ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Deep Neural Networks for Image Restoration in MNIST Dataset

Δημήτριος - Σ – Τσεσμελής
Απόστολος - Ν – Φλωράκης

Επιβλέπων: **Σταματόπουλος Παναγιώτης,** Επίκουρος Καθηγητής

ΑΘΗΝΑ

ΣΕΠΤΕΜΒΡΙΟΣ 2018

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Deep Neural Networks for Image Restoration in MNIST Dataset

Δημήτριος Σ. Τσεσμελής
Α.Μ.: 1115201400208

Απόστολος Ν. Φλωράκης
Α.Μ.: 1115201400217

ΕΠΙΒΛΕΠΟΝΤΕΣ: **Σταματόπουλος Παναγιώτης,** Επίκουρος Καθηγητής

ΠΕΡΙΛΗΨΗ

Στην εργασία μας προσπαθούμε να παρουσιάσουμε συνοπτικά και ταυτόχρονα περιεκτικά τα βασικά στοιχεία της δομής ενός νευρωνικού δικτύου πολλών επιπέδων καθώς και ορισμένες θεμελιώδεις τεχνικές και μεθοδολογίες που καθορίζουν τη λειτουργία του. Η προσέγγισή μας σε ορισμένα κεφάλαια που είναι αρκετά δυσνόητα είναι διαισθητική, δηλαδή περιέχει οπτικό και περιγραφικό υλικό, ώστε να αποδοθεί κατά το δυνατό καλύτερα το νόημα των εννοιών που εξετάζουμε. Παράλληλα, δίνεται ιδιαίτερη έμφαση στη μαθηματική τεκμηρίωση σε όσα σημεία κρίνεται απαραίτητο δεδομένου ότι τα νευρωνικά δίκτυα και γενικότερα ο τομέας της Μηχανικής Μάθησης βασίζονται σε αυστηρά ορισμένα μαθηματικά μοντέλα.

Η παρουσίασή μας έχει ως στόχο τη μύηση ενός νέου αναγνώστη στο Deep Learning, ανεξάρτητα με τις γνώσεις του στο αντικείμενο, ώστε μετά την ανάγνωση και κατανόησή της να έχει εφοδιαστεί με τις απαραίτητες γνώσεις, διαμορφώνοντας μία ολοκληρωμένη εικόνα για το αντικείμενο. Παρά το γεγονός ότι στην εργασία μας έχουμε συμπεριλάβει εκτός των στοιχειωδών τεχνικών και αρκετές πιο σύγχρονες που χρησιμοποιούνται ακόμα και σε σημερινές εφαρμογές, θα προτείναμε σε κάποιον αναγνώστη που σκοπεύει να ασχοληθεί περαιτέρω, να μελετήσει τις νέες και πιο εξειδικευμένες μεθόδους που αναπτύσσονται.

Γνωρίζοντας πως να νευρωνικά δίκτυα αποτελούν πλέον μία από τις βασικότερες και αποτελεσματικότερες μεθόδους επεξεργασίας εικόνων αποφασίσαμε να πειραματιστούμε με ορισμένους τύπους νευρωνικών δικτύων για την ανακατασκευή εικόνων που έχουν υποστεί κάποια αλλοίωση.

Για το πρακτικό κομμάτι της εργασίας μας αναπτύξαμε δύο είδη νευρωνικών δικτύων. Συγκεκριμένα, υλοποιήσαμε ένα Convolutional Neural Network (CNN) και ένα Encoder - Decoder για την επίλυση των προβλημάτων Image Inpainting και Image Denoising. Το Dataset που χρησιμοποιήσαμε ονομάζεται MNIST και περιέχει σχετικά μικρές σε μέγεθος εικόνες που αναπαριστούν χειρόγραφα ψηφία στο εύρος 0 έως 9. Το Dataset αυτό είναι το καταλληλότερο καθώς διευκολύνει τους πειραματισμούς με τις παραμέτρους του δικτύου και δεν απαιτεί κάποιο πολύ ακριβό υπολογιστικό σύστημα.

Στόχος της εργασίας μας είναι η επίλυση των παραπάνω προβλημάτων και η εξαγωγή εικόνων υψηλής ποιότητας. Ακόμη, αποσκοπούμε στην εξαγωγή συμπερασμάτων μέσω πειραματισμών με τις παραμέτρους τόσο των δικτύων όσο και των προβλημάτων.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Μηχανική Μάθηση, Επεξεργασία Εικόνας

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: Νευρωνικά δίκτυα, Ανακατασκευή εικόνων, Συμπλήρωση εικόνας, Αφαίρεση θορύβου από εικόνα, MNIST σύνολο δεδομένων

ABSTRACT

In our work we try to present concisely and comprehensively the basic features of the structure of a multiple layer neural network and some fundamental technics and methodologies that determine its function. Concerning the fact that some chapters are quite tricky, our approach is more intuitive in order to be explained and understood better. At the same time, taking into account that neural networks and Machine Learning are based on strictly mathematical models, we focus on mathematic explanation wherever is essential.

Our goal is the initiation of a new reader to Deep Learning, regardless of his elder knowledge of this subject. After reading this thesis he will be able to understand how Deep Learning works and its general applications. Although in our work we have included beyond the basic techniques and several modern ones that are used even in today's applications, we would suggest to someone who is interested in studying further, to look for the new trends and methods that are more specialized and up to date.

Considering that Neural Networks are one of the most basic and effective image processing methods, we experimented with several types of Neural Networks in order to reconstruct corrupted images.

We developed two kinds of Neural Networks. More specifically, we implemented a Convolutional Neural Network (CNN) and an Encoder - Decoder to solve both Image Inpainting and Image Denoising. We used MNIST Dataset which consists of tiny images that illustrate handwritten digits in range 0 to 9. This Dataset is the most suitable as it facilitates our experiments with the networks' parameters and it does not require an expensive machine to be used.

One of our goals is to solve these problems and to produce high quality images. We also aim to draw conclusions by experimenting with both networks' and problems' parameters.

SUBJECT AREA: Machine Learning, Image Processing

KEYWORDS: Deep Learning, Neural networks, Image Restoration, Image inapainting, Image Denoising, MNIST dataset

ΕΥΧΑΡΙΣΤΙΕΣ

Με το παρόν σύγγραμμα θα θέλαμε να ευχαριστήσουμε και εγγράφως όλους όσους στήριξαν έμπρακτα, είτε άμεσα είτε έμμεσα, την προσπάθειά μας για τη διεκπεραίωση της διπλωματικής εργασίας, αλλά και γενικότερα για την ολοκλήρωση και απόκτηση των πτυχίων μας.

Στη συνέχεια, θα θέλαμε να ευχαριστήσουμε τον καθηγητή μας κύριο Παναγιώτη Σταματόπουλο, ο οποίος εξαρχής μας καθοδήγησε στην επιλογή του θέματος της εργασίας μας και μας έδινε καθ' όλη τη διάρκεια της εκπόνησής της χρήσιμες συμβουλές ώστε να επιτύχουμε εν τέλει τους στόχους μας.

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΑΤΑΛΟΓΟΣ ΣΧΗΜΑΤΩΝ

ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

ΠΡΟΛΟΓΟΣ

Η εργασία πραγματοποιήθηκε σε διάστημα 11 μηνών, από τη στιγμή που επιλέξαμε το θέμα της εργασίας μέχρι και τη στιγμή που την παραδώσαμε, εκ των οποίων οι πρώτοι 5 μήνες αφιερώθηκαν για την κατανόηση και καταγραφή της θεωρίας και της δουλείας που έχει ήδη γίνει πάνω στον τομέα που ασχοληθήκαμε. Κατά το υπόλοιπο διάστημα εργαστήκαμε πάνω στην υλοποίηση των δικτύων και στην εξαγωγή των αποτελεσμάτων.

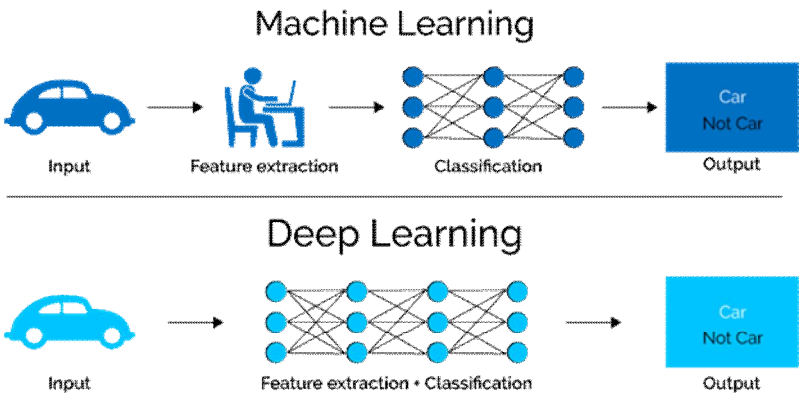
1 ΘΕΩΡΙΑ

1.1 Deep Learning

1.1.1 Γενικός ορισμός του Deep Learning

Με τον όρο Deep Learning αναφερόμαστε στα Νευρωνικά Δίκτυα (Neural Networks) πολλών επιπέδων που στόχο έχουν την προσομοίωση ενός ανθρώπινου εγκεφάλου για την επίλυση ενός συγκεκριμένου προβλήματος.

Τα νευρωνικά δίκτυα είναι γνωστά από το 1990 και αποτελούν μία πτυχή της Τεχνητής Νοημοσύνης. Η βασική διαφορά τους από τα δίκτυα που χρησιμοποιούνται στο Deep Learning είναι ότι αποτελούνται από ένα επίπεδο εισόδου (input), ένα επίπεδο εξόδου (output) και ένα μόνο κρυφό (hidden) επίπεδο. Αντίθετα, στο Deep Learning χρησιμοποιούνται δίκτυα με παραπάνω από ένα κρυφά επίπεδα, γεγονός που τα καθιστά πολύ πιο ισχυρά και αποτελεσματικά. Πρακτικά η διαφορά είναι ότι στα απλά νευρωνικά δίκτυα ο προγραμματιστής πρέπει να εξάγει κάποια χαρακτηριστικά από τα δεδομένα ώστε να σχηματίσει ένα μοντέλο που περιγράφει και κατηγοριοποιεί τα δεδομένα ενώ στο Deep Learning η παραπάνω διαδικασία γίνεται αυτόματα από το δίκτυο.



Εικόνα 1: Machine Learning και Deep Learning

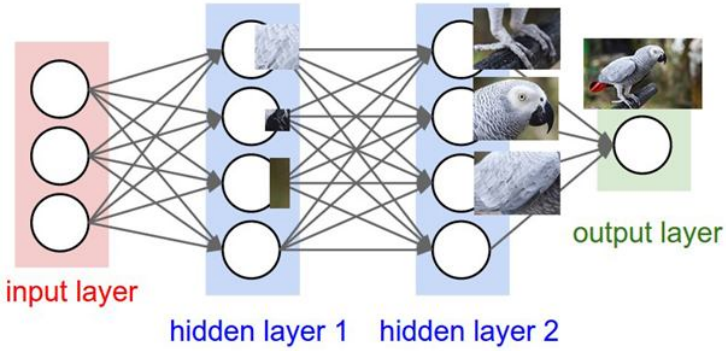
Γιατί όμως το Deep Learning άρχισε να χρησιμοποιείται τα τελευταία χρόνια και όχι παλαιότερα;

Δύο είναι οι βασικοί λόγοι που καθιστούσαν ανέφικτη τη χρήση τέτοιων δικτύων στο παρελθόν. Ο πρώτος λόγος συνίσταται στο γεγονός ότι η υπολογιστική πολυπλοκότητα που χρειάζεται για την εκπαίδευση δικτύων με χιλιάδες νευρώνες είναι πολύ μεγάλη και θα ήταν εξαιρετικά δύσκολο έως αδύνατο για έναν υπολογιστή ή συστοιχία (cluster) υπολογιστών της δεκαετίας του 90 να τη φέρει εις πέρας. Στις μέρες μας η εκπαίδευση τέτοιων δικτύων γίνεται πολύ γρηγορότερα με κάρτες γραφικών (GPUs) που έχουν τη δυνατότητα να επεξεργάζονται τα δεδομένα παράλληλα και σε πολύ υψηλές ταχύτητες. Ο δεύτερος και βασικότερος λόγος ήταν η έλλειψη δεδομένων για την εκπαίδευση των δικτύων. Είναι γεγονός πως η επιτυχία που γνώρισαν τα μεγάλα νευρωνικά δίκτυα οφείλεται σε ένα σημαντικό βαθμό στην πληθώρα δεδομένων που υπάρχουν σήμερα διαθέσιμα στο διαδίκτυο. Μάλιστα προσβάσιμες είναι πολλές βάσεις δεδομένων που προσφέρουν κατηγοριοποιημένα δεδομένα που μπορούν να χρησιμοποιηθούν ανάλογα με τις ανάγκες της εφαρμογής.

Το Deep Learning χρησιμοποιείται ευρέως σε πεδία όπως επεξεργασία και αναγνώριση εικόνων και βίντεο, αναγνώριση ομιλίας, επεξεργασία φυσικής γλώσσας και άλλα.

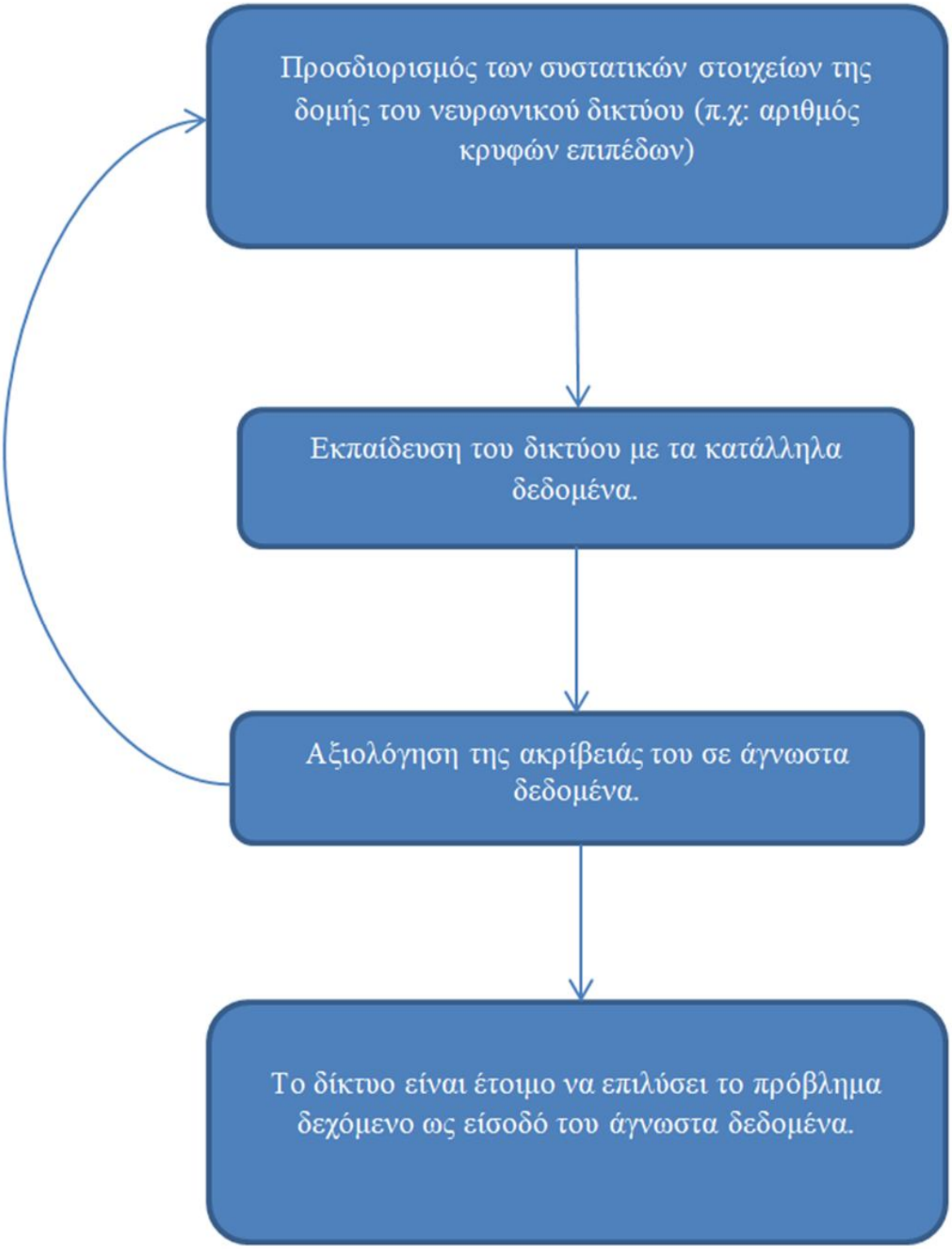
Τα δίκτυα αυτά αποτελούνται από πολλά επίπεδα, όπου το κάθε ένα συνδέεται με το επόμενο και αναπαριστά κάποια αυθαίρετη και αρχέγονη πληροφορία. Όσο μεταβαίνουμε προς τους δεξιότερους νευρώνες ενός δικτύου παίρνουμε μία πιο ολοκληρωμένη πληροφορία για τα δεδομένα μας. Η ιδιότητά τους αυτή μας δίνει μία εικόνα του τρόπου με τον οποίο λειτουργούν τα δίκτυα αυτά.

Για παράδειγμα, αν θεωρήσουμε ένα νευρωνικό δίκτυο δύο (2) επιπέδων που χρησιμοποιείται για την ανίχνευση παπαγάλων σε μία εικόνα, τότε μπορούμε να σκεφτούμε πως το 1ο επίπεδο του δικτύου αναπαριστά κάποια απλά σχήματα της εικόνας, το 2ο επίπεδο συνθέτει τα σχήματα αυτά και φτιάχνει τα μέρη του σώματος ενός παπαγάλου, όπως τα πόδια, τα φτερά, το κεφάλι (...) και τέλος το επίπεδο εξόδου συνθέτοντας τα μέρη του σώματος φτιάχνει τελικά την εικόνα με ολόκληρο τον παπαγάλο.



Εικόνα 2: Διαισθητική προσέγγιση

Η ανάπτυξη ενός νευρωνικού δικτύου για την επίλυση ενός προβλήματος συνοψίζεται στα εξής βήματα:

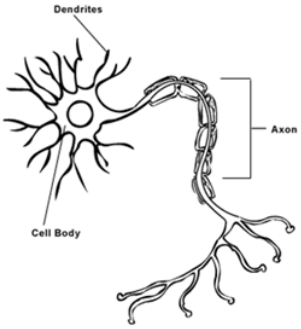


Σχήμα 1: Βήματα Ανάπτυξης Νευρωνικού Δικτύου

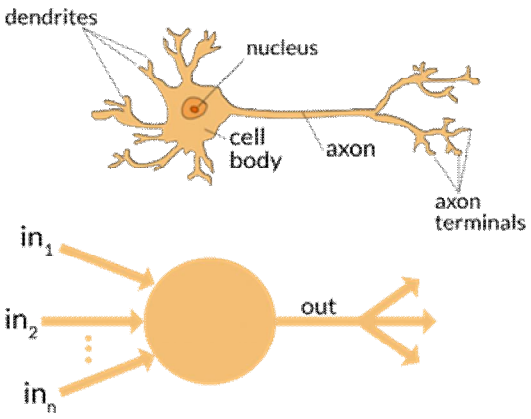
1.1.2 Δομή Νευρωνικού Δικτύου

Ανάλυση πραγματικού νευρώνα:

Η κεντρική ιδέα των νευρωνικών δικτύων βασίζεται στον τρόπο λειτουργίας του ανθρώπινου εγκεφάλου και συγκεκριμένα των νευρώνων που υπάρχουν σε αυτόν. Ο εγκέφαλος είναι ένα μεγάλο δίκτυο διασυνδεδεμένων νευρώνων, οι οποίοι αποτελούν το μέσο διακίνησης της πληροφορίας από ένα σημείο σε ένα άλλο. Όταν λοιπόν για δυο δεδομένα σημεία, δυο διαφορετικές "πληροφορίες" στέλνονται από το ένα στο άλλο, ο εγκέφαλος ενεργοποιεί διαφορετικά "σετ" νευρώνων και ως εκ τούτου οι διαδρομές που ακολουθούνται θα είναι και διαφορετικές.



Εικόνα 3: Νευρώνας



Εικόνα 4: Αντιστοιχία Πραγματικού με Τεχνητού Νευρώνα

Όπως βλέπουμε οι δενδρίτες είναι υπεύθυνοι για τη λήψη εισερχόμενων σημάτων από άλλους νευρώνες. Όταν ο νευρώνας λάβει ένα επαρκές σε ισχύ συνολικό σήμα (υπολογίζεται ως άθροισμα όλων των επιμέρους εισερχομένων σημάτων) στέλνει ένα ηλεκτρικό παλμό στις ηλεκτρικές του απολήξεις. Τα σήματα αυτά εξόδου αποτελούν είσοδο για άλλους νευρώνες. Η βασική ιδέα πάνω στο παραπάνω μοντέλο είναι πως κάθε νευρώνας επεξεργάζεται δεδομένη πληροφορία (input), παράγοντας ένα αποτέλεσμα (output) το οποίο θα αποτελέσει με τη σειρά του είσοδο για κάποιο επόμενο νευρώνα. Υπολογιστικές λοιπόν προσομοιώσεις που προσπαθούν να αναπαραστήσουν ένα τέτοιο δίκτυο διασυνδεδεμένων νευρώνων, προσομοιώνοντας τη λειτουργία του ονομάζονται Artificial Neural Networks (ANN's).

Δομή του νευρωνικού δικτύου:

Ένα τυπικό νευρωνικό δίκτυο έχει από μερικές δεκάδες μέχρι χιλιάδες τέτοιους τεχνητούς νευρώνες (units) τοποθετημένους σε επίπεδα (layers). Κάθε ένας συνδέεται με νευρώνες που ανήκουν σε επόμενο επίπεδο. Ένα νευρωνικό δίκτυο ονομάζεται πλήρως διασυνδεδεμένο όταν κάθε επίπεδο συνδέεται με όλα τα units προηγούμενου και επόμενου επιπέδου με εξαίρεση το πρώτο και το τελευταίο επίπεδο που δεν έχουν προηγούμενο και επόμενο επίπεδο αντίστοιχα. Κάποιοι νευρώνες είναι γνωστοί και ως input units και έχουν ως στόχο λαμβάνοντας πληροφορία από τον έξω κόσμο να πραγματοποιούν λειτουργίες αναγνώρισης και επεξεργασίας αυτών. Από την άλλη πλευρά δηλαδή στο τέλος σχηματικά του δικτύου, υπάρχουν τα output units που υποδηλώνουν πως ανταποκρίθηκε το δίκτυο στην πληροφορία που δέχθηκε ως είσοδο και τι αποτέλεσμα παρήγαγε. Μεταξύ input και output units υπάρχουν τα hidden units τα

οποία αποτελούν την καρδιά του νευρωνικού δικτύου. Ορατά σε εμάς είναι μόνο τα δυο αυτά επίπεδα input-output ενώ όλα τα ενδιάμεσα αποκρύπτονται.

Οι συνδέσεις μεταξύ των units αναπαριστώνται με έναν αριθμό το w (weight) ή αλλιώς βάρος, το οποίο είναι θετικό όταν ένας νευρώνας διεγείρει τον επόμενο με τον οποίο είναι συνδεδεμένος ή αρνητικό στην περίπτωση που τον καταστέλλει. Επίσης όσο μεγαλύτερο είναι το νούμερο αυτό τόσο μεγαλύτερη επιρροή θα έχει το unit αυτό στο επόμενο unit (τεχνική που προσομοιώνει πραγματική συμπεριφορά του ανθρώπινου εγκεφάλου). Στην πράξη λοιπόν όταν ένα input εισέρχεται σε έναν νευρώνα, πολλαπλασιάζεται με το αντίστοιχο βάρος. Στην γενική περίπτωση πολλών input κάθε ένα πολλαπλασιάζεται με το αντίστοιχο βάρος και όλα τα επιμέρους γινόμενα αθροίζονται μεταξύ τους. Τα weights αρχικοποιούνται τυχαία και αναπροσδιορίζονται αποκτώντας διαφορετικές τιμές κατά τη διαδικασία της εκμάθησης (training). Η εξίσωση που περιγράφει το παραπάνω άθροισμα είναι γραμμική και είναι της μορφής.

$$W \cdot X = w_1 x_1 + w_2 x_2 + \ldots + w_m x_m = \sum_{i=1}^{m} w_i x_i$$

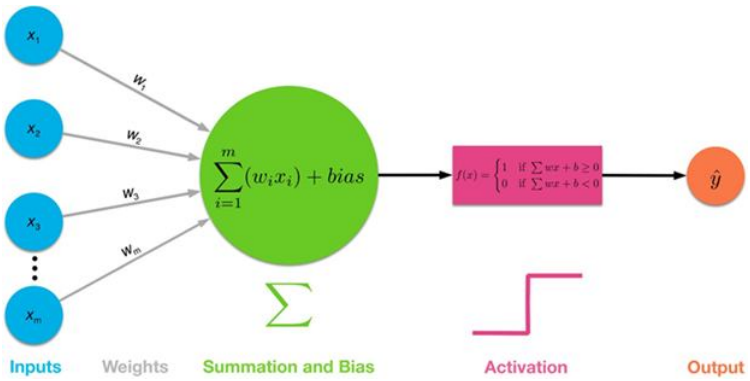
όπου με wi συμβολίζονται τα βάρη weights και με xi τα αντίστοιχαα input.

Λαμβάνεται επίσης υπόψιν μια ακόμη παράμετρος η οποία ονομάζεται bias και είναι χαρακτηριστική για κάθε νευρώνα, αναπαριστώντας το ελάχιστο άθροισμα γινομένων $w_i x_i$ που απαιτείται για την ενεργοποίηση του νευρώνα. Η προσθήκη του bias έχει ως συνέπεια την αλλαγή της τιμής και του εύρους του τελικού αποτελέσματος.

Μετά την κατασκευή της παραπάνω γραμμικής συνάρτησης εφαρμόζεται σε αυτή μια μη γραμμική συνάρτηση η λεγόμενη συνάρτηση ενεργοποίησης (activation function) η οποία μετατρέπει τα input σε output signals. Ενδεικτικά θεωρώντας ως συνάρτηση ενεργοποίησης τη σιμγοειδή συνάρτηση, η εξίσωση είναι της μορφής

$$z = \sum_{i=1}^{m} w_i x_i + bias$$
$$\text{Sigmoid Function is: } \sigma(z) = \frac{1}{1+e^{-z}}$$

Σε επόμενο κεφάλαιο της εργασίας υπάρχει αναλυτικότερη περιγραφή των actiavtion functions και των χαρακτηριστικών τους. Συμπεραίνουμε, λοιπόν, πως η ενεργοποίηση ενός νευρώνα εξαρτάται από τα inputs, τα weights, τα biases καθώς και τις activation functions. Μπορούμε να φανταστούμε σχηματικά ένα νευρώνα όπως φαίνεται στην εικόνα 5.



Εικόνα 5: Πράξη Υπολογισμού Ενεργοποίησης Νευρώνα

1.1.3 Συναρτήσεις ενεργοποίησης

Ένα από τα κρισιμότερα βήματα της κατασκευής ενός νευρωνικού δικτύου είναι η επιλογή της συνάρτησης ενεργοποίησης. Η συνάρτηση αυτή είναι το χαρακτηριστικό που προσδίδει στο δίκτυο τη δυνατότητα να υπολογίζει και να αναπαριστά αυθαίρετα πολύπλοκες συναρτήσεις. Οι συναρτήσεις ενεργοποίησης που χρησιμοποιούνται συνήθως είναι μη γραμμικές συναρτήσεις, καθώς οι γραμμικές είναι ισοδύναμες με ένα νευρωνικό δίκτυο με ένα κρυφό επίπεδο και συνεπώς η δυναμικότητά τους είναι περιορισμένη. Σε επόμενο κεφάλαιο θα περιγράψουμε το πρόβλημα XOR το οποίο δεν είναι γραμμικώς διαχωρίσιμο. Η επιτυχής ταξινόμηση ενός συνόλου δεδομένων σε ένα τέτοιο πρόβλημα καθίσταται αδύνατη με τη χρήση γραμμικής συνάρτησης ενεργοποίησης, αφού όπως περιγράψαμε χρειαζόμαστε τουλάχιστον δύο κρυφά επίπεδα. Συνεπώς οι συναρτήσεις ενεργοποίησης που θα μας απασχολήσουν είναι μη γραμμικές.

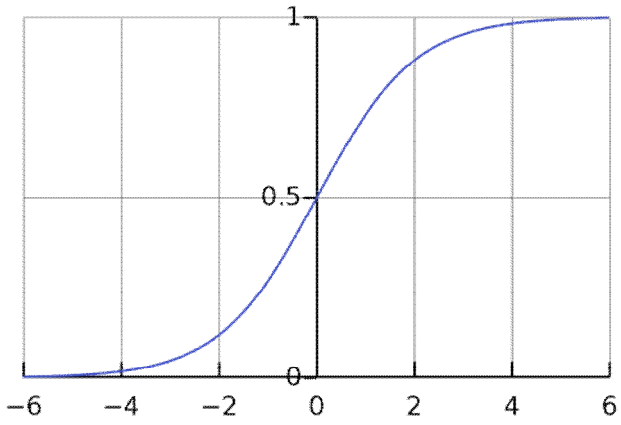
Επιπρόσθετα, η χρήση συνάρτησης ενεργοποίησης περιορίζει την τιμή εξόδου του κάθε νευρώνα σε ένα διάστημα. Η τιμή που προκύπτει στους νευρώνες του δικτύου ενδέχεται να είναι αρκετά μεγάλη αφού πρόκειται για άθροισμα των εξόδων όλων των προηγούμενων νευρώνων πολλαπλασιασμένων με ένα βάρος. Μάλιστα όσο μεταβαίνουμε προς τα δεξιά στο δίκτυο οι τιμές θα μεγαλώνουν και η διαδικασία επεξεργασίας του θα γίνεται όλο και πιο δύσκολη. Για το λόγο αυτό εφαρμόζεται στο άθροισμα μία συνάρτηση ενεργοποίησης που μετατρέπει την τιμή εξόδου του κάθε νευρώνα σε μία νέα τιμή στο διάστημα (α, β).

Κάποιες από τις πιο γνωστές συναρτήσεις ενεργοποίησης είναι οι παρακάτω:

1) <u>Sigmoid:</u>

   Η συνάρτηση αυτή λαμβάνει ως είσοδο έναν πραγματικό αριθμό και δίνει σαν έξοδο έναν αριθμό στο διάστημα 0 έως 1. Μετατρέπει μεγάλους αρνητικούς αριθμούς σε 0 και μεγάλους θετικούς σε 1.

   Τύπος: $$\sigma(x) = \frac{1}{1 + e^{-x}}$$



Σχήμα 2: Sigmoid Activation Function

Η απλή αυτή συνάρτηση έχει ορισμένα μειονεκτήματα. Το πρώτο και σημαντικότερο πρόβλημα (vanishing gradients) εμφανίζεται στα δίκτυα που εκπαιδεύονται με τη μέθοδο backpropagation. Το πρόβλημα έγκειται στο γεγονός ότι η σιγμοειδής συνάρτηση αντιστοιχίζει τόσο την τιμή 0 όσο και την τιμή 1 στην τιμή 0, κάτι το οποίο έχει σαν αποτέλεσμα να μην γίνεται τροποποίηση στα βάρη κάποιων νευρώνων που θα έπρεπε να γίνει. Ακόμη, η έξοδος της συνάρτησης δεν είναι "κεντραρισμένη" στο μηδέν (not zero centered) και ο υπολογισμός της εκθετικής συνάρτησης expr() είναι υπολογιστικά πιο ακριβός συγκριτικά με άλλες συναρτήσεις ενεργοποίησης. Η σιγμοειδής συνάρτηση χρησιμοποιείται σπάνια σήμερα, κυρίως σε απλά προβλήματα ταξινόμησης δύο κλάσεων και μόνο στο τελευταίο επίπεδο (output layer).

2) Softmax:

Η συνάρτηση αυτή, όπως και η σιγμοειδής, αντιστοιχίζει την είσοδο με μία τιμή στο διάστημα 0 έως 1. Η διαφορά των δύο συναρτήσεων είναι πως η Softmax παράγει ένα διάνυσμα (z) μεγέθους όσες και οι μονάδες του τελευταίου επιπέδου του δικτύου (j), η κάθε θέση του οποίου περιέχει την πιθανότητα η είσοδος που δόθηκε στο δίκτυο να ανήκει στην αντίστοιχη κλάση.

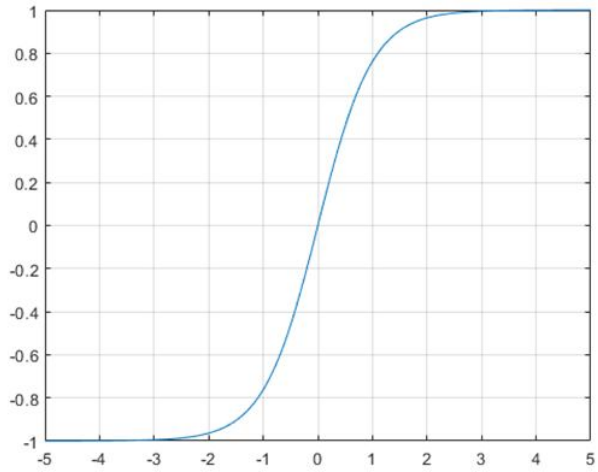
Τύπος: $$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

Η παραπάνω ιδιότητα καθιστά τη συνάρτηση αυτή ιδιαίτερα χρήσιμη για προβλήματα ταξινόμησης όπου οι κλάσεις είναι περισσότερες από δύο και χρησιμοποιείται συνήθως ως συνάρτηση ενεργοποίησης για το επίπεδο εξόδου του δικτύου.

3) Hyperbolic Tangent function (Tanh):

Η συνάρτηση αυτή είναι αρκετά παρόμοια με τη σιγμοειδή συνάρτηση, με τη διαφορά ότι λαμβάνει ως είσοδο έναν πραγματικό αριθμό και δίνει σαν έξοδο έναν αριθμό στο διάστημα -1 έως 1. Θεωρείται καλύτερη συνάρτηση από τη σιγμοειδή αφού ξεπερνά το πρόβλημα not zero centered αλλά όχι το vanishing gradients.

Τύπος: $$\tanh x = \frac{\sinh x}{\cosh x} = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$



Σχήμα 3: Tanh Activation Function

4) Rectified Linear Unit (ReLU):

Η συνάρτηση αυτή αντιστοιχίζει την τιμή εισόδου με μία τιμή στο διάστημα 0 έως x, όπου x είναι ένας θετικός αριθμός. Για αρνητικές τιμές εισόδου η έξοδος είναι 0 ενώ για θετικές εισόδους η έξοδος είναι x. Η ReLU χρησιμοποιείται ευρέως σήμερα καθώς ξεπερνά εν μέρη το πρόβλημα vanishing gradients (τουλάχιστον για τις θετικές τιμές εισόδου) και είναι υπολογιστικά πολύ αποδοτική αφού υλοποιείται αρκετά εύκολα.

Τύπος: $f(x) = max(0,x)$

Σχήμα 4: ReLU Activation Function

Τα δύο βασικά μειονεκτήματα της συνάρτησης αυτής είναι το γεγονός ότι είναι not zero centered καθώς επίσης το ότι για αρνητικές τιμές εισόδου δεν αντιμετωπίζει το πρόβλημα vanishing gradients. Παρόλα αυτά στην πράξη σήμερα είναι από τις συναρτήσεις που χρησιμοποιούνται συχνότερα σαν συνάρτηση ενεργοποίησης των κρυφών μονάδων.

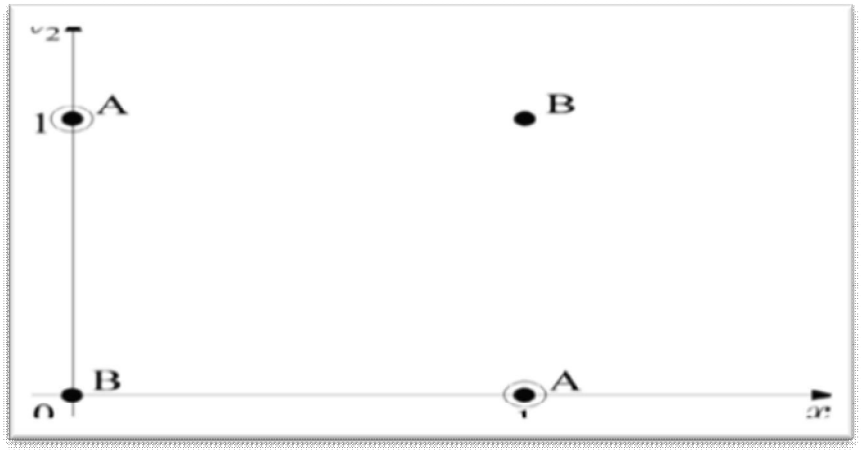
1.1.4 Επιλογή αριθμού Κρυφών Επιπέδων:

Όπως έχουμε προαναφέρει το Deep Learning ασχολείται με νευρωνικά δίκτυα πολλών κρυφών επιπέδων.

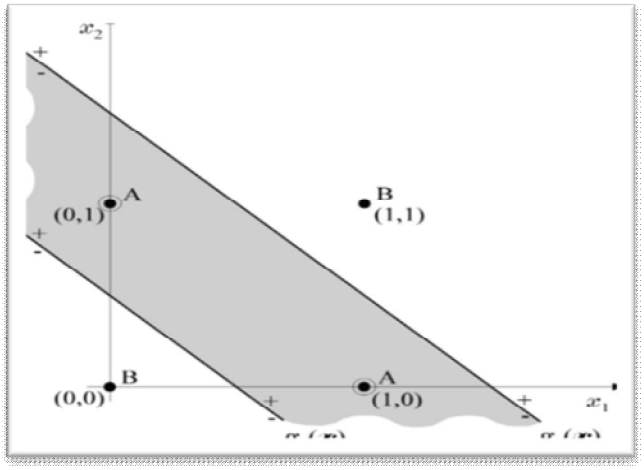
Για ποιό λόγο όμως χρειαζόμαστε τέτοια νευρωνικά δίκτυα και δεν μας αρκούν τα πιο κλασικά νευρωνικά δίκτυα με μόνο ένα κρυφό επίπεδο, τα οποία μάλιστα βάση του θεωρήματος της καθολικής προσέγγισης (universal approximation theorem) μπορούν να προσεγγίσουν οποιαδήποτε συνεχή συνάρτηση σε συμπαγή υποσύνολα του R_n, υπό κάποιες χαλαρές υποθέσεις σχετικά με τη συνάρτηση ενεργοποίησης;

Η απάντηση έγκειται στη δυσκολία που απαιτείται από τον προγραμματιστή προκειμένου να σχηματίσει ένα νευρωνικό δίκτυο με ένα κρυφό επίπεδο που να επιλύει δύσκολα προβλήματα. Αντίθετα, με τη χρήση πολλών κρυφών επιπέδων το δίκτυο μπορεί να εκπαιδευτεί και να “μάθει” από μόνο του τις “ιδιαιτερότητες” της συνάρτησης, όσο πολύπλοκες και αν είναι.

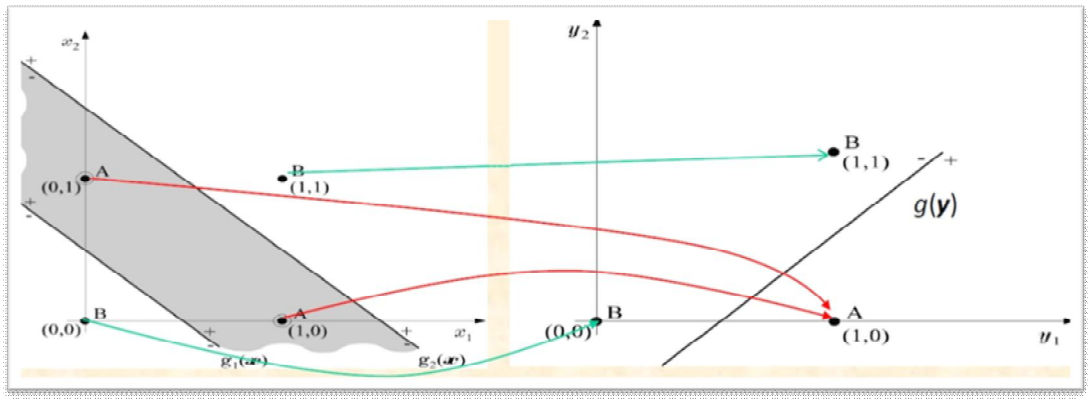
Θα προσπαθήσουμε να εξηγήσουμε τι ακριβώς συμβαίνει σε ένα νευρωνικό δίκτυο όσο προσθέτουμε κρυφά επίπεδα. Ας υποθέσουμε ότι έχουμε ένα πρόβλημα ταξινόμησης, το πρόβλημα XOR. Όπως φαίνεται και στην εικόνα 6, υπάρχουν δύο κλάσεις (A, B) που θα θέλαμε να τις διαχωρίσουμε με μία ευθεία, κάτι το οποίο είναι αδύνατο στο χώρο αυτό αφού καμία ευθεία δεν διαχωρίζει την μία κλάση από την άλλη. Για να επιτευχθεί ο διαχωρισμός χρειάζονται τουλάχιστον δύο ευθείες (εικόνα 7), δηλαδή χρειάζεται προσθήκη ενός κρυφού επιπέδου. Έτσι στο νέο μετασχηματισμένο χώρο οι δύο κλάσεις είναι γραμμικώς διαχωρίσιμες αφού υπάρχει μία ευθεία που τις διαχωρίζει (εικόνα 8).



Εικόνα 6: Πρόβλημα XOR



Εικόνα 7: Το πρόβλημα XOR διαχωρισμένο μη γραμμικά



Εικόνα 8: Το πρόβλημα XOR διαχωρισμένο γραμμικά στο νέο μετασχηματισμένο χώρο

Στο πρόβλημα XOR αρχικοποιήσαμε το δίκτυο με μηδέν κρυφά επίπεδα και στη συνέχεια προσθέσαμε ένα για να γίνει το πρόβλημα γραμμικώς διαχωρίσιμο στο μετασχηματισμένο χώρο. Για να καθορίσουμε τον αριθμό των κρυφών επιπέδων σε ένα πρόβλημα που καλούμαστε να λύσουμε θα πρέπει να μελετήσουμε τα χαρακτηριστικά που έχουμε διαθέσιμα από τα δεδομένα με τα οποία θα εκπαιδεύσουμε το δίκτυο και με πειραματισμούς να επιλέξουμε τελικά τον αριθμό που προσδίδει στο δίκτυο τη βέλτιστη συμπεριφορά. Με άλλα λόγια δεν υπάρχει κάποιος αυστηρός κανόνας που να προσδιορίζει τον αριθμό των κρυφών επιπέδων καθώς εξαρτάται αποκλειστικά από τα δεδομένα εκπαίδευσης και από την πολυπλοκότητά τους. Παρόλα αυτά μπορούμε να ακολουθήσουμε κάποιες απλές τακτικές για τον προσδιορισμό του μεγέθους του δικτύου όπως είναι οι παρακάτω:

i. Υιοθετούμε μια σταθερή δομή δικτύου, το εκπαιδεύουμε με το σύνολο δεδομένων μας και στη συνέχεια υπολογίζουμε την απόδοσή του. Αν δεν είναι ικανοποιητική τότε επαναλαμβάνουμε τη διαδικασία με μια τροποποιημένη αρχιτεκτονική έως ότου προκύψει ένα δίκτυο ικανοποιητικής απόδοσης.

ii. Υιοθετούμε μια μεταβλητή δομή δικτύου, δηλαδή ένα δίκτυο που μεταβάλλεται κατά την διάρκεια την εκπαίδευσής του μέχρι να “μάθει” όλες τις λεπτομέρειες του συνόλου εκπαίδευσης.

1.1.5 Επιλογή αριθμού νευρώνων σε κάθε επίπεδο

Καθοριστικής σημασίας επιλογή για την αρχιτεκτονική του δικτύου είναι αυτή του αριθμού των νευρώνων σε κάθε κρυφό επίπεδο. Κατά τη διαδικασία αυτή συχνά προκύπτουν προβλήματα που οφείλονται είτε στην επιλογή μικρού είτε στην επιλογή μεγάλου αριθμού νευρώνων.

Στην πρώτη περίπτωση, έχουμε το λεγόμενο underfitting, όπου εάν τα δεδομένα είναι περίπλοκα καθίσταται ανέφικτη η επαρκής ανίχνευση των σημάτων μεταξύ των νευρώνων του ενός επιπέδου με το επόμενο.

Στην περίπτωση όπου έχουμε μεγάλο αριθμό νευρώνων ενδέχεται να καταλήξουμε σε overfitting, δηλαδή σε ένα δίκτυο το οποίο δεν έχει εκπαιδευτεί πλήρως με τα διαθέσιμα δεδομένα καθώς οι νευρώνες σε κάθε επίπεδο είναι πολλοί ενώ τα δεδομένα εκπαίδευσης πολύ περιορισμένα. Επίσης, σε ένα δίκτυο με πολλούς νευρώνες η διαδικασία της εκπαίδευσης μπορεί να καθυστερεί πολύ και πολλές φορές σε βαθμό μη αποδεκτό.

Είναι, λοιπόν, επιτακτική ανάγκη να τηρούνται οι ισορροπίες κατά την επιλογή του πλήθους των νευρώνων ώστε να αποφευχθεί η αποτυχία του δικτύου.

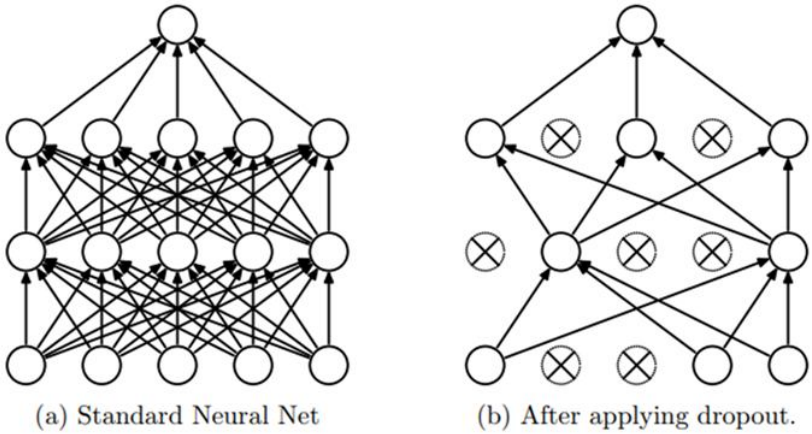
Το overfitting ή αλλιώς overtraining είναι από τα κρισιμότερα και πιο συχνά εμφανιζόμενα προβλήματα στα νευρωνικά δίκτυα. Πιο συγκεκριμένα, το πρόβλημα που προκύπτει είναι πως ενώ το σφάλμα (error) του δικτύου στο σύνολο εκπαίδευσης (training set) είναι πολύ μικρό, στο σύνολο δοκιμής (test set) είναι αρκετά μεγαλύτερο. Αυτό συμβαίνει, διότι το δίκτυο έχει “μάθει” τις ιδιομορφίες του συνόλου εκπαίδευσης αλλά δεν έχει γενικευτεί ώστε να μπορεί να έχει το ίδιο καλά αποτελέσματα και στο σύνολο δοκιμής.

Η τροποποίηση της αρχιτεκτονικής του δικτύου τις περισσότερες φορές αποτυγχάνει να επιλύσει το πρόβλημα αυτό. Για το λόγο αυτό έχουν αναπτυχθεί αρκετές τεχνικές που αποσκοπούν στη γενίκευση των νευρωνικών δικτύων, δηλαδή στη μείωση του σφάλματος σε άγνωστα δεδομένα. Μία από τις πιο αποδοτικές τεχνικές αποτροπής του overfitting είναι το Dropout.

Η βασική ιδέα πίσω από τη μέθοδο αυτή είναι η τυχαία αφαίρεση κάποιων κρυφών νευρώνων κατά τη διαδικασία εκπαίδευσης. Με τον όρο αφαίρεση εννοούμε την προσωρινή αφαίρεση ενός νευρώνα από το δίκτυο καθώς και των εισερχόμενων και εξερχόμενων συνδέσεων σε αυτόν (εικόνα 9).

Πρακτικά αυτό που συμβαίνει κατά το overfitting είναι η δημιουργία εξαρτήσεων μεταξύ κάποιων νευρώνων κατά την εκπαίδευση του δικτύου. Με το Dropout εξαλείφονται οι εξαρτήσεις αυτές αφού για κάθε ένα από τα δεδομένα εκπαιδεύεται διαφορετικό υποσύνολο των νευρώνων του δικτύου. Αν αναλογιστούμε πως για ένα δίκτυο με N νευρώνες υπάρχουν 2N αραιωμένα δίκτυα τότε είναι αρκετά σπάνιο σε ένα δίκτυο με εκατοντάδες νευρώνες να εκπαιδευτεί πάνω από μία φορά το ίδιο αραιωμένο δίκτυο.

Συγκριτικά με άλλες μεθόδους έχει παρατηρηθεί πως το Dropout για διάφορα προβλήματα ταξινόμισης περιορίζει κατά πολύ το σφάλμα του δικτύου σε άγνωστα δεδομένα.



Εικόνα 9: Dropout

1.1.6 Προς τα εμπρός διάδοση (Forward Propagation)

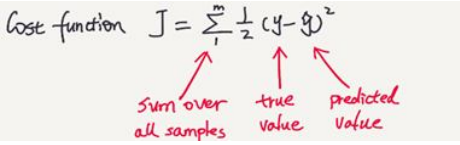
Με τον όρο forward propagation ορίζουμε τη διαδικασία επεξεργασίας μιας εισόδου η οποία εισερχόμενη αρχικά από το input layer και μέσω των hidden layers, καταλήγει στο output layer, ακολουθώντας μόνο την πορεία αυτή. Όπως προηγουμένως έχουμε αναλύσει για κάθε νευρώνα δεδομένων των εισόδων του, των αντίστοιχων weights και λαμβάνοντας υπόψιν το bias του μπορούμε να υπολογίσουμε αν ο νευρώνας αυτός θα ενεργοποιηθεί. Στη συνέχεια και με τη χρήση μιας συνάρτησης ενεργοποίησης μπορούμε να παράξουμε μια έξοδο η οποία θα αποτελέσει είσοδο για νευρώνα επόμενου επιπέδου. Η διαδικασία αυτή και ξεκινώντας από το πρώτο επίπεδο με input την αρχική μας πληροφορία, επαναλαμβάνεται για κάθε νευρώνα μέχρι και το τελευταίο επίπεδο όπου και θα παραχθεί το αποτέλεσμα μας. Η ροή διάδοσης της “πληροφορίας“ είναι προς μια κατεύθυνση, συγκεκριμένα forward (από το input προς το output layer) όποτε και ονομάζεται προς τα εμπρός διάδοση (forward propagation).

1.1.7 Συνάρτηση κόστους (Cost-Loss Function)

Κατά τη διαδικασία της εκπαίδευσης, οι τιμές των παραμέτρων ενός νευρωνικού δικτύου θα αρχικοποιηθούν με κάποια τυχαιότητα. Όπως είναι λογικό τα τελικά αποτελέσματα τα οποία θα προκύψουν σε πρώτη φάση, θα απέχουν κατά πολύ από τις πραγματικές τιμές που επιθυμούμε. Πρέπει να ορίσουμε, λοιπόν, κάποιο κριτήριο ώστε να ρυθμίσουμε την διαδικασία αυτή, επιλέγοντας τις κατά το δυνατόν καλύτερες τιμές παραμέτρων.

Κατά την κατασκευή του νευρωνικού δικτύου προσπαθούμε να προβλέψουμε κάθε έξοδο όσο πιο κοντά στην αντίστοιχη πραγματική τιμή. Την ακρίβεια της δυνατότητας αυτής του δικτύου την μετράμε χρησιμοποιώντας τη συνάρτηση κόστους (cost/loss function). Με τη χρήση αυτής προσπαθούμε να προσομοιώσουμε την επιβολή κάποιας “διόρθωσης” στο δίκτυο, στην περίπτωση που γίνονται λάθη. Ως στόχο μας έχουμε να αυξήσουμε την ακρίβεια πρόβλεψης και να μειώσουμε το σφάλμα ελαχιστοποιώντας τη συνάρτηση κόστους. Κάτι τέτοιο ουσιαστικά επιτυγχάνεται αναπροσαρμόζοντας τις παραμέτρους του δικτύου μας όπως για παράδειγμα τα βάρη (weights) αλλά και το bias κάθε νευρώνα, γεγονός που οδηγεί στην πραγματοποίηση προβλέψεων κατά το δυνατό

πιο κοντινών στα δεδομένα μας. Μια συνήθης πρακτική είναι να ορίσουμε τη συνάρτηση ως το μέσο τετραγωνικό σφάλμα (squared-error):



όπου m είναι το πλήθος των στοιχείων που χρησιμοποιούμε σαν είσοδο κατά τη διαδικασία εκπαίδευσης, y' είναι η τιμή που προβλέπουμε και y η πραγματική τιμή. Αντιλαμβανόμαστε, λοιπόν, πως δεδομένου ότι η παράμετρος y' αφορά τις προβλεπόμενες τιμές, στην συνάρτηση κόστους τελικά εμπλέκονται όλες οι παράμετροι (weights, biases) που εμπλέκονται στον υπολογισμό τους, και η κατάλληλη επιλογή των οποίων θα οδηγήσει εν τέλει στην ελάχιστη τιμή της συνάρτησης.

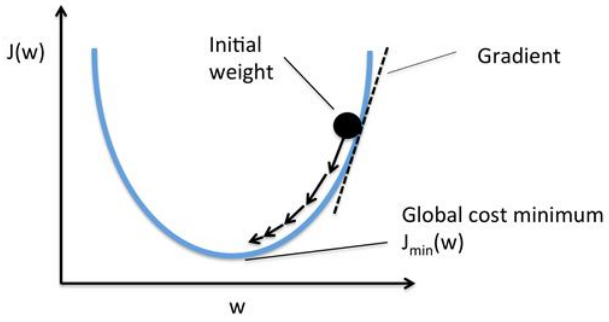
Ας εξετάσουμε το εξής παράδειγμα. Έστω πως έχουμε στη διάθεση μας μερικές χιλιάδες δείγματα καθώς και την κλάση στην οποία ανήκει το καθένα. Ξεκινάμε εισάγοντας το πρώτο στο δίκτυο μας και το διαδίδουμε προς τα εμπρός. Ιδανικά στα output units θα έπρεπε να υπάρχει τιμή 1 στο αντίστοιχο unit και 0 στα υπόλοιπα, οπότε και το συνολικό κόστος να προκύπτει ίσο με το μηδέν. Έχοντας παραμετροποιήσει με τυχαίο τρόπο το δίκτυο μας οι τιμές των output units θα έχουν τυχαιότητα και θα είναι αρκετά μακριά από τις αναμενόμενες, με αποτέλεσμα το κόστος σύμφωνα με την εξίσωση του squared-error να είναι μεγάλο. Η παραπάνω διαδικασία υπολογίζει το κόστος για έναν κύκλο εκμάθησης. Στη συνέχεια επαναλαμβάνουμε με τα υπόλοιπα ψηφία και υπολογίζουμε το μέσο κόστος, παραμετροποιώντας κατάλληλα το δίκτυο μας ώστε να το ελαχιστοποιούμε με τεχνικές που θα αναλυθούν στη συνέχεια.

Συμπεραίνουμε λοιπόν πως όσο μικρότερη είναι η τιμή της τόσο καλύτερα αναγνωρίζει το δίκτυο τα δεδομένα που δέχεται ως είσοδο ενώ σε περίπτωση μεγάλης τιμής υπάρχουν αρκετές αναντιστοιχίες.

Εκτός της προαναφερόμενης Mean Squared Error function υπάρχουν και άλλες συναρτήσεις που χρησιμοποιούνται ευρέως όπως οι: Mean Squared Logarithmic Error, Mean Absolute Error, Mean Absolute Percentage Error, Negative Logarithmic Likelihood, Cross Entropy και άλλες. Ως βασικά κριτήρια για την επιλογή τους έχουμε το είδος και την μορφή των δεδομένων μας, τη συνάρτηση ενεργοποίησης που χρησιμοποιούμε, καθώς και το κατά πόσο τα επιμέρους σφάλματα είναι μεγάλα ή μικρά σε σχέση με το βαθμό εκμάθησης (learning rate) που θέλουμε να πετύχουμε. Αξίζει να αναφερθεί πως στις περισσότερες περιπτώσεις η χρήση κάποιας πιο εξεζητημένης συνάρτησης κόστος αντί της Mean Squared Error δεν εγγυάται αναγκαστικά σημαντικές βελτιώσεις στην απόδοση.

1.1.8 Gradient Descent

Όπως έχουμε δει οι μεταβλητές της συνάρτησης κόστους είναι όλες οι μεταβλητές παράμετροι του νευρωνικού δικτύου με την κατάλληλη επιλογή των οποίων θέλουμε να ελαχιστοποιήσουμε την τιμή της.



Εικόνα 10: Αναπαράσταση της ποσότητας Gradient Descent

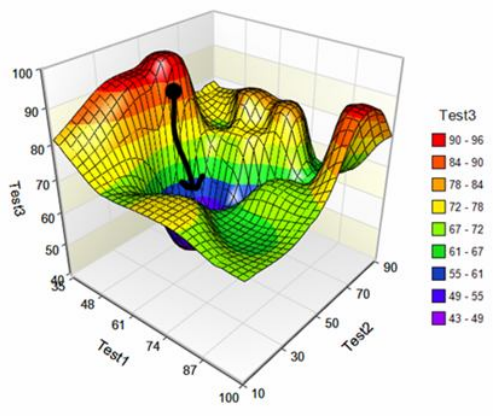
Ας θεωρήσουμε αρχικά συνάρτηση μιας μεταβλητής. Οσοδήποτε πολύπλοκη και αν είναι μπορούμε ξεκινώντας από οποιοδήποτε σημείο της να βρούμε ένα τοπικό της ελάχιστο. Υπολογίζουμε την κλήση της συνάρτησης στο σημείο αυτό υπολογίζοντας την παράγωγο της συνάρτησης στο σημείο και έπειτα επιλέγουμε ως επόμενο ένα αριστερότερο σημείο αν η κλίση της συνάρτησης είναι θετική ή ένα δεξιότερο σημείο αν είναι αρνητική. Επιλέγοντας ένα κατάλληλο βήμα, δηλαδή σωστή επιλογή διαδοχικών σημείων θα καταλήξουμε μετά από διαδοχικά βήματα σε σημείο όπου η κλήση θα είναι μηδενική οπότε και θα έχουμε βρει ένα τοπικό ελάχιστο της συνάρτησης. Εάν σε επόμενο βήμα η κλίση είναι αντίθετης φοράς με προηγουμένως, έχουμε ξεπεράσει το σημείο τοπικού ελαχίστου καθώς είχαμε μεγαλύτερο βήμα από το ενδεδειγμένο, οπότε και πρέπει να το αναπροσαρμόσουμε μικραίνοντας το και να ακολουθήσουμε προς τα πίσω φορά ώστε να βρούμε το ελάχιστο. Μπορούμε να καθορίσουμε το βήμα αναλόγως με την κλήση της συνάρτησης, μικραίνοντάς το όσο η κλήση μικραίνει, ώστε να μην ξεπεράσουμε κατά πολύ το ελάχιστο. Αντιλαμβανόμαστε πως η επιλογή του σημείου που ξεκινάμε καθορίζει σε ποιο τοπικό ελάχιστο της συνάρτησης θα καταλήξουμε (σε περίπτωση που έχει πάνω από ένα), και ότι δεν υπάρχει εγγύηση πως αυτό θα είναι και το ολικό ελάχιστο της συνάρτησης.

Βάσει λοιπόν του παραπάνω σκεπτικού και γενικεύοντας το για μια συνάρτηση δυο μεταβλητών, θέλουμε να δούμε ποια κατεύθυνση δεδομένου ενός αρχικού σημείου πρέπει να ακολουθήσουμε ώστε να επιτύχουμε ένα τοπικό ελάχιστο, μειώνοντας την τιμή της συνάρτησης όσο είναι δυνατό. Σε αυτό το σημείο χρησιμοποιούμε το μαθηματικό εργαλείο gradient, όπως περιγράφεται στην παρακάτων εξίσωση, το οποίο μας δίνει για δεδομένο σημείο την κατεύθυνση που πρέπει να ακολουθηθεί ώστε να βρούμε ένα τοπικό μέγιστο.

$$\nabla f(x,y) = \left\langle f_x(x,y),\ f_y(x,y) \right\rangle = \frac{\partial f}{\partial x}\,\mathbf{i} + \frac{\partial f}{\partial y}\,\mathbf{j}$$

Το αντίθετο του διανύσματος gradient θα μας δώσει την κατεύθυνση για ένα τοπικό ελάχιστο, όπου δηλαδή η τιμή της συνάρτησης θα μειώνεται (gradient descent). Οπότε ο παραπάνω αλγόριθμος εύκολα μπορεί να γενικευτεί για συναρτήσεις πολλών μεταβλητών όπως οι cost functions που συνήθως έχουμε. Επιλέγοντας ένα σημείο πραγματοποιούμε διαδοχικά υπολογισμούς του gradient και μετά του gradient descent

και επιλέγοντας κάθε φορά ένα βήμα προς την κατεύθυνση που μας υποδεικνύει το αποτέλεσμα του υπολογισμού φτάνουμε τελικά σε ένα τοπικό ελάχιστο. Στο σημείο αυτό να τονίσουμε πως δεδομένου ότι η cost function υπολογίζεται ως ο μέσος όρος για όλα τα input με τα οποία εκπαιδεύουμε το δίκτυο μας θα πρέπει να ελαχιστοποιήσουμε τη συνάρτηση κόστους για όλα τα training data.



Εικόνα 11: Ο υπολογισμός διαδοχικών διανυσμάτων σχηματίζει τελικά την παραπάνω πορεία προς το τοπικό ελάχιστο.

Στην πιο γενική περίπτωση των N διαστάσεων, η κατεύθυνση που πρέπει να ακολουθηθεί για να επιτύχουμε το τοπικό ελάχιστο δεν μπορεί να αναπαρασταθεί σχηματικά. Η πληροφορία που μπορούμε να εξάγουμε από ένα τέτοιο διάνυσμα είναι η βαρύτητα κάθε μίας τιμής που αντιστοιχεί σε κάθε παράμετρο (weight ή bias) του δικτύου μας για να καθορίσουμε τελικά ποιες από αυτές χρειάζονται τροποποίηση.

Το αποτέλεσμα του υπολογισμού του gradient descent για κάθε σημείο είναι ένα διάνυσμα όπου κάθε θέση του αναπαριστά μία από τις μεταβλητές της cost function (weight ή bias). Η τιμή κάθε μεταβλητής, ανάλογα με το πόσο μεγάλη είναι, υποδεικνύει τη βαρύτητα την οποία έχει για την ελαχιστοποίηση της συνάρτησης κόστους ενώ το πρόσημο δηλώνει το αν θα πρέπει να αυξήσουμε ή μειώσουμε την τιμή της.

1.1.9 Backpropagation

Η μέθοδος Backpropagation έχει ως στόχο την εκμάθηση και βελτίωση του δικτύου μέσω της καταλληλότερης τροποποίησης των παραμέτρων (weights) με σκοπό την ελαχιστοποίηση της συνάρτησης κόστους για ένα σύνολο δεδομένων εκπαίδευσης. Η τροποποίηση αυτή ξεκινά από το τελευταίο επίπεδο του νευρωνικού δικτύου, ολοκληρώνεται στο επίπεδο εισόδου και σε κάθε βήμα της εμπλέκονται δύο γειτονικά επίπεδα. Σε κάθε επανάληψη της μεθόδου γίνεται τροποποίηση όλων των βαρών που συνδέουν ένα επίπεδο νευρώνων με το επόμενο και τελικά αναδρομικά προς τα πίσω θα τροποποιηθούν όλα τα βάρη του νευρωνικού δικτύου. Η μεταβολή που θα υποστεί κάθε βάρος εξαρτάται από την αντίστοιχη τιμή του διανύσματος gradient descent.

Πιο συγκεκριμένα, η αλλαγή που θα γίνει σε κάθε βάρος δίνεται από την παρακάτω εξίσωση:

$$w_i(\mathrm{t}+1) = w_i(\mathrm{t}) + \Delta w_i(t)$$

όπου $w_i(t+1)$ είναι η νέα τιμή του βάρους i, $w_i(t)$ η παλιά τιμή του βάρους i και

Η ποσότητα $\Delta w_i(t)$ εκφράζει το πόσο πρέπει να μεταβληθεί το κάθε βάρος (διάνυσμα gradient descent).

Σε ένα νευρωνικό δίκτυο με πολλά επίπεδα για το επίπεδο εξόδου από το οποίο και ξεκινάμε την τροποποίηση των βαρών, ο υπολογισμός του νέου βάρους προκύπτει από τον παρακάτω τύπο:

$$W_{j,i} = W_{j,i} + \alpha * \alpha_j * \Delta_i$$

όπου α είναι ο συντελεστής μάθησης (learning rate), aj η τιμή εισόδου του νευρώνα και

$$\Delta_i = Err_i * g'(in_i)$$

όπου Err_i είναι το σφάλμα του κάθε νευρώνα και g η συνάρτηση ενεργοποίησης.

Για το επίπεδο εξόδου το σφάλμα Err_i είναι ξεκάθαρο και υπολογίζεται από τον τύπο y-hw, όπου y είναι η έξοδος του δικτύου για μια συγκεκριμένη είσοδο και hw η αναμενόμενη έξοδος. Η διαφορά του επιπέδου εξόδου με τα κρυφά επίπεδα έγκειται στο γεγονός ότι το σφάλμα στα κρυφά επίπεδα μοιάζει μυστηριώδες επειδή τα δεδομένα εκπαίδευσης δεν αποκαλύπτουν τι τιμή θα πρέπει να έχουν οι κρυφοί κόμβοι. Αποδεικνύεται ότι μπορούμε να οπισθοδιαδώσουμε (back-propagate) το σφάλμα από το επίπεδο εξόδου στα κρυφά επίπεδα.

Θα πρέπει λοιπόν να ορίσουμε μια ποσότητα ανάλογη με τον όρο σφάλματος των κόμβων εξόδου. Στο σημείο αυτό είναι που θα κάνουμε οπισθοδρόμηση σφάλματος βασιζόμενοι στην ιδέα ότι κάθε κρυφός κόμβος j είναι “υπεύθυνος” για ένα κλάσμα του σφάλματος Δ_i σε κάθε έναν από τους κόμβους εξόδου στους οποίους συνδέεται. Έτσι οι τιμές Δ_i διαιρούνται ανάλογα με την ισχύ της σύνδεσης μεταξύ του κρυφού κόμβου και του κόμβου εξόδου και διαδίδονται προς τα πίσω για να δώσουν τιμές Δ_j για το κρυφό επίπεδο.

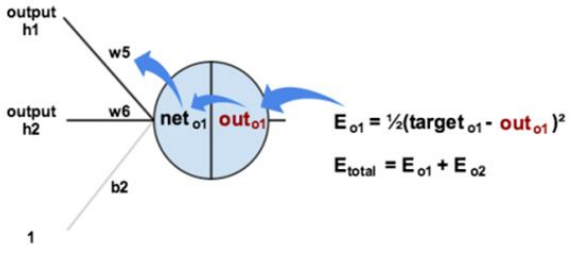
Ο υπολογισμός του νέου βάρους για τα κρυφά επίπεδα δίνεται από τον τύπο:

$$W_{k,j} = W_{k,j} + \alpha * \alpha_k * \Delta_j$$

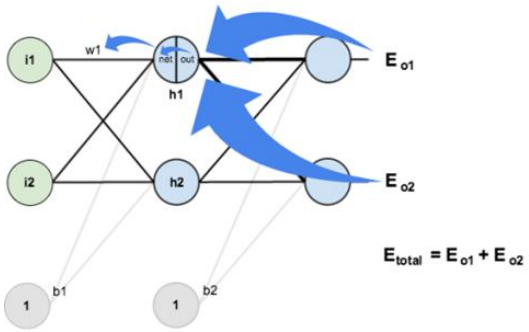
όπου α είναι ο συντελεστής μάθησης (learning rate), αk η τιμή εισόδου του νευρώνα και

$$\Delta_j = g'(in_j) * \sum_i W_{j,i} * \Delta_i$$

όπου $W_{j,i}$ είναι το βάρος που συνδέει το νευρώνα j με τον νευρώνα i και g η συνάρτηση ενεργοποίησης.



Εικόνα 12: Υπολογισμός του σφάλματος



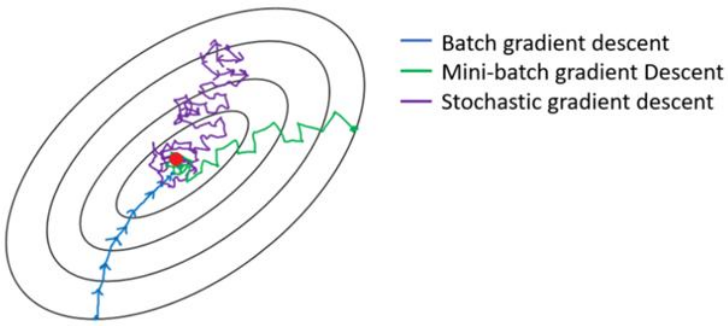
Εικόνα 13: Υπολογισμός του σφάλματος για το κρυφό επίπεδο

1.1.9.1 Learning rate

Παρατηρώντας τις παραπάνω εξισώσεις βλέπουμε πως η μεταβολή των βαρών εξαρτάται από το γινόμενο του διανύσματος gradient descent με μία σταθερά α. Η ταχύτητα σύγκλισης της μεθόδου backpropagation εξαρτάται από την τιμή της σταθεράς αυτής. Η τιμή της πρέπει να είναι αρκούντως μικρή για να εγγυάται τη σύγκλιση, αλλά όχι πάρα πολύ μικρή, γιατί η σύγκλιση γίνεται πολύ αργή. Η καλύτερη επιλογή της εξαρτάται κατά πολύ από το πρόβλημα και από το σχήμα της συνάρτησης κόστους στο χώρο των βαρών. Τα ελάχιστα που έχουν ευρύ σχήμα δίνουν μικρές παραγώγους, επομένως, μεγάλες τιμές του α οδηγούν σε ταχύτερη σύγκλιση. Από την άλλη πλευρά, για απότομα και στενά ελάχιστα απαιτούνται μικρές τιμές του α ώστε να αποφευχθεί η υπέρβαση του ελαχίστου.

1.1.9.2 Διαδικασία μάθησης

Μέχρι τώρα έχουμε περιγράψει τον τρόπο με τον οποίο λειτουργεί η μέθοδος gradient descent και η backpropagation. Στο σημείο αυτό θα εξετάσουμε τους τρόπους με τους οποίους εκμεταλλευόμαστε τα δεδομένα εκπαίδευσης που έχουμε στη διάθεσή μας καθώς και κάποια άλλα ζητήματα που προκύπτουν πολλές φορές κατά τη διαδικασία εκπαίδευσης ενός νευρωνικού δικτύου.



Εικόνα 14: Κύριες στρατηγικές εκπαίδευσης

Αναφορικά με το πρώτο ζήτημα, υπάρχουν τρεις κύριες στρατηγικές με τις οποίες χρησιμοποιούνται τα δεδομένα εκπαίδευσης. Η πρώτη στρατηγική ονομάζεται Επεξεργασία κατά συρροή (Batch mode) κατά την οποία τα βάρη του δικτύου τροποποιούνται μόνο μία φορά μετά την εμφάνιση όλων των ζευγών εισόδου-επιθυμητής εξόδου

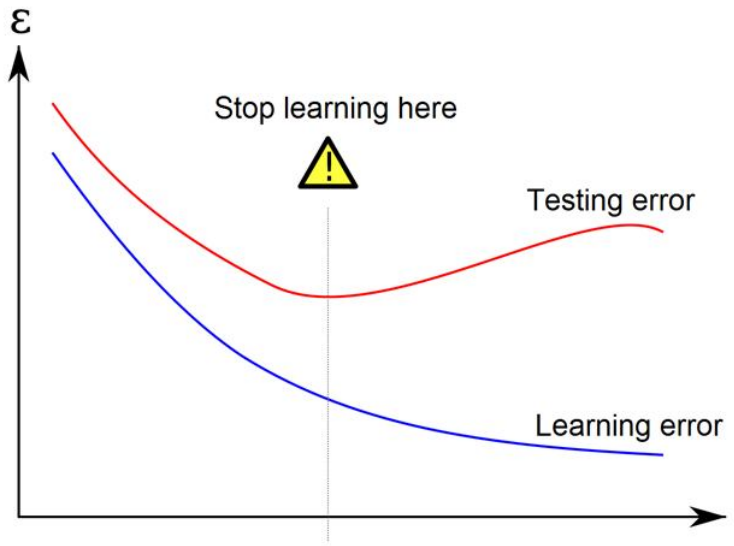
στο δίκτυο. Η δεύτερη στρατηγική ονομάζεται Επεξεργασία κατά μόνας (Pattern ή Online mode) και στην περίπτωση αυτή τα βάρη ανανεώνονται για κάθε ένα από τα δεδομένα εκπαίδευσης. Τέλος, μία ενδιάμεση στρατηγική είναι η Mini Batch mode κατά την οποία το αρχικό σύνολο δεδομένων χωρίζεται σε ορισμένα κομμάτια και γίνεται τροποποίηση στα βάρη μετά από κάθε εμφάνιση των επιμέρους κομματιών.

Μία σημαντική απόφαση που καλούμαστε να πάρουμε κατά την διαδικασία της εκπαίδευσης του δικτύου είναι ο αριθμός των επαναλήψεων (steps) του αλγορίθμου backpropagation. Ο τερματισμός του αλγορίθμου μπορεί να οριστεί είτε όταν η συνάρτηση κόστους γίνει μικρότερη από ένα ορισμένο κατώφλι, είτε όταν το διάνυσμα μερικών παραγώγων της ως προς τα βάρη γίνει μικρό.

Αντιλαμβανόμαστε πως βάσει των παραπάνω κριτηρίων τερματισμού υπάρχει η πιθανότητα παγίδευσης της συνάρτησης κόστους σε κάποιο τοπικό ελάχιστο. Αν το τοπικό ελάχιστο είναι αρκετά “βαθύ”, δηλαδή προσεγγίζει το ολικό ελάχιστο της συνάρτησης, τότε αυτό μπορεί να αποτελεί μία καλή λύση. Ωστόσο, αν τοπικό ελάχιστο δεν είναι τόσο κοντά στο ολικό, η σύγκλιση σε ένα τέτοιο ελάχιστο είναι μη επιθυμητή και ο αλγόριθμος θα πρέπει να αρχικοποιηθεί ξανά με ένα διαφορετικό σύνολο αρχικών συνθηκών.

1.1.9.3 Διαδικασία αξιολόγησης:

Όπως έχουμε ήδη αναφέρει για την εκπαίδευση ενός νευρωνικού δικτύου χρειαζόμαστε ένα σύνολο δεδομένων (dataset). Προκειμένου να σχηματίσουμε ένα δίκτυο ικανό να επιλύσει το πρόβλημά μας, δεν μας αρκεί η διαδικασία εκπαίδευσης που προαναφέραμε. Απαραίτητες είναι οι διαδικασίες ελέγχου και τελικής αξιολόγησης. Για το λόγο αυτό, το σύνολο δεδομένων μας θα πρέπει να χωριστεί σε τρία μέρη, το training set, το validation set και το test set. Το πρώτο χρησιμοποιείται για την εκπαίδευση του δικτύου, δηλαδή για την εύρεση των κατάλληλων βαρών. Το δεύτερο σύνολο χρησιμοποιείται, αφού ολοκληρωθεί η διαδικασία εκπαίδευσης, για τον έλεγχο και την πιθανή τροποποίηση των παραμέτρων που δεν υφίστανται τροποποίηση στην πρώτη φάση και ονομάζονται υπερπαράμετροι (hyperparameters). Τέτοιες παράμετροι είναι ο αριθμός των hidden layers και των hidden units του δικτύου, η επιλογή των αρχικών συνθηκών καθώς και ο αριθμός των επαναλήψεων του αλγορίθμου backpropagation. Μετά από κάθε αλλαγή κατά τη δεύτερη φάση, πρέπει να επαναληφθεί η διαδικασία της εκπαίδευσης. Η διαδικασία αυτή επαναλαμβάνεται έως ότου δεν προκύψει καμία αλλαγή των παραμέτρων στη φάση του validation. Τέλος, ακολουθεί η φάση του testing, κατά την οποία υπολογίζεται η τελική ακρίβεια του δικτύου.



Σχήμα 5: Overfitting

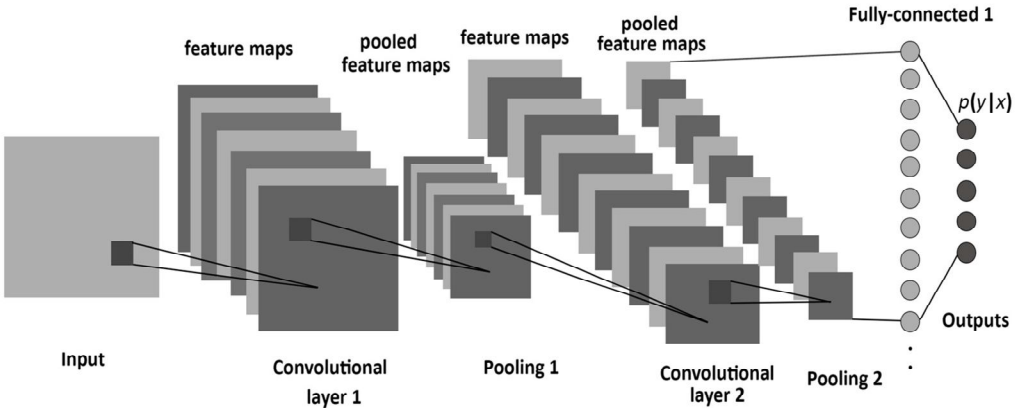
Στο παραπάνω διάγραμμα φαίνεται το πρόβλημα του overfitting, όπου το δίκτυο έχει προσαρμοστεί υπερβολικά στις ιδιαιτερότητες του training set παρουσιάζοντας όμως μεγάλο error (testing error) στο validation set.

1.2 Συνελικτικά Νευρωνικά Δίκτυα (Convolutional Neural Networks)

1.2.1 Εισαγωγή και γενικά χαρακτηριστικά του δικτύου

Τα Συνελικτικά Νευρωνικά Δίκτυα (Convolutional Neural Networks – CNN) αποτελούν μια υποκατηγορία νευρωνικών δικτύων και χρησιμοποιούνται ευρέως στην αναγνώριση εικόνων και βίντεο (image and video recognition) στην υλοποίηση recommender συστημάτων, επεξεργασίας φυσικής γλώσσας (natural language processing) καθώς σε ορισμένα παιχνίδια βοηθώντας τον υπολογιστή να ακολουθήσει μια συγκεκριμένη στρατηγική σε αυτά. Ως συμπλήρωση της σύγκρισης με τον πραγματικό ανθρώπινο νευρικό σύστημα που έχει προηγηθεί, πρέπει να τονιστεί ότι το ιδιαίτερο χαρακτηριστικό που προσπαθούν να προσομοιώσουν τα δίκτυα αυτά, είναι πως μεμονωμένοι φλοιώδεις νευρώνες ανταποκρίνονται σε ερεθίσματα μόνο σε μια περιορισμένη περιοχή του οπτικού πεδίου που είναι γνωστή ως το δεκτικό πεδίο (receptive field). Τα δεκτικά πεδία διαφορετικών νευρώνων επικαλύπτονται εν μέρει έτσι ώστε να καλύπτουν ολόκληρο το οπτικό πεδίο. Έτσι και στα δίκτυα αυτά επιχειρούμε τον διαχωρισμό αυτών των χαρακτηριστικών της εικόνας ώστε η επιμέρους επεξεργασία τους από τους νευρώνες να οδηγήσει στην αναγνώριση της εικόνας εισόδου.

Παρουσιάστηκαν πρώτη φορά το 2012 από τον Alex Krizhevsky που τα χρησιμοποίησε στον διαγωνισμό ImageNet πετυχαίνοντας να μειώσει το σφάλμα σε ένα πρόβλημα κατηγοριοποίησης από 26% σε 15% , επίτευγμα εξαιρετικό για την εποχή εκείνη. Από τότε έχουν εξελιχθεί και χρησιμοποιηθεί ευρέως από πολλές γνωστές εταιρίες όπως η Google η Amazon το Facebook και το Instagram. Αποτελούνται στο πρώτο κομμάτι τους από ορισμένα subsampling η αλλιώς pooling αλλά και convolutional επίπεδα τα οποία εφαρμόζουν τεχνικές επεξεργασίας της εικόνας (ή γενικότερα του δισδιάστατου σήματος) που δίνεται ως είσοδος, ενώ ακολουθεί ένα πλήρως διασυνδεδεμένο νευρωνικό δίκτυο. Ως είσοδο, το δίκτυο δέχεται μια εικόνα διαστάσεων MxNxR όπου M το μήκος, Nτο πλάτος της εικόνας και R ο αριθμός των καναλιών channels αυτής (για μια RGB εικόνα ο αριθμός αυτός είναι 3). Το convolutional επίπεδο αποτελείται από K φίλτρα (filters) διαστάσεων M'xN'xQ, η αλλιώς kernels, όπου τα M' και N' είναι αυστηρά μικρότερα από τις διαστάσεις της εικόνας και το Q μικρότερο η ίσο του αριθμού των channels και μπορεί να διαφέρουν σε κάθε φίλτρο. Τα φίλτρα αυτά δημιουργούν μια τοπικά συνδεδεμένη δομή, η οποία είναι υπεύθυνη για την εξαγωγή K χαρακτηριστικών μεγέθους M-N'+1 με βάση την εικόνα. Στη συνέχεια κάθε ένα από αυτά μετατρέπεται μέσω του επιπέδου pooling σε μια εικόνα μικρότερων διαστάσεων PxL όπου οι μεταβλητές P και L κυμαίνονται από 2 έως 5 ανάλογα με το μέγεθος της αρχικής εικόνας. Μετά το convolutional επίπεδο εφαρμόζεται μια μη γραμμική συνάρτηση στην εικόνα ώστε να εισαχθεί στο δίκτυο η μη γραμμικότητα. Οι εικόνες αυτές που προκύπτουν θα είναι και η είσοδος του τελικού νευρωνικού δικτύου. Ο αριθμός, η σειρά που τοποθετούνται καθώς και τα επιμέρους στοιχεία των pooling και convolutional επιπέδων, αποτελούν μεταβλητές παραμέτρους τις οποίες θα πρέπει να καθορίσουμε ανάλογα με το πρόβλημα που θέλουμε να επιλύσουμε άρα και του δικτύου που εν τέλει θα κατασκευάσουμε. Βάσει λοιπόν της παραπάνω αρχιτεκτονικής, τα CNN έχουν τη δυνατότητα, εκμεταλλευόμενα τη δομή μιας εικόνας δυο διαστάσεων, να εξάγουν αποτελεσματικά τα χαρακτηριστικά της και κατά συνέπεια να χρησιμοποιούν λιγότερες παραμέτρους για την γρηγορότερη εκπαίδευση του νευρωνικού δικτύου. Είναι επίσης ανεξάρτητα του μεγέθους αλλά και του προσανατολισμού του αντικειμένου (shift invariant or space invariant artificial neural networks) ιδιότητα που οφείλεται στην επεξεργασία που υφίσταται η εικόνα στα pooling και convolutional επίπεδα.



Εικόνα 15: Αρχιτεκτονική ενός τυπικού CNN δικτύου

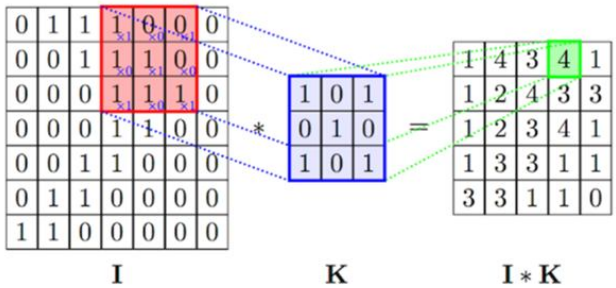
1.2.2 Filtering

Σε ένα Συνελικτικό Νευρωνικό Δίκτυο απαραίτητη είναι η ύπαρξη φίλτρων (filters). Πρόκειται για πίνακες μικρής διάστασης, οι τιμές των οποίων ονομάζονται παράμετροι (parameters) και επιλέγονται με τέτοιο τρόπο ώστε μετά την εφαρμογή τους σε μία εικόνα, να εξαχθούν "ενδιαφέροντα" χαρακτηριστικά για αυτή. Τα φίλτρα αυτά χρησιμοποιούνται για να σχηματιστούν τα λεγόμενα Convolutional Layer. Συγκεκριμένα, αν υποθέσουμε ότι έχουμε σαν είσοδο μία εικόνα μεγέθους 31x31 pixels και εφαρμόσουμε σε αυτή ένα φίλτρο μεγέθους 3x3 τότε θα προκύψει ένας πίνακας μικρότερος από τον αρχικό. Με την εφαρμογη K διαφορετικών φίλτρων θα προκύψουν K διαφορετικοί πίνακες οι οποίο συνθέτουν το Convolutional Layer. Πολλές φορές επιθυμούμε η είσοδος και η έξοδος ενός Convolutional Layer να είναι ίδιας διάστασης. Τα περισσότερα εργαλεία που υλοποιούν CNN δίκτυα αυτοματοποιούν την διαδικασία αυτή με την επιλογή κατάλληλου ταιριάσματος (padding), δηλαδή προσθέτοντας κατάλληλο αριθμό μηδενικών στον πίνακα εισόδου.

Δεδομένης μίας εικόνας εισόδου I και ενός φίλτρου K διαστάσεων h x w, υπολογίζουμε τον συνεστραμμένο (convolved) πίνακα (I*K) εφαρμόζοντας το φίλτρο K σε όλες τις δυνατές θέσεις της εικόνας I από την παρακάτω εξίσωση:

$$(I * K)_{xy} = \sum_{i=1}^{h} \sum_{j=1}^{w} K_{ij} \cdot I_{x+i-1,y+j-1}$$

Για να γίνει πιο κατανοητή η διαδικασία του filtering, στην εικόνα 16 φαίνεται η εφαρμογή ενός φίλτρου μεγέθους 3x3 σε μία εικόνα 7x7.



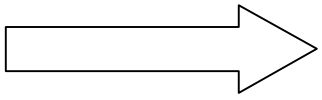
Εικόνα 16: Εφαρμογή ενός φίλτρου 3x3 σε μία εικόνα 7x7

1.2.3 Convolutional Layer

Όπως προαναφέραμε το Convolutional Layer αποτελείται από το σύνολο των εικόνων που προκύπτουν μετά την εφαρμογή Κ φίλτρων σε μία εικόνα. Φυσικό είναι κάποιος να αναρωτηθεί για ποιό λόγο χρησιμοποιούμε πολλά διαφορετικά φίλτρα σε κάθε επίπεδο. Θα προσπαθήσουμε να απαντήσουμε την ερώτηση αυτή λίγο πιο διαισθητικά με τη βοήθεια ενός παραδείγματος. Έστω πως η αρχική μας εικόνα είναι η εικόνα 17 και το φίλτρο που θα χρησιμοποιήσουμε είναι αυτό που φαίνεται στην εικόνα 18. Υποθέτουμε επίσης πως η εικόνα 19 αντισοιχεί στον πίνακα με τις τιμές των pixels του φίλτρου. Εφαρμόζοντας το φίλτρο στο πάνω αριστερό μέρος της αρχικής εικόνας βλέπουμε πως στην αντίστοιχη θέση του τελικού πίνακα (I*K) θα υπάρχει μία αρκετά μεγάλη τιμή που υποδηλώνει αρκετά μεγάλη ταύτιση του φίλτρου με το συγκεκριμένο μέρος της εικόνας (εικόνα 20). Αντίθετα, εφαρμόζοντας το ίδιο φίλτρο στο πάνω δεξιά μέρος της εικόνας (εικόνα 21) βλέπουμε πως το αποτέλεσμα θα είναι πολύ κοντά στο μηδέν, αφού η καμπύλη που αναπαριστά το φίλτρο μας δεν μπορεί να ταυτιστεί με το μοτίβο που υπάρχει στο συγκεκριμένο κομμάτι της εικόνας. Είναι, λοιπόν, απαραίτητη η χρήση πολλών διαφορετικών φίλτρων έτσι ώστε στο Convolutional Layer που θα προκύψει τελικά να υπάρχουν αντιπροσωπευτικά χαρακτηριστικά για όλα τα συστατικά στοιχεία της αρχικής εικόνας.

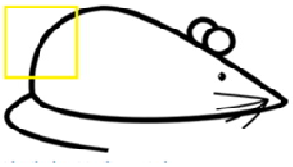
Εικόνα 17: Αρχική εικόνα

Εικόνα 18: Φίλτρο



0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Εικόνα 19: Το φίλτρο της εικόνας 18 υπό μορφή πίνακα



0	0	0	0	0	0	30
0	0	0	0	50	50	50
0	0	0	20	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0

*

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Εικόνα 20: $(I*K)_{IJ} = 6600$

0	0	0	0	0	0	0
0	40	0	0	0	0	0
40	0	40	0	0	0	0
40	20	0	0	0	0	0
0	50	0	0	0	0	0
0	0	50	0	0	0	0
25	25	0	50	0	0	0

*

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Εικόνα 21: $(I*K)_{IJ} = 0$

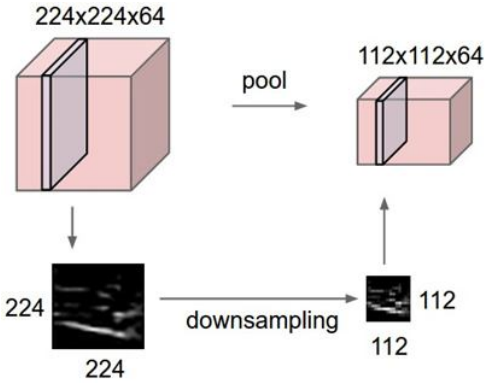
1.2.4 ReLU Layer

Είναι αρκετά σύνηθες σε Συνελικτικά Νευρωνικά Δίκτυα η εφαρμογή μη γραμμικών συναρτήσεων στις εξόδους των Convolutional Layer. Όπως έχουμε αναφέρει στα Convolutional Layer γίνονται γραμμικές πράξεις, δηλαδή προσθέσεις και πολλαπλασιασμοί, ενώ εμείς θα θέλαμε να εισάγουμε στο δίκτυό μας μη γραμμικότητα. Για το λόγο αυτό, μετά από κάθε τέτοιο Layer υπάρχει ένα Layer μη γραμμικής συνάρτησης. Η συνάρτηση που χρησιμοποιείται συνήθως είναι η ReLU και τα επίπεδα αυτά καλούνται ReLU Layers. Οι λόγοι που χρησιμοποιείται η ReLU και όχι κάποια άλλη μη γραμμική συνάρτηση είναι η ταχύτητα εκπαίδευσης του δικτύου καθώς και η αντιμετώπιση του προβλήματος vanishing gradient, πλεονεκτήματα που έχουν αναλυθεί σε προηγούμενο κεφάλαιο.

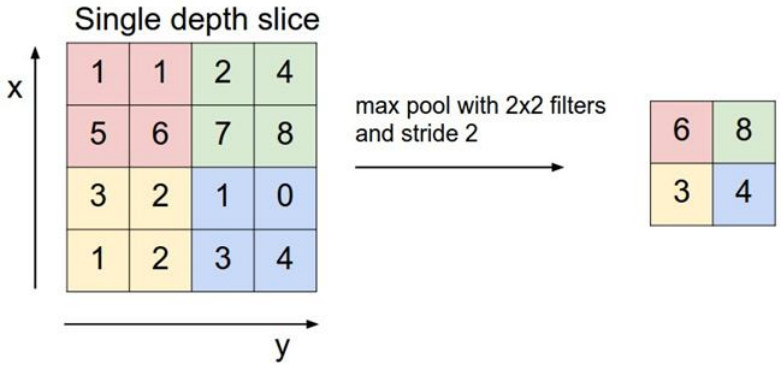
1.2.5 Pooling Layer

Βάσει της γενικής αρχιτεκτονικής των CNN πολλές φορές μπορούν να προστεθούν στο δίκτυο περιοδικά ορισμένα pooling layer, τα οποία παρεμβάλλονται των συνεχόμενων Convolutinal Layer. Η λειτουργία τους αφορά την προοδευτική μείωση των διαστάσεων της εισόδου και αποσκοπεί στην ελάττωση των παραμέτρων, των υπολογισμών καθώς και στον έλεγχο του φαινομένου του overfitting. Το επίπεδο αυτό λειτουργεί αυτόνομα, και τοποθετούμενο σε οποιεσδήποτε θέσεις της αρχικής συνδεσμολογίας προεπεξεργασίας της εικόνας , δεχόμενο ως είσοδο μια εικόνα επιτελεί τη διαδικασία μείωσης των διαστάσεων της (Subsampling), συνήθως βάση του κριτηρίου MAX (MAX Pooling). Η αρχική είσοδος έχει διστάσεις KxLxN όπου K το μήκος L το πλάτος και N το βάθος της. Οι αριθμοί K,L θα υποστούν μείωση ενώ το N θα παραμείνει αμετάβλητο. Με τη χρήση φίλτρων μεγέθους PxM τα οποία τοποθετούνται επαναληπτικά, με βήμα M pixel στην εικόνα, επιλέγεται κάθε φορά το pixel της περιοχής της εικόνας που καλύπτει για τη τρέχουσα επανάληψη το φίλτρο με τη μεγαλύτερη τιμή. Τα υπόλοιπα pixel της περιοχής απορρίπτονται και έτσι τα PxM pixel της αρχικής εικόνας θα αντικατασταθούν

με το pixel που επιλέχθηκε. Το φίλτρο μετατοπίζεται στην επόμενη περιοχή της εικόνας (stride) και η διαδικασία επαναλαμβάνεται έως ότου καλυφθεί ολόκληρη και παραχθεί η αντίστοιχη μικρότερων διαστάσεων. Οι τιμές του P, M που συνήθως επιλέγονται είναι μικρές (2-5). Σύμφωνα με το παραπάνω κριτήριο σε κάθε βήμα του αλγορίθμου επιλέγεται ο μεγαλύτερος από P*M αριθμούς. Θα μπορούσε επίσης αντί αυτού να υπολογίζεται ο μέσος όρος των τιμών αυτών, τεχνική γνωστή ως Average Pooling η οποία όμως έχει αποδειχθεί λιγότερο αποτελεσματική σε σχέση με το Max Pooling. Αντιλαμβανόμαστε πως οι παράμετροι P, M που καθορίζουν το μέγεθος του φίλτρου διαφέρουν ανάλογα με το πρόβλημα άρα και οι τιμές τους θα πρέπει να καθορίζονται δυναμικά.



Εικόνα 22: Pooling and Downsampling



Εικόνα 23: Max pooling with a 2x2 filter

Στις δύο παραπάνω εικόνες αποτυπώνεται η διαδικασία του Pooling.

Στην εικόνα 22 παρατηρούμε ότι το βάθος της εικόνας παραμένει αμετάβλητο ενώ ύψος και πλάτος υποδιπλασιάζονται.

Στην εικόνα 23 με τη χρήση ενός φίλτρου μεγέθους 2x2 πραγματοποιείται Max Pooling για μία εικόνα διαστάσεων 4x4. Από την κάθε περιόχη της εικόνας που εφαρμόζουμε το φίλτρο επιλέγεται το pixel με τη μεγαλύτερη τιμή και τοποθετείται στην αντίστοιχη θέση της νέας εικόνας, αντικαθιστόντας τα 4 pixels της αρχικής.

1.2.6 Backpropagation

Η εκπαίδευση ενός συνελικτικού νευρωνικού δικτύου συνίσταται στον καθορισμό των τιμών των φίλτρων, τα οποία χρησιμοποιούνται στα Convolutional Layers και στην εκπαίδευση του πλήρως διασυνδεδεμένου δικτύου, που βρίσκεται στο τελικό του τμημά. Αναφορικά με το δεύτερο σκέλος, πραγματοποιείται με την μέθοδο backpropagation που έχει αναλυθεί παραπάνω. Ως προς τα επίπεδα προεπεξεργασίας της εικόνας, τα pooling layers δεν επηρεάζουν τη διαδικασία εκμάθησης άρα δεν απαιτείται ο υπολογισμός παραμέτρων που τα αφορούν, ενώ για τα Convolutional Layers θα πρέπει να ακολουθηθεί η παρακάτω διαδικασία.

Στη φάση αυτή σκοπός μας είναι να τροποποιήσουμε κατάλληλα τις τιμές των pixels (βάρη) των φίλτρων. Για να γίνει αυτό χρειάζεται να υπολογίσουμε το πόσο θα μεταβληθεί κάθε βάρος του φίλτρου. Η ποσότητα αυτή φαίνεται στην παρακάτω εξίσωση:

$$\frac{\partial E}{\partial w^l_{m',n'}} = \sum_{i=0}^{H-k_1} \sum_{j=0}^{W-k_2} \delta^l_{i,j} o^{l-1}_{i+m',j+n'}$$

όπου το l είναι ο αριθμός του layer στο δίκτυο, το H και το W οι διαστάσεις της εικόνας εισόδου στο επίπεδο αυτό, τα k_1 και k_2 οι διαστάσεις του φίλτρου και το o^{l-1} είναι η έξοδος του προηγούμενου επιπέδου δηλαδή η είσοδος του επιπέδου l.
Κάθε βάρος του φίλτρου συνεισφέρει σε κάθε pixel της εικόνας που προκύπτει στην έξοδο του layer. Συνεπώς, κάθε αλλαγή που υφίσταται επηρεάζει όλη την εικόνα εξόδου και άρα όλες οι αλλαγές μαζί επηρεάζουν το τελικό σφάλμα. Όπώς φαίνεται και από την παραπάνω εξίσωση η μεταβολή κάθε βάρους είναι συνάρτηση όλων των pixel της εικόνας εξόδου.

Εκτός αυπό την παραπάνω εξίσωση χρειαζόμαστε και μία άλλη που να εκφράζει τη διάδοση του σφάλματος από το τέλος προς την αρχή. Πρόκειται για το δ^l δηλαδή όλα τα gradient που προέρχονται από τις εξόδους του επιπέδου l και υπολογίζονται από τον τύπο:

$$\delta^l_{i,j} = \frac{\partial E}{\partial x^l_{i',j'}}$$

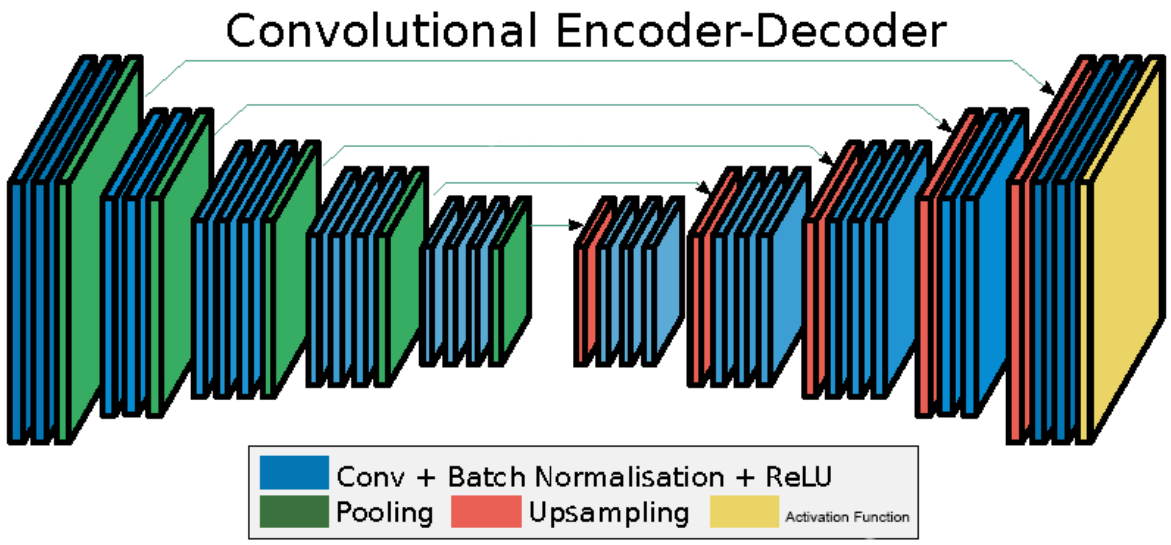
όπου

$$\frac{\partial E}{\partial x^l_{i',j'}} = \sum_{m=0}^{k_1-1} \sum_{n=0}^{k_2-1} \delta^{l+1}_{i'-m,j'-n} w^{l+1}_{m,n} f'\left(x^l_{i',j'}\right)$$

1.3 Convolutional Encoder - Decoder

1.3.1 Αρχιτεκτονική δικτύου

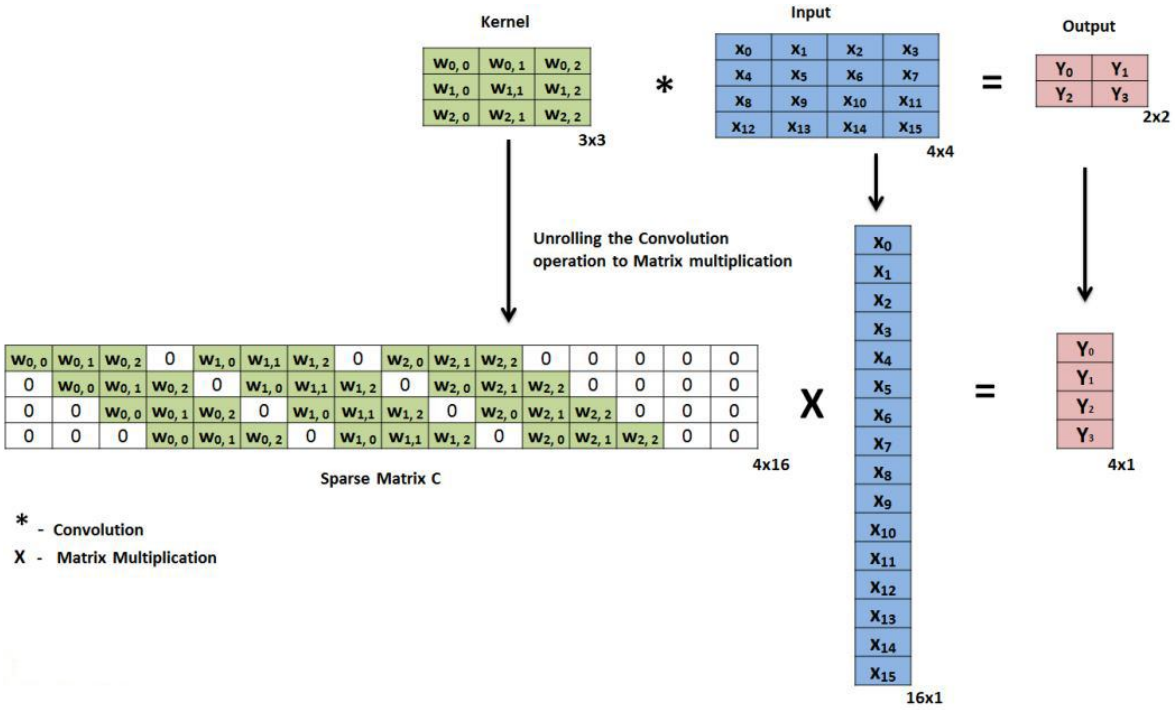
Ένα τέτοιο δίκτυο αποτελείται από 2 κύρια μέρη, το encoder και decoder part. Το πρώτο χρησιμοποιείται για την μείωση της διάστασης των δεδομένων εισόδου (downsampling). Συγκεκριμένα, απαρτίζεται από διαδοχικά convolutional και pooling layer, τα οποία μειώνουν σταδιακά τη διάσταση και τελικά καταλήγουν στο λεγόμενο bottleneck layer το οποίο δίνεται ως είσοδος στο decoder part. Τα convolutional και pooling layer είναι ακριβώς ίδια με αυτά που χρησιμοποιούνται στα CNN δίκτυα. Στη συνέχεια, ακολουθεί το decoder part το οποίο αυξάνει τη διάσταση του bottleneck layer μέχρι την επιθυμητή διάσταση του output layer (upsampling). Αποτελείται από convolutional και transposed convolutional layer τα οποία εκτελούν την αντίθετη διαδικασία από τα pooling layer, δηλαδή δέχονται στην είσοδό τους ένα διάνυσμα διάστασης k και δίνουν στην έξοδό τους ένα μεγαλύτερο διάνυσμα διάστασης q (q > k).



Εικόνα 24: Convolutional Encoder-Decoder

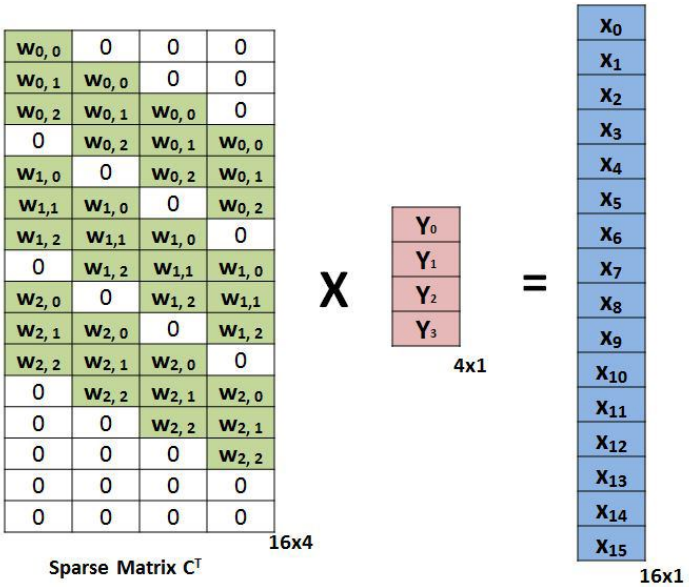
1.3.2 Αντίστροφη Συνέλιξη (Transposed Convolution)

Παρακάτω περιγράφεται η διαδικασία του upsampling μέσω των transposed convolutional layer. Βάσει του ορισμού του convolution που έχουμε δώσει στην ενότητα 1.2, η διαδικασία αυτή μπορεί να αναπαρασταθεί και ως πολλαπλασιασμός πινάκων όπως φαίνεται στην εικόνα 25.



Εικόνα 25: Αναπαράσταση της Συνέλιξης μέσω πολλαπλασιασμού πινάκων

Όπου input το διάνυσμα εισόδου, kernel το φίλτρο και output το διάνυσμα εξόδου. Ορίζεται επίσης ο sparse matrix, κάθε γραμμή του οποίου περιέχει όλα τα στοιχεία του kernel σε διαφορετικές θέσεις, γεγονός που προσομοιώνει τη διαδικασία της συνέλιξης. Ακόμη, προστίθενται μηδενικά στις κενές θέσεις του πίνακα ώστε να μπορεί να γίνει ο πολλαπλασιασμός του πίνακα με το διάνυσμα.
Η αντίστροφη διαδικασία μπορεί εύκολα να οριστεί με τον ίδιο τρόπο, με τη διαφορά ότι στον πολλαπλασιασμό χρησιμοποιείται ο αντίστροφος sparse matrix όπως φαίνεται στην εικόνα 26.



Εικόνα 26: Αναπαράσταση της αντίστροφης Συνέλιξης μέσω πολλαπλασιασμού πινάκων

Στην εικόνα αυτή βλέπουμε πως από ένα πίνακα διάστασης 2x2 παίρνουμε ένα νέο πίνακα διάστασης 4x4.

Συμπερασματικά, ένα δίκτυο encoder-decoder συμπιέζει αρχικά τα δεδομένα, μαθαίνοντας έτσι μία αφηρημένη αναπαράσταση των δεδομένων εισόδου και στη συνέχεια αποσυμπιέζει τα δεδομένα που παράχθηκαν ώστε να παράγει νέα δεδομένα. Διαισθητικά, το μοντέλο αυτό δημιουργεί μέσω του encoder μία χαμηλής διάστασης διανυσματική αναπαράσταση της εισόδου (latent space) την οποία ο decoder μετατρέπει σε διάνυσμα εξόδου διάστασης που εμείς καθορίζουμε.

2 ΑΝΑΚΑΤΑΣΚΕΥΗ ΕΙΚΟΝΑΣ (IMAGE RESTORATION)

2.1 Related work

Αρχικά για την επίλυση του προβλήματος της βελτίωσης μίας εικόνας είχαν χρησιμοποιηθεί αναλυτικές προσεγγίσεις (**analytical approaches**) ενώ αργότερα ξεκίνησε η χρήση του **Deep Learning** (DL) χάρη στην εμφάνιση πολλών δεδομένων. Το DL σε σχέση με τις αρχικές μεθόδους έχει εμφανώς καλύτερη απόδοση όταν ο θόρυβος στην εικόνα είναι μεγάλος. Με DL και όλες τις παρακάτω τεχνικές επιλύονται πολλά παρόμοια προβλήματα όπως το image restoration, super resolution, inpainting.

Πιο συγκεκριμένα, χρησιμοποιούνταν πολυεπίπεδα πλήρως διασυνδεδεμένα νευρωνικά δίκτυα για αναπαράσταση οποιασδήποτε συνάρτησης (με συγκεκριμένη συνάρτηση ενεργοποίησης) [1]. Αργότερα, ξεκίνησε η χρήση **CNN** για επεξεργασία εικόνας και βίντεο, χάρη στην απλούστευση των υπολογισμών (λιγότεροι παράμετροι προς εκμάθηση) και στην ευκολότερη εξαγωγή των χαρακτηριστικών. Τα CNN δίκτυα ξεχωρίζουν από τα υπόλοιπα γιατί εφαρμόζουν συνελίξεις για κάθε επίπεδο.

Όσο τα datasets και η υπολογιστική ισχύς μεγαλώνουν τόσο τα επίπεδα των νευρωνικών δικτύων μπορούν να αυξάνονται και συνεπώς να δίνουν καλύτερα αποτελέσματα αφού μαθαίνουν καλύτερες τις ιδιαιτερότητες των δεδομένων. Μία ακόμη τεχνική που επιταχύνει την διαδικασία της εκμάθησης είναι τα **CNN με Residual Block** [2]**.** Τα δίκτυα αυτά περιέχουν skip connections μεταξύ των layers γεγονός που αυξάνει την απόδοση του δικτύου αφού η διαδικασία της εκπαίδευσης του μικρότερου δικτύου που προκύπτει είναι ταχύτερη.

Μία νέα προσέγγιση αρκετά διαφορετική από τις ήδη υπάρχουσες είναι τα λεγόμενα **Encoder-Decoder CNN** [3]. Το δίκτυο χωρίζεται σε δύο μέρη, το encoder part και το decoder part. Στο πρώτο μέρος γίνεται συμπίεση (downsampling) της εικόνας και στο δεύτερο επανάκτηση του αρχικού μεγέθους (upsampling).

Μία διαφορετική δομή δικτύου παρόμοια με αυτή των Encoder-Decoder είναι οι **Autoencoders** [4]. Σε αυτή την περίπτωση το δίκτυο αποτελείται πάλι από δύο μέρη, το encoder part και το decoder part. Η κύρια διαφορά τους από τους Encoder-Decoder είναι το γεγονός ότι πρόκειται για unsupervised learning, δηλαδή για την διαδικασία της εκπαίδευσης του δικτύου δεν χρησιμοποιούνται labeled dataset. Τα δίκτυα αυτά προσπαθούν δεχόμενα μία είσοδο x να βγάλουν σαν έξοδο ένα y το οποίο θα προσεγγίζει κατά το δυνατό καλύτερα το x. Σκοπός τους δηλαδή είναι να προσομοιώσουν επακριβώς το input και για το λόγο αυτό το input και το output layer έχουν ακριβώς τον ίδιο αριθμό νευρώνων.
Η αρχιτεκτονική ενός τέτοιου δικτύου είναι αρκετά απλή και μοιάζει με αυτή των Συνελικτικών δικτύων. Συγκεκριμένα, το encoder part αποτελείται από αυθαίρετο αριθμό Convolutional Layers μέχρι ένα συγκεκριμένο μέγιστο, κάθε ένα από τα οποία περιέχει αυθαίρετο αριθμό φίλτρων. Κάθε Convolutional Layer ακολουθείται από ένα ReLU Layer, προαιρετικά από ένα Pooling Layer και ενδέχεται να υπάρχουν skip connections μεταξύ των Layers του encoder part και των αντίστοιχων του decoder part. Για τον καθορισμό των υπερπαραμέτρων αυτών χρησιμοποιείται ο αλγόριθμος **(1+λ) evolutional strategy** που πραγματοποιεί αναζήτηση στο χώρο των αρχιτεκτονικών του δικτύου έχοντας ως κριτήριο βελτιστοποίησης την εύρεση της βέλτιστης αρχιτεκτονικής.
Η χρήση Autoencoder δικτύου σε συνδυασμό με τον αλγόριθμο εύρεσης μίας καλής αρχιτεκτονικής έχει αποδειχθεί ότι είναι πιο αποδοτική σε σχέση με άλλες πιο σύγχρονες τεχνικές όπως το Adversarial training για τα προβλήματα inpainting και denoising.

Μία ακόμη unsupervised learning μέθοδος είναι το **Adversarial training** [5]. Χρησιμοποιούνται δύο νευρωνικά δίκτυα Generative Adversarial Networks (GANs) [6], όπου το ένα λειτουργεί ανταγωνιστικά του άλλου και βασίζεται στο πλαίσιο του Zero-sum game. Παρουσιάστηκαν για πρώτη φορά από τον Ian Goodfellow και άλλους ερευνητές από το πανεπιστήμιο του Montreal το 2014, και θεωρείται μια από τις πιο ενδιαφέρουσες ιδέες στην περιοχή της Μηχανικής Μάθησης τα τελευταία χρόνια. Το ένα δίκτυο (Generative Neural Network) είναι υπεύθυνο για την δημιουργία των υποψήφιων εισόδων του δεύτερου δικτύου. Ένα βασικό πλεονέκτημα του δικτύου αυτού είναι πως μπορεί προσαρμοζόμενο στην εκάστοτε κατανομή την οποία ακολουθούν τα δεδομένα εισόδου να παραγάγει με την ίδια ευκολία πολλούς τύπους δεδομένων όπως ήχο και εικόνα. Οι είσοδοι του πρώτου δικτύου είναι μια συλλογή χαρακτηριστικών (latent space) που έχουν προκύψει από μια προεπεξεργασία των εικόνων του συνόλου δεδομένων μας και η έξοδος είναι νέες εικόνες που προσεγγίζουν τις αρχικές. Το πρώτο δηλαδή δίκτυο πραγματοποιεί μια αντίστροφη διαδικασία από αυτή που ακολουθούμε σε προβλήματα κατηγοριοποίησης. Αντί να προσπαθεί, δεδομένων κάποιων χαρακτηριστικών, να κατατάξει δεδομένα, επιχειρεί να δημιουργήσει νέα, βάσει κάποιων αρχικών στοιχειωδών χαρακτηριστικών. Η λογική αυτή ονομάζεται generative model. Αντιλαμβανόμαστε λοιπόν πως το αρχικό δίκτυο αποτελεί μια παραλλαγή ενός Inverse Convolutional Network, καθώς δεχόμενο έναν πίνακα χαρακτηριστικών πραγματοποιεί διαδικασία upsampling μετατρέποντας τα σε εικόνα , αντί διαδικασίας downsampling που θα πραγματοποιούσε ένα κλασσικό CNN σε εικόνες εισόδου για να εξάγει πιθανότητες. Το δεύτερο δίκτυο (Discriminative Neural Network) αναπαριστάται από ένα κλασσικό CNN, το οποίο δέχόμενο ως είσοδο τα δεδομένα αυτά και labelled training data υπολογίζει μια πιθανότητα 0 ή 1 για κάθε δείγμα αναλόγως αν είναι προιόν του πρώτου δικτύου ή “αυθεντικό” αντίστοιχα. Ο όρος Adversarial training έγγυται στο γεγονός πως το πρώτο δίκτυο επιχειρεί να δημιουργήσει δεδομένα τα οποία θα οδηγήσουν το δεύτερο σε λάθος εκτίμιση παρότι προσομοιάζουν τα training data, ενώ το δεύτερο καλείται να λάβει σωστές αποφάσεις για το σύνολο των εισόδων του, οθώντας το πρώτο να παράγει όλο και πιο αληθοφανείς εικόνες. Αυτό επιτυγχάνεται με τη χρήση δυο συναρτήσεων κόστους, μία που σχετίζεται με τις εικόνες του training set που επιθυμούμε να μεγιστοποιεί τις πιθανότητες τους και μια που αφορά τις δημιουργημένες από το πρώτο δίκτυο εικόνες για την οποία θέλουμε να ελαχιστοποιεί τις αντίστοιχες πιθανότητες αυτών. Η εκπαίδευση των δυο τμημάτων του δικτύου πραγματοποιείται σε δυο στάδια σε κάθε ένα από τα οποία διατηρούνται σταθερές οι παράμετροι του ενός δικτύου ενώ καθορίζονται δυναμικά του άλλου.

Μία άλλη τεχνική που χρησιμοποιείται αρκετά για το πρόβλημα Image Restoration και δεν χρησιμοποιεί νευρωνικά δίκτυα είναι η λεγόμενη **Sparse Dictionary Learning**. Η μέθοδος αυτή βασίζεται στην αρχή της γραμμικής άλγεβρας κατά την οποία κάθε διάνυσμα στον N-διάστατο χώρο μπορεί να γραφτεί σαν γραμμικός συνδυασμός N ανεξάρτητων διανυσμάτων που αποτελούν βάση του χώρου. Αντίστοιχα, φτιάχνοντας ένα λεξικό (dictionary) μπορούμε να σχηματίσουμε τις εικόνες που θέλουμε ως γραμμικό συνδυασμό των συστατικών στοιχείων του λεξικού. Το λεξικό αυτό επιθυμούμε να έχει όσο το δυνατό λιγότερα μη μηδενικά στοιχεία γεγονός που δικαιολογεί και το χαρακτηρισμό του σαν “αραιό” (sparse).

2.2 Ορισμοί Προβλημάτων που επιλύουμε

2.2.1 Image Inpainting:

Δεδομένης μίας εικόνας και μίας περιοχής Ω στο εσωτερικό της, το πρόβλημα έγγυται στην τροποποίηση των τιμών των pixel της εικόνας στην περιοχή αυτή, έτσι ώστε αυτή να μην ξεχωρίζει σε σχέση με τα περίχωρά της. Ο σκοπός της διαδικασίας μπορεί να είναι η αποκατάσταση ζημιωμένων τμημάτων μιας εικόνας (όταν για παράδειγμα στην εικόνα υπάρχουν συγκεκριμένες μαύρες περιοχές) ή η αφαίρεση ανεπιθύμητων στοιχείων που υπάρχουν στην εικόνα (για παράδειγμα κάποιο κείμενο που υπάρχει σε αυτή). Η διαδικασία επίσης αναφέρεται και ως “image fill-in” και “image completion”. Στα πειράματα που κάναμε προσπαθήσαμε να διορθώσουμε κατεστραμμένες εικόνες, περιοχές των οποίων είχαν αντικατασταθεί από μαύρα pixel.

2.2.2 Image Denoising:

Αποσκοπεί στην αφαίρεση του θορύβου από μια εικόνα ώστε αυτή να διατηρήσει κατά το δυνατό τα αρχικά χαρακτηριστικά και λεπτομέρειές της. Η εφαρμογή θορύβου στην εικόνα έχει ως αποτέλεσμα την αλλαγή της τιμής ενός αριθμού pixel της βάσει της εξίσωσης του θορύβου. Το denoising αποσκοπεί στην επαναφορά της τιμής των pixel αυτών όσο πλησιέστερα είναι εφικτό στις αρχικές τους τιμές και ταυτόχρονα τη διατήρηση των τιμών των pixel που διατηρούνται σταθερά. Ως αποτέλεσμα λαμβάνουμε μία εικόνα απαλλαγμένη από θόρυβο που προσεγγίζει την αρχική.

Τα είδη θορύβων που χρησιμοποιήσαμε είναι τα εξής:

- ***Gaussian*** : Ο Gaussian θόρυβος είναι στατιστικός θόρυβος με συνάρτηση πυκνότητας πιθανότητας (PDF) ίση με εκείνη της κανονικής κατανομής, η οποία είναι επίσης γνωστή ως Gaussian κατανομή. Συνεπώς, οι τιμές που μπορεί να πάρει ο θόρυβος είναι κατανεμημένες σε κανονική κατανομή.

$$p_G(z) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(z-\mu)^2}{2\sigma^2}}$$

  Με z αντιπροσωπεύεται το gray scale, με μ η μέση τιμή και με σ η τυπική απόκλιση.

- ***Salt and Pepper*** : Είναι θόρυβος που προκύπτει από την τυχαία τοποθέτηση ορισμένου αριθμού λευκών ή μαύρων pixel στην εικόνα. Οι θέσεις και ο αριθμός τους είναι τυχαίες.

2.3 Βιβλιοθήκη Tensorflow

Για την υλοποίηση της εργασίας χρησιμοποιήθηκε η open source βιβλιοθήκη Tensorflow. Η βιβλιοθήκη αυτή αναπτύχθηκε από την Google και χρησιμοποιείται ευρέως για την υλοποίηση νευρωνικών δικτύων και άλλων τεχνικών machine learning. Μία σημαντική δυνατότητα που παρέχεται από τη βιβλιοθήκη είναι η υποστήριξη κάρτας γραφικών (GPU) καθώς με τη χρήση της επιταχύνεται σημαντικά η διαδικασία εκπαίδευσης του δικτύου. Για το λόγο αυτό, χρησιμοποιήσαμε μία κάρτα GeForce GTX 860M της Nvidia.

Ακόμη, ένας εξίσου σημαντικός λόγος που αποφασίσαμε να αναπτύξουμε την εφαρμογή μας στο περιβάλλον αυτό είναι η ίδια του η αρχιτεκτονική. Πιο συγκεκριμένα, το Tensorflow χρησιμοποιεί έναν dataflow graph για την αναπαράσταση των εργασιών (κόμβοι) που πρόκειται να γίνουν καθώς και των σχέσεων μεταξύ τους (ακμές). Έτσι, ο προγραμματιστής έχει τη δυνατότητα να ορίσει στην αρχή το μοντέλο που θα επιλύσει το πρόβλημα και στη συνέχεια να ξεκινήσει ένα Session κατά το οποίο θα εκτελέσει ορισμένες από τις εργασίες του γράφου. Με το τρόπο αυτό, οι εργασίες μπορούν να εκτελεστούν πιο αποδοτικά, αφού η ανάθεσή τους μπορεί να γίνει σε διαφορετικούς επεξεργαστές ή κάρτες γραφικών.

Ένα ακόμη δυνατό σημείο της αρχιτεκτονικής αυτής είναι το γεγονός ότι προσομοιώνει σε μεγάλο βαθμό τον τρόπο λειτουργίας των νευρωνικών δικτύων, καθώς ένα νευρωνικό δίκτυο μπορεί να θεωρηθεί ως ένας γράφος. Γνωρίζοντας, λοιπόν, πως η πιο χρονοβόρα διαδικασία είναι η εκπαίδευση του δικτύου, η ίδια επιταχύνεται σημαντικά όταν οι εργασίες του προγράμματος αναπαρίστανται ως γράφος καθώς με την αρχιτεκτονική αυτή μοντελοποιούνται πιο αποδοτικά οι βασικές λειτουργίες των νευρωνικών δικτύων. Για παράδειγμα η διαδικασία της οπισθοδρόμησης κατά την εκπαίδευση μπορεί να γίνει με ένα απλό backtrack στον προηγούμενο κόμβο και επίσης η είσοδος ενός επιπέδου που αναπαρίσταται από ένα κόμβο συμπίπτει με την έξοδο του προηγούμενου κόμβου του γράφου.

2.4 Image Restoration in MNIST Dataset

2.4.1 Περιγραφή Dataset, εισόδου και εξόδου του προγράμματος

Χρησιμοποιήθηκε το MNIST dataset, το οποίο αποτελεί μια συλλογή χειρόγραφων ψηφίων (0-9) σε αναπαράσταση gray-scale εικόνων, και περιέχει 60.000 δείγματα εκπαίδευσης (training set) και 10.000 δείγματα ελέγχου (test set). Όλες οι εικόνες είναι κανονικοποιημένες και ίδιων διαστάσεων 28x28 με όλα τα ψηφία να έχουν τον ίδιο προσανατολισμό. Αποτελεί ένα σχετικά απλό Dataset το οποίο δεν απαιτεί πολύπλοκη μορφοποίηση και προεπεξεργασία. Για το πρόβλημα image inpainting έχουμε υλοποιήσει δύο εκδοχές. Στην πρώτη εφαρμόζεται στις εικόνες η συνάρτηση add_black_square_to_image η οποία αντικαθιστά με ένα τετραγωνικό πλαίσιο μαύρου χρώματος και διάστασης hxh την κεντρική περιοχή της εικόνας, τοποθετώντας τιμή pixel ίση με το 0 στα αντίστοιχα σημεία της. Στη δεύτερη εκδοχή πραγματοποιείται παρόμοια διαδικασία, με το μαύρο πλαίσιο να τοποθετείται στο αριστερό μισό κομμάτι τις εικόνας. Με τον τρόπο αυτό προκύπτει η είσοδος του δικτύου. Η αντίστοιχη είσοδος για το πρόβλημα του denoising κατασκευάζεται με την τοποθέτηση θορύβου στα αρχικά labels. Με τη χρήση είτε της συνάρτησης Gaussian noise είτε της salt_and_pepper συνάρτησης τις οποίες έχουμε ορίσει εφαρμόζεται στις εικόνες Gaussian ή Salt and Pepper θόρυβος αντίστοιχα δημιουργώντας εικόνες εισόδου για το δίκτυο μας για το συγκεκριμένο τύπο προβλήματος. Η έξοδος του δικτύου για το denoising πρόβλημα είναι η εικόνα εισόδου απαλλαγμένη όσο το δυνατό περισσότερο από θόρυβο ώστε να προσεγγίζει το αρχικό label. Για το inpainting απαιτείται η κλήση της συνάρτησης fill_dark_part ώστε να τοποθετήσει τα pixel εξόδου του δικτύου που αντιστοιχούν στο αποχρωματισμένο κομμάτι της εικόνας στις αντίστοιχες θέσεις της εικόνας εισόδου, ώστε τελικά αυτή να ανακατασκευαστεί προσεγγίζοντας το αρχικό label. Συγκεκριμένα, στο inpainting πρόβλημα παρότι η έξοδος του δικτύου είναι ίδιου μεγέθους με την είσοδο χρησιμοποιούμε εν τέλει μόνο τις προβλέψεις των “άγνωστων” pixel εισόδου του αποχρωματισμένου κομματιού. Για τη δοκιμή του δικτύου χρησιμοποιείται το test set, στο οποίο εφαρμόζονται όλες οι παραπάνω διαδικασίες και τεχνικές αναλόγως του προβλήματος που κάθε φορά επιλύουμε.

2.4.2 Περιγραφή Δομής και Λειτουργίας του CNN Δικτύου

Το δίκτυο το οποίο έχουμε υλοποιήσει ακολουθεί τη βασική αρχιτεκτονική ενός CNN δικτύου. Αποτελείται αρχικά από το επίπεδο εισόδου, από δύο convolutional layer και δύο pooling layer τα οποία διαδέχονται το ένα το άλλο ξεκινώντας με convolutional layer, στη συνέχεια από ένα “πυκνά” συνδεδεμένο (dense) νευρωνικό δίκτυο και τέλος από το επίπεδο εξόδου. Αρχικά ορίζουμε τη μορφή των tensor βάσει της οποίας η πληροφορία θα οδηγείται από το ένα επίπεδο στο άλλο και η οποία είναι η εξής: [BATCH_SIZE, IMAGE_WIDTH, IMAGE_HEIGHT, CHANNELS]. Το IMAGE_WIDTH αντιπροσωπεύει το πλάτος της εικόνας και έχει τιμή 28, το IMAGE_HEIGHT το ύψος της και έχει τιμή επίσης 28, ενώ το CHANNELS το πλήθος των χρωματικών καναλιών της εικόνας που στην περίπτωση έχει τιμή 1 αφού επεξεργαζόμαστε μονοχρωματικές εικόνες. Το πρώτο στοιχείο αφορά το BATCH_SIZE, το πλήθος δηλαδή των δειγμάτων που θα χρησιμοποιηθούν σε κάθε βήμα υπολογισμού του gradient descent κατά τη φάση της εκπαίδευσης του δικτύου και επιλέγουμε ως τιμή το -1. Κάτι τέτοιο μας επιτρέπει να το καθορίσουμε αργότερα ανάλογα με το μέγεθος των δεδομένων εισόδου μας , θεωρώντας το ως υπερπαράμετρο και διατηρώντας σταθερές τις υπόλοιπες διαστάσεις του tensor. Για να μετατρέψουμε την είσοδο μας στη μορφή αυτή χρησιμοποιούμε τη συνάρτηση reshape. Στη συνέχεια δημιουργούμε το πρώτο

convolutional layer με τη χρήση της συνάρτησης conv2d, όπου εφαρμόζουμε 32 φίλτρα διάστασης 5x5 με συνάρτηση ενεργοποίησης Relu. Ως είσοδο του επιπέδου αυτού θεωρείται η έξοδος του προηγούμενου (κανόνας που μπορεί να γενικευτεί και ισχύει για οποιαδήποτε διαδοχικά επίπεδα) δηλαδή η αρχική είσοδος του προγράμματος μας. Με το όρισμα padding = same επιλέγουμε να διατηρήσουμε σταθερή τη μορφή του tensor που ορίσαμε παραπάνω, τοποθετώντας μηδενικά στις κατάλληλες θέσεις της εισόδου του επιπέδου ώστε οι διαστάσεις της εικόνας εισόδου να ταυτίζονται με αυτές της εξόδου. Ακολουθεί ένα pooling layer το οποίο δημιουργούμε με τη χρήση της συνάρτησης max_pooling2d(), το οποίο εφαρμόζει max pooling διάστασης 2x2 με τιμή stride 2. Η τιμή αυτή υποδηλώνει πως τα φίλτρα τοποθετούνται με βήμα 2 χωρίς να υπάρχει επικάλυψη μεταξύ διαδοχικών τοποθετήσεων τους με τις υποπεριοχές που το καθένα εξάγει να απέχουν απόσταση 2. Έπειτα τοποθετούνται διαδοχικά ένα convolutional layer όπου εφαρμόζονται 64 φίλτρα μεγέθους 5x5 και ένα pooling layer που πάλι χρησιμοποιείται max pooling διάστασης 2x2 με stride 2. Η ροή της πληροφορίας και του τρόπου αναπαράστασης της βάσει των tensor έως αυτό το σημείο είναι η εξής: Ως αρχική μορφή tensor όπως και περιγράψαμε έχουμε τις εξής: [-1,28,28,1]. Μετά την εφαρμογή του πρώτου convolutional layer γίνεται [-1,28,28,32] με το πλάτος και ύψος να παραμένουν ίδια, χάρη στο όρισμα padding = same άλλα με τα κανάλια να γίνονται 32 ώστε να αποθηκεύσουν την πληροφορία των 32 φίλτρων που το επίπεδο εφαρμόζει στην εικόνα. Η δομή αυτή δίνεται ως είσοδος στο πρώτο pooling layer από το οποίο προκύπτει ως έξοδος ένα tensor μορφής [-1,14,14,32]. Το tensor αυτό δίνεται ως είσοδος στο δεύτερο convolutional layer από το οποίο προκύπτει έξοδος διάστασης [-1,14,14,64] ενώ στη συνέχεια και με παρόμοια διαδικασία προκύπτει από το δεύτερο pooling layer έξοδος διάστασης [-1,7,7,64]. Στο σημείο αυτό το tensor μετασχηματίζεται σε δυο διαστάσεις [-1,7*7*64] δηλαδή [-1,3136] ώστε να δοθεί ως είσοδος στο “πυκνό” νευρωνικό δίκτυο το οποίο θα έχει 1024 νευρώνες (units). Πραγματοποιείται μέσω της συνάρτησης dropout κανονικοποίηση του δικτύου ώστε το 40% των units του δικτύου να μην χρησιμοποιηθεί κατά την εκπαίδευση διευκολύνοντάς τη. Μετά το dropout, λοιπόν, προκύπτει ένα tensor διάστασης [-1,1024] και δίνεται ως είσοδος στο επίπεδο output_layer το οποίο και παράγει έξοδο διάστασης [-1,IMAGE_SIZE] με τη μεταβλητή IMAGE_SIZE να υποδηλώνει το μέγεθος σε pixel της αναδιαμορφωμένης εικόνας εξόδου.
Ως συνάρτηση κόστους έχει οριστεί η συνάρτηση μέσου τετραγωνικού όπου μέσω της κλήσης tf.reduce_mean(tf.square(labels – output_layer)) υπολογίζεται το μέσο τετραγωνικό σφάλμα μεταξύ των επιπέδων input που αποτελεί την είσοδο του δικτύου και output_layer που αναπαριστά την έξοδο του. Για την ελαχιστοποίηση του σφάλματος και την κατάλληλη παραμετροποίηση του δικτύου χρησιμοποιείται η συνάρτηση GradientDescentOptimizer.

2.4.3 Περιγραφή Δομής και Λειτουργίας του Encoder-Decoder Δικτύου

Το δίκτυο που έχουμε υλοποιήσει αποτελείται από δύο μέρη, τα Encoder και Decoder part.
Το Encoder part αποτελείται από το επίπεδο εισόδου, 2 convolutional layer και 2 pooling layer τα οποία διαδέχονται το ένα το άλλο. Τα convolutional layer αποτελούνται από kernel μεγέθους [3,3], 512 φίλτρα, padding = “same” και συνάρτηση ενεργοποίησης Relu. Τα pooling layer περιέχουν παράθυρο μεγέθους [2,2] και strides = 2. Κατά συνέπεια, δεδομένης μίας εισόδου μεγέθους [28,28,1], η έξοδος του 1ου pooling layer θα έχει μέγεθος [14,14,512] και η έξοδος του 2ου pooling layer (bottleneck) θα έχει μέγεθος [7,7,512].
Ακολουθεί το Decoder part, το οποίο λαμβάνει την έξοδο του Encoder. Αποτελείται από ένα convolutional layer ίδιο με τα προηγούμενα, δύο transpose convolutional layer με

kernel μεγέθους [3,3], 512 φίλτρα, stride = 2 και padding = “same” και τέλος ένα convolutional layer το οποίο διαφέρει σε σχέση με τα άλλα αφού αποτελείται από 1 φίλτρο. Έτσι, όταν το Decoder part λάβει την είσοδο [7,7,512] πραγματοποιεί αρχικά upsampling με το 1ο transpose layer σε [14,14,512], έπειτα με το 2ο transpose layer σε [28,28,512] και τέλος με το convolutional layer, ανακτά το αρχικό μέγεθος της εικόνας, δηλαδή [28,28,1].
Ως συνάρτηση κόστους έχει οριστεί η συνάρτηση μέσου τετραγωνικού όπου μέσω της κλήσης tf.reduce_mean(tf.square(labels – output_layer)) υπολογίζεται το μέσο τετραγωνικό σφάλμα μεταξύ των επιπέδων input που αποτελεί την είσοδο του δικτύου και output_layer που αναπαριστά την έξοδο του. Για την ελαχιστοποίηση του σφάλματος και την κατάλληλη παραμετροποίηση του δικτύου χρησιμοποιείται η συνάρτηση AdamOptimizer.

Η υλοποίηση της αρχιτεκτονικής του Encoder-Decoder διαφέρει από αυτή του απλού CNN δικτύου, καθώς δεν έχουμε φτιάξει κάποιον custom Estimator που να επιλύει το πρόβλημα άλλα έχουμε χρησιμοποιήσει το mid-level API του Tensorflow. Για το λόγο αυτό ήταν αναγκαίος ο ορισμός της βοηθητικής συνάρτησης next_batch(data, num_of_batch) η οποία παίρνει ως όρισμα τα δεδομένα που θέλουμε να διαχωρίσουμε σε ριπές καθώς και τον αριθμό της ριπής στην οποία βρισκόμαστε την εκάστοτε στιγμή και επιστρέφει το ανάλογο υποσύνολο των δεδομένων.
Όπως έχουμε ξανά αναφέρει, το Tensorflow απαιτεί τον ορισμό ενός μοντέλο και στη συνέχεια ένα session στο οποίο θα εκτελεστούν οι εργασίες ώστε να παραχθεί ένα αποτέλεσμα. Για το λόγο αυτό, στην αρχή της συνάρτησης main() ορίζουμε 3 tf.placeholder (inputs, outputs και labels), έπειτα ορίζεται το μοντέλο όπως περιγράφηκε παραπάνω και τέλος εκκινούμε ένα νέο session το οποίο αναθέτει τις τιμές στα inputs και labels και παράγει κάποιες τιμές στη μεταβλητή outputs.

2.4.4 Μετρικές αξιολόγησης των ανακατασκευασμένων εικόνων

2.4.4.1 PSNR

Είναι μια μετρική αξιολόγησης της ποιότητας ανακατασκευής των εικόνων, που χρησιμοποιήσαμε για να διαπιστώσουμε την ικανότητα του δικτύου μας στην επίλυση και των δυο προβλημάτων. Καλείται μέγιστος λόγος του σήματος προς το θόρυβο (PSNR – Peak Signal to Noise Ratio) και αντιπροσωπεύει την αναλογία μεταξύ της μέγιστης δυνατής ισχύος ενός σήματος και της αντίστοιχης του θορύβου που επηρεάζει την πιστότητα της αναπαράστασής του. Επειδή πολλά σήματα έχουν ένα πολύ εκτενές δυναμικό εύρος, το PSNR εκφράζεται συνήθως με βάση τη λογαριθμική κλίμακα ντεσιμπέλ (dB).

Ο υπολογισμός του γίνεται με τη βοήθεια του μέσου τετραγωνικού σφάλματος (MSE – Mean Squared Error) :

$$MSE = \frac{1}{m\,n} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i,j) - K(i,j)]^2$$

Όπου I αναπαριστά την αρχική εικόνα και K την ανακατασκευασμένη. Αμφότερες οι εικόνες έχουν διαστάσεις MxN.

Το PSNR δίνεται βάσει του τύπου:

$$PSNR = 10 \cdot \log_{10}\left(\frac{MAX_I^2}{MSE}\right)$$

Όπου MSE το μέσο τετραγωνικό σφάλμα δυο εικόνων (της αρχικής και της ανακατασκευασμένης) και MAX_I μια εικόνα ίδιων διαστάσεων με τις άλλες δυο άλλα με μέγιστη τιμή ισχύος σε κάθε pixel της (δηλαδή ίση με 255) που αναπαριστά τη μεγίστη ισχύ ενός σήματος διαστάσεων MxN.

Όσο το δυνατόν μεγαλύτερες τιμές του PSNR υποδηλώνουν καλύτερη ποιότητα ανακατασκευής των εικόνων.

2.4.4.2 SSIM

Η SSIM μετρική διαφέρει σε σχέση με την PSNR, η οποία υπολογίζει το απόλυτο σφάλμα μεταξύ των τιμών των pixel δυο εικόνων, συγκρίνοντας ουσιαστικά την ομοιότητα των αντίστοιχων εικονοστοιχείων τους. Ο υπολογισμός της βασίζεται στο γεγονός πως η υποβάθμιση μιας εικόνας μπορεί να θεωρηθεί και ως αλλαγή της δομικής πληροφορίας (structural information) αυτής. Το γεγονός αυτό επαληθεύεται και διαισθητικά, καθώς σε πολλές περιπτώσεις θέλοντας να συγκρίνουμε δυο εικόνες εξετάζουμε τον προσανατολισμό, το περίγραμμα και το σχήμα των αντικειμένων που αναπαριστώνται σε αυτές. Έτσι μέσω του ορισμού της μετρικής αυτής θεωρούμε πως οι σχέσεις των τιμών των pixel που γειτνιάζουν καθορίζουν σε μεγάλο βαθμό τα δομικά χαρακτηριστικά της εικόνας τα οποία μπορούν να είναι το περίγραμμα ή ιδιαίτεροι σχηματισμοί σε συγκεκριμένα σημεία της. Η δομική λοιπόν αυτή πληροφορία αποτυπώνεται μέσω της συσχέτισης των τιμών των γειτονικών pixel των συγκρινόμενων εικόνων.

Η μετρική SSIM ορίζεται ως:

$$\mathrm{SSIM}(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}$$

- X και Y οι δύο συγκρινόμενες εικόνες ίδιων διαστάσεων.
- M_x η μέση τιμή της X εικόνας
- M_y η μέση τιμή της Y εικόνας
- σ_x η τυπική απόκλιση της X εικόνας
- σ_y η τυπική απόκλιση της Y εικόνας

- σ_{xy} η συνδιακύμνση (covariance) των X Y
- c1, c2 μεταβλητές που εξαρτώνται από το εύρος τιμών των εικόνων και χρησιμοποιούνται ώστε να αποφευχθούν προβλήματα στη πράξη της διαίρεσης.

Η μετρική λοιπόν αυτή υπολογίζει την παραμόρφωση της εικόνας ως συνδυασμό τριών παραμέτρων: της απώλειας συσχέτισης (loss of correlation), της παραμόρφωσης φωτεινότητας (luminance distortion) και της διαστρέβλωσης της αντίθεσης (constrast distortion). Οι τρεις αυτοί παράγοντες εμπεριέχονται στην παραπάνω εξίσωση μέσω των παρακάτω σχέσεων:

- Η πρώτη παράμετρος εκφράζεται μέσω του συντελεστή συσχέτισης (correlation) των δυο εικόνων έχει εύρος [-1,1] και λαμβάνει τη βέλτιστη τιμή 1 όταν η μια εικόνα είναι γραμμικός συνδυασμός της άλλης.
- Η δεύτερη παράμετρος έχει εύρος [0,1] και λαμβάνει τη μέγιστη τιμή της 1 όταν οι μέσες τιμές των δύο εικόνων είναι ίδιες.
- Η τρίτη παράμετρος έχει εύρος [0,1] και λαμβάνει τη βέλτιστη τιμή 1 όταν οι τυπικές αποκλίσεις των δυο εικόνων είναι ίδιες.

Βέλτιστη τιμή για την SSIM μετρική είναι η τιμή 1.

2.4.5 Παρουσίαση αποτελεσμάτων

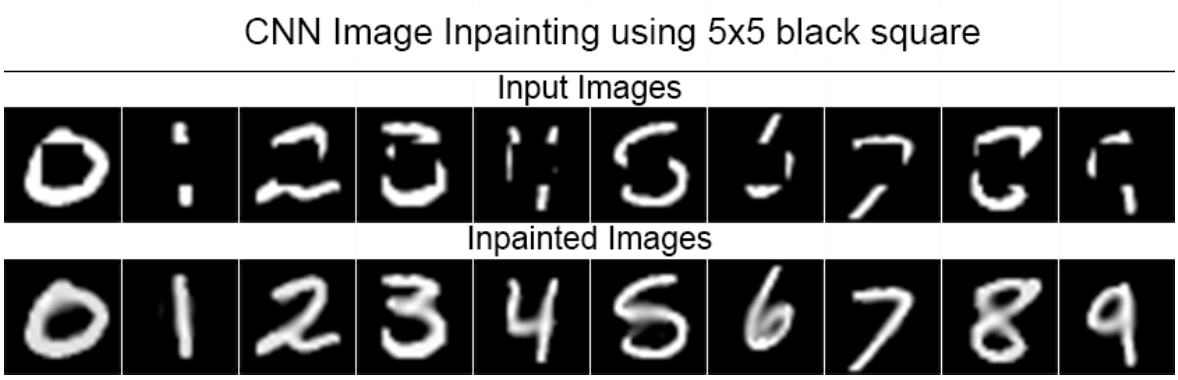
Στη συνέχεια ακολουθούν οι καλύτερες μετρήσεις που προέκυψαν για τα δύο δίκτυα καθώς και οι εικόνες που παράχθηκαν. Αναφορικά με τις εικόνες που παρουσιάζονται παρακάτω επιλέχθηκαν με βάση την ποιότητά τους οπτικά και δεν είναι κατ' ανάγκη βέλτιστες ως προς τις μετρικές που χρησιμοποιήσαμε. Ακόμη, οι μετρήσεις PSNR και SSIM είναι οι μέσες τιμές που προέκυψαν από την αξιολόγηση του test set.
Για το CNN δίκτυο, χρησιμοποιήσαμε batch_size = 100, number_of_steps = 25000, dropout_rate = 0.4 και για το Encoder-Decoder αριθμό φίλτρων = 512. Οι τιμές των παραμέτρων επιλέχθηκαν με γνώμονα την παραγωγή ποιοτικών εικόνων σε ανεκτό χρόνο. Αυτό σημαίνει ότι δεν έχουν χρησιμοποιηθεί οι βέλτιστες τιμές των υπερπαραμέτρων αλλά τιμές αρκετά κοντά σε αυτές, καθώς η επιλογή των βέλτιστων τιμών καθυστερεί αρκετά το χρόνο εκπαίδευσης και απαιτεί πόρους που ο υπολογιστής μας δεν διαθέτει.

2.4.5.1 Image Inpainting

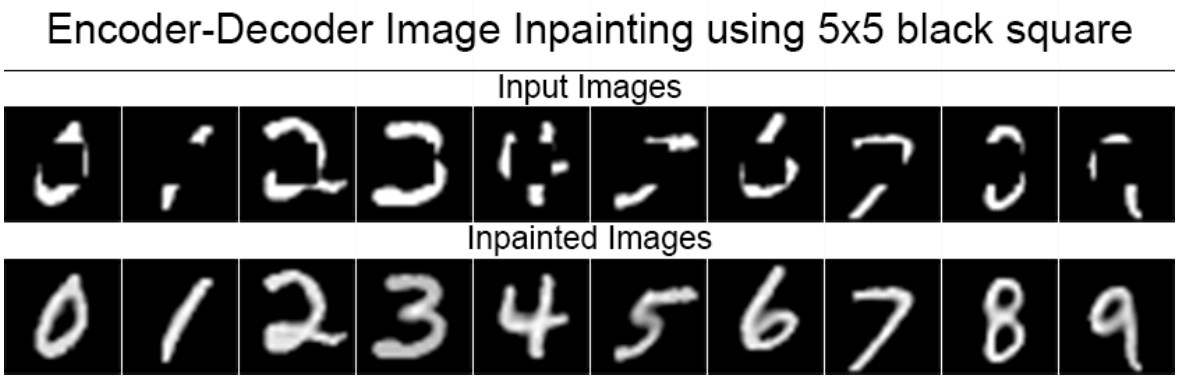
Και για τα δύο είδη δικτύων έχουμε πειραματιστεί με τρία διαφορετικά παράθυρα θορύβου. Πιο συγκεκριμένα, έχουμε τοποθετήσει ένα μαύρο παράθυρο διάστασης 5x5 pixel, ένα 7x7 στα κέντρα των εικόνων καθώς και ένα παράθυρο στο μισό αριστερό μέρος των εικόνων. Σκοπός του δικτύου είναι να συμπληρώσει τις κατάλληλες τιμές pixel στα παράθυρα αυτά.

Πίνακας 1: Μετρήσεις Image Inpainting

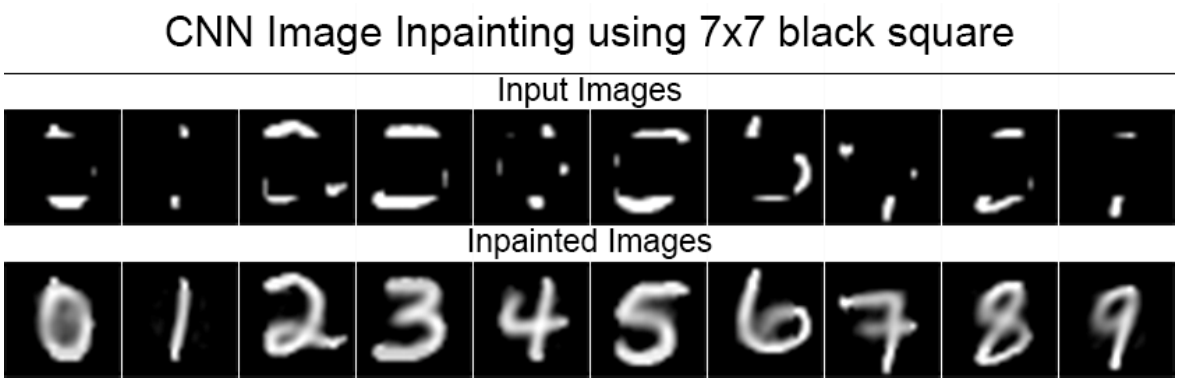
Network - Task	PSNR	SSIM
CNN Inpainting 5x5	22.296930919	0.9214396676907707
Enc-Dec Inpainting 5x5	**22.5693517198**	**0.9292746178341258**
CNN Inpainting 7x7	**18.340896374**	**0.7839343014108258**
Enc-Dec Inpainting 7x7	17.6265811808	0.7754064391044144
CNN Inpainting half dark	19.7161238289	0.8037880505981232
Enc-Dec Inpainting half dark	18.8287715593	0.7691347717590131



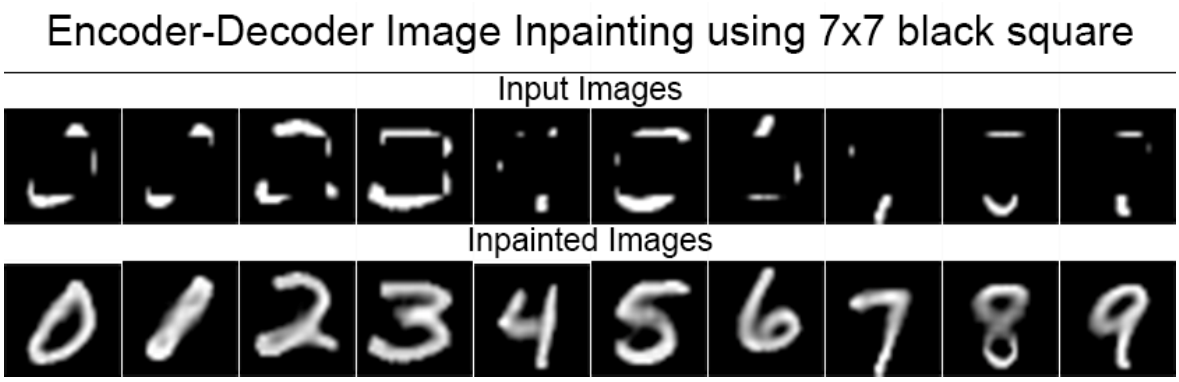
Εικόνα 27: CNN Image Inpainting using 5x5 black square



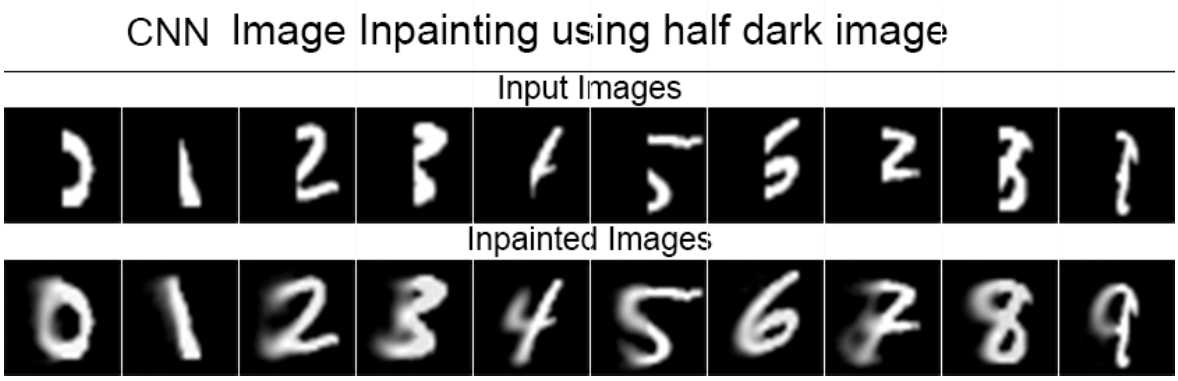
Εικόνα 28: Encoder-Decoder Image Inpainting using 5x5 black square



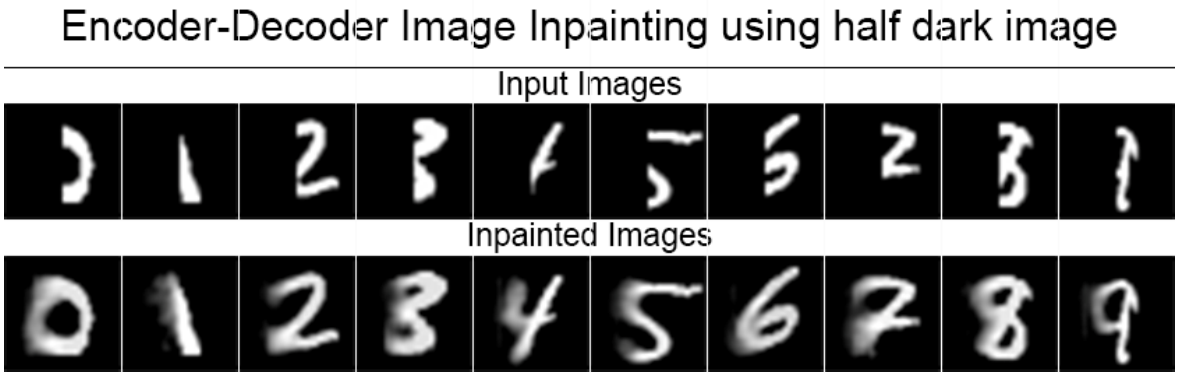
Εικόνα 29: CNN Image Inpainting using 7x7 black square



Εικόνα 30: Encoder-Decoder Image Inpainting using 7x7 black square



Εικόνα 31: CNN Image Inpainting using half dark image



Εικόνα 32: Encoder-Decoder Image Inpainting using half dark image

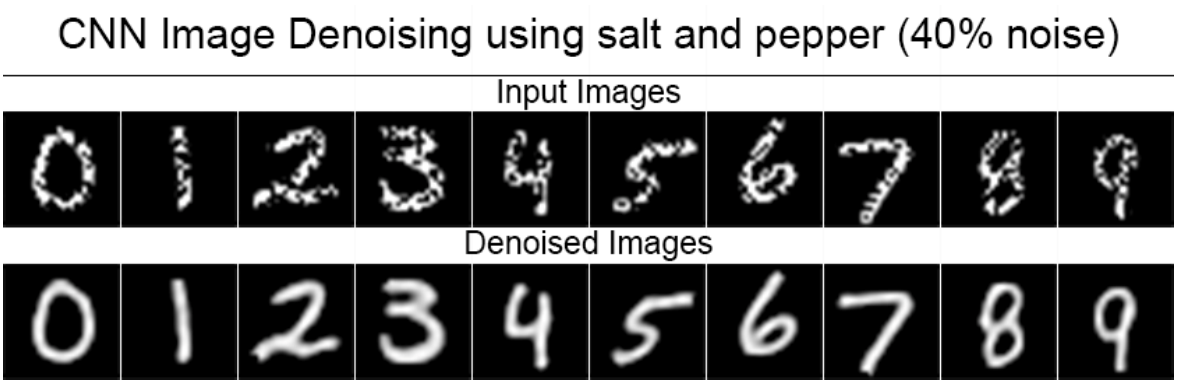
2.4.5.2 Image Denoising

Και για τα δύο είδη δικτύων έχουμε πειραματιστεί με δύο θορύβους καθώς και με διαφορετικές εντάσεις του ίδιου θορύβου. Συγκεκριμένα, στο Gaussian έχουμε πειραματιστεί με 4 τιμές της παραμέτρου sigma (15, 20, 30 και 50). Τα αποτελέσματα στο πρώτο μέρος αφορούν τον Salt and Pepper θόρυβο ενώ στο δεύτερο μέρος τον Gaussian θόρυβο.
Για τις εικόνες που έχουμε προσθέσει Gaussian θόρυβο έχει χρησιμοποιηθεί μόνο το δίκτυο Encoder-Decoder, καθώς το CNN δεν παρουσίασε υψηλή απόδοση με το ρυθμό εκπαίδευσης που χρησιμοποιήσαμε σε όλα τα tasks. Θα μπορούσαμε να αυξήσουμε τα βήματα του αλγορίθμου Gradient Descent ώστε να πάρουμε καλύτερα αποτελέσματα, κάτι τέτοιο όμως δε θα μας ήταν χρήσιμο αφού η εκπαίδευση του δικτύου θα καθυστερούσε αρκετά και τα αποτελέσματα θα ήταν σαφώς χειρότερα από αυτά του Encoder-Decoder.

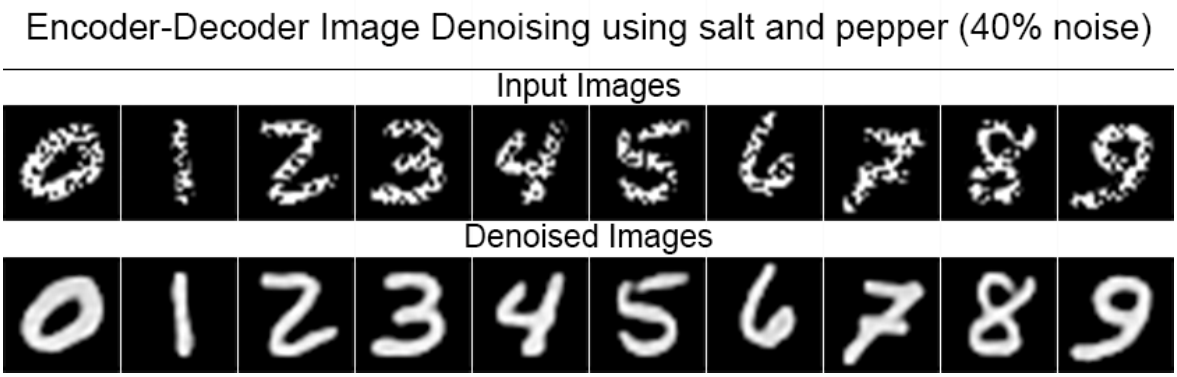
Salt and Pepper θόρυβος:

Πίνακας 2: Μετρήσεις Image Denoising με Salt and Pepper θόρυβο

Network - Task	**PSNR**	**SSIM**
CNN Denoising S&P	20.7730720941	0.9183908249039525
Enc-Dec Denoising S&P	**24.4722276916**	**0.951891031722976**



Εικόνα 33: CNN Image Denoising using salt and pepper (40% noise)

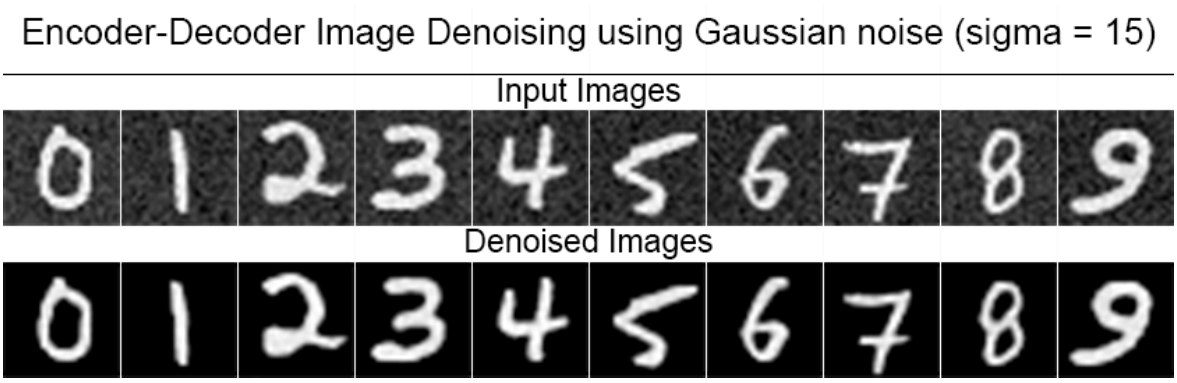


Εικόνα 34: Encoder-Decoder Image Denoising using salt and pepper (40% noise)

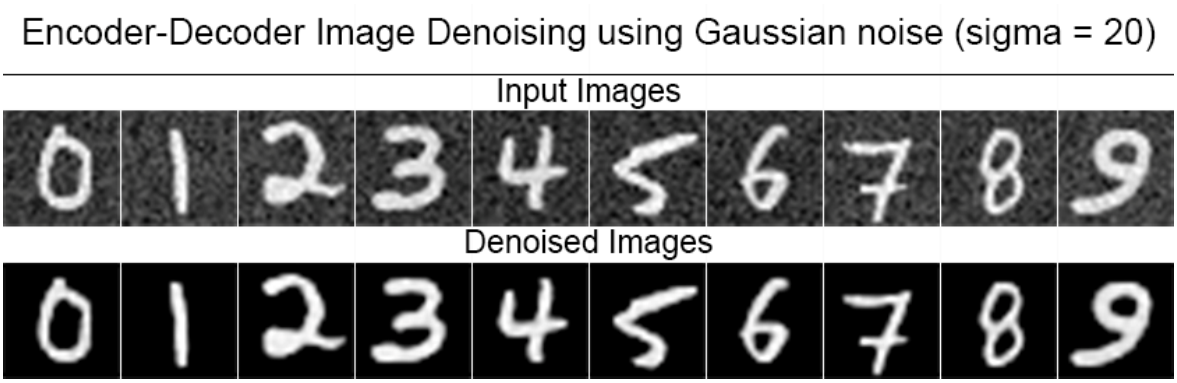
Gaussian θόρυβος:

Πίνακας 3: Μετρήσεις Image Denoising με Gaussian θόρυβο

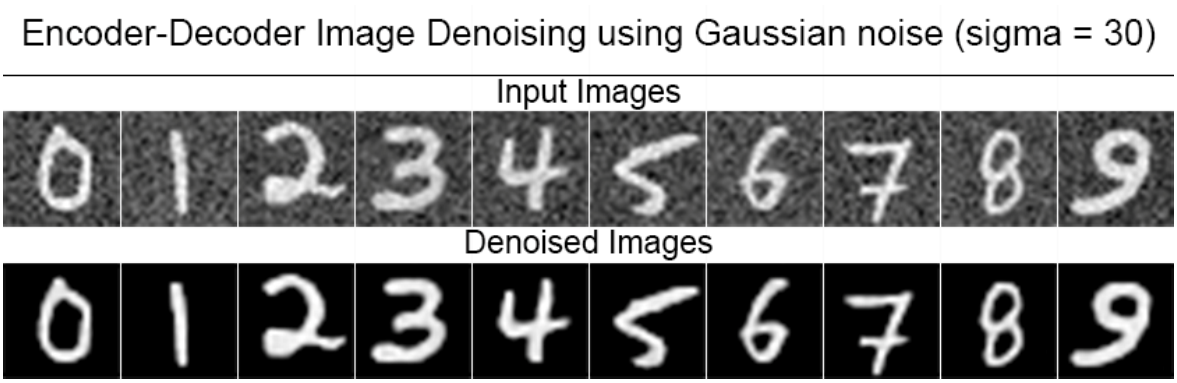
Network - Task	PSNR	SSIM
Enc-Dec Denoising Gaussian (sigma = 15)	28.6926760427	0.9792571112862103
Enc-Dec Denoising Gaussian (sigma = 20)	28.4600046791	0.9737220263974139
Enc-Dec Denoising Gaussian (sigma = 30)	26.9575139793	0.965009625829102
Enc-Dec Denoising Gaussian (sigma = 50)	25.3009132898	0.9456466499033107



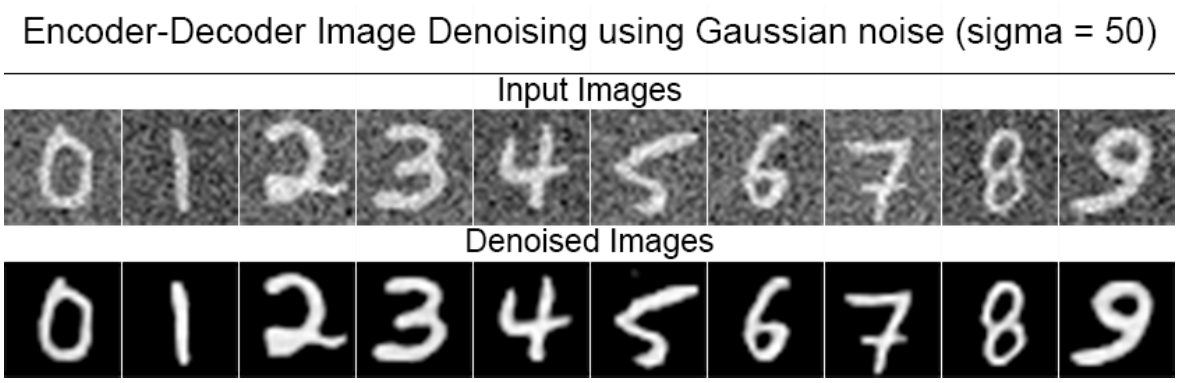
Εικόνα 35: Encoder-Decoder Image Denoising using Gaussian noise (sigma = 15)



Εικόνα 36: Encoder-Decoder Image Denoising using Gaussian noise (sigma = 20)



Εικόνα 37: Encoder-Decoder Image Denoising using Gaussian noise (sigma = 30)



Εικόνα 38: Encoder-Decoder Image Denoising using Gaussian noise (sigma = 50)

2.4.5.3 Πειραματισμός με παραμέτρους των δικτύων

Οι παράμετροι με τις οποίες πειραματιστήκαμε ήταν το batch size, το number_of_steps και το dropout_rate στο πρώτο τύπο δικτύου που υλοποιήσαμε (απλό CNN) και ο αριθμός των φίλτρων (filters) στο δεύτερο τύπο (encoder-decoder). Οι επιλογές μας αυτές βασίστηκαν στο γεγονός πως οι τιμές των παραμέτρων αυτών είχαν φανερή επίδραση στη συμπεριφορά του εκάστοτε δικτύου και στην ποιότητα των αποτελεσμάτων που παρήγαγε βάσει δοκιμών που κάναμε. Πραγματοποιήσαμε εκτελέσεις για διάφορες τιμές batch_size, number_of_steps, dropout_rate και filters αντίστοιχα και λάβαμε τις εξής τιμές PSNR όπως παρουσιάζονται στους παρακάτω πίνακες:

Πίνακας 4: Μετρήσεις για διάφορες τιμές batch_size

Task – Batch_size	**PSNR**
Image Inpainting(5x5) - 20	21.9463796468
Image Inpainting(5x5) - 40	22.1433124857
Image Inpainting(5x5) - 60	22.1671861216
Image Inpainting(5x5) - 80	22.2402920249
Image Inpainting(5x5) - 100	22.296930919
Image Inpainting(5x5) - 120	22.3137967568
Image Denoising S&P - 20	19.8118409303
Image Denoising S&P - 40	20.2802053634
Image Denoising S&P - 60	20.4400316665
Image Denoising S&P - 80	20.8078954383
Image Denoising S&P - 100	20.7730720941
Image Denoising S&P - 120	20.7435289814

Πίνακας 5: Μετρήσεις για διάφορες τιμές number_of_steps

Task – Number_of_steps	**PSNR**
Image Inpainting(5x5) – 500	18.3097784899
Image Inpainting(5x5) – 1000	19.0952223139
Image Inpainting(5x5) – 5000	20.1947748092
Image Inpainting(5x5) – 15000	21.923522337
Image Inpainting(5x5) – 25000	22.296930919

Image Inpainting(5x5) – 35000	22.2137434174
Image Inpainting(5x5) – 45000	22.4550087918
Image Denoising S&P - 500	15.2807303308
Image Denoising S&P - 1000	15.8206108802
Image Denoising S&P - 5000	18.8238262501
Image Denoising S&P - 15000	20.4568328118
Image Denoising S&P - 25000	20.7730720941
Image Denoising S&P - 35000	21.2770613455
Image Denoising S&P - 45000	20.8204040881

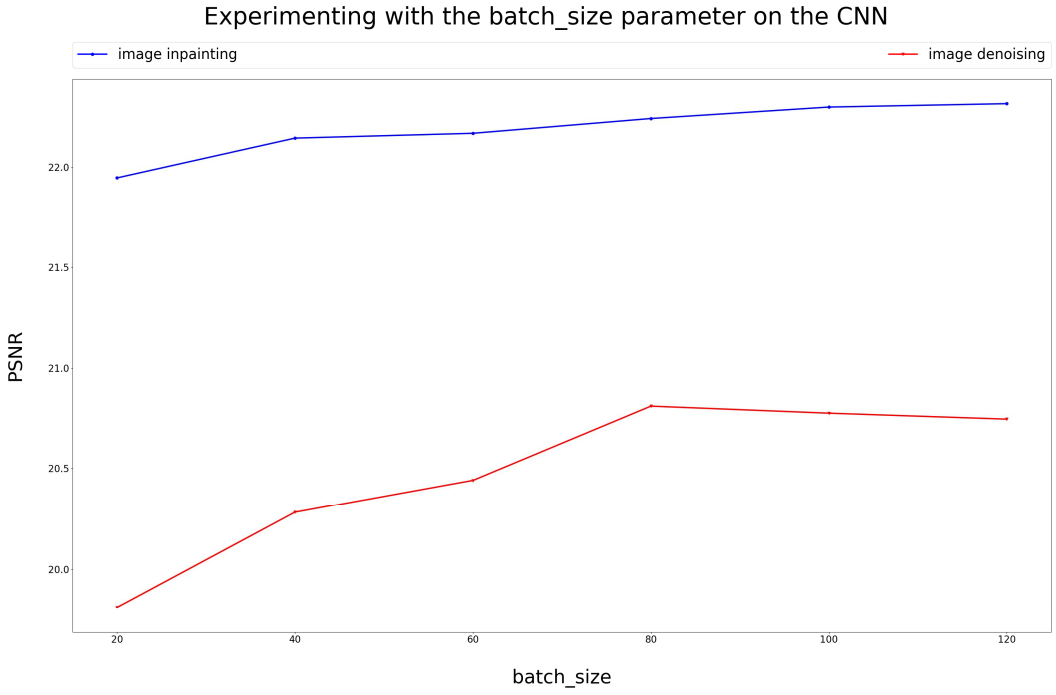
Πίνακας 6: Μετρήσεις για διάφορες τιμές dropout_rate

Task – Dropout_rate	**PSNR**
Image Inpainting(5x5) – 0.2	22.4213868219
Image Inpainting(5x5) – 0.3	22.3783729208
Image Inpainting(5x5) – 0.4	22.296930919
Image Inpainting(5x5) – 0.5	22.0737570023
Image Inpainting(5x5) – 0.6	21.9076688838
Image Inpainting(5x5) – 0.7	21.6867925406
Image Inpainting(5x5) – 0.8	21.2336318228
Image Inpainting(5x5) – 0.9	20.0596825281
Image Inpainting(5x5) – 0.95	19.1264353957
Image Denoising S&P - 0.2	20.9485768944
Image Denoising S&P - 0.3	20.8883746542
Image Denoising S&P - 0.4	20.7730720941
Image Denoising S&P - 0.5	20.5984768509
Image Denoising S&P - 0.6	20.2987485786
Image Denoising S&P - 0.7	20.0127384752
Image Denoising S&P - 0.8	19.7675532455
Image Denoising S&P - 0.9	18.7986765784
Image Denoising S&P - 0.95	17.989768798

Πίνακας 7: Μετρήσεις για διάφορες τιμές filters

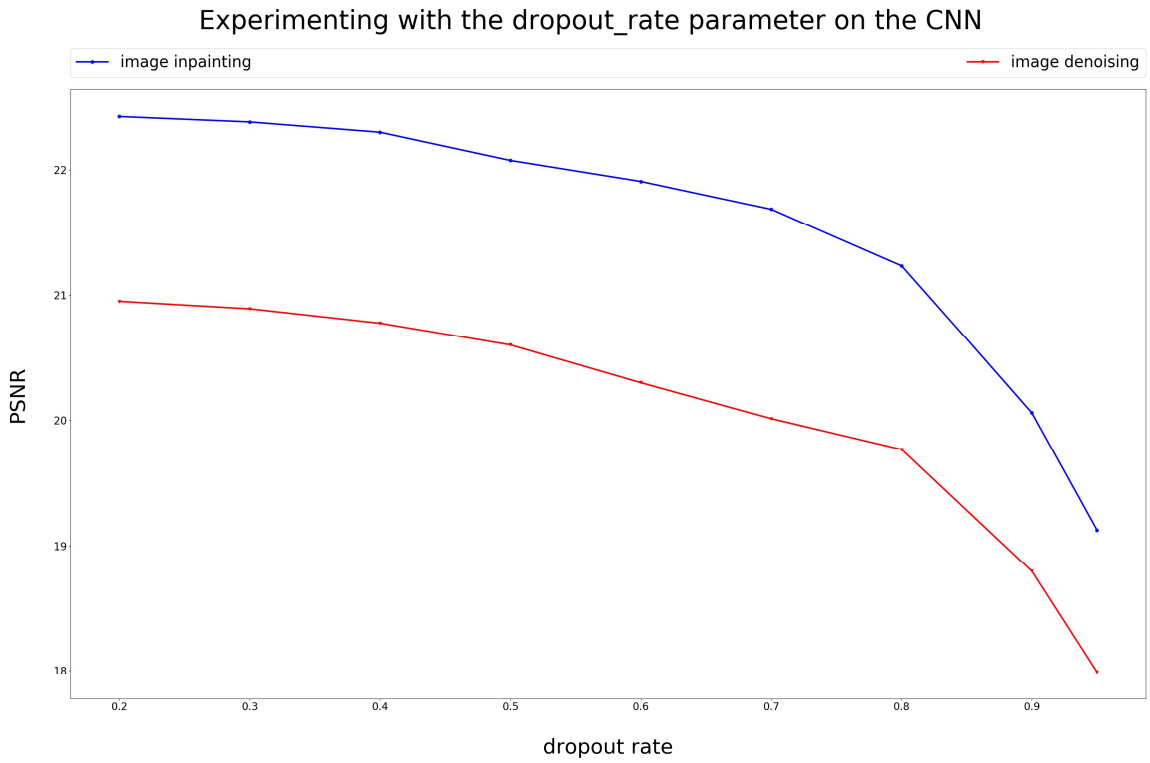
Task – Filters	**PSNR**
Image Inpainting(5x5) – 4	16.9943197251
Image Inpainting(5x5) – 8	18.0840204465
Image Inpainting(5x5) – 16	19.1057482175
Image Inpainting(5x5) – 32	19.9571173236
Image Inpainting(5x5) – 64	20.9224572774
Image Inpainting(5x5) – 128	21.5209033041
Image Inpainting(5x5) – 256	22.111178561
Image Inpainting(5x5) – 512	22.5693517198
Image Denoising S&P - 4	16.9788417497
Image Denoising S&P - 8	18.8299741244
Image Denoising S&P - 16	20.2027577276
Image Denoising S&P - 32	20.414997398
Image Denoising S&P - 64	21.5983688564
Image Denoising S&P - 128	23.7530497953
Image Denoising S&P - 256	24.4722276916
Image Denoising S&P - 512	25.0002270441

Βάσει των παραπάνω τιμών προκύπτουν οι εξής γραφικές παραστάσεις:



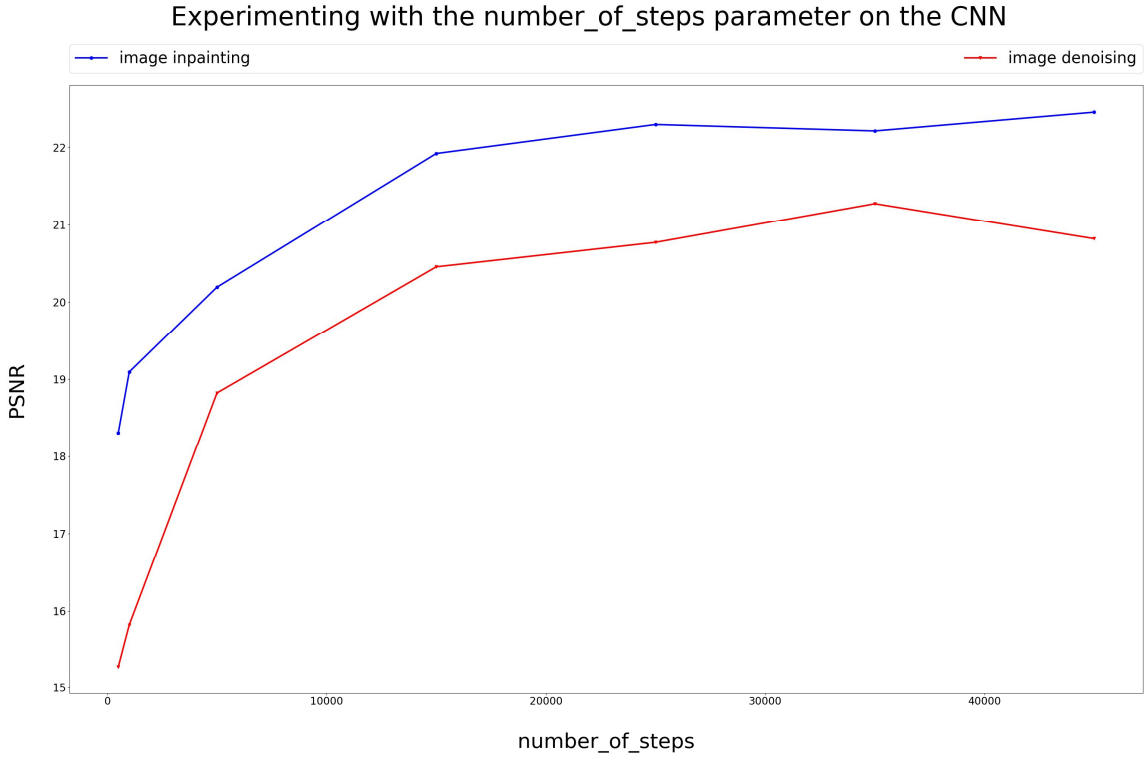
Σχήμα 6: Γραφική παράσταση παραμέτρου batch_size συναρτήσει PSNR

Από την παραπάνω γραφική παράσταση παρατηρούμε πως αύξηση του batch size συνεπάγεται μικρή αλλά σταθερή αύξηση του PSNR και για τα δυο πρόβλημα που επιλύουμε. Συγκεκριμένα, για το πρόβλημα του denoising η αύξηση είναι μεγαλύτερη και πιο αισθητή γεγονός που υποδηλώνει πως αύξηση του btach size βελτιώνει περισσότερο την απόδοση του δικτύου για το πρόβλημα του denoising σε σχέση με το inpainting. Δοκιμές έγιναν και με μεγαλύτερες τιμές batch size όμως οι εκτελέσεις ήταν εξαιρετικά χρονοβόρες λόγω δυσκολίας στην εκπαίδευση του δικτύου, οπότε περιοριστήκαμε στις τιμές που παρουσιάζονται παραπάνω.



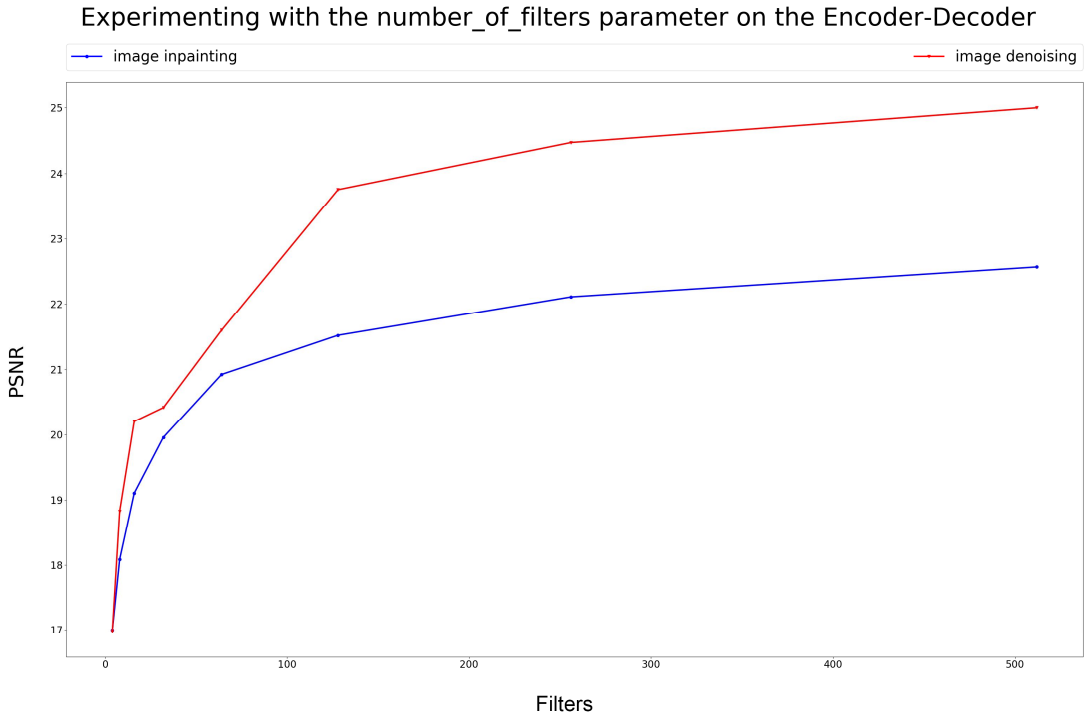
Σχήμα 7: Γραφική παράσταση παραμέτρου dropout_rate συναρτήσει PSNR

Όπως έχουμε αναλύσει και σε προηγούμενο κεφάλαιο, η παράμετρος dropout καθορίζει το πόσοι νευρώνες παραμένουν ενεργοί κατά τη διάρκεια της εκπαίδευσης. Για παράδειγμα, όταν το dropout_rate είναι ίσο με 0.4 μόνο το 60% των νευρώνων του δικτύου είναι ενεργοί και συνεπώς τροποποιούνται τα αντίστοιχα βάρη.
Από τη γραφική παράσταση παρατηρούμε πως όσο αυξάνεται η τιμή του dropout το PSNR ελαττώνεται, γεγονός που δικαιολογείται αφού το δίκτυο μικραίνει και γίνεται λιγότερο ακριβές. Μάλιστα για τιμές μεγαλύτερες του 0.7 η συμπεριφορά αυτή είναι περισσότερο έντονη αφού το δίκτυο συρρικνώνεται και δεν μπορεί να παράγει ικανοποιητικά αποτελέσματα.
Η τιμή που έχουμε επιλέξει ως βέλτιστη είναι **0.4** καθώς έδινε τον καλύτερο συνδυασμό υψηλού PSNR και σχετικά μικρού χρόνου εκπαίδευσης.



Σχήμα 8: Γραφική παράσταση παραμέτρου number_of_steps συναρτήσει PSNR

Πειραματιστήκαμε επίσης με την παράμετρο number_of_steps, η οποία αντιπροσωπεύει την τιμή των βημάτων που χρειάζεται ο αλγόριθμος gradient descent ώστε να βρει τις κατάλληλες παραμέτρους (weights και biases) για το δίκτυό μας. Όπως είναι και διαισθητικά αντιληπτό, όσες περισσότερες επαναλήψεις πραγματοποιήσει ο αλγόριθμος είναι πιθανότερο να βρει βέλτιστες παραμέτρους, εκείνες δηλαδή που ελαχιστοποιούν τη συνάρτηση κόστους του δικτύου μας, προσεγγίζοντας κάποιο τοπικό της ελάχιστο. Έτσι λοιπόν και από τις γραφικές παραστάσεις αριθμού βημάτων σε συνάρτηση με το PSNR και για τα δυο προβλήματα παρατηρούμε πως αρχικά για μικρές τιμές αριθμού βημάτων οι τιμές PSNR είναι χαμηλές ενώ όσο αυξάνουμε τις τιμές η ποιότητα των ανακατασκευασμένων εικόνων βελτιώνεται. Αρχικά η βελτίωση αυτή είναι μεγάλη ενώ στη συνέχεια πιο ομαλή και για τα δυο προβλήματα.
Παρατηρούμε ακόμη πως μετά από τις 25000 επαναλήψεις το PSNR αυξάνεται πολύ αργά ενώ ο χρόνος εκτέλεσης αυξάνεται πολύ ταχύτερα. Για το λόγο αυτό επιλέξαμε για το δίκτυό μας η τιμή number_of_steps να είναι ίση με 25000. Επίσης, βάσει της θεωρίας πολύ μεγάλη αύξηση του number_of_steps ενδέχεται να οδηγήσει όχι μόνο σε προσέγγιση ενός τοπικού ελαχίστου αλλά και σε "προσπέρασή" του καθώς ο αλγόριθμος εκτελεί περιττές επαναλήψεις. Έτσι το δίκτυο δεν παραμετροποιείται με τον καλύτερο δυνατό τρόπο και για τον λόγο αυτό δεν δοκιμάσαμε πολύ μεγάλες τιμές.



Σχήμα 9 Γραφική παράσταση παραμέτρου filters συναρτήσει PSNR

Αναφορικά με τη γραφική παράσταση που απεικονίζει την εξέλιξη του PSNR όσο αυξάνουμε τον αριθμό φίλτρων παρατηρείται συνεχής άνοδος η οποία μάλιστα είναι ιδιαίτερα σημαντική μέχρι και την τιμή των 150 φίλτρων. Πιο συγκεκριμένα μέχρι και την τιμή 150 η αύξηση είναι αξιοσημείωτη υποδηλώνοντας πως μικρές τιμές φίλτρων δεν αποτελούν την πιο ενδεδειγμένη επιλογή και πως η δεδομένη αρχιτεκτονική ανταποκρίνεται καλύτερα όσο αυξάνουμε τον αριθμό τους παράγοντας καλύτερα αποτελέσματα. Μετά την τιμή 150 οι τιμές του PSNR εξακολουθούν να αυξάνονται όμως πιο ομαλά. Δοκιμάστηκαν και μεγαλύτερες τιμές φίλτρων όπως 1024 οι οποίες όμως αύξησαν σημαντικά το μέγεθος του δικτύου μας σε επίπεδα στα οποία η μνήμη που είχαμε διαθέσιμη στην κάρτα γραφικών μας δεν επαρκούσε. Η παραπάνω συμπεριφορά είναι αναμενόμενη καθώς για μεγάλες τιμές φίλτρων απαιτείται η δέσμευση tensor μεγάλων διαστάσεων για την οποία η μνήμη δεν επαρκούσε. Οι παραπάνω παρατηρήσεις αφορούν και τα δύο προβλήματα (denoising και inpainting). Γενικότερα δεδομένου πως ο συγκεκριμένος τύπος δικτύου δεν έχει πλήρως διασυνδεδεμένο επίπεδο, εξοικονομεί χώρο δημιουργώντας σχετικά μικρά tensor άρα και μας δίνει τη δυνατότητα να εισάγουμε μεγάλο αριθμό φίλτρων ώστε να εξάγουμε πιο λεπτομερή χαρακτηριστικά των εικόνων. Συνοπτικά λοιπόν μπορούμε να παρατηρήσουμε πως αύξηση των φίλτρων μπορεί να επιφέρει καλύτερα αποτελέσματα για την αρχιτεκτονική αυτή σε ένα αρχικό εύρος τιμών φίλτρων , ενώ για μεγαλύτερες τιμές η αύξηση είναι πιο ομαλή και σχετικά μικρή σε σχέση με το χρόνο εκπαίδευσης και το χώρο που απαιτείται, κρίνοντας την περαιτέρω αύξηση των φίλτρων μάλλον ασύμφορη.

2.4.6 Σχολιασμός αποτελεσμάτων

Από τα παραπάνω αποτελέσματα προκύπτουν τα εξής συμπεράσματα για την αποδοτικότητα του κάθε δικτύου σε κάθε πρόβλημα που επιλύσαμε.
Αρχικά, παρατηρούμε πως για το Image Inpainting 5x5 το Encoder-Decoder δίκτυο παρουσιάζει λίγο καλύτερη συμπεριφορά ενώ για το Image Inpainting 7x7 το CNN υπερέχει ελαφρώς.
Σχετικά με το Image Denoising, παρατηρήθηκε μία αναμενόμενη υπεροχή του Encoder-Decoder, καθώς θεωρείται αρκετά καλό δίκτυο για το συγκεκριμένο πρόβλημα. Ο ισχυρισμός αυτός δικαιολογείται και από τα αποτελέσματά μας, καθώς το Encoder-Decoder δίκτυο παρουσίασε αυξημένο κατά 3.7 PSNR σε σχέση με το CNN δίκτυο για το Image Denoising με Salt and Pepper θόρυβο και έδωσε πολύ καλά αποτελέσματα με τη χρήση Gaussian θορύβυ, για τον οποίο το CNN, όπως σημειώσαμε και παραπάνω, δεν μπορούσε να εξάγει ικανοποιητικά αποτελέσματα. Μάλιστα, ακόμα και στις υψηλές τιμές Gaussian θορύβου που δοκιμάσαμε το Encoder-Decoder δίκτυο παράγει εικόνες καλής ποιότητας, δηλαδή υψηλού PSNR.
Συμπερασματικά, μπορούμε να πούμε με βεβαιότητα ότι στην εργασία μας επιλύουμε πιο αποτελεσματικά το πρόβλημα του Image Denoising σε σχέση με το Inpainting.

Ως προς τις τιμές των παραμέτρων που δοκιμάσαμε, ο πειραματισμός και με τις τέσσερις παραμέτρους οδήγησε σε βελτίωση των αποτελεσμάτων και εξαγωγή χρήσιμων συμπερασμάτων, με την αύξηση όμως των φίλτρων να επιφέρει τα πιο θεαματικά.

Αναφορικά με τις μετρικές αξιολόγησης των αποτελεσμάτων μας, χρησιμοποιούμε δύο διαφορετικής προσέγγισης μετρικές. Αυτό φαίνεται άλλωστε και από τις τιμές τους, καθώς οι τιμές του PSNR κυμαίνονται από 17 /100 έως 28 /100 ενώ αυτές του SSIM από 0.91 / 1 έως 0.97 / 1. Η διαφορά αυτή των δύο μετρικών έγκειται στο γεγονός ότι η PSNR υπολογίζει το απόλυτο σφάλμα μεταξύ δύο εικόνων ενώ η SSIM συγκρίνει τις δύο εικόνες πιο συνολικά, εξετάζοντας εκτός από την απόλυτη διαφορά των pixel και τις συσχετίσεις που υπάρχουν μεταξύ τους, δίνοντας έτσι τη δυνατότητα να μπορεί να εντοπίσει ομοιότητες στο σχήμα και στο περίγραμμα των εικόνων.
Είναι γνωστό επίσης πως η SSIM θα θεωρήσει μία εικόνα που έχει αλλοιωμένο κάποιο δομικό συστατικό (για παράδειγμα η γωνία ενός τετραγώνου μοιάζει με καμπύλη) χαμηλότερης ποιότητας ενώ η ίδια εικόνα θα αξιολογηθεί με περισσότερη επιείκεια από την PSNR αφού υπολογίζει απόλυτο σφάλμα μεταξύ των pixel χωρίς να μεριμνά για την διατήρηση της δομικής πληροφορίας.

Τέλος, ένα αξιοσημείωτο πόρισμα είναι πως ένα καλά σχεδιασμένο δίκτυο μπορεί να ανταποκριθεί στην επίλυση διαφορετικών προβλημάτων αλλά και σε διαφορετικές εκδοχές του ίδιου προβλήματος. Για παράδειγμα, με το CNN δίκτυο επιλύουμε το Image Inpainting και το Image Denoising, αλλάζοντας μόνο τα δεδομένα εκπαίδευσης και κρατώντας το δίκτυο σταθερό.

3 ΣΥΜΠΕΡΑΣΜΑΤΑ

Φθάνοντας στο τελευταίο κεφάλαιο της παρούσας πτυχιακής εργασίας, μπορούμε να καταλήξουμε με βεβαιότητα στο συμπέρασμα πως επιτεύχθηκαν οι αρχικοί στόχοι που είχαν τεθεί.

Πιο συγκεκριμένα, θεωρούμε πως το πρώτο μέρος της εργασίας, δηλαδή η θεωρητική προσέγγιση των νευρωνικών δικτύων και τη χρήση τους στην επεξεργασία εικόνας, αποτυπώνει όλη την απαραίτητη γνώση που απαιτείται από κάποιον που επιθυμεί να γνωρίσει τον τομέα αυτό.

Επιπρόσθετα, μέσω της εργασίας μας καταφέραμε να αναδείξουμε τις δυνατότητες των νευρωνικών δικτύων και συγκεκριμένα του Deep Learning στην ανακατασκευή εικόνων, πειραματιζόμενοι με διαφορετικές αρχιτεκτονικές δικτύων και παραμέτρους.

Τέλος, τα δίκτυα που αναπτύξαμε είναι ικανά να παράγουν εικόνες υψηλής ποιότητας, κυρίως με τη χρήση εικόνων μικρών σχετικά διαστάσεων. Χρησιμοποιώντας προηγμένα υπολογιστικά συστήματα που διαθέτουν μία ή περισσότερες κάρτες γραφικών με μεγάλη χωρητικότητα μνήμης τα δίκτυα που αναπτύξαμε, με τροποποίηση των κατάλληλων παραμέτρων τους, είναι εξίσου αποδοτικά στην επεξεργασία μεγαλύτερων εικόνων. Παρόλα αυτά, δεδομένου ότι ο τομέας των νευρωνικών δικτύων εξελίσσεται ταχύτατα, αναδύονται συνεχώς νέες αρχιτεκτονικές και μέθοδοι που επιλύουν τα ίδια προβλήματα πιο αποδοτικά.

ΠΙΝΑΚΑΣ ΟΡΟΛΟΓΙΑΣ

Ξενόγλωσσος όρος	**Ελληνικός Όρος**
Convolutional neural network	Συνελικτικά νευρωνικά δίκτυα
Encoder-Decoder	Κωδικοποιητής-Αποκωδικοποιητής
Image inpainting	Συμπλήρωση εικόνας
Image denoising	Αφαίρεση θορύβου εικόνας
Dataset	Σύνολο δεδομένων
Neural networks	Νευρωνικά δίκτυα
Input	Είσοδος
Output	Έξοδος
Hidden	Κρυφό
Cluster	Συστοιχία
Artificial neural networks	Τεχνητά νευρωνικά δίκτυα
Unit	Μονάδα
Layer	Επίπεδο
Weight	Βάρος
Bias	Δυναμικό πολώσεως
Activation function	Συνάρτηση ενεργοποίησης
Signal	Σήμα
Vanishing gradients	Εξάλειψη κλίσης
Backpropagation	Οπισθοδρόμιση
Not zero centered	Μη κεντραρισμένο στο μηδέν
Universal approximation theorem	Θεώρημα καθολικής προσέγγισης
Underfitting	Υποεκπαίδευση
Overfitting	Υπερεκπαίδευση
Overtraining	Υπερεκπαίδευση
Training set	Σύνολο εκπαίδευσης
Test set	Σύνολο δοκιμής
Pixel	Εικονοστοιχίο
Dropout	Απόρριψη
Forward Propagation	Προς τα εμπρός διάδοση
Loss function	Συνάρτηση κόστους
Mean Squared Error	Μέσο τετραγωνικό σφάλμα
Learning rate	Ρυθμός εκμάθησης
Batch mode	Επεξεργασία κατά συρροή
Pattern/Online mode	Επεξεργασία κατά μόνας
Step	Βήμα
Validation set	Σύνολο επιβεβαίωσης
Hyperparameter	Υπερπαράμετρος
Image	Εικόνα
Stride	Βήμα μετακίνησης παραθύρου
Recognition	Αναγνώριση
Recommend	Πρόταση
Natural language processing	Επεξεργασία φυσικής γλώσσας
Receptive field	Γνωστικό πεδίο
Subsampling	Υποβάθμιση διαστάσεων
Pooling	Συγκέντρωση
Filter	φίλτρο
Shift invariant or space invariant	ανεξάρτητα του μεγέθους αλλά και του

	προσανατολισμού του αντικειμένου
Parameter	Παράμετρος
Padding	Ταίριασμα
Average	Μέσος όρος
Max	Μέγιστο
Transposed convolution	Αντίστροφη συνέλιξη
Sparse matrix	Αραιός πίνακας
Latent space	Χαμηλής διάστασης διανυσματική αναπαράσταση
Analytical approache	Αναλυτική προσέγγιση
Super resolution	Υψηλή ανάλυση
Residual block	Υπολειπόμενο τμήμα
Skip connections	Συνδέσεις παράκαμψης
Unsupervised learning	Μη επιβλεπόμενη εκμάθηση
Evolutional strategy	Επαναστατική στρατηγική
Bottleneck	Μεσαίο τμήμα Encoder-Decoder
Dictionary	Λεξικό
Image fill-in	Γέμισμα εικόνας
Image completion	Συμπλήρωση εικόνας
Gray scale	Γκρι κλίμακα
Open source	Κώδικας ελεύθερης πρόσβασης
Dataflow graph	Γράφος ροής δεδομένων
Session	Περίοδος
Tensor	Είδος διανύσματος
Dense	Πυκνά συνδεδεμένο
Channels	Κανάλια
Width	Πλάτος
Height	Ύψος
Batch size	Μέγεθος ριπής
Reshape	Τροποποίησης διάστασης
Estimator	Εκτιμητής
structural information	Δομική πληροφορία
Covariance	Συνδιακύμνση
Loss of correlation	Απώλεια συσχέτισης
Luminance distortion	Παραμόρφωση φωτεινότητας
Constrast distortion	Διαστρέβλωση της αντίθεσης

ΣΥΝΤΜΗΣΕΙΣ – ΑΡΚΤΙΚΟΛΕΞΑ – ΑΚΡΩΝΥΜΙΑ

CNN	Convolutional Neural Network
GPU	Graphical Processing Unit
ANN	Artificial Neural Network
XOR	Text Encoding Initiative
RGB	Universal System for information in Science and technology
DL	Deep Learning
ReLU	Rectified Linear Unit
GANs	Generative Adversarial Networks
PDF	Probability Density Function
API	Application Programming Interface
PSNR	Peak Signal to Noise Ratio
DB	DeciBel
MSE	Mean Squared Error
SSIM	Structural Similarity Index for Measuring

ΠΑΡΑΡΤΗΜΑ

Στο παράρτημα αυτό παρουσιάζονται ορισμένα θέματα που αφορούν τη λειτουργικότητα των προγραμμάτων καθώς και σχολιασμό του πηγαίου κώδικα. Αρχικά, τα αρχεία που περιέχουν τον πηγαίο κώδικα βρίσκονται στον φάκελο "code" και είναι τα εξής. Τα αρχεία calculate_metrics, image_process και plotter είναι βοηθητικά και περιέχουν συναρτήσεις που εξυπηρετούν τον υπολογισμό των μετρικών, την επεξεργασία των εικόνων και την κατασκευή των γραφικών παραστάσεων αντίστοιχα. Στη συνέχεια, υπάρχουν οι φάκελοι inpainting και denoising που περιέχουν το καθένα τους φακέλους cnn και encoder-decoder. Στους τελευταίους βρίσκονται τα αρχεία με τα αντίστοιχα μοντέλα καθώς και αρχείο utils που περιέχει χρήσιμες συναρτήσεις.

Το κάθε αρχείο model περιέχει τον κώδικα που επιλύει το αντίστοιχο πρόβλημα και έχει την εξής λειτουργικότητα. Στην αρχή του αρχείου ορίζονται οι απαραίτητες πληροφορίες που σχετίζονται με το μέγεθος των εικόνων προς επεξεργασία. Για το πρόβλημα image inpainting υπάρχει η σταθερά WINDOW_SIZE που λαμβάνει την επιθυμητή τιμή για το μέγεθος του παραθύρου. Έπειτα, ακολουθεί ο ορισμός των υπερμαραμέτρων των μοντέλων. Τέλος, για την ομαλή λειτουργία του προγράμματος απαιτείται σωστός ορισμός των φακέλων από τους οποίου γίνεται η ανάγνωση του συνόλου δεδομένων. Πρέπει δηλαδή να γίνει σωστή ανάθεση στις σταθερές DIR_TRAIN_LABELS, DIR_TRAIN_INPUTS, DIR_TEST_LABELS, DIR_TEST_INPUTS και DIR_TO_WRITE_RESULTS.

Για τη σωστή επίλυση του προβλήματος image inpainting χρειάζεται να αποσχολιαστεί/σχολιαστεί η γραμμή 186/187 ανάλογα με τον τύπο του παραθύρου που χρησιμοποιείται. Για τετραγωνικό παράθυρο στο κέντρο της εικόνας χρειάζεται η συνάρτηση utils.fill_dark_part ενώ για το παράθυρο στο αριστερό μισό τμήμα η utils.fill_half_dark_part.

Για την κατασκευή του συνόλου εκπαίδευσης και δοκιμής χρειάζεται το αρχείο image_process. Το πρόγραμμα δέχεται ως όρισμα από την γραμμή εντολής τον τύπο της παραμόρφωσης που θα υποστούν οι εικόνες (addBlack, addHalfBlack ή addNoise) καθώς και ένα δεύτερο όρισμα με τον τύπο του θορύβου (gaussian ή sap) ή το μέγεθος του παραθύρου που θα εφαρμοστεί.

ΑΝΑΦΟΡΕΣ

[1] S. Zhang and E. Salari, “Image denoising using a neural network based non-linear filter in wavelet domain,” in Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing, Ma r. 2005, pp. 989–992.
[2] V. Papyan, Y. Romano, and M. Elad, “Convolutional neural networks analyzed via convolutional sparse coding,” arXiv preprint, arXiv:1607.08194, 2016.
[3] X. J. Mao, C. Shen, and Y. Ya n g , “Image denoising using very deep fully convolu- tional encoder-decoder networks with symmetric skip connections,” arXiv:abs/1606.08921, 2016.
[4] Masanori Suganuma, Mete Ozay, Takayuki Okatani Exploiting the Potential of Standard Convolutional Autoencoders for Image Restoration by Evolutionary Search, 2018.
[5] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, Victor Lempitsky; “Domain-Adversarial Training of Neural Networks”, 17(59):1−35, 2016.
[6] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio, Generative Adversarial Networks, 2014.
[7] Βιβλίο «Τεχνητή Νοημοσύνη – Μια σύγχρονη προσέγγιση», Russell Norvig.
[8] Βιβλίο «Αναγνώριση Προτύπων», Σ. Θεοδωρίδης, Κ. Κουτρούμπας
[9] http://www.heatonresearch.com/2017/06/01/hidden-layers.html
[10] https://en.wikipedia.org/wiki/Universal_approximation_theorem
[11] https://www.cs.toronto.edu/~hinton/absps/JMLRdropout.pdf
[12] https://www.learnopencv.com/understanding-activation-functions-in-deep-learning
[13] https://en.wikipedia.org/wiki/Sigmoid_function
[14] https://en.wikipedia.org/wiki/Hyperbolic_function
[15] https://isaacchanghau.github.io/2017/06/07/Loss-Functions-in-Artificial-Neural-Networks
[16] https://www.analyticsvidhya.com/blog/2017/05/25-must-know-terms-concepts-for-beginners-in-deep-learning
[17] https://www.kdnuggets.com/2016/12/artificial-neural-networks-intro-part-1.html
[18] http://www.explainthatstuff.com/introduction-to-neural-networks.html
[19] http://www.bogotobogo.com/python/scikit-learn/Artificial-Neural-Network-ANN-2-Forward-Propagation.php
[20] https://machinelearningmastery.com/difference-test-validation-datasets
[21] https://machinelearningmastery.com/gentle-introduction-mini-batch-gradient-descent-configure-batch-size
[22] https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks
[23] https://cambridgespark.com/content/tutorials/convolutional-neural-networks-with-keras/index.html
[24] http://ufldl.stanford.edu/tutorial/supervised/ConvolutionalNeuralNetwork
[25] http://www.wseas.us/e-library/conferences/2010/Faro/VIS/VIS-08.pdf
[26] https://en.wikipedia.org/wiki/Structural_similarity
[27] https://en.wikipedia.org/wiki/Peak_signal-to-noise_ratio
[28] https://arxiv.org/pdf/1603.07285.pdf
[29] http://yann.lecun.com/exdb/mnist/