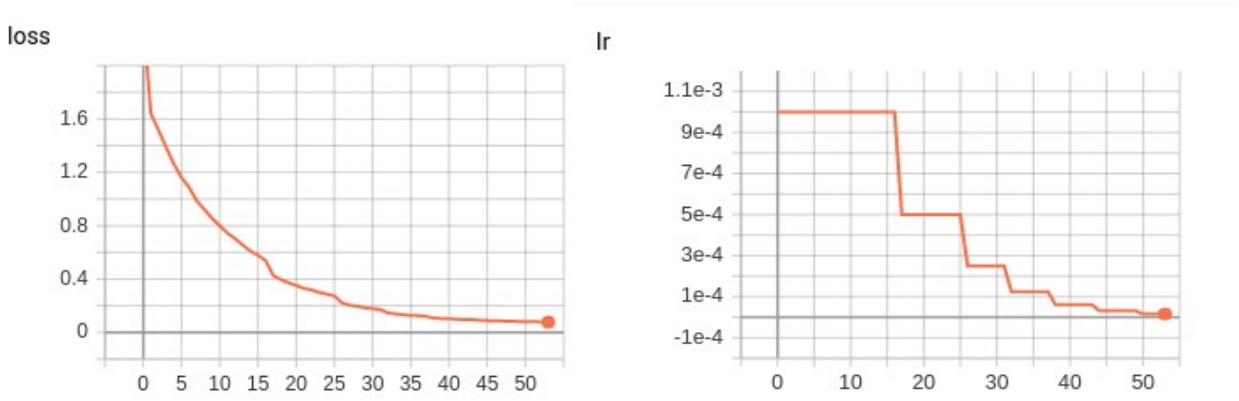
flips	horizontal or vertical
rotations	90,180,270
zoom	0.8-1.2
width slide	0.2
height slide	0.2
featurewise centering	True
featurewise std normalization	True

Πίνακας 4.8: Data augmentation for ResNet50

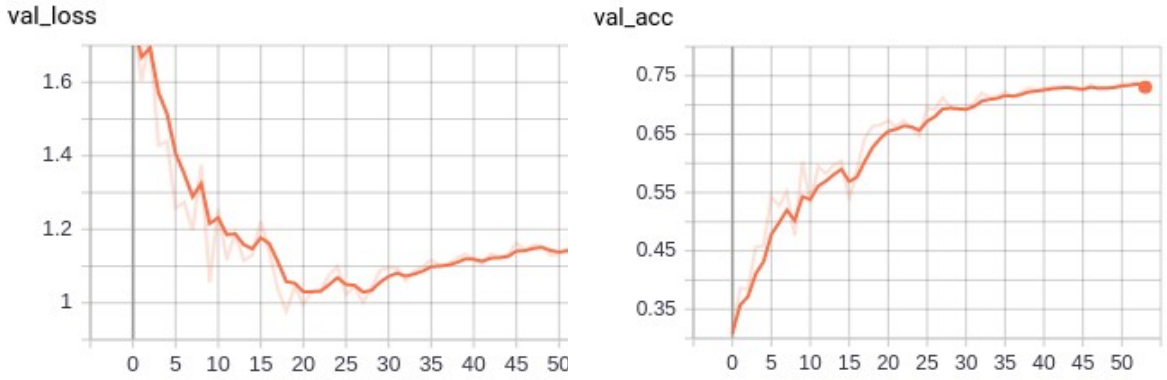
Με feature-wise-centering και feature-wise-std-normalization εννοούμε την κανονικοποίηση που πετυχαίνουε με αφαίρεση της μέσης τιμής των εικονοστοιχείων για κάθε κανάλι χρώματος ξεχωριστά και διαίρεση με την τυπική απόκλιση ,όπως αυτές οι τιμές υπολογίζονται για 75 τυχαία δείγματα. Μάλιστα επαναλαμβάνουμε την διαδικασία του υπολογισμού 10 φορές για διαφορετικά σύνολα 75 δειγμάτων ώστε να μειώσουμε την διακύμανση στον υπολογισμό αυτών των τιμών και να έχουμε καλύτερη εκτίμηση για τη μέση τιμή και τη διακύμανση των εικονοστοιχείων του συνόλου δεδομένων ανά χρώμα. Τέλος επιλέγουμε να μην εκτελέσουμε κάποιο μετασχηματισμό στο χρώμα της εικόνας μιας και φαίνεται ότι πολλές φορές αυτό αποτελεί κριτήριο διαχωρισμού των κλάσεων, πέρα από το σχήμα.

Εκπαίδευση

Αφήνουμε την εκπαίδευση για περίπου 50 εποχές και λαμβάνουμε τις παρακάτω γραφικές .



Σχήμα 4.17: training loss και learning rate στην εκπαίδευση resnet50



Σχήμα 4.18: validation loss και validation accuracy στην εκπαίδευση resnet50

Παρατηρούμε ότι το σφάλμα στο validation set από ένα σημείο και μετά αρχίζει να αυξάνεται καθώς προσεγγίζουμε ένα τοπικό ελάχιστο με μικρό ρυθμό εκμάθησης.
Ταυτόχρονα όμως το η ακρίβεια συνεχίζει να αυξάνεται. Βέβαια επειδή το validation set συνεχίζει να κουβαλάει της ανισορροπίες στις κλάσεις του αρχικού συνόλου δεδομένων η ακρίβεια δεν είναι ιδιαίτερα καλή μετρική. Παρά ταύτα στο training set έχει γίνει υπερδειγματοληψία στις κλάσεις με λίγα στοιχεία και δεν περιμένουμε biased μοντέλο.

Αξιολόγηση

Παρακάτω παραδίδονται συγκεντρωτικά τα αποτελέσματα της απόδοσης του δικτύου στο test set. Φαίνεται ότι δεν έχουμε και ιδιαίτερα καλά αποτελέσματα. Δεν περιμένουμε και τρομερά καλή απόδοση λόγω της δυσκολίας του προβλήματος, τις πολλές κλάσεις και τα υπερβολικά λίγα στοιχεία που μερικές από αυτές έχουν. Θα δοκιμάσουμε να βελτιώσουμε αυτήν την απόδοση παρακάτω.

-	**akiec**	**bcc**	**bkl**	**df**	**mel**	**nv**	**scc**	**vasc**
recall	0.372	0.702	0.466	0.317	0.552	0.579	0.34	0.468
precision	0.423	0.549	0.378	0.30	0.453	0.779	0.409	0.169
average precision	0.359	0.621	0.344	0.274	0.504	0.779	0.297	0.163

Πίνακας 4.9: Αποτελέσματα ανα κλάση στο ResNet50

accuracy	0.562
balanced accuracy	0,454
mean Average Precision	0.418

Πίνακας 4.10: Αποτελέσματα για όλες τις κλάσεις στο ResNet50

4.3.3 Χρήση Resnet101

Επαναλαμβάνουμε τη διαδικασία με χρήση του μοντέλου Resnet101. Δημιουργούμε πάλι το μοντέλο σύμφωνα με την καθιερωμένη αρχιτεκτονική και ρυθμίζουμε την διαδικασία της εκπαίδευσης με τις παραμέτρους και το augmentation που παρουσιάζονται στον παρακάτω πίνακα.

Network	Resnet101
input size	(448,448,3)
batch size	16
optimizer	adam
learning rate	0.001
Reduce Learning Rate on Plateu	True
RL RoP value monitored	validation accuracy
Rl RoP patience	5 epochs
Rl RoP factor	0.5
Use of augmentation	True

Πίνακας 4.11: ResNet101 training configuration

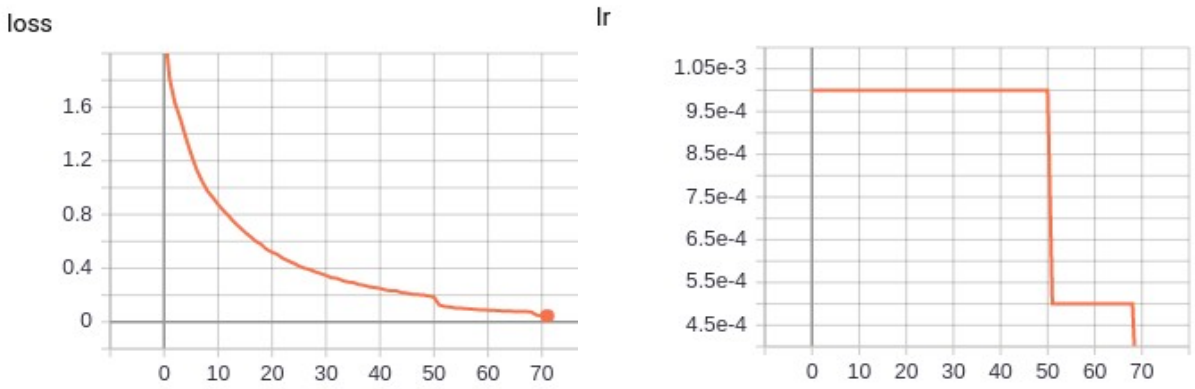
flips	horizontal or vertical
rotations	90,180,270
zoom	0.8-1.2
width slide	0.2
height slide	0.2
featurewise centering	True
featurewise std normalization	True

Πίνακας 4.12: Data augmentation for ResNet101

Μια διαφορά εδώ σε σχέση με πριν είναι το μέγεθος του batch size λόγω μνήμης. Επίσης για τη μείωση του ρυθμού εκμάθησης γίνεται χρήση της μετρικής accuracy. Βέβαια πρέπει να επισημάνουμε ότι ο βελτιστοποιητής Adam μειώνει από μόνος του το ρυθμό ανά βάρος στο δίκτυο και η αλλαγή του ολικού ρυθμού απλά δίνει ένα άνω όριο. Γενικά η μείωση είναι εκεί για να έχουμε κάποια γρηγορότερη σύγκλιση μετά απο 40-50 εποχές οι οποίες απαιτούν αρκετό χρόνο για να τρέξουν.

Εκπαίδευση

Αφήνουμε την εκπαίδευση να τρέξει για περίπου 70 εποχές. Οι γραφικές με την εξέλιξη της εκπαίδευσης παρουσιάζονται παρακάτω.



Σχήμα 4.19: training loss και learning rate στην εκπαίδευση resnet101

Σχήμα 4.20: validation loss και validation accuracy στην εκπαίδευση resnet101

Αξιολόγηση

Τα αποτελέσματα εδώ είναι σαφώς καλύτερα. Βέβαια θα μπορούσαν εν μέρη να αποδοθούν στη μικρότερη παρέμβαση μας στον ρυθμό εκμάθησης το οποίο βέβαια έκανε το δίκτυο να συγκλίνει πιο αργά, αλλά σε παραμετροποιήσεις που γενίκευαν καλύτερα. Και πάλι βέβαια η ακρίβεια στο validation set συνέχιζε να αυξάνει μέχρι το τέλος παρόλο που το δίκτυο γινόταν σχεδόν εντελώς fit στα δεδομένα εκπαίδευσης και το validation loss έπαιρνε την ανιούσα, κάτι που είναι αξιοπερίεργο. Πιθανόν να είναι απόρροια του γεγονότος ότι έχουμε λιγότερα δείγματα στη διάθεση μας από ότι χρειαζόμαστε για να δημιουργήσουμε ένα δίκτυο με πολύ καλή απόδοση σε όλες τις κλάσεις (χρησιμοποιώντας μόνο ένα δίκτυο μόνο του). Επίσης υπάρχουν πάρα πολλά τοπικά ελάχιστα που οδηγούν σε σχεδόν τέλειο fit στο training set σε λιγότερες από 100 εποχές χωρίς να γενικεύουν απαραίτητα.

-	akiec	bcc	bkl	df	mel	nv	scc	vasc
recall	0.554	0.723	0.542	0.414	0.592	0.677	0.627	0.52
precision	0.518	0.664	0.441	0.533	0.527	0.823	0.307	0.654
average precision	0.472	0.723	0.413	0.366	0.570	0.835	0.380	0.538

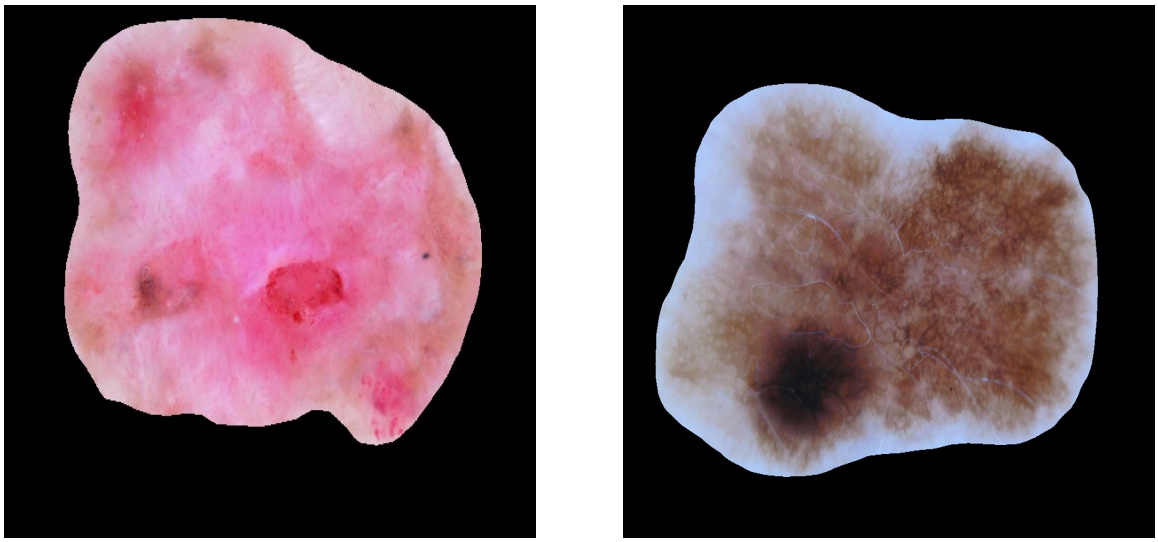
Πίνακας 4.13: Results for ResNet101

accuracy	0.6435
balanced accuracy	0,581
mean Average Precision	0.538

Πίνακας 4.14: average results for ResNet101

4.3.4 Ταξινόμηση μετά από segmentation και απομόνωση της περιοχής

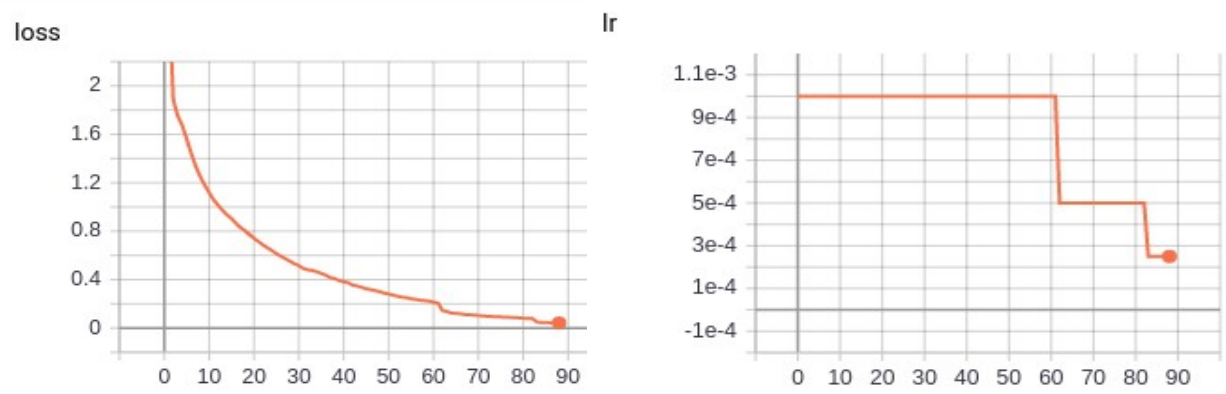
Σε αυτό το στάδιο επαναλαμβάνουμε για μια ακόμα φορά την εκπαίδευση του δικτύου resnet101 αλλά πρώτα περνάμε όλες τις εικόνες από το μοντέλο Mask R-CNN για segmentation της επίμαχης περιοχής. Ο μετασχηματισμός αυτός γίνεται και στα validation set και test set. Στην πολύ σπάνια περίπτωση που το segmentation δίνει κενή μάσκα απλά δίνουμε την ίδια εικόνα. Ο σκοπός μας είναι να δούμε αν η αφαίρεση της περιττής πληροφορίας δηλαδή του υγιούς τμήματος του δέρματος μπορεί να βελτιώσει την απόδοση του δικτύου ταξινόμησης. Τα υπόλοιπα στοιχεία της εκπαίδευσης είναι ίδια με πριν, ενώ προφανώς τα στάδια feature centering και standardization επαναλαμβάνονται για τα νέα δεδομένα.



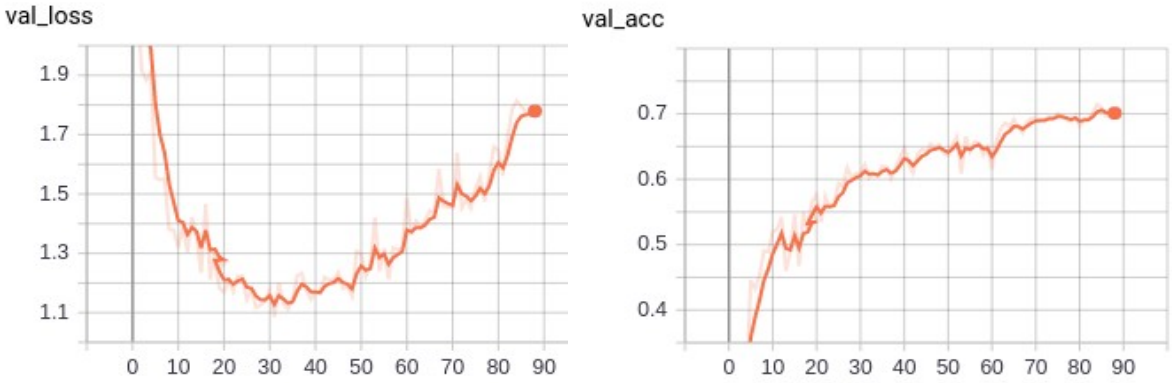
Σχήμα 4.21: εικόνες μετά από segmentation

Εκπαίδευση

Αφήνουμε την εκπαίδευση να τρέξει για περίπου 80-90 εποχές. Οι γραφικές με την εξέλιξη της εκπαίδευσης παρουσιάζονται παρακάτω. Παρατηρούμε ότι έχει υπάρξει overfitting και έτσι επιλέγουμε τελική εποχή ανάλογα.



Σχήμα 4.22: training loss και learning rate



Σχήμα 4.23: validation loss και validation accuracy

Αξιολόγηση

Βλέπουμε γενικά χειρότερα αποτελέσματα από χωρίς segmentation και resnet101. Πιθανόν φταίει το γεγονός ότι αντικαταστήσαμε την περιοχή παρασκηνίου με μηδενικές τιμές και αυτό οδήγησε το δίκτυο στο να επικεντρωθεί περισσότερο στο σχήμα του σπίλου. Το σχήμα δεν είναι πάντα χρήσιμη πληροφορία για το είδος. Βέβαια βλέπουμε ότι κλάσεις όπως Vascular lesion παρουσίασαν αρκετή βελτίωση σε σχέση με πριν.

-	**akiec**	**bcc**	**bkl**	**df**	**mel**	**nv**	**scc**	**vasc**
recall	0,456	0,698	0,431	0,425	0,603	0,716	0,36	0,78
precision	0,465	0,591	0.51	0,571	0.53	0,774	0,391	0,527
average precision	0.435	0.660	0.462	0.332	0.623	0.834	0.34	0.604

Πίνακας 4.15: Αποτελέσματα για Resnet101 και segmentation ανά κλάση

accuracy	0,638
balanced accuracy	0,558
mean Average Precision	0,536

Πίνακας 4.16: Αποτελέσματα για Resnet101 και segmentation

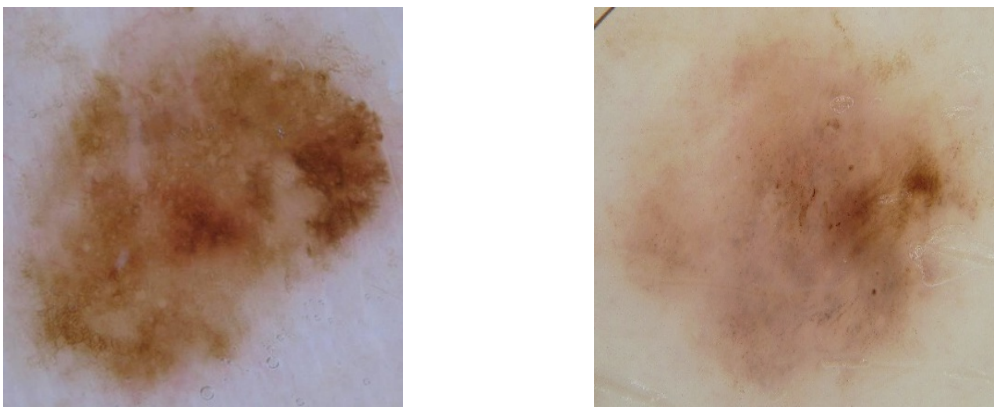
-	akiec	bcc	bkl	df	mel	nv	scc	vasc
akiec	79	38	19	1	18	13	5	0
bcc	33	464	26	1	54	71	15	0
bkl	18	65	226	4	77	120	11	3
df	2	8	3	20	2	10	2	0
mel	13	50	43	1	545	228	17	7
nv	20	118	114	8	313	1557	20	25
scc	5	37	13	0	13	12	45	0
vasc	0	5	0	0	5	1	0	39

Πίνακας 4.17: Confusion matrix για τις κλάσεις

4.3.5 Ταξινόμηση μετά από απομόνωση της παθογενούς περιοχής βάση του ορθογωνίου οριοθέτησης (crops)

Τελικά σε αυτό το στάδιο περνάμε όλες τις εικόνες ξανά από το Mask R-CNN αλλά τώρα χρησιμοποιούμε το bounding box για να δημιουργήσουμε αποκόμματα της εικόνας που περιέχουν μόνο τον σπίλο.

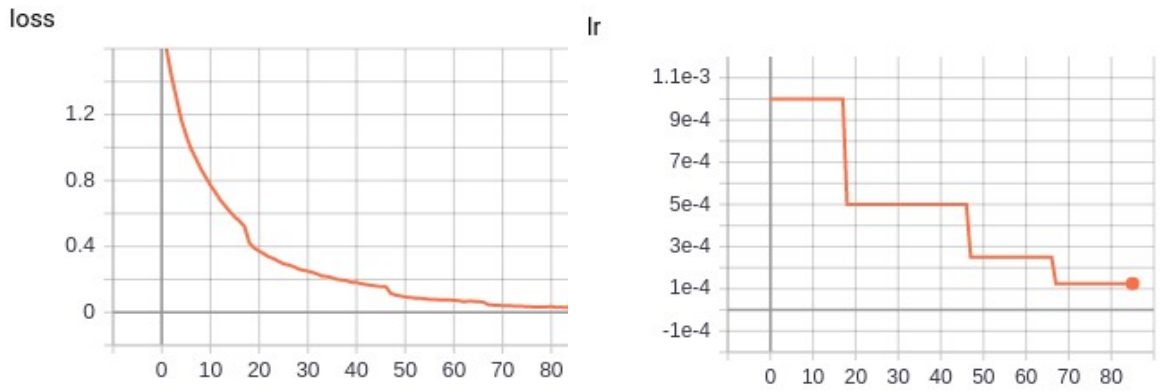
Αυτή η προσέγγιση σε αντίθεση με το segmentation με χρήση μάσκας, δεν δημιουργεί την απότομη αποκοπή στα όρια του σπίλου ενώ επίσης μειώνει γενικά την ανάλυση της εικόνας χωρίς να αφαιρεί χρήσιμη πληροφορία. Εδώ περιμένουμε τουλάχιστον ισάξια απόδοση με αυτή που είχαμε με απλή χρήση του ResNet101 ενώ ιδανικά ευελπιστούμε για κάποια αύξηση όλων των μετρικών



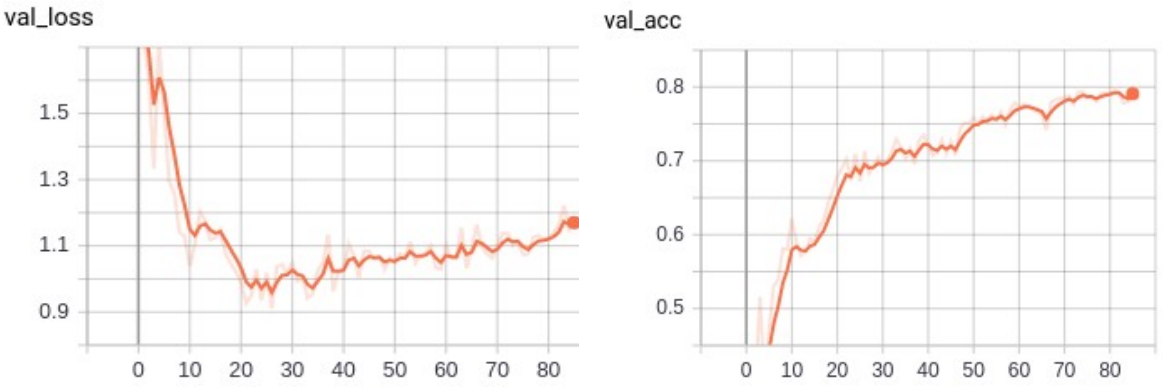
Σχήμα 4.24: Cropped images

Εκπαίδευση

Αφήνουμε το δίκτυο να εκπαιδευτεί με τις ίδιες υπέρ παραμέτρους ακριβώς όπως και στις άλλες φορές. Τα στάδια του centering και normalization επαναλαμβάνονται για τα νέα δεδομένά/εικόνες. Σημειώνεται ότι επανεκτελέσαμε τον διαχωρισμό των εικόνων σε σύνολα training,testing και validation και πάλι τυχαία, διατηρώντας όμως τις αναλογίες στις κλάσεις. Επίσης εκτελέστηκε η ίδια διαδικασία upsampling για τις κλάσεις με λίγα στοιχεία όπως και πριν.



Σχήμα 4.25: training loss και learning rate



Σχήμα 4.26: validation loss και validation accuracy

Αξιολόγηση

Όπως αναμέναμε εδώ παίρνουμε καλύτερα αποτελέσματα για όλες σχεδόν τις κλάσεις κάτι που αυξάνει αρκετά τη μέση απόδοση του δικτύου. Σημειώνεται εδώ ότι χρησιμοποιήσαμε συνολικά 12875 για την κλάση melanocytic nevus δηλαδή 2000 περισσότερες από πριν και για αυτό και έχουμε περισσότερα συνολικά δείγματα για αυτή την κλάση και στο σύνολο αξιολόγησης. Βέβαια αυτό δεν επηρεάζει στην αξιολόγηση την μετρική recall των άλλων κλάσεων αλλά επηρεάζει ελαφρώς τις precision και average precision. Υπενθυμίζεται τέλος ότι balanced accuracy είναι ουσιαστικα ο μέσος όρος των recall για όλες τις κλάσεις και είναι η καλύτερη μετρική για το πρόβλημα μας.

-	akiec	bcc	bkl	df	mel	nv	scc	vasc
recall	0,479	0,789	0,517	0,659	0,609	0,782	0,432	0,74
precision	0,522	0,618	0,494	0,659	0,613	0,835	0,580	0,649
average precision	0.473	0.733	0.53	0.618	0.67	0.886	0.499	0.697

Πίνακας 4.18: Αποτελέσματα για Resnet101 και cropping ανά κλάση

accuracy	0,704
balanced accuracy	0,625
mean Average Precision	0,638

Πίνακας 4.19: Αποτελέσματα για Resnet101 και cropping

-	akiec	bcc	bkl	df	mel	nv	scc	vasc
akiec	83	37	19	0	18	14	2	0
bcc	24	524	20	3	25	55	8	5
bkl	17	52	271	1	68	110	4	1
df	0	4	3	31	3	4	1	1
mel	9	71	51	4	551	207	5	6
nv	12	125	170	7	221	2015	19	6
scc	14	30	13	0	10	3	54	1
vasc	0	4	1	1	3	4	0	37

Πίνακας 4.20: Confusion matrix για τις κλάσεις

Κεφάλαιο 5

Επίλογος-Συμπεράσματα

5.1 Σύνοψη

Στην παρούσα εργασία κάναμε μια λεπτομερή επισκόπηση στις τεχνικές βαθιάς μηχανικής μάθησης για ανάλυση εικόνας με σκοπό να εφαρμόσουμε αυτές τις τεχνολογίες σε ένα πρόβλημα ιατρικής σημασίας.

Αρχικά και αφού παρουσιάσαμε τα διάφορα μοντέλα που χρησιμοποιήσαμε, προσπαθήσαμε να μεγιστοποιήσουμε την απόδοση στο πρόβλημα του προσδιορισμού του περιγράμματος του δερματικού σπίλου σε εικόνες που προκύπτουν από δερμοσκόπηση. Το περίγραμμα προδίδει χρήσιμες πληροφορίες και αποτελεί ένα πρώτο στάδιο για την επεξεργασία της εικόνας.

Για την κατάτμηση χρησιμοποιήσαμε το Mask R-CNN για δύο διαφορετικά backbone δίκτυα εξαγωγής χαρακτηριστικών και συγκρίναμε την επίδοση. Καταλήξαμε σε αρκετά καλά αποτελέσματα αν και είχαμε σοβαρούς περιορισμούς σε μνήμη ενώ είδαμε πόσο σημαντική είναι η εκπαίδευση με transfer learning για να πετύχουμε σύγκλιση ειδικά σε καταστάσεις χαμηλής μνήμης και επεξεργαστικής ισχύς. Έπειτα εκπαιδεύσαμε το DeepLab V3 το οποίο είναι αρκετά πιο γρήγορο μοντέλο πάλι με χρήση transfer learning. Το DeepLab πέτυχε σύγκλιση αρκετά πιο γρήγορα από το Mask R-CNN ενώ το γεγονός ότι εκτελεί μόνο κατάτμηση εικόνας δεν μας επηρέασε στο πρόβλημα μας αφού έχουμε ένα σπίλο ανά εικόνα.

Στη συνέχεια δοκιμάσαμε να εκπαιδεύσουμε τα Residual Networks πάνω στο πρόβλημα της ταξινόμησης σπίλων σε 8 κατηγορίες ιατρικής σημασίας. Βρεθήκαμε αντιμέτωποι με το πρόβλημα της μη ισορροπημένης κατανομής των κλάσεων στο σύνολο δεδομένων. Μάλιστα είχαμε κλάσεις με δύο τάξεις μεγέθους μικρότερο πλήθος από την κυρίαρχη. Επιχειρήσαμε να λύσουμε το πρόβλημα αυτό με τεχνικές upsampling και augmentation. Δοκιμάσαμε τα ResNet50 και Resnet101 για ταξινόμηση και είδαμε την υπεροχή των περισσότερων στρωμάτων όταν υπάρχει residual σύνδεση. Τέλος επιχειρήσαμε να επανεκπαιδεύσουμε το δίκτυο ResNet101 στις δύο παρακάτω περιπτώσεις.

1. Εφαρμόσαμε κατάτμηση της εικόνας με εφαρμογή της μάσκας που υπολογίζει το Mask R-CNN, δηλαδή αντικαθιστώντας το παρασκήνιο με μηδενικές τιμές / μαύρο χρώμα. Είδαμε ότι αν και αυτή η προσέγγιση βελτίωσε την απόδοση σε ορισμένες κλάσεις γενικά μείωσε τη συνολική απόδοση του δικτύου. Θεωρούμε πως η απότομη αποκοπή στα όρια του σπίλου οδήγησε το δίκτυο στο να μαθαίνει το περίγραμμα για να κάνει προβλέψεις κάτι το οποίο δεν είναι πάντα αντιπροσωπευτικό της κλάσης που ανήκει ο σπίλος.

2. Απομονώσαμε τον σπίλο με cropping στην επίμαχη περιοχή με σκοπό και πάλι να κρατήσουμε την χρήσιμη πληροφορία χωρίς την απότομη αποκοπή που είχαμε με χρήση μάσκας. Αυτή η προσέγγιση πετυχαίνει το κεντράρισμα του σπίλου στις εικόνες καθώς και μειώνει την ανάλυση των εικόνων χωρίς να χαθεί πολύτιμη πληροφορία. Έτσι το δίκτυο μπορεί εξειδικευτεί περισσότερο στην ταξινόμηση σπίλων κάθε φορά στο ίδιο σημείο της εικόνας. Είδαμε εντέλει ότι πήραμε καλύτερα αποτελέσματα από κάθε άλλη περίπτωση.

5.2 Παρατηρήσεις-Επέκταση της εργασίας

Καθ' όλη την πειραματική φάση της εργασίας υπήρχαν σοβαροί περιορισμοί σε υπολογιστικούς πόρους που οδήγησαν σε ρυθμίσεις υπέρ παραμέτρων που δεν ήταν οι ιδανικές. Με χρήση transfer learning μπορέσαμε να αντισταθμίσουμε την έλλειψη πόρων αλλά για να εφαρμοστούν αυτές οι τεχνολογίες επιτυχώς και να επιτευχθεί μέγιστη απόδοση στα προβλήματα που επιχειρήσαμε να λύσουμε, ιδανικά χρειάζονται περισσότεροι πόροι.

Επίσης όσον αφορά το στάδιο της ταξινόμησης των σπίλων θα μπορούσαμε να πετύχουμε καλύτερα αποτελέσματα χρησιμοποιώντας διάφορες νέες τεχνικές που εφαρμόζονται σήμερα για τέτοια δύσκολα προβλήματα. Κάποιες από αυτές είναι οι παρακάτω:

- Upsampling με συνθετικές εικόνες ή χρήση βαρών ανά κλάση στον υπολογισμό της κλίσης μέσα στο batch.
- Υψηλότερη ανάλυση των 448x448 εικονοστοιχείων που χρησιμοποιήθηκε και μεγαλύτερου batch size (περιορισμοί σε υλικό).
- Δοκιμή περισσότερων μοντέλων και για διάφόρες εκδοχές της αρχιτεκτονικής τους (βλ. DensNet)
- Χρήση ensemble μοντέλων ή και stacked δηλαδή εκπαίδευση πολλών μοντέλων ξεχωριστά και επιπλέον εκπαίδευση ενός δικτύου για την επιλογή του δικτύου που θα κάνει την πρόβλεψη κάθε φορα.
- Δοκιμή διαφορετικών optimizers πέρα από τον Adam.
- Χρήση εξωτερικής πληροφορίας για τις εικόνες του συνόλου δεδομένων για την κατανομή τους στο σύνολο της εκπαίδευσης σε αντίθεση με την τυχαία υπερδειγματοληψία καθώς και χειροκίνητη επεξεργασία τους όπως απομάκρυνση τριχών και φυσαλίδων.
- Δοκιμή πολλών ταξινομητών, έναντι ενός για όλες τις κλάσεις, που εκπαιδεύονται σε ταξινόμηση σε διάφορα στάδια για κλάσεις υπερσύνολα των πραγματικών. Για παράδειγμα ταξινόμηση σε δύο διαφορετικές κλάσεις που αντιστοιχούν η καθεμία στις μισές αρχικές κλάσεις από το σύνολο και έπειτα διοχέτευση της εικόνας για νέα ταξινόμηση για να προσδιορίσουμε περαιτέρω την κλάση στόχο.

Βιβλιογραφία

[Abad15] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu and Xiaoqiang Zheng, "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems", 2015. Software available from tensorflow.org.

[Abdu17] Waleed Abdulla, "Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow", , 2017.

[Amit14] Yali Amit and Pedro Felzenszwalb, *Object Detection*, pp. 537–542, 01 2014.

[Cele10] M. Emre Celebi, Hitoshi Iyatomi, Gerald Schaefer and William V. Stoecker, "Lesion Border Detection in Dermoscopy Images", *CoRR*, vol. abs/1011.0640, 2010.

[Chen18a] Liang-Chieh Chen, Maxwell D. Collins, Yukun Zhu, George Papandreou, Barret Zoph, Florian Schroff, Hartwig Adam and Jonathon Shlens, "Searching for Efficient Multi-Scale Architectures for Dense Image Prediction", in *NIPS*, 2018.

[Chen18b] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff and Hartwig Adam, "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation", in *ECCV*, 2018.

[Chol15] François Chollet et al., "Keras", , 2015.

[Chol16] François Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions", *CoRR*, vol. abs/1610.02357, 2016.

[Code17] Noel C. F. Codella, David Gutman, M. Emre Celebi, Brian Helba, Michael A. Marchetti, Stephen W. Dusza, Aadi Kalloo, Konstantinos Liopyris, Nabin K. Mishra, Harald Kittler and Allan Halpern, "Skin Lesion Analysis Toward Melanoma Detection: A Challenge at the 2017 International Symposium on Biomedical Imaging (ISBI), Hosted by the International Skin Imaging Collaboration (ISIC)", *CoRR*, vol. abs/1710.05006, 2017.

[Code19] Noel C. F. Codella, Veronica Rotemberg, Philipp Tschandl, M. Emre Celebi, Stephen W. Dusza, David Gutman, Brian Helba, Aadi Kalloo, Konstantinos Liopyris, Michael A. Marchetti, Harald Kittler and Allan Halpern, "Skin Lesion Analysis Toward Melanoma Detection 2018: A Challenge Hosted by the International Skin Imaging Collaboration (ISIC)", *CoRR*, vol. abs/1902.03368, 2019.

[Comb19] Marc Combalia, Noel C. F. Codella, Veronica Rotemberg, Brian Helba, Veronica Vilaplana, Ofer Reiter, Allan C. Halpern, Susana Puig and Josep Malvehy, "BCN20000: Dermoscopic Lesions in the Wild", *CoRR*, vol. abs/1710.05006, 2019.

[Dala05] Navneet Dalal and Bill Triggs, “Histograms of Oriented Gradients for Human Detection”, in *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05) - Volume 1 - Volume 01*, CVPR ’05, pp. 886–893, Washington, DC, USA, 2005, IEEE Computer Society.

[Dumo16] Vincent Dumoulin and Francesco Visin, “A guide to convolution arithmetic for deep learning”, *CoRR*, vol. abs/1603.07285, 2016.

[Ever10] Mark Everingham, Luc Gool, Christopher K. Williams, John Winn and Andrew Zisserman, “The Pascal Visual Object Classes (VOC) Challenge”, *Int. J. Comput. Vision*, vol. 88, no. 2, pp. 303–338, June 2010.

[Girs13] Ross B. Girshick, Jeff Donahue, Trevor Darrell and Jitendra Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation”, *CoRR*, vol. abs/1311.2524, 2013.

[Girs15] Ross B. Girshick, “Fast R-CNN”, *CoRR*, vol. abs/1504.08083, 2015.

[He15] Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun, “Deep Residual Learning for Image Recognition”, *CoRR*, vol. abs/1512.03385, 2015.

[He17] Kaiming He, Georgia Gkioxari, Piotr Dollár and Ross B. Girshick, “Mask R-CNN”, *CoRR*, vol. abs/1703.06870, 2017.

[JEME08] A JEMEL, “Cancer statistics 2008”, *CA Cancer J. Clin.*, vol. 58, pp. 71–96, 2008.

[Kriz12] Alex Krizhevsky, Ilya Sutskever and Geoffrey E Hinton, “ImageNet Classification with Deep Convolutional Neural Networks”, in F. Pereira, C. J. C. Burges, L. Bottou and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pp. 1097–1105, Curran Associates, Inc., 2012.

[LeCu99] Yann LeCun, Patrick Haffner, Léon Bottou and Yoshua Bengio, “Object Recognition with Gradient-Based Learning”, in David A. Forsyth, Joseph L. Mundy, Vito Di Gesù and Roberto Cipolla, editors, *Shape, Contour and Grouping in Computer Vision*, vol. 1681 of *Lecture Notes in Computer Science*, p. 319, Springer, 1999.

[Lin14] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár and C. Lawrence Zitnick, “Microsoft COCO: Common Objects in Context”, *CoRR*, vol. abs/1405.0312, 2014.

[Lin16] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan and Serge J. Belongie, “Feature Pyramid Networks for Object Detection”, *CoRR*, vol. abs/1612.03144, 2016.

[M95] Binder M, Schwarz M and Winkler A et all., “Epiluminescence Microscopy. A Useful Tool for the Diagnosis of Pigmented Skin Lesions for Formally Trained Dermatologists. Archives of Dermatology.”, vol. 131(3):286–291, 1995.

[Ren15] Shaoqing Ren, Kaiming He, Ross B. Girshick and Jian Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”, *CoRR*, vol. abs/1506.01497, 2015.

[Rude16] Sebastian Ruder, “An overview of gradient descent optimization algorithms”, *CoRR*, vol. abs/1609.04747, 2016.

[Tsch18] Philipp Tschandl, Cliff Rosendahl and Harald Kittler, “The HAM10000 Dataset: A Large Collection of Multi-Source Dermatoscopic Images of Common Pigmented Skin Lesions”, *CoRR*, vol. abs/1803.10417, 2018.

[Viol04] Paul Viola and Michael J. Jones, "Robust Real-Time Face Detection", *Int. J. Comput. Vision*, vol. 57, no. 2, pp. 137–154, May 2004.