ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Μελέτη και πειραματική εκτίμηση των διαφόρων προσεγγίσεων Βαθιάς Μάθησης και σουίτες λογισμικού

Απόστολος – Γεώργιος - Γιούλης

Επιβλέπουσα: **Αθανασία Αλωνιστιώτη,** Επίκουρος Καθηγητής

ΑΘΗΝΑ

ΙΟΥΛΙΟΣ 2015

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Μελέτη και Πειραματική εκτίμηση των διαφόρων προσεγγίσεων Βαθιάς Μάθησης και σουίτες Λογισμικού

Απόστολος Γ. Γιουλής

Α.Μ.: 1115200900215

ΕΠΙΒΛΕΠΟΝΤΕΣ: **Αθανασία Αλωνιστιώτη,** Επίκουρος Καθηγητής

ΠΕΡΙΛΗΨΗ

Η παρούσα εργασία έχει ως αντικείμενο την μελέτη του ανερχόμενου πεδίου μηχανικής μάθησης γνωστό και ως Βαθιά Μάθηση (Deep Learning DL). Η εργασία έχει επικεντρωθεί στην ανάλυση και μελέτη των παρακάτω αλληλένδετων υπο-κεφαλαίων του πεδίου:

- Στην ανάλυση μεθόδων μάθησης που χρησιμοποιούνται για την εκπαίδευση μοντέλων Deep Learning, δηλαδή της επιβλεπόμενης, μη-επιβλεπόμενης και υβριδικής μάθησης, (με έμφαση στην επιβλεπόμενη μάθηση). Ιδιαίτερη έμφαση έχει δοθεί και στην παρουσίαση του αλγορίθμου της Οπισθοδρομικής Διάδοσης Σφάλματος, ο οποίος αποτελεί ακρογωνιαία λίθο των περισσότερων εκ των υπάρχοντών τεχνικών μάθησης. Έμφαση έχει δοθεί επίσης και στις δυσκολίες που παρουσιάζονται κατά την εκπαίδευση μοντέλων DL καθώς και σε διάφορες τεχνικές αντιμετώπισης τους.
- Στην παρουσίαση και ανάλυση πολλαπλών μοντέλων (νευρωνικών δικτύων) που χρησιμοποιούνται στα πλαίσια της βαθιάς μάθησης.
- Στην παρουσίαση των βασικών χαρακτηριστικών, των πιο αξιόλογων και ολοκληρωμένων λύσεων λογισμικού που έχει να προσφέρει η βιομηχανία.

Τέλος στο πρακτικό τμήμα της πτυχιακής έχουν κατασκευαστεί μοντέλα Βαθιάς Μάθησης τόσο με χρήση της γλώσσας προγραμματισμού Python όσο και με χρήση μερικών εκ των παρουσιασμένων λύσεων λογισμικού.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Μηχανική Μάθηση

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: Βαθιά Μάθηση, Βαθιά Νευρωνικά Δίκτυα

ABSTRACT

The following paper's main goal is to study the up-and-coming, machine learning field of Deep Learning. The main focus was to analyze and study the following interrelated chapters of the field:

- Analysis of the learning methods, supervised, unsupervised and hybrid learning, used while training deep learning models. Emphasis was laid on presenting the Back-Propagation algorithm, the cornerstone of most of the available training methods. Emphasis was also placed on presenting the difficulties that may occur while training a deep learning model, as well as some of the available solutions.
- Presentation and analysis of many of the available models (neural networks) used in modern deep learning applications
- Presentation of the main features of the available deep learning software suites.

Lastly, for the practical section of the study, there has been a construction of deep learning models, with the use of python as the chosen programming language, as well as with the use of some of the software solutions presented in the theoretical part.

SUBJECT AREA: Machine Learning

KEYWORDS: Deep Learning, Deep Neural Networks

ΕΥΧΑΡΙΣΤΙΕΣ

Για τη διεκπεραίωση της παρούσας Πτυχιακής Εργασίας, θα ήθελα να ευχαριστήσω την επιβλέπουσα, επ. καθ . Αλωνιστιώτη Αθανασία καθώς και τον κ. Μανώλη Κιαγια για τη συνεργασία και την πολύτιμη συμβολή τους στην ολοκλήρωση της.

ΠΕΡΙΕΧΟΜΕΝΑ

Κατάλογος Εικόνων

Κατάλογος Διαγραμμάτων

Κατάλογος Πινάκων

ΠΡΟΛΟΓΟΣ

"Everything we do, every thought we've ever had, is produced by the human brain. But exactly how it operates remains one of the biggest unsolved mysteries, and it seems the more we probe its secrets, the more surprises we find"

Neil deGrasse Tyson

Η μηχανική μάθηση αποτελεί ένα από τα πιο συναρπαστικά πεδία της σύγχρονης πληροφορικής το οποίο έχει αλλάξει εξολοκλήρου τον τρόπο με τον οποίο λειτουργούν και "σκέφτονται" οι μηχανές. Μια μηχανή μπορεί πλέον να δράσει χωρίς να έχει προγραμματιστεί ρητά να δράσει αναλόγως. Ρομποτική, αεροδιαστημική, στρατιωτικές εφαρμογές, μεγάλα δεδομένα, βιοιατρική, τεχνίτη νοημοσύνη, δεν είναι πάρα λίγα από τα πεδία στα οποία εφαρμόζονται επιτυχώς τεχνικές μηχανικής μάθησης. Παρά όμως την τεράστια επιτυχία που έχουν γνωρίσει οι κλασσικές, πλέον, θεωρίες και μέθοδοι τις μηχανικής μάθησης κάποια προβλήματα παραμένουν χρονοβόρα και δύσκολα. Το χαρακτηριστικότερο εξ αυτών είναι η εξαγωγή χαρακτηριστικών από δεδομένα τα οποία να κατοπτρίζουν τίς δομές και τις ιδιαιτερότητες δεδομένων αυτών.

Μέχρι πρόσφατα, μια ήταν η "μηχανή" που είχε καταφέρει να επιλύσει αποτελεσματικά την οικογένεια των δύσκολων αυτών προβλημάτων . Η "μηχανή" αυτή δεν ήταν άλλη, από τον ανθρώπινο εγκέφαλο ο οποίος καταφέρνει σε χρόνους της τάξης των 100–200 millisecond να αναγνωρίσει, να μοντελοποιήσει και να επεξεργαστεί τεράστιο όγκο δεδομένων. Ο ανθρώπινος εγκέφαλος και η λειτουργεία του έχουν συναρπάσει και έχουν προβληματίσει την ανθρώπινη σκέψη από αρχαιοτάτων χρονών. Δεν ήταν όμως παρά στα τέλη του 20 αιώνα που παρουσιάστηκαν οι πρώτες ενδείξεις επιτυχούς μοντελοποίησης και κατανόησης της νευρωνικής λειτουργεία του εγκεφάλου. Αυτές οι πρώιμες ανακαλύψεις δώσαν το έναυσμα για την δημιουργία των τεχνιτών νευρωνικών δικτιών τα οποία προσομοιάζουν και μιμούνται, ως ένα βαθμό, την λειτουργεία του ανθρώπινου εγκεφάλου. Στην συνέχεια, και μετά από χρόνια εκτενούς μελέτης και ανάπτυξης των νευρωνικών δικτύων και των μεθόδων εκπαίδευσης τους οι επιστήμονες οδηγήθηκαν στην δημιουργία του πεδίου της βαθιάς μάθησης.

Η βαθιά μάθηση είναι ένα νέο συναρπαστικό υπό-πεδίο της μηχανικής μάθησης το οποίο επιδιώκει να μάθει περίπλοκες προβλέψεις απευθείας από τα δεδομένα. Ορισμός :"Βαθιά μάθηση είναι μια κατηγορία τεχνικών μηχανικής μάθησης η οποίες εκμεταλλεύονται πολλαπλά στρώματα μη γραμμικής επεξεργασίας πληροφορίας με σκοπό την επιβλεπόμενη ή μη-επιβλεπόμενη εξαγωγή χαρακτηριστικών και μετασχηματισμών καθώς και την ανάλυση προτύπων ή και την κατηγοριοποίηση."

Οι περισσότερες προσεγγίσεις της βαθιάς μάθησης, οι οποίες και μελετώνται στην παρούσα εργασία, βασίζοντα σε νευρωνικά δίκτυα, δηλαδή δίκτυα διασυνδεδεμένων τεχνιτών νευρώνων, εμπνευσμένα από την δια-νευρωνική λειτουργεία του εγκεφάλου. Παρά το ότι ο όρος νευρωνικά δίκτυα έχει καταλήξει να ταυτίζεται με αυτόν της βαθιάς μάθησης, η ταύτιση αυτή δεν είναι πάντα ακριβής καθώς μόνο στα πολύ-στρωματικά δίκτυα (πάνω από 3 στρώματα) άπτει ο χαρακτηρισμός «βαθιά».

Οι νέες αυτές τεχνικές έχουν χρησιμοποιηθεί με μεγάλη επιτυχία για την επίλυση πολλών, μέχρι πρότινος, δύσκολών προβλημάτων όπως είναι η αναγνώριση αντικειμένων, η όραση υπολογιστή (computer vision), η αναγνώριση κειμένου, η μοντελοποίηση γλώσσας, η επεξεργασία φυσικής γλώσσας, η ανάκτηση πληροφορίας κ.λπ. Τέτοια είναι η βαρύτητα του νέου αυτού πεδίου που το 2013 αναγνωρίστηκε ως μια εκ των 10 επαναστατικότερων τεχνολογιών της χρονιάς αυτής. Πολλοί ακαδημαϊκοί οργανισμοί με πρωταγωνιστή, ίσως, το πανεπιστήμιο του Toronto έχουν δημιουργήσει εξειδικευμένες ερευνητικές ομάδες αφιερωμένες αποκλειστικά σε αυτόν τον τομέα. Όπως ήταν αναμενόμενο άλλωστε, το ευρύ αυτό φάσμα των επιτυχιών τις βαθιάς μάθησης δεν θα μπορούσε να περάσει απαρατήρητό και από στην τεχνολογική βιομηχανία.

Τα τελευταία χρόνια παρά πολλές επιχειρήσεις έχουν στρέψει το ενδιαφέρων τους στο νέο αυτό τομέα ενσωματώνοντας βαθιά νευρωνικά δίκτυα σε πολλά από τα προϊόντα και υπηρεσίες τους. Τεχνολογικοί κολοσσοί με πρωταγωνιστές τις google, Microsoft, Facebook και Baibu έχουν συμβάλει σημαντικά στην ανάπτυξη της βαθιάς μάθησης έχοντας δημιουργήσει εντυπωσιακά εμπορικά προϊόντα και υπηρεσίες βασισμένες σε αυτήν. Ακολουθεί μια ενδεικτική λίστα από προϊόντα και υπηρεσίες που χρησιμοποιούν βαθιά μάθηση.

- Υπηρεσία φωνητικής αναζήτησης του Bing, η οποία μετά την ενσωμάτωση βαθιών νευρωνικών δικτύων σε αυτήν, γνώρισε βελτίωση της τάξης του 15% στο ρυθμό σφάλματος αναγνώρισης λέξης [101].
- Φωνητική αναγνώριση φυσικής γλώσσας στην κονσόλα Xbox One [1]
- Azure Media Indexer, μια νέα cloud υπηρεσία που επιτρέπει στους χρήστες της να κατατάσσουν βίντεο βάσει των λέξεις που έχουν ειπωθεί σε αυτά. [2]
- Συστατικές μηχανές διαφόρων εταίρων όπως Netflix [3], Spotify [4], Amazon [5] κ.λπ.
- Enlitic: Υπηρεσία που χρησιμοποιεί βαθιά μάθηση για την επεξεργασία ακτινογραφιών και τομογραφιών. Υποβοηθώντας έτσι το ιατρικό προσωπικό κατά την διάγνωση ασθενειών. [6]
- Supercomputer αφιερωμένος στην αγνώριστή εικόνας μέσω βαθιάς μάθησης για την μηχανή αναζήτησης της Baibu. [7]
- Πρόβλεψη των προτιμήσεων καταναλωτών με χρήση βαθιάς μάθησης, έχοντας ως σκοπό την εξατομικευμένη προβολή σχετικών διαφημίσεων από την Baibu. Τεχνολογία η οποία οδήγησε μάλιστα σε σημαντική αύξηση των εσόδων της. [5]
- Ανίχνευση ηλεκτρονικής απάτης της PayPal [8]

Συμπερασματικά, η βαθιά μάθηση είναι ένας γοργά αναπτυσσόμενος και πολλά υποσχόμενος τομέας μηχανικής μάθησης ο οποίος έχει επιφέρει επανάσταση στον τομέα της μηχανικής μάθησης και που ίσως κάποια μέρα να συμβάλει στην δημιουργία μιας ευρείας τεχνίτης νοημοσύνης, παρόμοιας με εκείνη που χαρακτηρίζει τον ανθρώπινο εγκέφαλο.

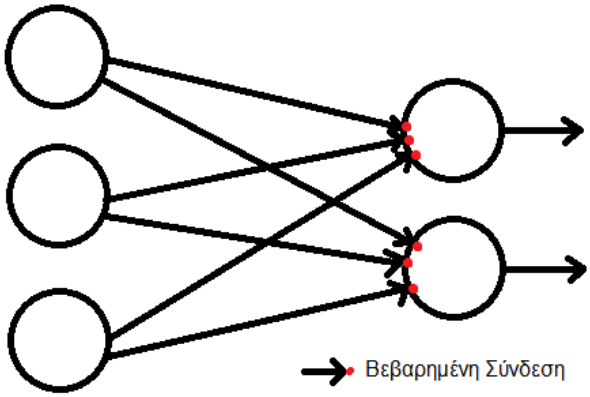
1. Νευρωνικά Δίκτυα

1.1 Εισαγωγή

Η μελέτη των τεχνητών νευρωνικών δικτύων τα οποία συχνά αναφέρονται και ως νευρωνικά δίκτυα, είναι εμπνευσμένη από τον τρόπο με τον οποίο ο ανθρώπινος εγκέφαλος περατώνει υπολογισμούς προκείμενου να επιτύχει αναγνώριση των εισερχόμενων ερεθισμάτων. Τρόπος, ο οποίος είναι άρδην διαφορετικός με αυτόν που υπολογίζει ένας ψηφιακός υπολογιστής καθώς ο εγκέφαλος είναι ένας εξαιρετικά περίπλοκος, μη γραμμικός, παράλληλος υπολογιστής με την δυνατότητα να οργανώνει τα δομικά του στοιχεία (νευρώνες) με τέτοιο τρόπο ώστε να μπορεί να πραγματοποιεί, ταχύτατα, ορισμένους υπολογισμούς, όπως αναγνώριση προτύπων και η αντίληψη και ο έλεγχος κίνησης.

Τα τεχνητά νευρωνικά δίκτυα, καθ' ομοίωση του εγκεφάλου, αποτελούνται από απλές υπολογιστικές μονάδες οι οποίες αλληλοεπιδρούν μέσω βεβαρημένων συνδέσεων. Οι μονάδες αυτές χαρακτηρίζονται ως νευρωνικοειδείς (neuron-like) αν και συχνά αναφέρονται ως νευρώνες (όρος που θα προτιμηθεί στο παρόν κείμενο). Τα νευρωνικά δίκτυα μπορούν να υλοποιηθούν τόσο σε software όσο και σε hardware ωστόσο το μεγαλύτερο μέρος των τρεχόντων έρευνών έχει αφιερωθεί στα νευρωνικά δίκτυα λογισμικού λόγο της εύπλαστης και ευκολά παραμετροποίησης τους φύσης. Παρόλα αυτά hardware νευρωνικά δίκτυα, σε ορισμένες περιπτώσεις, έχουν επιτύχει αποδόσεις εφάμιλλές ή και καλύτερες των αντίστοιχών δικτύων λογισμικού [9].

Ο τρόπος με τον δομείτε ένα νευρωνικό δίκτυο καλείτε αρχιτεκτονική. Η πιο απλή αρχιτεκτονική νευρωνικού δικτύου συνίσταται από μία ομάδα νευρώνων εισόδου η οποία συνδέεται μέσω κατευθυνόμενων, βεβαρημένων συνδέσεων με μια ομάδα νευρώνων εξόδου. Πάντα στα τεχνητά νευρωνικά δίκτυα το στρώμα εισόδου θεωρείται η αρχή ή το κατώτερο στρώμα του δικτύου, αντιθέτως το στρώμα εξόδου θεωρείται ως το τέλος ή η κορυφή ενός δικτύου. Αναλόγως καθορίζεται και η μπροστινή ή ανοδική κατεύθυνση (από είσοδο σε έξοδο).



Εικόνα 1.1: Γράφος Απλού Perceptron

Τα βάρη των νευρώνων, τα γνωστά και ως συναπτικά βάρη, περιγράφουν ουσιαστικά το κατά πόσο συνεισφέρει η εκάστοτε είσοδος στην απόφαση την οποία ένας νευρώνας καλείται να λάβει. Προκειμένου να εκπαιδευτεί λοιπόν ένα δίκτυο αναζητούνται τα βέλτιστα

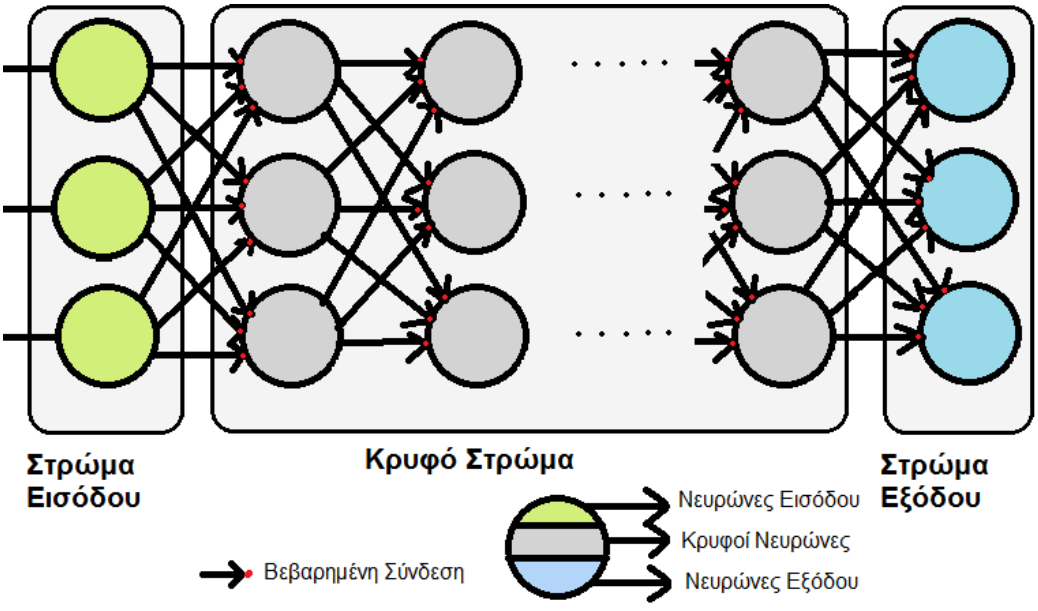
αυτά βάρη για τα οποία το δίκτυο λαμβάνει τις σωστότερες αποφάσεις . Συνήθως, οι τεχνικές εκπαίδευσης υλοποιούν μέθοδούς σφάλματος (gradient methods) με αρκετά μεγάλη επιτυχία.

Τέτοια δυσστρωματικά δίκτυα, τύπου Perceptron, μελετήθηκαν εκτενώς την δεκαετία του 60 καθώς αποτελούν πολύ απλούς μαθησιακούς αλγορίθμους οι οποίο παρέχουν εγγυήσεις ότι τα εντέλει τα βέλτιστα βάρη των δια-νευρωνικών συνδέσεων θα βρεθούν (εφόσον πληρούνται ορισμένες προϋποθέσεις). Δυστυχώς τόσο απλά δίκτυα μπορούν να υπολογίσουν ένα πολύ περιορισμένο εύρος συναρτήσεων [10]. Για παράδειγμα, είναι αδύνατο να υπολογίσουν το XOR δύο διάδικων εισόδων.

Αυτοί οι περιορισμοί μπορούν να ξεπεραστούν, αν εισαχθεί στο δίκτυο ένα τρίτο στρώμα μη γραμμικών νευρώνων μεταξύ των στρωμάτων εισόδου και εισόδου, οι αποκαλούμενοι και ως κρυφοί νευρώνες. Με αρκετούς κρυφούς νευρώνες σε έναν στρώμα, είναι δυνατό να βρεθούν κατάλληλα βάρη τέτοια ώστε να μπορούν να προσεγγίσουν οποιαδήποτε συνεχής και διαφορίσιμη χαρτογράφηση από έναν συμπαγή χώρο εισόδου σε έναν συμπαγή χώρο εξόδου.

Στα νευρωνικά δίκτυα που έχουν αναφερθεί ως τώρα, εμπίπτει ο χαρακτηρισμός «ρηχά» (shallow) καθώς υλοποιούν μέχρι ένα κρυφό στρώμα. Έως αρκετά πρόσφατα, οι περισσότερες τεχνικές μηχανικής μάθησης και επεξεργασίας σήματος εκμεταλλεύονταν ρηχές αρχιτεκτονικές οι οποίες αν και έχει δειχτεί ότι είναι αποτελεσματικές στην επίλυση κάποιον απλών ή περιορισμένων προβλημάτων, έχουν περιορισμένες δυνατότητες μοντελοποίησης και αναπαράστασης κάτι το οποίο μπορεί να δημιουργήσει προβλήματα κατά την αντιμετώπιση περίπλοκων προβλημάτων. Για τον λόγο αυτό οι ερευνητές οδηγήθηκαν στην δημιουργία νέων βαθιών αρχιτεκτονικών.

Οι βαθιές αρχιτεκτονικές αποτελούνται από πολλαπλά κρυφά στρώματα. Τυπικά λοιπόν μια βαθιά αρχιτεκτονική έχει ένα στρώμα εισόδου αποτελούμενα από νευρώνες εισόδου, πολλαπλά κρυφά στρώματα αποτελούμενα από κρυφούς νευρώνες και ένα στρώμα εξόδου αποτελούμενο από νευρώνες εξόδου.



Εικόνα 1.2: Γράφος Βαθιού Νευρωνικού δικτύου

Οι διάφορες αρχιτεκτονικές βαθιών δικτύων παρουσιάζουν μεγάλη ποικιλία, τόσο ως προς τον τρόπο κατασκευής τους όσο και στον τρόπο με τον οποίο εκπαιδεύονται. Για παράδειγμα ένα δίκτυο μπορεί να είναι πλήρως συνδεδεμένο, με βεβαρημένες διασυνδέσεις κατευθυνόμενες προς τα εμπρός και να εκπαιδεύεται με επιβλεπόμενο τρόπο ενώ ένα άλλο μπορεί να αποτελείται από αραιά διασυνδεδεμένους νευρώνες των οποίων ο το γράφημα παρουσιάζει κύκλους και το οποίο να εκπαιδεύεται με μη επιβλεπόμενο τρόπο.

Τέλος να σημειωθεί ότι, παρά την μεγάλη χρησιμότητα και επιτυχία των βαθιών τεχνιτών νευρωνικών δικτύων , η εκπαίδευση τους είναι ένα αρκετά δύσκολο εγχείρημα, καθώς ο χώρος των παραμέτρων τους είναι μεγάλος και περίπλοκος. Παρόλα αυτά τα τελευταία χρόνια (από το 2006 και έπειτα) με την άνθηση του κλάδου της βαθιάς μάθησης έχουν αναπτυχθεί πολλές τεχνικές οι οποίες ξεπερνάνε, σε έναν βαθμό, τις θεμελιώδεις αυτές δυσκολίες.

Κατά την σχεδίαση ενός νευρωνικού δικτύου, οι βασικές προδιαγραφές που πρέπει να αναλογιστεί κάποιος είναι [9]:

- Η τοπολογία του δικτιού (π.χ. έμπροσθεν τροφοδοτούμενο, αναδρομικό κ.λπ.)
- Το πλήθος των νευρώνων εισόδου και εξόδου
- Το πλήθος των κρυφών νευρώνων
- Η συνάρτηση ενεργοποίησης των νευρώνων ή αλλιώς το είδος τον νευρώνων (σιγμοειδής, Softmax κλπ.)
- Το πλήθος των κρυφών στρωμάτων
- Ο αλγόριθμος εκμάθησης

1.2 Ιστορική Αναδρομή

Ακολουθεί μια σύντομη ιστορική αναδρομή με τα κομβικότερους σταθμούς στην σύντομη ιστορία των νευρωνικών δικτύων. [11] [12] [13] [14]

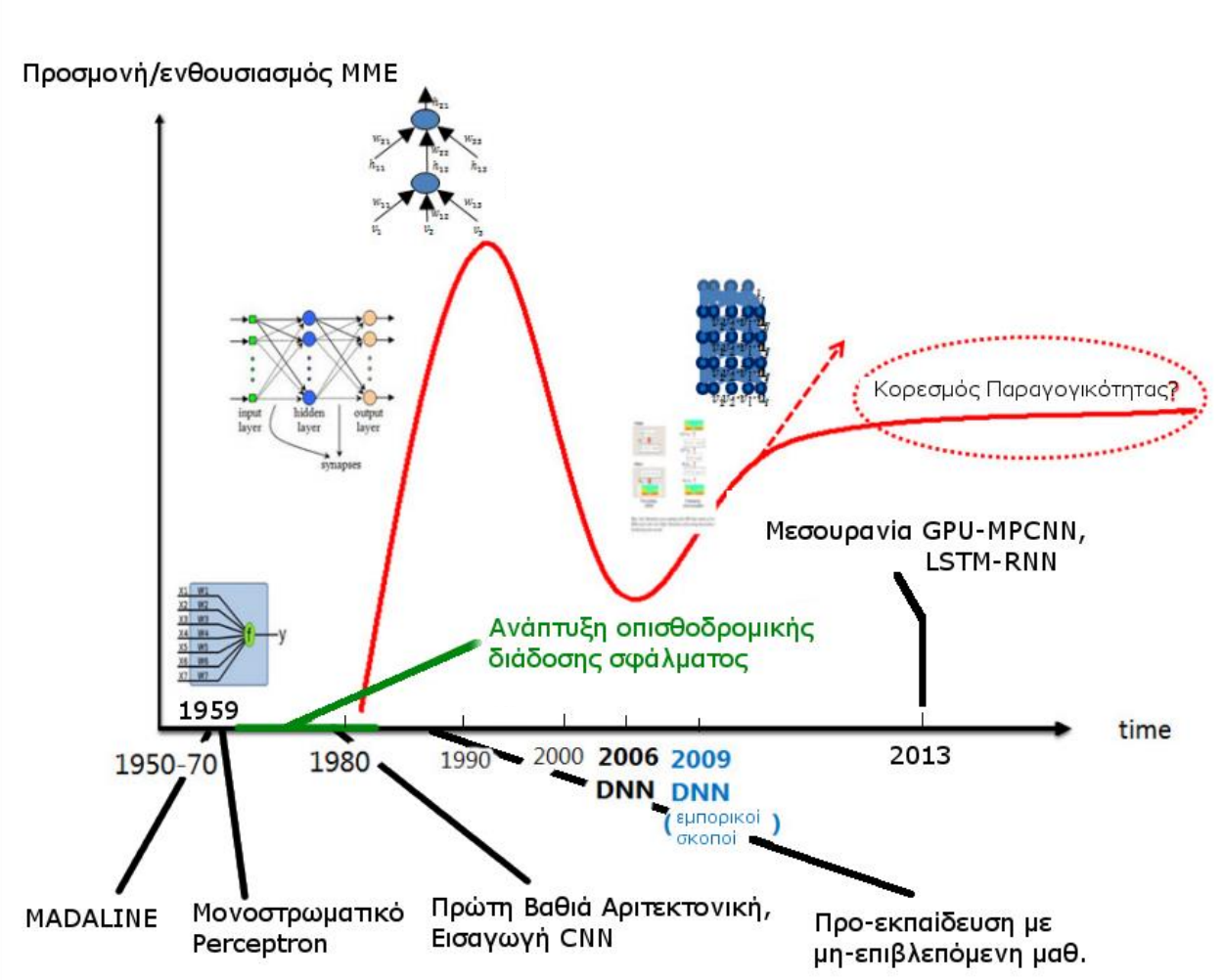
- **1943:** Πρώτη μαθηματική αναπαράσταση νευρωνικών δικτύων από τον νευροφυσικό Warren McCulloch και τον μαθηματικό Walter Pitts. Στην σχετική τους εργασία ανέλυσαν το πώς θα μπορούσαν να λειτουργούν οι νευρώνες του εγκεφάλου.
- **1949:** Ο Donald Hebb προτείνει το γεγονός ότι τα νευρικά μονοπάτια ενισχύονται σε κάθε τους χρήση.
- **1959:** Αναπτύσσεται από τους Bernard Widrow και Marcian Hoff του Stanford δύο μοντέλα αποκαλούμενα ADELINE και MADALINE (Multiple ADAptive LINear Elements). Το δίκτυο MADALINE ήταν το πρώτο δίκτυο που χρησιμοποιήθηκε για μη εργαστηριακή χρήση. Συγκεκριμένα χρησιμοποιήθηκε για το φιλτράρισμα ήχους κατά την διάρκεια τηλεφωνικών κλήσεων.
- **1960:** Κατασκευάζεται στο Cornell η πρώτη μηχανή εκμάθησης, το Perceptron μονο-στρωματικό δίκτυο
- **1965:** Βαθιά δίκτυα βασισμένα στην Ομαδική μέθοδο διαχείρισης δεδομένων (Group Method of Data Handling (GMDH)) εισάγονται από τον Ivakhnenko και άλλους.
- **1979:** Αναπτύσσεται το πρώτο πολυστρωματικό δίκτυο. Ήταν μη επιβλεπόμενης μάθησης.
- **1979:** Εισαγωγή των CNN, της υποδειγματοληψίας και της αναπαραγωγής βαρών.
- **1960-1981:** Ανάπτυξη του Αλγορίθμου οπισθοδρομικής διάδοσης σφάλματος.

- **1982:** Το ενδιαφέρον για τον κλάδο αναζωπυρώνεται από τον John Hopfield. Είσαγε την ιδέα ανάπτυξης δικτύων με χρήση αμφίδρομων συνδέσεων.
- **1980-1989:** Η οπισθοδρομική διάδοση σφάλματος αποκτά μεγάλη δημοτικότητα για την εκμάθηση παραμέτρων σε νευρωνικά δίκτυα.
- **1987:** Πρώτη αναφορά στην προ-εκπαίδευση δικτύων με χρήση μη-επιβλεπόμενων αλγορίθμων.
- **1989:** Χρήση αλγορίθμου οπισθοδρομικής διάδοσης σε CNN
- **1991:** Σε εργασία του Hochreiter αναγνωρίζεται ως κύριο αίτιο της δυσκολίας εκπαίδευσης βαθιών αρχιτεκτονικών με οπισθοδρομική διάδοση, το πρόβλημα των εξαφανιζόμενων κλίσεων. Με το πέρασμα του χρόνου προτάθηκαν πολλές μέθοδοι για την αντιμετώπιση του θεμελιώδους αυτού προβλήματος.
- **1992:** Αρχίζουν και χρησιμοποιούνται στρώματα max-pooling σε νευρωνικά δίκτυα. Αυτό ανοίγει τον δρόμο για την μετέπειτα χρήση τους σε CNN με μεγάλη επιτυχία.
- **1994:** Μοντέλα νευρωνικών δικτύων αρχίζουν και αναδεικνύονται σε νικήτριες θέσης διαγωνισμών αναγνώρισης προτύπων.
- **1995:** Δημιουργείται το δίκτυο επιβλεπόμενής μάθησης LSTM RNN το οποίο μπορεί να ξεπεράσει το πρόβλημα εξαφανιζόμενων κλίσεων χωρίς την χρήση μη-επιβλεπόμενης προ-εκπαίδευσης.
- **1997:** Bi-Directional RNNs
- **2003:** Τα νευρωνικά δίκτυα όχι μόνο εξακολουθούν να κερδίζουν διαγωνισμούς αναγνώρισης πρότυπων αλλά αρχίζουν και θέτουν ρεκόρ. Σε αυτό βοήθησε η περεταίρω ανάπτυξη των CNNs τα οποία πέτυχαν νέο ρεκόρ στο MNIST 0,4%.
- **2006/7:** Εγκαθιδρύεται ό όρος “Deep Learning”. Ο Hinton χρησιμοποιώντας DBNs δείχνει, θεωρητικά τουλάχιστον, ότι η προσθήκη περισσότερων στρωμάτων βελτιώνει ένα όριο στην αρνητική λογαριθμική πιθανότητα. Πετυχαίνοντας έτσι 1,2% ρυθμό σφάλματος στο MNIST. Οι Stacked Autoencoders γίνονται μια δημοφιλής εναλλακτική μέθοδος για την εκπαίδευση βαθιών έμπροσθεν τροφοδοτούμενων αρχιτεκτονικών. Περαιτέρω βελτίωση των CNN καθώς και χρήση GPU για την γρήγορη εκπαίδευσή τους.
- **2009:** Τα RNN καταφέρνουν για πρώτη φορά να κερδίσουν παγκόσμιους διαγωνισμούς αναγνώρισης προτύπων. Συγκεκριμένα γίνεται χρήση βαθιών αρχιτεκτονικών LSTM καθώς και άλλων.
- **2010:** Επιτυγχάνεται ρεκόρ 0,35% ρυθμός λάθους στο σετ δεδομένων MNIST. Χρησιμοποιήθηκε μια απλή βαθιά αρχιτεκτονική (MLP) εκπαιδευμένο με οπισθοδρομική διάδοση. Κομβικής σημασίας ήταν η χρήση στρεβλωμένων προτύπων δημιουργώντας έτσι ένα ακόμα μεγαλύτερο σετ δεδομένων και άρα αποφεύγοντας το overfitting.
- **2011:** Υπεράνθρωπη επίδοση όρασης επιτυγχάνεται στον IJCNN διαγωνισμό αναγνώρισης προτύπων με χρήση ενός Max Pooling CNN εκπαιδευμένο σε GPU (GPU-MPCNN). Συγκεκριμένα το βαθύ δίκτυο επέτυχε 0,56% ρυθμό σφάλματος, επίδοση δύο φορές καλύτερη από την αντίστοιχη ανθρώπινη.
- **2011:** Αποδείχτηκε ότι με την χρήση Hessian-Free βελτιστοποίησης μπορεί να ανακουφιστεί το βασικό πρόβλημα της βαθιάς μάθησης (εξαφανιζόμενες κλίσεις).
- **2012:** Για πρώτη φορά βαθύ δίκτυο κερδίζει πρώτη θέση σε διαγωνισμούς αναγνώρισης του σετ δεδομένων ImageNet. Συγκεκριμένα τα δίκτυα ήταν CNN με Max Pooling εκπαιδευμένα σε GPU. Την ίδια χρονιά το μεγαλύτερο νευρωνικό δίκτυο (10^9 παράμετροι) εκπαιδεύεται με μη-επιβλεπόμενο τρόπο σε 1000 μηχανές

και σύνολο σε 16000 πυρήνες πετυχαίνοντας το καλύτερο ως τώρα αποτέλεσμα σε σετ δεδομένων με 20000 κατηγορίες.

- **2013:** Τα βαθιά νευρωνικά δίκτυα με τα GPU-MPCNN και LSTM-RNN να μεσουρανούν, σπάνε πολλαπλά ρεκόρ σε διαγωνισμού αναγνώρισης προτύπων.

Ακολουθεί ένας κύκλος υπερβολής Gartner (Gartner hyper cycle) ο οποίος ευθυγραμμίζει τις διαφορετικές φάσεις της ανάπτυξης των Νευρωνικών Δικτύων με τις προσδοκίες της επιστημονικής κοινότητας καθώς και τον ενθουσιασμό των ΜΜΕ. [12]



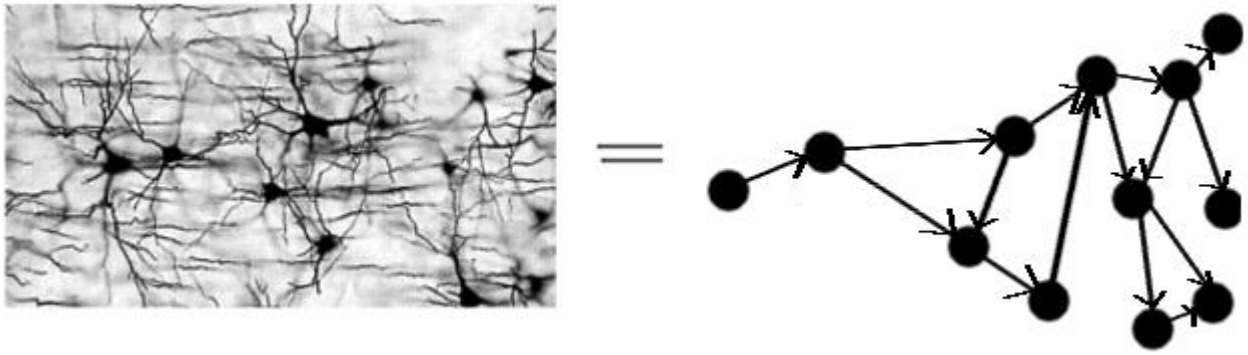
Εικόνα 1.3: Κύκλος Υπερβολής Gartner Αναφορικά με την Ιστορία των Νευρωνικών Δικτύων [12]

1.3 Νευρώνες

Στον πυρήνα της βαθιάς μάθησης βρίσκεται ο τεχνητός νευρώνας ο οποίος αποτελεί μαθηματική οντότητά εμπνευσμένη από την αντίστοιχη βιολογική. Η σχετική έρευνα έχει βασιστεί σε ανακαλύψεις των Santiago Ramon y Cajal και Sir Charles Scott Sherrington οι οποίοι στα μέσα του 20ου αιώνα διατύπωσαν την πρόταση, ότι παρά την ικανότητα των νευρώνων να επικοινωνούν μεταξύ τους, στην πραγματικότητα είναι διακριτές οντότητες και άρα πρέπει να μελετηθούν ως τέτοιές.

Ο εγκέφαλος αποτελείτε από 10 δισεκατομμύρια περίπου νευρώνες καθένας από τους οποίους συνδέετε με 10000 άλλους νευρώνες. Κάθε βιολογικός νευρώνας αποτελείται από

το κυτταρικό σώμα και από τις αποφυάδες οι οποίες διακρίνονται στους δενδρίτες και στο νευράξονα ή νευρίτη. Οι νευρώνες συνδέονται με άλλους νευρώνες (ή εκτελεστικά όργανα) μέσω ενός δικτύου δενδρίτων και τη βοήθεια νευροδιαβιβαστών. Μαθηματικά λοιπόν είναι δυνατή η αναπαράσταση ενός δίκτυού πολλαπλών νευρώνων με χρήση ενός δενδροειδούς γραφήματος όπου οι κόμβοι αναπαριστούν τους νευρώνες και οι ακμές τις συνάψεις μεταξύ τους. Και εφόσον οι συνάψεις επιτρέπουν ροή προς μια μόνο κατεύθυνση το γράφημα θα είναι κατευθυνόμενο



Εικόνα 1.4: Γραφική αναπαράσταση βιολογικού νευρωνικού δικτύου [15]

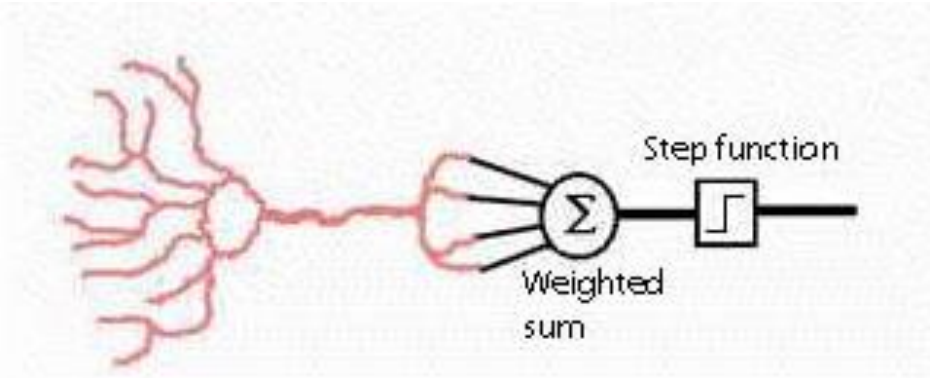
Όταν ένας νευρώνας δεν δέχεται σήματα από τους γειτονικούς νευρώνες, διατηρείται μια συγκεκριμένη τιμή διαφοράς δυναμικού, η επονομαζόμενη κατάσταση ηρεμίας. Η κατάσταση ηρεμίας διαταράσσεται μόνο αν ο νευρώνας δεχτεί ερέθισμα (μέσω των συνάψεων συνδεόμενων δενδρίτων) με ένταση μεγαλύτερη μίας τιμής κατωφλιού th. Όταν λοιπόν ένας νευρώνας δεχτεί σε κάποιο σημείο της μεμβράνης του ερέθισμα, με ένταση μεγαλύτερη του th, τιμή που διαφέρει από νευρώνα σε νευρώνα, το δυναμικό στο εσωτερικό του μεταβάλλεται αναλόγως της έντασης του εισερχόμενου σήματος. Οι σύντομες μεταβολές στο δυναμικό της μεμβράνης (δυναμικό ενεργείας) αποστέλλουν το ερέθισμα για αντίστοιχες αλλαγές σε γειτονικές περιοχές της μεμβράνης. Με αυτό τον τρόπο το δυναμικό ενεργείας μεταδίδεται κατά μήκος του νευράξονα σε στους συνδεόμενος νευρώνες (νευρική ώση). Η συνάρτηση ενός νευρώνα έχει λοιπόν την εξής μορφή:

$$output = \begin{cases} 1, & \sum_i x_i > th \\ 0, & x \leq b \end{cases}$$

Στην πράξη όμως, τα διάφορα σήματα εισόδου συνεισφέρουν διαφορετικά στο σήμα εισόδου το ποσοστό συμμετοχής κάθε σήματος εισόδου σε μια σύναψη ονομάζεται συναπτικό βάρος. Με την νέα αυτή πληροφορία η συνάρτηση του νευρώνα γίνεται:

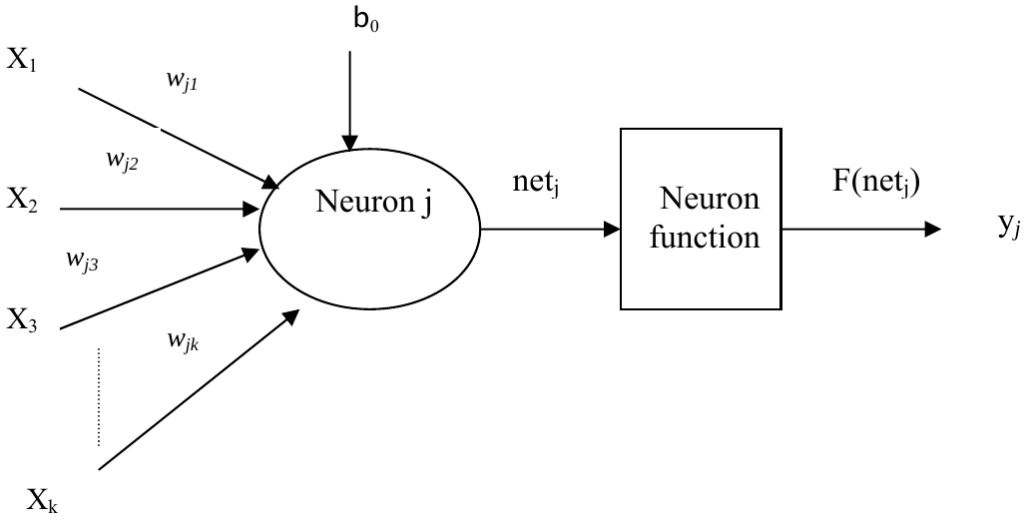
$$output = \begin{cases} 1, & \sum_i (input_i * weight_i) > b \\ 0, & x \leq b \end{cases}$$

Σχηματικά λοιπόν το αποτέλεσμα που προκύπτει είναι:



Εικόνα 1.5: Γραφική Αναπαράσταση Βιολογικού Νευρώνα [97]

Οι τεχνητοί νευρώνες έχουν επηρεαστεί από της βασικές ιδέες του βιολογικού νευρώνα. Στην γενικευμένη τους μορφή λοιπόν δέχονται k εισόδους x στης οποίες εφαρμόζουν τα συναπτικά βάρη w_{ji} (συναπτικό βάρος του νευρώνα j εφαρμοσμένο στη είσοδο i) και στη συνέχεια οι βεβαρημένες τιμές εισόδου αθροίζονται. Στο άθροισμα αυτό $\sum_i(w_{ij}x_i)$ θα προστεθεί τιμή ίση με το αντίθετό του κατωφλιού $b_0 = -th$. Η χρήση του b (bias ή σταθερά πόλωσης) αντί του κατωφλιού γίνεται προκειμένου να διευκολυνθεί η διαδικασία μάθησης ενός νευρωνικού δικτύου καθώς με αυτόν τον τρόπο δεν είναι απαραίτητη η χρήση ενός κανόνα εκμάθησης για το threshold και ενός άλλου για τα συναπτικά βάρη εφόσον το b μπορεί να γίνει αντιληπτό ως μια είσοδος x_0 της οποίας το βάρος είναι πάντα 1 και άρα να ενσωματωθεί αρμονικά στον κανόνα μάθησης για τα βάρη. Από το αποτέλεσμα των παραπάνω πράξεων παράγεται μια τιμή $net_j = \sum_i(w_{ij}x_i) - b_j$ τιμή η οποία θα «περάσει» μέσα από μια συνάρτηση $F(x)$ επονομαζόμενη ως συνάρτηση ενεργοποίησης (activation function), η οποία μάλιστα προσδιορίζει και το είδος του νευρώνα, παράγοντας έτσι την έξοδο y_j η οποία συναντάται και στην βιβλιογραφία με τον όρο «ενεργοποίηση του νευρώνα». Σχηματικά η λειτουργεία του νευρώνα συμπυκνώνεται στην Εικόνα 1.6:

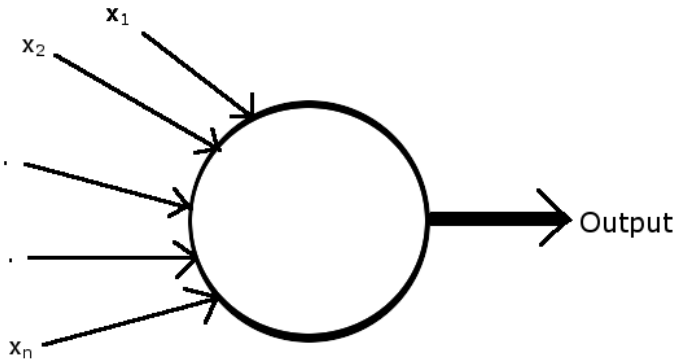


Εικόνα 1.6: Γραφική Αναπαράσταση Τεχνητού Νευρώνα

Ακολουθούν μερικά από τα βασικότερα είδη τεχνιτών νευρώνων τα οποία κατά κανόνα βασίζονται στο παραπάνω σχεδιάγραμμα.

1.3.1 Perceptrons

Τα Perceptrons των οποίον η μετάφραση στα ελληνικά θα μπορούσε να είναι οι «αντιληπτές» είναι ο πιο βασικός τύπος τεχνητού νευρώνα. Αναπτυχθήκαν κατά τα μέσα του εικοστού αιώνα από τον Frank Rosenblatt ο οποίος στηρίχτηκε κυρίως στην δουλειά του McCulloch-Pitts . Ένα Perceptron δέχεται πολλαπλές δυαδικές εισόδους και παράγει μία μοναδική, δυαδική έξοδο.

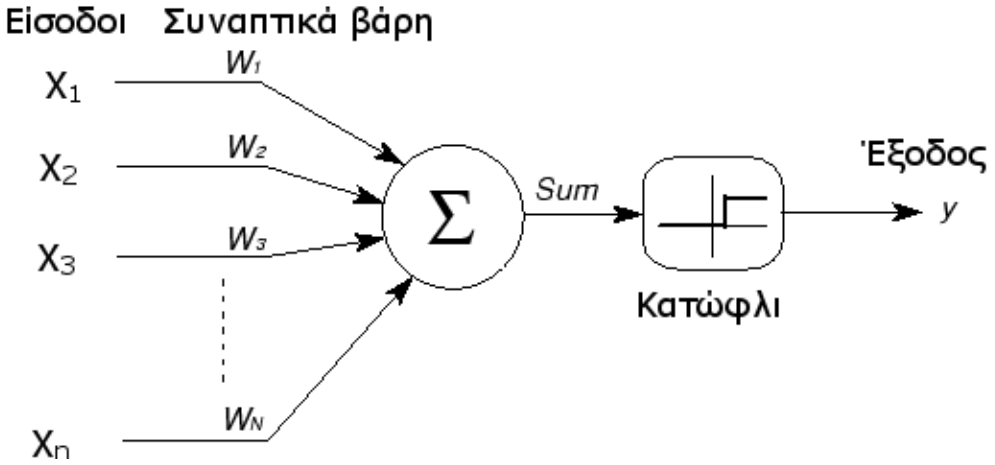


Εικόνα 1.7: Γράφος Νευρώνα τύπου Perceptron

Ο Rosenblatt πρότεινε έναν απλό κανόνα για τον υπολογισμό της εξόδου. Εισήγαγε τα συνοπτικά βάρη w_1, w_2, w_3 …., πραγματικούς αριθμούς οι οποίοι εκφράζουν την σημαντικότητα της κάθε εισόδου. Άρα λοιπόν, η έξοδος ενός perceptron ενεργοποιείται μόνο όταν το άθροισμα των εισόδων, μετά της επίδρασής των βαρών τους, ξεπεράσει μια τιμή κατωφλιού. Σε αλγεβρικούς όρους:

$$output = \begin{cases} 0, & \sum_j w_j * x_j \leq threshold \\ 1, & \sum_j w_j * x_j > threshold \end{cases}$$

Σχηματικά η λειτουργία ενός perceptron μπορεί να αναπαρασταθεί ως εξής:



Εικόνα 1.8: Σχηματική Αναπαράσταση Νευρώνα τύπου Perceptron

Στην σύγχρονη βιβλιογραφία το κατώφλι εισάγεται στην εξίσωση των νευρώνων υπο την μορφή της σταθεράς πόλωσης (bias), όπου $bias = -threshold$ η οποία όπως αναφέρθηκε και προηγουμένως, μπορεί να εκληφθεί ως ένα επιπλέων συναπτικό βάρος του οποίου η είσοδος είναι πάντα ίση με ένα. Με την εισαγωγή της σταθεράς πόλωσης η εξίσωση του perceptron γίνεται:

$$y = \begin{cases} 0, & \sum_j (w_j * x_j) + b \leq 0 \\ 1, & \sum_j (w_j * x_j) + b > 0 \end{cases}$$

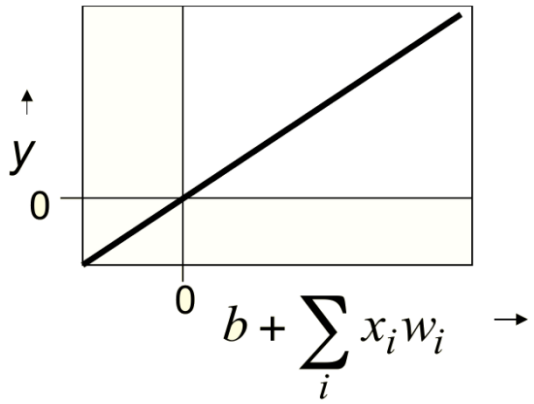
Η εφεύρεση του Perceptron αρχικά δημιούργησε μεγάλες προσδοκίες στην επιστημονική κοινότητα, οι οποίες σύντομα όμως, ανεδείχθησαν εσφαλμένες. Το 1969 οι Minsky και Paparet [10] απέδειξαν ότι τα Perceptrons δεν μπορούσαν να εκπαιδευτούν με σκοπό την αναγνώριση πολλαπλών κατηγοριών προτύπων, εξαιτίας των περιορισμών που επιβάλει η δυαδική τους φύση. Παρ' όλα αυτά, χάρη στις αλματώδες εξελίξεις των τελευταίων χρόνων, Perceptrons χρησιμοποιούνται στο Deep Learning υπο την μορφή των πολυστρωματικών Perceptrons (MLP) με αρκετή επιτυχία.

1.3.2 Γραμμικοί νευρώνες (Linear Neurons)

Η έξοδος των γραμμικών νευρώνων όπως άλλωστε δηλώνει και το όνομα τους αποτελεί γραμμική εξάρτηση των εισόδων. Όπως είναι αναμενόμενο άλλωστε, η υπολογιστική τους δυνατότητα είναι μικρή εξαιτίας της απλότητας τους. Οι είσοδοι ενός γραμμικού νευρώνα, ακριβώς όπως στα Perceptrons, πολλαπλασιάζονται με τα αντίστοιχα συναπτικά βάρη και από το άθροισμα αφαιρείται η σταθερά πόλωσης. Ή έξοδος ενός γραμμικού νευρώνα ισούται με το αποτέλεσμα των παραπάνω πράξεων. Αλγεβρικά:

$$y = \sum_j (w_j * x_j) + b$$

Άρα το διάγραμμα της συνάρτησης εξόδου είναι:



Διάγραμμα 1.1: Σχέση Εισόδου-Εξόδου Γραμμικού Νευρώνα [16]

1.3.3 Επιδιορθωμένοι Γραμμικοί νευρώνες (Rectified Linear Neurons RELUs)

Οι Επιδιορθωμένοι Γραμμικοί νευρώνες ή αλλιώς νευρώνες γραμμικού κατωφλιού είναι γραμμικοί νευρώνες στην έξοδο των οποίων εφαρμόζεται ένας ανορθωτής της μορφής :

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

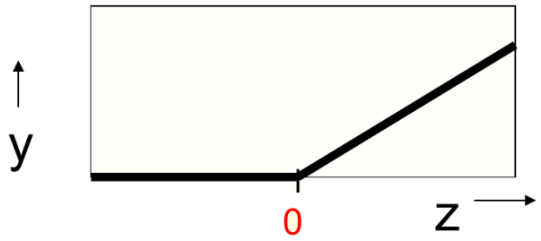
Για τον υπολογισμό της τιμής ενεργοποίησης του νευρώνα (έξοδος y), υπολογίζεται αρχικά το βεβαρημένο άθροισμα των εισόδων μετ' επίδραση της σταθεράς πόλωσής και στην συνέχεια το αποτέλεσμα αυτό δίδεται ως είσοδος στον ανορθωτή. Άρα λοιπόν ή έξοδος ενός νευρώνα γραμμικού κατωφλιού έχει την μορφή:

$$y = \begin{cases} 0, & \zeta \leq 0 \\ \zeta, & \zeta > 0 \end{cases}$$

Όπου:

$$\zeta = \sum_j (w_j * x_j) + b$$

Σχηματικά:



Διάγραμμα 1.2: Σχέση Εισόδου-Εξόδου Επιδιορθωμένου Γραμμικού Νευρώνα [16]

Η εφαρμογή του ανορθωτή στην έξοδο ενός γραμμικού νευρώνα παρουσιάζει διάφορα πλεονεκτήματα [17], κάποια εκ των οποίων είναι:

- Σπανιότερη ενεργοποίηση του νευρώνα και άρα μικρότερο υπολογιστικό κόστος κατά την εκπαίδευση του δικτύου.
- Αποδοτική διάδοση του διανύσματος κλίσης (χρήσιμο στον gradient descent καθώς και σε άλλους αλγορίθμους) καθώς δεν εμφανίζεται το θεμελιώδες πρόβλημα της βαθιάς μάθησης.
- Αποδοτικός υπολογισμός καθώς για τον υπολογισμό της εξόδου γίνεται χρήση σύγκρισης, πρόσθεση και πολλαπλασιασμού

1.3.4 Maxout Νευρώνες

Οι νευρώνες Maxout [18] (αυτολεξεί μετάφραση: «μέγιστο προς τα έξω») προτάθηκαν το 2013 από τον Ian J. Goodfellow και άλλους [19] και χρησιμοποιούνται κυρίως ως κρυφοί νευρώνες ή νευρώνες εξόδου. Οι νευρώνες αυτοί έχουν ως σκοπό την μέγιστη συγκέντρωση ή οποία είναι μορφή μη γραμμικής υποδειγματοληψίας (χρησιμοποιείται συχνά στα Convolutional neural networks (CNN)). Ένας τέτοιος νευρώνας ο οποίος βρίσκεται στο στρώμα l καλείται να επιλέξει την μέγιστη εκ των k εισόδων. Δοθείσας εισόδου $x \in \mathbb{R}^d$, η οποία μπορεί να είναι είτε το διάνυσμα εισόδου είτε η ενεργοποίηση του προηγούμενου νευρώνα, ένας Maxout νευρώνας υπολογίζει την έξοδο του νευρώνα ως εξής:

$$y_l^i(x) = \max_{j \in [1,k]} z_l^{ij}$$

Όπου ο όρος z^{ij} πρόκειται ουσιαστικά για έναν χάρτη χαρακτηριστικών (feature map), είναι δηλαδή ένα διάνυσμα το οποίο περιγράφει κάποια συγκεκριμένα χαρακτηριστικά της εισόδου. Σκοπός του νευρών είναι να επιλέξει το χαρακτηριστικό που παρουσιάζεται πιο έντονα στην δοθείσα είσοδο. Το διάνυσμα $z_l \in R^z$ αποτελούμενο από χάρτες χαρακτηριστικών του προηγούμενου στρώματος υπολογίζεται ως εξής:

$$z_l = W_{l-1}^T y_{l-1} + b_l$$

- Όπου W_{l-1}^{T} είναι ο πίνακας βαρών του προηγούμενου στρώματος $l-1$
- Όπου b_i είναι το διάνυσμα των σταθερών πόλωσής του στρώματος l
- Όπου y_{l-1} είναι το διάνυσμα των ενεργοποιήσεων του προηγούμενου στρώματος $l-1$.

Συνοψίζοντας, ένας νευρώνας Maxout, πραγματοποιεί λειτουργεία συγκέντρωσης (pooling operation) επί ενός υπο-χώρου k γραμμικών αντιστοιχίσεων (η λειτουργεία αυτή αναφέρεται και ως συγκέντρωση υπο-χώρου (subspace pooling) και χρησιμοποιείται στα CNN).

1.3.4.1 Probout Νευρώνες

Οι Probout νευρώνες [20] αποτελούν στοχαστική γενίκευση των Maxout οι οποίοι επιτυγχάνουν να διατηρήσουν τις επιθυμητές ιδιότητες των προγόνων τους ενώ παράλληλα βελτιώνουν την διάδοση των διανυσμάτων κλίσης κατά την εκπαιδευτική διαδικασία. Η συνάρτηση ενεργοποίηση των νευρώνων αυτών προκύπτει από την αντικατάσταση τής Maxout συνάρτησης με μία πιθανολογική συνάρτηση δειγματοληψίας. Συγκεκριμένα γίνεται χρήση μιας κατανομής Boltzmann επί k γραμμικών χαρτών χαρακτηριστικών p_i και στην συνέχεια δειγματοληπτείται η συνάρτηση ενεργοποίησης $y(x)_{probout}$ από την ενεργοποίηση των νευρώνων που ανήκουν στον αντίστοιχο υπο-χώρο.

$$p_i = \frac{e^{\lambda z_i}}{\sum_j^k e^{\lambda \zeta_j}}$$

Όπου λ είναι μια υπερπαράμετρος, ονομαζόμενή ως παράμετρος αντίστροφης θερμοκρασίας, η οποία ελέγχει την διακύμανση της κατανομής. Άρα η συνάρτηση ενεργοποίησης ενός νευρώνα γίνεται:

$$y(x) = \zeta_i, \qquad \acute{o}\pi ου\ \iota \sim Πολυωνιμικό\{p_1, p_2, \dots, p_k\}$$

Ουσιαστικά λοιπόν ο νευρώνας Maxout είναι ο Probout με $\lambda \to \infty$. Προκειμένου να επιτευχθεί ομαλοποίηση του μεμαθημένου μοντέλου συχνά εισάγεται και η τεχνική dropout που θα περιγράφει ενδελεχώς αργότερα. Με χρήση dropout η συνάρτηση των πιθανοτήτων γίνεται:

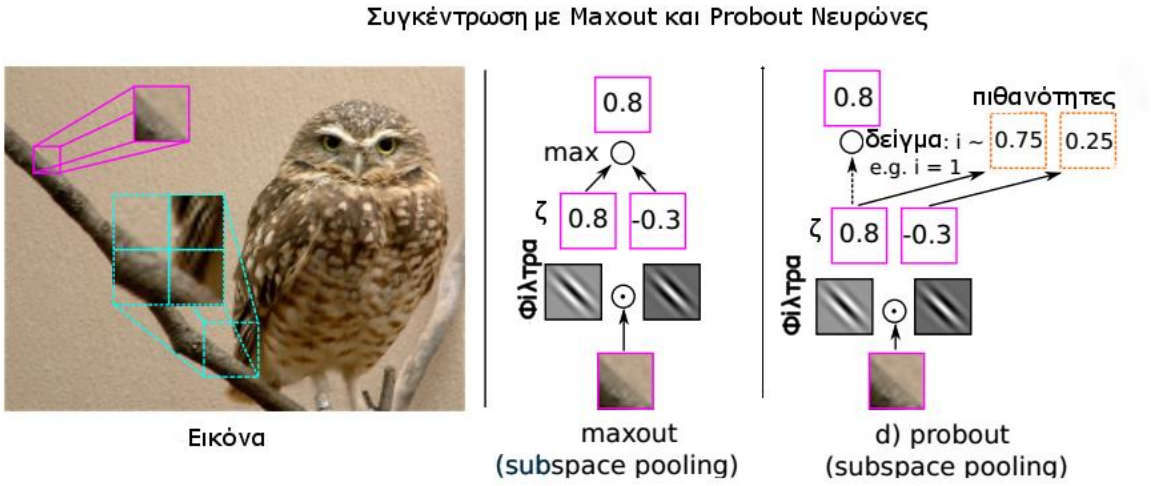
$$\widehat{p_0} = 0.5$$

$$\widehat{p_{\iota \neq 0}} = \frac{e^{\lambda \zeta_i}}{2 * \sum_j^k e^{\lambda \zeta_j}}$$

Και άρα η συνάρτηση ενεργοποίησης γίνεται:

$$y_{probout}(x) = \begin{cases} 0, & i = 0 \\ z_i, & \alpha\lambda\lambda\iota\acute{\omega}\varsigma \end{cases}, \qquad \text{όπου } \iota \sim Πολυωνιμικό\ \{\widehat{p_0}, \widehat{p_1}, \dots, \widehat{p_k}\}$$

Στην παρακάτω εικόνα απεικονίζεται η λειτουργεία υποδειγματοληψίας δύο νευρώνων Maxout και Probout σε μία εικόνα.

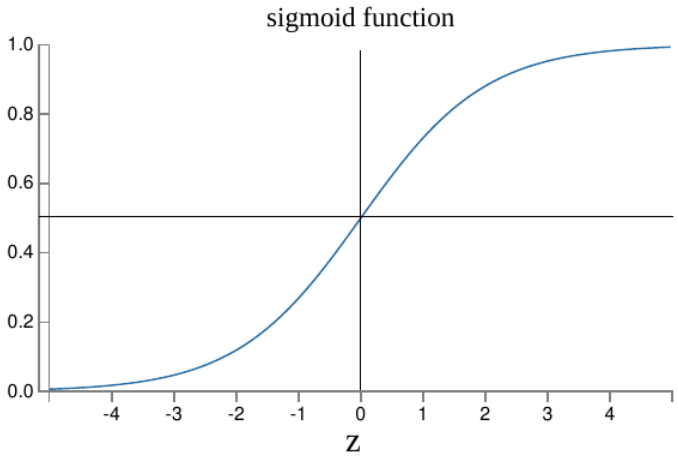


Εικόνα 1.9: Λειτουργεία Συγκέντρωσης με χρήση Maxout και Probout Νευρώνες [20]

1.3.5 Σιγμοειδείς νευρώνες (Sigmoid Neurons)

Οι σιγμοειδής νευρώνες [21] έχουν πάρει το όνομα τους από την σιγμοειδής συνάρτηση ενεργοποίησης που χρησιμοποιούν η οποία με την σειρά ονομάστηκε έτσι εξαιτίας της ομοιότητας της με τον αγγλικό χαρακτήρα «S» όπως φαίνεται και από το σχήμα:

$$s(\zeta) = \frac{1}{1 + e^{-\zeta}} \quad \text{όπου:} \quad \zeta = \sum_j (w_j * x_j) + b \qquad \text{(SGM1)}$$

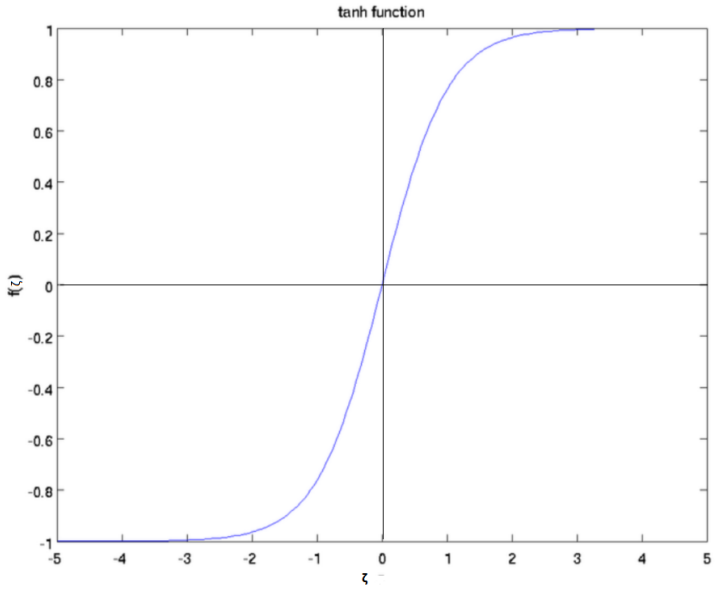


Διάγραμμα 1.3: Σχέση Εισόδου-Εξόδου Σιγμοειδούς Νευρώνα [22]

Η σιγμοειδής πρόκειται για μια συνεχή συνάρτηση της οποία ο όρος $e^{-\zeta}$, για μεγάλες τιμές εισόδου τείνει στο 0 ($\lim_{\infty} e^{-\zeta} = 0$) και άρα η έξοδος του νευρώνα τείνει στο 1 και αντίστοιχα για μικρή και αρνητική είσοδο γίνεται $\lim_{-\infty} e^{-\zeta} = \infty$ και άρα η έξοδος του νευρώνα τείνει στο 0,5. Η έξοδος του σιγμοειδούς νευρώνα δηλαδή, κοινωνικοποιείται στο διάστημα $[0,1]$.

Οι σιγμοειδείς νευρώνες είναι ευρέως χρησιμοποιημένοι σήμερα. Ένας από τους κύριους λόγους είναι ότι κατά την εκπαίδευση τους, διαδικασία κατά την οποία γίνονται αλλαγές στα συναπτικά βάρη και στις σταθερές πόλωσης των νευρώνων ενός δικτύου, μια μικρή αλλαγή στις τιμές ενός νευρώνα προκαλεί εξίσου μικρή αλλαγή στην εξόδου κάνοντας την εκπαιδευτική διαδικασία πιο ομαλή. Αυτή η ομαλή συμπεριφορά τους οφείλεται στο γεγονός ότι τα παράγωγα της συνάρτησης είναι ομαλά και άρα «συμπεριφέρονται καλά».

Να σημειωθεί ότι υπάρχουν σιγμοειδείς νευρώνες των οποίων η συνάρτηση ενεργοποίησης παρεκκλίνει ελαφρός αυτήν που δόθηκε (SGM1). Ένα τέτοιο παράδειγμα είναι ο $tanh$ σιγμοειδής νευρώνας με συνάρτηση ενεργοποίησης την $tanh(\zeta) = \frac{e^{\zeta} - e^{-\zeta}}{e^{\zeta} + e^{-\zeta}}$ η οποία κανονικοποιεί την ενεργοποίηση του στο διάστημα $[-1,1]$.



Διάγραμμα 1.4 Σχέση Εισόδου-Εξόδου Σιγμοειδούς Νευρώνα $tanh$ [22]

1.3.6 Στοχαστικοί δυαδικοί νευρώνες (Stochastic Binary neurons)

Οι στοχαστικοί νευρώνες παράγουν εγγενώς τυχαία έξοδο. Η διαδικασία υπολογισμού της βεβαρημένης είσοδού είναι ακριβώς ίδια με αυτή των προαναφερθέντων (ντετερμινιστικοί νευρώνες) και χρησιμοποιούν μια από τις παραπάνω συναρτήσεις ενεργοποίησης για να υπολογίσουν όχι την έξοδο αλλά την πιθανότητα να παράξουν 1 στην έξοδο (πιθανότητα ενεργοποίησης νευρώνα). Για παράδειγμα ένας στοχαστικός δυαδικός σιγμοειδής νευρώνας παράγει 1 στην έξοδο με πιθανότητα ίση με το αποτέλεσμα της εξόδου ενός σιγμοειδούς νευρώνα. Αλγεβρικά:

$$P(y=1) = \frac{1}{1+e^{-\zeta}} \qquad \text{όπου}, \zeta = \sum_j (w_j * x_j) + b$$

1.4 Μάθηση

Η πιο ενδιαφέρουσα και ελκυστική ιδιότητα των νευρωνικών δικτύων είναι η δυνατότητα τους να μαθαίνουν. Δηλαδή, δοθέντος ενός συγκεκριμένου προβλήματος προς επίλυση και μιας τάξης συναρτήσεων F (συναρτήσεις ενεργοποίησης νευρώνων) ένα νευρωνικό δίκτυο μπορεί να μάθει, δηλαδή να βρει ομάδα συναρτήσεων $f^* \in F$ η οποία να λύνει το πρόβλημα αυτό με έναν βέλτιστό τρόπο.

Προκειμένου τα νευρωνικά δίκτυα να μάθουν είναι απαραίτητη η ύπαρξη κάποιον παραδειγμάτων εισόδου (π.χ. εικόνες, φωνήματα κ.λπ.). Τα δεδομένα αυτά μπορούν να τροφοδοτηθούν στο νευρωνικό δίκτυο με δύο τρόπους οι οποίοι αν και παρόμοιοί παράγουν διαφορετικά αποτελέσματα.

- Κατά Παρτίδες (Batch Learning): Συνηθώς όταν χρησιμοποιείται ένα μεγάλο σετ δεδομένων, αυτό διαιρείται σε μικρές υπο-ομάδες (mini-batches). Αυτό γίνεται με σκοπό την εκπαίδευση του δικτύου συνολικά στην υπο-ομάδα και όχι σε κάθε μεμονωμένο δεδομένο. Ως αποτέλεσμα μειώνεται ελαφρώς ο χρόνο εκπαίδευσης. Κατά την εκπαίδευση σε παρτίδες απαιτείται να γίνει επιλογή του μεγέθους υπο-παρτίδας, μέγεθος το οποίο επηρεάζει άμεσα την ταχύτητα εκπαίδευσης.
- Σε απευθείας σύνδεση (Online learning): Το νευρωνικό δίκτυο μαθαίνει κάθε δεδομένο ξεχωριστά. Δηλαδή τα βάρη του δικτύου αναβαθμίζονται μετά το «πέρασμα» κάθε μεμονωμένου παραδείγματος. Η μέθοδος αυτή πρόκειται ουσιαστικά για Batch Learning αλλά με μέγεθος υπο-ομάδας ίσο με ένα.

Όσον αφορά τον τρόπο εκμάθησης δικτύου συνηθώς ακολουθείται ο εξής τριμερής διαχωρισμός:

- Επιβλεπόμενη Μάθηση: Επιτηρούμενη μάθηση ή μάθηση με επίβλεψη (Supervised learning), όπου ο αλγόριθμος κατασκευάζει μια συνάρτηση που απεικονίζει δεδομένες εισόδους σε γνωστές, επιθυμητές εξόδους (σύνολο εκπαίδευσης), με απώτερο στόχο τη γενίκευση της συνάρτησης αυτής και για εισόδους με άγνωστη έξοδο (σύνολο ελέγχου).
- Μη-Επιβλεπόμενη Μάθηση: ανεπίβλεπτη μάθηση ή Μη-Επιτηρούμενη μάθηση (Unsupervised learning), όπου ο αλγόριθμος κατασκευάζει ένα μοντέλο για κάποιο σύνολο εισόδων χωρίς να γνωρίζει επιθυμητές εξόδους για το σύνολο εκπαίδευσης.
- Ενισχυτική Μάθηση: (Reinforcement learning), όπου ο αλγόριθμος μαθαίνει μια στρατηγική ενεργειών για μια δεδομένη παρατήρηση. Ανακαλύπτει ουσιαστικά να επιλέγει μια ενέργεια προκειμένου να μεγιστοποιήσει την «πληρωμή» (payoff). Η ενισχυτική μάθηση δεν θα αναλυθεί στην παρούσα εργασία καθώς εφαρμογές της στα νευρωνικά δίκτυα είναι περιορισμένες.

1.4.1 Επιβλεπόμενη Μάθηση

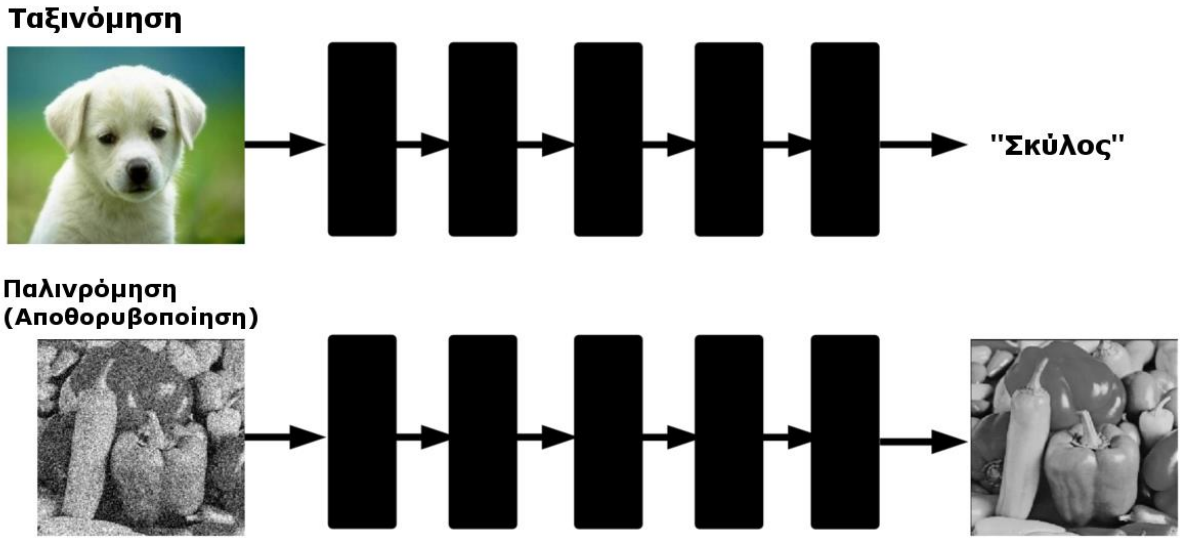
[16] Τα βαθιά δίκτυα που χρησιμοποιούν επιβλεπόμενη μάθηση είναι ιδιαίτερα αποδοτικά στην ταξινόμηση προτύπων ιδιότητα που συνεισφέρει στην μεγάλη δημοτικότητα που έχουν γνωρίσει τα τελευταία χρόνια. Προκειμένου να λειτουργήσει η επιβλεπόμενη μάθηση απαιτείται ένα σύνολο από δεδομένα εκπαίδευσης με ετικέτα. Πρέπει δηλαδή, για κάθε ένα

από τα δείγματα εισόδου να δίδεται και η πληροφορία του τι αναπαριστά. Τυπικά οι αλγόριθμοι επιβλεπόμενης μάθησης λειτουργούν ως εξής:

- Αρχικά επιλέγεται η κατηγορία του μοντέλου, ουσιαστικά δηλαδή γίνεται επιλογή του είδους των τεχνιτών νευρώνων και συνεπώς της συναρτήσεων ενεργοποίησης $y = f(x; W)$. Η συνάρτηση αυτή έχει ως σκοπό την χαρτογράφηση του διανύσματος εισόδου x σε μια προβλεπόμενη έξοδο y χρησιμοποιώντας τις παραμέτρους W.
- Στην συνέχεια ακολουθεί η διαδικασία εκπαίδευσης του δικτύου. Η διαδικασία αυτή, συνήθως, ισοδυναμεί με την προσαρμογή των παραμέτρων του δικτύου (συναπτικά βάρη και σταθερών πόλωσής) με σκοπό την ελαχιστοποίηση της ασυμφωνίας τής εξόδου του δικτύου y και της επιθυμητής εξόδου t (ετικέτας) για κάθε εκπαιδευτική περίπτωση. Η ασυμφωνία των δύο αυτών μεγεθών μετράτε με την χρήση μιας συνάρτησης κόστους $C(x, t)$

Υπάρχουν δύο κύρια είδη επιβλεπόμενης εκμάθησης:

- Η παλινδρόμηση (regression) όπου το $y \in \mathbb{R}$ ή y είναι διάνυσμα πραγματικών αριθμών (π.χ. η τιμή μιας μετοχής σε διάστημα 6 μηνών).
- Η ταξινόμηση (classification) όπου η επιθυμητική έξοδος y αντιπροσωπεύει ετικέτα τάξης (π.χ. επιλογή ανάμεσα σε 10 ψηφία)



Εικόνα 1.10: Σχηματική Αναπαράσταση λειτουργείας Ταξινόμησης και Παλινδρόμησης σε εικόνα από βαθύ τεχνητό νευρωνικό δίκτυο

1.4.1.1 Αλγόριθμοι Ταξινόμησης/Παλινδρόμησης

1.4.1.1.1 Γραμμική παλινδρόμηση (linear regression)

Είναι το βασικότερο εργαλείο της επιβλεπόμενης μάθησης [22] και αποτελεί βάση πιο περίπλοκών αλγορίθμων. Ο στόχος της γραμμικής παλινδρόμησης είναι να προβλέψει μια επιθυμητή τιμή t από έναν πίνακα τιμών εισόδου $x \in \mathbb{R}^n$. Για παράδειγμα, κατά διαδικασία αναγνώρισης του αριθμού που απεικονίζει μια εικόνα , το x^τ είναι γραμμικό διάνυσμα με τα pixel της εικόνας και το y^τ συμβολίζει την πρόβλεψη του αριθμού που απεικονίζει η εικόνα αυτή. Η είσοδος και η έξοδος συνδέονται με μία γραμμική εξίσωση $F_\theta(x) = \theta^\tau * x = y$ η οποία, εφόσον στα νευρωνικά δίκτυα χρησιμοποιούνται διανύσματα βαρών w και σταθερές πόλωσης b για τον προσδιορισμό της εξόδου, είναι συναρτήσει του w, b. Έτσι λοιπόν η γραμμική εξίσωση γίνεται $F_{w,x}(x)$.

Προκειμένου να εκπαιδευτεί ένα δίκτυο θα δοθούν πολλές τέτοιές εικόνες όπου $x^{(i)}$ και $t^{(i)}$ είναι τα χαρακτηριστικά (είσοδος – επιθυμητή έξοδος) της i-οστής εικόνας. Σκοπός είναι να μεταβληθεί η εξίσωση έτσι ώστε να ισχύει: $t^{(\mathrm{i})} = F(x^{(i)})$ για όσο το δυνατών περισσότερες εικόνες του δειγματικού χώρου. Με άλλα λόγια, επιδιώκεται η ελαχιστοποίηση της διανυσματικής απόστασης των $F_{w,x}(x)$ και t. Η απόσταση τους εκφράζεται μέσω μια συνάρτησης κόστους $C(w, b)$ εκ τον οποίων η πιο δημοφιλής, είναι η τετραδική συνάρτηση κόστους (quadratic cost function/mean squared error)

$$C(w, b) = \frac{1}{2n}\left(\sum_i \left(F_{w,b}\left(x^{(i)}\right) - t^{(i)}\right)\right)^2$$

Ο λόγος της δημοτικότητας αυτής της συνάρτησης κόστους είναι, ότι ακόμα και μικρές αλλαγές στα βάρη και στις σταθερές πόλωσης προκαλούν παρατηρήσιμες αλλαγές στην τιμή της [21].

Για την ελαχιστοποίηση της $C(w, b)$ μέσω της κατάλληλης επιλογής συνοπτικών βαρών και σταθερών πόλωσης των νευρώνων του δικτύου υπάρχει μεγάλη πληθώρα διαθέσιμων αλγορίθμων. Οι περισσότεροι εξ αυτών απαιτούν των υπολογισμό της συνάρτησης κόστους $C(w, b)$ καθώς και του διανύσματος κλίσης (αναδέλτα) της, $\nabla C(w, b)$, για κάθε επιλογή (w,b). Αναδέλτα:

$$\nabla_\theta J(\theta) = \begin{bmatrix} \frac{\vartheta J(\theta)}{\vartheta \theta_1} \\ \frac{\vartheta J(\theta)}{\vartheta \theta_2} \\ \vdots \\ \frac{\vartheta J(\theta)}{\vartheta \theta_n} \end{bmatrix}$$

Να σημειωθεί εδώ ότι η το διάνυσμα κλίσεως μια συνάρτησης μας διαφορίσιμης συνάρτησης $f(x)$ είναι ένα διάνυσμα που «δείχνει» στην κατεύθυνση στην οποία η τιμές της αυξάνονται με μεγαλύτερο ρυθμό. Αφού υπολογιστούν αυτά τα μεγέθη ο αλγόριθμος βελτιστοποίησης θα βρει την καλύτερη επιλογή ζευγών (w,b) ένας εκ τον οποίον είναι η απότομη κάθοδος (gradient descent).

1.4.1.1.2 Εφοδιαστική Παλινδρόμηση (logistic regression)

Η Εφοδιαστική Οπισθοδρόμηση [22] είναι ένας απλός αλγόριθμος ταξινόμησης (classification algorithm). Εφαρμόζεται σε περιπτώσεις που οι προς πρόβλεψη μεταβλητές είναι διακριτές και οι επιθυμητές έξοδοι δυαδικές $\left(t^{(\mathrm{i})} \in \{0,1\}\right)$. Κατά την εφοδιαστική παλινδρόμηση επιδιώκεται η πρόβλεψη της πιθανότητας, η δοθείσα είσοδος να ανήκει στην κλάση «1» ή στην «0» . Η συνάρτηση $F_{w,b}(x)$ αποκτά την εξής μορφή:

$$P(t=1|x) = F_{w,b}(x) = \frac{1}{1+\exp(-(w,b)^{\tau}x)} \equiv \sigma[(w,b)^{\tau}x]$$

$$P(t=0|x) = 1 - P(y'=1|x) = 1 - F_{w,b}(x)$$

Οπότε αν $y^{(i)} = P\left(t^{(\mathrm{i})} = 1\middle|x\right) > 0{,}5$ η είσοδος ταξινομείται ως κλάσης «1». Αντιστοίχως αν $y^{(i)} = P\left(t^{(\mathrm{i})} = 0\middle|x\right) > 0{,}5$ ταξινομείται ως κλάσης «0»

Αξίζει να σημειωθεί ότι η διανομή $P(t|x)$ που ακολουθούν οι παραπάνω συναρτήσεις είναι Bernoulli καθώς η εξαρτώμενη μεταβλητή είναι δυαδική. Ένας νευρώνας που μπορεί να χρησιμοποιηθεί κατά την εφοδιαστική παλινδρόμηση είναι ο σιγμοειδείς, καθώς η συνάρτηση ενεργοποίησης του, $s(z) = \frac{1}{1+e^{-z}}$,κανονικοποιεί την έξοδο του νευρώνα στο χώρο $[0,1]$.

Όσον αφορά το ζητούμενο, σε αυτή την περίπτωση, είναι να δοθούν κατάλληλες τιμές στις μεταβλητές (w,b) ώστε η πιθανότητα $P(t=1|x) = F_{w,b}(x)$ να είναι μεγάλη όταν το x ανήκει στην κλάση «1» και αντιστοίχως μικρή όταν ανήκει στην «0». Για ένα εκπαιδευτικό δείγμα m δειγμάτων η ακόλουθη συνάρτηση κόστους εκφράζει το πόσο επιτυχώς το πετυχαίνει αυτό.

$$C(w,b) = -\sum_i \left(t^{(i)} log\left(F_{w,b}\left(x^{(i)}\right)\right) + \left(1-t^{(i)}\right)\log\left(1-F_{w,b}\left(x^{(i)}\right)\right)\right)$$

Προφανώς, λόγο της δυαδικής φύσης της ετικέτας $t^{(i)}$ μόνο ένας εκ των δύο όρων του αθροίσματος θα είναι μη μηδενικός σε κάθε περίπτωση, άρα όταν $t^{(i)} = 1$, ελαχιστοποίηση της $C(w,b)$ σημαίνει ότι πρέπει να αυξηθεί η τιμή της εξόδου $F_{w,b}\left(x^{(i)}\right)$, αντιστοίχως όταν $t^{(i)} = 0$ πρέπει να αυξηθεί η τιμή του όρου $\left(1-F_{w,b}\left(x^{(i)}\right)\right)$ και άρα να μειωθεί το $F_{w,b}\left(x^{(i)}\right)$. Για την ελαχιστοποίηση της συνάρτησης κόστους μπορούν να χρησιμοποιηθούν αλγόριθμοι αντίστοιχοι με αυτοί της γραμμικής οπισθοδρόμησης να γίνεται δηλαδή χρήση του κανόνα Δέλτα [23].

1.4.1.1.3 Softmax παλινδρόμηση (Softmax Regression)

Η Softmax Οπισθοδρόμηση [22] ανήκει στην κατηγορία των «Multinomial logistic regression» που αποτελούν γενικεύσεις της γραμμικής οπισθοδρόμησης σε προβλήματα ταξινόμησης. Η Softmax Οπισθοδρόμηση θεωρεί επιτρεπτό μια ετικέτα t να λάβει πάνω από τιμή χωρίς να είναι απαραίτητο να είναι δυαδική. Αυτή η ιδιότητα είναι χρήσιμη σε διάφορων τύπων προβλημάτων όπως για παράδειγμα η ταξινόμηση ψηφιών του MNIST dataset οπού οι ετικέτες ανήκουν στον δειγματικό χώρο $[0,9]$. Ο αλγόριθμος αυτός αν και εγγενώς, είναι αλγόριθμος επιβλεπόμενής μάθησης χρησιμοποιείται, (σε συνδυασμό με άλλους) και σε μεθόδους μη επιβλεπομένης.

Δοθείσας μιας εισόδου x προβλέπεται η πιθανότητα $P(t = j|x),\ j = \{1,2,3, \dots \dots, k\}$ δηλαδή ένα διάνυσμα με στοιχεία την πιθανότητα να ανήκει η είσοδος σε μία εκ των k τω πλήθος κατηγοριών. Ως $\theta^{(\iota)} \in \mathbb{R}^n$ ορίζονται οι παράμετροι του δικτύου (βάρη και σταθερές πόλωσής). Η υπόθεση αυτή παράγει ως έξοδο ένα k-διαστατό κανονικοποιημένο διάνυσμα (του οποίου τα στοιχεία αθροίζονται στο 1) της μορφής:

$$F_{w,b}(x) = \begin{bmatrix} P(t=1|x;\theta) \\ P(t=2|x;\theta) \\ \vdots \\ P(t=k|x;\theta) \end{bmatrix} = \frac{1}{\sum_{j=1}^{k} e^{\theta^{(j)^\tau} x}} \begin{bmatrix} e^{\theta^{(1)^\tau} x} \\ e^{\theta^{(2)^\tau} x} \\ \vdots \\ e^{\theta^{(K)^\tau} x} \end{bmatrix}$$

Συχνά στην Softmax γίνεται χρήση της παρακάτω συνάντησης:

$$C(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=1}^{k} 1\{t^{(i)} = j\} log \frac{e^{\theta^{(j)^\tau} x^{(i)}}}{\sum_{k}^{K} e^{\theta^{(k)^\tau} x^{(i)}}}\right]$$

- $1\{y^{(i)} = j\}$: είναι συνάρτηση δείκτης, η οποία λαμβάνει την τιμή ένα όταν η παράσταση εντός των αγκύλων είναι αληθείς και την τιμή 0 σε περίπτωση που δεν είναι.
- $\frac{e^{\theta^{(j)^\tau} x^{(i)}}}{\sum_{k}^{K} e^{\theta^{(k)^\tau} x^{(i)}}} = P\left(t^{(\iota)} = j \middle| x^{(\iota)}; \theta\right)$. Δηλαδή την πιθανότητα η είσοδος $x^{(\iota)}$ να ανήκει στην κλάση j για δεδομένο $\theta = (w,b) = \begin{bmatrix} (w,b)_1^\tau \\ (w,b)_2^\tau \\ \vdots \\ (w,b)_k^\tau \end{bmatrix}$

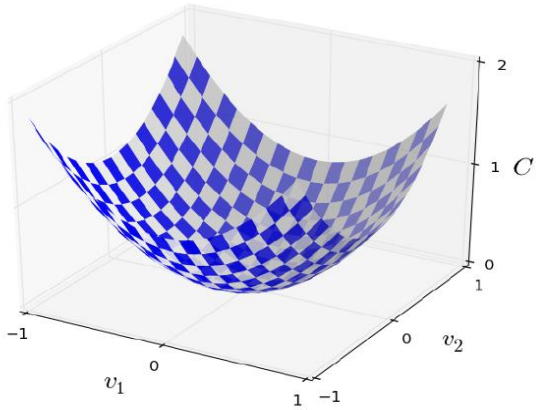
1.4.1.2 Αλγόριθμοι βελτιστοποίησης

Σκοπός των αλγορίθμων βελτιστοποίησης είναι, εμμέσου διαδικασίας πολλαπλών επαναλήψεων (iterations), να βρουν μια καθολικά (για όλο το δίκτυο) βέλτιστη συνάρτηση. Θα δοθούν δύο αλγόριθμοι οι οποίοι χρησιμοποιούνται εκτενώς στην βαθιά μάθηση τόσο στην ακαριαία μορφή τους όσο και σε παραλλαγές τους.

1.4.1.2.1 Απότομη κάθοδος (gradient descent)

Ο αλγόριθμος απότομης καθόδου (gradient descent) [21] είναι ένας από τους πιο δημοφιλείς αλγορίθμους βελτιστοποίησης στην επιβλεπόμενη μάθηση. Συνοπτικά, προκειμένου να βρεθεί το τοπικό ελάχιστο μιας διαφορίσιμης συνάρτησης κόστους γίνονται μικρά βήματα σε κατεύθυνση αντίθετη αυτής που «δείχνει» η συνάρτηση κλίσης σε συγκεκριμένο σημείο a.

Έστω ότι έχουμε ένα πρόβλημα εκμάθησης με 2 παραμέτρους. Θα γίνει χρήση της τετραδικής συνάρτησης κόστους $C_{quadr}(v_1, v_2)$. Η συνάρτηση στο επίπεδο θα έχει την εξής μορφή.



Διάγραμμα 1.5: Σχέση δύο παραμέτρων και της Τετραδικής Συνάρτησης Κόστους τους [21]

Σκοπός είναι να βρεθεί το καθολικό ελάχιστο της συνάρτησης. Για να γίνει αυτό υπολογίζεται υπολογιστεί μια τιμή ΔC ώστε η τιμή της συνάρτησης να μετακινηθεί προς το αυτό. Αλγεβρικά η μεταβολή της C εκφράζεται ως:

$$\Delta C \approx \frac{\vartheta C}{\theta u_1}\Delta u_1 + \frac{\vartheta C}{\theta u_2}\Delta u_2 \approx \nabla C * \Delta u$$

όπου $(u1, u2)$ το σημείο στον χώρο που βρίσκεται η συνάρτηση για της συγκεκριμένες τιμές και $\Delta u = (\Delta u_1, \Delta u_2)^\tau$ το διάνυσμα για το οποίο θα μετατοπιστεί η συνάρτηση. Προκειμένου να διασφαλιστεί ότι το ΔC είναι αρνητικό και βέλτιστο, η μετατόπιση θα είναι ίση με:

$$\Delta u = -\eta * \nabla C$$

Όπου η είναι μια, προκαθορισμένη εξ αρχής, θετική υπερπαράμετρος γνωστή ως ρυθμός εκμάθησης (learning rate). Σε περίπτωση που κατά την εκπαίδευση παρατηρηθεί αυξάνον ρυθμός σφάλματος ενδείκνυται η μείωση της τιμής του η. Όσο πιο μικρή τιμή δοθεί στον ρυθμό εκμάθησης τόσο μεγαλύτερη ακρίβεια θα επιτυγχάνεται αλλά σε περισσότερο χρόνο. Η επιλογή του η συνεπώς πρέπει να πετύχει ισορροπία μεταξύ απόδοσης και χρονικού κόστους. Πρακτικά ο όρος αυτός μπορεί να αλλάζει τιμή κατά την εκπαιδευτική

διαδικασία. Για παράδειγμα έχει δειχτεί ότι η μείωση της τιμής του προς το τέλος μιας εποχής μειώνει τυχαίες διακυμάνσεις στην μείωση του λάθους.

Προκειμένου να διασφαλιστεί ότι το ΔC θα είναι πάντα αρνητικό και άρα κάθε μετατόπιση θα γίνεται προς το καθολικό ελάχιστο της συνάρτησης του κόστους το ΔC γίνεται:

$$\Delta C \approx -\eta * \|\nabla C\|^2$$

Στην εφαρμογή της κλίσης καθόδου σε τεχνητά νευρωνικά δίκτυα, για κάθε είσοδο $x^{(i)}$ πρέπει να υπολογιστεί η συνάρτηση κλίσης $\nabla C^{(i)}$ η οποία με την σειρά της απαιτεί υπολογισμό των μερικών παραγώγων $\left(\frac{\vartheta C}{\vartheta w_k}\right)^{(i)}$ και $\left(\frac{\vartheta C}{\vartheta b_k}\right)^{(i)}$ (των μερικών παραγώγων του κόστους ώς προς τα συναπτικά βάρη και ώς προς τις σταθερές πόλωσης αντίστοιχα). Αφου υπολογιστούν αυτές οι τιμές, για κάθε είσοδο, υπολογίζεται ο μέσος όρος των συναρτήσεων κλίσης $\nabla C = \frac{1}{n}\sum_i \nabla C_i$ και στην συνέχεια πρέπει να ενημερωθούν τα βάρη και οι σταθερές πόλωσης του δικτύου και να προκύψουν νέες βελτιωμένες τιμές w' και b' αντίστοιχά. Η ενημέρωση αυτή γίνεται ακολουθώντας τους παρακάτω κανόνες:

$$w_k \rightarrow {w'}_k = w_k - \Delta w_k = w_k - \eta * \frac{\vartheta C}{\vartheta w_k} = w_k - \eta * \frac{1}{n}\sum_i \nabla_{\mathrm{w}} C\left(F_{w,x}\left(x^{(i)}\right), t_i\right)$$

$$b_k \rightarrow b'_k = b_k - \eta * \frac{\vartheta C}{\vartheta b_k}$$

Ο αριθμός των εισόδων που θα χρησιμοποιηθούν για τον συνυπολογισμό της μέσης κλίσης ∇C ονομάζεται batch (παρτίδα).

Η μέθοδος της απότομης καθόδου δεν ενδείκνυται για μεγάλες, βαθιές αρχιτεκτονικές καθώς τέτοια δίκτυα έχουν πολλούς νευρώνες και άρα ακόμα περισσότερες παραμέτρους προς εκπαίδευση. Για τον λόγο αυτό χρησιμοποιείται μια παραλλαγή του αλγορίθμου αυτού, η στοχαστική κλίση καθόδου (stochastic gradient Descent) όντας υπολογιστικά φτηνότερη.

1.4.1.2.2 Στοχαστική Απότομη κάθοδος (stochastic gradient descent SGD)

1.4.1.2.2.1Γενικά

Ο αλγόριθμος Στοχαστικής Απότομης καθόδου (SGD) [21] λειτουργεί σύμφωνα με τις ίδιες αρχές που λειτουργεί και η απλή κλίση καθόδου με την διαφορά ότι, προχωρά πιο

γρήγορα, εκτιμώντας την μέση συνάρτηση κλίσης ∇C υπολογίζοντας τα ∇C_x για ένα μικρό δείγμα τυχαία επιλεγμένων εισόδων x της παρτίδας εισόδου και άρα εξού και η στοχαστικότατα. Οι τυχαία επιλεγμένες είσοδοι είναι οι $X_1, X_2, \dots\dots, X_m$ σχηματίζουν μια υπο-παρτίδα (mini batch) και θα χρησιμοποιηθούν για τον υπολογισμό του διανύσματος κλίσης. Αν χρησιμοποιηθεί υπό-παρτίδα επαρκούς μεγέθους αναμένεται το διάνυσμα κλίσης να είναι σχεδόν ίσο με αυτό που θα προέκυπτε με την συνεισφορά της πλήρους παρτίδας.

$$\nabla C = \frac{1}{n}\sum_i \nabla C_i \approx \frac{1}{m}\sum_i \nabla C_i$$

Άρα ο κανόνας ανανέωσής των βαρών ενός δίκτυού γίνεται:

$$w_{k+1} = w_k - \eta * \frac{1}{m}\sum_i \nabla_{\mathrm{w}}\, C\left(F_{w,x}\left(x^{(i)}\right), y_i\right)$$

1.4.1.2.2.2 Ορμή (momentum)

Προκειμένου να επιτευχθεί μεγαλύτερη ταχύτητα κατά την εκπαίδευση ενός δικτύου με αλγόριθμο απότομης καθόδου (και παραλλαγών του) είναι δυνατή η εισαγωγή ενός περαιτέρω όρου γνωστού και ως ορμή p (momentum) [24]. Με την προσθήκη του όρου το βάρος w_k θα ενημερωθεί ως εξής:

$$w_{k+1} = \boldsymbol{p} * w_k - \eta * \frac{1}{m}\sum_i \nabla_{\mathrm{w}}\, C\left(F_{w,x}\left(x^{(i)}\right), y_i\right)$$

Διαισθητικά η λογική πίσω από την χρήση ενός τέτοιου όρου είναι ότι η κάθοδος είναι ιδιαίτερα αργή σε σημεία στα οποία η συνάρτηση κόστους παρουσιάζει μακρόστενες πεδιάδες με αποτέλεσμα το σύστημα να ταλαντεύεται μπρος και πίσω στην κατεύθυνση του στενού άξονα και να κινείται ελάχιστα στην κατεύθυνση του μακρού όπου είναι και το ζητούμενου. Η ορμή λοιπόν, βοηθάει το σύστημα να ξεφύγει από αυτήν την ταλάντωση καθώς και συνεισφέρει στην κίνηση στον μακρύ άξονα επιταχύνοντας την κάθοδο. [22] Με την εισαγωγή της ορμής είναι αρκετά πιθανό ο ρυθμός εκμάθησης να χρειαστεί να μειωθεί εφόσον η βαρύτητα της κλίσης στην εκπαίδευση μεγαλώνει. Η τιμή της ορμής τίθεται αρχικά στο 0,5 και σε μετέπειτά εποχές όταν η μάθηση έχει αρχίσει και σταθεροποιείται αυξάνεται σε τιμές μεγαλύτερες η ίσες του 0,9.

1.4.2 Μη επιβλεπόμενη μάθηση

“Self-education is, I firmly believe, the only kind of education there is.”

— Isaac Asimov

Τα βαθιά δίκτυα που χρησιμοποιούν μη επιβλεπόμενη μάθηση στοχεύουν στον προσδιορισμό φυσικών ομάδων ή όμοιων χαρακτηριστικών σε ένα δεδομένο σύνολο προτύπων όταν δεν υπάρχει κάποιο πληροφορία για την επιθυμητή έξοδο. Αφού αυτά τα δίκτυα χρησιμοποιούν μόνο πρότυπα εισόδου ο τρόπος εκπαίδευσης τους αναφέρεται σαν αλγόριθμος μάθησης χωρίς επίβλεψη Ο χρήστης ενός δικτύου με μάθηση χωρίς επίβλεψη πρέπει να εξετάσει το αποτέλεσμα του δικτύου προκειμένου να ελέγξει αν η ταξινόμηση έχει πρακτική σημασία [25]. Αν δεν είναι ικανοποιητική τότε γίνεται επαναρύθμιση παραμέτρων εκπαίδευσης και το δίκτυο επανεκπαιδεύεται.

Σκοπός της μάθησης δεν είναι πλέον η υλοποίηση κάποιας απεικόνισης από τα δεδομένα εισόδου στα δεδομένα εισόδου, αλλά η αυτό-οργάνωση και η ανακάλυψη διάφορων χαρακτηριστικών ιδιοτήτων των εισαγόμενων δεδομένων. Χαρακτηριστικά προβλήματα που εμπίπτουν στην κατηγορία της μάθησης χωρίς επίβλεψη είναι:

- Ομαδοποίηση (Clustering)
- Ανάλυση βασικών συνιστωσών (Principal Component analysis)
- Μείωση της διάστασης των δεδομένων (προβολή τους σε χώρο μικρότερης διάστασης από τον αρχικό) (dimensionality reduction).

Για όλες τις παραπάνω κατηγορίες προβλημάτων έχουν αναπτυχθεί μοντέλα νευρωνικών δικτύων και αντίστοιχες τεχνικές εκπαίδευσης. Για παράδειγμα στην περίπτωση της ανάλυσης βασικών συνιστωσών έχουν αναπτυχθεί τεχνικές βασισμένες στη μάθηση Hebb (Hebbian Learning. Εναλλακτικά η ανάλυση των βασικών συνιστωσών μπορεί να γίνει με κλασική άλγεβρα.

Οι μη-επιβλεπόμενες αρχιτεκτονικές αποτελούν ένα σημαντικό και ταχέως αναπτυσσόμενο πεδίο των νευρωνικών δικτύων [26]. Κατέχουν εξέχουσα σημασία στην ανάπτυξη αλγορίθμων εκμάθησης για βαθιές αρχιτεκτονικές που στοχεύουν στην Τεχνίτη Νοημοσύνη. Κάποιοι από τους λόγους που συντελούν στην σημασία τους είναι :

- Υπάρχει μεγάλη πληθώρα παραδειγμάτων χωρίς ετικέτες ενώ αντιθέτως παραδείγματα με ετικέτες παραμένουν περιορισμένα.
- Άγνωστες μελλοντικές εργασίες: Αν το δίκτυο δεν γνωρίζει τι εργασίες θα πρέπει να αντιμετωπίσει στο μέλλων αλλά γνωρίζει τον χώρο στον οποίο θα ανήκουν αυτές (π.χ. τυχαίες μεταβλητές) είναι πολύ πιο λογικό να συλλέξει και συναναστραφεί με όσο το δυνατών περισσότερες πληροφορίες από τον χώρο αυτών.
- Όταν μία έχει εκμαθευτεί μια καλή αναπαράσταση υψηλού επιπέδου τότε οι εργασίες εκμάθησης (π.χ. επιβλεπόμενης) θα μπορούσαν να γίνουν πολύ πιο εύκολες.

- Δυνατότητα εφαρμογής μη επιβλεπόμενης μάθησης σε επίπεδο στρωμάτων. Δηλαδή μεγάλο μέρος της διαδικασίας μπορεί να πραγματοποιηθεί χωρίς να είναι απαραίτητη η μεταφορά τιμών εμμέσων μεγάλων διάδρομων μέσα στο δίκτυο (π.χ. μεταφορά κλίσης με χρήση οπισθοδρομικής μετάδοσης σφάλματος).

Πολλά από τα δίκτυα που ανήκουν σε αυτή την κατηγορία [12] μπορούν να χρησιμοποιηθούν για την παραγωγή δειγμάτων εμμέσου δειγματοληψίας από άλλα δίκτυα και για αυτό τον λόγο ονομάζονται Παραγωγικά Μοντέλα (Generative Models). Παραδείγματα Παραγωγικών μοντέλων είναι τα Restricted Boltzmann Machines, Deep Belief Networks, Deep Boltzmann Machines και η γενική μορφή των Denoising Autoencoders. Παρόλα αυτά σε κάποια από τα δίκτυα μη επιβλεπόμενης μάθησης η δειγματοληψία είναι δύσκολη και έτσι δεν είναι παραγωγικής φύσης (π.χ. οι Deep Autoencoders στην γνήσια μορφή τους). Η πιο κοινή μορφή των μη-επιβλεπόμενων βαθιών δικτύων είναι τα ενεργειακά βαθιά μοντέλα (energy-based) τα οποία προσπαθούν να μάθουν ελαχιστοποιώντας μια συνάρτηση ενέργειας. Τα πιο γνωστά απ' αυτά τα Boltzmann Machines τα οποία αποτελούν έμπνευση πολλά άλλα δίκτυα όπως Restricted Boltzmann Machines και Deep Belief Networks.

Εξαιτίας του χαοτικού πλήθους τεχνικών μη-επιβλεπόμενης μάθησής αλλά και τις πολυπλοκότητας τους, έχει επιλεγεί να γίνει παρουσίαση των κυριότερων τεχνικών μη-επιβλεπόμενη μάθησης εμμέσων της παρουσίασης των αντίστοιχων νευρωνικών δικτύων που τις υλοποιούν. Παρόλα αυτά στην παρούσα φάση θα γίνει μια σύντομη αναφορά στα ενεργειακά μοντέλα λόγο της εξέχουσας σημασίας που κατέχουν.

1.4.2.1 Ενεργειακά Μοντέλα

Ο βασικός σκοπός των στατιστικών μοντέλων και της μηχανικής μάθησης είναι η κωδικοποίηση εξαρτήσεων μεταξύ μεταβλητών [27]. Ένα μοντέλο λοιπόν απαθανατίζοντας τέτοιες εξαρτίσεις μπορεί να απαντήσει σε ερωτήσεις που αφορούν τιμές άγνωστων μεταβλητών χρησιμοποιώντας τιμές μεταβλητών ήδη γνωστών σε αυτό. Τα ενεργειακά μοντέλα (Energy Based Models EBMs) απαθανατίζουν εξαρτήσεις μεταξύ μεταβλητών συσχετίζοντας μία βαθμοτή ενέργεια με κάθε διαμόρφωση μεταβλητών. Η διαδικασία που ακολουθούν όταν καλούνται να κάνουν μια πρόβλεψη ή να λάβουν μια απόφαση περιλαμβάνει την εκχώρηση τιμών σε μεταβλητές που έχουν παρατηρήσει και την εύρεση των τιμών των υπολειπόμενων μεταβλητών. Η εκμάθηση αποτελείται από την εύρεση μιας ενεργειακής συνάρτησης η οποία συσχετίζει χαμηλές ενέργειες με σωστές τιμές των εναπομενόντων μεταβλητών και υψηλές ενεργείς με λανθασμένες τιμές. Μια λειτουργικότητα απώλειας, η οποία ελαχιστοποιείται κατά την διάρκεια της εκπαίδευσης, χρησιμοποιείται για να μετρήσει την ποιότητα των διαθέσιμων ενεργειακών συναρτήσεων.

1.4.3 Αλγόριθμοι Εκμάθησης

Μετά από ένα εκπαιδευτικό πέρασμα, δηλαδή μετά από μία προς τα εμπρός διάδοση ενός, ή μιας ομάδας εκπαιδευτικών δειγμάτων, το δίκτυο πρέπει με κάποιο τρόπο να ανανεώσει

τα συναπτικά βάρη και τις σταθερές πόλωσης των νευρώνων του προκειμένου να βελτιώσει την απόδοση του αναφορικά με τον σκοπό του. Με αυτό τον σκοπό γίνεται η χρήση των αλγορίθμων εκμάθησης. Ο πιο γνωστός εξ αυτών παραμένει ο αλγόριθμος οπισθοδρομικής διάδοσης σφάλματος.

Ο αλγόριθμος αυτός αποτελεί θεμελιώδη λίθο της σύγχρονής βαθιάς μάθησης και άρα τα περισσότερα αν όχι όλα τα σύγχρονα δίκτυα χρησιμοποιούν αυτόν ή παραλλαγές του. Ο αλγόριθμος αυτός θα αναλυθεί ενδελεχώς, στην απλή μορφή του, στο παρόν κεφάλαιο.

1.4.3.1 Back-Propagation Algorithm (Αλγόριθμος οπισθοδρομικής διάδοσης σφάλματός)

Ο αλγόριθμος της οπισθοδρομικής διάδοσης σφάλματος (ΟΔΣ) [25] καθιερώθηκε το 1986 στην περίφημη εργασία των D. Rummelhart, G. Hinton και R. Williams και είναι μακράν ο πιο διαδεδομένος αλγόριθμος εκμάθησης. Η ιδέα, περιγράφηκε για πρώτη φορά από τον Werbos στην διδακτορική του διατριβή [28] το 1974. Την περίοδο 1985-1986 χρησιμοποιήθηκε εκτενώς από μεγάλα ονόματα στον τομέα των νευρωνικών δικτύων. Ονόματα που περιλαμβάνουν τούς Rummelhart, Hinton, Williams, McClelland, Parker , LeCun. Η ανάπτυξη του αλγορίθμου ΟΔΣ αποτελεί ορόσημο στην εξέλιξη των νευρωνικών δικτύων, διότι παρέχει μία υπολογιστικά αποτελεσματική μέθοδο εκπαίδευσης των πολυστρωματικών νευρωνικών δικτύων. Παρά την ζωογόνο πνοή που έδωσε ο αλγόριθμος στις βαθιές αρχιτεκτονικές, ο ισχυρισμός ότι ο αλγόριθμος μπορεί να επιλύσει όλα τα προβλήματα που παρουσιάζονται στην βαθιά μάθησης παραμένει αναληθείς.

Στον πυρήνα του αλγορίθμου οπισθοδρομικής διάδοσης σφάλματος βρίσκεται μια έκφραση μερικών παραγώγων $\frac{\vartheta C}{\vartheta w}$, δηλαδή των μερικών παραγώγων της συνάρτησης του κόστους C αναφορικά με κάθε συναπτικό βάρος και σταθεράς πόλωσης του δικτύου [21]. Η έκφραση αυτή περιγράφει το πόσο γρήγορα το κόστος μεταβάλλεται όταν γίνεται προσαρμογή των βαρών και των σταθερών πόλωσης. Ο υπολογισμός αυτών των μερικών παραγώγων αποτελεί άλλωστε και στόχο του αλγορίθμου. Προκειμένου να λειτουργήσει ο αλγόριθμος πρέπει να γίνουν οι εξής υποθέσεις για την συνάρτηση του κόστους που θα χρησιμοποιηθεί.

- Πρώτων, η συνάρτηση του κόστους πρέπει να μπορεί να γραφεί ως ένας μέσος όρος των συναρτήσεων κόστους όλων τον εισόδων x του δικτύου. Αυτό οφείλεται στο γεγονός ότι ο αλγόριθμος επιτρέπει τον υπολογισμό των μερικών παραγώγων $\frac{\vartheta C_x}{\vartheta w}$, $\frac{\vartheta C_x}{\vartheta b}$ της εκάστοτε εισόδου x και άρα για τον εξεύρεση των ολικών $\frac{\vartheta C_x}{\vartheta w}$, $\frac{\vartheta C_x}{\vartheta b}$ απαιτείται υπολογισμός του μέσου όρου τους. Για παράδειγμα, στην περίπτωση της τετραδικής συνάρτησης κόστους ο μέσος όρος επί n τω πλήθος δειγμάτων εισόδου είναι $C = \frac{1}{n}\sum C_x$ όπου $C_x = \frac{1}{2}\left\|F_{w,b}(x^i) - t^i\right\|^2$.
- Δεύτερον, το κόστος πρέπει να μπορεί να γραφεί συναρτήσει των εξόδων του νευρωνικού δικτύου κάτι το οποίο προφανώς ισχύει και για την τετραδική συνάρτηση κόστους.

1.4.3.1.1 Λεπτομέρειες

Αναφορικά τώρα με τον υπολογισμό των μερικών παραγώγων $\frac{\vartheta C}{\vartheta w_{jk}^{l}}$, $\frac{\vartheta C}{\vartheta b_{j}^{l}}$ (όπου w_{jk}^{l} συμβολίζει το k-οστό βάρος του νευρώνα j που βρίσκεται στο στρώμα l, και το b_j^l είναι αντιστοίχως η σταθερά πόλωσης του νευρώνα j του στρώματος l) εισάγεται μια διαμεσολαβητική ποσότητα επονομαζόμενη ως σφάλμα, δ_j^l από τον j-οστό νευρώνα του l-οστού στρώματος. Ο αλγόριθμος παρέχει μέθοδο για τον υπολογισμό του σφάλματος , τιμή από την οποία στην συνέχεια θα προκύψουν τα $\frac{\vartheta C}{\vartheta w_{jk}^{l}}$, $\frac{\vartheta C}{\vartheta b_{j}^{l}}$.

Κατά την εκπαίδευση ενός δικτύου λοιπόν, εισάγονται μικρές αλλαγές Δz_j^l στην βεβαρημένη είσοδο z του νευρώνα j του l-οστού στρώματος. Με κάθε τέτοια αλλαγή η έξοδος του νευρώνα μετατρέπεται, από $f\left(z_j^l\right)$ (όπου $f(\dots)$ η συνάρτηση ενεργοποίησης του νευρώνα) σε $f\left(z_j^l + \Delta z_j^l\right)$. Η αλλαγή στην συνέχεια μεταδίδεται διαμέσου των μετέπειτα στρωμάτων και προκαλεί αλλαγή του συνολικού κόστους κατά $\frac{\vartheta C}{\vartheta z_j^l}\Delta z_j^l$. Κατά την εκπαίδευση λοιπόν αναζητείται μια τιμή Δz_j^l τέτοια ώστε η συνάρτηση κόστους να ελαχιστοποιηθεί όσο το δυνατών περισσότερο. Προκειμένου λοιπόν να επιτευχθεί μέγιστη ελάττωση του κόστους το Δz_j^l θα λαμβάνει κατάλληλη τιμή. Για παράδειγμα, αν στην καλύτερη των περιπτώσεων, το $\frac{\vartheta C}{\vartheta z_j^l}$ έχει λάβει μεγάλη τιμή (αρνητική ή θετική) το Δz_j^l θα λάβει πρόσημο αντίθετο από αυτό του $\frac{\vartheta C}{\vartheta z_j^l}$. Αντιθέτως το $\frac{\vartheta C}{\vartheta z_j^l}$ λάβει τιμή κοντά στο μηδέν, ότι τιμή και να λάβει το Δz_j^l θα επιτευχθεί μικρή ελάττωση του κόστος. Το σφάλμα του νευρώνα j του l-οστού στρώματος ορίζεται λοιπόν ως:

$$\delta_j^l = \frac{\vartheta C}{\vartheta z_j^l}$$

Πριν δοθεί ο αλγόριθμος της ΟΔΣ αυτός κάθε αυτός, πρέπει πρώτα να γίνουν κατανοητές οι τέσσερις θεμελιακές εξισώσεις στις οποίες βασίζεται η λειτουργεία του. Συνδυασμένες αυτές οι εξισώσεις παρέχουν τρόπους για τον υπολογισμό τόσο του σφάλματος δ_j^l όσο και το διανύσματος κλίσης $\frac{\vartheta C}{\vartheta w}$. Ακολουθούν οι τέσσερις αυτές συναρτήσεις.

- Εξίσωση υπολογισμού σφάλματος στρώματος εξόδου:

$$\delta_j^L = \frac{\vartheta C}{\vartheta y_j^l} f'(z_j^L) \qquad \text{BP (1)}$$

  - Η σήμανση L συμβολίζει το τελευταίο στρώμα του δικτύου (στρώμα εξόδου)
  - Ο όρος $\frac{\vartheta C}{\vartheta y_j^L}$ μετράει το πόσο γρήγορα μεταβάλλεται το κόστος συναρτήσει της j-οστής εξόδου ενεργοποίησης. Αν, παραδείγματος χάριν το C δεν εξαρτάται

ιδιαιτέρως από την έξοδο του j-οστοού νευρώνα, τότε η τιμή του δ_j^L θα είναι μικρή.

- Ο όρος $f'(z_j^L)$ δηλαδή η παράγωγος της συνάρτησης ενεργοποίησης εξετάζει πόσο γρήγορα μεταβάλλεται η $f(\dots)$ αναφορικά με το z_j^L

Να σημειωθεί ότι η εν λόγω συνάρτηση είναι εύκολα υπολογίσιμη, καθώς, μετά τον υπολογισμό του z_j^L ο υπολογισμός του $f'(z_j^L)$ γίνεται με ελάχιστο παραπανίσιο κόστος. Επίσης ο υπολογισμός του $\frac{\vartheta C}{\vartheta y_j^L}$ για τις περισσότερες συναρτήσεις κόστους είναι αρκετά φτηνός, παραδείγματος χάριν για $C_{quadr_x} = \frac{1}{2}\sum_j (y-t)^2$ η μερική παράγωγος $\frac{\vartheta C_{quadr}}{\vartheta {y'}_j^L} = \sum_j (y-t)$ είναι εύκολα υπολογίσιμη. Η συνάρτηση αυτή προκειμένου να έχει πρακτική εφαρμογή για τον αλγόριθμο πρέπει υλοποιηθεί με την χρήση πινάκων:

$$\delta_j^L = \nabla_y C \odot f'(z^L)$$

- Ο όρος $\nabla_y C$ είναι πίνακας με στοιχεία τα μερικές παραγώγους $\frac{\vartheta C}{\vartheta y_j^L}$
- Η πράξη $\odot$ πρόκειτα για το Γινόμενο Hadamard $\left[A \odot B = \left(c_{ij}\right) \acute{o}\pi o\upsilon\ c_{ij} = a_{ij} * b_{ij}, \gamma\iota\alpha\ \forall\ 1 \le i \le m,\ 1 \le j \le n\right]$

Για παράδειγμα η συνάρτηση στην περίπτωση της συνάρτησης τετραδικού κόστους C_{quadr}, λαμβάνει την μορφή: $\delta_j^l = ({y'}^L - y) \odot f'(z^L)$

- Συνάρτηση του σφάλματος δ^l αναφορικά με το σφάλμα του επόμενου στρώματος δ^{l+1}

$$\delta^l = ((w^{l+1})^T \delta^{l+1}) \odot f'(z^l) \qquad \text{BP (2)}$$

- Ο όρος $(w^{l+1})^\tau$ είναι ο ανάστροφος του πίνακα βαρών (w^{l+1}) του στρώματος $l+1$ του δικτύου.

Συνδυάζοντας λοιπόν τις συναρτήσεις BP(1) και BP(2) είναι δυνατών να υπολογιστεί το σφάλμα δ^l για κάθε στρώμα του δικτύου. Χρησιμοποιώντας αρχικά την BP(1) για τον υπολογισμό του σφάλματος δ^L και στην συνέχεια χρησιμοποιώντας επανειλημμένως την δεύτερη υπολογίζονται τα $\delta^{L-1}, \delta^{L-2} \dots \dots, \delta^0$ δηλαδή τα διανύσματα σφάλματος του κάθε στρώματος $l \in \{0, L\}$.

- Συνάρτηση για τον ρυθμό μεταβολής του κόστους συναρτήσει οποιασδήποτε σταθεράς πόλωσης του δικτύου.

$$\frac{\vartheta C}{\vartheta b_j^l} = \delta_j^l \qquad \text{BP (3)}$$

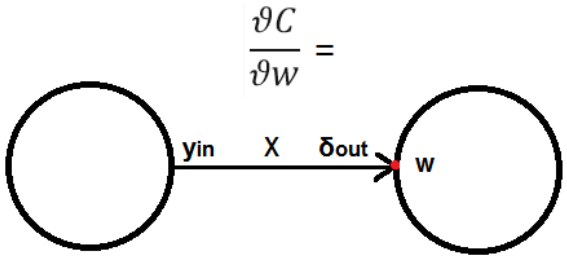
- Συνάρτηση για τον ρυθμό μεταβολής του κόστους συναρτήσει οποιουδήποτε συναπτικού βάρους του δικτύου.

$$\frac{\vartheta C}{\vartheta w_{jk}^l} = y_k^{l-1}\delta_j^l \qquad \text{BP (4)}$$

Η συνάρτηση αυτή προσδιορίζει τον τρόπο με τον οποίο καθίσταται δυνατών να αποφευχθεί ο υπολογισμός των μερικών παραγώγων $\frac{\vartheta C}{\vartheta w_{jk}^l}$. Ο υπολογισμός τους θα επιτευχθεί με την χρήση των τιμών τα y_k^{l-1} και δ_j^l που είναι ήδη γνωστό το πώς υπολογίζονται. Η συνάρτηση αυτή μπορεί να ξαναγραφεί με λιγότερες σημάνσεις ως:

$$\frac{\vartheta C}{\vartheta w} = y_{in}\delta_{out}$$

Όπου y_{in} είναι η έξοδος των προηγούμενων νευρώνων οι οποίες αποτελούν είσοδο του εν λόγω νευρώνα στην είσοδο με βάρος w και δ_{out} είναι το σφάλμα της εξόδου του νευρώνα για το συγκεκριμένο βάρος w. Μεγεθύνοντας λοιπόν σε δύο νευρώνες που συνδέονται με βεβαρημένη σύνδεση βάρους w, η εξίσωση BP(4) μπορεί να παρασταθεί γραφικά όπως στην Εικόνα 1.4.



Εικόνα 1.11: Γραφική Αναπαράσταση της τέταρτης εξίσωση [BP (4)] του Αλγορίθμου ΟΔΣ

1.4.3.1.2 Αλγόριθμος

- Είσοδος x: Βρες την έξοδο ενεργοποίησης y_1 του στρώματος εισόδου.
- Εμπροσθοδρομηση (Feedforward) : Για ∀ $l = 2, 3, \dots, L$ υπολόγισε $z^l = w^l y^{l-1} + b^l$ και $y = f(z^l)$.
- Σφάλμα στρώματος εξόδου δ^L: Υπολόγισε το διάνυσμα $\delta^L = \nabla_y C \odot f'(z^L)$

- Οπισθοδρόμηση του σφάλματος: Για $\forall\, l = L-1, L-2, \dots, 2$ υπολόγισε $\delta^l = ((w^{l+1})^\tau \delta^{l+1}) \odot f'(z^l)$
- Έξοδος: Βρες το κλίση της συναρτήσεως κλίσης υπολογίζοντας $\frac{\vartheta C}{\vartheta b_j^l} = \delta_j^l$ και $\frac{\vartheta C}{\vartheta w_{jk}^l} = y_k^{l-1}\delta_j^l$ για συναπτικά βάρη και σταθερές πόλωσης αντίστοιχα.

Στην πράξη ο αλγόριθμος οπισθοδρομικής διάδοσης συνδυάζεται με έναν αλγόριθμο εκμάθησης ή αλγόριθμο βελτιστοποίησης (π.χ. στοχαστικής απότομης καθόδου). Ο αλγόριθμος εκμάθησης χρησιμοποιείται προκειμένου να υπολογιστούν τα διανύσματα κλίσης για πολλαπλές εισόδους εκπαίδευσης. Συγκεκριμένα, δοθείσας μία υπό-παρτίδας m εισόδων εκπαίδευσης και με χρήση στοχαστικής καθόδου με ρυθμό εκμάθησης η, ο αλγόριθμος οπισθοδρομικής διάδοσης σφάλματος γίνεται:

- Εισαγωγή υπό-παρτίδας εισόδων εκπαίδευσης.
- Για κάθε είσοδο x της δοθείσας υπό-παρτίδας:
  - Για $\forall\, l = 2, 3, \dots, L$ υπολόγισε $z^{x,l} = w^l y^{x,l-1} + b^l$ και $y^{x,l} = f(z^{x,l})$.
  - Σφάλμα στρώματος εξόδου $\delta^{x,L}$: Υπολόγισε το διάνυσμα $\delta^{x,L} = \nabla_y C \odot f'(z^{x,L})$
  - Οπισθοδρόμηση του σφάλματος: Για $\forall\, l = L-1, L-2, \dots, 2$ υπολόγισε $\delta^{x,l} = ((w^{l+1})^\tau \delta^{x,l+1}) \odot f'(z^{x,l})$
- Απότομη Κάθοδος: Για $\forall\, l = L-1, L-2, \dots, 2$ ενημέρωσε τα συναπτικά βάρη σύμφωνα με τον κανόνα ανανέωσης $w^l = w^l - \eta * \frac{1}{m}\sum_x \delta^{x,l}(y^{x,l-1})^\tau$ και τις σταθερές πόλωσης σύμφωνα με τον $b^l = b^l - \eta * \frac{1}{m}\delta^{x,l}$

1.4.4 Δυσκολίες κατά την εκπαίδευση

Σε αυτό το κεφάλαιο θα αναλυθούν μερικές από τις βασικές θεμελιώδεις δυσκολίες που αντιμετωπίζονται κατά την εκπαίδευση βαθιών νευρωνικών δικτύων. Οι δυσκολίες αυτές πριν αντιμετωπιστούν (έστω και μερικώς) συμβάλαν στο τέλμα στο οποίο επήλθε στον κλάδο των νευρωνικών δικτύων τις δεκαετίας τών 60’ και 70’.

1.4.4.1 Vanishing / exploding gradients

Η διπλωματική εργασία του Hochreiter το 1991 αποτέλεσε ορόσημο στην ιστορία των βαθιών αρχιτεκτονικών [21]. Όπως αναφέρθηκε και στο κεφάλαιο 1.2 τότε ήταν που για πρώτη φορά αποκαλύφθηκε ο λόγος για τον οποίον τα παραδοσιακά βαθιά δίκτυα είναι δύσκολο να εκπαιδευτούν. Ενώ διαισθητικά λοιπόν, θα περίμενε κάνεις ότι με την προσθήκη περσότερων κρυφών στρωμάτων, ένα δίκτυο θα μπορούσε να μάθει πιο περίπλοκες εξαρτήσεις και άρα βελτιωθεί η απόδοση του ή έστω να διατηρηθεί στα ίδια επίπεδα, στην πράξη συμβαίνει ακριβώς το αντίθετο. Δηλαδή το , βαθύ πλέον, δίκτυο αποκτά χειρότερη απόδοση. Το βασικό αίτιο του προβλήματος αυτού, του γνωστού και ως θεμελιώδες πρόβλημα της βαθιάς μάθησης (Fundamental Deep Learning Problem), αναγνωριστικέ ότι είναι η αστάθεια των διανυσμάτων κλίσης τα οποία κατά την διάδοση τους μετατρέπονται είτε σε εξαφανιζόμενα είτε σε εκργνυώμενα (vanishing or exploding gradients). Εξαφανιζόμενα διανύσματα κλίσης παρουσιάζονται όταν διανύσματα κλίσης ενός δικτύου κατά την διάδοση τους προς τα πίσω (μέσω οπισθοδρομικής διάδοσης

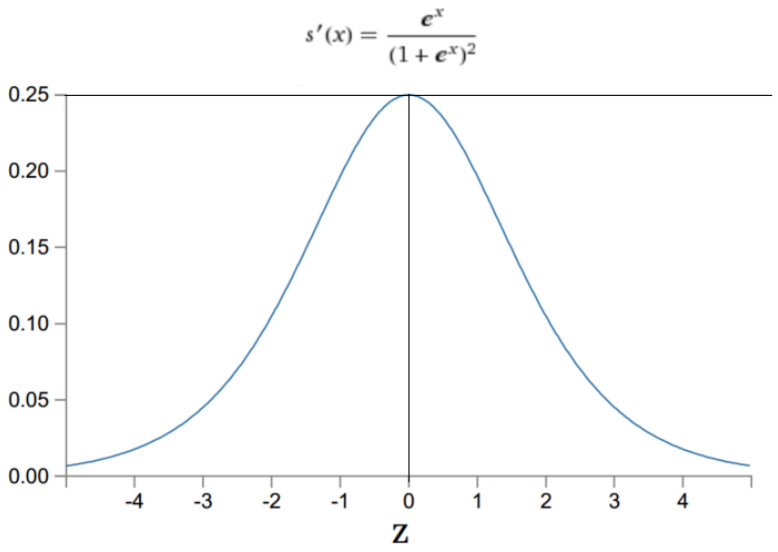
σφάλματος) αποκτούν την τάση να μικραίνουν διαρκώς με αποτέλεσμα οι νευρώνες στα χαμηλότερα στρώματα να μαθαίνουν πιο αργά από τους υπόλοιπους. Αντιθέτως εκρηγνυώμενα διανύσματα κλίσης παρουσιάζει ένα δίκτυο όταν το διάνυσμα κλίσης γίνεται δυσανάλογα μεγάλο όσο διαδίδεται προς τα κατώτερα στρώματα. Το επικρατέστερο εκ των δυο προβλημάτων είναι το vanishing gradient καθώς εμφανίζεται σε δίκτυα που χρησιμοποιούν σιγμοειδή συνάρτηση ενεργοποίησης και γι' αυτό το λόγο θα αναλυθεί εκτενέστερα παρακάτω.

Έστω ότι δίδεται δίκτυο με L κρυφά στρώματα, με απλούς σιγμοειδής νευρώνες το οποίο εκπαιδεύεται με τον αλγόριθμο οπισθοδρομικής διάδοσης σφάλματός. Όπως προαναφέρθηκε, το διάνυσμα κλίσης ενός στρώματος l δίνεται από τον τύπο της οπισθοδρομικής διάδοσης του σφάλματος. Ο τύπος που θα δοθεί, διαφέρει από αυτόν που δόθηκε στο κεφάλαιο 1.4.3.1.1.2 μόνο στο γεγονός ότι δεν χρησιμοποιούνται πίνακες για λόγους κατανόησης.

$$\delta^L = \Sigma'(z^l)(w^{l+1})\Sigma'(z^{l+1})(w^{l+2})^\tau \dots \Sigma'(z^L)\nabla_\alpha C$$

- Το α συμβολίζει τις ενεργοποιήσεις (εξόδους) νευρώνων.
- Όπου z^l είναι διάνυσμα με στοιχεία τις τιμές $z_j = w_j a_{j-1} + b_j$ (Βεβαρημένη είσοδος του νευρώνα j) για κάθε νευρώνα του στρώματος l
- Όπου w^l το διάνυσμα βαρών των νευρώνων του στρώματος l
- Όπου $\nabla_\alpha C$ είναι ο πίνακας των μερικών παραγώγων της συνάρτησης κόστους αναφορικά με τις ενεργοποιήσεις του στρώματος εξόδου.

Η τιμή $\Sigma'(z^l)$ που επαναλαμβάνεται διαρκώς είναι διαγώνιος πίνακας με στοιχεία τα $\sigma'(z^l)$ των εισόδων (μετά την επίδραση των βαρών) του στρώματος l. Λαμβάνοντας υπόψιν ότι χρησιμοποιούνται σιγμοειδείς νευρώνες $s(z) = \frac{1}{1+e^{-z}}$ η παράγωγός της συνάρτησης $s'(z) = \frac{e^x(z)}{(1+e^x)^2}$ έχει ολικό μέγιστο την τιμή $s'(0) = 0{,}25$ όπως φαίνεται και στο διάγραμμα.



Διάγραμμα 1.6: Πρώτη Παράγωγος της Σιγμοειδούς Συνάρτησης

Είναι προφανές ότι κάθε στοιχείο του πίνακα $\Sigma'(z^l)$ είναι μικρότερο ή ίσο του $\frac{1}{4}$. Εάν λοιπόν τα βάρη του δικτύου w^l δεν έχουν λάβει μεγάλες τιμές κατά την αρχικοποίηση τους τότε οι τιμές $\Sigma'(z^l)(w^{l+1})$ τείνουν να οδηγούν το διάνυσμα κλίσης προς σμίκρυνσή και άρα το δίκτυο εν τέλη θα καταλήξει σε κατάσταση εξαφανιζόμενου σφάλματος.

Η πρώτη λογική αντιμετώπιση αυτού το προβλήματος θα ήταν να αυξηθεί η τιμή των αρχικών βαρών του δικτύου. Δυστυχώς αυτή η μέθοδος μπορεί να οδηγήσει στην εξίσου ανεπιθύμητη κατάσταση των εκργνυώμενων σφαλμάτων.

Το θεμελιώδες πρόβλημα των βαθιών αρχιτεκτονικών δεν οφείλεται σε αυτό κάθε αυτό στο φαινόμενο των εξαφανιζόμενων ή των εκργνυώμενων διανυσμάτων κλίσης. Το πρόβλημα οφείλεται στο γεγονός ότι το διάνυσμα κλίσης των κατώτερων στρωμάτων είναι προϊών των διανυσμάτων κλίσης των ανώτερων. Όταν λοιπόν υπάρχουν πολλαπλά νευρωνικά στρώματα, οι πολλαπλές αυτές εξαρτήσεις οδηγούν σε εγγενώς ασταθείς καταστάσεις καθώς όλα τα στρώματα δεν μαθαίνουν με την ίδια ταχύτητα. Το πρόβλημα αυτό ονομάζεται «πρόβλημα ασταθών διανυσμάτων κλίσης» και εμπεριέχει τα προβλήματα των εκργνυώμενων και εξαφανιζόμενων διανυσμάτων κλίσης. Ο μόνος τρόπος να μάθουν με, περίπου, την ίδια ταχύτητα είναι όλα τα παράγωγα όρων του δικτύου να εξισορροπηθούν με χρήση κάποιάς συνθήκης η μηχανισμού εξισορρόπησης.

Προκειμένου να αντιμετωπιστεί (έστω και μερικώς) το θεμελιώδες αυτό πρόβλημα της βαθιάς μάθησης έχουν προταθεί αρκετές τεχνικές [13]. Μεταξύ άλλων:

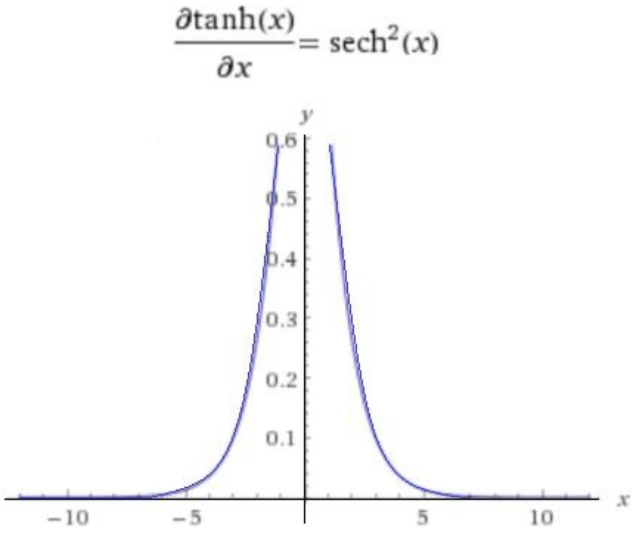
- Το 1991 προτάθηκε από τον Schmidhuber μια αρχιτεκτονική γνωστή ως «Ένας πολύ βαθύς μαθητής» [29] [30] η οποία αποτρέπει, ως ένα βαθμό, την εμφάνιση ασταθών διανυσμάτων κλίσης σε μια ιεραρχία RNN μέσου της μη-επιβλεπόμενης προ εκπαίδευσης της. Παρόμοια αποτελέσματα έχουν επιτευχθεί και σε έμπροσθεν τροφοδοτούμενα δίκτυα τα οποία προ εκπαιδεύονται με συστάδες Autoencoders ή DBNs.
- Τα δίκτυα τα οποία προσομοιάζουν την αρχιτεκτονική LSTM (LSTM-like networks) (1.5.2.4), εγγενώς, δεν επηρεάζονται από αυτό το πρόβλημα.
- Στην ανακούφιση των βαθιών αρχιτεκτονικών από αυτό το φαινόμενο έχει βοηθήσει ιδιαίτερα η δυνατότητα εκπαίδευσης νευρωνικών δικτύων σε υπολογιστικά ισχυρές GPUs. Δίνεται έτσι η δυνατότατα, τα σφάλματα να διαδίδονται πολύ πιο γρήγορα στα κατώτερα στρώματα με αποτέλεσμα. Έτσι είναι δυνατή η εύρεση ιδανικών βαρών σε πολλαπλά στρώματα εντός ενός επιθυμητού χρονικού διαστήματος.
- Χρήση Hessian-free τεχνικών βελτιστοποίησης [31].

1.4.4.2 Πρόωρος Κορεσμός (Premature Saturation)

Γενικά [30], η μικρή ταχύτητα σύγκλισής του αλγορίθμου οπισθοδρομικής διάδοσης σφάλματος κατά την εκπαίδευση έμπροσθεν διαδιδόμενων, πολυστρωματικών, νευρωνικών δικτύων αποδίδεται συνήθως στο γεγονός ότι ο αλγόριθμος είναι βασισμένος στην μέθοδο της απότομης καθόδου (gradient descent). Ένα επιπλέον αίτιο στο οποίο μπορεί να αποδοθεί αυτή η ανεπιθύμητα αργή σύγκλιση του αλγορίθμου οπισθοδρομικής διάδοσης σφάλματος είναι η εμφάνιση του φαινομένου του πρόωρου κορεσμού των σιγμοειδών νευρώνων εξόδου ενός δικτύου. Αυτό το φαινόμενο μερικές φορές αναφέρεται στην βιβλιογραφία και ως «πρόβλημα επίπεδου σημείου» (flat spot problem) το οποίο χαρακτηρίζεται από μία κατάσταση κατά την οποία οι νευρώνες εξόδου (όταν είναι

σιγμοειδής (sigmoid-like)) «παγιδεύονται» σε μία κορεσμένη τιμή ενεργοποίησης κατά τα αρχικά στάδια της εκπαιδευτικής διαδικασίας. Όσο παραμένουν παγιδευμένοι, κάτι το οποίο μπορεί να συμβεί για πολλές (δεκάδες χιλιάδες) επαναλήψεις, οι κορεσμένοι αυτοί νευρώνες εξόδου, αποτρέπουν οποιαδήποτε σημαντική αλλαγή από το να συμβεί στα βάρη των άλλων, άμεσα συνδεδεμένων νευρώνων.

Αναλυτικότερα [31], έστω ένα δίκτυο με ένα κρυφό στρώμα το οποίο χρησιμοποιείται για εφαρμογές κατηγοριοποίησης προτύπων. Οι νευρώνες του δικτύου αυτού είναι $tanh$ σιγμοειδής και άρα κανονικοποιούν το αποτέλεσμα στο διάστημα $[-1,1]$. Ως γνωστόν κατά την εκπαίδευση υπολογίζονται τα αθροίσματα των βεβαρημένων εισόδων από το οποίο αφαιρείται η τιμή της σταθεράς πόλωσης και στην συνέχεια το αποτέλεσμα κανονικοποιείται στο $[-1,1]$ με χρήση της $tanh$. Όταν το βεβαρυμμένο αυτό άθροισμα τυχαίνει να απέχει πολύ από την σταθερά πόλωσης η κλίση της σιγμοειδούς συνάρτησης που σχετίζεται με αυτό θα είναι πολύ μικρή (όπως φαίνεται και στο διάγραμμα τής) καθώς η ενεργοποίηση του νευρώνα θα είναι πολύ κοντά στα άκρα του πεδίου τιμών της (κοντά στο 1 ή -1).



Διάγραμμα 1.7: Πρώτη Παράγωγος σιγμοειδούς συνάρτησης tanh

Όταν λοιπόν τα βεβαρημένο άθροισμα των εισόδων απέχει πολύ από την σταθερά πόλωσης τότε λέγεται ότι τα βάρη είναι εντός της περιοχής κορεσμού και άρα ο συγκεκριμένος νευρώνας εξόδου είναι κορεσμένος το οποίο σημαίνει ότι οι αλλαγές στα βάρη του θα είναι μικρές παρά τις μεγάλες τιμές που μπορεί να έχουν τα σφάλματα. Τυπικά απαιτείται πολύς χρόνος προκειμένου ένας κορεσμένος νευρώνας να εξέλθει της καταστάσεως κορεσμού.

Το φαινόμενο αυτό έχει μελετηθεί από διάφορους ερευνητές [32], οδηγώντας σε πολύ ενδιαφέροντα αποτελέσματα. Διάφορες τεχνικές έχουν προταθεί για την αντιμετώπιση αυτού του ανεπιθύμητου φαινομένου μερικές εκ τις οποίες θα αναφερθούν.

- Σε έρευνές των Lee και άλλων [32] δείχτηκε ότι η πιθανότητα εμφάνισης κορεσμένων νευρώνων μπορεί να υπολογιστεί από την μέγιστη τιμή του αρχικού βάρους των νευρώνων, τον αριθμό των κρυφών νευρώνων και την μέγιστη κλίση της σιγμοειδούς

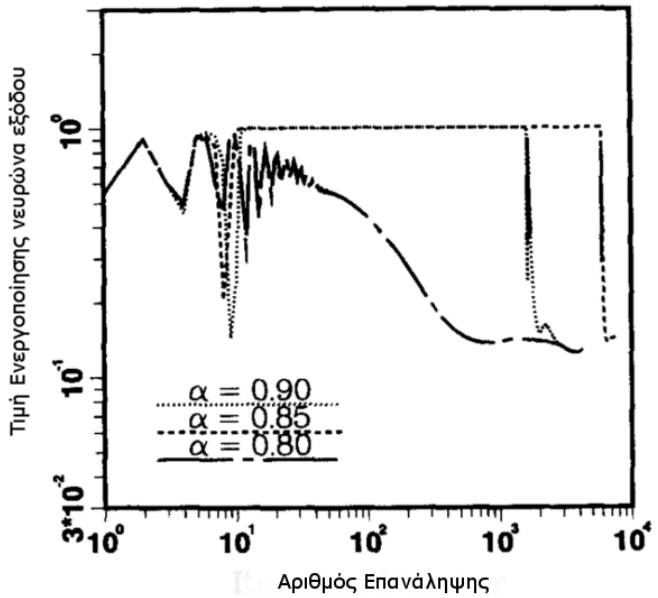
συνάρτησης ενεργοποίησης. Αφού υπολογιστεί αυτή η πιθανότητα, το φαινόμενο μπορεί να αποφεύγει θέτοντας ορθώς τα αρχικά βάρη του δικτύου. Αυτή τεχνική όμως προϋποθέτει τα βάρη και οι σταθερές πώλησης να ακολουθούν την ομοιόμορφη κατανομή κάτι το οποίο δεν είναι πάντα εφικτό σε πρακτικές εφαρμογές.

- Σε μια διαφορετική προσέγγιση των Vitela και Reifman [33] προτάθηκε ο διαχωρισμός της διαδικασίας μάθησης σε 3 στάδια.

1. Αρχή του σταδίου κορεσμού
2. Οροπέδιο του σταδίου κορεσμού
3. Στάδιο ολικής ανάρρωσής.

Επίσης προτάθηκαν τέσσερις απαραίτητες συνθήκες εμφάνισης του φαινομένου του πρόωρου κορεσμού κατά την εκπαίδευση με αλγόριθμο οπισθοδρομικής διάδοσης σφάλματος. Οι συνθήκες αυτές εξαρτώνται από τα διανύσματα κλίσης, τους όρους τις ορμής 1.4.1.2.2.2 (momentum) και της κατάστασης ακραίου σφάλματος. Σύμφωνα με αυτή την μέθοδο λοιπόν, αφού κατασκευαστεί ο μηχανισμός πρόωρου σφάλματος και οι απαραίτητες συνθήκες, στην συνέχεια αποτρέπεται η εμφάνιση οποιασδήποτε εκ των τεσσάρων αυτών συνθηκών και άρα αποφεύγεται ο πρόωρος κορεσμός που οδηγεί σε αργές μαθησιακές καταστάσεις.

Στα πλαίσια την εν λόγω έρευνας, ο όρος της ορμής στοχοποιείται ως ο κύριος ένοχος της εμφάνισης του πρόωρου κορεσμού και συνεπώς γίνεται προσπάθεια για την εύρεση της βέλτιστης τιμής του αποφεύγοντας την αργή σύγκλιση. Στο παρακάτω σχήμα παρουσιάζεται πειραματικά η σημασία της επιλογής της σωστής τιμής της ορμής κατά την εκπαίδευσης. Όπως φαίνεται στο Διάγραμμα 1.8 , τα επίπεδα ενεργοποίησης ενός νευρώνα εξόδου σε δίκτυο στο οποίο έχει δοθεί ένα συγκεκριμένο πρότυπο, περιέρχονται σε κατάσταση πρόωρου κορεσμού για ορμή $\alpha = 0.9$ και $\alpha = 0{,}85$. Αντιθέτως για $\alpha = 0{,}80$ ο νευρώνας λειτουργεί ομαλά.



Διάγραμμα 1.9: Επίδραση τιμής του όρου ορμής στην πρόωρο κορεσμό νευρώνα [33]

Αν και αποτελεσματική στην αντιμετώπιση του πρόωρου κορεσμού κατά τα πρώιμα εκπαιδευτικά στάδια, η μέθοδος αυτή έχει επικριθεί [34] ότι είναι περιοριστική τη φύσει, καθώς απαιτεί τον σχεδιασμό ενός σενάριού κατά το οποίο περιγράφεται η σχέση μεταξύ του όρου ορμής και των αθροισμένων βαρών τα οποίο όμως πρέπει να λαμβάνουν πολύ μικρή τιμή εξαιτίας της μεγάλη τιμής της ορμής κατά τα αρχικά, εκπαιδευτικά στάδια.

- Τέλος, έρευνες [34] έχουνε θεωρήσει ως βασικό αίτιο του πρόωρου κορεσμού το φαινόμενο του κορεσμού σφάλματος, φαινόμενο το οποίο επίσης περιγράφει νευρώνες που έχουν περιέλθει σε κατάσταση κορεσμού (έξοδος του νευρώνα στα όρια του εύρους τιμών της σιγμοειδούς συνάρτησης ενεργοποίησης) με την διαφορά όμως, ότι στην προκείμενη περίπτωση υπάρχουν εμφανής διάφορες μεταξύ της επιθυμητής εξόδου και της ενεργοποίησης του νευρώνα. Στην εργασία αυτή χρησιμοποιείται ο αλγόριθμος της απότομης καθόδου και απλή σιγμοειδείς συνάρτηση ενεργοποίησης $S(t) = \frac{1}{1+e^{-t}}$ με $t = \sum_j (w_j * x_j) + b$.

  Αναλυτικότερα, ορίζεται ο μαθησιακός όρος για των νευρώνα εξόδου ι ως:

$$\xi_\iota = (y_\iota - y_i')ky_i(1 - y_i) \; οπου \; k = 1$$

  και η μεταβολή του βάρους του νευρώνα (κατά την εκπαίδευση) από τον j-οστό νευρώνα του στρώματος $L-1$, ενός δικτύου με L νευρώνες, προς τον i-οστό νευρώνα του στρώματος εξόδου L ως:

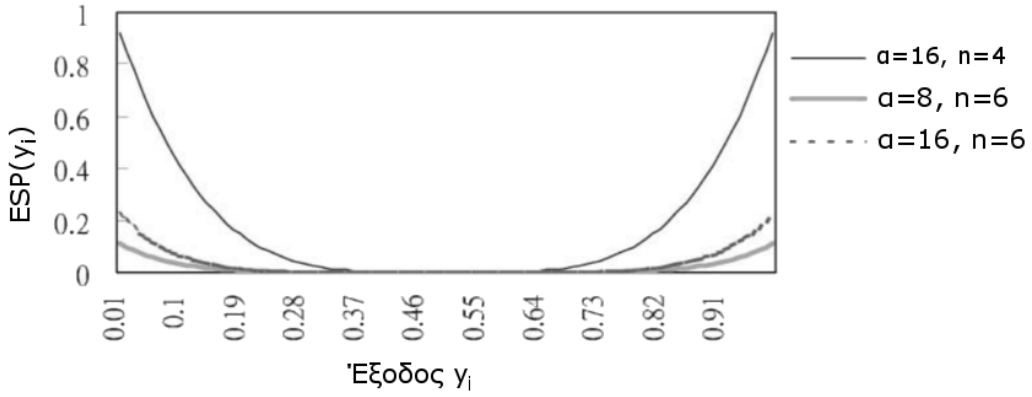
$$\Delta w_{ji} = \eta * \frac{\vartheta C}{\vartheta w_{ji}} = -\eta \xi_i y_j$$

  Αποδεικνύεται ότι όταν η έξοδος ενός νευρώνα λαμβάνει τις οριακές τιμές 0 ή 1 ο μαθησιακός όρος γίνεται πολύ μικρός ακόμα και αν τιμή $(1 - y_i)$ είναι μεγάλη. Αυτό

ισχύει ακόμα και αν η έξοδος y_i απέχει πολύ από την επιθυμητή έξοδο y_i'.Προκειμένου να αποφευχθεί το φαινόμενο του κορεσμένου σφάλματος που αναλύθηκε προτάθηκε η χρήση της παραβολικής συνάρτησης ESP (Error Saturation Prevention):

$$ESP(y_i) = a(y_i - 0{,}5)^n$$

Όπου α είναι ένας όρος κλίμακας και n ένας εκθετικός όρος των οποίων η επιρροη στην ESP διαφαίνεται στο παρακάτω σχήμα.



Διάγραμμα 1.10: Συνάρτηση ESP για διάφορες τιμές όρου κλίμακας και εκθετικού όρου [34]

Όπως φαίνεται και από το διάγραμμα της, η ESP έχει μεγάλη επιρροή στις εξόδους όταν αυτές βρίσκονται στο όρια του πεδίου ορισμού της σιγμοειδούς (δηλαδή στο 0 και στο 1) ενώ μικρή έως καθόλου για μη-οριακές εξόδους. Με την προσθήκη της ESP λοιπόν ο μαθησιακός όρος γίνεται:

$$\xi_\iota = (y_\iota - y_i')ky_i(1 - y_i) + ESP(y_i)$$

Να σημειωθεί ότι ο όρος $ESP(y_i)$ μπορεί να προστεθεί και σε κρυφούς νευρώνες αλλά μόνο εάν συρρικνωθεί πολλαπλασιαζόμενος κατά έναν πολύ μικρό παράγοντα c (π.χ. $c = 0.01$) καθώς ο μαθησιακός όρος στους κρυφούς νευρώνες είναι πολύ πιο μικρός και ευεπηρέαστος.

Αναφορικά με την επιλογή των παραμέτρων α και n, μετά από την ανάλυση της κατανομής του όρου εκμάθησης και του ορισμού της παραβολικής συνάρτησης, αποδείχτηκε ότι:

- Ο εκθετικός όρος n απαιτείται να είναι ζυγός αριθμός εξαιτίας της θετικής συμμετρικής κατανομής στην είσοδο 0,5.
- Εάν η βελτίωση της σύγκλισης της μάθησης δεν είναι προφανής με την εφαρμογή της ESP τότε είτε ο όρος κλίμακας πρέπει να μεγεθυνθεί είτε πρέπει να συρρικνωθεί ο εκθετικός όρος.
- Εάν η σύγκλιση της μάθησης επιταχύνεται αλλά η γενίκευση (δες επόμενο κεφάλαιο) μπορεί να παρουσιαστεί φαινόμενο ταλάντωσης. Σε αυτή την περίπτωση ο όρος κλίμακας πρέπει να συρρικνωθεί προκειμένου να μειωθεί η επιρροή της ESP. Εάν το φαινόμενο αυτό εξακολουθεί να εμφανίζεται και

μετά την μείωση του όρου κλίμακας τότε πρέπει να εξεταστεί αύξηση του εκθετικού όρου.

Τέλος παρατίθενται πειραματικά αποτελέσματα από έναν νευρωνικό δίκτυο $2 * 2 * 1$ (2 νευρώνες εισόδου, δύο κρυφοί και ένας εξόδου)το οποίο εκπαιδεύεται με σκοπό την εκμάθηση της παραλλαγμένης πύλης XOR με πίνακα αλήθειας:

Πίνακας 1.1: Πίνακας αλήθειας παραλλαγμένης XOR

Είσοδος 1	Είσοδος 2	Επιθυμητή έξοδος
0,0	0,0	0,0
0,0	1,0	1,0
1,0	0,0	1,0
1,0	1,0	0,0
0,5	0,5	1,0

Στον παρακάτω πίνακα συγκρίνονται 4 εκπαιδευτικές μέθοδοι αναφορικά με τον αριθμό των επαναλήψεων που απαιτήθηκαν προκειμένου το μέσω τετραγωνικό σφάλμα να λάβει την τιμή 0.0001.

Πίνακας 1.2: Πίνακας αποτελεσμάτων εκπαίδευσης δικτύου για εκμάθηση παραλλαγμένης XOR [34]

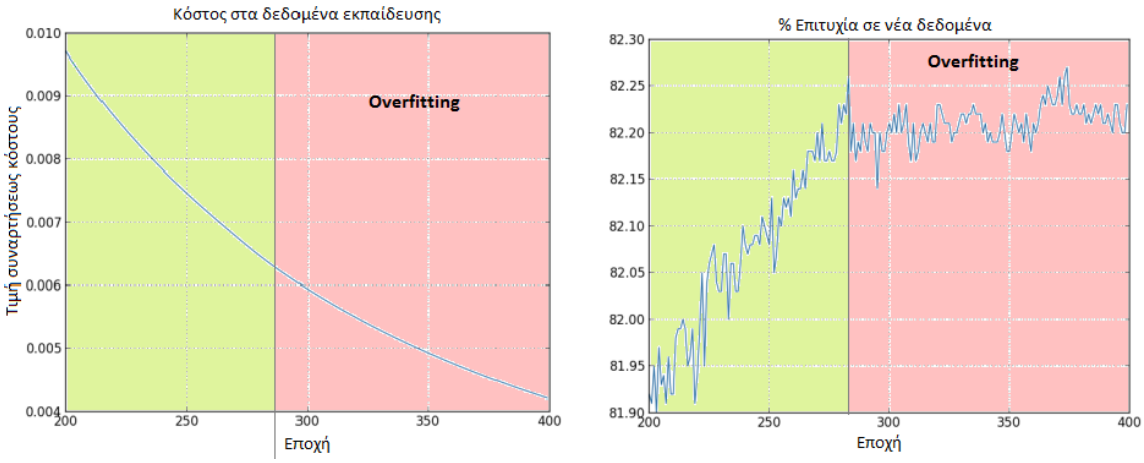
Μέθοδος	Μέσος Αριθμός επαναλήψεων	Μέσος χρόνος εκπαίδευσης	Ρυθμός εκμάθησης	Ορμή
Αλγ.ΟΔΣ.	10832	17,24	0,5	-
Αλγ. ΟΔΣ.	8703	12,25	0,5	0,2
Conjugate gradient ΟΔΣ **[35]**	7616	14,32		-
Αλγ. ΟΔΣ. με ESP (α=16, n=4)	4615	7,76	0,5	-

1.4.4.3 Υπερπροσαρμογή (Overfitting)

Η βασική πρόκληση των νευρωνικών δικτύων και γενικά της μηχανικής μάθησης είναι να μπορέσει, το δημιουργηθέν, μοντέλο να λειτουργήσει ορθά όταν συναντήσει νέα δεδομένα τα οποία δεν έχει συναντήσει προηγουμένους [21]. Η ικανότητα αυτή, ονομάζεται γενίκευση (generalization). Βασικός σκοπός λοιπόν, είναι η μείωση του λεγόμενου σφάλματος γενίκευσης. Δηλαδή μείωση του αναμενόμενου σφάλματος για νέα είσοδο. Όταν το σφάλμα αυτό είναι μεγάλο, εξαιτίας της μεγάλης απόστασης που χωρίζει το σφάλμα κατά την εκπαίδευση με το σφάλμα κατά την περάτωση δοκιμών με νέα δεδομένα τότε το δίκτυο λέγετε ότι βρίσκεται σε κατάσταση Overfitting. Δηλαδή, ένα δίκτυο έχει περιέλθει σε κατάσταση Overfitting όταν έχει υπέρ-προσαρμοστεί (over-fitting) στα εκπαιδευτικά

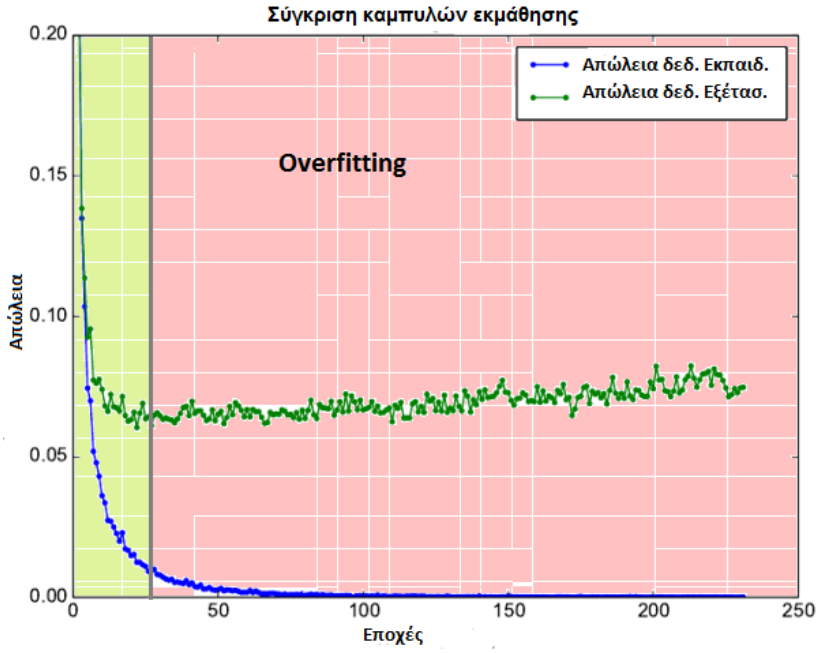
δεδομένα παρουσιάζοντας εικονικά αποτελέσματα επιτυχίας ενώ στην πράξη δεν έχει μάθει να αναγνωρίζει ουσιαστικά τα χαρακτηριστικά τους.

Η διάγνωση ενός δικτύου με Overfitting μπορεί να γίνει δια μέσου της καταγραφής και της σύγκρισης ορισμένων συμπτωμάτων που μπορεί να παρουσιάζονται. Για παράδειγμα, εάν κατά την εκπαίδευση η συνάρτηση κόστους μειώνεται διαρκώς (κάτι το επιθυμητό) αλλά κατά την διεξαγωγή δοκιμών με διαφορετικά δεδομένα ο ρυθμός σφάλματος μεταβάλλεται με διαφορετικό πρόσημο απ' αυτό του κόστους τότε το δίκτυο έχει περιέλθει σε κατάσταση Overfitting. Αυτή ακριβώς η κατάσταση απεικονίζεται και στο παρακάτω σχήμα. Στο αριστερό διάγραμμα απεικονίζεται η τιμή της συνάρτησης κόστους ενός δικτύου, την περίοδο $[200\ epoch - 400\ epoch]$. Το κόστος στο διάγραμμα μειώνεται διαρκώς και ομαλά και άρα το δίκτυο παρουσιάζει την εικόνα ότι εκπαιδεύεται επιτυχώς. Στο δεξί σχήμα όμως, στο οποίο απεικονίζεται το ποσοστό επιτυχούς πρόβλεψης μιας ομάδας νέων-αγνώστων δεδομένων παρατηρείται το εξής φαινόμενο. Ενώ μέχρι την εποχή 280 η ρυθμός επιτυχίας ήταν ανοδικός, όπως άλλωστε ήταν αναμενόμενο (καθώς το κόστος μειώνεται και άρα η έξοδος του δικτύου πλησιάζει την ετικέτα), για το επόμενο διάστημα $[280\ epoch - 400\ epoch]$ η συμπεριφορά του δεν η προβλεπόμενη καθώς δεν παρουσιάζει αδιάληπτη αύξηση.



Διάγραμμα 1.11: Ενδείξεις Overfitting δικτύου συγκρίνοντας κόστος δεδομένων και ποσοστό επιτυχίας σε νέα δεδομένα [21]

Ένα άλλο παρόμοιο δείγμα δυσανάλογης μεταβολής μεγεθών των δεδομένων εκπαίδευσης και των δεδομένων εξέτασης παρουσιάζει το παρακάτω σχήμα στο οποίο γίνεται σύγκριση των καμπυλών εκμάθησης των δεδομένων εκπαίδευσης και των δεδομένων εξέτασης. Overfitting παρατηρείται από την εικοστή πέμπτη εποχή και μετά όταν η απώλεια των δεδομένων εξέτασης ακολουθεί ανοδική πορεία σε αντίθεση με την καθοδική πορεία της απώλειας δεδομένων εκπαίδευσης.



Διάγραμμα 1.12: Overfitting [21]

Το Overfitting είναι ένα πολύ σημαντικό πρόβλημα στα νευρωνικά δίκτυα και κυρίως στα μοντέρνα δίκτυα τα οποία συχνά αποτελούνται από μεγάλο αριθμό παραμέτρων. Για τον λόγω αυτό έχουν βρεθεί πολλές μέθοδοι για την αντιμετώπιση του οι οποίες στοχεύουν στην γενίκευση του μοντέλου. Όλες οι παρακάτω μέθοδοι είναι μέθοδοι ομαλοποίησης οι οποίες εισάγοντας περαιτέρω περιορισμούς σε ένα μοντέλο νευρωνικού δικτύου (π.χ. περιορισμούς στις παραμέτρους) επιδιώκουν να αποτρέψουν το «ταίριασμα» του μοντέλου με το σετ δεδομένων εκπαίδευσης.

1.4.4.3.1 Πρώιμη Διακοπή (Early Stoppage)

Ο πιο απλός τρόπος είναι να παρακολουθείται το δίκτυο κατά την εκπαίδευση του και ανά τακτά διαστήματα να γίνονται δοκιμές με άγνωστα δεδομένα. Σε περίπτωση που ο ρυθμός σφάλματος στις δοκιμές αυτές αρχίζει να αυξάνεται αντί να μειώνεται, τότε το δίκτυο έχει, πιθανώς, περιέλθει σε κατάσταση Overfitting και άρα η εκπαίδευση πρέπει να διακοπεί [21]. Συγκεκριμένα, σε μια εφαρμογή, το σετ δεδομένων διασπάτε σε τρείς ομάδες, την ομάδα εκπαίδευσης (training set), την ομάδα επαλήθευσης (validation set) και την ομάδα εξέτασης (test set). Μετά από ένα χρονικό διάστημα (π.χ. στο τέλος κάθε εποχής) εκπαίδευσης του δικτύου, θα δοκιμασθεί η επίδοση του στην ομάδα δεδομένων επαλήθευσης. Κάθε φορά που παρατηρείται βελτίωση της απόδοσης στα δεδομένα επαλήθευσης οι παράμετροι του δικτύου αποθηκεύονται. Αν παρατηρηθεί κορεσμός στην βελτίωση, τότε η εκπαίδευση του δικτύου διακόπτεται μετά από λίγες εποχές προκειμένου να καταστεί σίγουρο ότι ο κορεσμός αυτός δεν είναι προσωρινός. Έτσι λοιπόν το δίκτυο καταλήγει να έχει βέλτιστη απόδοση στην ομάδα επαλήθευσης χωρίς να έχει εκπαιδευτεί σε αυτήν, αποφεύγοντας σε έναν βαθμό το Overfitting. Η τεχνική αυτή ονομάζεται πρώιμη διακοπή (early stoppage) και αποτελεί την πιο κοινώς χρησιμοποιημένη μορφή ομαλοποίησης λόγω της απλότητας αλλά και της αποτελεσματικότητας της. Ακολουθεί ο αλγόριθμος.

Έστω n ο αριθμός βημάτων που μεσολαβούν μεταξύ αξιολογήσεων.

Έστω p η «υπομονή», δηλαδή ο αριθμός των φορών που θα γίνουν ανεκτές χειρότερες επιδώσεις στα δεδομένα επικύρωσης.

Έστω θ_0 οι αρχικές παράμετροι του δικτύου.

$\theta \leftarrow \theta_0$

$i \leftarrow 0$

$j \leftarrow 0$

$v \leftarrow \infty$

$\theta\text{^} * \leftarrow 0$

$i\text{^} * \leftarrow i$

Όσο $j < p$ επανέλαβε:

Ενημέρωσε το θ τρέχοντας τον αλγόριθμο εκπαίδευσης για n βήματα

$i \leftarrow i + n$

$v' \leftarrow Βρές\_Σφάλμα\_Δεδομένων\_αξιολόγησης(\theta)$

Αν $v' < v$ τότε:

$j \leftarrow 0$

$\theta^* \leftarrow \theta$

$i^* \leftarrow i$

$v \leftarrow v'$

Αλλιώς:

$j \leftarrow j + 1$

Τέλος_Άν

Τέλος_Όσο

Οι καλύτερές παράμετροι είναι οι θ^* και ο βέλτιστος αριθμός βημάτων εκπαίδευσης είναι ίσος με i

Προκειμένου να γίνει περεταίρω εκμετάλλευσή των πληροφοριών που ελήφθησαν κατά την διαδικασία πρώιμης διακοπής, το δίκτυο μπορεί να επανεκπαιδευτεί, αλλά αυτή την φόρα με χρήση όλων των διαθέσιμών δεδομένων κατά την εκπαίδευση. Υπάρχουν δύο βασικές προσεγγίσεις γι' αυτήν την δεύτερη επανεκπαίδευση του δικτύου.

- Επάναρχικοποίηση μοντέλου και επανεκπαίδευση του σε όλα τα δεδομένα. Σε αυτό το δεύτερο «πέρασμα», η εκπαιδευτική διαδικασία θα διαρκέσει τον βέλτιστο αριθμό βημάτων όπως αυτός ορίστηκε κατά την διαδικασία της πρώιμής διακοπής. Η τεχνική αυτή παρουσιάζει κάποιες δυσκολίες. Για παράδειγμα δεν υπάρχει κάποιος ορθός

τρόπος προκειμένου να καθοριστεί αν θα διατηρεί ο ίδιος αριθμός ανανεώσεων παραμέτρων ή ο ίδιος αριθμός «περασμάτων» στο σετ δεδομένων (εποχών). Συνήθως, εάν υπάρχουν σοβαρές ενδείξεις Overfitting προτιμάται η διατήρηση του ιδίου αριθμού εποχών και όχι των ανανεώσεων παραμέτρων. Αντιθέτως αν υπάρχει πρόβλημα βελτιστοποίησης τότε η διατηρείται ο ίδιος αριθμός ανανεώσεων παραμέτρων.

- Διατήρηση των παραμέτρων που αποκτήθηκαν κατά την πρώτη εκπαίδευση και επανεκπαίδευση με χρήση όλων των δεδομένων. Σε αυτή την τεχνική δεν είναι απαραίτητη η καθοδήγηση του δικτύου για το πότε θα σταματήσει η εκπαιδευτική διαδικασία. Αντιθέτως μέσω της παρακολούθηση του κόστους, το δίκτυο θα σταματήσει την εκπαίδευση όταν πετύχει καλύτερα αποτελέσματα απ' αυτά που αποκτήθηκαν κατά την πρώτη εκπαίδευση. Το βασικό πρόβλημα που παρουσιάζει αυτή η τεχνική είναι ότι δεν υπάρχει εγγύησή ότι θα επιτευχθούν καλύτερα αποτελέσματα κατά την δεύτερη εκπαίδευση και άρα ελλοχεύει ο κίνδυνος η εκπαίδευση να συνεχιστεί αενάως.

1.4.4.3.2 Αποσύνθεση Βάρους (weight decay) /Ομαλοποίηση L2 (L2 regularization)

Η ομαλοποίηση L2 ή αλλιώς L^2 [21] είναι μία από τις πιο απλές και συνηθισμένες μεθόδους ομαλοποίησης. Η L2 όταν εφαρμόζεται σε νευρωνικά δίκτυα η τεχνική αυτή ονομάζεται και ως Αποσύνθεση Βάρους (weight decay). Η ιδέα πίσω από την L2 ομαλοποίηση είναι η προσθήκη ενός περαιτέρω όρου στην συνάρτηση κόστους. Ο όρος αυτός ονομάζεται όρος ομαλοποίησης. Άρα αν σε μία συνάρτηση κόστους C εφαρμοστεί weight decay γίνεται:

$$C = C_0 + \frac{\lambda}{2n}\sum_w w^2$$

Ουσιαστικά στην συνάρτηση το κόστους προστίθεται το άθροισμα όλων των βαρών μεγεθυμένο κατά έναν παράγοντα $\frac{\lambda}{2n}$ όπου λ είναι η παράμετρος ομαλοποίησης και n συνήθως είναι το μέγεθος του σετ δεδομένων εκπαίδευσης. Ενδεικτικά η τετραδική συνάρτηση κόστους μετά εφαρμογής weight decay γίνεται:

$$C_{quadr} = \frac{1}{2n}\left(\sum_i (F_{w,b}(x^i) - y'^i)) + \frac{\lambda}{2n}\sum_w w^2\right)$$

Διαισθητικά, αυτό που κάνει η ομαλοποίηση είναι να εξαναγκάζει το δίκτυο να δείχνει προτίμηση στην εκμάθηση μικρών βαρών. Τα μεγάλα βάρη θα επιτραπούν μόνο αν βελτιώνουν επαρκώς το πρώτο μέρος της εξίσωσης κόστους C_0. Παραφράζοντας, η ομαλοποίηση μπορεί να γίνει αντιληπτή ως μια συμβιβαστική λύση μεταξύ της εξεύρεσης μικρών βαρών και της ελαχιστοποίησης της αρχικής συνάρτησης κόστος C_0. Αυτό είναι επιθυμητό καθώς το διάνυσμα βάρους μίας μη-ομαλοποιημένης συνάρτηση κόστους είναι πιθανόν να αρχίσει να μεγαλώνει. Εν καιρώ το υπερμεγεθυμένο διάνυσμα βαρών θα δείχνει πάνω κάτω στην ίδια διεύθυνση (στον χώρο της συνάρτησης κόστους στον οποίο αναζητείται το ελάχιστο) διαρκώς καθώς οι αλλαγές κατά την εκπαίδευση θα είναι μικρές.

1.4.4.3.3 L1 Ομαλοποίηση

Σε αυτή την προσέγγιση ομαλοποίησης [21], η συνάρτηση κόστους τροποποιείται προσθέτοντας της το άθροισμα των απολύτων των βαρών.

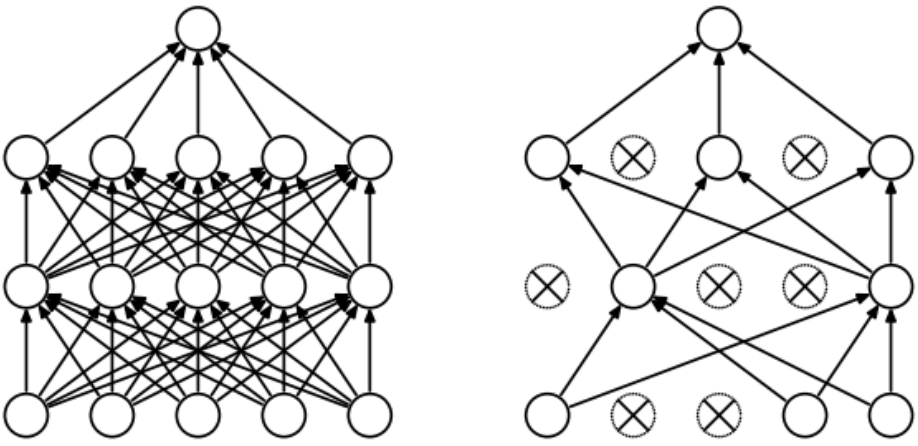
$$\mathrm{C} = \mathrm{C}_0 + \frac{\lambda}{n}\sum_{w}\|w\|$$

Παρά την ομοιότητα της L1 με την L2 δεν παρουσιάζουν την ίδια συμπεριφορά. Αυτό γίνεται εμφανέστερο όταν κατά την διαδικασία εκπαίδευσης οι συναρτήσεις αυτές παραγοντοποιηθούν κατά τον υπολογισμό των κλίσεων. Χαρακτηριστικά η μερική ς παράγωγος της ομαλοποιημένης συνάρτησης με L1 είναι $\frac{\partial C}{\partial w} = \frac{\partial C_0}{\partial w} + \frac{\lambda}{n} sgn(w)$ ενώ της L2 είναι $\frac{\partial C}{\partial w} = \frac{\partial C_0}{\partial w} + \frac{\lambda}{n} w$. Άρα η συνεισφορά της L1 είναι απλά ένας σταθερός παράγοντας του οποίου το πρόσημο εξαρτάται από το w ενώ αντιθέτως το L2 μεταβάλλεται γραμμικά με το w. Συγκριτικά με αυτά της L2, λοιπόν, τα αποτελέσματα της κανονικοποίησης L1 είναι πιο αραιά, δηλαδή υπάρχουν ελεύθερες παράμετροι του μοντέλου οι οποίες μέσω της διαδικασίας ομαλοποίησης αποκτούν ως βέλτιστη την τιμή 0. Για τον λόγο αυτό η L1 χρησιμοποιείται συχνά σε εφαρμογές εξαγωγής χαρακτηριστικών καθώς οι αχρείαστες μεταβλητές τείνουν να συρρικνωθούν στο μηδέν.

1.4.4.3.4 Dropout

Το dropout είναι μια εντελώς διαφορετική τεχνική ομαλοποίησης [36]. Σε αντίθεση με τις προηγούμενες τεχνικές το dropout δεν βασίζεται στην τροποποίηση της συνάρτησης κόστους αλλά στην τροποποίησης του ίδιου του δικτύου. Με αυτό τον τρόπο παρέχεται μία υπολογιστικά φτηνή λύση για την ομαλοποίηση ενός μεγάλου εύρους αρχιτεκτονικών. Ενδιαφέρον παρουσιάζει το γεγονός ότι η τεχνική αυτή είναι μερικώς εμπνευσμένη από τον ρόλο της αναπαραγωγής στην εξέλιξη. Κατά την αναπαραγωγή, λαμβάνονται τα μισά γονίδια από τον έναν γονέα και τα μισά από τον άλλον, στην συνέχεια προστίθεται μια μικρή, τυχαία μετάλλαξη και τέλος τα γονίδια συνδυάζονται για την παραγωγή του απογόνου.

Το dropout λοιπόν αποτρέπει το Overfitting διευκολύνοντας ταυτόχρονα τον συνδυασμό εκθετικών τω πλήθος διαφορετικών αρχιτεκτονικών. Ο όρος dropout αναφέρεται στην εξαναγκασμένη εγκατάλειψη (dropping out) νευρώνων (ορατών ή και κρυφών) ενός νευρωνικού δικτύου. Ουσιαστικά, όταν έναν νευρώνας εξαναγκάζεται να εγκαταλείψει το δίκτυο, αφαιρετέ προσωρινά από το δίκτυο μαζί με όλες της εισερχόμενές και εξερχόμενες συνδέσεις του. Όπως φαίνεται και στο σχήμα το δίκτυο b) είναι το δίκτυο a) μετά την επίδραση Dropout.

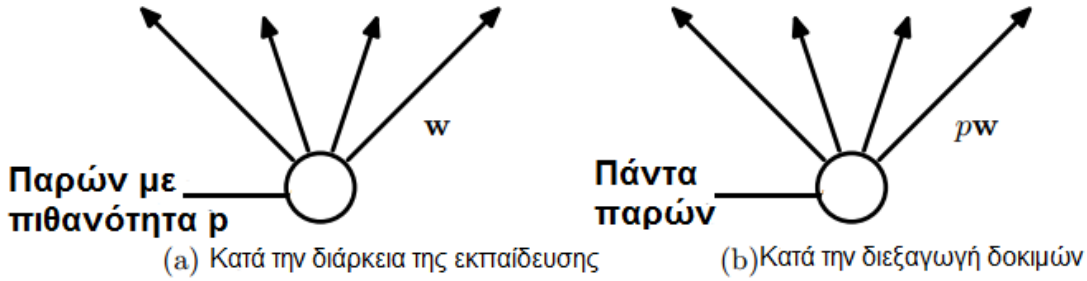


Εικόνα 1.12: Γράφος Βαθιού Δικτύου πριν και μετά την εφαρμογή Dropout [36]

Η επιλογή για το ποιοι νευρώνες θα εγκαταλειφθούν είναι τυχαία. Στην πιο απλή των περιπτώσεων, κάθε νευρώνας θα παραμείνει στο δίκτυο μία προκαθορισμένη και σταθερή πιθανότητα p ανεξάρτητη των άλλων νευρώνων. Το p μπορεί να επιλεχτεί από ένα σετ επικύρωσης (validation set) η απλά του ανατεθεί η τιμή $p = 0{,}5$, η οποία δείχνει να είναι η βέλτιστη για ένα μεγάλο εύρος εφαρμογών και δικτύων. Για νευρώνες εισόδου παρόλα αυτά η βέλτιστη πιθανότητα διατήρησης συχνά είναι μια τιμή κοντά στο 1 και όχι στο 0,5.

Η εφαρμογή dropout σε ένα δίκτυο έχει ως αποτέλεσμα την λήψη μιας «αραιωμένη» εκδοχής του. Αυτή η εκδοχή του δικτύου αποτελείται από όλους το νευρώνες που επιβίωσαν τα Dropout .Ένα δίκτυο με n νευρώνες μπορεί λοιπόν να εκληφθεί ως μια συλλογή από 2^n πιθανά νευρωνικά δίκτυα που έχουν αραιωθεί. Αυτά τα δίκτυα έχουν κοινά βάρη, έτσι ώστε ο συνολικός αριθμός τον παραμέτρων να εξακολουθεί να είναι $O(2n)$ ή λιγότερο. Κάθε φορά που παρουσιάζεται μια εκπαιδευτική περίπτωση δειγματοληπτείται και εκπαιδεύεται ένα νέο, αραιωμένο δίκτυο. Άρα η εκπαίδευση ενός δικτύου με χρήση dropout μπορεί να γίνει αντιληπτή ως η εκπαίδευση μιας συλλογής από 2^n αραιωμένα δίκτυα τα οποία μοιράζονται εκτενώς τα βάρη τους και καθένα από τα οποία εκπαιδεύεται από σπάνια ως καθόλου.

Εξαιτίας του μεγάλου υπολογιστικού κόστους δεν είναι δυνατόν ,προς το παρών τουλάχιστον, να ληφθεί ο μέσος όρος των προβλέψεών από κάθε αραιωμένο δίκτυο καθώς όπως προαναφέρθηκε είναι εκθετικά τω πλήθος. Για τον λόγο αυτό έχει εισαχθεί μια άλλη τεχνική για τον υπολογισμό του μέσου όρου κατά προσέγγιση. Σύμφωνα με αυτή την τεχνική το δίκτυο χρησιμοποιείται χωρίς την εφαρμογή dropout κατά την διεξαγωγή δοκιμών. Τα βάρη αυτού του δικτύου είναι συρρικνωμένες εκδώσεις των βαρών που προέκυψαν κατά την εκπαίδευση. Συγκεκριμένα τα εξερχόμενα βάρη κάθε νευρώνα του δικτύου (Εικόνα 1.13 b)) ισούνται με τα αντίστοιχά βάρη που προέκυψαν κατά την εκπαίδευση (Εικόνα 1.4.4 a)) πολλαπλασιασμένα με την πιθανότητα διατήρησης p.



Εικόνα 1.13: Dropout Νευρώνας κατά την εκπαίδευση και κατά την διεξαγωγή δοκίμων [36]

Εφαρμόζοντας αυτή την συρρίκνωση καθιστάτε δυνατός ο συνδυασμός των 2^n δικτυών με κοινά βάρη σε ένα μοναδικό δίκτυο.

Η εφαρμογή του dropout δεν περιορίζεται μόνο στα έμπροσθεν τροφοδοτούμενα δίκτυα (Feed Forward (FF)) αλλά είναι δυνατή η εφαρμογή τους και σε άλλα μοντέλα όπως για παράδειγμα σε Μηχανές Boltzmann. Ακολουθεί η μαθηματική αναπαράσταση ενός δικτύου Dropout και συγκεκριμένα ενός έμπροσθεν τροφοδοτούμενου δικτύου με Dropout. Έστω το δίκτυο έχει L τω πλήθος κρυφά στρώματα όπου $l \in \{1, \dots, \dots, L\}$ το l-οστό κρυφό στρώμα. Με $z^{(l)}$ θα συμβολίζεται το διάνυσμα εισόδου του στρώματος l και με $y^{(l)}$ η έξοδος του στρώματος αυτού (άρα $y^{(0)} = x$ είναι το διάνυσμα εισόδου). Τα διανύσματα βάρους και σταθερών πώλησης του κρυφού στρώματος l δίνονται από τα $W^{(l)}$ και $\mathrm{b}^{(l)}$ αντίστοιχα ενώ με $f(\ )$ συμβολίζεται η συνάρτηση ενεργοποίησης του νευρώνα. Η λειτουργεία ενός έμπροσθεν τροφοδοτούμενου δικτύου και συγκεκριμένα του κρυφού νευρώνα i στο l-οστό στρώμα είναι η παρακάτω:

$$z_i^{(l+1)} = w_i^{(i+1)} y^i + b_i^{(l+1)}$$

Εικόνα 1.4.4 a)

$$y_i^{(l+1)} = f\left(z_i^{(l+1)}\right)$$

Με την εισαγωγή του dropout το δίκτυο γίνεται ως εξής:

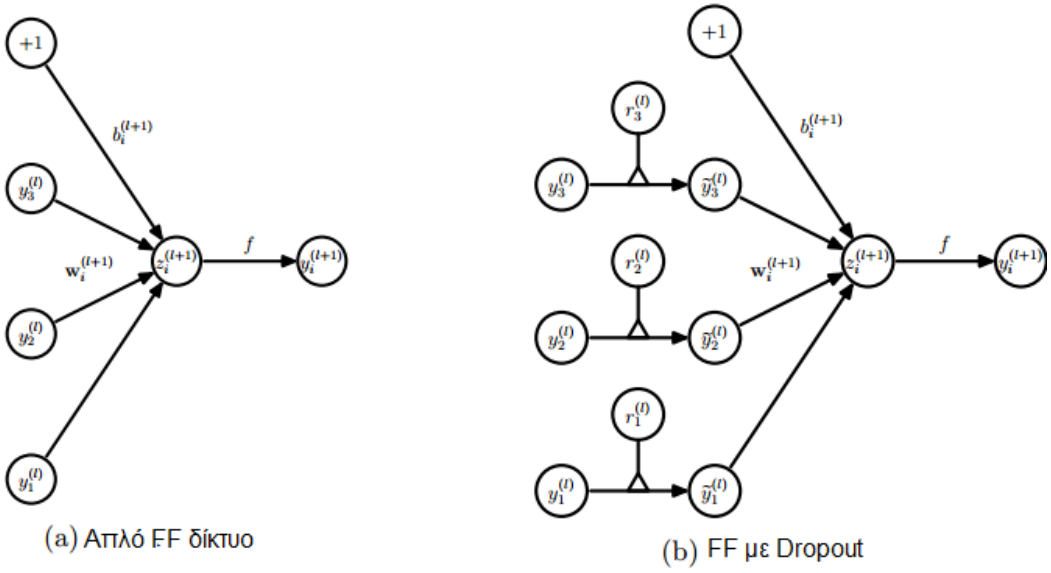
$$z_j^{(l)} \sim Bernoulli(p),$$

$$\tilde{\mathrm{y}}^{(\mathrm{l})} = \mathrm{r}^{(l)} * y^{(l)},$$

Εικόνα 1.4.4 b)

$$z_i^{(l+1)} = w_i^{(l+1)} \tilde{\mathrm{y}}^{(\mathrm{l})} + b_i^{(l+1)},$$

$$y_i^{(l+1)} = f\left(z_i^{(l+1)}\right)$$



Εικόνα 1.14: Έμπροσθεν τροφοδοτούμενο δίκτυο με και χωρίς Dropout [36]

Για κάθε στρώμα l το $r(l)$ είναι ένα διάνυσμα αποτελούμενο από ανεξάρτητες τυχαίες μεταβλητές Bernoulli κάθε μια από τις οποίες ισούται με 1 με πιθανότητα p. Αυτό το διάνυσμα δειγματοληπτείται και πολλαπλασιάζεται (ανά στοιχείο) με το διάνυσμα εξόδου $y^{(l)}$ του εν λόγω στρώματος. Στην συνέχεια οι αραιωμένες έξοδοι χρησιμοποιούνται ως είσοδοι για το επόμενο στρώμα $l+1$. Η διαδικασία αυτή γίνεται σε όλα τα στρώματα και ισοδυναμεί με την δειγματοληψία ενός υποδικτύου από ένα μεγαλύτερο δίκτυο. Κατά την διαδικασία εκπαίδευσης, οι μερικές παράγωγοι της συνάρτησης απώλειας διαδίδονται προς τα πίσω με χρήση οπισθοδρομικής διάδοσης σφάλματος. Όταν αρχίσει η διεξαγωγή των δοκιμών το δίκτυο χρησιμοποιείται στο ακέραιο του με τα βάρη να γίνονται $W_{δοκιμής}(ι) = p * W_{εκπαίδευσης}(ι)$ όπως περιεγράφηκε και προηγουμένως.

Η απόδοση του Dropout μπορεί να βελτιωθεί περαιτέρω με την χρήση μια τεχνικής ομαλοποίησης γνωστή ως max-norm κανονικοποίηση. Η τεχνική αυτή θέτει ένα άνω όριο σε κάθε διάνυσμα βαρών εισερχόμενο σε κρυφό νευρώνα το οποίο είναι ίσο με μια σταθερά c άρα θέτει $\|w\| \leq c$. Με άλλα λόγια κατά την βελτιστοποίηση (π.χ. κατά την χρήση στοχαστικής απότομης καθόδου) το w περιορίζεται εντός των ορίων μία σφαίρας ακτίνας c. Η παράμετρος c είναι μία παράμετρος της οποία η βέλτιστη τιμή μπορεί να βρεθεί πειραματικά με την χρήση ενός σετ δεδομένων επαλήθευσης. Η τεχνική αυτή τυπικά βελτιώνει την απόδοση της στοχαστικής απότομης καθόδου ακόμα και χωρίς την εφαρμογή Dropout.

Καταλήγοντας, παρουσιάζεται ένας πίνακας με πειραματικών αποτελεσμάτων με χρήση διαφόρων αρχιτεκτονικών. Τα αποτελέσματα προκύπτουν αρχικά με απλή εφαρμογή του εκάστοτε δικτύου και στην συνέχεια με χρήση Dropout. Τα δίκτυα αυτά εκπαιδευτήκαν για την κατηγοριοποίηση του σετ δεδομένων MNIST. Συγκεκριμένα χρησιμοποιήθηκαν εικόνες $28 * 28$ στην κλίμακα του γκρι. Το σετ δεδομένων εκπαίδευσης είχε μέγεθος 60000 ενώ το σετ εξέτασης 10000 εικόνες.

Πίνακας 1.3: Συγκριτικά αποτελέσματα Διαφόρων Νευρωνικών Δικτύων στο σετ Δεδομένων MNIST με και χωρίς Dropout. [36]

Μέθοδος	Τύπος Νευρώνα	Αρχιτεκτονική	Σφάλμα
Απλό Νευρωνικό Δίκτυο (ΝΔ)	Χρήση Λογιστικής Παλινδρόμηση	2 στρώματα, 800 νευρώνες	1,60
ΝΔ με Dropout	Χρήση Λογιστικής Παλινδρόμηση	3 στρώματα, 1024 νευρώνες	1,35
ΝΔ με Dropout	ReLUs	3 στρώματα, 1024 νευρώνες	1,25
ΝΔ με Dropout + max-norm	ReLUs	3 στρώματα, 1024 νευρώνες	1,06
ΝΔ με Dropout + max-norm	ReLUs	3 στρώματα, 2048 νευρώνες	1,04
ΝΔ με Dropout + max-norm	ReLUs	2 στρώματα, 4096 νευρώνες	1,01
ΝΔ με Dropout + max-norm	ReLUs	2 στρώματα, 8192 νευρώνες	0,95

1.4.4.3.5 Αύξηση σετ δεδομένων (dataset augmentation)

Ο καλύτερος τρόπος να επιτευχθεί γενίκευση σε έναν οποιοδήποτε αλγόριθμο μηχανικής μάθησης είναι να εκπαιδευτεί με περισσότερα δεδομένα [21]. Δυστυχώς, το πλήθος των υπαρχόντων δεδομένων και ειδικά των δεδομένων με ετικέτες είναι περιορισμένο. Ένας τρόπος να παρακαμφθεί αυτό το πρόβλημα είναι η τεχνητή δημιουργία δεδομένων. Αυτή η μέθοδος ιδανική για εφαρμογές κατηγοριοποίησης αλλά δεν ενδείκνυται για όλους τους τύπους δεδομένων. Ο ιδανικότερος τύπος δεδομένων είναι προφανώς οι εικόνες οι οποίες μπορούν ευκολά να αλλάξουν υπό την εφαρμογή διαφόρων μετασχηματισμών και άρα να παράξουν νέες εισόδους προς εκπαίδευση. Η μέθοδος dataset augmentation χαρακτηριστικά, χρησιμοποιείται από το δίκτυο CNN LeNet-5, το οποίο αναλύεται σε επόμενο κεφάλαιο, με μεγάλη επιτυχία. Πέρα από την εφαρμογή του σε εικόνες επιτυχία έχει συναντηθεί και σε εφαρμογές αναγνώρισης ήχου με εισαγωγή θορύβου.

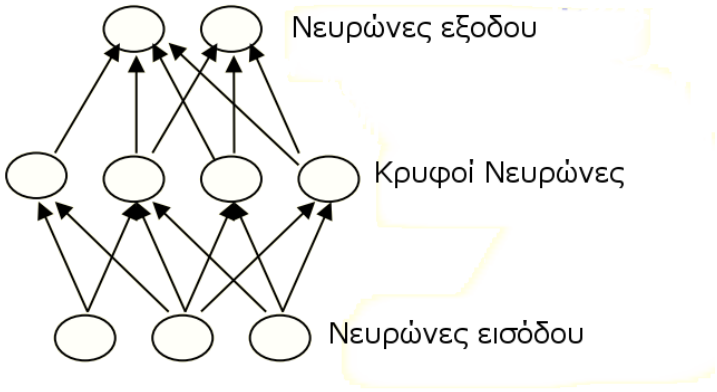
1.5 Αρχιτεκτονικές

Οι διάφορες αρχιτεκτονικές Νευρωνικών δίκτυών μπορούν να ταξινομηθούν ακολουθώντας δυο διαφορετικές προσεγγίσεις. Η μία τις ταξινομεί ανάλογα με τον τρόπο με τον οποίο κατασκευάζονται και η άλλη ανάλογα με την μέθοδο που ακολουθείται για την εκπαίδευση τους.

Όσον αφορά την ταξινόμηση τους ανάλογα με τρόπο κατασκευής τους μια αρχιτεκτονική μπορεί να καταταγεί σε μία από της ακόλουθες βασικές κατηγορίες:

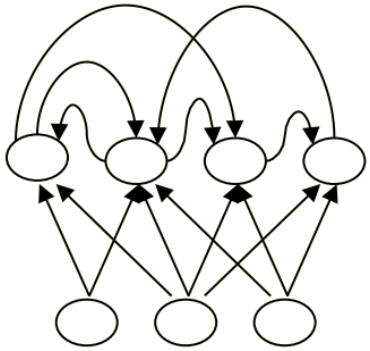
- **Έμπροσθεν τροφοδοτούμενα δίκτυα (Feed-Forward Networks FFN):** [16] Οι αρχιτεκτονικές τύπου FF είναι οι πλέων διαδομένες για πρακτικές εφαρμογές. Αποτελούνται από 3 ή περισσότερα στρώματα. Το πρώτο στρώμα είναι το λεγόμενο

στρώμα εισόδου από το οποίο τροφοδοτείται το δίκτυο με δεδομένα εισόδου. Στην συνέχεια το δίκτυο αποτελείται από ένα ή περισσότερα κρυφά στρώματα (σε περίπτωση πολλαπλών κρυφών στρωμάτων η αρχιτεκτονική αποκαλείται βαθιά) και στο τέλος του δικτύου υπάρχει ένα στρώμα εξόδου. Τα δίκτυα αυτού του τύπου υπολογίζουν μια ακολουθία μετασχηματισμών οι όποιοι αλλάζουν τις ομοιότητες μεταξύ των περιπτώσεων. Οι ενεργοποιήσεις των νευρώνων του εκάστοτε στρώματος αποτελούν μη γραμμική συνάρτηση των ενεργοποιήσεων του προηγούμενου στρώματος.



Εικόνα 1.15: Γράφος Έμπροσθεν Τροφοδοτούμενου Δικτύου (FFN) [16]

- **Δίκτυα συναρτήσεως ριζοσπαστικής βάσης (Radical Basis Function Network (RBF)):** Τα δίκτυα αυτά υλοποιούν συναρτήσεις ριζοσπαστικής βάσης οι οποίες αποτελούν ισχυρές τεχνικές για παρεμβολή στον πολυδιάστατο χώρο. Τα δίκτυα αυτά δεν θα αναλυθούν περεταίρω στην παρούσα εργασία.
- **Αναδρομικά δίκτυα (Recurrent Networks RNN):** [16] Τα δίκτυα αυτά έχουν κατευθυνόμενους κύκλους στον γράφο συνδέσεων τους. Δηλαδή, είναι δυνατό ακολουθώντας έναν μονοπάτι μεταξύ νευρώνων να καταλήξει κάποιος στο σημείο εκκίνησης του. Τα δίκτυα αυτά αν και πιο ρεαλιστικά από βιολογικής απόψεως μπορούν να έχουν περίπλοκες δυναμικές και άρα η εκπαίδευση τους να καθιστάτε δύσκολη. Παρόλα αυτά υπάρχουν αρκετές ενδιαφέρουσες, αποτελεσματικοί μέθοδοι για την εκπαίδευση τους.



Εικόνα 1.16: Γράφος Αναδρομικού Δικτύου (RNN) [16]

- **Συμμετρικά συνδεδεμένα δίκτυα (Symmetrically Connected Networks SCN):** Οι αρχιτεκτονικές τέτοιου τύπου είναι παρόμοιες με τα RNN με την διαφορά όμως ότι οι διανευρωνικές συνδέσεις είναι συμμετρικές, έχουν δηλαδή κοινές παραμέτρους και

στις δύο κατευθύνσεις. Η συμμετρικής τους φύση τίς κάνει πιο εύκολες στην εκπαίδευση τους αλλά με αντίτιμο τον περιορισμό των λειτουργειών τους καθώς υπακούν σε μια ενεργειακή συνάρτηση. Τα συμμετρικά συνδεδεμένα δίκτυα χωρίς κρυφούς νευρώνες ονομάζονται «Hopfield δίκτυα».

- **Συμμετρικά συνδεδεμένα δίκτυα με κρυφούς νευρώνες (Symmetrically Connected Networks with hidden Units (Boltzmann machines) SCNwH):** Τα δίκτυα αυτού του τύπου αποκαλούνται μηχανές Boltzmann. Είναι πολύ πιο ισχυρά μοντέλα από τα δίκτυα Hopfield αλλά λιγότερο ισχυρά από τα RNN. Μεγάλο ενδιαφέρον παρουσιάζει ο απλός αλγόριθμος εκμάθησης που υλοποιούν.
- **Υβριδικές:** Οι αρχιτεκτονικές αυτές συντίθενται από αρχιτεκτονικές που ανήκουν σε μία από τις παραπάνω κατηγορίες. Η κατηγορία αυτή ουσιαστικά συμπίπτει με την αντίστοιχη κατηγορία της μεθόδου κατηγοριοποίησης ως προς την τεχνική μάθησης.

Η άλλη μεθοδολογία κατηγοριοποίησής των διαφόρων νευρωνικών δικτυών είναι με βάση την τεχνική μάθησης που υλοποιούν [12]:

- **Δίκτυα Μη-Επιβλεπόμενης ή Παραγωγικής Μάθησης:** Τα δίκτυα αυτά προορίζονται για εφαρμογές κατά τις οποίες απαιτείται ανίχνευση συσχετίσεων υψηλού επιπέδου των, προς παρατήρηση/ορατών, δεδομένων, με σκοπό την ανάλυση τους ή της σύνθεσης προτύπων. Τέτοιου είδους δίκτυα χρησιμοποιούνται όταν δεν υπάρχει πληροφορία σχετικά με τις ετικέτες των κλάσεων στις οποίες εμπίπτουν τα δεδομένα. Σε αυτή την κατηγορία ουσιαστικά εμπίπτουν και οι κατηγορίες της μη-επιβλεπόμενη μάθησης χαρακτηριστικών (Unsupervised feature learning) καθώς και της μάθησης αναπαραστάσεων (representation learning). Όταν τέτοια δίκτυα χρησιμοποιούνται στην παραγωγική τους λειτουργεία (παραγωγή δειγμάτων εμμέσου δειγματοληψίας από άλλα δίκτυα) μπορούν να χρησιμοποιηθούν για τον χαρακτηρισμό των από κοινού στατιστικών κατανομών (joint statistical distributions) ορατών δεδομένων και των σχετικών κλάσεων τους, εάν αυτές υπάρχουν.
- **Δίκτυα Επιβλεπόμενης Μάθησης:** Οι αρχιτεκτονικές νευρωνικών δικτύων αυτής της κατηγορίας χρησιμοποιούνται για κατηγοριοποίηση προτύπων, συχνά μέσω του χαρακτηρισμού των μεταγενέστερων κατανομών (posterior distributions) κλάσεων οι οποίες έχουν ανατεθεί στα ορατά δεδομένα. Ουσιαστικά δίδεται μαζί με τα δεδομένα προς εκπαίδευση και μια ετικέτα (label) η οποία προσδιορίζει την φύση της εισόδου (π.χ. Δίδετε μια εικόνα ενός ψηφίου συνοδευόμενη από τον αριθμό του ψηφιού που αυτή εικονίζει). Οι ετικέτες αυτές είναι πάντα διαθέσιμες για εφαρμογές επιβλεπόμενής μάθησης. Αποκαλούνται επίσης διακρίνοντα δίκτυα (discriminative networks).
- **Υβριδικά Δίκτυα:** Τα δίκτυα αυτά έχουν ως σκοπό την διάκριση (discrimination) η οποία όμως, σε αντίθεση με τα δίκτυα επιβλεπόμενης μάθησης, υποβοηθάτε, συχνά μάλιστα σε σημαντικό βαθμό, από τα αποτελέσματα παραγωγικής/μη-επιβλεπόμενης μάθησης. Αυτό μπορεί να επιτευχθεί μέσω της καλύτερης βελτιστοποίηση και/είτε κανονικοποίησης των αρχιτεκτονικών της προηγούμενης (επιβλεπόμενης μάθηση) κατηγορίας. Ο στόχος αυτός μπορεί επίσης να επιτευχθεί με χρήση διακρινόντων κριτήριων επιβλεπόμενης μάθησης σε οποιοδήποτε δίκτυο της κατηγορίας της μη επιβλεπόμενης/παραγωγική μάθησης. Ουσιαστικά λοιπόν τέτοιου είδους αρχιτεκτονικές προ-εκπαιδεύουν ένα δίκτυο επιβλεπόμενης μάθησης με ένα δίκτυο μη-επιβλεπόμενης μάθησης με σκοπό την καλύτερη αφομοίωση των δεδομένων.

Στο παρόν κεφάλαιο έχει επιλεχτεί η πρώτη μεθοδολογία κατηγοριοποίησης Νευρωνικών Δικτύων. Τα δίκτυα που θα αναλυθούν λοιπόν, κατηγοριοποιούνται ανάλογα με τον τρόπο κατασκευής τους. Εσκεμμένα έχει προτιμηθεί το κάθε κεφάλαιο να τιτλοφορείται με την τυπική, αγγλική, ονομασία του εκάστοτε δικτύου. Ο λόγος της επιλογής αυτής είναι η διευκόλυνση του αναγνώστη καθώς , πολλά εκ των υπαρχόντων δικτύων δεν έχουν μεταφραστεί επίσημα ακόμα. Ακολουθεί μια σύντομη αναφορά των αναφερόμενων αρχιτεκτονικών νευρωνικών δικτύων στον Πίνακας 1.4.

Πίνακας 1.4: Νευρωνικά Δίκτυα που αναλύονται στην παρούσα εργασία

Αρχιτεκτονική	Συντομογραφία	Κατηγορία μάθησης	Κατηγορία Κατασκευής
Convolutional Neural Networks	CNN	Επιβλεπόμενη	FFN
Autoencoders Γενικά	AE/ANN	Μη-Επιβλεπόμενη	FFN
Denoising Autoencoders	dA/dAE	Μη-Επιβλεπόμενη	FFN
Stacked Autoencoders	SA/SAE	Μη-Επιβλεπόμενη	FFN
Emphasized Denoising Autoencoders	EdA (?)	Μη-Επιβλεπόμενη	FFN
k-Sparse Autoencoders	-	Μη-Επιβλεπόμενη	FFN
Sparse Autoencoders	-	Μη-Επιβλεπόμενη	FFN
Transforming-Autoencoders	-	Μη-Επιβλεπόμενη	FFN
Boltzmann Machines	BM	Μη-Επιβλεπόμενη	SCNwH
Deep Boltzmann Machines	DBM	Μη-Επιβλεπόμενη	SCNwH
Restricted BM	RBM	Μη-Επιβλεπόμενη	SCNwH
Deep Belief Network	DBN	Μη-Επιβλεπόμενη	SCNwH
Recurrent Neural Networks Γενικά	RNN	Μη-Επιβλεπόμενη/Επιβλεπόμενη	RNN
Deep RNN Γενικά	DRNN	Μη-Επιβλεπόμενη/Επιβλεπόμενη	RNN
Bi-directional RNN	BRNN	Επιβλεπόμενη	RNN
Long Short Term Memory	LSTM	Επιβλεπόμενη	RNN
Continuous time RNN	CTRNN	Επιβλεπόμενη	RNN
Echo State Networks	ESN	Επιβλεπόμενη	RNN
Time-Wrapping-Invariant ESN	TWI-ESN	Επιβλεπόμενη	RNN
Leaky-ESN	Leaky-ESN	Επιβλεπόμενη	RNN

1.5.1 Feed Forward Networks (FFN)

[37] Τα έμπροσθεν τροφοδοτούμενα νευρωνικά δίκτυα, είναι αλγόριθμοι επιβλεπόμενης μάθησης οι οποία μάλιστα ήταν από τους πρώτους καθώς και πιο επιτυχημένους αλγορίθμους μάθησης. Τα δίκτυα αυτά συχνά αποκαλούνται και βαθιά δίκτυα (αν έχουν

πάνω από έναν κρυφό στρώμα), Πολύστρωματικό Perceptron (Multi-Layer Perceptron MLP) ή απλώς, νευρωνικά δίκτυα. Τα δίκτυα αυτά μπορούν να μάθουν ισχυρούς μη-γραμμικούς μετασχηματισμούς με μεγάλη επιτυχία. Με αρκετούς κρυφούς νευρώνες είναι δυνατό μάλιστα, να παραστήσουν, αυθαίρετα περίπλοκες, ομαλές συναρτήσεις. Αυτό επιτυγχάνεται εμμέσου μίας διαδικασίας διαρκούς παραγωγής απλουστέρων μη-γραμμικών μεμαθημένων μετασχηματισμών. Μετασχηματίζοντας, λοιπόν, τα δεδομένα με μη-γραμμικό τρόπο σε ένα νέο χώρο, ένα πρόβλημα κατηγοριοποίησης το οποίο δεν ήταν γραμμικά διαχωρίσιμο (και άρα μη-επιλύσμο από έναν γραμμικό κατηγοριοποιητή (linear classifier)) καθιστάτε δυνατό να διαχωριστεί.

Προκειμένου να κατασκευαστεί ένα έμπροσθεν τροφοδοτούμενο δίκτυο πρέπει να γίνουν οι εξής επιλογές (κάποιες από αυτές τις επιλογές τις μοιράζεται και με άλλες κατηγορίες δικτύων όπως RNN):

- Επιλογή Νευρώνων εισόδου, εξόδου ή αλλιώς επιλογή των συναρτήσεων εισόδου-εξόδου των μονάδων του δικτύου καθώς και επιλογή κρυφών μονάδων. Κάποιες από τις πιο βασικές επιλογές έχουν δοθεί στο κεφάλαιο 1.3.
- Επιλογή συναρτήσεως απώλειας ή και συναρτήσεως λογαριθμικής πιθανοφάνειας. Η πιο κοινή συνάρτηση απώλειας είναι αυτή του μέσου τετραγωνικού σφάλματος αλλά μεγάλη επιτυχία έχουν γνωρίσει και μοντέλα που χρησιμοποιούν δεσμευμένη πιθανότητα και Softmax.
- Προαιρετικά χρήση βελτιστοποίησής και κανονικοποίησης.

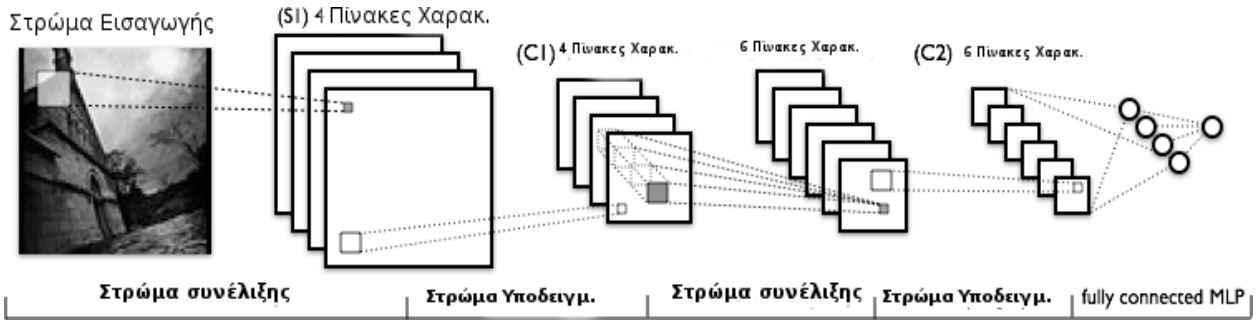
1.5.1.1 Convolutional neural networks (CNN)

Τα CNN (νευρωνικά δίκτυα συνέλιξης) είναι ένα είδος έμπροσθεν τροφοδοτούμενου δίκτυού του οποίο ο εκάστωτε νευρώνας είναι στοιβαγμένος, με τέτοιο τρόπο, ώστε να ανταποκρίνεται σε αλληλεπικαλυπτόμενες περιοχές ενός οπτικού πεδίου. Τα CNN είναι εμπνευσμένα από βιολογικές διαδικασίες και συγκεκριμένα από την δομή του οπτικού συστήματος και αποτελούν παραλλαγή των πολυστρωματικών perceptron (MLP) και άρα εγγενώς χρειάζονται ελάχιστη προ-επεξεργασία. Χρησιμοποιούνται κυρίως για αναγνώριση εικόνας και βίντεο. Συγκρινόμενα με άλλα έμπροσθεν τροφοδοτούμενα δίκτυα με στρώματα όμοιων διαστάσεων, έχουν λιγότερες παραμέτρους και συνδέσεις με αποτέλεσμα η εκπαίδευση τους να καθίσταται πιο εύκολη ενώ η θεωρητικά, βέλτιστη τους απόδοση παραμένει δυνητικά, ελάχιστα κατώτερη των άλλων FFN.

Αναπτύχθηκαν το 1980 από τον Kunihiko Fukushima. Από τότε έχουν βελτιωθεί από πολλούς επιστήμονες. Όταν χρησιμοποιούνται για αναγνώριση εικόνας τα CNN αποτελούνται από μικρές συστάδες νευρώνων (τα αποκαλούμενα πεδία υποδοχής, receptive fields) οι οποίοι «κοιτάνε» σε μικρές περιοχές μιας εικόνας προς αναγνώριση. Τα αποτελέσματα των πεδίων υποδοχής στοιβάζονται και άρα επικαλύπτονται προκείμενου να υπάρξει καλύτερη αναπαράσταση της αρχικής εικόνας. Λόγο αυτής τους της ιδιότητα τα CNN μπορούν να αναγνωρίσουν εικόνες παρά τυχών αλλοιώσεις που μπορεί να έχούν υποστεί. Τα CNN μπορεί ακόμα να περιέχουν τοπικά ή και καθολικά pooling layers (στρώματα συγκέντρωσης) τα οποία χρησιμοποιούνται για να συγκεντρώσουν τις εξόδους των πολλαπλών πεδίων υποδοχής. Αποτελούνται επίσης από διαφόρους συνδυασμούς convolutional layers (στρωμάτων συνέλιξης) και fully connected layers (πλήρως συνδεδεμένων στρωμάτων). Τυπικά αποτελούνται από πέντε έως εφτά τέτοια στρώματα.

1.5.1.1.1 Χαρακτηριστικά

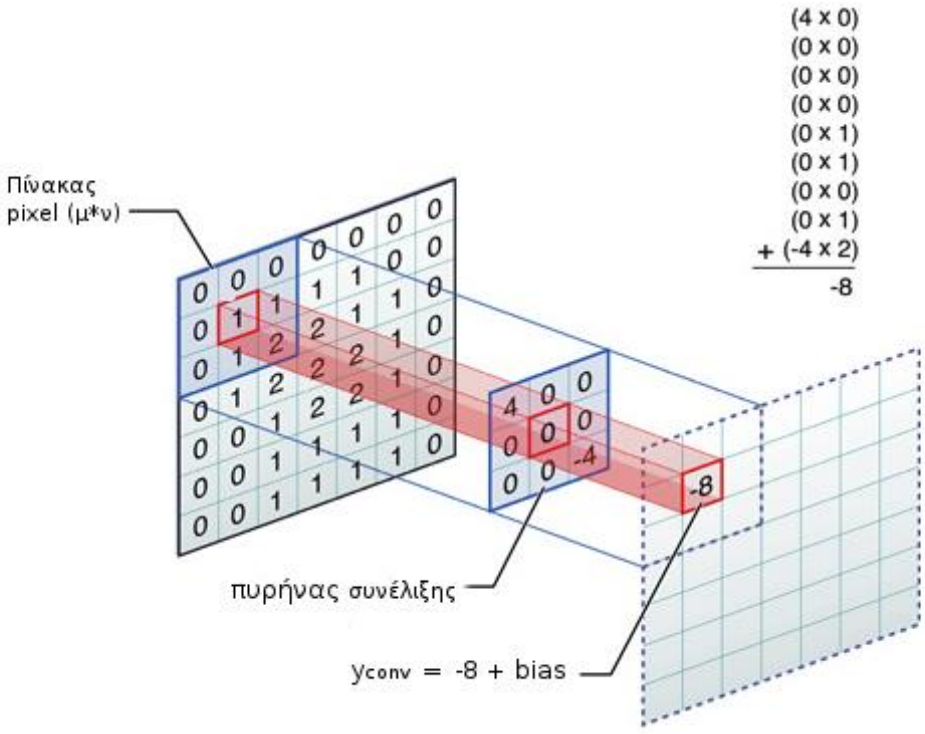
Τα CNN όπως αναφέρθηκε και προηγούμενος, αποτελούνται από μια ακολουθία από στρώματα, όπου κάθε στρώμα υλοποιεί είτε, λειτουργεία συνέλιξης (convolution), είτε υποδειγματοληψίας (subsampling) είτε είναι ένα απλό πλήρες συνδεδεμένο στρώμα (fully connected).



Εικόνα 1.17: CNN πέντε στρωμάτων [38]

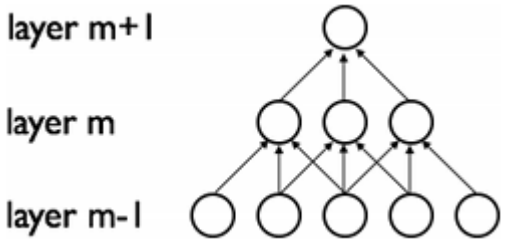
1.5.1.1.1.1 Στρώμα Συνέλιξης

Ένα στρώμα συνέλιξης λειτουργεί ως εξαγωγέας χαρακτηριστικών [38]. Η έξοδος ενός τέτοιου στρώματος είναι προϊόν της συνέλιξης της μια η περισσοτέρων εξόδων του προηγούμενου στρώματος (όντας FFN) και έχει έναν η περισσότερους πίνακες (πυρήνες) σταθερού μεγέθους. Αναλυτικά, ένα τέτοιο στρώμα, λαμβάνει ένα πίνακα (pixel) $m * n$ ως είσοδο. Ο πίνακας αυτός θα υποστεί συνέλιξη με Ν πυρήνες μεγέθους $k * k$ και στην συνέχεια θα προστεθεί στο αποτέλεσμα μία εκ των Ν διαθέσιμων τιμών σταθεράς πόλωσης. Άρα σαν έξοδο προκύπτουν Ν πίνακες μεγέθους $(m - k + 1)$. Αυτή η διαδικασία έχει ως σκοπό να επιτύχει αναγνώριση χαρακτηριστικών που ανήκουν σε ένα από τα Ν χαρακτηριστικά που αναπαριστούν οι πυρήνες. Με αυτή την μέθοδο το υπολογιστικό κόστος του δικτύου μειώνεται δραματικά.



Εικόνα 1.18: Σχηματική Αναπαράσταση Διαδικασίας Συνέλιξης [39]

Προς αποφυγή πολυπλοκότητας κατά την εκπαίδευση ενός τέτοιου δικτύου τα CNN εκμεταλλεύονται την χωρικά τοπική συσχέτιση εφαρμόζοντας ένα πρότυπο τοπικής συνδεσιμότητας μεταξύ των νευρώνων διαδοχικών στρωμάτων δηλαδή τα στρώματα συνέλιξης ενός CNN δεν είναι πλήρως συνδεδεμένα [38].

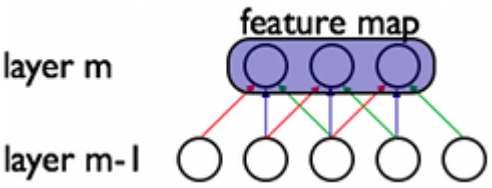


Εικόνα 1.19: Εφαρμογή τοπικής συνδεσιμότητας σε Νευρωνικό Δίκτυο [38]

Με άλλα λόγια οι είσοδοι των κρυφών νευρώνων στο στρώμα m είναι υποσύνολο των εξόδων νευρώνων του στρώματος m-1 . Περιγραφικά, στην Εικόνα 1.19, το στρώμα m-1 είναι το στρώμα εισαγωγής. Οι νευρώνες του επόμενου στρώματος έχουν δεκτικά πεδία (receptive fields) πλάτους 3 και άρα μπορούν να συνδεθούν με μόνο 3 νευρώνες του στρώματος εισαγωγής. Αντίστοιχα νευρώνες στο στρώμα m+1 έχουν δεκτικό πεδίο πλάτους 3 αναφορικά με το στρώμα m, αλλά αναφορικά με το m-1 πλάτους 5. Ο κάθε νευρώνας λοιπόν δεν επηρεάζεται από αλλαγές εκτός του δεκτικού πεδίου του αναφορικά με το στρώμα εισόδου. Με αυτό τον τρόπο επιτυγχάνεται η καλύτερη δυνατή απόδοση για χωρικά τοπικά πρότυπα εισαγωγής (πχ μόρια κατακερματισμένης εικόνας). Παρόλα αυτά όπως διαφαίνεται και στο παραπάνω παράδειγμα οτι όσο περισσότερα στρώματα εισάγονται στην συνέλιξη, τόσο μεγαλύτερη είσοδο επεξεργάζεται το υπό-δίκτυο με

αποτέλεσμα την απώλεια της χωρικής τοπικότητας. Εκμεταλλευμένοι λοιπόν αυτήν την ιδιότητα (Sparse Connectivity) γίνεται συνέλιξη πολλαπλών τέτοιον μικρών υπό-δικτύων υπευθύνων για μια υπό-περιοχή του προς επεξεργασία αντικειμένου. Με αποτέλεσμα να μην είναι απαραίτητο όλα τα στρώματα να είναι πλήρως συνδεδεμένα μεταξύ τους ελαχιστοποιώντας με αυτόν τον τρόπο τις παραμέτρους που χρησιμοποιούνται.

Ένα από τα μεγαλύτερα πλεονεκτήματα της χρήσης CNN είναι η χρήση κοινών φίλτρων h_l μεταξύ των νευρώνων του εκάστοτε στρώματος συνέλιξης l. Άρα οι νευρώνες του εκάστοτε πεδίου έχουν κοινό πίνακα βαρών w^T και κοινή τιμή κατωφλιού b_l και σχηματίζουνε ένα χάρτη χαρακτηριστικών (feature map)



Εικόνα 1.20: Χάρτης Χαρακτηριστικών (feature map) [38]

Στην Εικόνα 1.20 οι νευρώνες του στρώματος m ανήκουν σε έναν χάρτη χαρακτηριστικών καθώς τα βάρη ιδίων χρωμάτων (στην εικόνα οι συνδέσεις αυτές είναι παράλληλες μεταξύ τους) είναι προκαθορισμένο να είναι κοινά. Αυτή η ιδιότητα έχει ως αποτέλεσμα την δραματική μείωση της απαιτούμενη μνήμης και της απαιτούμενη υπολογιστική ισχύος. Για ένα χάρτη χαρακτηριστικών h_l η εξίσωση του φίλτρου τυπικά είναι:

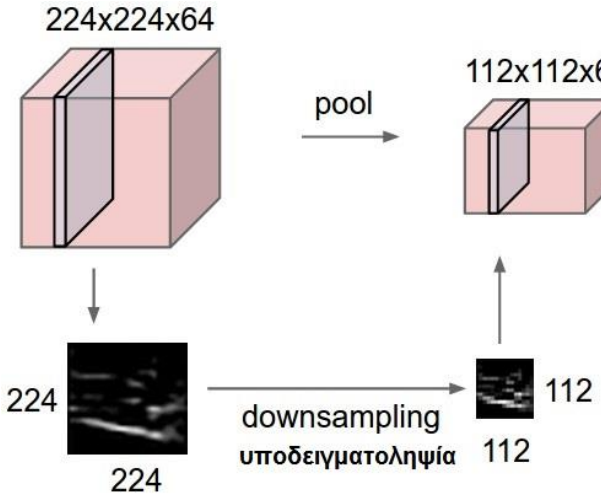
$$h_l^\tau = tanh\{(w_l^\tau * x_l^\tau) + b_l\}$$

Στην πράξη προκειμένου να καθίσταται δυνατή η πλουσιότερή αναπαράσταση των δεδομένων, κάθε κρυφό στρώμα αποτελείται από πολλαπλούς χάρτες χαρακτηριστικών $\{h^{\{k\}}, k = 0, \dots, K\}$.

1.5.1.1.1.2 Στρώμα Υποδειγματοληψίας

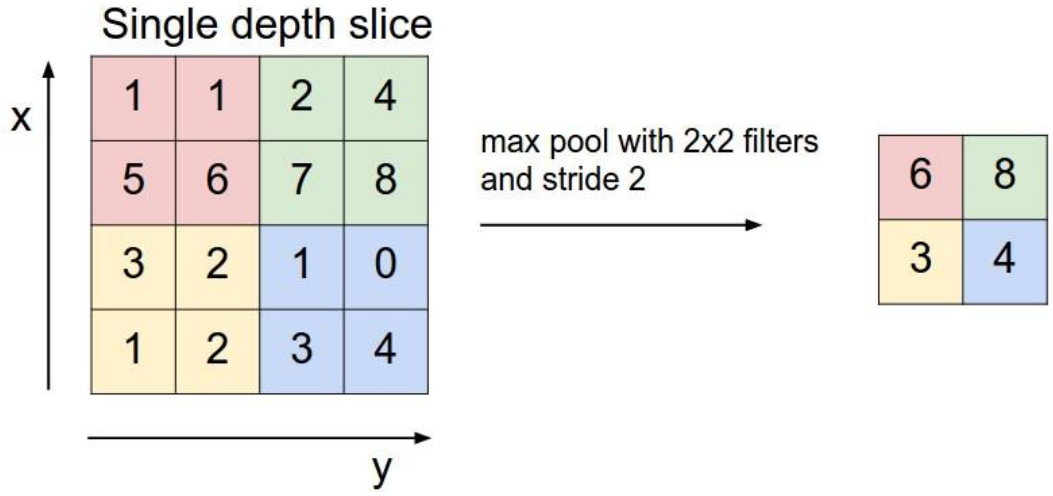
Το στρώμα αυτό αναφέρεται συχνά και ως στρώμα συγκέντρωσης (pooling layer). Η υποδειγματοληψία αναφέρεται σε διαδικασία μείωσης του συνολικού μεγέθους ενός σήματος (συμπίεση ήχου, σμίκρυνση εικόνας κ.ο.κ.). Ουσιαστικά το στρώμα αυτό, είναι υπεύθυνο για τον συνοψισμό των εξόδων γειτονικών ομάδων νευρώνων σε έναν κοινό πυρήνα. Μπορεί να αναπαρασταθεί σαν ένα πλέγμα μονάδων συγκέντρωσής σε απόσταση s (stride) μονάδες (π.χ. pixels) μεταξύ τους. Κάθε τέτοιο πλέγμα είναι υπεύθυνο για τον συνοψισμό μιας περιοχής $z * z$ (υπο-περιοχή εξόδου στρώματος συνέλιξης). Αν λοιπόν τεθεί $z = s$ οι έξοδοι του στρώματος δεν επικαλύπτονται, τεχνική που συναντάται στις περισσότερες εφαρμογές CNN. Αντιθέτως αν $s < z$ τότε υπάρχει επικάλυψη των εξόδων του στρώματος επιτυγχάνοντας την διαδικασία σε απώλεια ακρίβειας.

Μια χαρακτηριστική εφαρμογή του στρώματος αυτού, ονομάζεται μέγιστη συγκέντρωση (max pooling) όπως φαίνεται και στην Εικόνα 1.21 . Η μέθοδος αυτή χρησιμοποιείτε στα LeNet CNN (1.5.1.1.2.1).



Εικόνα 1.21: Υποδειγματοληψία 64 δειγμάτων εικόνας από 224*244px σε 112*112px [40]

Αυτή η διαδικασία περιλαμβάνει τον διαχωρισμό ενός πίνακα (έξοδος στρώματος συνέλιξης) σε μικρές ομάδες, μη επικαλυπτόμενων, υπο-πινάκων. Στην συνέχεια για κάθε μια απ' αυτές τις ομάδες επιλέγεται ο υπο-πίνακας με την μεγαλύτερη τιμή και επανενώνονται σε έναν πίνακα μεγέθους μικρότερου του αρχικού όπως φαίνεται και στην Εικόνα 1.22.



Εικόνα 1.22: Διαδικασία Max Pool [40]

Στην πράξη λοιπόν τα στρώματα υποδειγματοληψίας επιτρέπουν στα στρώματα συνέλιξης αν εφαρμόσουν φίλτρα σε γενικότερες περιοχές της εισόδου με αποτέλεσμα την επιτυχή αναγνώριση προτύπων ακόμα και σε περίπτωση στρέβλωσης τους.

1.5.1.1.1.3Πλήρες Συνδεδεμένο στρώμα

Υλοποιεί ένα ή περισσότερα έμπροσθεν τροφοδοτούμενα νευρωνικά δίκτυα και αποτελεί τον τελευταίο σταθμό ενός CNN. Το στρώμα αυτό είναι υπεύθυνο για την επίλυση του

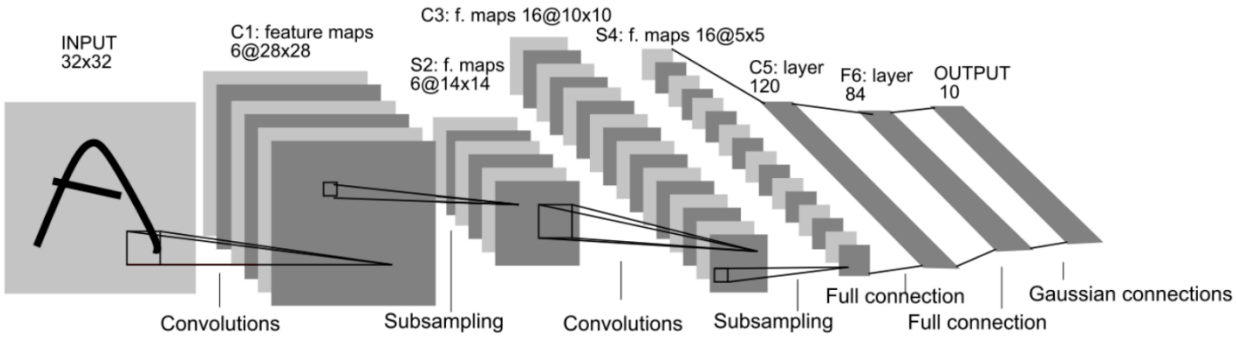
προβλήματος χρησιμοποιώντας ως εισόδους το αποτέλεσμα που παράχθηκαν από τα προηγούμενα στρώματα. Τυπικά γίνεται χρήση MLPs αλλά και πλήρως συνδεδεμένων στρωμάτων συνέλιξης.

1.5.1.1.2 Υλοποιήσεις

Στο υπο-κεφάλαιο αυτό παρουσιάζονται δύο υλοποιήσεις CNN οι οποίες έχουν κερδίσει διάφορους διαγωνισμούς αναγνώρισης εικόνας σε μερικές περιπτώσεις μάλιστα, ξεπερνώντας κάθε προηγούμενη απόδοση.

1.5.1.1.2.1LeNet5

Μια από τις πιο δημοφιλείς υλοποιήσεις CNN είναι το LeNet5 [38]. Αναπτύχθηκε από τους Y. LeCun, L. Bottoy, Y. Bengio και P. Hahhner το 1998 και από τότε έχει υποβληθεί σε σημαντική βελτίωση. Δέχεται εικόνες αριθμών 32*32 (χρήση MNIST dataset) και προβλέπει τον ψηφίο που αναπαριστά η εικόνα με μεγάλη επιτυχία. Για την ενημέρωση των βαρών χρησιμοποιείται ο αλγόριθμος οπισθοδρομική διάδοσης σφάλματος. Το LeNet-5 είναι ένα CNN έξι στρωμάτων (συν ένα στρώμα εξόδου δέκα νευρώνων) με την εξής δομή:



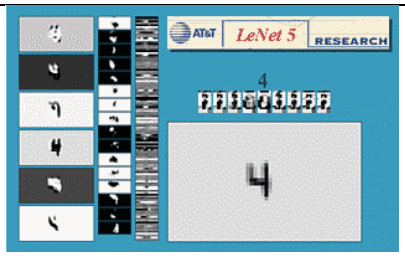
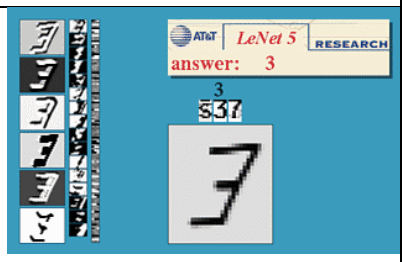
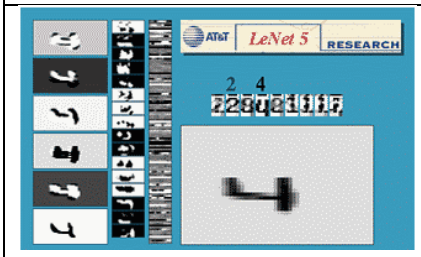
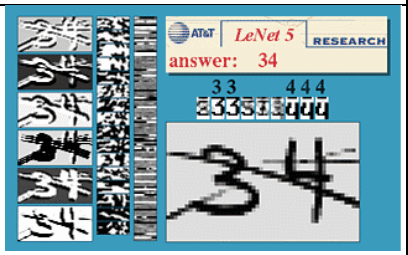
Εικόνα 1.23: LeNet5 CNN [98]

- **Στρώμα C1:** Στρώμα συνέλιξης με έξι χάρτες χαρακτηριστικών μεγέθους $28 * 28$. Κάθε μόριο του C1 έχει δεκτικό πεδίο 5*5 προς το στρώμα εισόδου. Λόγο της χρήσης κοινών βαρών και μερικής σύνδεσης υπάρχουν μόνο $(5 * 5 + 1) * 6 = 156$ παράμετροι προς εκπαίδευση. Ενδεικτικά αν το στρώμα ήταν πλήρως συνδεδεμένο θα υπήρχαν $28 * 28 * (5 * 5 + 1) * 6 = 122304$ παράμετροι.
- **Στρώμα S2:** Στρώμα Υποδειγματοληψίας με έξι χάρτες χαρακτηριστικών μεγέθους $14 * 14$ και 2*2 μη επικαλυπτόμενα δεκτικά πεδία. Υπάρχουν $6 * 2 = 12$ παράμετροι προς εκπαίδευση και $14 * 14 * (2 * 2 + 1) * 6 = 5880$ συνδέσεις.
- **Στρώμα C3:** Στρώμα συνέλιξης με 16 χάρτες χαρακτηριστικών μεγέθους $10 * 10$. Κάθε μόριο του C3 είναι συνδεδεμένο με αρκετά άλλα μόρια. Τα δεκτικά πεδία είναι μεγέθους $5 * 5$. Υπάρχου 1516 παράμετροι προς εκπαίδευση και 151500 συνδέσεις.
- **Στρώμα S4:** Στρώμα Υποδειγματοληψίας με δεκαέξι χάρτες χαρακτηριστικών μεγέθους $5 * 5$ και συνδέεται με τον αντίστοιχο 2*2 δεκτικό πεδίο του C3. Υπάρχουν $16 * 2 = 32$ παράμετροι προς εκπαίδευση και $5 * 5 * (2 * 2 + 1) * 16 = 2000$ συνδέσεις.

- **Στρώμα C5:** Στρώμα συνέλιξης με 120 χάρτες χαρακτηριστικών μεγέθους $1 * 1$. Κάθε μόριο του C3 με κάθε έναν από τα δεκαέξι $5 * 5$ δεκτικά πεδία του S4. Υπάρχουν $120 * (16 * 25 + 1) = 48120$ παράμετροι προς εκπαίδευση και ισάριθμες συνδέσεις (καθώς είναι πλήρως συνδεδεμένο).
- **Στρώμα F6:** Πλήρως συνδεδεμένο στρώμα με 84 μόρια. Άρα υπάρχουν 84*(120+1)=10164 παράμετροι προς εκπαίδευση και ισάριθμες συνδέσεις
- **Στρώμα εξόδου:** Δέκα έξοδοι (μια για κάθε δεκαδικό ψηφίο)

Το LeNet-5 είναι ένα από τα πιο ευρέως γνωστά CNN λόγω της καλής τους απόδοσης αναλογικά με των χρόνο εκπαίδευσης. Χαρακτηριστικά εκπαιδεύοντας ένα LeNet με 60000 εικόνες του MNIST dataset επιτυγχάνεται 99,05% ακρίβεια. [41] Με την χρήση 540000 τεχνητά στρεβλωμένων εικόνων και 60000 κανονικών εικόνων του MNIST η ακρίβεια φτάνει στο 99,2% . [42] Στον Πίνακας 1.5 παρουσιάζονται κάποια παραδείγματα αναγνώρισης ψηφίων με LeNet-5 παρά την εφαρμογή αλλοιώσεων.

Πίνακας 1.5: Αναγνώριση αλλοιωμένων αριθμών από LeNet5 [42]

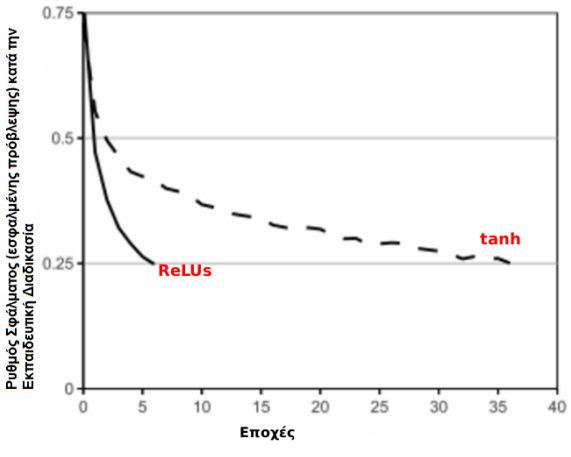
Περιστροφή	Σμίκρυνση	Στρέβλωση
		
Συμπίεση	**Θόρυβός + Διψήφια**	**Πάχυνση**
		

1.5.1.1.2.2 AlexNet

Το μοντέλο αναπτύχθηκε από τους Alex Krizhevsky, Ilya Sutskever, George E. Hinton του πανεπιστημίου του Toronto το 2010. Κατασκευάστηκε με σκοπό την αναγνώριση αντικειμένων του ImageNet dataset με βασικό γνώμονα την επίτευξη γρήγορης εκπαίδευσης και μετέπειτα γρήγορης ταξινόμησης [43] [38]. Παρουσιάζει μεγάλη επιτυχία τόσο σε top-1 error tests (Μια δυνατή πρόβλεψη για κάθε ετικέτα) όσο και σε top-5 error tests (Πέντε δυνατές προβλέψεις για κάθε ετικέτα). Είναι CNN οκτώ στρωμάτων τρία εκ των οποίων είναι πλήρως συνδεδεμένα και πέντε συνέλιξης. Ακολουθούν κάποια ενδιαφέροντα χαρακτηριστικά του συγκεκριμένου CNN.

- **Συνάρτηση ενεργοποίησης**: Με γνώμονα την γρήγορη εκπαίδευση αντί για την χρήση συνάρτησης ενεργοποίησής $tanh(x)$ χρησιμοποιούνται στοχαστικοί διάδικοι

νευρώνες (ReLUs) με συνάρτηση ενεργοποίησης $f(x) = max(0, x)$. Έχει αποδειχτεί ότι η χρήση ReLUs επιτρέπει σε CNN πολλών επιπέδων να εκπαιδεύονται αρκετές φορές πιο γρήγορα από αντίστοιχά με $tanh$ νευρώνες. Για παράδειγμα ένα CNN τεσσάρων επιπέδων με ReLUs επιτυγχάνει 75% επιτυχία κατά την εκπαίδευση έξι φορές πιο γρήγορα από αντίστοιχο CNN με $tanh$ νευρώνες



Διάγραμμα 1.13: Σύγκριση απόδοσης CNN με χρήση σιγμοειδών ($tanh$) και RELU (Επιδιορθωμένων Γραμμικών) Νευρώνων κατά την εκπαιδευτική Διαδικασία. [40]

- **Επικαλυπτόμενη συγκέντρωση:** Τα στρώματα Υποδειγματοληψίας το CNN υλοποιούν επικαλυπτόμενη συγκέντρωση με τις μονάδες συγκέντρωσης στα πλέγματα να απέχουν 2 pixel μεταξύ τους $(s = 2)$ και η κάθε περιοχή προς συνοψισμό έχει μέγεθος 3*3 pixels $(z = 3)$. Αυτή η επιλογή έχει ως αποτέλεσμα βελτίωση στην απόδοση του συστήματος (0,4% λιγότερο ρυθμό λάθους σε top-1 και 0,3% σε top-5.
- **Κανονικοποίηση:** Παρά το γεγονός ότι, τυπικά, τα CNN δεν χρειάζονται κανονικοποίηση της εισόδου προκειμένου να αποφευχθεί ο κορεσμός, στο συγκεκριμένο μοντέλο χρησιμοποιείτε τοπική κανονικοποίηση απάντησης σε συγκεκριμένα στρώματα. Επιτυγχάνεται έτσι περεταίρω βελτίωση στην απόδοση του (11% αντί για 13% ρυθμός σφάλματος με χρήση του CIFAR-10 dataset)
- **Εκμάθηση:** Γίνεται χρήση στοχαστικής κλίσης καθόδου (stochastic gradient descent) με μέγεθος δέσμης 128 δείγματα, ορμή μεγέθους 0,9 και αποσύνθεση βάρους 0,0005. Αποδείχτηκε ότι αυτή η μικρή ποσότητα αποσύνθεσης βάρους ήταν σημαντική για την εκπαίδευση του μοντέλου. Άρα ο κανόνας ενημέρωσης για βάρος w είναι:

$$v_{i+1} := 0.9 * v_i - 0.0005 * \eta * w_i - \eta * \langle \frac{\vartheta C}{\vartheta w_k}|_{w_i} \rangle_{D_i}$$

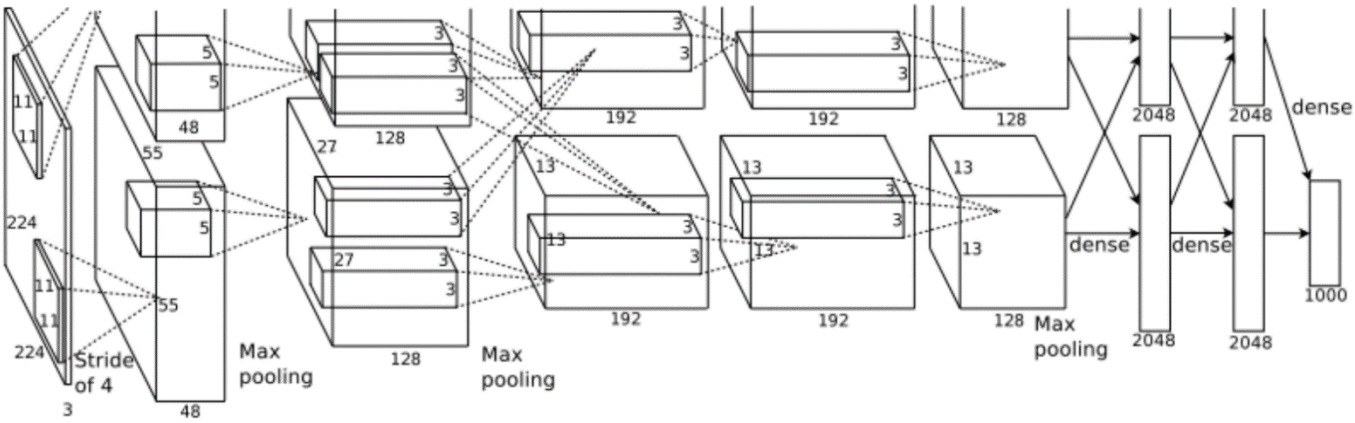
$$w_{i+1} := w_i + v_{i+1}$$

Όπου ι είναι ο δείκτης της τρέχουσας επανάληψης, v είναι παράμετρος της ορμής, η είναι ο ρυθμός εκμάθησης και D_i η ι-οστή δέσμη εισόδων.

- **Αρχικοποίηση:** Τα βάρη έχουν αρχικοποιηθεί γύρο από μια τυποποιημένη κανονική κατανομή με τυπική απόκλιση 0,001. Οι σταθερές πόλωσης (bias) στο

δεύτερο, τέταρτο και πέμπτο στρώμα συνέλιξης έχουν αρχικοποιηθεί με την σταθερά 1. Στα υπόλοιπα στρώματα τα bias είναι 0.

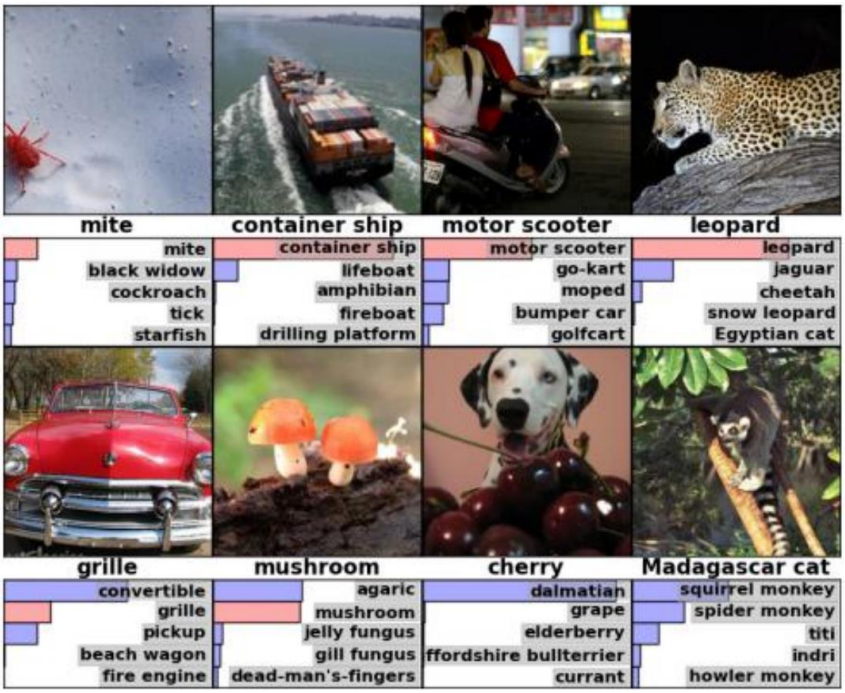
- **Εγκατάλειψη (Dropout):** Αν και ο συνδυασμός πολλών διαφορετικών μοντέλων (μη επιβλεπόμενη προ-εκπαίδευση) είναι ένας πολύ επιτυχημένος τρόπος να μειωθεί ο ρυθμός σφάλματος, η τεχνική αυτή παραμένει πολύ ακριβή για μεγάλα νευρωνικά δίκτυα απαιτώντας μερικές μέρες για την επαρκή εκπαίδευση τους. Ένας άλλος τρόπος να μειωθεί ο χρόνος εκπαίδευσης είναι η χρήση της τεχνικής ομαλοποίησης dropout (1.4.4.3.4). Πρόκειται για μια τεχνική κατά την οποία η έξοδος κάθε νευρώνα γίνεται 0 με πιθανότητα 50%. Έτσι οι νευρώνες που «εγκαταλείπονται» δεν συνεισφέρουν μειώνοντας έτσι το κόστος τόσο της έμπροσθεν διάδοσης όσο και της οπισθοδρομικής. Επίσης μειώνονται περίπλοκες αναπροσαρμογές νευρώνων εφόσον δεν μπορούν πια να βασίζονται στην παρουσία άλλων συγκεκριμένων νευρώνων. Με αποτέλεσμα αυτοί οι νευρώνες να μαθαίνουν πιο εύρωστα χαρακτηριστικά.
- **Δομή:** Πέντε Στρώματα συνέλιξης και τρία πλήρως συνδεδεμένα καθώς και 3 στρώματα υποδειγματοληψίας με max pooling
    - 650.000 νευρώνες
    - 60.000.000 παράμετροι προς εκπαίδευση
    - 630.000.000 συνδέσεις
    - 4096-διάστατο Τελικό στρώμα χαρακτηριστικών



Εικόνα 1.24: AlexNet CNN [43]

- **Επιδώσεις:** Το παραπάνω μοντέλο έχει καταφέρει πολύ υψηλές αποδόσεις, κατά πολύ μεγαλύτερες του ανταγωνισμού. Σε μια δοκιμή που έγινε σε δύο GTX 580 3GB GPUs με εκπαίδευση διάρκειας πέντε έως έξι ημερών χρησιμοποιώντας το ImageNet dataset επιτευχθεί:
    - top-1 error rate: 37.5%
    - top-5 error rate: 17.0%

Χαρακτηριστικά, η δεύτερη νικήτρια ομάδα στον διαγωνισμό ILSVRC-2012 επέτυχε μόνο 26,2% ακρίβεια σε προβλέψεις.



Εικόνα 1.25: Δείγμα top-5 προβλέψεων στο σετ δεδομένων ImageNet [43]

1.5.1.2 Autoencoders

Οι Autoencoders (δίκτυα αυτόματης κωδικοποίησης) είναι δίκτυα που ειδικεύονται στην μη επιβλεπόμενη μάθηση. Ανήκουν σε μία ειδική κατηγόρια Νευρωνικών Δικτύων των οποίον το διάνυσμα εισόδου έχει τις ίδιες διαστάσεις με το διάνυσμα εξόδου. Οι Autoencoders ή αλλιώς auto-associators ή αλλιώς Diabolo networks βασίστηκαν στην ιδέα της «Αραιής Κωδικοποίησης» (Sparse Coding) η οποία προτάθηκε στην βαρυσήμαντή εργασία του Olshausen [44]. Οι Autoencoders μπορούν να στοιβαχτούν σχηματίζοντας βαθιές αρχιτεκτονικές με αποτέλεσμα να βελτιώνεται η λειτουργεία τους. Την ιδιότητα αυτή την κληρονομούν και οι διάφορες παραλλαγές τους. Όταν στοιβάζονται λοιπόν, δηλαδή όταν η έξοδος ενός Autoencoder λαμβάνεται ως είσοδος από έναν άλλον στο ανώτερο στρώμα, τότε λαμβάνουν τον τίτλο Stacked (π.χ. Stacked Denoising Autoencoder, Stacked Sparse Autoencoder κ.ο.κ.).

Χρησιμοποιούνται συνήθως για τον προσδιορισμό της αναπαράστασης ή της κωδικοποίησης των εισαγόμενων δεδομένων (διανύσματα εισόδου) καθώς και για προ-εκπαίδευση δικτύων με επιβλεπόμενη μάθηση (συνήθως οι Autoencoders προ-εκπαίδευσης κανονικοποιούνται [38]). Ένας τρόπος για την επιτυχημένη προ-εκπαίδευση μιας βαθιάς αρχιτεκτονικής είναι η αρχικοποίηση των βαρών όλων τον στρωμάτων (πλην του τελευταίου) με έναν καθαρά μη επιβλεπόμενο τρόπο. Στην συνέχεια εμμέσου μιας άπλησης διαδικασίας η οποία δρα σε κάθε στρώμα ξεχωριστά, το εκάστοτε στρώμα προσπαθεί να ανακατασκευάσει την είσοδο του. [45]

Οι Autoencoders είναι μη γραμμικές μέθοδοι για την εξαγωγή χαρακτηριστικών (feature extraction). Όντας δίκτυα μη-επιβλεπόμενης μάθησης δεν χρησιμοποιούν ετικέτες, άρα τα

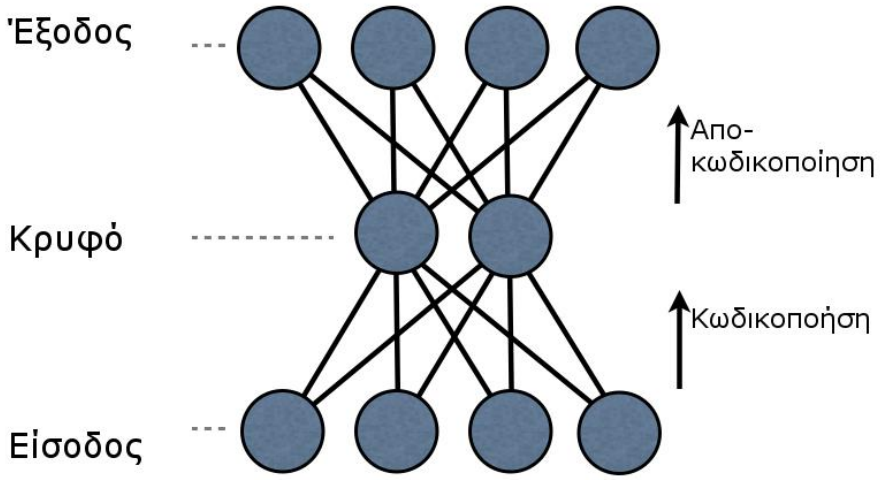
χαρακτηριστικά που εξάγονται στοχεύουν στην καλύτερη αναπαράσταση χρήσιμης πληροφορίας και όχι στην κατηγοριοποίηση της (αν και συχνά οι δύο αυτοί στόχοι συσχετίζονται). Ουσιαστικά εκπαιδεύεται προκειμένου να κωδικοποιήσει μία είσοδο x σε μία αναπαράσταση $c(x)$ έτσι ώστε η είσοδος να μπορείς να ανακατασκευαστεί από αυτή την αναπαράσταση. Έτσι η επιθυμητή έξοδος του δικτύου είναι η ίδια η είσοδος.

Οι Autoencoders συχνά εκπαιδεύονται χρησιμοποιώντας μια εκ των παραλλαγών του αλγορίθμου οπισθοδρομικής διάδοσης σφάλματος και κυρίως της μεθόδου με χρήση στοχαστικής απότομης καθόδου. Παρά το ότι είναι αρκετά αποτελεσματικός στην περίπτωση των βαθιών αρχιτεκτονικών με πολλά κρυφά στρώματα, παρουσιάζονται θεμελιώδη προβλήματα. Όταν δηλαδή τα σφάλματα οπισθοδρομούν και φτάνουν στα πρώτα στρώματα του δικτύου έχουν γίνει υποτυπώδους μεγέθους και η εκπαίδευση γίνεται αναποτελεσματική. Αν και αυτό μπορεί να αντιμετωπιστεί ως ένα σημείο χρησιμοποιώντας πιο εξελιγμένες μορφές του αλγορίθμου τα επιθυμητά αποτελέσματα εκπαίδευσης θα παραμένουν απογοητευτικά. Το πρόβλημα αυτό μπορεί να αντιμετωπιστεί αποτελεσματικότερα αν κάθε στρώμα εκληφθεί ως έναν απλό Autoencoder και προ-εκπαιδευτεί ξεχωριστά από τα υπόλοιπα. Αυτή η τεχνική έχει χρησιμοποιηθεί για την επιτυχή κατασκευή Autoencoders μεγάλου βάθους για την χαρτογράφηση εικόνων σε σύντομους διάδικους κώδικες καθιστώντας δυνατή την γρήγορη ανάκτησή τους (χρησιμοποιώντας κριτήρια αναφορικά με το περιεχόμενο τους) καθώς και για την κωδικοποίηση χαρακτηριστικών ομιλίας (φασματογράφου τύπου) [12] .

Μία άλλη τεχνική που έχει αναπτυχθεί [46] για την εκπαίδευση βαθιών Autoencoders περιλαμβάνει την αντιμετώπιση κάθε ζευγαριού γειτονικών στρωμάτων ως ένα RBM κατά την διαδικασία της προ-εκπαίδευσης έτσι ώστε να προσεγγιστεί μια καλή λύση. Στην συνέχεια με χρήση μια τεχνικής οπισθοδρομικής διάδοσης σφάλματος με σκοπό την εξομάλυνση και άρα την ελαχιστοποίηση του σφάλματος κατά την κωδικοποίηση. Αυτός ο αλγόριθμος έχει χρησιμοποιηθεί για την χαρτογράφηση εικόνων σε σύντομους διάδικους κώδικες με σκοπό την γρήγορη ανάκτηση τους με γνώμονα το περιεχόμενο τους, στην κωδικοποίηση εγγράφων [47] και στην κωδικοποίηση χαρακτηριστικών ομιλίας (σε μορφή σπεκτρογραφίματος) [48].

Όσον αφορά την αρχιτεκτονική τους ένας Autoencoder τυπικά αποτελείται από:

1. Ένα εισαγωγικό στρώμα το οποίο αναπαριστά τα πρωτότυπα δεδομένα ή σε άλλες περιπτώσεις το εισαγόμενο διάνυσμα χαρακτηριστικών (π.χ. pixel μιας φωτογραφίας)
2. Ένα ή περισσότερα κρυφά στρώματα τα οποία αναπαριστούνε τα μεταμορφωμένα χαρακτηριστικά. (Όταν τα κρυφά στρώματα είναι περισσότερα του ενός τότε χαρακτηρίζουμε τον Autoencoder ως βαθύ). Η διάσταση των κρυφών στρωμάτων μπορεί να είναι είτε μικρότερη (όταν ο στόχος είναι η συμπίεση των χαρακτηριστικών) είτε μεγαλύτερη (όταν ο στόχος είναι η χαρτογράφηση των χαρακτηριστικών σε χώρο μεγαλύτερων διαστάσεων) της διάστασης του χώρου εισόδου. Όταν τα κρυφά στρώματα είναι δεν εμπεριέχουν μη-γραμμικότητες τότε το δίκτυο πραγματοποιεί ανάλυση σε κύριες συνιστώσες (PCA [49]). Αντιθέτως όταν τα κρυφά στρώματα εμπεριέχουν μη-γραμμικότητές τότε μπορεί να αντιληφθεί πολύτροπες πτυχές της εισαγόμενης κατανομής. [26]
3. Ένα στρώμα εξόδου το οποίο ταιριάζει με το στρώμα εισόδου έτσι ώστε να είναι δυνατή η ανακατασκευή των δεδομένων.



Εικόνα 1.26: Τυπικό Autoencoder [50]

Όσον αφορά την λειτουργεία ενός Autoencoder [51], σε πρώτο στάδιο (έμπροσθεν χαρτογράφηση) δέχεται μία είσοδο $x \in [0,1]^d$ την οποία χαρτογραφεί (χρησιμοποιώντας έναν κωδικοποιητή) σε μία κρυφή αναπαράσταση $y \in [0,1]^{d'}$ εμμέσου μιας διαδικασίας, ντετερμινιστικής χαρτογράφησης (deterministic mapping). Μια τέτοια διαδικασία θα μπορούσε να έχει ως εξής:

$$y = f_\theta(x) = s(Wx + b)$$

Όπου s είναι μία μή-γραμμική συνάρτιση (π.χ. σιγμοειδής) παραμετροποιημένη με $\theta = \{W, b\}$ όπου W ο $d' * d$ πίνακας συναπτικών βαρών και b το διάνυσμα σταθερών πόλωσης. Στην συνέχεια (στάδιο αντίστροφης χαρτογράφησης) η λανθάνουσα αναπαράσταση y χαρτογραφείται πάλι σε ένα διάνυσμα-ανακατασκευή z (με την χρήση ενός αποκωδικοποιητή) το οποίο έχει το ίδιο σχήμα (ίδιες διαστάσεις) με το x. Η χαρτογράφηση πραγματοποιείται χρησιμοποιώντας έναν παρόμοιο μετασχηματισμό όπως:

$$z = g_{\theta'}(y) = s(W'y + b')$$

Με $\theta = \{W', b'\}$. Το z εκλαμβάνεται ως η πρόβλεψη του x δοθέντος του κωδικού y. Προαιρετικά ο πίνακας συναπτικών βαρών W' της ανάστροφης χαρτογράφησης μπορεί να προκύψει από την αντιστροφή του πίνακας βαρών W της έμπροσθεν χαρτογράφησης $W' = W^\tau$. Αυτή η τεχνική αναφέρεται στην βιβλιογραφία ως δεμένα βάρη (tied weights).

Κάθε είσοδος λοιπόν, $x^{(i)}$ χαρτογραφείτε σε έναν κωδικό $y^{(i)}$ και σε μια ανακατασκευή $z^{(i)}$. Κατά την εκπαίδευση λοιπόν οι παράμετροι θ $(W, b, b'$ και αν δεν χρησιμοποιούνται δεμένα βάρη $W')$ πρέπει να βελτιστοποιηθούν, τυπικά με χρήση στοχαστικής κλίσης καθόδου, ώστε το μέσο σφάλμα ανακατασκευής να ελαχιστοποιηθεί.

$$\theta^*, \theta'^* = arg \min_{\theta,\theta'} \frac{1}{n} \sum_{i=1}^{n} L\left(x^{(i)}, z^{(i)}\right) = arg \min_{\theta,\theta'} \frac{1}{n} \sum_{i=1}^{n} \left(Lx^{(i)}, g_{\theta'}\left(f_\theta\left(x^{(i)}\right)\right)\right)$$

Όπου L είναι μια συνάρτηση απώλειας η οποία επιλέγεται αναλόγως των υποθέσεων που γίνανε αναφορικά με την κατανομή που ακολουθάει το διάνυσμα εισόδου. Παραδείγματα συναρτήσεων απώλειας (σε άλλες βιβλιογραφίες συναντώνται και ως συναρτήσεις κόστους ή συναρτήσεις σφάλματος) που μπορούν να χρησιμοποιηθούν είναι:

- Το μέσω τετραγωνικό σφάλμα

$$L(x, y) = \|x - z\|^2$$

- Η δια-εντροπία (cross-entropy) $L_H(x, z)$ της ανακατασκευής. Η συνάρτηση αυτή προκύπτει από την ερμηνεία των x και z είτε ως διανύσματα bit είτε ως διανύσματα διάδικων πιθανοτήτων (Bernoullis).

$$L_{\mathcal{H}}(x, z) = \mathcal{H}(B_x||B_z) = -\left(\sum_{k=1}^{d} [x_k \log z_k + (1 - x_k) \log(1 - z_k)]\right)$$

Αν το x είναι δυαδικό διάνυσμα, η συνάρτηση $L_H(x, z)$ είναι μια αρνητική λογαριθμική-πιθανοφάνεια της εισόδου x δοθέντων των Bernoulli παραμέτρων z. Η εξίσωση του μέσου σφάλματος ανακατασκευής με χρήση αυτής της συνάρτησης απώλειας γίνεται:

$$\theta^*, \theta'^* = arg \min_{\theta,\theta'} E_{q^0(x)}\left(L_H\left(X, g_{\theta'}\left(f_\theta(X)\right)\right)\right)$$

Όπου $E_{p(x)}\big(f(x)\big) = \int p(x) f(x)$ είναι η προσδοκία και $q^0(X)$ συμβολίζει την εμπειρική κατανομή που συσχετίζεται με τις n δοθείσες εισόδους.

Ακολουθεί ένα παράδειγμα χρήσης ενός Autoencoder: [52] Έστω ότι οι είσοδοι x αντιπροσωπεύουν την πυκνότητα των pixels μιας εικόνας μεγέθους $10 * 10$ (άρα $n = 100$ και $z \in \mathbb{R}^{100}$) και το δίκτυο που χρησιμοποιείται έχει 50 κρυφούς νευρώνες στο στρώμα L_2. Εφόσον το πλήθος των κρυφών νευρώνων είναι μικρότερο των νευρώνων εισόδου, το δίκτυο θα αναγκαστεί να μάθει μία συμπιεσμένη αναπαράσταση της εισόδου καθώς πρέπει να προσπαθήσει να αναπαραστήσει είσοδο μεγέθους 100 pixels έχοντας ως μόνη πληροφορία κρυφή αναπαράσταση $y \in \mathbb{R}^{50}$. Αν η είσοδος ήταν τυχαία τότε η διαδικασία της κωδικοποίησης θα ήταν πολύ δύσκολη. Αντιθέτως αν η είσοδος δεν ήταν τυχαία και κάποια από τα χαρακτηριστικά της συσχετίζονταν τότε ο αλγόριθμος θα μπορούσε «ανακαλύψει» κάποιες από αυτές τις συσχετίσεις.

1.5.1.2.1 Denoising Autoencoders

Ο denoising (αποθορυβοποίητής) Autoencoder είναι μια στοχαστική έκδοση του Autoencoder έχει ως στόχο να ελαχιστοποιήσει το σφάλμα που προκύπτει από την ανακατασκευή της εισόδου. Προκειμένου να το πετύχει αυτό δηλαδή προκειμένου τα κρυφά στρώματα να εξαναγκαστούν να ανακαλύψουν πιο έντονα χαρακτηριστικά της δοθείσας εισόδου και να μην μαθαίνουν απλά την ταυτότητα της, εκπαιδεύουμε το δίκτυο με τέτοιο τρόπο ώστε να ανακατασκευάζει την είσοδο από μία στοχαστικά μετασχηματισμένης έκδοση της. Χρησιμοποιώντας αυτή την μέθοδο μεγιστοποιείται το κατώτερο όριο της λογαριθμικής πιθανότητας της εισόδου [26]. Μια κομβική διαφορά των Denoising από τους απλούς Autoencoders είναι ότι δεν είναι απαραίτητο το στρώμα εισόδου να έχει μεγαλύτερη διάσταση από το κρυφό στρώμα για την ορθή τους λειτουργεία [53]. Η λειτουργεία ενός Denoising Autoencoder μπορεί να συμπυκνωθεί στα παρακάτω βήματα:

- Στοχαστική στρέβλωση: Εφαρμόζεται μια διαδικασία στρέβλωσης στα δεδομένα εισόδου x και γίνονται $x \to \tilde{x}$.Κάποιες από τις διαδικασίες στρέβλωσης που χρησιμοποιούνται είναι [54]:

    - Προσθετικός ισοτροπικός Γκαουσιανός θόρυβος (GS): $\tilde{x}|x \sim N(x, \sigma^2 I)$
    - Θόρυβος συγκάλυψης (Masking Noise MN). Κάποιές εκ των δοθεισών εισόδων (η ποσότητα τους μπορεί να είναι ακόμα και το ½ του συνόλου) γίνονται ίσες με μηδέν και έτσι κάθε πληροφορία που μπορεί να περιείχαν χάνετε.
    - Θόρυβος αλατοπίπερου (Salt-and-paper noise SP): Ένα υποσύνολο ν των στοιχείων της εισόδου x (το οποίο επιλέγεται τυχαία για κάθε είσοδο) αλλάζει τιμή. Το κάθε στοιχείο του ν παίρνει την μέγιστη ή την ελάχιστη επιτρεπτή τιμή του (συνήθως 0 ή 1) με πιθανότητα 50-50.

- Κωδικοποιητής: Προσπαθεί να κωδικοποιήσει την στρεβλωμένη είσοδο διατηρώντας την χρήσιμη πληροφορία της. (Όπου $s(\dots)$ σιγμοειδής συνάρτηση)

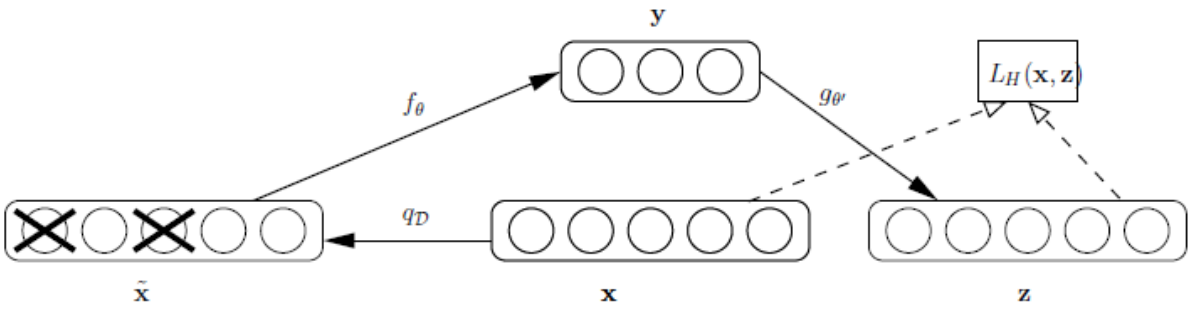
$$y = f_\theta(\tilde{x}) = s(W\tilde{x} + b)$$

- Αποκωδικοποιητής: Στην συνέχεια προσπαθεί να αναστρέψει το αποτέλεσμα της διαδικασίας της στοχαστικής στρέβλωσης που έχει εφαρμοστεί στην είσοδο του αποκωδικοποιητή. Αυτό μπορεί να επιτευχθεί χρησιμοποιώντας της στατιστικές εξαρτίσεις μεταξύ των εισόδων.

$$z = g_{\theta'}(y) = s(Wy + b)$$

Το δίκτυο λοιπόν εκπαιδεύεται προκείμενου να προβλέψει τις απολεσθέντες εισόδους και να «γεμίσει» τα «κενά». Όπως και στους απλούς Autoencoders οι παράμετροι πρέπει να

εκπαιδευτούν προκειμένου η έξοδος του δικτύου z να είναι σο το δυνατόν πιο κοντά στη αναλλοίωτη είσοδο x (μείωση της μεταξύ τους απόστασης). Για τον σκοπό αυτό μπορεί να χρησιμοποιηθεί πάλι η $L_H(x,z) = \mathcal{H}(B_x||B_z)$ (Cross-Entropy) προς ελαχιστοποίηση. Σχηματικά ένας Denoising Autoencoder έχει την ακόλουθη μορφή:



Εικόνα 1.27: Denoising Autencoder [51]

Η βασική διαφορά με τους απλούς Autoencoders λοιπόν, είναι ότι τώρα το z είναι μια ντετερμινιστική συνάρτηση του $\tilde{x}$ και όχι του x και άρα προϊόν μιας στοχαστικής χαρτογράφησης του x. Η κοινή κατανομή μπορεί να οριστεί ως εξής:

$$q^0\left(X, \tilde{X}, Y\right) = q^0(X) q_D\left(\tilde{X}\middle|X\right) \delta_{f_\theta(\tilde{X})}(Y)$$

Όπου η συνάρτηση $\delta_u(v)$ τοποθετεί μαζικά μηδενικά όταν $u \neq v$ και $q^0(X)$ συμβολίζει την εμπειρική κατανομή που συσχετίζεται με τις n δοθείσες εισόδους. Άρα το Y είναι μια ντετερμινιστική συνάρτηση του $\tilde{X}$ και το $q^0\left(X, \tilde{X}, Y\right)$ παραμετροποιείται ως προς θ. Οπότε η συνάρτηση που εκφράζει τον επιθυμητό στόχο του δικτύου, (αν ελαχιστοποιηθεί με χρήση στοχαστικής απότομής κάθοδού) γίνεται:

$$arg \min_{\theta,\theta'} E_{q^0(X,\tilde{X})}\left(L_H\left(X, g_{\theta'}\left(f_\theta\left(\tilde{X}\right)\right)\right)\right)$$

Από την σκοπιά της στοχαστικής απότομης καθόδου πέρα από την επιλογή ενός δείγματος εισόδου από τις εισόδους εκπαίδευσης, πρέπει να παράξει μια τυχαία στρεβλωμένη έκδοση του και να διορθώσει τις παραμέτρους προκειμένου να ανακατασκευαστεί μια μη-στρεβλωμένη έκδοση της στρεβλωμένης εισόδου.

Οι Denoising Autoencoders μπορούν να χρησιμοποιηθούν ως προ-εκπαιδευτές επιβλεπόμενων αρχιτεκτονικών με μια άπληστη προσέγγιση παρόμοια με αυτή των απλών Autoencoders.

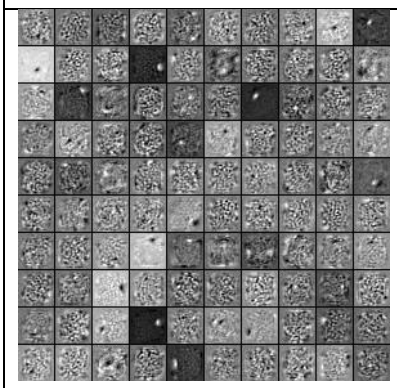
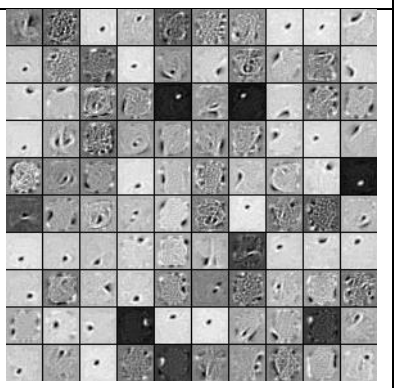
Στους παρακάτω πίνακες παρουσιάζονται δύο ομάδες φίλτρων ανίχνευσης χαρακτηριστικών.

- Πίνακας 1.6: Τα δύο φίλτρα είναι υλοποιημένα [38] με Theano. Το ένα φίλτρο είναι ένας απλός Autoencoder με το εισαγόμενο παράδειγμα να μην έχει θόρυβο.

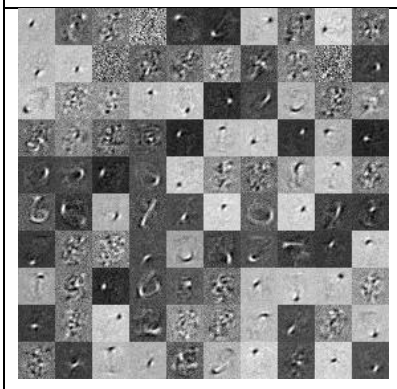
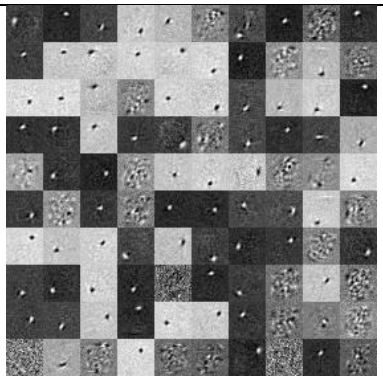
Το άλλο φίλτρο είναι ένας Denoising Autoencoder και στο δείγμα εισόδου έχει εισαχθεί 30% θόρυβος συγκάλυψης (MN).

- Πίνακας 1.7: Τα δύο φίλτρα είναι υλοποιημένα [50] με Python και είναι Denoising Autoencoders με το δείγμα εισόδου του ενός να έχει υποστεί Γκαουσιανό θόρυβο (GS) (500k επαναλήψεις) ενώ του άλλου να έχει υποστεί θόρυβο αλατοπίπερου (SP) (1M επαναλήψεις)

Πίνακας 1.6: Φίλτρα Autoencoders με Theano [38]

Autoencoder	Denoising Autoencoder (MN noise)
	

Πίνακας 1.7: Φίλτρα Autoencoders με Python [50]

Denoising Autoencoder (GS noise)	Denoising Autoencoder (MN noise)
	

1.5.1.2.1.1 Emphasized Denoising Autoencoders

Πρόκειται για μια επέκταση των Denoising Autoencoders . Όπως προδίδει και το όνομα τους τα δίκτυα αυτά αντί να αναθέτουν ίδιο βάρος στην ανακατασκευή όλων των συνιστωσών της εισόδου δίνουν μεγαλύτερη έμφαση σε αυτές που έχουν υποστεί στρέβλωση [50]. Για να επιτευχθεί αυτό προσθέτουμε δύο υπέρ-παραμέτρους α και β. Όπου α πρόσδιορίζει το βάρος που θα δοθεί από το σφάλμα ανακατασκευής σε συνιστώσες που στρεβλωθήκαν και αντίστοιχα το β σε αυτές που έμειναν αναλλοίωτες. Άρα στην περίπτωση της συνάρτησης τετραγωνικής απώλειας:

$$L_{2,a}(x,z) = a\left(\left(\sum_{j\in \mathrm{J}(\tilde{\mathrm{x}})}\left(x_j - z_j\right)^2\right)\right) + \beta\left(\sum_{j\notin \mathrm{J}(\tilde{\mathrm{x}})}\left(x_j - z_j\right)^2\right)$$

Όπου το $\mathrm{J}(\tilde{\mathrm{x}})$ είναι το σύνολο των στοιχείων που στρεβλώθηκαν. Αντίστοιχα η συνάρτηση της δια-εντροπίας (cross-entropy) γίνεται:

$$L_{\mathcal{H},\alpha}(x,z) = \alpha\left(-\sum_{j\in \mathrm{J}(\tilde{\mathrm{x}})}\left[x_k \log z_k + (1-x_k)\log(1-z_k)\right]\right) + \beta\left(-\sum_{j\notin \mathrm{J}(\tilde{\mathrm{x}})}\left[x_k \log z_k + (1-x_k)\log(1-z_k)\right]\right)$$

Μια ειδική περίπτωση η οποία ονομάζεται πλήρη έμφαση προκύπτει όταν τεθεί το $\alpha = 1$ και το $\beta = 0$ και άρα λαμβάνεται υπόψιν το σφάλμα μόνο κατά την πρόβλεψη των στρεβλωμένων εισόδων.

1.5.1.2.2 Sparse Autoencoders

Οι αραιοί Autoencoders (Sparse Autencoders) [52] είναι μια παραλλαγή των απλών Autoencoders που κάνει χρήση της έννοιας της αραιότητας (sparsity). Με την εισαγωγή της αραιότητας είναι δυνατή η κατασκευή Autoencoders με μεγάλο πλήθος κρυφών νευρώνων (μπορεί να ξεπερνάει ακόμα και το μέγεθος του διανύσματος εισόδου) οι οποίοι μπορούν ακόμα να ανακαλύψουν ενδιαφέρονται χαρακτηριστικά της εισόδου.

Τυπικά ένας νευρώνας είναι ενεργός, όταν η τιμή της εξόδου του πλησιάζει το 1 και μη ενεργός όταν πλησιάζει το 0 (αν έχουμε απλή σιγμοειδή συνάρτηση σε περίπτωση χρήσης $tanh$ ο νευρώνας θεωρείται ανενεργός αν η τιμή του πλησιάζει το -1). Με την εισαγωγή αραιότητας στο δίκτυο περιορίζονται οι νευρώνες και εξαναγκάζονται να είναι ανενεργοί για την μεγαλύτερο μέρος της ζωής τους. Τεθείτε $a_j^{(l)} = f_\theta\left(x^{(i)}\right) = s(W_j x^{(i)} + b_j)$ ως η ενεργοποίηση του j-οστού κρυφού νευρώνα στο l-οστό στρώμα όταν έχει δεχθεί είσοδο $x^{(i)}$. Επιπλέον ορίζεται η μέση (μέση τιμή αναφορικά με το σετ εισόδων εκπαίδευσης ως:

$$\widehat{p_j} = \frac{1}{m}\sum_{i=1}^{m} a_j^{(l)}\left(x^{(i)}\right)$$

Η αραιότητα εισάγει την ακόλουθη προϋπόθεση:

$$\widehat{p_j} = p$$

Όπου p ονομάζεται παράμετρος αραιότητας (sparsity parameter) και τυπικά έχει τιμή κοντά στο μηδέν (θεωρώντας πάντα ότι χρησιμοποιείται απλή σιγμοειδής συνάρτηση) π.χ. $p = 0.05$. Προκειμένου ένας κρυφός νευρώνας να πληροί αυτή την προϋπόθεση πρέπει να είναι τις περισσότερες φορές απενεργοποιημένος. Για να επιτευχθεί αυτός ο στόχος εισάγετε ένας επιπλέων όρος ποινής ο οποίος "τιμωρεί" τα $\widehat{p_j}$ που απέχουν σημαντικά από το p. Υπάρχουν πολλές επιλογές για τον όρο ποινής και ένας απ' αυτούς είναι ο:

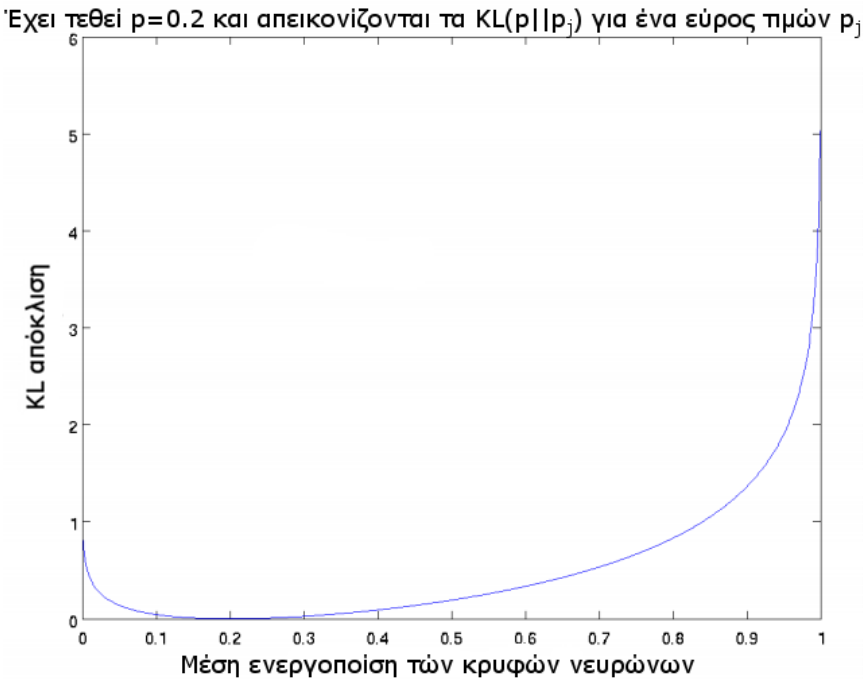
$$\sum_{j=1}^{s_2} p \log \frac{p}{\widehat{p_j}} + (1-p) \log \frac{1-p}{1-\widehat{p_j}}$$

Η παραπάνω συνάρτηση ποινής βασίζεται στην απόκλιση KL η οποία μετράει το κατά πόσο διαφέρουν δύο διανομές. Άρα η εξίσωση μπορεί να γραφτεί και ως:

$$\sum_{j=1}^{s_2} KL\left(p||\widehat{p_j}\right)$$

Όπου $KL\left(p||\widehat{p_j}\right) = p \log \frac{p}{\widehat{p_j}} + (1-p) \log \frac{1-p}{1-\widehat{p_j}}$ είναι η Kullback-Leibler (KL) απόκλιση μεταξύ μιας τυχαίας μεταβλητής Bernoulli με μέσω p και μιας τυχαίας μεταβλητής Bernoulli με μέσω $\widehat{p_j}$. Αυτή η συνάρτηση ποινής έχει την εξής πολύ χρήσιμη ιδιότητα:

- Αν $(\widehat{p_j} = p)$ τότε $KL\left(p||\widehat{p_j}\right) = 0$ όπως φαίνεται και στο παρακάτω διάγραμμα η συνάρτηση παρουσιάζει ελάχιστον για $\widehat{p_j} = p$
- Αλλιώς αυξάνεται μονότονα όσο το $\widehat{p_j}$ παρεκκλίνει από το p από το διάγραμμα φαίνεται μάλιστα ότι τείνει στο άπειρο.



Διάγραμμα 1.14: Μέση ενεργοποίηση Νευρώνων – KL απόκλιση [22]

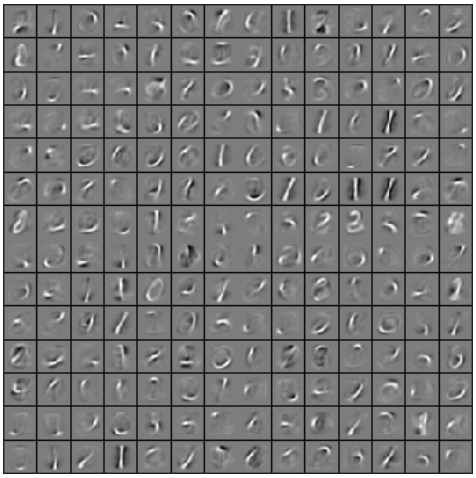
Με την εισαγωγή του όρου ποινής λοιπόν η συνάρτηση απώλειας γίνεται:

$$L_{sparse}(W,b) = L(W,b) + \beta \sum_{j=1}^{s_2} KL\left(p||\widehat{p_j}\right)$$

Όπου $L(W,b)$ είναι η επιλεγείσα συνάρτηση απώλειας και το β ελέγχει το βάρος του όρου ποινής που εισάγει η αραιότητα. Είναι προφανές ότι ο όρος $\widehat{p_j}$ εξαρτάται άμεσα από τις παραμέτρους $\theta = (W,b)$ καθώς εξαρτάται από την ενεργοποίηση του νευρώνα η οποία εξαρτάται με την σειρά της από το θ.

Οι Sparse Autoencoders μπορούν να εκπαιδευτούν με την χρήση αλγορίθμου οπισθοδρομικής διάδοσης σφάλματος και απότομης καθόδου με την επιπρόσθετη επιβάρυνση του υπολογισμού των παραγώγων του όρου ποινής ($\beta\left(-\frac{p}{\widehat{p_j}} + \frac{1-p}{1-\widehat{p_j}}\right)$) και άρα της μέσης ενεργοποίησης $\hat{p}$ κάθε νευρώνα. Για να υπολογιστεί το $\hat{p}$ πρέπει να πριν εφαρμοστεί ο αλγόριθμος οπισθοδρομικής διάδοσης πρέπει να γίνει ένα πέρασμα στο δίκτυο (προς τα εμπρός). Αυτό είναι εφικτό σε μικρά δίκτυα με λίγους όρους αλλά σε μεγάλα δίκτυα οι τιμές δεν χωράνε στην μνήμη και άρα απαιτείται ο επανυπολογισμός του $\hat{p}$ για κάθε είσοδο κάνοντας το αλγόριθμο λιγότερο αποδοτικό.

Στη Εικόνα 1.28 φαίνεται η χρήση ενός Sparse Autoencoder ως ανιχνευτής χαρακτηριστικών στο σετ δεδομένων MNIST.

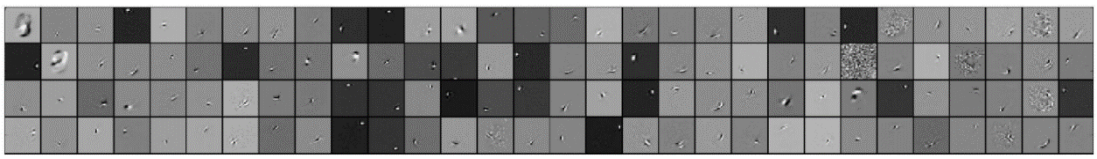
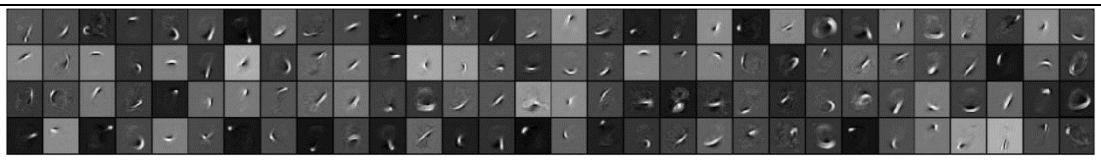


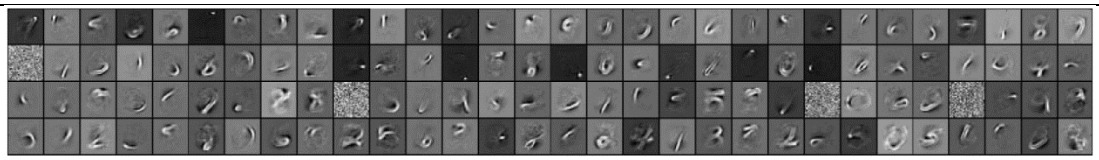
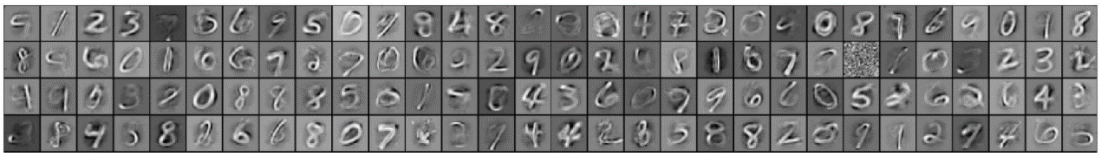
Εικόνα 1.28: Sparse Autoencoder ως ανιχνευτής χαρακτηριστικών στο σετ δεδομένων MNIST [22]

1.5.1.2.2.1 k-Sparse Autoencoder

Ο k-Sparse Autoencoder [55] βασίζεται πάνω σε έναν Autoencoder με γραμμικές συναρτήσεις ενεργοποίησης και του οποίου τα βάρη είναι δεμένα (ο πίνακας συναπτικών βαρών W' της ανάστροφης χαρτογράφησης (αποκωδικοποίησης) προκύπτει από την αντιστροφή του πίνακας βαρών W της έμπροσθεν χαρτογράφησης (κωδικοποίησης) $W' = W^{\tau}$). Η μεγάλη διαφορά όμως έγκειται στο γεγονός ότι κατά την πριν την αποκωδικοποίηση επιλέγονται k νευρώνες οι οποίοι έχουν τίς μεγαλύτερες τιμές εξόδου και όλοι οι υπόλοιποι τίθενται ίσοι με το. Το k ονομάζεται βαθμός αραιότητας κα έχει αποδειχτεί πειραματικά ότι ενδείκνυται η επιλογή μεγάλου k (π.χ. $k = 100$ στο MNIST) κατά την αρχή της διαδικασίας εκπαίδευσης και η γραμμική του μείωση κατά το πρώτο ήμισυ της διάρκειας εκπαίδευσης σε μια μικρότερη τιμή (π.χ. $k = 15$ στο MNIST). Εφόσον όπως προαναφέρθηκε η συνάρτηση που υπολογίζεται από το δίκτυ είναι γραμμική, μη-γραμμικότητα εισάγεται μόνο κατά την διαδικασία επιλογής των k μεγαλύτερων τιμών ενεργοποίησης. Στον παρακάτω πίνακα ακολουθούν παραδείγματα επιλογής διαφορετικού βαθμού αραιότητας στο MNIST με χρήση k-Sparse Autoencoder χιλίων κρυφών νευρώνων.

Πίνακας 1.8: Επίδραση της τιμής του k ενός k-Sparse Autoencoder [55]

k=70	
k=40	

k=25	
k=10	

Μετά την επιτυχή εκπαίδευση του δικτύου η αραιή αναπαράσταση που προκύπτει μπορεί να χρησιμοποιηθεί σε εφαρμογές κατάντης κατηγοριοποίησης (downstream classification tasks). Ακολουθεί ο αλγόριθμος ενός k-Sparse Autoencoder:

Εκπαίδευση:

- Εμπροσθοδρομική φάση (κωδικοποίηση): Υπολόγισε το $y = s(Wx + b)$
- Βρες τίς k μεγαλύτερες ενεργοποιήσεις από το y και θέσε τις υπόλοιπες ίσες με 0.
- Υπολόγισε την έξοδο και το σφάλμα χρησιμοποιώντας το αραιωμένο πλέον y.

$$z = s(W'y + b')$$
$$L(x, y) = \|x - z\|^2$$

- Με χρήση οπισθοδρομικής διάδοσης σφάλματος μετάδωσε το σφάλμα χρησιμοποιώντας τις k μεγαλύτερες ενεργοποιήσεις και επανάλαβε.

Αραιή Κωδικοποίηση

- Υπολόγισε τα χαρακτηριστικά $h = W^{\tau}x + b$
- Βρες τις αk μεγαλύτερες ενεργοποιήσεις και θέσε τις υπόλοιπες ίσες με 0. Όπου $\alpha > 1$ του οποίου η χρήση έχει αποδειχτεί ότι βελτιώνει ελάχιστα την απόδοση σε εφαρμογές κατάντης κατηγοριοποίησης

1.5.1.2.3 Sparse Denoising Autoencoder (SpDAE)

Ο αραιός αποθορυβοποίητής Autoencoder είναι ουσιαστικά ένας Denoising Autoencoder στον οποίο έχει εφαρμοστεί ένας κανονικοποιητής αραιότητας (δες Sparse Autoencoder) μετατρέποντας το δίκτυο έτσι σε Sparse Denoising Autoencoder. Εξαιτίας του μεγάλου υπολογιστικού κόστους όμως η παράμετρος αραιότητας εφαρμόζεται σχεδόν πάντα αποκλείστηκα κατά την διάρκεια προ-εκπαίδευσης του δικτύου [56].

Η λειτουργεία ενός spDAE μπορεί να συμπυκνωθεί στις εξής λειτουργίες:

- Αρχικά εκπαιδεύει έναν κωδικοποιητή, ο οποίος χαρτογραφεί από ένα χώρο δεδομένων καθοριζόμενος από τις εισόδους εκπαίδευσης οι οποίες έχουν υποστεί κάποια στρέβλωση σε έναν αραιό λανθάνοντας χώρο ο οποίος καθορίζεται από την κανονικοποίηση που επιφέρει αραιότητα.
- Στην συνέχεια ο αποκωδικοποιητής του spDAE χαρτογραφεί από τον αραιό λανθάνοντας χώρο στον χώρο δεδομένων.

Σε εργασίες το 2012 από τον Burger και τον Xie χρησιμοποιήθηκαν πολύ βαθιά spDAE για την αποθορυβοποίηση εικόνων. Η αποδόσεις των αρχιτεκτονικών αυτό ήταν συγκρίσιμες ή και καλύτερες από συμβατικές μεθόδους αποθορυβοποίησης εικόνας. Αποδεικνύοντας έτσι ότι η εφαρμογή μιας απλή μορφή αραίωσης (sparsification) μπορεί να βελτιώσει της αποθορυβοποιές ικανότητες ενός Denoising Autoencoder [57].

1.5.1.2.4 Transforming Auto-encoder

Με την χρήση βαθιών Autoencoders είναι δυνατή η εξαγωγή ενός συμπαγούς κώδικα (χαρτογράφησης)για ένα διάνυσμα χαρακτηριστικών εκμεταλλευομένη τα πολλά στρώματα και την μη γραμμικότητα της αρχιτεκτονική. Παρόλα αυτά ο εξαγόμενος κώδικας μπορεί να αλλάξει απρόβλεπτα όταν το διάνυσμα εισόδου μετασχηματίζεται. Για τον λόγο αυτόν προτάθηκαν οι μετασχηματιζόμενοι Autoencoders το 2011 από τον Hinton [58]. Τα ντετερμινιστικά αυτά δίκτυα έχουν ως στόχο ο κώδικας που παράγεται να μπορεί να αλλάξει με προβλέψιμο τρόπο ο οποίος να ανακλά τους μετασχηματισμούς του διανύσματος εισόδου.

Η βασική δομική μονάδα ενός Transforming Autoencoder είναι η «κάψουλα», ένα ανεξάρτητο υποδίκτυο το οποίο εξάγει ένα μοναδικό παραμετροποιημένο χαρακτηριστικό το οποίο αναπαριστά μια οντότητα είτε αυτή είναι οπτική είτε ηχητική. Το δίκτυο αυτό δέχεται ένα διάνυσμα εισόδου και ένα διάνυσμα εξόδου. Το διάνυσμα εξόδου είναι το προϊόν της εφαρμογής ενός καθολικού μετασχηματισμού στο διάνυσμα εισόδου (π.χ. μετακίνηση εικόνας ή αλλαγή συχνότητας ηχητικού δείγματος). Τα στρώματα υπεύθυνα για την κωδικοποίηση λοιπόν αποτελούνται από της εξόδους διαφόρων τέτοιων καψουλών. Στην συνέχεια οι πληροφορίες αφορούν ειδικά την αναγνώριση εικόνων μιας και η λογική σε άλλες εφαρμογές είναι παρόμοια.

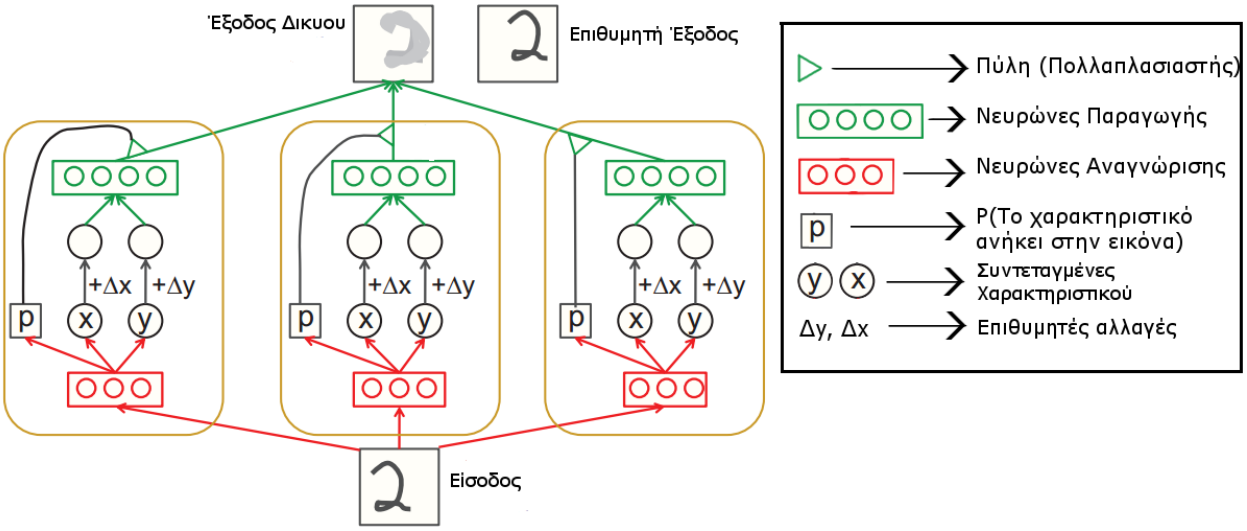
Ένα από τα βασικά πλεονεκτήματα της χρήσης καψουλών είναι ότι παρέχουν έναν απλό τρόπο να αναγνωρίζουν ολότητες μέσω της αναγνώρισης των δομικών τους στοιχείων. Σκοπός μιας κάψουλας είναι να μάθει να εξάγει την στάση (πόζα) της οπτικής της οντότητας σε ένα διάνυσμα το οποίο συγγενεύει γραμμικά με στις «φυσικές» αναπαραστάσεις στάσεων που χρησιμοποιούνται στα γραφικά υπολογιστών. Έτσι λοιπόν αν δύο κάψουλες A και B είναι ενεργές και έχουν κατάλληλη χωρική σχέση θα ενεργοποιήσουν την κάψουλα υψηλότερου επιπέδου C. Αναλυτικά: Έστω ότι οι έξοδοι της κάψουλας A αναπαρίστανται από τον πίνακα T_A, ο οποίος προσδιορίζει την αλλαγή συντεταγμένων μεταξύ της κανονική οπτικής οντότητας του Aου στιγμιότυπου της οντότητας που βρήκε η κάψουλα. Αν στην συνέχεια πολλαπλασιάσουμε τον πίνακα T_A με τον πίνακα T_C της κάψουλας C προκύπτει ο πίνακας T_{AC} ο οποίος σχετίζει την κανονική οπτική οντότητα του A με κανονική οπτική οντότητα C και έτσι λαμβάνεται μια πρόβλεψη για το T_C. Αντίστοιχα, υπολογίζεται ο πίνακας T_{BC} και λαμβάνεται μια δεύτερη πρόβλεψη για το T_C. Ένα αυτές οι προβλέψεις ταιριάζουν δηλαδή αν T_{AC} και T_{BC} έχουν κατάλληλη χωρική

σχέση τότε η κάψουλα C ενεργοποιείται και από το μέσο όρο των δύο προβλέψεων λαμβάνουμε πληροφορία για το πώς η μεγαλύτερη οπτική οντότητα που αναπαριστά η C έχει μετασχηματιστεί αναφορικά με την κανονική οπτική οντότητα της C. Για παράδειγμα η κάψουλα A θα μπορούσε να αναπαριστά ένα στόμα η B μία μύτη και έτσι μαζί μπορούνε να προβλέψουνε την πόζα που έχει πάρει το πρόσωπο.

Για καλύτερη κατανόηση της αρχιτεκτονικής δίνεται ένα Transforming Autoencoder με τρεις κάψουλες. Το δίκτυο λαμβάνει ως είσοδο μια εικόνα καθώς και τις επιθυμητές αλλαγές $\Delta x, \Delta y$ και εξάγει την μετασχηματισμένη εικόνα. Οι κάψουλες του δικτύου αλληλοεπιδρούν μόνο στο τελευταίο στρώμα όπου και συνεργάζονται για να παράξουν την επιθυμητή μετασχηματισμένη εικόνα. Κάθε κάψουλα έχει τα εξής δομικά στοιχεία:

- Νευρώνες αναγνώρισης: Οι οποίοι συμπεριφέρονται ως ένα κρυφό στρώμα υπεύθυνο για τον υπολογισμό των παρακάτω τιμών οι οποίες θα σταλούν στα ανώτερα στρώματα του δικτύου:
  - x, y: Οι συντεταγμένες της θέσης της οπτικής οντότητας στην εικόνα
  - p: Η πιθανότητα η οντότητα που αναπαριστά η κάψουλα να είναι μέρος της εικόνας που εισαχθεί
- Νευρώνες Παραγωγής: Οι οποίοι είναι υπεύθυνοι για τον υπολογισμό της συνεισφοράς της κάψουλας στην μετασχηματισμένη εικόνα. Οι νευρώνες αυτοί λαμβάνουν ως είσοδο τις τιμές:
  - $x + \Delta x$
  - $y + \Delta y$:

Η συνεισφορά αυτών των νευρώνων πολλαπλασιάζεται με την πιθανότητα p έτσι ώστε οι ανενεργές κάψουλες να μην επηρεάζουν το αποτέλεσμα.



Εικόνα 1.29: Transforming Autoencoder [58].

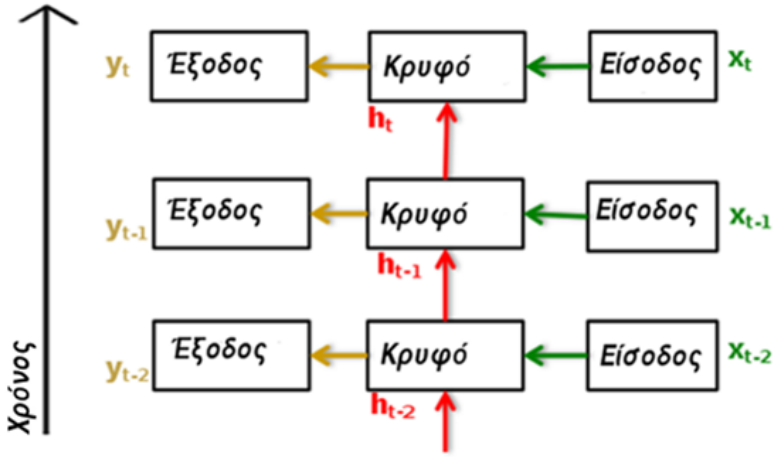
Οι Transforming Autoencoders παρουσιάζουν εντυπωσιακά και το παρακάτω πείραμα () αναδεικνύει ακριβώς αυτό. Για το συγκεκριμένο πείραμα χρησιμοποιήθηκαν δίκτυα με 25 κάψουλες καθεμία από τις οποίες περιείχε 40 νευρώνες αναγνώρισης και 40 παραγωγής. Ζητήθηκε από το δίκτυο να μετασχηματίσει εικόνες από το σετ δεδομένων MNIST. Δίνονται και η σωστά μετασχηματισμένες εικόνες για σύγκριση:

Πίνακας 1.9: Μετασχηματισμός ψηφίων MNIST από Transforming Autoencoder [58]

Εικόνες Εισόδου	
Εικόνες Εξόδου	
Επιθυμητό Αποτέλεσμα	

1.5.2 Recurrent Neural Networks (RNN)

Τα RNN [16] [59], (ελληνιστί Αναδρομικά νευρωνικά δίκτυα) είναι βαθιά επηρεασμένα από την βιολογία καθώς ο ανθρώπινος εγκέφαλος αποτελεί ουσιαστικά ένα RNN (αναδρομικό νευρωνικό δίκτυο) δηλαδή ένα δίκτυο νευρώνων με συνδέσεις ανατροφοδότησης. Τα RNN είναι δηλαδή μια τάξη τεχνιτών νευρωνικών δικτύων των οποίων οι νευρώνες σχηματίζουν έναν κατευθυνόμενο κύκλο, δημιουργώντας έτσι μια εσωτερική κατάσταση που τους επιτρέπει να παρουσιάζουν χρονικά δυναμική συμπεριφορά. Αρχικά τα RNN γνώρισαν επιτυχία σε εφαρμογές φωνητικής αναγνώρισης στην τηλεφωνία.



Εικόνα 1.30: RNN στον χρόνο [16]

Τα διάφορα στρώματα ενός RNN λειτουργούν σαν μνήμη και όχι σαν ιεραρχικοί επεξεργαστές δεδομένων όπως είναι στα DNN. Νέα πληροφορία εισάγεται σε κάθε στρώμα (κάθε επανάληψη του δικτύου) και το δίκτυο μπορεί να περάσει αυτή την πληροφορία για άπειρες ενημερώσεις παρέχοντας ουσιαστικά ανεξάντλητη μνήμη στο RNN.

Σε αντίθεση με τα έμπροσθεν τροφοδοτούμενα δίκτυα τα RNN μπορούν να χρησιμοποιήσουν την εσωτερική τους μνήμη για να επεξεργαστούν αυθαίρετες ακολουθίες. Αυτή η ιδιότητα καθιστά τα RNN εφαρμόσιμα σε εφαρμογές όπως η αναγνώριση χειρόγραφων κειμένων, εφαρμογή στην οποία έχουν επιτύχει βέλτιστα αποτελέσματα. Πρόκειται για εξαιρετικά ισχυρά, μη γραμμικό, μοντέλα. Χαρακτηριστικό είναι άλλωστε ότι με αρκετούς νευρώνες και αρκετό χρόνο RNN δίκτυα μπορούν να υπολογίσουν οτιδήποτε δύναται να υπολογιστεί από υπολογιστή. Η ισχύς τους απορρέει από τον συνδυασμό των δύο αυτών βασικών ιδιοτήτων:

- Έχουν την δυνατότητα να συγκρατούν πληροφορίες περασμένης απόδοσης σε κρυφούς νευρώνες κατάστασης

- Ακολουθούν μη γραμμικές δυναμικές κάτι που τους επιτρέπει να ενημερώνουν την κρυφή κατάσταση τους με περίπλοκους τρόπους

Η ισχύς που τα χαρακτηρίζει εισάγει όμως πολυπλοκότητα καθιστώντας την εκπαίδευση τους εξαιρετικά δύσκολη. Στα DNN η εισαγωγή δεδομένων γίνεται στο κατώτερο στρώμα και η έξοδος παράγεται στο ανώτερο. Τα RNN λειτουργούν διαφορετικά καθώς δέχονται είσοδο και παράγουν έξοδο άμεσα στο επόμενο χρονικό βήμα. Κατά την εκπαίδευση μια ακολουθία τροφοδοτείται στο δίκτυο και μέσω ενός αλγορίθμου βελτιστοποίησης (π.χ. χρήση απότομης καθόδου) ελαχιστοποιείται το σφάλμα μεταξύ της επιθυμητής εξόδου και της εξόδου του δικτύου. Τα βάρη των νευρώνων προσαρμόζονται κατάλληλα ώστε να ελαχιστοποίηση της συνάρτησης κόστους να γίνει όσο το δυνατό πιο γρήγορα. Πιθανές εφαρμογές των RNN είναι στη ρομποτική, στην αναγνώριση ομιλίας, στη σύνθεση μουσικής, σε ανάλυση πρωτεϊνών και σε αναρίθμητες άλλες εφαρμογές που προκειμένου να ληφθεί η βέλτιστη απόφαση είναι απαραίτητη η διατήρηση παλαιότερων δεδομένων.

Ένα RNN λοιπόν είναι ένα νευρωνικό δίκτυο που προσομοιάζει ένα δυναμικό σύστημα διακριτού χρόνου που έχει είσοδο x_t, έξοδο y_t και κρυφή κατάσταση h_t όπου το δείκτης t συμβολίζει χρόνο. Το δυναμικό σύστημα ορίζεται από:

$$h_t = f_h(x_t, h_{t-1})$$

$$y_t = f_o(h_t)$$

- Όπου f_h και f_o είναι μια συνάρτηση μετάβασης κατάστασης και μια συνάρτηση εξόδου αντίστοιχα.

Κάθε συνάρτηση παραμετροποιείται από μια σειρά παραμέτρων (w_h, b_h) και (w_o, b_o). Δοθείσας μιας σειράς N ακολουθιών εκπαίδευσης $D = \{({x_1}^n, {y_1}^n), ({x_2}^n, {y_2}^n), \dots\dots, ({x_{T_n}}^n, {y_{T_n}}^n)\}$, εκτιμιόνται οι τιμές των βαρών σε προσπάθεια να ελαχιστοποιηθεί η συνάρτηση κόστους της γενικής μορφής:

$$C(\theta) = \frac{1}{N}\sum_{n=1}^{N}\sum_{t=1}^{T_N} d\left({y_t}^{(n)}, f_o\left({h_t}^{(n)}\right)\right)$$

Όπου ${h_t}^{(n)} = f_h({x_t}^n, {h_{t-1}}^n)$ και ${h_o}^{(n)} = 0$. Το $d(a, b)$ είναι μια προκαθορισμένη μετρική συνάρτηση απόκλισης μεταξύ του a και b όπως η ευκλείδεια απόσταση.

Ένα κοινότυπο RNN λοιπόν κατασκευάζεται ορίζοντας την συνάρτηση μετάβασης και την συνάρτηση εξόδου ως:

$$h_t = f_h(x_t, h_{t-1}) = \varphi(W^\tau h_{t-1} + U^\tau x_t)$$

$$y_t = f_o(h_t) = \varphi_o(V^\tau h_t)$$

Όπου W,U και V αντιστοίχως είναι ο πίνακας μετάβασης, εισόδου και εξόδου και όπου

φ_h και φ_o είναι οι μη γραμμικές συναρτήσεις αναφορικά πρός στο εκάστοτε στοιχείο. Συνηθίζεται η χρήση μη γραμμικών εξισώσεων όπως της εφοδιαστικής σιγμοειδούς συνάρτησης ή της υπερβολικής εφαπτομένης για την φ_h.

1.5.2.1 Deep Recurrent Neural Networks (DRNN)

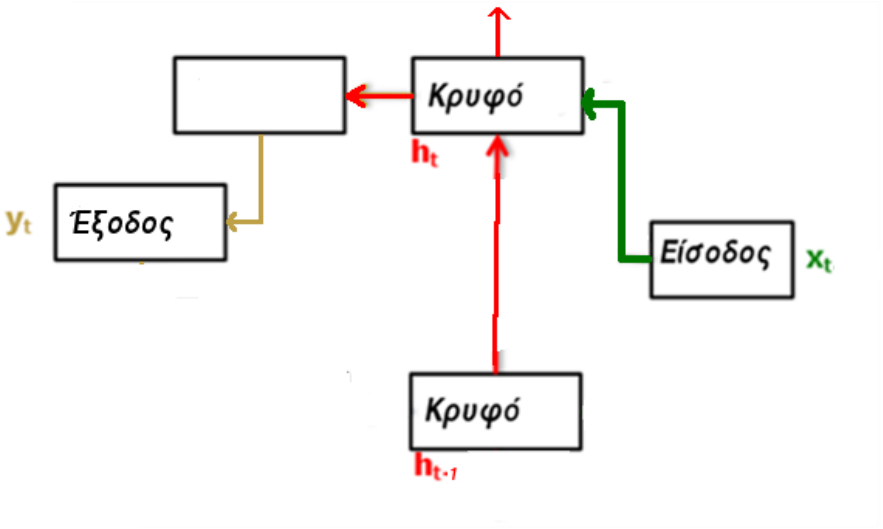
Μία από πιθανή αδυναμία ενός κοινού RNN θα μπορούσε να είναι η ανάγκη ορισμένων εφαρμογών για περίπλοκή, ιεραρχική επεξεργασία μια τρέχουσας εισόδου του δικτύου, κάτι το οποίο είναι αδύνατο με χρήση μόνο ενός στρώματος μεταξύ της εισόδου και της εξόδου [57]. Επίσης μπορεί να χρειαστεί να γίνει επεξεργασία μας χρονικής σειράς σε διαφορικές κλίμακες χρόνου. Για παράδειγμα, κατά την επεξεργασία φυσικής ομιλίας, τα δεδομένα στο κατώτατο στρώμα απαρτίζονται από φωνήματα τα οποία υπάρχουν για ελάχιστες μόνο χρονικές στιγμές. Στα ανώτερα στρώματα υπάρχουν οι συλλαβές, οι λέξεις, οι φράσεις κ.ο.κ. Τα κοινά RNN δεν υποστηρίζουν ρητά πολλαπλές κλίμακες χρόνου και κάθε είδους χωρική ιεραρχία στο σήμα εισόδου πρέπει να ενσωματωθεί στην δυναμική του δικτύου. Για τους λόγους αυτούς και λόγω της εμπειρικής υπόθεσης ότι τα βαθιά, ιεραρχικά μοντέλα αποδίδουν καλύτερα στην αναπαράσταση ορισμένων λειτουργειών από τα αντίστοιχα ρηχά, έχουν αναπτυχθεί τα τελευταία χρόνια βαθιά RNN (Deep RNN, DRNN).

Όταν γίνεται αναφορά βάθους στην περίπτωση των έμπροσθεν τροφοδοτούμενων δικτύων αναφερόμαστε στην ύπαρξη πολλαπλών μη γραμμικών στρωμάτων μεταξύ εισόδου και εξόδου. Η περίπτωση των RNN αποκλίνει από τον παραπάνω ορισμό λόγω της χρονικής φύσης της δομής τους. Για να επιτευχθεί βαθιά αρχιτεκτονική σε RNN υπάρχουν τρείς διαφορετικές προσεγγίσεις εκβάθυνσης (προσθήκης δια-στρωματικών μεσολαβούντων στρωμάτων) κάθε μία από τις οποίες προσδίδει στο RNN διαφορετικές ιδιότητες.

- Εκβάθυνση του τμήματος από είσοδο προς κρυφό στρώμα $(x_t \rightarrow h_t)$
- Εκβάθυνση του τμήματος από κρυφό προς κρυφό στρώμα $(h_{t-1} \rightarrow h_t)$
- Εκβάθυνση του τμήματος από κρυφό στρώμα προς έξοδο$(h_t \rightarrow y_t)$
- Στοίβαγμα πολλαπλών αναδρομικών κρυφών στρωμάτων το ένα πάνω από τ’ άλλο

Ακολουθούν αναλυτικότερα τρόποι με τους οποίους μπορεί να επιτευχθεί εκβάθυνση (Οι ονομασίες μπορεί να διαφέρουν από βιβλιογραφία σε βιβλιογραφία αλλά οι τεχνικές παραμένουν παρόμοιες).

1.5.2.1.1 Deep Output RNN (DO-RNN)



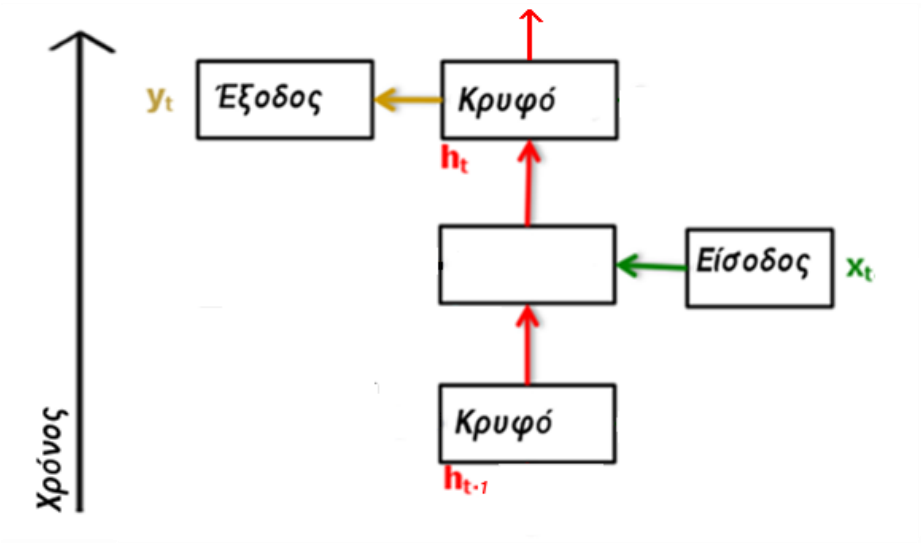
Εικόνα 1.31: DO-RNN, RNN Βαθιάς εξόδου

Ένα DO-RNN [60] (RNN βαθιάς εξόδου) επιτυγχάνει εκβάθυνση με την προσθήκη περεταίρω σταδίων ανάμεσα στην κρυφή κατάσταση και την έξοδο. Ένα από τα μοντέλα που έχουν προταθεί για υλοποίηση DO-RNN είναι η χρήση πολυστρωματικού perceptron για την προσέγγιση της συνάρτησης εξόδου f_o. Σε αυτή την περίπτωση υλοποιείται η f_h σε L ενδιάμεσά στρώματα ως εξής:

$$y_t = f_o(h_t) = \varphi_o\left[V_L^\tau \varphi_{L-1}\left(V_{L-1}^\tau \varphi_{L-2}\left(\dots \varphi_1(V_1^\tau h_t)\right)\right)\right]$$

Όπου φ_ι και W_i αντιστοίχως είναι η μη γραμμική συνάρτηση αναφορικά στο εκάστοτε στοιχείο και ο πίνακας βαρών του ι-οστού στρώματος. Με αυτήν την διάταξη ένα RNN μπορεί να «μάθει» μια μη τετριμμένη, άκρος μη γραμμική μετάβαση μεταξύ συνεχόμενων κρυφών καταστάσεων.

1.5.2.1.2 Deep Transition RNN (DT-RNN)



Εικόνα 1.32: DT-RNN, RNN Βαθιάς Μετάβασης

Τα RNN βαθιάς μετάβασης (DT-RNN) υλοποιούν εκβάθυνση του τμήματος από κρυφό προς κρυφό στρώμα $(h_{t-1} \rightarrow h_t)$.

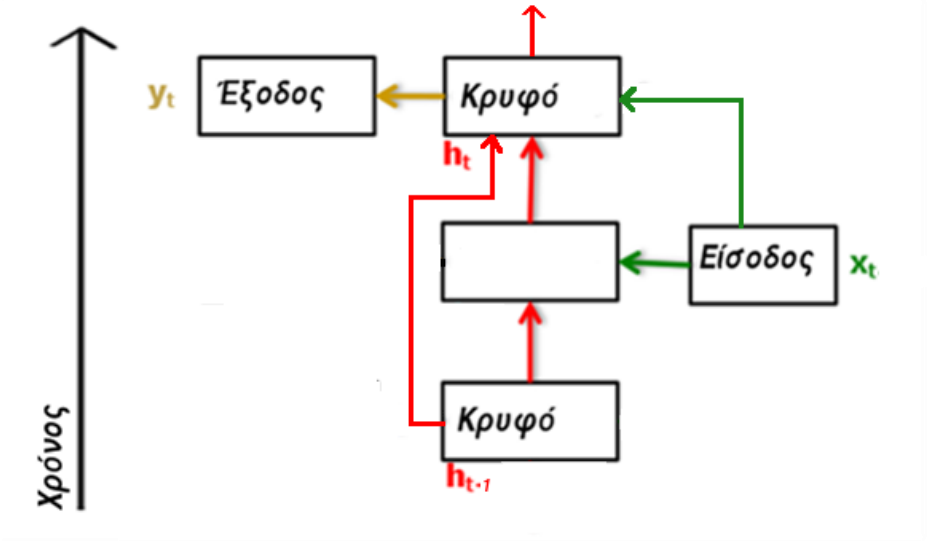
Αυτή η μεταβατική κατάσταση μεταξύ των κρυφών στρωμάτων h_t και h_{t-1} εισάγει μία νέα είσοδο στην «περίληψη» των προηγούμενων εισόδων. Έχει προταθεί [src] ότι αυτή η νέα μεταβατική κατάσταση πρέπει να είναι εξαιρετικά μη γραμμική. Με αυτό τον τρόπο θα μπορούσε να επιτρέψει στην κρυφή κατάσταση ενός RNN να προσαρμοστεί τάχιστα σε γρήγορα μεταβαλλόμενες διαφορετικές εισόδους ενώ ταυτόχρονα να διατηρεί μια χρήσιμη «περίληψη» του παρελθόντος.

Ένα από τα μοντέλα που έχουν προταθεί για υλοποίηση DT-RNN είναι η χρήση πολυστρωματικού perceptron για την προσέγγιση της συνάρτησης μετάβασης κατάστασης f_h. Σε αυτή την περίπτωση υλοποιείται η f_h σε L ενδιάμεσά στρώματα ως εξής:

$$h_t = (f_h x_t, h_{t-1}) = \varphi_h \left[W_L^\tau \varphi_{L-1} \left(W_{L-1}^\tau \varphi_{L-2} (\dots \varphi_1 (W_1^\tau h_{t-1} + U^\tau x_t)) \right) \right]$$

Όπου φ_ι και W_i αντιστοίχως είναι η μη γραμμική συνάρτηση αναφορικά στο εκάστοτε στοιχείο και ο πίνακας βαρών του ι-οστού στρώματος. Με αυτήν την διάταξη ένα RNN μπορεί να «μάθει» μια μη τετριμμένη, άκρος μη γραμμική μετάβαση μεταξύ συνεχόμενων κρυφών καταστάσεων.

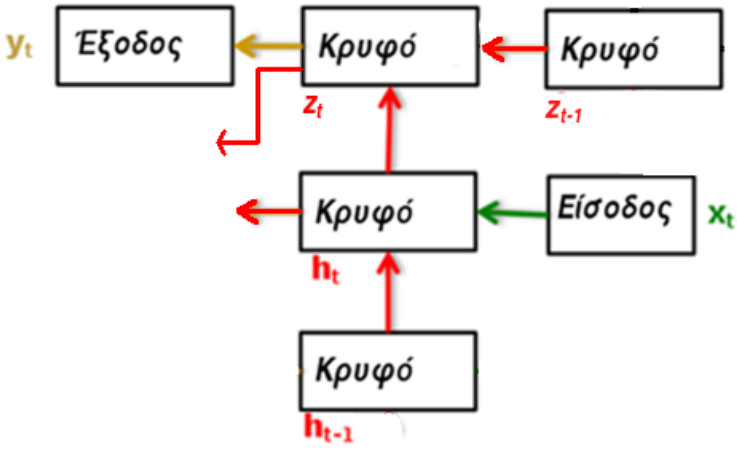
1.5.2.1.3 Deep Transition (Shortcut) RNN (DT(S)-RNN)



Εικόνα 1.33: DT(S)-RNN, RNN Βαθιάς Μετάβασης με συντόμευση

Τα προαναφερθέντα DT-RNN παρουσιάζουν ένα πιθανό πρόβλημα. Με την εισαγωγή της νέας μεταβατικής κατάστασης, τα μη γραμμικά βήματα που απαιτούνται για τον υπολογισμό της κλίσης σε βάθος χρόνου αυξάνονται κάνοντας έτσι την εκπαίδευση ενός τέτοιου δικτύου δύσκολη. Για το λόγο αυτό προτάθηκε η εισαγωγή συνδέσεων συντόμευσης μεταξύ των κρυφών καταστάσεων αποφεύγοντας έτσι τα ενδιάμεσα μεταβατικά στάδια κατά τον υπολογισμό της κλίσης. Τέτοιοι τύποι δικτύου ονομάζονται DT(S)-RNN.

1.5.2.1.4 Stacked RNN (s-RNN)



Εικόνα 1.34: s-RNN, Στοιβαγμένο RNN

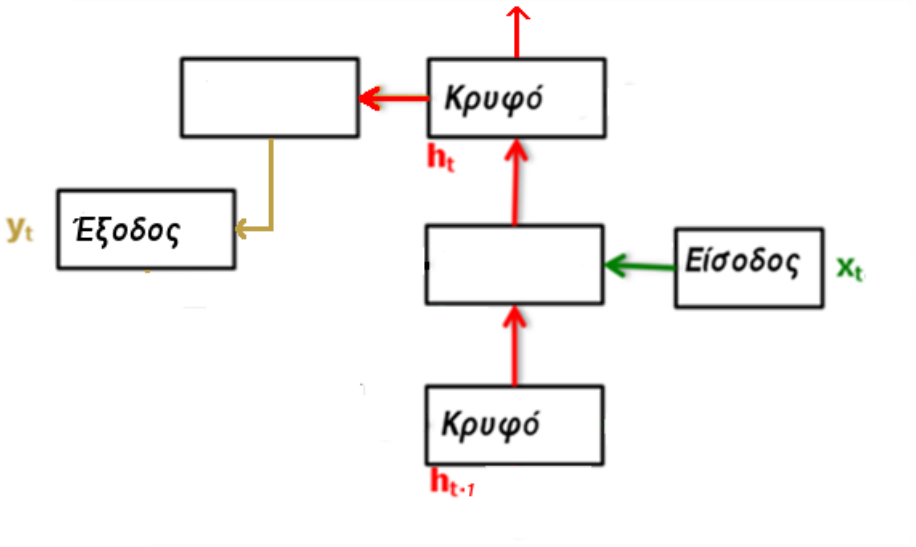
Ένα RNN δύναται να επεκταθεί εις βάθος στοιβάζοντας πολλαπλά αναδρομικά κρυφά στρώματα το ένα πάνω από το άλλο. Το μοντέλο αυτό αποκαλείται στοιβαγμένο RNN (s-RNN) Ο στόχος ενός τέτοιου μοντέλου είναι να ενθαρρύνει κάθε αναδρομικό στρώμα να λειτουργεί σε διαφορετική κλίμακα χρόνου.

Ένα s-RNN λοιπόν έχει πολλαπλά στάδια συναρτήσεων μετάβασης που ορίζονται ως:

$$h_t^{(i)} = f_h{}^i\left(h_t^{(i-1)}, h_{t-1}{}^i\right) = \varphi\left(W_i^\tau h_{t-1}^{(i)} + U_i^\tau h_t^{i-1}\right)$$

Όπου $h_t^{(i)}$ είναι η κρυφή κατάσταση του ι-οστού επιπέδου την χρονική στιγμή t. Για $t = 1$, η τιμή της κατάστασης υπολογίζεται χρησιμοποιώντας το x_t αντί του $h_t^{(i-1)}$. Η κρυφές καταστάσεις όλων των επιπέδων υπολογίζονται αναδρομικά από το κατώτατο στρώμα $l = 1$

1.5.2.1.5 Deep Transition, Deep Output RNN (DOT-RNN)



Εικόνα 1.35: DOT-RNN, RNN Βαθιάς μετάβασης, Βαθιάς εισόδου

Τα RNN βαθιάς μετάβασης, βαθιάς εξόδου (DOT-RNN) αποτελούν συνδυασμό εκβάθυνσης RNN με χρήση τεχνικών εκβάθυνσης από κρυφό προς κρυφό στρώμα $(h_{t-1} \to h_t)$ και από κρυφό στρώμα προς έξοδο$(h_t \to y_t)$. Αν γίνει και προσθήκη μεταβατικών καταστάσεων ανάμεσα στα κρυφά στρώματα το μοντέλο θα ονομαστεί DOT(S)-RNN.

1.5.2.2 Bi-directional RNN (BRNN)

Τα αμφίδρομα RNN [61] (BRNN) αναπτύχθηκαν προκειμένου να καταστεί δυνατή η αξιοποίηση και των μελλοντικών δεδομένων εκτός των παρελθοντικών, αδυναμία που περιορίζει τις άλλες αρχιτεκτονικές RNN. Αυτό σημαίνει ότι για κάθε σημείο της δοθείσας

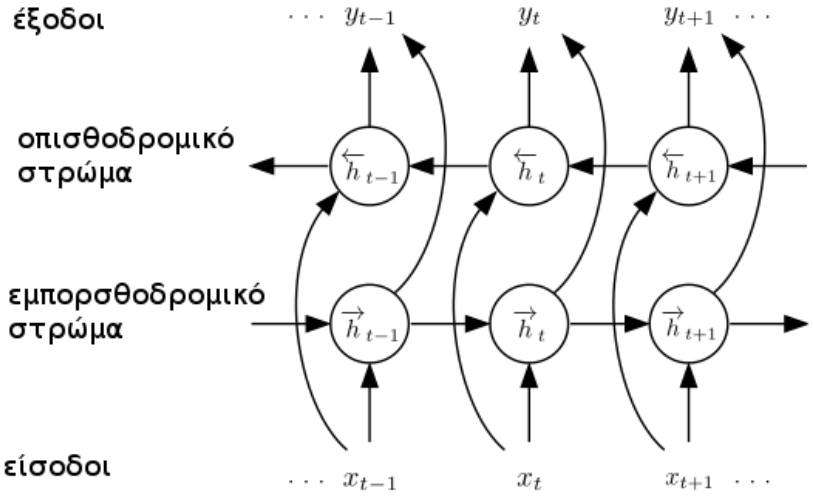
ακολουθίας το δίκτυο έχει πλήρη ακολουθιακή πληροφορία για όλα τα σημεία πριν και μετά από αυτό. Για παράδειγμα κατά την διαδικασία αναγνώρισης φωνής, όταν ολόκληρες προτάσεις παρουσιάζονται στιγμιαία η εκμετάλλευση μελλοντικών λέξεων θα είχε θετικές συνέπειες. Τα BRNN το επιτυγχάνουν αυτό διαχωρίζοντας το κρυφό στρώμα σε δύο υποστρώματα καθένα απ' τα οποία επεξεργάζεται δεδομένα προερχόμενα από μια εκ των δύο κατευθύνσεων (παρελθών και μέλλον) τα οποία στην συνέχεια τροφοδοτούν ένα κοινό στρώμα εξόδου.

Υπολογίζεται η εμπρόσθοδρομική κρυφή ακολουθία $\vec{h}$, η οπισθοδρομική ακολουθία $\overleftarrow{h}$ και η έξοδος y:

$$\vec{h} = H\left(W_{x\vec{h}}x_t + W_{\vec{h}\vec{h}}\vec{h}_{t-1} + b_{\vec{h}}\right)$$

$$\overleftarrow{h} = H\left(W_{x\overleftarrow{h}}x_t + W_{\overleftarrow{h}\overleftarrow{h}}\overleftarrow{h}_{t-1} + b_{\overleftarrow{h}}\right)$$

$$y_t = W_{\vec{h}y}\vec{h}_t + W_{\overleftarrow{h}y}\overleftarrow{h}_t + b_y$$



Εικόνα 1.36: BRNN, Αμφίδρομο RNN [99]

Η διαδικασία εκπαίδευσης του δικτύου μπορεί να συμπυκνωθεί στον ακόλουθο αλγόριθμο [62]:

1. Εμπροσθοδρομικό Πέρασμα: Για ένα χρονικό διάστημα, $1 \leq t \leq T$, πέρνα όλα τα δεδομένα εισόδου από το BRNN και υπολόγισε όλες τις προβλεπόμενες εξόδους.
    a. Κάνε εμπροσθοδρομικό πέρασμα για της έμπροσθεν (από $t = 1$ έως $t = T$) και τις όπισθεν καταστάσεις (από $t = T$ έως $t = 1$).
    b. Κάνε εμπροσθοδρομικό πέρασμα για τους νευρώνες εξόδου
2. Οπισθοδρομικό Πέρασμα: Υπολόγισε τις μερικές παραγώγους της συνάρτησης y_t για το χρονικό διάστημα $1 \leq t \leq T$ που χρησιμοποιήθηκε κατά το εμπροσθοδρομικό πέρασμα.
    a. Κάνε το οπισθοδρομικό πέρασμα για τους νευρώνες εξόδου

b. Κάνε οπισθοδρομικό πέρασμα για της έμπροσθεν (από $t = T$ έως $t = 1$) και τις όπισθεν καταστάσεις (από $t = 1$ έως $t = T$).

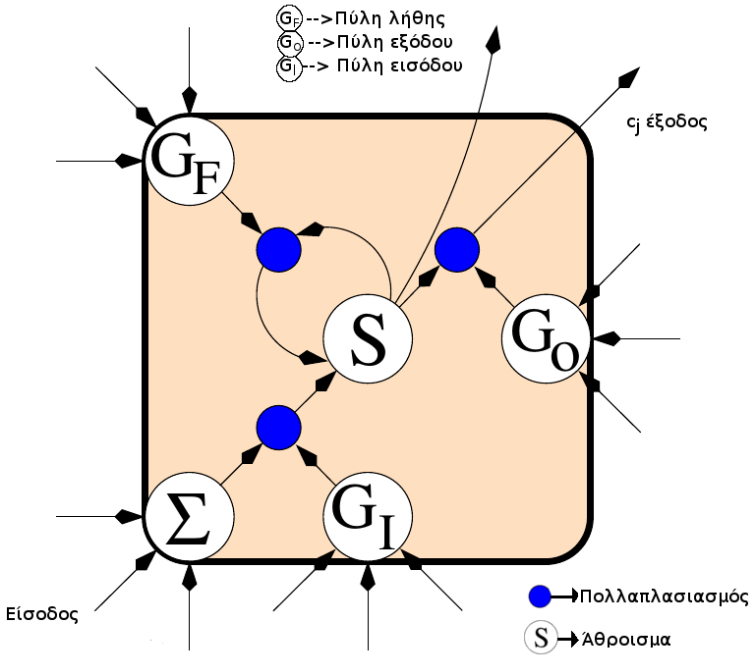
3. Ενημέρωσε τα Βάρη

Η τεχνική αυτή έχει αποδειχθεί εξαιρετικά χρήσιμή ειδικά όταν συνδυάζεται με LSTM. Πειραματικά έχει επιτευχθεί 78,6% επιτυχία κατά την κατηγοριοποίηση φωνημάτων της βάσης δεδομένων TIMIT με χρήση αμφίδρομου LSTM.

1.5.2.3 Long Short-Term Memory (LSTM)

Ένα δίκτυο μακράς βραχυπρόθεσμης μνήμης (LSTM) [63] είναι ένα RNN σχεδιασμένο με σκοπό την εκμάθηση μακροπρόθεσμων εξαρτήσεων μέσω gradient descent. Το χαρακτηριστικό που διαχωρίζει μια αρχιτεκτονική LSTM από τα κοινότυπα RNN είναι τα κελιά μνήμης (memory cells) που είναι ικανά να διατηρήσουν την ενεργοποίηση τους επ' άπειρον. Τα LSTM αναπτύχθηκαν το 1997 από τους Sepp Hochreiter και Jürgen Schmidhube. Μερικές από τις εφαρμογές που έχουν χρησιμοποιηθεί είναι ο χειρισμός ρομπότ, η εκμάθηση ρυθμού, αναγνώριση χειρόγραφου κειμένου, αναγνώριση ανθρώπινων ενεργειών (στην οποία μάλιστα κέρδισε και το ICDAR handwriting Competition το 2009 [63]) κα άλλες. Τα LSTM μπορούν να συνδυαστούν με τα BRNN σε μια αρχιτεκτονική ονομαζόμενη και ως bi-directional LSTM τα οποία έχουν επιτύχει πολύ ενθαρρυντικά αποτελέσματα σε εφαρμογές όπως φωνητική αναγνώριση [64].

Ένα κελί μνήμης αποτελείται από μια μη γραμμική μονάδα η οποία φυλάει την κατάσταση του κελιού και τρεις πύλες που μπορούν να ανοιγοκλείσουν ανά πάσα στιγμή. Οι πύλες πρόκεινται για σιγμοειδής νευρώνες των οποίων η λειτουργεία έχει άμεσο αντίκτυπο στην έξοδο μιας της μονάδας. Η πύλη G_i εισόδου «προστατεύει» έναν νευρώνα από την είσοδο του, επιτρέποντας της να επηρεάσει την εσωτερική κατάσταση του μόνο όταν αυτή είναι ανοιχτή. Όταν η έξοδος της G_i είναι κοντά στο μηδέν η τιμή της εισόδου θα μπλοκαριστεί. Η πύλης εξόδου ελέγχει το πότε η κατάσταση του κελιού μπορεί να διαρρεύσει σε άλλα κομμάτια του δικτύου. Τέλος η πύλη λήθης επιτρέπει στην κατάσταση του κελιού να ξεχαστεί σε περίπτωση που η έξοδος της πύλης αυτής τείνει στο μηδέν.



Εικόνα 1.37: LSTM, Δίκτυο μακράς βραχυπρόθεσμης μνήμης [63]

Η κατάσταση ενός κελιού $s_i(t)$ υπολογίζεται ως:

$$s_i(t) = net_i(t) g_i^{in}(t) + g_i^{forget}(t) s_i(t-1)$$

Όπου $g_i^{in}(t)$ και g_i^{forget} αντιστοίχως είναι η πύλες εισόδου και λήθης. Το net το άθροισμα των εξωτερικών εισόδων (μετ' επίδρασης συναπτικών βαρών)

$$net_i(t) = h\left(\sum_j w_{ij}^{cell} c_j(t-1) + \sum_k w_{ik}^{cell} u_k(t-1)\right)$$

Όπου $h()$ είναι συνηθώς η συνάρτηση $f(x) = x$ και c_j είναι η έξοδος του κελιού j:

$$c_j(t) = \tanh\left(g_j^{out}(t) s_j(t)\right)$$

Όπου g_j^{out} είναι η πύλη εξόδου του κελιού j. Το αν μία πύλη είναι κλειστή ή ανοιχτή την χρονική στιγμή t υπολογίζεται ως εξής:

$$g_j^{type}(t) = \sigma\left(\sum_j w_{ij}^{type} c_j(t-1) + \sum_k w_{ik}^{type} u_k(t)\right)$$

Ανάλογα το είδος της πύλης αντικαθιστούμε το type με το είδος της πύλης (output, input, forget) και $\sigma()$ είναι η σιγμοειδής συνάρτηση.

1.5.2.4 Continuous Time RNN (CTRNN)

Τα RNN συνεχούς χρόνου CTRNN μελετήθηκαν, σε μια πρώιμη μορφή τους, στα μέσα του 20ου αιώνα (από τον Grossberg και άλλους) και διαδόθηκαν το 1984 από τον Hopfield στο πλαίσιο των μελετών του για συσχετιστική μνήμη. Χρησιμοποιούνται συχνά στο πεδίο της εξελικτικής ρομποτικής για την υλοποίηση όρασης [65], συνεργασίας [66] και γνωστικής συμπεριφοράς [67]. Πρόκεινται για νευρωνικά δίκτυα των οποίων οι νευρώνες ακολουθούν την γενική μορφή:

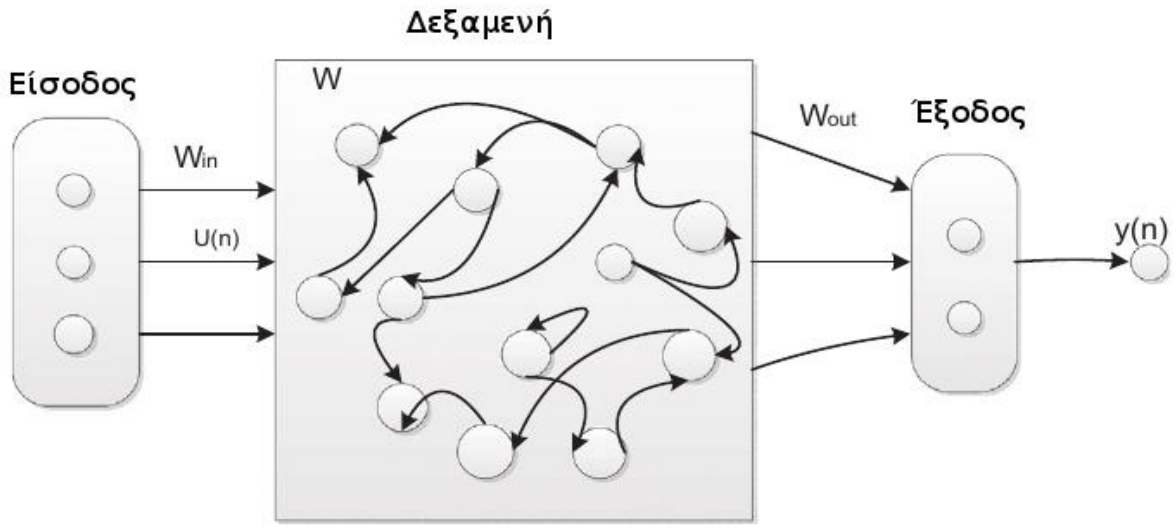
$$\tau_i y_i == y_i + \sum_{j=1}^{N} w_{ji}\sigma(y_j + b_j) + I_i \quad , i = 1, \dots, N$$

- y είναι η κατάσταση κάθε νευρώνα,
- τ η σταθερά χρόνου,
- w_{ji} το συναπτικό βάρος (από τον ι-οστό προς τον j-οστό νευρώνα),
- b_j μια σταθερά πόλωσης
- $\sigma(\dots) = \frac{1}{(1+e^{-x})}$
- I_ι αντιπροσωπεύει μια σταθερή εξωτερική έξοδο

Τα CTRNNs είναι αναμφισβήτητα το πιο απλό μοντέλο δυναμικού νευρωνικού δικτύου συνεχούς χρόνου. Παρά την απλότητα τους αποτελούν καθολικές δυναμικές προσεγγίσεις με την έννοια ότι για οποιοδήποτε περατό χρονικό διάστημα, μπορούν να προσεγγίσουν την πορεία οποιουδήποτε ομαλού δυναμικού συστήματός σε έναν συμπαγές υποσύνολο του $\mathbb{R}^n$. Η δυναμική των CTRNN έχει μελετηθεί εκτενώς και έχει καταστεί δυνατή η εκτενής ανάλυση, τουλάχιστον, μερικών εκ των γνωρισμάτων της δυναμικής τους. Το υπολογιστικό κόστος τους είναι σχετικά μικρό, κάτι το οποίο καθιστά δυνατή την αναπαράσταση ευμεγεθών κυκλωμάτων σχετικά γρήγορα. Τέλος τα CTRNNs μπορούν εύκολα να ερμηνευτούν βιολογικά, με την κατάσταση y να αναπαριστά την μέση δυνητικότητα της μεμβράνης ενός νευρικού κυττάρου και η έξοδος $\sigma(y)$ να σχετίζεται με την βραχυπρόθεσμη συχνότητα ενεργοποίησης του νευρώνα.

1.5.2.5 Echo State Networks (ESN)

Τα δίκτυα κατάστασης αντήχησης (ESN) αναπτύχθηκαν στο ινστιτούτο Fraunhofer το οποίο διατηρεί και την πατέντα για την εμπορική τους χρήση. Το όνομα τους προκύπτει από την μεταφορική αντίληψη της κατάστασης x(n) ενός δικτύου ως μια «αντήχηση» της ιστορίας των εισόδων του. Τα ESN χρησιμοποιούν μεγάλα, τυχαία παραγόμενα και αραιά συνδεδεμένα RNN ως «δεξαμενές» (reservoirs). Η έξοδος των δεξαμενών αυτών αποτελεί πλούσια πηγή σημάτων, τα οποία είναι προσβάσιμα με την χρήση ενός μοναδικού νευρώνα «ανάγνωσης» ή ακόμα και με χρήση ενός απλού γραμμικού συνδυασμού. Με την χρήση του συνδυασμού δεξαμενών και νευρώνων «ανάγνωσης» καθίσταστε δυνατή η προσέγγιση οποιουδήποτε επιθυμητού σήματος.



Εικόνα 1.38: ESN, Δίκτυο Κατάστασης Αντήχησης [68]

Όπως φαίνεται και στην Εικόνα 1.38 μια δεξαμενή μεγέθους N, K εισόδων και L εξόδων αποτελείται: από ένα πίνακα συναπτικών βαρών εισόδου W_{in}, μεγέθους $N * K$ και N νευρώνες αραιά συνδεδεμένους των οποίων τα βάρη δίνονται από ένα πίνακα W μεγέθους $N * N$. Τα βάρη των εξόδων δεν είναι μέρος της δεξαμενής και δίδονται ως πίνακας W_{out} μεγέθους $L * (K + N)$. Η κατάσταση μιας δεξαμενής την χρονική στιγμή n δίδεται από το διάνυσμα κατάστασης $x(n)$ το οποίο έχει ως στοιχεία τις εξόδους των νευρώνων εντός της δεξαμενής. Σε μία πιο περίπλοκη μορφή του ESN εισάγονται και βάρη ανατροφοδότησης W_{fb} από την έξοδο προς την δεξαμενή. Η σήμανση $U(n)$ συμβολίζει την εξωτερική είσοδο. Ο κανόνας για την ενημέρωση της κατάστασης είναι ο:

$$x(n+1) = tanh\big(Wx(n) + W_{in}u(n)\big)$$

Με ανατροφοδότηση:

$$x(n+1) = tanh\Big(Wx(n) + W_{in}u(n) + W_{fb}y(n)\Big)$$

Ο κανόνας υπολογισμού εξόδου δίνεται από την:

$$y = g(W_{out}\big(x(n+1); u(n+1); y(n)\big)$$

- η συνάρτηση $g(\dots)$ είναι η συνάρτηση ενεργοποίησης εξόδου συνήθως $tanh$ ή η $f(x) = x$
- Το ; συμβολίζει την πράξη συναλύσωσης πινάκων (concatenation)

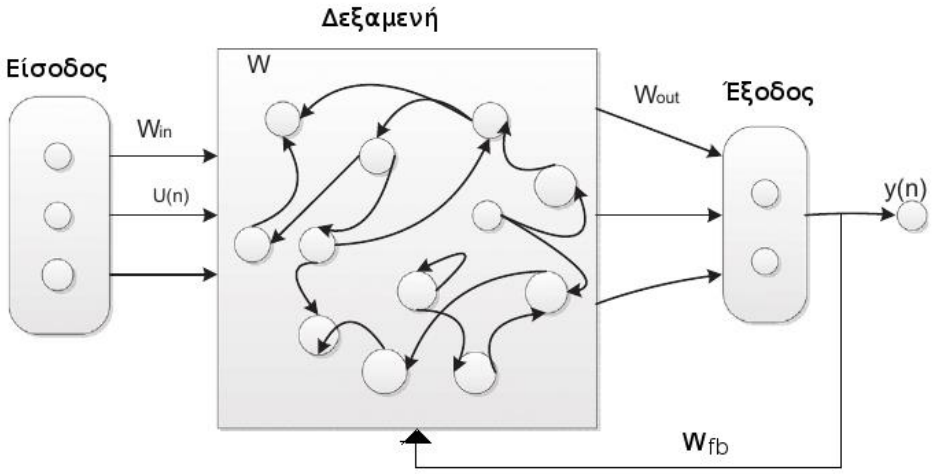
Το πλεονέκτημα που παρουσιάζουν τα ESN αναφορικά με παραδοσιακά RNN είναι ότι το μεγαλύτερο πλήθος των βαρών μπορεί να παραχθεί τυχαία και έτσι δεν χρίζεται εκπαίδευσης μειώνοντας έτσι το υπολογιστικό τους κόστος κατά ένα μεγάλο βαθμό. Μόνον τα βάρη εξόδου προσαρμόζονται ενώ τα άλλα παραμένουν αμετάκλητα. Επιπλέων, η επιφάνεια του μέσου τετραγωνικού σφάλματος (MSE), όσον αφορά τα βάρη εξόδου, μπορεί να μετατραπεί σε γραμμική επιφάνεια εφαρμόζοντας σχετική αντιστροφή της συνάρτησης ενεργοποίησης των νευρώνων εξόδου ανάλογα πάντα την επιθυμητή απάντηση. Έτσι η επιφάνεια του MSE γίνεται μονότροπη, επιτρέποντας την χρήσης διαφόρων μεθόδων όπως γραμμικά φίλτρα. Αντιθέτως, η εκπαίδευση απλών RNN παρουσιάζει εντόνως πολύτροπο RNE.

Η κραταιά μέθοδος κατά την χρήση ESN είναι η ανάθεση των βαρών δεξαμενής (W_{in}, W) με τυχαία δειγματοληψία τιμών από μια κατανομή (συνήθως ομοιόμορφη σε διάστημα $[-1,1]$). Στην συνέχεια ο πίνακας W κλιμακώνεται προκειμένου να αποκτήσει συγκεκριμένη φασματική ακτίνα μικρότερή του ενός προκειμένου το ESN να λειτουργήσει ορθά. Επιπλέων τα βάρη εισόδου W_{in} κλιμακώνονται κατά ένα παράγοντα μεταξύ του μηδέν και του ένα. Οι δεξαμενές που παράγονται έτσι ονομάζονται «κλασσικές δεξαμενές» (‘classical reservoirs’).

Τα ESN από την δημιουργία τους έχουν αναπτυχθεί περαιτέρω από μια ολοένα αυξανόμενη επιστημονική κοινότητά. Η κοινότητα αυτή έχει εντρυφήσει κυρίως στην προσαρμογή των κανόνων ενημέρωσης των βαρών καθώς και γενικότερα στην βελτιστοποίηση της απόδοσης τους. Έτσι λοιπόν αναφορικά με τον κανόνα ενημέρωσης βαρών, τα ESN έχουν διακλαδωθεί σε δύο κύριες κατηγορίες το ESN διαρρέοντος ολοκληρωτή (leaky integrator ESN: Leaky-ESN) και το αμετάβλητο ESN αναδιπλωμένου χρόνου (Time-Warping Invariant ESN: TWI-ESN).

1.5.2.5.1 Leaky-ESN

Το Leaky-ESN [68] _πρόκειται για ένα απλό ESN με την προσθήκη μιας μονάδας διαρρέοντος ολοκληρωτή στην συνάρτηση ενημέρωσης του. Η μονάδα αυτή μπορεί να



Εικόνα 1.39: Leacky ESN [68]

προσαρμόσει την είσοδο ενός ESN κατά την διάρκεια της εκπαίδευσης.

Οι σημάνσεις που ακολουθούνται είναι ίδιες με το ESN. Εισάγεται ένα νέο στοιχείο τα βάρη ανατροφοδότησης W_{fb} , πίνακας μεγέθους $N * L$. Η συνάρτηση (συνεχούς χρόνου) του Leaky-ESN γίνεται:

$$x = \frac{1}{c}\left(-ax + f\left(W_{in}u + Wx + W_{fb}y\right)\right)$$

$$y = g(W_{out}[x; u]$$

- Το c είναι μια σταθερά χρόνου (άρα $c > 0$)
- το a είναι η σταθερά διαρροής των νευρώνων εντός δεξαμενής (μπορεί να θεωρηθεί ως η ταχύτητα με την οποία η εξίσωση ενημέρωσης κατάστασης διακριτοποιείται στον χρόνο).

Οι παραπάνω εξισώσεις αν διακριτοποιηθούν στον χρόνο με μεθόδους Euler τότε προκύπτουν οι εξισώσεις:

$$x(n+1) = (1-a)x(n) + f\left(W_{in}u(n+1) + Wx(n) + W_{fb}y(n)\right)$$

$$y(n) = g(W_{out}[x(n); u(n)]$$

- Η συνάρτηση $f(\dots)$ όπως προαναφέρθηκε και στα απλά ESN είναι σιγμοειδής και συνήθως η $tanh$

Τα τυχαία παραγόμενα W_{in}, W, W_{fb} παραμένουν αμετάκλητα και κατά την εκπαίδευση προσαρμόζονται μόνο τα βάρη εξόδου W_{out} σύμφωνα με την ακόλουθη εξίσωση:

$$W_{out} = (X^{\tau}X)^{-1}X^{\tau}Y$$

- Το X^{τ} συμβολίζει την μεταφορά του πίνακα X
- Το $(X^{\tau}X)^{-1}$ συμβολίζει την αναστροφή του πίνακα $X^{\tau}X$

Κατά την εκπαίδευση λοιπόν επιδιώκεται η μείωση μιας συνάρτησης ρίζας του κανονικοποιημένο μέσου τετραγωνικό σφάλματος (NRMSE):

$$E(y,d) = \sqrt{\frac{\|y(n) - d(n)\|^2}{\|d(n) - \langle d(n)\rangle\|^2}}$$

- y είναι η έξοδος
- $d(n)$ είναι η επιθυμητή έξοδος (διδασκόμενη έξοδος)
- $\langle \quad \rangle$ δηλώνει την μέση τιμή.

1.5.2.5.2 TWI-ESN

Η αναδίπλωση του χρόνου των προτύπων εισόδου είναι ένα κοινό πρόβλημα όταν επιχειρείται αναγνώριση ανθρωπίνως παραγόμενης εισόδου καθώς και όταν αντιμετωπίζονται δεδομένα τα οποία μετατράπηκαν τεχνητά σε χρονοσειρές (ο όρος «αναδίπλωση του χρόνου» αναφέρεται στην αταξία κατά τον συγχρονισμό μια σειράς γεγονότων). Για την αντιμετώπιση τέτοιων δυσκολίων προτάθηκαν από τους M. Lukosevicius, H.Jaeger, D.Popovici, U.Siewert τα TWI-ESN (σε διάφορες βιβλιογραφίες παρουσιάζονται και ως TWIESN) τα οποία έχουν εφαρμοστεί, ως τώρα, κυρίως σε εφαρμογές φωτοβολταϊκών. Αποτελούν παραλλαγή των ESN και μπορούν αποτελεσματικά να αντιμετωπίσουν αναδιπλωμένο χρόνο σε κατά την αναγνώριση δυναμικών προτύπων.

Στην εξίσωση του leaky-ESN μπορώ να εισαχθεί την τιμή του χρονικού διαστήματος μεταξύ δύο διαδοχικών χρονικών βημάτων $\big(n - (n+1)\big)$ η οποία έχει διαιρεθεί με μια χρονική σταθερά. Η τιμή αυτή θα ονομαστεί χρονικό χάσμα Δt. Ενώ λοιπόν συνάρτηση της εξόδου θα παραμένει ως έχει:

$$y(n) = g(W_{out}[x(n); u(n)]$$

Η εξίσωση ενημέρωσης όμως θα μεταβληθεί μετά την προσθήκη του Δt ως εξής:

$$x(n+1) = (1 - a\Delta t)x(n) + \Delta t f\left(W_{in}u(n+1) + Wx(n) + W_{fb}y(n)\right)$$

Εφαρμόζοντας λοιπόν την θεωρεία αναλλοίωτης αναδίπλωσης χρόνου που αναπτύχθηκε το 2006 από τον Lukosevicius και άλλους μπορούμε να εφαρμόσουμε αναδίπλωση χρόνο αναλύοντας το χρονικό χάσμα σε υπο-χάσματα $\Delta t = \Delta t'(n+1)$, όπου $\Delta t'(n+1)$ είναι ένα ψευδο-χασμα μεταξύ των χρονικών βημάτων n και $n+1$ το οποίο μπορεί να γραφτεί και ως:

$$\Delta t'(n+1) = b * \|u(n+1) - u(n)\|$$

Όπου b σταθερός παράγοντας. Αντικαθιστώντας έτσι το $\Delta(t)$ στην εξίσωση ενημέρωσης με $\Delta t'(n+1)$ προκύπτει η εξίσωση ενημέρωσης ενός TWI-ESN:

$$x(n+1) = x(n) - b * \|u(n+1) - u(n)\| \left(ax(n) + f\left(W_{in}u(n+1) + Wx(n) + W_{fb}y(n)\right)\right)$$

Κατά την διαδικασία «συγκομιδής καταστάσεων» είναι συνηθισμένη πρακτική να προστίθεται ένα διάνυσμα θορύβου $\theta(n)$ στις καταστάσεις τις δεξαμενής. Γίνεται η υπόθεση ότι το $\theta(n)$ είναι ένα κατάλληλα κανονικοποιημένο διάνυσμα θορύβου πχ ομοιόμορφος θόρυβος στο διάστημα $[-0{,}5, 0{,}5]$ ή Gaussian θόρυβος. Με αυτή την προσθήκη η εξίσωση ενημέρωσης κατάστασης ενός TWI-ESN καταλήγει στην τελικής της μορφή:

$$x(n+1) = x(n) - b * \|u(n+1) - u(n)\| \left(ax(n) + f\left(W_{in}u(n+1) + Wx(n) + W_{fb}y(n)\right)\right) + \theta(n)$$

Διαφαίνεται λοιπόν ότι προκειμένου να επιτευχθεί βελτιστοποίηση της μάθησης πρέπει να γίνει κατάλληλη επιλογή των εξής μεγεθών: το πλήθος των νευρώνων δεξαμενής, το φασματικό εύρος του πίνακα βαρών W, η σταθερά διαρροής α και το χρονικό χάσμα μεταξύ δύο χρονικών βημάτων $\Delta t(n)$. Κατά την εκπαίδευση γίνεται προσπάθεια ελαχιστοποίησης του μέσου τεραγονικού κόστους (MSE) μεταξύ της εξόδου του δικτύου και της επιθυμητής εξόδου. Ακολουθεί ο αλγόριθμος της εκπαιδευτικής διαδικασίας στο σύνολο της.

- Δημιουργία RNN με K νευρώνες εισόδου και L νευρώνες εξόδου. Εισαγωγή εκπαιδευτικής εισόδου και περισυλλογή των καταστάσεων ενεργοποίησης της δεξαμενής $x(n)$ που ανταποκρίνονται σε αυτή την είσοδο
- Υπολογισμός των βαρών εξόδου W_{out} χρησιμοποιώντας γραμμική παλινδρόμηση (linear regression) ελαχιστοποιώντας το MSE μεταξύ των εξόδων του δικτύου $y(n)$ και της επιθυμητής εξόδου $y^{(i)}(n)$
- Εγγραφή του W_{out} στις συνδέσεις εξόδου. Το δίκτυο εκπαιδεύτηκε.

- Χρήση του εκπαιδευμένου TWI-ESN για τον υπολογισμό του $y(n)$ για την νέα είσοδο $u(n)$

Για την παραγωγή των βαρών δεξαμενής W ακολουθείται η εξής διαδικασία:

- Τυχαία παραγωγή πίνακα W_0. Ο πίνακας αυτός αντιπροσωπεύει την πυκνότητα των συνδέσεων μεταξύ των νευρώνων της δεξαμενής η οποία τίθεται συνήθως μεταξύ του 1% και 5%
- Κανονικοποίηση του πίνακα W_0 από την οποία προκύπτει ο $W_1 = \frac{W_0}{\sigma_{max}(W_0)}$, όπου $\sigma_{max}(W_0)$ είναι η φασματική ακτίνα του W_0
  - Αλλαγή κλίμακας του W_1 κατά παράγοντα $0 < \alpha < 1$ από την οποία προκύπτει ο $W = aW_1$ο οποίος έχει φασματική ακτίνα μικρότερη το ενός. Μια μελέτη από τους Venayagamoorthy και Shishir, το 2009 έχει δείξει ότι η βέλτιστη τιμή για το α είναι γύρο στο 0,8.

1.5.3 Boltzman Machines (BM)

Η μηχανή Boltzmann είναι ένα στοχαστικό δυαδικό νευρωνικό δίκτυο το οποίο αναπτύχθηκε το 1985 από τους Geoffrey Hinton και Terry Sejnowski [27] [16] [69] [70]. Το όνομα τους προκύπτει από την διανομή Boltzmann την οποία ακολουθεί η συνάρτηση δειγματοληψίας που υλοποιούν. Οι μηχανές Boltzmann αποτελούν το χαρακτηριστικότερο δείγμα των ενεργειακών μοντέλων (energy-based models). Αποτελείται από στοχαστικούς νευρώνες (σε άλλες βιβλιογραφίες ονομάζονται νευρωνικοειδής μονάδες (neuron-like units), παρόλα αυτά θα προτιμηθεί όρος «νευρώνες» όπως εισάγεται από το [26])) οι οποίοι εμμέσου μιας πιθανοτικής διαδικασίας, δύνανται να βρίσκονται σε μία εκ των δύο δυνατών καταστάσεων (ενεργός ή κλειστός). Ένα άλλο ιδιαίτερο χαρακτηριστικό των μηχανών Boltzmann είναι η χρήση συμμετρικών συναπτικών συνδέσεων μεταξύ των νευρώνων του, χαρακτηριστικό εμπνευσμένο και από φυσικές στατιστικές διαδικασίες.

Οι μηχανές Boltzmann πρόκεινται ουσιαστικά για Στοχαστικά δίκτυα Hopfield στα οποία έχουν εισαχθεί στρώματα κρυφών νευρώνων και είναι αποδοτικές σε εφαρμογές μοντελοποίησης δυαδικών δεδομένων. Μία μηχανή Boltzmann λοιπόν, στην οποία δίδεται μια ομάδα δυαδικών διανυσμάτων μπορεί να χρησιμοποιήσει τούς κρυφούς νευρώνες της προκειμένου να δημιουργήσει ένα μοντέλο το οποίο θα αναθέσει μία πιθανότητα σε κάθε πιθανό διάδικο διάνυσμα. Μια εφαρμογή που θα μπορούσε να έχει μια μηχανή Boltzmann είναι η παρακολούθηση σύνθετων συστημάτων με σκοπό την ανίχνευση ασυνήθιστων συμπεριφορών. Πιο συγκεκριμένα έστω ότι σε ένα πυρηνικό εργοστάσιο όλοι οι πίνακες ελέγχου παρήγαν δυαδικά δεδομένα. Έτσι κάθε δυαδικό διάνυσμα που παράγει ένας τέτοιος πίνακας συμβολίζει μια διαφορετική κατάσταση του εργοστασίου. Θα μπορούσε να κατασκευαστεί μια μηχανή Boltzmann στην οποία θα ανατεθεί να «χτίσει» ένα μοντέλο των κανονικών καταστάσεων και στην συνέχεια να παρατηρούνται τυχών αποκλίσεις από αυτό εγείροντας συναγερμό σε τέτοιες περιπτώσεις.

Μια μηχανή Boltzmann αποτελείται από μια ομάδα ορατών νευρώνων $v \in \{0,1\}^D$ και από μία ομάδα κρυφών νευρώνων $h \in \{0,1\}^P$. Εφέσων είναι ενεργειακό μοντέλο όλα τα χρήσιμα μέτρα ορίζονται ως ενέργειες της συν-διαμόρφωσης των κρυφών και ορατών νευρώνων. Η ενέργεια μίας κατάστασης $\{v, h\}$ ορίζεται ώς:

$$E(v,h;\theta) = -\frac{1}{2}v^T L v - \frac{1}{2}h^T J h - v^T W h$$

- Όπου $\theta = \{W, L, J\}$ είναι οι παράμετροι του δικτύου (συναπτικά βάρη και σταθερές πόλωσης οι οποίες χάριν ευκολίας θεωρούνται ως το μηδενικό βάρος του εκάστοτε στρώματος , $b_j = w_{j.0}$).
  - W παράμετροι από ορατό προς κρυφό.
  - L παράμετροι από ορατό προς ορατό.
  - J παράμετροι από κρυφό προς κρυφό.
- Όπου v^T, h^T είναι πίνακας με τις δυαδικές καταστάσεις όλων των ορατών και αοράτων νευρώνων αντίστοιχα.

Εικόνα 1.40: Γράφος Boltzmann Machine [70]

Η ενεργειακή συνάρτηση μπορεί να γραφτεί και σε μορφή αθροισμάτων, δηλώνοντας ξεχωριστά τις σταθερές πόλωσης. Η συνάρτηση θα δοθεί και με αυτόν τον τρόπο καθώς είναι ευρέως διαδεδομένη στην σχετική βιβλιογραφία και καθώς η κατανόηση της καθίσταται ευκολότερη.

$$E(v,h) = -\sum_{i\in vis} v_i b_i - \sum_{k\in hid} h_k b_k - \sum_{i<j} v_i v_j w_{ij} - \sum_{i,k} v_i h_k w_{ik} - \sum_{k<l} h_k h_l w_{kl}$$

- Όπου v_ι, h_ι η δυαδική κατάσταση του i-οστού ορατού και κρυφού νευρώνα αντίστοιχα.
- Όπου b_i η σταθερά πόλωσης του i-οστού νευρώνα και άρα οι πρώτοι δύο όροι $-\sum_{i\in vis} v_i b_i - \sum_{k\in hid} h_k b_k$ αναφέρονται στην επιρροή των σταθερών πόλωσης
- Οι υπόλοιποι όροι, $-\sum_{i<j}(v_i v_j w_{ij}) - \sum_{i,k}(v_i h_k w_{ik}) - \sum_{k<l}(h_k\, h_l\, w_{kl})$, αναφέρονται στην νευρωνική αλληλεπίδραση ορατών με ορατούς, ορατών με κρυφούς και των κρυφών με κρυφούς. Προκειμένου να αποφευχθεί η εμφάνιση διπλοτύπων εισάγεται η συνθήκη $i < j$ αντιθέτως θα έπρεπε να πολλαπλασιαστούν τα αθροίσματα με $\frac{1}{2}$.

Να σημειωθεί ότι τα συναπτικά βάρη είναι συμμετρικά δηλαδή $w_{ji} = w_{ij},\ \forall\, i,j$ και επίσης τα στοιχεία της διαγώνιου των πινάκων L, J έχουν τεθεί ίσα με 0 ($w_{ii} = 0,\ \forall\, i$). Η πιθανότητα να παρατηρηθεί μια συγκεκριμένη διαμόρφωση των μεταβλητών (διαμόρφωση κρυφών και ορατών νευρώνων) εξαρτάται από την ενέργεια αυτής της διαμόρφωσης και την ενέργεια όλων των διαμορφώσεων:

$$P(v,h;\theta) = \frac{1}{Z}\exp\big(-E(v,h;\theta)\big)$$

- Όπου $Z(\theta)$, η ενέργεια όλων των διαμορφώσεων, είναι η συνάρτηση διχοτόμησης (partition function) η οποία επιστρέφει τον μέσω όρο όλων των

πιθανών ζευγαριών ορατών και κρυφών διανυσμάτων και χρησιμοποιήτε για κανονικοποίηση.

$$Z(\theta) = \sum_{v}\sum_{h} \exp\big(-E(v,h;\theta)\big)$$ (Συνάρτηση Διαμόρφωσης)

Η χρήση αυτής της εξίσωσης συνίσταται μόνο για μηχανές με μικρό αριθμό νευρώνων εξαιτίας του μεγάλου υπολογιστικού κόστους της. Σε περιπτώσεις μεγάλων μηχανών γίνεται χρήση άλλων μεθόδων όπως αυτή της τυχαίας δειγματοληψίας με χρήση Markov Chain Monte Carlo [69].

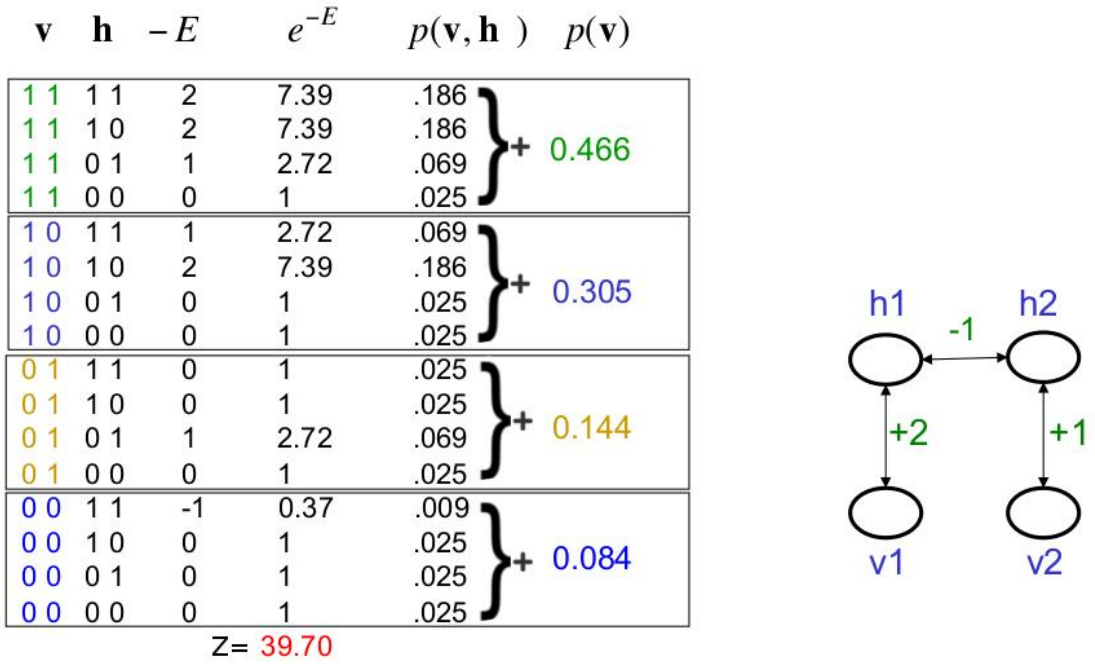
Αντίστοιχα η πιθανότητα μιας συγκεκριμένης διαμόρφωσης ορατών νευρώνων ισούται με το άθροισμα όλων τον διαμορφώσεων των κρυφών νευρώνων προς το σύνολο των πιθανών διαμορφώσεων:

$$p(v;\theta) = \frac{p^*(v;\theta)}{Z(\theta)} = \sum_{h} P(v,h;\theta) = \frac{1}{Z(\theta)}\sum\nolimits_{h} \exp\big(-E(v,h;\theta)\big)$$

- Όπου p^* συμβολίζει μη-κανονική πιθανότητα

Ακολουθεί ένα παράδειγμα μίας μηχανής Boltzmann με δύο κρυφούς και δύο ορατούς νευρώνες [16]. Οι όροι των σταθερών πόλωσης έχουν αφαιρεθεί για λόγους ευκολίας. Στο σχήμα που ακολουθεί (Εικόνα 1.41) έχει ακολουθηθεί η εξής διαδικασία:

1. $\boldsymbol{v,h}$**:** Καταγράφεται κάθε πιθανή κατάσταση (συνδυασμός διαμορφώσεων ορατών και κρυφών νευρώνων του δικτύου). Σύνολο $4^2 = 16$ πιθανές καταστάσεις.
2. $\boldsymbol{-E}$**:** Υπολογίζεται μέσω της ενεργειακής εξίσωσης, το αρνητικό της ενέργειας της εκάστοτε κατάστασης.
3. $\boldsymbol{e^{-E}}$: Υπολογίζεται οι μη- κανονική πιθανότητα εμφάνισης τής κάθε κατάστασης.
4. $\boldsymbol{Z = 39,70}$**:** Αθροίζοντας τα στοιχεία της τέταρτης στήλης υπολογίζεται η τιμή της συναρτήσεως διχοτόμησης
5. $\boldsymbol{p(v,h)}$**:** Υπολογίζονται η πιθανότητα της κάθε κατάστασης κανονικοποιόντας την μη κανονική πιθανότητα που υπολογίστηκε προηγούμενος $\frac{p(v,h)}{Z}$
6. $\boldsymbol{p(v)}$**:** Υπολογίζεται η πιθανότητα συγκεκριμένης κατάστασης των ορατών νευρώνων. Αυτό επιτυγχάνεται αθροίζοντας τα στοιχεία του κάθε μπλοκ.



Εικόνα 1.41: Διαδικασία εκμάθησης μίας Μηχανής Boltzmann [16]

Εφόσον η μηχανή Boltzmann είναι μια στοχαστική μηχανή είναι λογικό να αναζητηθούν μέθοδοι από την θεωρεία της πιθανότητας προκειμένου να δοθεί ένα μέτρο της απόδοσης της [69]. Ένα τέτοιο κριτήριο είναι η συνάρτηση της πιθανοφάνειας και πιο συγκεκριμένη η λογαριθμική πιθανοφάνεια την οποία μία μηχανή Boltzmann καλείται να μεγιστοποιήσει. Στο υπόλοιπο αυτής της ενότητας θα ακολουθηθούν οι εξής σημάνσεις:

- $\Im$: Ένα δυαδικό δείγμα προς εκπαίδευση το οποίο ακολουθάει την πιθανοτική κατανομή ενδιαφέροντος.
- h, v, x : Συγκεκριμένες καταστάσεις των κρυφών και των ορατών και όλων των νευρώνων αντίστοιχα.
- H, V, X : Το σύνολό των καταστάσεων των κρυφών και των ορατών και όλων των νευρώνων αντίστοιχα.
- w : Τα συναπτικά βάρη όλου του δικτύου

Η λειτουργεία μίας μηχανής Boltzmann γίνεται σε δύο στάδια τα οποία έχουν ως στόχο την ελαχιστοποίηση της πιθανότητας $P(V = v) = \frac{\sum_{h \in \Im} \exp(-E(x))}{\sum_{v \in \Im} \sum_{h \in \Im} \exp(-E(x))}$:σ

1. Θετικό Στάδιο: Το δίκτυο βρίσκεται σε κατάσταση σύσφιξης, δηλαδή είναι υπό την άμεση επήρεια του $\Im$. Κατά την διάρκεια αυτού του σταδίου εντοπίζονται οι διαμορφώσεις των κρυφών νευρώνων οι οποίες συνεργάζονται βέλτιστα με το διάνυσμα εισόδου $\Im$ και στην συνέχεια μειώνετε η ενέργεια τους με αποτέλεσμα να μειωθεί ο αριθμητής της $p(V = v)$.
2. Αρνητικό Στάδιο: Γίνεται επιτρεπτό στο δίκτυο να λειτουργήσει μόνο του και άρα δεν υπάρχει είσοδος. Κατά την διάρκεια αυτού το σταδίου εντοπίζονται οι κοινές διαμορφώσεις συνεισφέρουν μέγιστα στην συνάρτηση διχοτόμησης (παρονομαστής της $p(V = v)$) και αυξάνεται η ενέργεια τους.

Θεωρούμε ότι τα συναπτικά βάρη w του δικτύου είναι ήδη αρχικοποιημένα. Η πιθανότητα οι οπτικοί νευρώνες να βρίσκονται στην κατάσταση x_a είναι $P(V = v)$. Γίνεται επίσης η θεώρηση ότι όλες οι πιθανές τιμές του x_a είναι στατιστικά ανεξάρτητες και άρα η ολική κατανομή πιθανότητας είναι η παραγοντική κατανομή (factorial distribution) $\prod_{x_a \in \Im} P(V = v)$. Για να σχηματιστεί η λογαριθμική πιθανοφάνεια $L(w)$, λογαριθμίζεται η συνάρτηση εκλαμβάνοντας ως άγνωστο, το διάνυσμα συναπτικών βαρών w:

$$L(w) = \log \prod_{v \in \Im} P(V = v) = \sum_{v \in \Im} \log P(V = v)$$

Η παραπάνω συνάρτηση μπορεί να μετασχηματιστεί, λαμβάνοντας υπ. όψη ότι η πιθανότητα $P(V = v) = \frac{1}{Z} \sum_h \exp\big(-E(x)\big)$ και ότι $Z = \sum_v \sum_h \exp\big(-E(x)\big) = \sum_x exp\big(-E(x)\big)$, στην εξής μορφή:

$$L(w) = \sum_{v \in \Im} \left(\log \sum_h \exp\big(-E(x)\big) - \log \sum_x \exp\big(-E(x)\big) \right)$$

Διαφορίζοντας το $L(w)$ ως προς w και έπειτα από μερικούς υπολογισμούς καταλήγουμε στην:

$$\frac{\vartheta L(w)}{\vartheta w_{ij}} = \sum_{v \in \Im} \left(\sum_h P(H = h|V = v)\, x_j x_i - \sum_x P(X = x) x_j x_i \right)$$

Χάριν απλοποίησης εισάγουμε δύο νέους ορισμούς:

$$p_{ji}^{+} = \langle x_j x_i \rangle^{+} = \sum_{v \in \Im} \sum_h P(H = h|V = v)\, x_j x_i$$

$$p_{ji}^{-} = \langle x_j x_i \rangle^{-} = \sum_{v \in \Im} \sum_x P(X = x) x_j x_i$$

Οι τιμές p_{ji}^{+}, p_{ji}^{-} μπορούν να εκληφθούν ως η μέση ενεργοποίηση νευρώνων ή αλλιώς η συσχέτιση μεταξύ των καταστάσεων των νευρώνων i και j όταν το δίκτυο βρίσκεται στο Θετικό και στο Αρνητικό Στάδιο αντίστοιχα. Με την εισαγωγή τον ορισμών αυτών το διαφορικό της $L(w)$ γίνεται:

$$\frac{\vartheta L(w)}{\vartheta w_{ij}} = p_{ji}^{+} - p_{ji}^{-}$$

Ό στόχος της εκμάθησης Boltzmann όπως προαναφέρθηκε είναι να μεγιστοποιήσει την λογαριθμική πιθανοφάνεια $L(w)$. Για την επίτευξη αυτού το στόχου μπορεί να χρησιμοποιηθεί ο αλγόριθμος της απότομης ανόδου (gradient ascent) σε συνδυασμό με και μία μορφή οπισθοδρομικής διάδοσης σφάλματος. (Για περαιτέρω [71]):

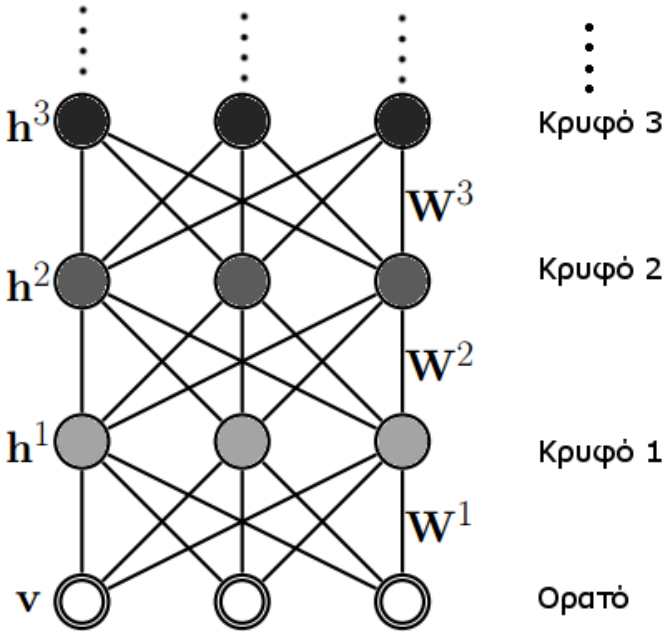
$$\Delta w_{ij} = \eta \frac{\vartheta L(w)}{\vartheta w_{ij}} = \eta\left(p_{ji}^{+} - p_{ji}^{-}\right)$$

- Όπου η είναι ο ρυθμός εκμάθησης όπως περιεγράφηκε στην ανάλυση του αλγορίθμου απότομης καθόδου.

1.5.3.1 Deep Boltzmann Machines (DBM)

Σε γενικές γραμμές, η εκπαίδευση μίας πλήρως συνδεδεμένης Μηχανής Boltzmann σπάνια παρουσιάζει ενδιαφέρων [70]. Αντί αυτής χρησιμοποιούνται βαθιές πολύ-στρωματικές μηχανές Boltzmann, των οποίων το κάθε στρώμα ανιχνεύει περίπλοκες συσχετίσεις μεταξύ των καταστάσεων κατώτερων κρυφών επιπέδων. Οι βαθιές μηχανές Boltzmann παρουσιάζουν ενδιαφέρον για διάφορους λόγους, μεταξύ άλλων:

- Έχουν προοπτική στο να μάθουν εσωτερικές αναπαραστάσεις δεδομένων οι οποίες γίνονται ολοένα και πιο περίπλοκες κάτι το οποίο θεωρείται ένας πολλά υποσχόμενος τρόπος για την αναγνώριση φωνής και αντικειμένων.
- Αναπαραστάσεις υψηλού επιπέδου μπορούν να κατασκευαστούν ΄χρησιμοποιώντας ως είσοδο δεδομένα χωρίς ετικέτες τα οποία υπάρχουν σε μεγάλο πλήθος. Στην συνέχεια μπορούν να χρησιμοποιηθεί μικρή ποσότητα δεδομένων με ετικέτα για την τελειοποίηση του μοντέλου.



Εικόνα 1.42: Βαθιά Μηχανή Boltzmann [70]

Η εξισώσεις που περιγράφουν μια βαθιά Μηχανή Boltzmann είναι αντίστοιχες με αυτές της απλής με την προσθήκη των περαιτέρω παραμέτρων. Ενδεικτικά, για ένα δίκτυο δύο κρυφών σταδίων:

$$Ενεργειακή\ Συνάρτηση: E(v, h^1, h^2) = -v^T W^1 h^1 - h^{1T} W^2 h^2$$

$$p(v) = \frac{1}{Z(\theta)} \sum_{h^1, h^2} \exp\left(E(v, h^1, h^2; \theta)\right)$$

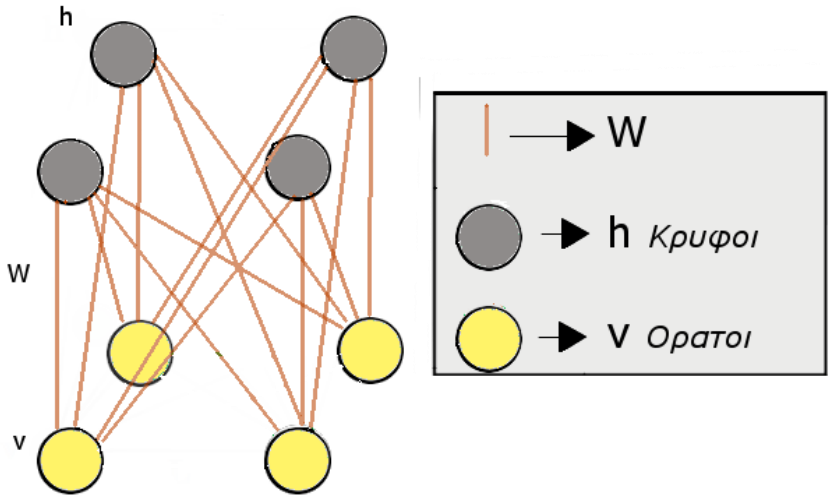
$$κ.ο.κ.$$

1.5.3.2 Restricted Boltzmann Machines (RBM)

Οι περιορισμένες μηχανές Boltzmann είναι παραμετροποιημένα παραγωγικά μοντέλα τα οποία αναπαριστούν μια πιθανοτική κατανομή. Δοθέντων των δεδομένων εκπαίδευσης το RBM προσαρμόζει της παραμέτρους του προκείμενου η πιθανοτική κατανομή που αυτό αντιπροσωπεύει να ταιριάζει όσον το δυνατόν περισσότερο με τα δεδομένα εκπαίδευσης. Τα RBM είναι χρήσιμα σε εφαρμογές μείωσης διάστασης δεδομένων, ταξινόμησης, εκμάθησης χαρακτηριστικών, φιλτραρίσματος (collaborative filtering) και μοντελοποίησης δεδομένων.

Η βασική διαφορά μεταξύ των περιορισμένων και των απλών μηχανών Boltzmann έγκειται στο γεγονός ότι δεν υπάρχουν οι διανευρωνικές συνδέσεις κρυφών με κρυφούς και ορατών με ορατούς. Αυτό έχει σαν αποτέλεσμα, την επιτάχυνση της εκπαιδευτικής διαδικασίας του δικτύου καθώς οι κρυφοί νευρώνες είναι πλέων υπό συνθήκη ανεξάρτητοι δοθέντος

ορατού διανύσματος και άρα αμερόληπτα δείγματα από το $p_{ji}^{+} = \langle x_j x_i \rangle^{+}$ μπορούν να αποκτηθούν σε ένα, παράλληλο βήμα. Ο διμερής γράφος του δικτύου λαμβάνει την μορφή:



Εικόνα 1.43: Restricted Boltzmann Machines, Περιορισμένες Μηχανές Boltzmann

Οι συναρτήσεις που περιγράφουν την λειτουργεία ενός RBM στο ίδιο πνεύμα με αυτές ενός Boltzmann Machine είναι οι εξής [72] [73]:

$$Ενεργειακή\ Συνάρτηση: E(v,h;\theta) = -\sum_{i,j} v_i h_j w_{i,j} - \sum_i v_i b_i - \sum_j h_j b_j$$
$$= \sum_{i=1}^{V}\sum_{j=1}^{H} v_i h_j w_{i,j} - \sum_{i=1}^{V} v_i b_i - \sum_{j=1}^{H} h_j b_j$$

$$Πιθανότητα\ διαμόρφωσης\ νευρώνων: P(v,h;\theta) = \frac{1}{Z}\exp\big(-E(v,h;\theta)\big)$$

$$Πιθανότητα\ διαμόρφωσης\ ορατών\ νευρώνων: P(v;\theta) = \frac{1}{Z}\sum_h \exp\big(-E(v,h;\theta)\big)$$

$$Λογαριθμική\ πιθανοφάνεια\ (μοναδικής\ κατάστασης): \mathrm{L}(w) = \log\sum_h \exp\big(-E(v,h)\big) - \log Z$$
$$= \log\sum_h \exp\big(-E(v,h)\big) - \log\sum_v\sum_h \exp\big(-E(v,h;\theta)\big)$$

$$\frac{\vartheta L(w)}{\vartheta w_{ij}} = p_{ji}^{+} - p_{ji}^{-} = v_i * P\big(h_J = 1|v\big) - P\big(v_i = 1|h_j = 1\big)$$

$$Δεσμευμένη\ κατανομη\ P(h|v) = \prod_j p(h_j|v) := P(h_j = 1|v;\theta) = \sigma\left(\sum_{i=1}^{V}(w_{ij}v_i + b_j)\right)$$

$$Δεσμευμένη\ κατανομη\ P(v|h) = \prod_i p(v_i|h) := P(v_i = 1|h;\theta) = \sigma\left(\sum_{i=1}^{H}(w_{ij}h_i + b_j)\right)$$

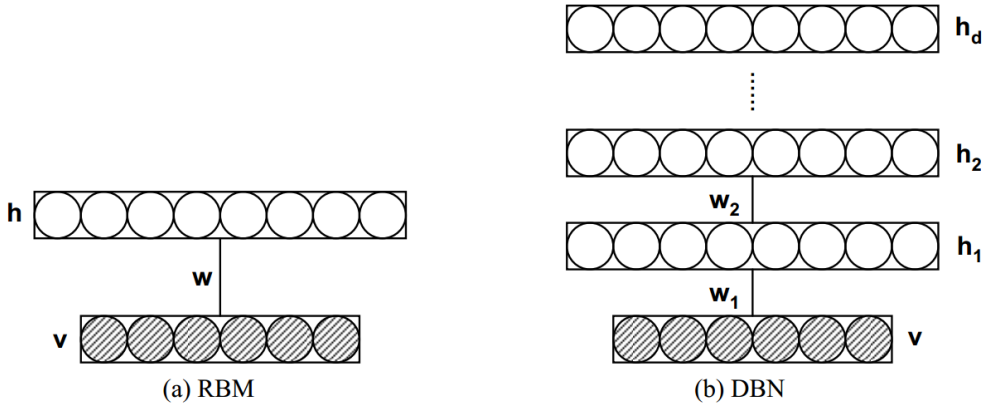
Ένα εκπαιδευμένο RBM θα μάθει την δομή των εισαγόμενων δεδομένών εμμέσου μιας διαδικασίας συνεχούς ανακατασκευής τους. Όσο προχωράει η διαδικασία ανακατασκευής τόσο αυξάνεται η ομοιότητα των ανακατασκευασμένων δεδομένων με τα αυτά που εισήχθησαν. Ο πιο καθιερωμένος αλγόριθμος για την εκπαίδευση ενός τέτοιου δικτύου είναι ο CD (Constructive Divergence Algorithm). Ο CD έχει σχεδιαστεί με τέτοιο τρόπο ώστε τουλάχιστον η κατεύθυνσής της κλίσης είναι σωστή ακόμα και αν το μέγεθος της είναι λανθασμένο. Η δημοτικότητα του αλγορίθμου έχει οδηγήσει στην ανάπτυξη πολλών παραλλαγών του με δημοφιλέστερη την CD-1 του οποίου θα γίνει μια σύντομη περιγραφή στην επόμενη παράγραφο. Μερικές παραλλαγές του, μεταξύ άλλων, είναι ο CD-10 (συνιστάτε σε περιπτώσεις που ο υπολογιστικός χρόνος δεν είναι παράγοντας), ο mean field CD (MF CD) (με το πρόσθετο πλεονέκτημα ότι αποτελεί ντετερμινιστική εκτίμηση της κλίσης και άρα είναι δυνατή η χρήση μεγαλύτερων ρυθμών μάθησης η.) και ο PCD (Persistent Contrastive Divergence).

Ο στόχος του CD-1 είναι ο υπολογισμός μιας εκτίμησης για τιμή του p_{ji}^{-}. Για να υπολογιστεί η τιμή αυτή ο καθιερωμένος τρόπος είναι η χρήση μιας αλυσίδας Markov της οποίας όμως το χρονικό κόστος είναι πολύ μεγάλο όταν «τρέχει» για πολλαπλά βήματα. Για να αντιμετωπιστεί η αναποτελεσματικότητα αυτή ο CD-1 εκμεταλλεύεται το γεγονός ότι σε κάθε βήμα ανανέωσης παραμέτρων το μοντέλο δεν αλλάζει παρά ελάχιστα. Για αυτό τον λόγο η αλυσίδα Markov αρχικοποιείται στην κατάσταση στην οποία κατέληξε το προηγούμενο μοντέλο. Η αρχικοποίηση αυτή είναι συχνά, αρκετά κοντά στην κατανομή του μοντέλου, παρά το ότι το μοντέλο έχει αλλάξει ελάχιστα μετά την τελευταία ανανέωση των παραμέτρων.

Ο αλγόριθμος CD-1 που περιεγράφηκε έχει χρησιμοποιηθεί και σε άλλα δίκτυα όπως είναι τα Sigmoid Belief Networks. Λαμβάνοντας υπ’ όψων όμως, μια μοναδική ιδιότητα των RBMs ο αλγόριθμος μπορεί να παραλλαχθεί και να γίνει ακόμα πιο αποδοτικός. Ο ανανεωμένος αλγόριθμος, ειδικά σχεδιασμένος για RBM, δεν είναι άλλος από τον PCD. Ο αλγόριθμος αυτός εκμεταλλεύεται το γεγονός ότι στα RBMs αναζητείται μία μοναδική κατανομή από την οποία χρειάζονται δείγματα και όχι μια κατανομή για κάθε διάνυσμα εισόδου. Έτσι ο PCD μπορεί να υπολογίσει εκτιμήσεις των κλίσεως (είτε υπό-ομάδων είτε με σειριακό τρόπο) τόσο κατά το θετικό όσο και για το αρνητικό στάδιο, χρησιμοποιώντας μόνο λίγες από τις καταστάσεις του μοντέλου για των υπολογισμό τους. Να σημειωθεί ότι ο αλγόριθμος αυτός λειτουργεί καλύτερα με χρήση μικρών τιμών ρυθμού μάθησης.

1.5.3.2.1 Deep Belief Networks (DBN)

Τα βαθιά δίκτυα πίστης (Deep Belief Networks) [74] είναι πιθανοτικά παραγωγικά μοντέλα τα οποία αποτελούνται από πολλαπλά στρώματα στοχαστικών λανθανόντων μεταβλητών. Οι μεταβλητές αυτές συνήθως έχουν δυαδικές τιμές και αποτελούν ουσιαστικά τους κρυφούς νευρώνες του δικτύου. Τα DBN πρόκεινται ουσιαστικά για διμερής γράφος αποτελούμενους από πολλαπλά RBM στοιβαγμένα το ένα πάνω στο άλλο και εκπαιδευμένα με μία άπληστη μέθοδο. Τα DBN έχουν χρησιμοποιηθεί για πολλαπλές εφαρμογές όπως αναγνώρισης και παραγωγής εικόνας και βίντεο, ανίχνευση κίνησης και προ-εκπαίδευσης άλλων βαθιών αρχιτεκτονικών.



Εικόνα 1.44: DBN, Δίκτυο Βαθιάς Πίστης [73]

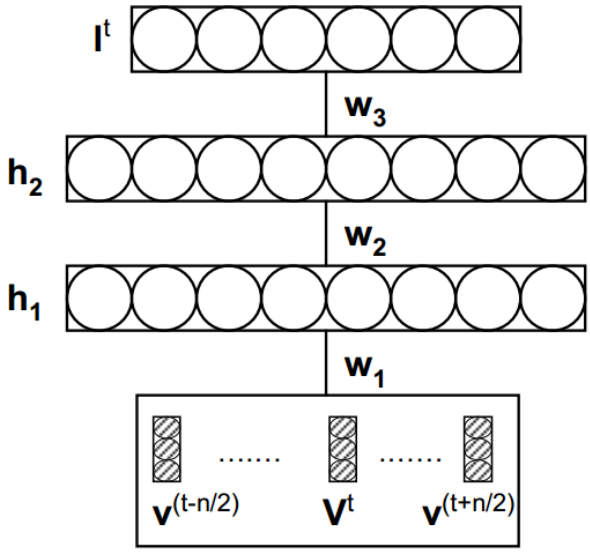
Κάθε κρυφό στρώμα ενός DBN [73] μαθαίνει να αναπαριστά χαρακτηριστικά τα οποία απαθανατίζουν συσχετίσεις υψηλότερου στρώματος των δεδομένων εισόδου. [74] Γενικά ένα τέτοιο δίκτυο εκπαιδεύεται ένα στρώμα την φορά χρησιμοποιώντας τις λανθάνουσες μεταβλητές του ενός στρώματος (οι οποίες προκύπτουν από τα δεδομένα εισόδου) ως την είσοδο του επόμενου. Αυτή η αποδοτική άπληστη μέθοδος μάθηση μπορεί είτε να χρησιμοποιηθεί από μόνη της είτε σε συνδυασμό με άλλες εκπαιδευτικές μεθόδους που βελτιώνουν τα βάρη προκείμενου να βελτιωθεί η λειτουργεία του δικτύου.

Η κεντρική ιδέα πίσω από ένα τέτοιο δίκτυο είναι ότι οι παράμετροι θ, τις οποίες μαθαίνει ένα RBM, μπορούν να ορίσουν τόσο την κατανομή $P(v|h;\theta)$ όσο και την προγενέστερη κατανομή επί των κρυφών νευρώνων $P(h|\theta)$ και άρα η πιθανότητα να παραχθεί ένα συγκεκριμένο διάνυσμα στους οπτικούς νευρώνες v μπορεί να γραφεί ως:

$$P(v) = \sum_{h} P(h|\theta)P(v|h;\theta)$$

Άρα αφού μέσω εκπαίδευσης εκμαθευτεί το θ, το $P(v|h;\theta)$ διατηρείται ενώ το $P(h|\theta)$ αντικαθίσταται από ένα καλύτερο μοντέλο το του οποίου η εκμάθηση γίνεται χρησιμοποιώντας τα διανύσματα των κρυφών νευρώνων ως δεδομένα εκπαίδευσης για το επόμενη RBM. Έχει αποδειχτεί ότι αυτή αντικατάσταση, εάν πραγματοποιηθεί με κατάλληλο ,τρόπο μπορεί να βελτιώσει ένα μεταβολικό κάτω όριο (variational lower bound) επί της πιθανότητας των δεδομένων εκπαίδευσης υπό το σύνθετο μοντέλο.

Όπως προαναφέρθηκε τα DBN έχουν χρησιμοποιηθεί σε εφαρμογές φωνητικής αναγνώρισης με μεγάλη επιτυχία. [73] Μια τέτοια εφαρμογή αναπτύχθηκε στο πανεπιστήμιο του Toronto με εντυπωσιακά αποτελέσματα. Χρησιμοποιηθήκαν δύο αρχιτεκτονικές αλλά στην παρούσα εργασία θα παρουσιαστούν κάποια από τα αποτελέσματα μόνο μίας εκ των δύο.



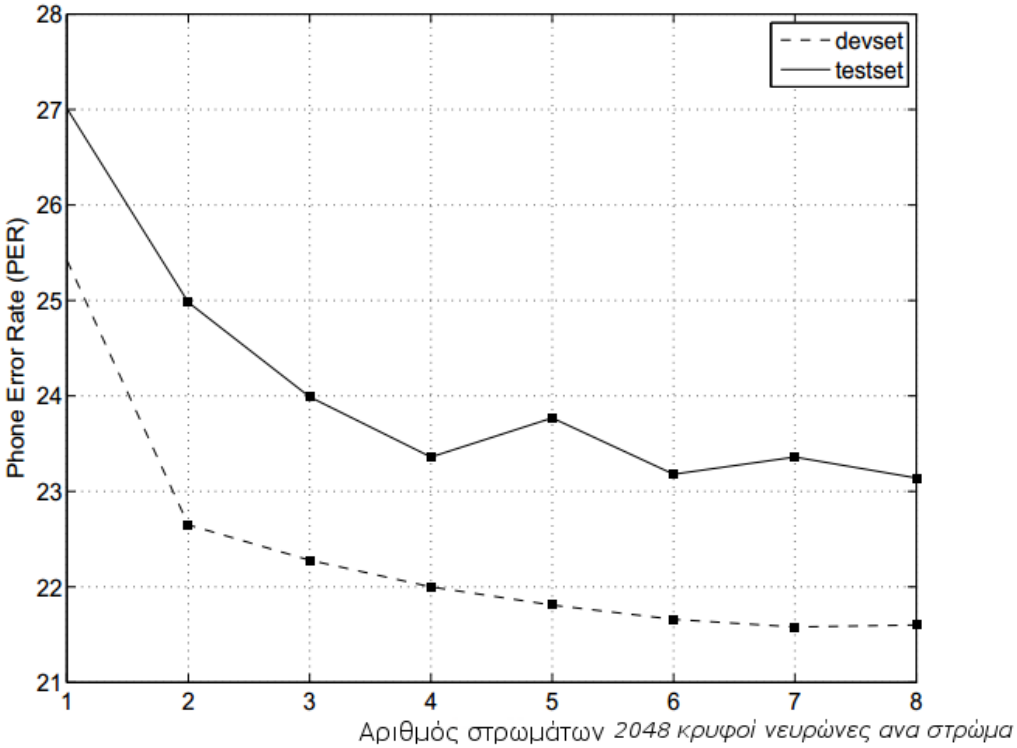
Εικόνα 1.45: BP-DBN [73]

Η αρχιτεκτονική ονομάζεται BP-DBN, αποτελείται από 2 στρώματα και με τα εξής ενδιαφέροντα χαρακτηριστικά:

- Για την αρχικοποίηση των βαρών χρησιμοποιείται άπληστη προ-εκπαίδευση ανά στρώμα με χρήση Contrastive Divergence.
- Εκτός των δύο κρυφών στρωμάτων προστέθηκε στο τέλος του δικτύου ένα τελευταίο στρώμα το οποίο έχει καθαρά λειτουργεία διακριτικής εξομάλυνσης με χρήση οπισθοδρομικής διάδοσης.
- Το ορατό στρώμα επίσης διαφοροποιείται καθώς χρησιμοποιεί διαφορετική κωδικοποίηση των κατηγοριών (class target) από την καθιερωμένη “ένα-εκ-των-Κ” κωδικοποίηση Softmax (one-of-K Softmax encoding [75]). Συγκεκριμένα χρησιμοποιήθηκε το γενικευμένο Softmax στρώμα (Generalized Softmax (GSM) output layer):
    - Γίνεται η υπόθεση ότι κάθε κατηγορία αναπαριστάτε από τον δικό της q-διάστατο διάνυσμα-κωδικό και άρα χρησιμοποιούνται q νευρώνες εξόδου στο μοντέλο. Τίθεται ως z το q-διάστατο διάνυσμα-στήλη το οποίοι παράγεται από το δίκτυο για δεδομένο κομμάτι ομιλάς (για συγκεκριμένη είσοδο). Το z πρέπει να μετασχηματιστεί σε έναν διάνυσμα «εκ των υστέρων» πιθανότητες κατηγορίας (vector of class posterior probabilities). Ένα θεωρηθεί C_j το διάνυσμα-στήλη το οποίο διατηρεί τον αντιπροσωπευτικό κωδικό της j-οστής κατηγορίας, τότε η έκφραση της πιθανότητας το μοντέλο να «δείξει» στην κατηγορία t δοθέντος διανύσματος z γίνεται:

$$P(t|\theta, z) = \frac{\exp(C_j z)}{\sum_j \exp(C_j z)}$$

Για τα πειράματα χρησιμοποιήθηκε το σετ δεδομένων TIMIT corpus me 462 ομάδες ομιλητών το οποίο όμως χρησιμοποιήθηκε επιλεκτικά και με κάποιες παραλλαγές [73]. Χρησιμοποιήθηκαν 183 ετικέτες κατηγοριών (3 καταστάσεις για καθ΄ μία από τα 63 phones). Ακολουθεί διάγραμμα που παρουσιάζει την επίδρασή του πλήθους των κρυφών στρωμάτων στην επιτυχία αναγνώρισης (όπου PER Phone Error Rate):



Διάγραμμα 1.15: Επίδραση πλήθους κρυφών στρωμάτων στην επίδοση αρχιτετονικής DBN [73]

Ο Πίνακας 1.10 δείχνει την επίδραση του πλήθους των νευρώνων ανά κρυφό στρώμα (χρήση BP-DBN τεσσάρων στρωμάτων) στην απόδοση του δικτύου.

Πίνακας 1.10: Επίδραση πλήθους νευρώνων ανά κρυφό στρώμα σε DBN [73]

Νευρώνες/Στρώμα	Devset PER	Testset PER
1024	21.94%	23.46%
2048	22.00%	23.36%
3072	21.74%	23.54%

Τέλος, στον Πίνακας 1.11 παρουσιάζει το πώς συγκρίνεται ένα BP-DBN πέντε στρωμάτων (2048 νευρώνες κάθε κρυφό στρώμα εκτός του τελευταίου που έχει 128 νευρώνες) χωρίς GSM με άλλα δίκτυα [73] στο σετ δεδομένων TIMIT core test ([76]):

Πίνακας 1.11: Σύγκριση DBN με άλλες αρχιτεκτονικές στο TIMIT core test [73]

Μέθοδος	PER
Stochastic Segmental Models	36%
Large-Margin GMM	33%
CD-HMM	27.3%
RNN	26.1%
Heterogeneous Classifiers	24.40%
Deep Belief Networks(DBNs)	**23.0%**

1.5.4 Υβριδικές

Η «υβριδικές» αρχιτεκτονικές [12] αναφέρεται σε βαθιές αρχιτεκτονικές που αποτελούνται από τόσο από παραγωγικά (μη-επιβλεπόμενης μάθησης) όσο από διακρίνοντα (επιβλεπόμενης μάθησης) μοντέλα. Άρα και από απόψεως κατηγοριοποίησης με γνώμονα τον τρόπο κατασκευής, οι αρχιτεκτονικές αυτές εμπίπτουν σε μια κατηγόρια υβριδικών αρχιτεκτονικών. Στις υπάρχουσες υβριδικές αρχιτεκτονικές το παραγωγικό τμήμα του δικτύου χρησιμοποιείται κυρίως για να βοηθήσει την διάκριση, η οποία άλλωστε είναι και ο ύστατος στόχος των υβριδικών αρχιτεκτονικών. Το πώς και το γιατί οι παραγωγικές τεχνικές βοηθούν στην διάκριση μπορεί να εξεταστεί από δύο σκοπιές:

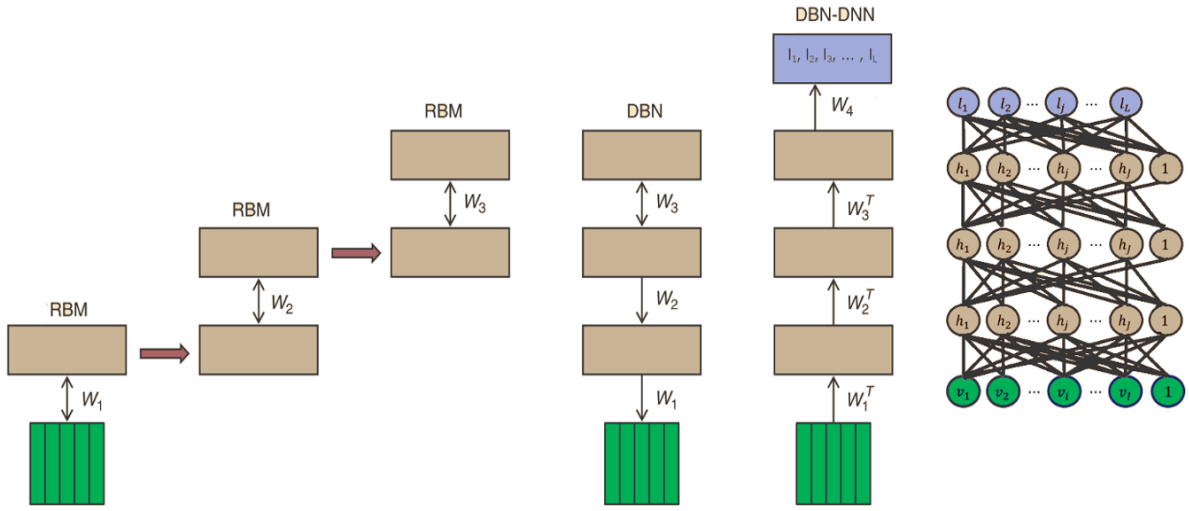
- Την σκοπιά της βελτιστοποίησης, από την οποία, παραγωγικά μοντέλα που εκπαιδεύονται με έναν μη-επιβλεπόμενο τρόπο μπορούν να παρέχουν εξαιρετική αρχικοποίηση στις παραμέτρων σε εξαιρετικά μη-γραμμικά προβλήματα εκτιμήσεων. Γι’ αυτό το λόγο άλλωστε η όλη διαδικασία αναφέρεται στην βιβλιογραφία ως προ-εκπαίδευσή.
- Την σκοπιά της κανονικοποίησης, από την οποία, τα μοντέλα μη-επιβλεπόμενης μάθησης μπορούν να παράσχουν ένα χρήσιμο προηγούμενο στις ομάδες των συναρτήσεων αντιπροσωπευτικών του μοντέλου. [77]

1.5.4.1 Deep Belief Network-Deep Neural Network (DBN-DNN)

Το DBN όπως προαναφέρθηκε είναι ένα παραγωγικό, βαθύ δίκτυο για μη-επιβλεπόμενη μάθηση το οποίο μπορεί να παραποιηθεί και να χρησιμοποιηθεί ως το αρχικό μοντέλο ενός βαθιού δίκτυού επιβλεπόμενης μάθησης, είτε με σκοπό την περεταίρω εκπαίδευση του είτε την τελειοποίηση του, χρησιμοποιώντας τις παρεχόμενες ετικέτες.

Προκειμένου να κατασκευαστεί το DBN-DNN ακολουθάτε η εξής διαδικασία [12].

- Αρχικά στοιβάζονται πολλαπλές περιορισμένες μηχανές Boltzmann σχηματίζοντας ένα DBN όπως περιεγράφηκε στο κεφάλαιο 1.5.3.2.1.
- Το DBN θα εκπαιδευτεί στην συνέχεια με τον άπληστο τρόπο που περιγράφηκε προηγούμενος (οι πιθανότητες ενεργοποίησης του κάθε RBM χρησιμοποιούνται ως είσοδος για το επόμενο) με αποτέλεσμα την επίτευξη εκμάθησης μέγιστης (κατά προσέγγιση) πιθανοφάνειας. Να σημειωθεί ότι εφόσον η εκπαίδευση είναι μη επιβλεπόμενη δεν απαιτούνται ετικέτες δεδομένων εισόδου.
- Το ήδη εκπαιδευμένο DBN μπορεί, έπειτα, να συνδυαστεί ή ακολουθηθεί από ένα μοντέλο επιβλεπόμενης εκμάθησης κατά την εκπαίδευση του οποίου θα τελειοποιηθούν τα βάρη βελτιώνοντας έτσι την απόδοση του δικτύου. Αυτή η διακριτή τελειοποίηση θα επιτευχθεί προσθέτοντας ένα τελευταίο στρώμα μεταβλητών στο δίκτυο οι οποίες αντιπροσωπεύουν τις επιθυμητές εξόδους (ετικέτες).
- Χάρη στο τελευταίο στάδιο καθίσταται δυνατή η χρήση του αλγορίθμου οπισθοδρομικής διάδοσης σφάλματος, με σκοπό με την προσαρμογή ή την τελειοποίηση των βαρών, ακριβώς όπως θα γινόταν σε ένα οποιοδήποτε έμπροσθεν τροφοδοτούμενο δίκτυο. Η επιλογή του ανώτατου αυτού στρώματος $\{l_1, l_2, l_3, \dots, l_j, \dots, l_L\}$ εξαρτάται από την εκάστοτε εφαρμογή π.χ. σε εφαρμογές αναγνώρισης ομιλίας το ανώτατο στρώμα μπορεί να αντιπροσωπεύει συλλαβές ή φωνήματα ή υπο-φωνήματα κ.ο.κ.



Εικόνα 1.46: Deep Belief Network-Deep Neural Network (DBN-DNN) [12] [79]

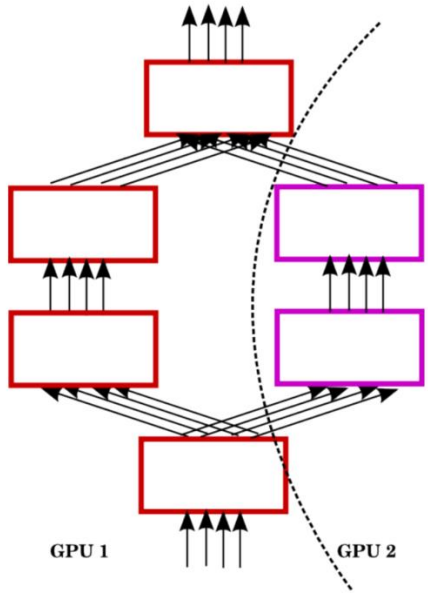
Διαδικασία αυτή έχει χρησιμοποιηθεί ενδελεχώς σε εφαρμογές αναγνώρισης ήχου και ομιλίας για παραγωγική (generative) προ-εκπαίδευση. Παρέχει μάλιστα πολύ καλύτερα αποτελέσματα απ' ότι θα παρείχε η τυχαία αρχικοποίηση των βαρών σε έναν μεγάλο εύρος εφαρμογών ανεξαρτήτου μεγέθους των δεδομένων προς εκπαίδευση.

2 Εργαλεία

Από την εγκαθίδρυση τους το 2006, η Βαθιά Μάθηση και τα βαθιά νευρωνικά δίκτυα έχουν επέλθει σε μια ταχέως αναπτυξιακή πορεία. Σε αυτό, έχει συμβάλει και η ανάπτυξη βιβλιοθηκών και πακέτων λογισμικού τα οποία έχουν κατασκευαστεί με γνώμονα την διευκόλυνση των ερευνητών αλλά και των μη εξειδικευμένων προγραμματιστών. Τα υπάρχοντα εργαλεία καλύπτουν ένα εύρη φάσμα λειτουργειών και έρχονται με υλοποιημένους αλγόριθμους μάθησης καθώς και ποικίλες άλλες διευκολύνσεις για την ανάπτυξη ρηχών ή βαθιών αρχιτεκτονικών.

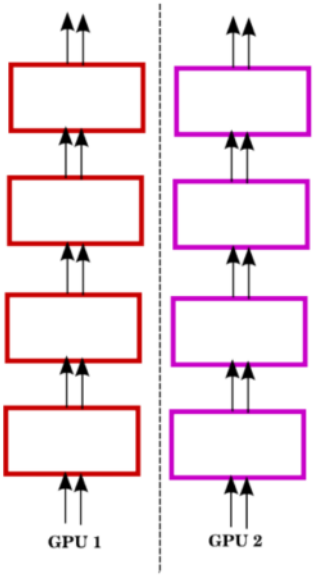
Τυπικά, μεγάλη έμφαση δίνεται στην ταχύτητα εκπαίδευσης δικτύων καθώς συχνά οι παράμετροι των δικτύου αριθμούν τις αρκετές χιλιάδες. Για να επιτευχθεί αυτό συχνά οι βιβλιοθήκες είναι υλοποιημένες (μερικώς ή πλήρως) σε γλώσσες χαμηλού επιπέδου όπως η C ενώ ταυτόχρονα συχνά παρέχουν διεπαφές με άλλες πιο εύχρηστες γλώσσες όπως Python. Πέρα από την ίδια την υλοποίηση βιβλιοθήκες συχνά προσφέρουν την δυνατότητα παράλληλης εκτέλεσης σε CPU ή και σε GPU εκμεταλλευμένες την τεράστια υπολογιστική ισχύ των λογικών αυτών μονάδων. Μερικές βιβλιοθήκες μάλιστα, υλοποιούν παραλληλία σε πολλαπλά επίπεδα στην GPU [78] προσφέροντας ακόμα μεγαλύτερη ταχύτητα τα οποία μάλιστα μπορούν να συνδυαστούν με εντυπωσιακά αποτελέσματα.(). Η παραλληλία επιτυγχάνεται:

- Σε επίπεδο μοντέλου: Κατά το οποίο πολλαπλές διεργασίες ομαδοποιούνται σε ρεπλίκες μοντέλου (model replica) προκειμένου να εκπαιδεύσουνε το ίδιο μοντέλο δηλαδή, ενώ χρησιμοποιούνται τα ίδια δεδομένα σε κάθε νήμα, το μοντέλο διαχωρίζεται. Συγκεκριμένα για τα νευρωνικά δίκτυα, αυτό σημαίνει ότι αρχικά, τα βάρη του δικτύου διαχωρίζονται ισομερώς ανάμεσα στα διαθέσιμα νήματα και στην συνέχεια όλα τα νήματα «δουλεύουν» παράλληλα στην επεξεργασία μιας κοινής υπο-παρτίδας δεδομένων. Αυτή η μέθοδος απαιτεί των συγχρονισμό των παραγόμενων εξόδων του κάθε στρώματος προκειμένου να παρέχεται εγκαίρως η είσοδος των επόμενων στρωμάτων. Για παράδειγμα ένα στρώμα συνέλιξης N τω πλήθος φίλτρων, μπορεί να τρέξει παράλληλα σε π.χ. 2 GPUs κάθε μια από τίς οποίες συνελίζει την είσοδο με χρήση $\frac{N}{2}$ φίλτρων.



Εικόνα 2.1: Παραλληλία σε επίπεδο μοντέλου [78]

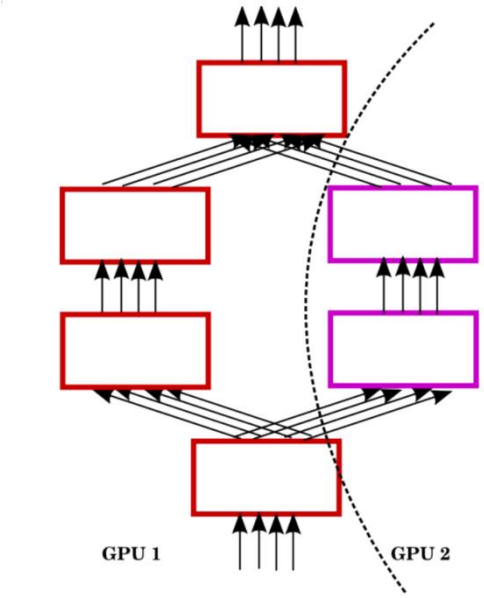
- Σε επίπεδο δεδομένων: Κατά το οποίο πολλαπλές ρεπλίκες μοντέλου επεξεργάζονται διαφορετικά τμήματα του συνόλου των δεδομένων. Δηλαδή το ίδιο μοντέλο χρησιμοποιείται σε κάθε νήμα καθ' ένα από τα οποία τροφοδοτούνται με διαφορετικά τμήματα των δεδομένων. Κάθε νήμα είναι υπεύθυνο για τον υπολογισμό των διανυσμάτων κλίσης αναφορικά σε όλες τις παραμέτρους του μοντέλου αλλά λαμβάνοντας υπ' όψων μόνο τμήματα της κατακερματισμένης υπο-παρτίδας (mini-batch) όπως φαίνεται και στο σχήμα στο οποίο κάθε νήμα (GPU) υπολογίζει τα σφάλματα και τα διανύσματα κλίσης για το $\frac{1}{2}$ των δειγμάτων της υπο-παρτίδας.



Εικόνα 2.2: Παραλληλία σε επίπεδο δεδομένων [78]

Αυτή η μέθοδος είναι η πιο απλή εκ' των δύο, αλλά παρ' όλα αυτά απαιτεί εκτενή επικοινωνία μεταξύ των νημάτων καθώς πρέπει να πρέπει να διαδίδονται τόσο οι παράμετροι όσο και τα διανύσματα κλίσης σε κάθε βήμα ανανέωσης βαρών. Επίσης η κάθε νήμα πρέπει να έχει στην διάθεση του μια ευμεγέθης υπο-παρτίδα δεδομένων προκειμένου να χρησιμοποιηθεί η παραλληλία στο επαρκών. Γι' αυτό τον λόγο το πλήθος των δειγμάτων μιας υπο-παρτίδας πολλαπλασιάζεται επί των αριθμό των διαθέσιμων νημάτων (GPUs).

Οι δύο αυτές τεχνικές είναι δυνατό να συνδυαστούν για ακόμα γρηγορότερα αποτελέσματα όπως φαίνεται και στο παρακάτω σχήμα, στο οποίο και το μοντέλο και τα δεδομένα διχοτομούνται και στην συνέχεια τα δύο δημιουργηθέντα νήματα «τρέχουν» παράλληλα σε δύο GPUs.



Εικόνα 2.3: Συνδυασμός επιπέδων παραλληλίας [78]

Τα υπάρχοντα πακέτα λογισμικού μπορούν να διαχωριστούν σε δύο κατηγορίες ανάλογα με την ιδεολογία και τον σκοπό ανάπτυξης τους [79].

- Των παραγωγικό-κεντρικών (productivity-oriented) εργαλείων που χρησιμοποιούνται για την ανάπτυξη νέων αλγοριθμικών τεχνικών (π.χ. Matlab/Octave) αλλά υστερούν στην δυνατότητα χειρισμού μεγάλου όγκου δεδομένων καθώς και στην εκμετάλλευση σύγχρονου hardware.
- Των εφαρμογό-κεντρικών (task-oriented) εργαλείων που προσανατολίζονται στην προγραμματιστική ευκολία και την ταχεία εκτέλεση εφαρμογών μεγάλου όγκου δεδομένων (π.χ. Caffe/Cuda-Convnet) αλλά η ευκολία αυτή έχει ως αποτέλεσμα την δυσκολία κατά την περάτωση ριζικών αλγοριθμικών αλλαγών.

Στην παρούσα εργασία θα αναφερθούν τα πιο εφαρμογό-κεντρικά εργαλεία τα οποία άλλωστε παρουσιάζουν και μεγαλύτερο ενδιαφέρων. Ακολουθεί ένας σύντομος πίνακας στον οποίο παρενθέτονται κάποια συγκριτικά χαρακτηριστικά για των βασικότερων πακέτων λογισμικού που θα αναφερθούν.

Πίνακας 2.1: Χαρακτηριστικά Βασικών Εργαλείων Λογισμικού για Deep Learning

Χαρακτηριστικά	Torch	Theano	Caffe	Dl4j	H_2o	Minerva
Άδεια κυκλοφορίας	BSD	BSD	BSD 2-Clause	Apache 2.0	Open Source	Apache 2.0
Γλώσσες	Lua	Python	C++, Bindings: Python/NumPy, Matlab	Java, Clojure, Scala	Java, Interfaces: R, Python, Scala, JSON, Coffescript, JavaScript	Bindings: C++, Python
OS	OS X, Windows, *nix	OS X, Windows, Linux	OS X, Windows (unofficial), Linux	Linux, OS X, Windows	Cross-Platform	*nix
Παραλληλία CPU	✓ (OpenMp)	✓ (Μόνο μέσω Blas και Conv2D)	✓	✓	✓ (Χρήση παράλληλου SGD)	✓ (Υλοποιημένα και τα δύο επίπεδα παραλληλίας)
Δυνατότητα χρήσης GPU	✓	✓	✓	✓	✓	✓
Παραλληλία GPU	✓ (CUDA)		✓ (CUDA)	✓ (CUDA)	× (Μελλοντικά εκτέλεση σε συστάδες GPU)	✓

* cuDNN is free for anyone to use for any purpose: academic, research or commercial.

2.1 Torch

Το Torch [80] είναι ένα επιστημονικό υπολογιστικό πλαίσιο με την ευρεία υποστήριξη αλγορίθμων μηχανικής μάθησης. Το torch είναι γραμμένο σε Lua, μια scripting γλώσσα προγραμματισμού σχεδιασμένη ώστε να είναι «ελαφριά», ούσα γραμμένη σε C, παραμένοντας παρόλα αυτά αρκετά ισχυρή. Ο βασικός γνώμονας επιλογής της Lua ήταν η ευκολία που προσφέρει στην υλοποίηση επιστημονικών αλγορίθμων, η επεκτασιμότητα της και ταχεία φύση της.

Ο στόχος της Torch είναι να προσφέρει μέγιστη ευελιξία και ταχύτητα στην οικοδόμηση επιστημονικών αλγόριθμων σας απλουστεύοντας την όλη διαδικασία . Υπάρχει εύρος

διαθέσιμων πακέτων με υλοποιήσεις, μεταξύ άλλων, μηχανικής μάθησης, computer vision, επεξεργασίας σήματος, παράλληλης επεξεργασίας, εικόνας, βίντεο και ήχου. Υπάρχουν υλοποιημένες βιβλιοθήκες με υλοποιήσεις αρκετών δημοφιλών νευρωνικών δικτύων και αλγορίθμων βελτιστοποίησης. Τέλος προσφέρει δυνατότητα παράλληλης επεξεργασίας σε CPU με χρήση OpenMP και SSE και σε GPU με χρήση CUDA. Τέλος είναι δυνατή η ενσωμάτωση τους σε λειτουργικά android FPGA και iOS.

Ακρογωνιαία λίθος του πλαισίου είναι η κλάση Tensor (τανυστής) παρεχόμενη από την αντίστοιχη βιβλιοθήκη της C που χρησιμοποιείτε εκτενώς για την αναπαράσταση δεδομένων. Τα Tensors είναι γεωμετρικά αντικείμενα που μπορούν να θεωρηθούν ως γενικευμένα διανύσματα.

Το Torch όντας ένα από τα πιο δημοφιλή frameworks υποστηρίζεται από μεγάλες εταιρίες όπως η Facebook. Αν και είναι δυνατή η χρήση του σε εφαρμογές μικρής κλίμακας συνίσταστε σε μεγάλες εφαρμογές μηχανικής μάθησης. Τέλος, φέρει άδεια BSD.

2.2 Theano theano

Το εν λόγο framework [81] είναι ένα project ανοιχτού λογισμικού σε άδεια BSD το οποίο αναπτύχθηκε κυρίως από την ομάδα μηχανικής μάθησης στο εργαστήριο LISA του πανεπιστημίου του Montreal. Πρόκειται για μια βιβλιοθήκη για Python που μπορεί να διαχειριστεί πολυδιάστατους πίνακες. Η ονομασία του προκύπτει από τον όνομα της αρχαίας Ελληνίδας μαθηματικού, Θεανούς που πιθανολογείται να είναι και σύζυγος του μεγάλου Πυθαγόρα.

Σύμφωνα με την ιστοσελίδα τους [82], το Theano επιτρέπει στον χρήστη να ορίσει, να βελτιστοποιήσει και να αξιολογήσει μαθηματικές εκφράσεις αποδοτικά. Πάντα σύμφωνα με την ιστοσελίδα μέσω του Theano δύναται η εκτέλεση εφαρμογών μηχανικής μάθησης μεγάλων δεδομένων με ταχύτητα συγκρίσιμη αντίστοιχων εφαρμογών γραμμένων σε C. Το Theano συνδυάζει τα χαρακτηριστικά ενός συστήματος αλγεβρικού υπολογιστή (computer algebra system CAS) με αυτά ενός μεταγλωττιστή βελτιστοποίησης. Αυτός ο συνδυασμός είναι εξαιρετικά χρήσιμος σε εφαρμογές κατά τις οποίες περίπλοκες μαθηματικές εκφράσεις υπολογίζονται επανειλημμένως και η ταχύτητα αποτελεί καίριο παράγοντα. Ακόμα όμως και σε περιπτώσεις στις οποίες, πολλαπλές μαθηματικές εκφράσεις, διαφορετικές μεταξύ τους, απαιτούνται να υπολογιστούν μόνο μια φορά η βιβλιοθήκη μπορεί να ελαχιστοποιήσει την υπολογιστική επιβάρυνση (computational overhead) παρέχοντας ταυτόχρονα χαρακτηριστικά όπως η αυτόματη διαφοροποίηση.

Το Theano παρέχει παραλληλία σε GPU καθώς μεταγλωττίζει υπολογισμούς μεγάλου όγκου δεδομένων σε CUDA ή OpenMP επιταχύνοντας τον υπολογισμό τους ως και 140 φορές. Αξίζει να σημειωθεί ότι δεν υποστηρίζονται παρά μόνο πραγματικοί αριθμοί 32bit (float32). Όσον αφορά την παραλληλία σε επίπεδο CPU αν και εφικτό έχει χαρακτηριστεί από χρήστες ως «εξαιρετικά δύσκολη».

Ακολουθεί μια λίστα μερικών εκ των διαφημιζόμενων χαρακτηριστικόν του πλαισίου:

- Εκτενής ενσωμάτωση με NumPy (χρήση *numpy.ndarray* για την υλοποίηση tensors στις μεταγλωττισμένες μεθόδους του Theano)
- Αποδοτική συμβολική διαφοροποίηση (επίλυση παραγώγων συναρτήσεων μοναδική ή πολλαπλής εισόδου)
- Βελτιστοποιήσεις ταχύτητας και σταθερότητας (π.χ. ορθός υπολογισμός του $log(1 + x)$ ακόμα και για πολύ μικρά x

- Μπορεί εύκολά να περιτυλίξει (wrap) κώδικα σε C ώστε να χρησιμοποιήσει Torch, cuda-convnet και άλλα. Αντιθέτως η περιτυλίξη της από κώδικα C παραμένει δύσκολη.
    - Αυτόματος εντοπισμός, ορισμένων, αριθμητικά ασταθών εκφράσεων και εν συνέχεια υπολογισμός τους με χρήση σταθερότερων αλγόριθμων

2.3 Caffe

Το Caffe είναι ένα Open Source framework με άδεια BSD η οποία ειδικεύεται σε εφαρμογές αναγνώρισης εικόνας ενώ έχει επεκταθεί (από κοινοτικές συνεισφορές) και σε εφαρμογές αναγνώρισης ομιλίας, ρομποτικής, νευροεπιστήμης και αστρονομίας. Υλοποιεί βιβλιοθήκη C++ ενώ παράλληλα παρέχει συνδέσεις (bindings) με Python/NumPy και MATLAB (σε άρθρο του Barkley γραμμένο εν έτι 2014 οι συνδέσεις αυτές χαρακτηρίστηκαν «Σχεδόν Ολοκληρωμένες»).

Το Caffe αρχικά δημιουργήθηκε από τον Yangqing Jia και στην συνέχεια αναπτύχθηκε και εξακολουθεί να συντηρείται από το κέντρο όρασης και μάθησης του Barkley (BVLC) με συνεισφορά ενεργούς κοινότητας στο GitHub (μετράει 2369 commits 3/7/2015).

Σύμφωνα με την επίσημη ιστοσελίδα το Caffe [83] και άλλες πηγές [84] χρίζει άξιο επιλογής χάρη των ακόλουθων χαρακτηριστικών του.

- Παρέχονται υλοποιήσεις των υποστηριζόμενων μοντέλων νευρωνικών δικτύων καθώς και υλοποιήσεις βελτιστοποίησης. Άρα το μόνο που απαιτείται από τον προγραμματιστή είναι η παραμετροποίηση τους.
- Εύκολη αλλαγή στην επιλογή μεταξύ εκτέλεσης σε CPU και GPU (με αλλαγή τιμής ενός flag).
- Υλοποιεί παράλληλη επεξεργασία δεδομένων σε GPU με χρήση CUDA.
- Ταχεία επεξεργασία δεδομένων. Υποστηρίζει ότι κατά την εκπαίδευση της CNN αρχιτεκτονικής supervision [43] με χρήση CUDA κατάφερε να επεξεργαστεί παραπάνω από 40 εκατομμύριά εικόνες την ημέρα χρησιμοποιώντας μια GPU τύπου Titan ή K40.
- Μεγάλο μέγεθός κοινότητας. Το Caffe χρησιμοποιείται για ακαδημαϊκούς αλλά και εμπορικούς σκοπούς σε εφαρμογές όρασης, ομιλίας και πολυμέσων.
- Το λογισμικό είναι από σχεδιασμού στοιχειοποιημένο (Modular) επιτρέποντας έτσι εύκολη επεκτασιμότητα με δυνατότητα προσθήκης νέων δομών δεδομένων, στρωμάτων νευρωνικών δικτύων και συναρτήσεων απώλειας (loss functions).
- Έρχεται με υλοποιήσεις προ-εκπαιδευμένων μοντέλων CNN και R-CNN για οπτικές εφαρμογές (μόνο για ακαδημαϊκή χρήση, μη-BSD άδεια χρήσης τους)
- Διαχωρισμός αναπαράστασης και υλοποίησης μοντέλων. Οι ορισμοί των μοντέλων νευρωνικών δικτύων έχουν γραφεί με την γλώσσα Protocol Buffer ως απλά αρχεία παραμετροποίησης (config files)

2.4 DL4J

Το Deeplearning4j [85] είναι μια open-source deep learning βιβλιοθήκη σε Java γραμμένη για το JVM και άρα λειτουργεί με Java, Clojure και Scala. Έχει αναπτυχθεί από

την Skymind (ιδιωτική εταιρία με βάση το San-Francisco) με σκοπό την χρήση σε εμπορικές εφαρμογές και όχι ως ερευνητικό εργαλείο. Το DL4J φέρει άδεια Apache 2.0 και άρα κάθε παραλλαγή του, ανήκει στους εφευρέτες της.

Η βιβλιοθήκη έχει πιθανές εφαρμογές στους ακόλουθους τομείς:

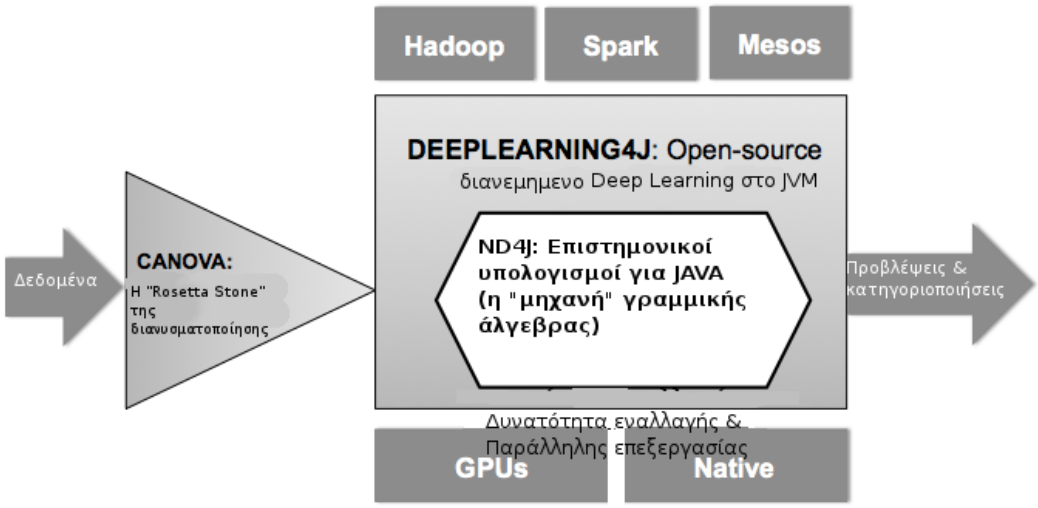
- Αναγνώρισης προσώπου/εικόνας
- Φωνητικής αναζήτησης
- Ομιλίας σε κείμενο (Speech to text)
- Φλιταρίσματος Ανεπιθύμητων Μηνυμάτων (Spam filtering (Anomaly Detection))
- Ανίχνευσης απάτης ηλεκτρονικού εμπορείου

Το DL4J επιτρέπει στον χρήστη να κατασκευάσει βαθιά νευρωνικά δίκτυα μέσω διαφόρων ρηχών αρχιτεκτονικών, κάθε μια από τις οποίες συνθέτει ένα στρώμα. Επιτρέπεται έτσι στον χρήστη να συνδυάσει με ευκολία διαφορετικές αρχιτεκτονικές σε μία. Τα νευρωνικά δίκτυα είναι τα ακόλουθα:

- RBM
- CNN
- Stacked Denoising Autoencoders
- LSTM
- Recursive Autoencoders
- DBN
- Deep Autoencoders
- RNTN

Τα βασικά χαρακτηριστικά που προσφέρει η βιβλιοθήκη είναι:

- Ενσωμάτωση με Hadoop, Spark και Mesos για διανεμημένη επεξεργασία δεδομένων
- Υποστήριξή παράλληλης εκτέλεσης σε GPU
- Υλοποίηση μίας ευέλικτης κλάσης N-διάστατου πίνακα.
- Ενσωμάτωση του εργαλείου διανυσματοποίησης Canova το οποίο καθιστά εύκολη την δημοσίευση (upload) δεδομένων (Σύμφωνα με τον Chris Nicholson προγραμματιστή και συνεισφερόμενο στην ανάπτυξη του DL4J)



Εικόνα 2.4: Διάγραμμα λειτουργείας DL4J [85]

2.5 H_2O

Το H_2O [86] είναι μια μαθηματική μηχανή για big data που υπολογίζει παράλληλους κατανεμημένους αλγορίθμους μηχανικής μάθησης όπως γραμμικά μοντέλα, random forests και νευρωνικά δίκτυα. Προσανατολίζεται σε εμπορικές εφαρμογές με μεγάλο όγκο δεδομένων. Ενδεικτικό της επιτυχίας τους είναι ο εκτενής κατάλογος των πελατών του που περιλαμβάνει μεγαθήρια όπως PayPal, Cisco και Nielsen. Το H_2O τρέχει σε Java και απαιτεί εγκατάσταση JDK έκδοσης μεγαλύτερης ή ίσης του 1.6.

Χρησιμοποιώντας τεχνικές συμπίεσης εντός μνήμης (in memory compression techniques) η πλατφόρμα μπορεί να διαχειριστεί εκατομμύρια γραμμές δεδομένων ακόμα και με χρήση μικρών συστάδων υπολογιστών. Το H_2O περιλαμβάνει διεπαφές για R, Python, Scala, Java, JSON και Coffescript/JavaScript καθώς και ένα ενσωματωμένο διαδικτυακό περιβάλλον. Η πλατφόρμα κατασκευάστηκε πάνω από συστάδες Hadoop και Spark.

Όσον αφορά την Βαθιά Μάθηση το H_20 παρέχει υλοποίηση ενός Autoencoder μη επιβλεπόμενης μάθησης καθώς και ενός έμπροσθεν τροφοδοτούμενου δικτύου επιβλεπόμενης μάθησης.

Το μοντέλο του έμπροσθεν τροφοδοτούμενου δικτύου εκπαιδεύεται με SGD (τροποποιημένου για καλύτερη παραλληλία) και οπισθοδρομική διάδοση σφάλματός. Επίσης προσφέρονται δυνατότητες κανονικοποίησης (dropout κ.λπ.) και βελτιστοποίησης (momentum). Το μοντέλο μπορεί να τροποποιηθεί ελαφρώς ως εξής:

- Νευρώνες
  - Σιγμοειδής ($tanh$)
  - RELUs
  - Maxout
- Συνάρτησες απώλειας
  - Μέσω τετραγωνικό λάθος
  - Cross Entropy

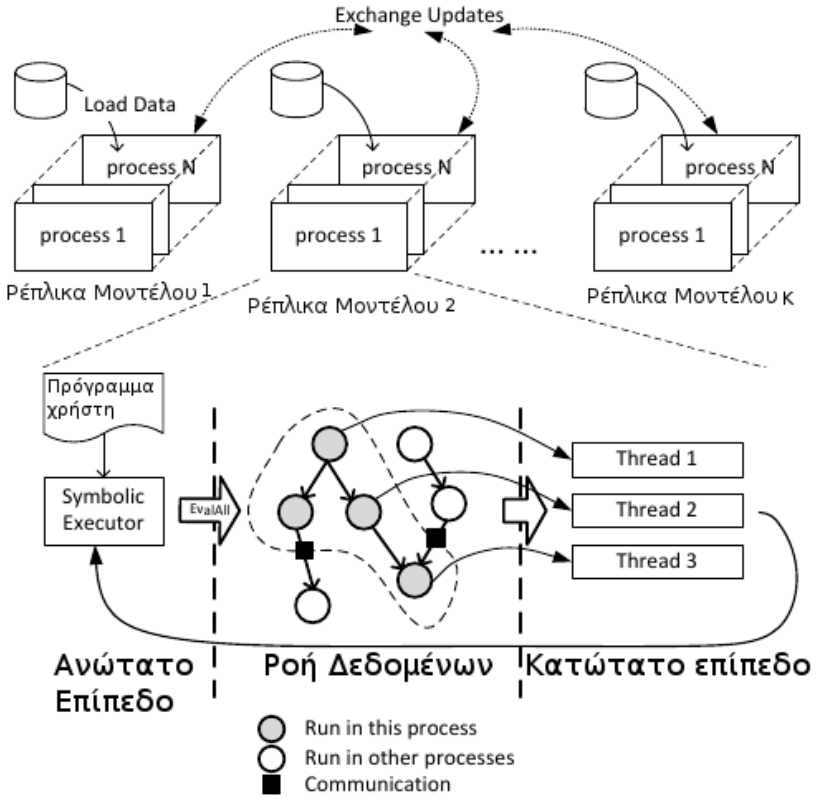
Το δίκτυο αυτό αφού εκπαιδευτικέ για 10 ώρες κατάφερε να ισοβαθμήσει την Microsoft η οποία μέχρι πρότινος κρατούσε μόνη τα ινία στα τεστ λάθους στο MNIST με 0,83% ποσοστό λάθους.

2.6 Minerva

Η Minerva [79] είναι μια αρκετά καινούρια βιβλιοθήκη αδείας Apache 2.0 με διασυνδέσεις για Python και C++ που αναπτύσσεται από το τμήμα έρευνας της Microsoft. Σύμφωνα με τους δημιουργούς της έρχεται να καλύψει το κενό μεταξύ των βασικών κατηγοριών εργαλείων ανάπτυξης μοντέλων deep Learning δηλαδή των παραγωγικό-κεντρικών (productivity-oriented) εργαλείων και των εφαρμογό-κεντρικών (task-oriented) εργαλείων.

Η Minerva λοιπόν αναπτύχθηκε σε μια στοιχειοποιημένη και πολύ-επίπεδη βιβλιοθήκη. Το υψηλότερο επίπεδο της βιβλιοθήκης εκφράζει deep learning αλγορίθμους χρησιμοποιώντας ένα API βασισμένο σε πίνακες το οποίο προσομοιάζει περιβάλλον Matlab. Στην συνέχεια η δυναμικά παραγόμενη ροή δεδομένων μπορεί να χαρτογραφηθεί σε διαφορετικό hardware. Έτσι ο ίδιος χρήστης μπορεί να τρέξει τον ίδιο κώδικα σε διαφορετικά μηχανήματα, με ή χωρίς επιτάχυνση GPU, επιτυγχάνοντας ταχύτητα και δυνατότητα κλιμάκωσης ισάξια ή και καλύτερη από τον ανταγωνισμό. Χαρακτηριστικά η υλοποίηση CNN της Minerva επιτυγχάνει 42,7% top-1 και 29,9% top-5 ρυθμούς λάθους στην κατηγορία ImageNet 1K, ταχύτητα διπλάσια από την βιβλιοθήκη ConvNet και παραπλήσια της βιβλιοθήκης Caffe (σε μία GPU).

Όσον αφορά την παραλληλία η βιβλιοθήκη την υλοποιεί και στα δύο επίπεδα παραλληλίας. Άρα το σύστημα διαχωρίζεται σε δύο λογικές μονάδες. Την άνω μονάδα που είναι υπεύθυνη για την παράλληλη επεξεργασία δεδομένων από ρέπλικες μοντέλου και την κάτω μονάδα που είναι υπεύθυνη για την εκμετάλλευση της παραλληλίας σε επίπεδο ρέπλικας μοντέλου.



Εικόνα 2.5: Διάγραμμα λειτουργία Minerva [79]

Ακολουθούν μερικά από τα διαφημιζόμενα χαρακτηριστικά της βιβλιοθήκες σύμφωνα με README.md στο Github

- Υποστήριξη προγραμματιστικού περιβάλλοντος N-D array
- Εύκολη ενσωμάτωση NumPy
- Αυτόματη παράλληλη εκτέλεση
- Υποστήριξη ταυτόχρονής επεξεργασίας σε πολλαπλές GPU καθώς και σε πολλαπλές CPU
- Συγχρονισμένη αξιολόγηση, δηλαδή το νήμα του χρήστη δεν μπλοκάρεται από τυχών εργασίες που εκτελούνται στο βάθος.

2.7 cuDNN

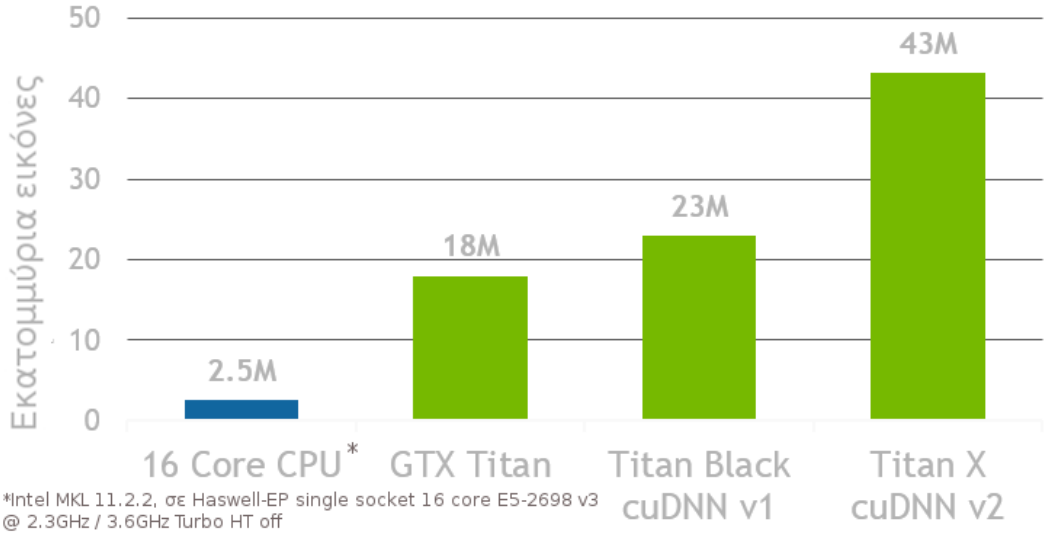
Το NVIDIA CUDA Deep Neural Network [87] ή εν συντομία cuDNN είναι μια βιβλιοθήκη επιταχυμένη μέσω GPU θεμελιωδών στοιχείων για βαθιά νευρωνικά δίκτυα. Δίνει έμφαση στην απόδοση, στην ευκολία χρήσης και στην ελαχιστοποίηση της επιβάρυνσης μνήμης (memory overhead). Η βιβλιοθήκη αυτή σχεδιάστηκες για να την ενσωμάτωσης της σε άλλα frameworks όπως για παράδειγμα τα Theano, Caffe και Torch (καθώς και άλλα). Ο απλουστευτικός σχεδιασμός της επιτρέπει στους προγραμματιστές να δίνουν έμφαση στον σχεδιασμό του μοντέλου τους και όχι τόσο σε θέματα επίδοσης. Η

cuDNN είναι διαθέσιμη δωρεάν σε προγραμματιστές εγγεγραμμένους στην ομάδα «CUDA Registered Developers».

Κάποια από τα κομβικά χαρακτηριστικά που προσφέρει είναι:

- Υποστήριξη πολλαπλών πλατφόρμων (Windows, Linux και Mac).
- Βελτιστοποίηση για κάρτες γραφικών NVIDIA® τελευταίας γενεάς.
- Υλοποιημένες μεθόδους μετασχηματισμών τανιστών (tensors).
- Υλοποιημένες ρουτίνες συνέλιξης σχεδιασμένες για CNN
- Εύκολή εκτέλεση σε πολλαπλά νήματα

Χαρακτηριστικό της επιτάχυνσης που προσφέρει, κατά την εκπαίδευση ενός δικτύου το cuDNN, είναι το παρακάτω διάγραμμα που αναπαριστά εκατομμύρια εικόνες ανά μέρα που εκπαιδεύτηκαν σε το μοντέλο CNN και συγκεκριμένα στο AlexNet υλοποιημένο σε Caffe.



Διάγραμμα 2.1: Υλοποίηση CNN σε Caffe με και χωρίς cuDNN [87]

2.8 PyLearn2

Το Pylearn2 [88] είναι μία open source βιβλιοθήκη μηχανικής μάθησης σε άδεια BSD 3-Clause η οποία είναι χτισμένη πάνω στην Theano. Η ανάπτυξη της ξεκίνησε το 2011 από τους David Warde-Farley, Pascal Lamblin, Ian Goodfellow (και άλλους) και αργότερα μεταφέρθηκε στο εργαστήρι LISA του Πανεπιστημίου του Montreal. Σήμερα (3/7/2015) το project στο GitHub μετράει 6987 commits από 115 contributors.

Ο στόχος της βιβλιοθήκης είναι η διευκόλυνση της ερευνάς μηχανικής μάθησης. Γι' αυτόν τον σκοπό δίνεται έμφαση στην ευκινησία και την επεκτασιμότητα της. Το γεγονός ότι είναι χτισμένη πάνω στην Theano της παρέχει δυνατότητα λειτουργείας τόσο σε CPU όσο και σε GPU.

Ακολουθούν μερικά από τα βασικά χαρακτηριστικά της:

- Χάριν της ερευνητικής φύσης της βιβλιοθήκης δεν τίθενται περιορισμοί στο πώς θα χρησιμοποιηθεί.
- Είναι χτισμένη από επαναχρησιμοποιήσιμα κομμάτια τα οποία μπορούν να χρησιμοποιηθούν σε ποικίλους συνδυασμούς. Για παράδειγμα αν ένας χρήστης θέλει να πειραματιστεί με την αρχιτεκτονική ενός μοντέλου χρησιμοποιώντας την κλάση Pylearn2 Model δεν είναι απαραίτητο να έχει μάθει να χρησιμοποιεί τις Pylearn υλοποιήσεις των αλγορίθμων εκπαίδευσης κ.λπ.
- Μέσω της παρεχόμενης YAML, (και ορισμένων επεκτάσεων) μίας domain-specific γλώσσας καθίσταται δυνατός ο προσδιορισμός όλων των υπέρ-παραμέτρων ενός πειράματος με ένα συμπαγή τρόπο. Ένα σύντομο αρχείο YAML μπορεί να δώσει υπόσταση σε είναι περίπλοκο πείραμα χωρίς έκθεση σε λεπτομέρειες της υλοποίησης του.

2.9 Cuda-convnet2

Το συγκεκριμένο εργαλείο πρόκειται για μια υλοποίηση CNN και γενικότερα έμπροσθεν τροφοδοτούμενων δικτύων σε C++/CUDA/Python που αναπτύχθηκε από τον Alex Krizhevsky. Μπορεί να μοντελοποιήσει αυθαίρετη διασύνδεση μεταξύ στρωμάτων και καθώς και αυθαίρετο βάθος δικτύου. Για την εκπαίδευση του δικτύου παρέχετε υλοποιημένος ο αλγόριθμος οπισθοδρομικής διάδοσης σφάλματος. Κυκλοφορεί σε άδεια Apache 2.0 [54].

Εφόσον είναι υλοποιημένη στο framework CUDA της NVIDIA ® προϋποθέτει την ύπαρξη κάρτας γραφικών Fermi (GTX 4xx, GTX 5xx, ή Tesla). Ως αποτέλεσμα ένα από τα βασικότερα χαρακτηριστικά που παρέχει είναι η ταχεία εκπαίδευση των δικτύων σε GPU της NVIDIA ®. Από την έκδοση 2.0 και μετά παρέχει δυνατότητα παράλληλης εκτέλεσης σε πολλαπλές GPU καθώς και ταχύτερη εκτέλεση σε GPU γενιάς Kepler.

Ακολουθούν ενδεικτικά παραδείγματα της ταχείας εκπαίδευσης που επιτυγχάνεται με χρήση Cuda-convnet2:

- CNN 9 στρωμάτων (2 υποδειγματοληψίας, 2 συνέλιξης 1 εξόδου και 4 άλλων (τους δόθηκε η ονομασία «τοπικά») config file [89]:
    - Ποσοστό σφάλματος 11% στο dataset CIFAR-10 μέσα σε 75 λεπτά (με μετατοπίσεις εικόνων και οριζόντιες ανακλάσεις)
    - Ποσοστό σφάλματος 13% στο dataset CIFAR-10 μέσα σε 25 λεπτά (με μετατοπίσεις εικόνων και οριζόντιες ανακλάσεις)
- LeNet5 [90]:
    - 15,4% ποσοστό σφάλματος

2.10 DistBelief

Το DistBelief είναι ένα framework που αναπτύχθηκε από την Google με σκοπό την εκπαίδευση μεγάλων μοντέλων. Το DistBelief μπορεί να χρησιμοποιήσει συστάδες υπολογιστών (computer clusters) για να επιταχύνει την εκπαίδευση τους. Το framework είναι αρκετά περιορισμένο στις εφαρμογές του και όντας closed source δεν υπάρχουν αρκετές πληροφορίες για αυτό πέρα από το σχετική εργασία του 2012 και περιοδικές αναφορές του σε άλλες εργασίες.

Στα πλαίσια της ανάπτυξης του δημιουργήθηκαν δύο νέοι αλγόριθμοι [91] για την κατανεμημένη εκπαίδευση μεγάλων διαστάσεων,

- Ο αλγόριθμος downpour Stochastic Gradient Descentα μίας ασύγχρονης παραλλαγή της στοχαστικής απότομης καθόδου που χρησιμοποιεί διαφορετικές ρέπλικες μοντέλου για να υπολογίσει με παράλληλο τρόπο τις κλίσεις συναρτήσεων διαφορετικών υπο-ομάδων των δεδομένων εκπαίδευσης)
- ο Sandblaster ένα framework που υποστηρίζει πληθώρα τεχνικών κατανεμημένης βελτιστοποίησης υπο-ομάδας (batch optimization procedures).

Με την χρήση αυτών των αλγόριθμων καθώς και την δυνατότητα παραλληλίας σε επίπεδο δεδομένων και σε επίπεδο μοντέλου (όπως το Minerva). Έχει χρησιμοποιηθεί επιτυχώς για την εκπαίδευση νευρωνικών δικτύων τεράστιων διατάσεων. Συγκεκριμένα εκπαιδεύτηκε δίκτυο τριών σταδίων (υποδειγματοληψία, φιλτράρισμα και κανονικοποίηση) 1,7 δισεκατομμυρίων παραμέτρων για αναγνώριση εικόνας. Η διαδικασία εκπαίδευσης επιταχύνθηκε 12Χ με την χρήση 81 μηχανημάτων ταυτόχρονα. Επετεύχθη παραπάνω από 15% ευστοχία πρόβλεψης κατά την κατηγοριοποίηση εικόνων του ImageNet 21000 κατηγοριών. Ο αριθμός είναι 60% καλύτερος από τα ως τότε γνωστά αποτελέσματα.

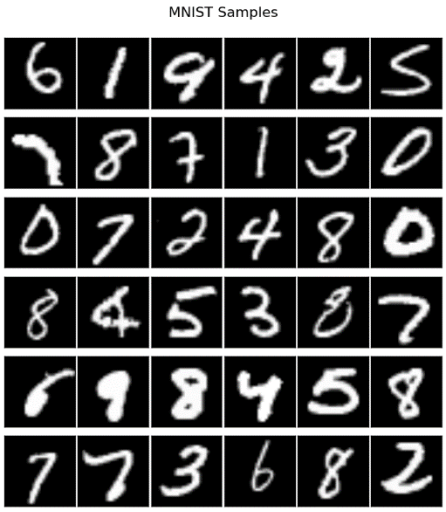
3 Πρακτικό Κομμάτι

3.1 Datasets

Όλα τα εργαλεία που αναπτυχθήκαν

3.1.1 MNIST

Στα πειράματα και των κώδικα χρησιμοποιήθηκε εκτενώς το σετ δεδομένων MNIST το οποίο αποτελείται από 60,000 δείγματα εκπαίδευσης, και από 10,000 δείγματα δοκιμής. Οι ασπρόμαυρες εικόνες του σετ δεδομένων, απεικονίζουν ψηφία στο διάστημα $[1-10]$. Το MNIST [94] είναι υπό-σύνολο μεγαλύτερων σετ δεδομένων διαθέσιμα στο NIST (National Institute of Standards and Technology). Οι εικόνες του MNIST έχουν κανονικοποιηθεί ως προς το μέγεθος ενώ κατά την προσαρμογή τους έχουν εγκεντριστεί σε εικόνες σταθερού μεγέθους $28 * 28$.



Εικόνα 3.1: Δείγμα 36 εικόνων του MNIST [92]:

Στην παρούσα υλοποίησης έχει χρησιμοποιηθεί η γλώσσα προγραμματισμού Python2.7. Όταν γίνεται χρήση της Python συνηθίζεται να μην χρησιμοποιείται το dataset στην αρχική μορφή του. Αντιθέτως τα δεδομένα σειριοποιούνται χρησιμοποιώντας την βιβλιοθήκη σειριοποίησης και αποσειριοποίησης Picle και συχνά, στην συνέχεια συμπιέζονται σε ένα αρχείο gzip. Το pickled αρχείο αντιπροσωπεύει μια πλειάδα αποτελούμενη από τρεις λίστες δεδομένων:

- Λίστα εκπαίδευσης; Αποτελούμενη από 50000 πλειάδες
- Λίστα επαλήθευσης: Αποτελούμενη από 10000 πλειάδες
- Λίστα εξέτασης: Αποτελούμενη από 10000 πλειάδες

Κάθε λίστα αποτελείται από έναν αριθμό δυσδιάστατών πλειάδων (x,y). Όπου:

- x: Λίστα Εικόνων: Με κάθε εικόνα αντιπροσωπεύεται από έναν μονοδιάστατο πίνακα δυαδικών (0 για μαύρο, 1 για άσπρο) αριθμών κινητής υποδιαστολής (float) της βιβλιοθήκης numpy 784 θέσεων $(28 * 28)$ (_numpy.ndarry_).

- y: Λίστα Ετικετών: Με κάθε ετικέτα να είναι ακαριαίος αριθμός στο διάστημα $[0,9]$ και η τιμή του να υποδεικνύει πιο εκ των 10 δεκαδικών ψηφίων απεικονίζει μια εικόνα (_numpy.ndarry_).

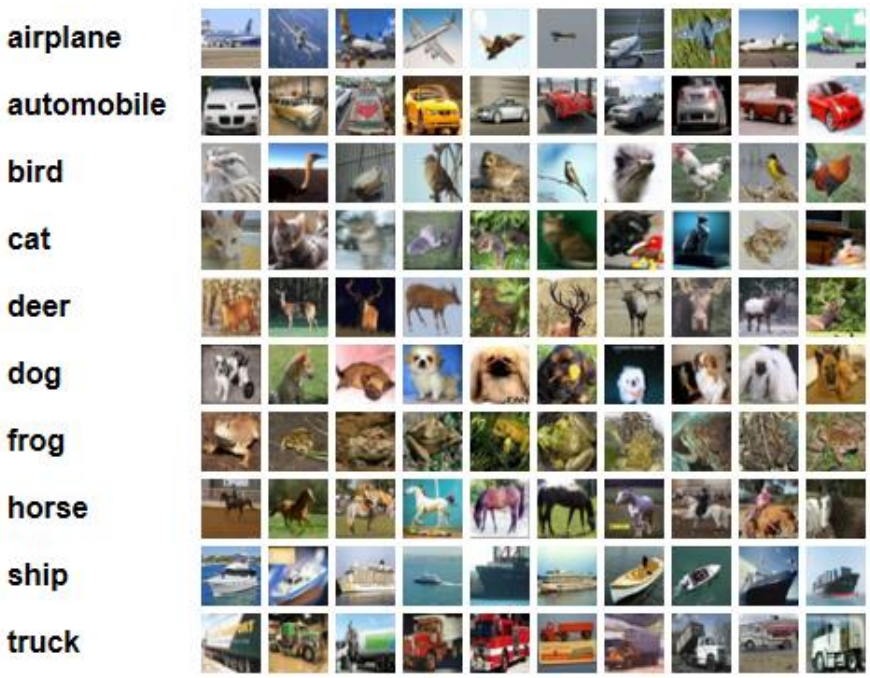
Προκειμένου να φορτωθούν τα δεδομένα σε πρόγραμμα Python χρησιμοποιείται η παρακάτω συνάρτηση:

```
def load_data():

    f = gzip.open('../../../data/mnist.pkl.gz', 'rb')

    training_data, validation_data, test_data = cPickle.load(f)

    f.close()

   return (training_data, validation_data, test_data)
```

3.1.2 Cifar-10

Στα πειράματα χρησιμοποιήθηκε και το σετ δεδομένων Cifar-10 το οποίο αποτελείται από 60000 χρωματιστές εικόνες μεγέθους 32x32 οι οποίες κατανέμονται ισομερώς σε 10 κατηγορίες. Το σετ δεδομένων διαιρείται σε 50000 εικόνες εκπαίδευσης και 10000 εικόνες επαλήθευσης. There are 50000 training images and 10000 test images.

Το σετ δεδομένων διαρείται σε πέντε υποομάδες εκπαίδευσης και μία υποομάδα εξέτασης με την κάθε υποομάδα να αριθμεί 10000 δείγματα. Η υποομάδα εξέτασης (test batch) περιλαμβάνει 1000 εικόνες από κάθε κατηγορία ενό ανηθέτώς οι εικόνες τών υποομάδων εκπαίδευσης ειναι τυχαία επελεγμένες και μπορεί να περιλαμβάνουν περισότερρα δείγματα από μια κατηγορία παρά από μια άλλη. Στην εικόνα απεικονείζονται δείγματα από κάθε κατηγορία.



Εικόνα 3.2: Δείγμα 100 εικόνων του Cifar-10 [100]

4 Συμπεράσματα

Όλα τα πειράματα σε αυτό το κεφάλαιο έχουν περατωθεί σε υπολογιστή Linux Fedora22, με επεξεργαστή Intel i7-5500U χρονισμένο στα 2,4Ghz.

4.1 Simple_NN_MNIST

Τα πειράματα αυτά χρησιμοποιούν το αρχείο RunNetwork.py το οποίο με την σειρά του καλεί το network.py και το mnist_loader.py. Εφόσον χρησιμοποιείται το σετ δεδομένων MNIST τα δίκτυα που εκπαιδευτήκαν έχουν απαραιτήτως 784 νευρώνες εισόδου, δηλαδή έναν για κάθε pixel μιας εικόνας ψηφίου ($28 * 28 = 784$) και 10 νευρώνες εξόδου, δηλαδή έναν για κάθε ετικέτα $t \in [0,9]$.

4.1.1 Έμπροσθεν Τροφοδοτούμενα Δίκτυα διαφόρων βαθών.

Η πειραματική διαδικασία στην παρούσα φάση έχει ως σκοπό να αναδείξει την δυσκολία που αντιμετωπίζεται κατά την εκπαίδευση βαθιών αρχιτεκτονικών καθώς και το χρονοβόρο της διαδικασίας. Όλα τα δίκτυα σε αυτό το κεφάλαιο έχουν εκπαιδευτεί με χρησιμοποιώντας την ίδια διαδικασία καθώς και με χρήση κοινών υπερ-παραμέτρων. Η εκπαιδευτική διαδικασία διαρκεί για 30 εποχές και έχει ως εξής:

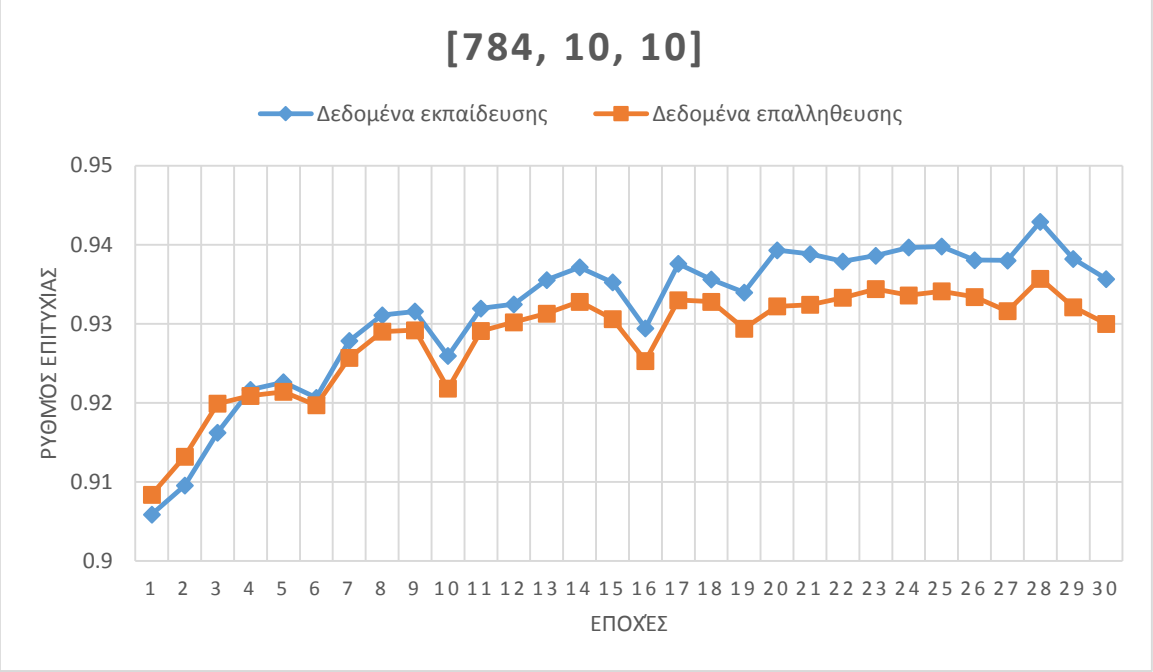
- Αλγόριθμος εκμάθησης: ΟΔΣ με χρήση στοχαστικής απότομής καθόδου
- Αλγόριθμος βελτιστοποίησης: Στοχαστική Απότομη Κάθοδος (SGD) με μέγεθος υπό-ομάδας 20 δείγματα και ρυθμός μάθησης 0,25
- Συνάρτηση Κόστους: Η συνάρτηση δια-εντροπίας (cross-entropy)

$$L_{\mathcal{H}}(x,z) = -\left(\sum_{k=1}^{d}[x_k \log z_k + (1-x_k)\log(1-z_k)]\right)$$

- Αρχικοποίηση βαρών: Default: Αρχικοποίηση όλων των βαρών και σταθερών πόλωσης τυχαία, από μία κανονική κατανομή με μέση τιμή $\mu = 0$ και διακύμανση $\sigma^2 = 1$. Στην συνέχεια, μόνο για τα βάρη, η τυχαία αυτή τιμή τους διαιρείται με την τιμή $\sqrt{x}$ με x να είναι η το πλήθος των συνδέσεων εισόδου του νευρώνα.
- Νευρώνες: Απλοί Σιγμοειδής
- Ομαλοποίηση: L2 με παράμετρο ομαλοποίησης $\lambda = 5$

Να σημειωθεί ότι ο χρόνος εκπαίδευσης που έχει μετρηθεί αναφέρεται μόνο στην χρονομέτρηση των απαραίτητων υπολογισμών για την εκπαίδευση. Στην πράξη ο χρόνος που απαιτήθηκε για την ολοκλήρωση του εκάστοτε πειράματος ήταν μεγαλύτερος καθώς απαιτήθηκαν περαιτέρω υπολογισμοί για την συλλογή στατιστικών στοιχείων (Κόστος ανά εποχή στα δεδομένα εκπαίδευσης και επαλήθευσης καθώς και ρυθμό επιτυχίας στα δεδομένα εκπαίδευσης και επαλήθευσης ανά εποχή).

Το πρώτο δίκτυο που εκπαιδεύτηκε είναι μια ρηχή αρχιτεκτονική με 10 κρυφούς νευρώνες. Άρα το δίκτυο έχει την μορφή $[784, 10, 10]$. Στο παρακάτω διάγραμμα (Διάγραμμα 4.1) παρουσιάζεται η ακρίβεια πρόβλεψης (ρυθμός επιτυχίας) στα δεδομένα εκπαίδευσης και στα δεδομένα επαλήθευσης όπως αυτά περιεγράφηκαν στο κεφάλαιο 3.1.1.

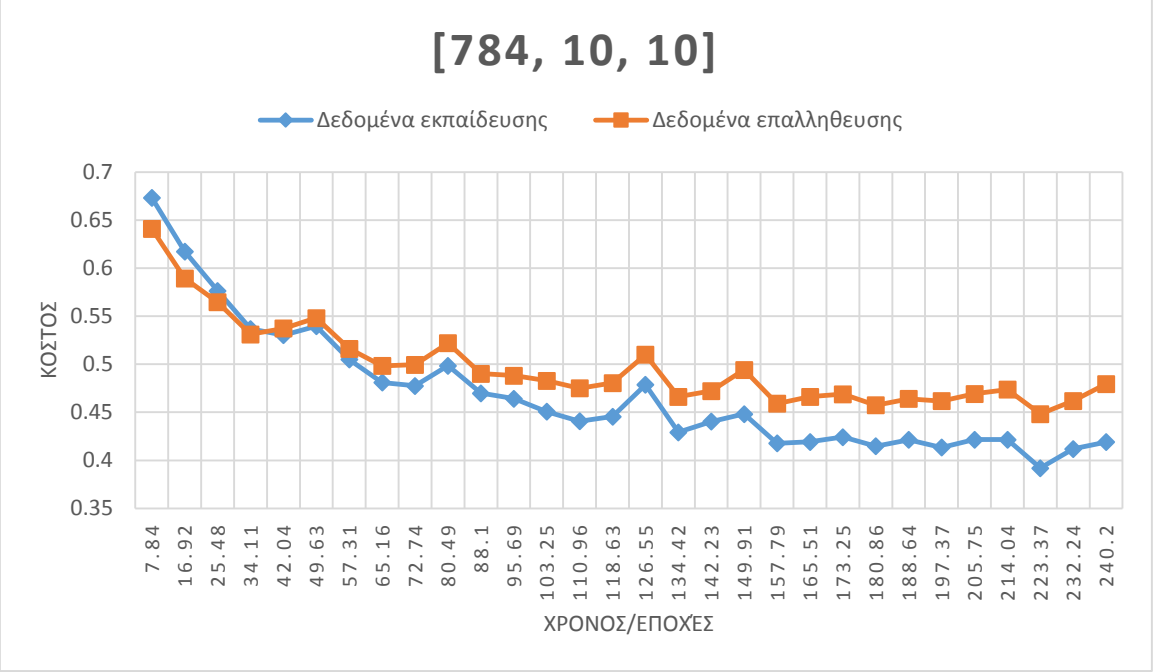


Διάγραμμα 4.1: Εξέλιξη ρυθμών επιτυχίας σε [784, 10, 10] FFN

Στο Διάγραμμα 4.1 παρατηρείται ότι ο ρυθμός επιτυχίας είναι αρκετά υψηλός από την πρώτη ακόμα εποχή. Αυτό είναι ισχυρή ένδειξη ότι ο αλγόριθμος αρχικοποίησης βαρών που επιλέχτηκε ήταν μια καλή επιλογή. Το διάγραμμα παρουσιάζει έντονες διακυμάνσεις, οι οποίες δεν εμφανίζονται τόσο έντονα στα μεγαλύτερα δίκτυα που εκπαιδευτήκαν. Οι διακυμάνσεις αυτές αναδεικνύουν την αδυναμία των πολύ μικρών-ρηχών αρχιτεκτονικών καθώς μια μικρή αλλαγή σε έναν νευρώνα μπορεί να επηρεάσει εντόνως το αποτέλεσμα.

- Ο μέγιστος ρυθμός επιτυχίας που επιτεύχθηκε στα δεδομένα εκπαίδευσης είναι 0.9429 ενώ η καταληκτική τιμή του είναι 0,93566.
- Στο διάγραμμα, ο ρυθμός επιτυχίας στα δεδομένα επαλήθευσης ακολουθάει κατά πόδας τον ρυθμό επιτυχίας των δεδομένων εκπαίδευσης και άρα δεν τίθεται θέμα υπερ-προσαρμογής. Η διαφορά έγκειται στο γεγονός ότι ο ρυθμός αυτός είναι σαφώς μικρότερος του ρυθμού επιτυχίας στα δεδομένα εκπαίδευσης, όπως άλλωστε ήταν αναμενόμενο (το δίκτυο δεν έχει εκπαιδευτεί σε αυτά). Παρόλα αυτά επιτυγχάνεται μέγιστος ρυθμός 0,9357 και η καταληχτική τιμή του είναι 0,93.

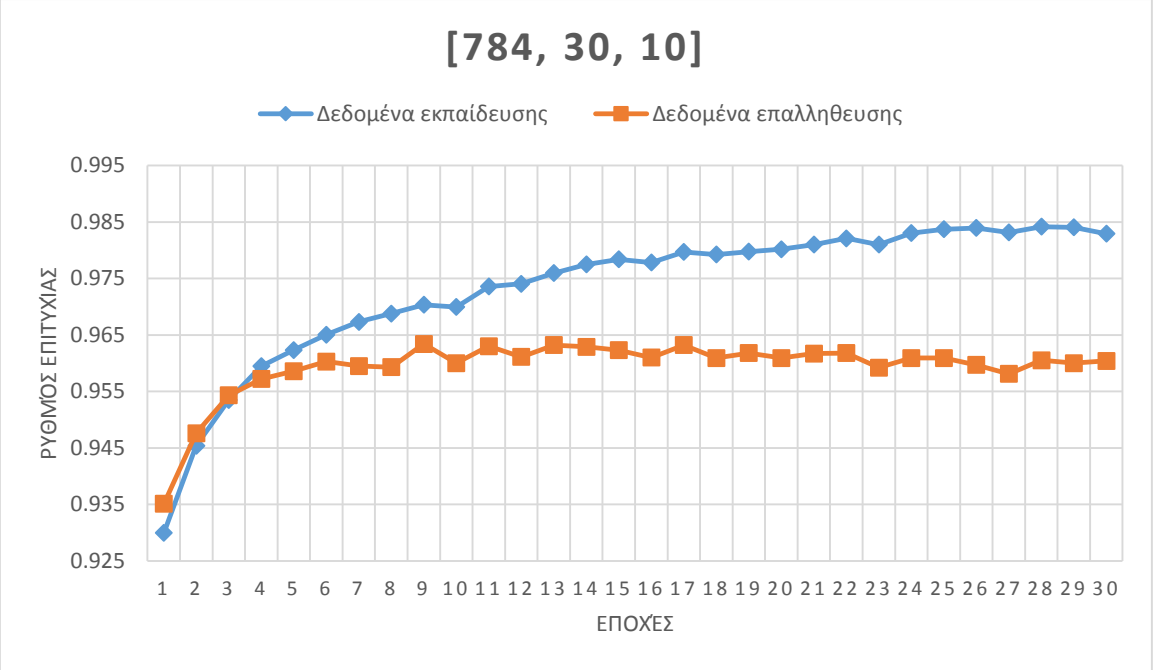
Το παρακάτω διάγραμμα (Διάγραμμα 4.2) παρουσιάζει δύο στοιχεία. Ο άξονας x παρουσιάζει το χρόνο (CPU time σε δευτερόλεπτα) που διήρκησε η κάθε εποχή ενώ ταυτόχρονο παρουσιάζεται η εξέλιξή του κόστους (το αποτέλεσμα της συνάρτησης κόστους) στον χρόνο εκπαίδευσης.



Διάγραμμα 4.2: Εξέλιξη κόστους / χρονική διάρκεια εποχών σε [784, 10, 10] FFN

Στο Διάγραμμα 4.2 οι τιμές του κόστους που εμφανίζονται είναι σε πλήρη αρμονία με τους αντίστοιχους ρυθμούς επιτυχίας στις δύο ομάδες δεδομένων. Το δίκτυο εκπαιδευτικέ συνολικά για 240.2 δευτερόλεπτα (4 λεπτά) με μέσο χρόνο ανά εποχή 8.01 δευτερόλεπτα.

Στην συνέχεια εκπαιδευτικέ άλλη μια ρηχή αρχιτεκτονική με 30 κρυφούς νευρώνες. Άρα το δίκτυο έχει την μορφή $[784, 30, 10]$. Στο παρακάτω διάγραμμα (Διάγραμμα 4.3) παρουσιάζεται η ακρίβεια στα δεδομένα εκπαίδευσης και στα δεδομένα επαλήθευσης.

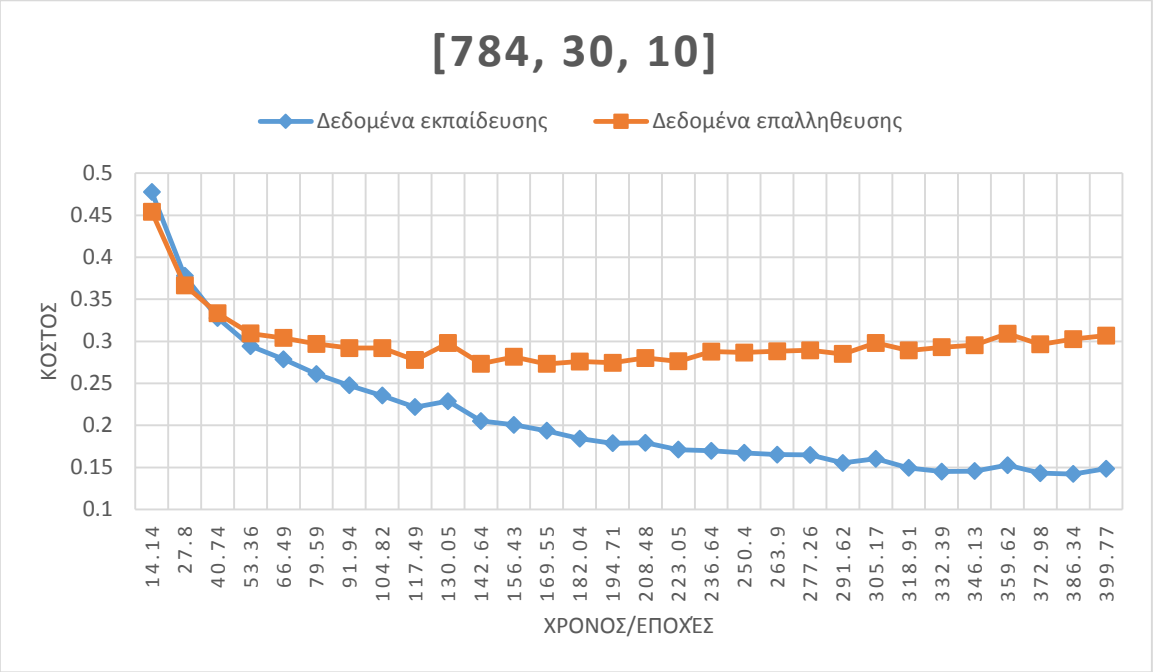


Διάγραμμα 4.3: Εξέλιξη ρυθμών επιτυχίας σε [784, 30, 10] FFN

Στο Διάγραμμα 4.3 φαίνεται ξεκάθαρα ότι επετεύχθη καλύτερος ρυθμός επιτυχίας τόσο στα δεδομένα εκπαίδευσης όσο και στα δεδομένα επαλήθευσής καθώς το δίκτυο όντας μεγαλύτερο κατά 20 νευρώνες κατάφερε να χαρτογραφήσει καλύτερα τα δεδομένα εισόδου.

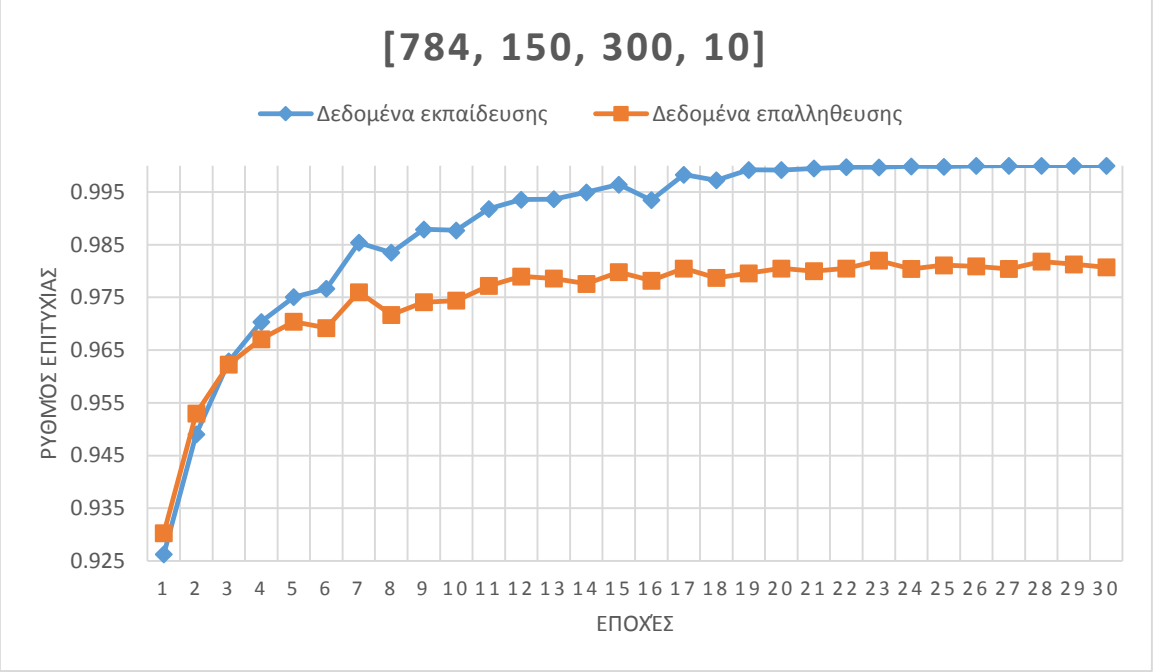
- Ο μέγιστος ρυθμός επιτυχίας που επιτεύχθηκε στα δεδομένα εκπαίδευσης είναι 0,98418 ενώ η καταληκτική τιμή του είναι 0,98212.
- Ο μέγιστος ρυθμός επιτυχίας στα δεδομένα επαλήθευσης είναι 9,634 ενό η καταληκτική τιμή είναι 0,9604

Το παρακάτω διάγραμμα (Διάγραμμα 4.4) παρουσιάζει δύο στοιχεία. Ο άξονας x παρουσιάζει το χρόνο που διήρκησε η κάθε εποχή ενώ ταυτόχρονο παρουσιάζεται η εξέλιξή του κόστους στον χρόνο εκπαίδευσης.



Διάγραμμα 4.4:Εξέλιξη κόστους / χρονική διάρκεια εποχών σε [784, 30, 10] FFN

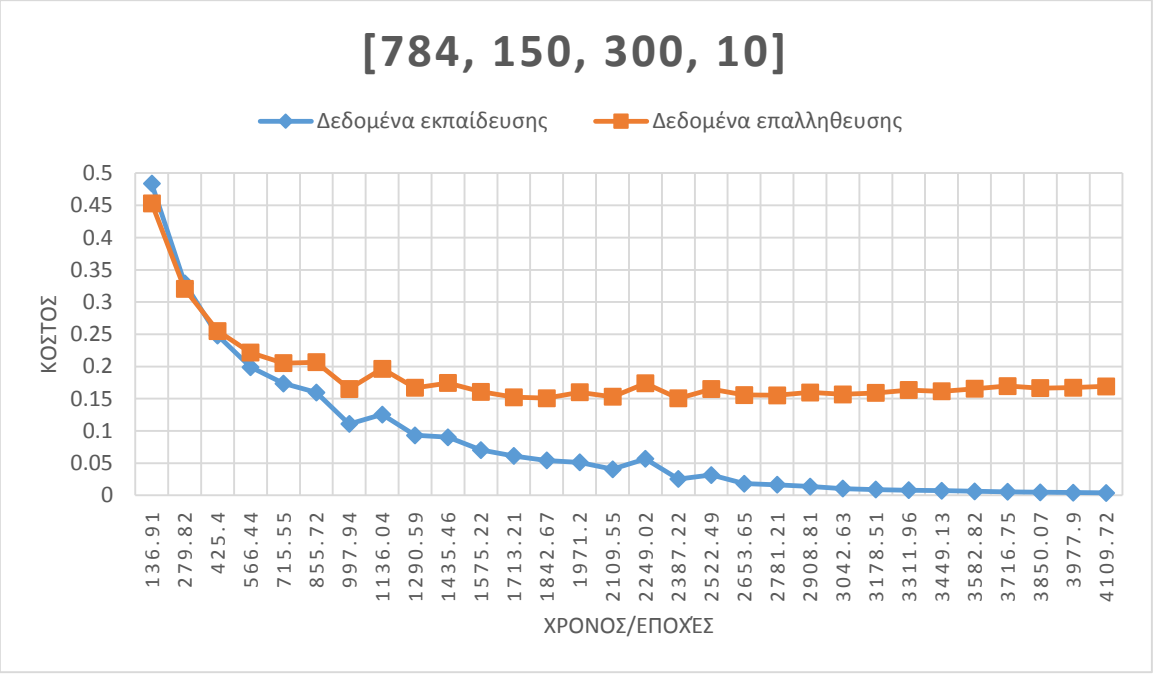
Στο Διάγραμμα 4.4: τα καταληκτικά κόστη των δεδομένων εκπαίδευσης και επαλήθευσης είναι 0,1484 και 0,3067 αντίστοιχα. Το δίκτυο εκπαιδευτικέ συνολικά για 399.77 δευτερόλεπτα (≈ 6,5 λεπτά) με μέσο χρόνο ανά εποχή 13.3 δευτερόλεπτα.
Η επόμενη αρχιτεκτονική που εκπαιδεύτηκε ήταν βαθιά, με δύο βαθιά στρώματα, το πρώτο με 150 νευρώνες και το δεύτερο με 300 νευρώνες. Ακολουθούν διαγράμματα (Διάγραμμα 4.5, Διάγραμμα 4.6) στην ίδια λογική με τα αντίστοιχα, προηγούμενα του κεφαλαίου.



Διάγραμμα 4.5: Εξέλιξη ρυθμών επιτυχίας σε [784, 150, 300, 10] FFN

Στο Διάγραμμα 4.5 διαφαίνονται αμέσως τα προτερήματα των βαθιών αρχιτεκτονικών. Η εξέλιξη των ρυθμών επιτυχίας είναι πιο ομαλή και οι τιμές τους υψηλότερες από τις αντίστοιχές των προηγούμενων ρηχών αρχιτεκτονικών.

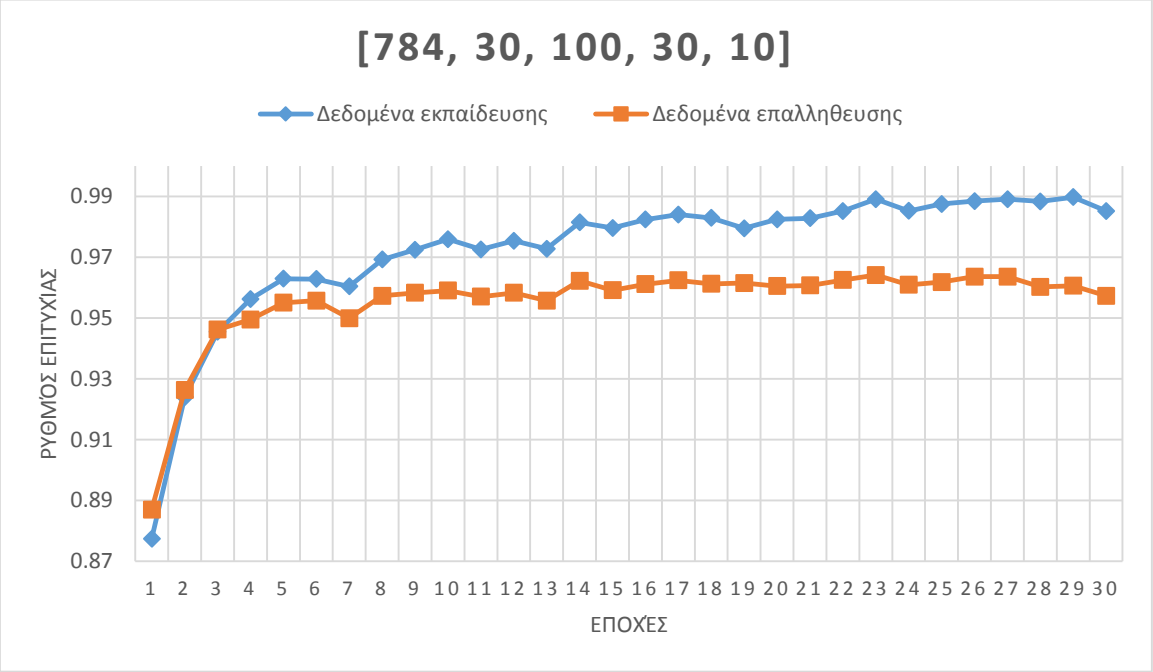
- Ο μέγιστος και καταληκτικός ρυθμός επιτυχίας που επιτεύχθηκε στα δεδομένα εκπαίδευσης είναι 1 (100% επιτυχία) σε αντίθεση με το 0,98418 μέγιστο ρυθμό των προηγούμενων δικτύων.
- Ο μέγιστος ρυθμός επιτυχίας στα δεδομένα επαλήθευσης είναι 9,82 ενώ η καταληκτική τιμή είναι 0,9807 (0, 98418 μέγιστο στα προηγούμενα ρηχά δίκτυα)



Διάγραμμα 4.6: Εξέλιξη κόστους / χρονική διάρκεια εποχών σε [784, 150, 300, 10] FFN

Στο Διάγραμμα 4.6Διάγραμμα 4.4: τα καταληκτικά κόστη των δεδομένων εκπαίδευσης και επαλήθευσης είναι 0, 004 και 0,1484 αντίστοιχα. Το δίκτυο εκπαιδευτικέ συνολικά για 4109,72 δευτερόλεπτα (68.48 λεπτά), τιμή δεκαπλάσια του δικτύου [748, 30, 10] (399.77 δευτερόλεπτα) καθώς η αρχιτεκτονική που επιλέχτηκε απαιτεί την εκπαίδευση πολύ περισσότερων παραμέτρων. Ο μέσος χρόνος ανά εποχή είναι 136.99 δευτερόλεπτα.

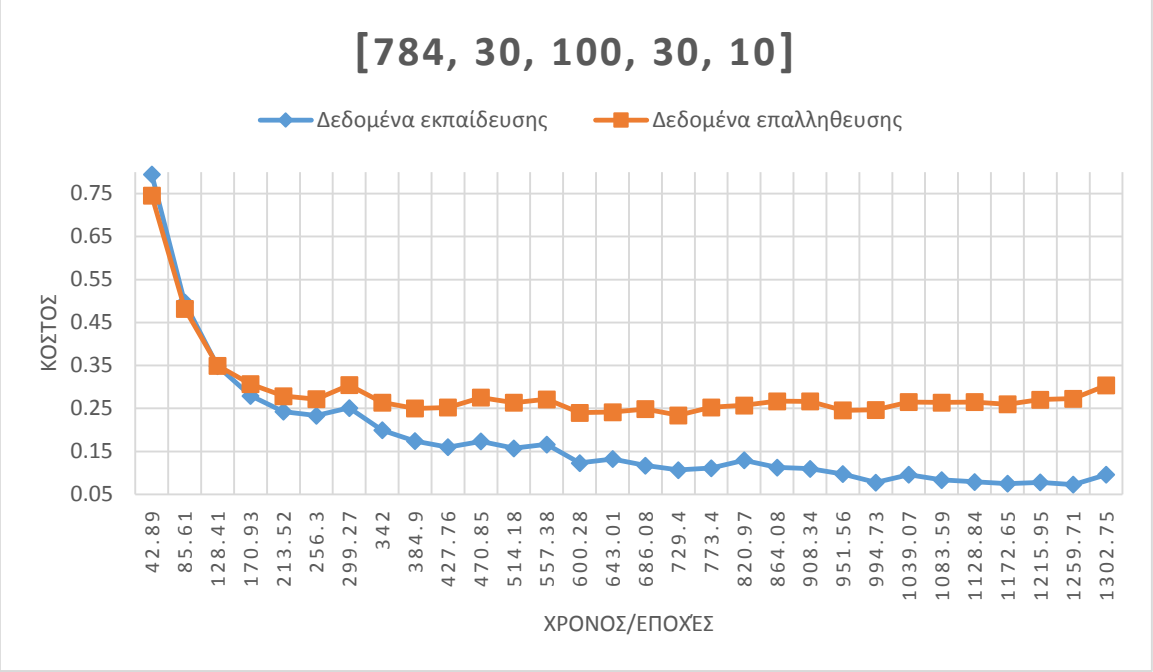
Έπειτα επιλέχτηκε άλλη μια βαθιά αρχιτεκτονική η οποίο όμως έχει συνολικά λιγότερους νευρώνες από την προηγούμενη. Η αρχιτεκτονική αυτή είχε 3 στρώματα κρυφών νευρώνων το πρώτο με 30 κρυφούς νευρώνες το δεύτερο με 100 και το τρίτο με 30. Άρα η αρχιτεκτονική έχει την μορφή $[784, 30,100,30,10]$. Ακολουθούν διαγράμματα στην ίδια λογική με τα αντίστοιχα, προηγούμενα του κεφαλαίου.



Διάγραμμα 4.7: Εξέλιξη ρυθμών επιτυχίας σε [784, 30, 100, 30, 10] FFN

- Ρυθμός επιτυχίας στα δεδομένα εκπαίδευσης: Μέγιστός 0.9897 , ρυθμός την τριακοστή εποχή, 0,9854 (1 και 1 στο προηγούμενο [784, 150, 300, 10])
- Ρυθμός επιτυχίας στα δεδομένα επαλήθευσης: Μέγιστός 0.964, ρυθμός την τριακοστή εποχή, 0,9573 (9,82 και 0,9807 στο προηγούμενο)

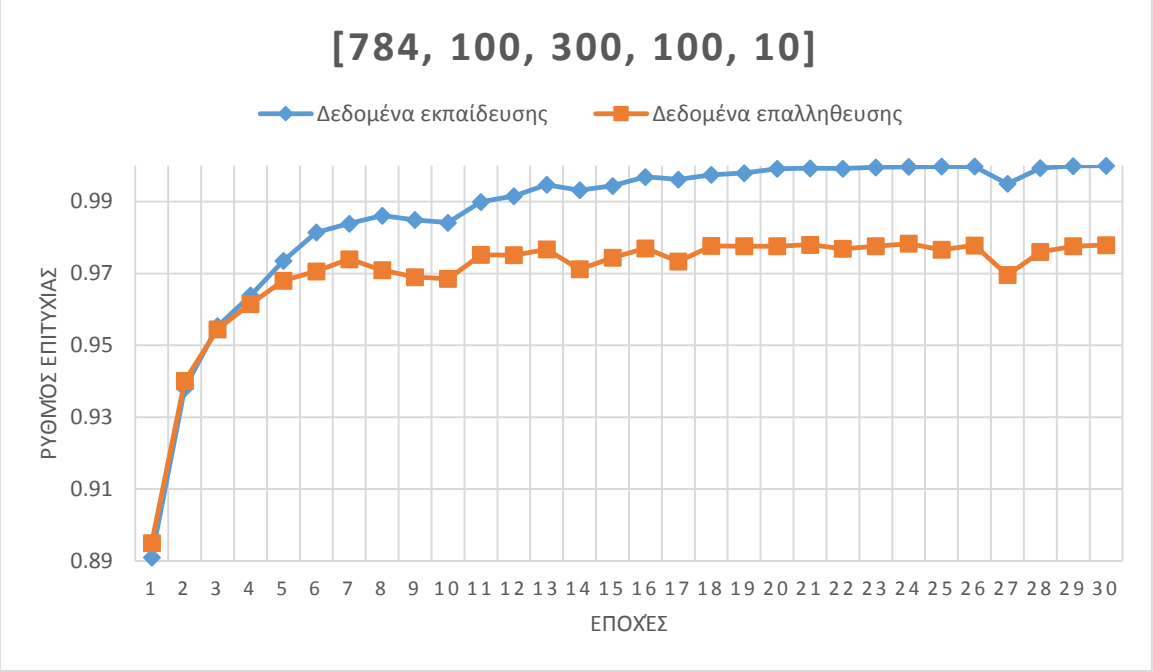
Οι τιμές αυτές αναδεικνύουν το γεγονός ότι μεγαλύτερο βάθος από μόνο του δεν εγγυάται και μεγαλύτερη επιτυχία. Το δίκτυο αυτό αν και πιο βαθύ είχε λιγότερους νευρώνες από το προηγούμενο με αποτέλεσμα να μην φτάσει την απόδοση του μεγαλύτερου δικτύου των διαγραμμάτων: Διάγραμμα 4.5, Διάγραμμα 4.6. Παρά το μεγαλύτερο βάθος όμως δεν παρατηρείται υπέρ-προσαρμογή καθώς έχει (όπως και στα προηγούμενα) χρησιμοποιηθεί η ομαλοποίηση L2 (weight decay 1.4.4.3.2)



Διάγραμμα 4.8: Εξέλιξη κόστους / χρονική διάρκεια εποχών σε [784, 30, 100, 30, 10] FFN

Διάγραμμα 4.8: τα καταληκτικά κόστη των δεδομένων εκπαίδευσης και επαλήθευσης είναι 0,095 και 0,3036 αντίστοιχα. Η εκπαίδευση διήρκησε 1302 δευτερόλεπτα (21.7 λεπτά) σαφώς λιγότερο χρόνο από το πολυπληθέστερο σε νευρώνες δίκτυο [780, 150, 300, 10] με την εποχή να διαρκεί κατά μέσο όρο 43.44 δευτερόλεπτα.

Στην συνέχεια εκπαιδευτικέ άλλο ένα τριστρωματικό δίκτυο αλλά αυτή την φορά με πολυπληθέστερα κρυφά στρώματα, συγκεκριμένα: το πρώτο κρυφό αριθμεί στους 100 νευρώνες, το δεύτερο στους 300 και το τρίτο στους 100. Αρά η αρχιτεκτονική είναι [784, 100, 300, 100].Τα ακόλουθα διαγράμματα ακολουθούν κοινή λογική με τα προηγούμενα του κεφαλαίου 4.1.1.

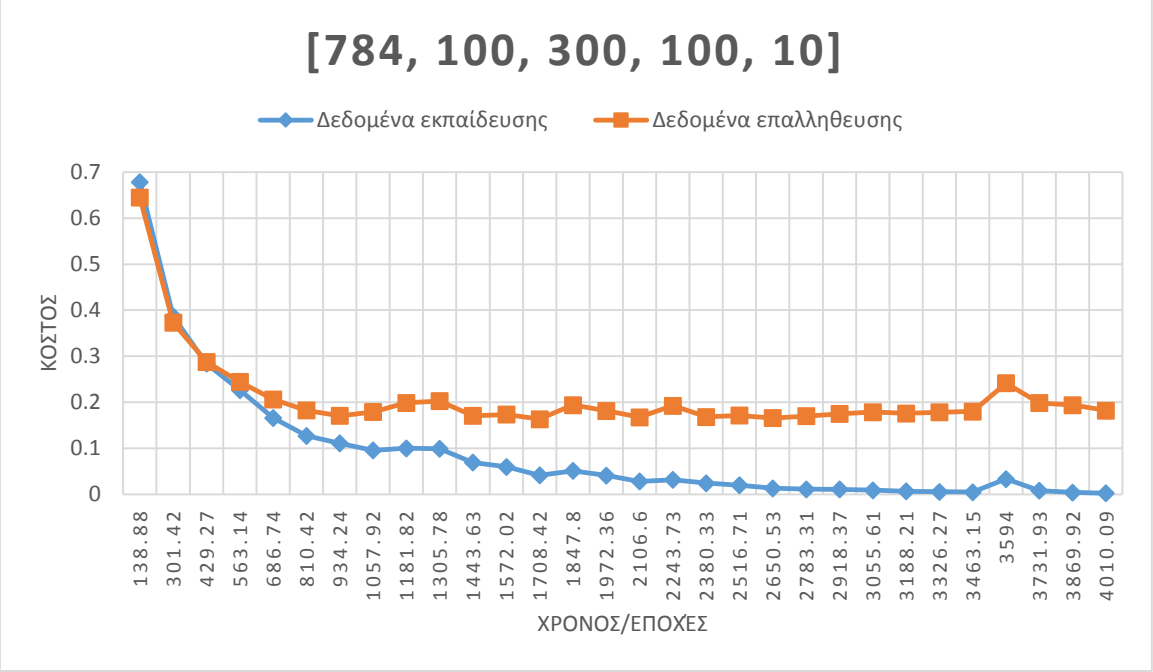


Διάγραμμα 4.9: Εξέλιξη ρυθμών επιτυχίας σε [784, 100, 300, 100, 10] FFN

Όπως ήταν αναμενόμενο το πολυπληθέστερο σε νευρώνες τριστρωματικό δίκτυο πέτυχε καλύτερα αποτελέσματα από το «φτωχότερο» [748, 30, 150, 30, 10]

- Ρυθμός επιτυχίας στα δεδομένα εκπαίδευσης: Ρυθμός την τριακοστή εποχή (καθώς και μέγιστος), 0,99996 έναντι του 0,9897 μέγιστο στο προηγούμενο δίκτυο.
- Ρυθμός επιτυχίας στα δεδομένα επαλήθευσης: Μέγιστός 0,9783, ρυθμός την τριακοστή εποχή, 0,9779 (9,9783 και 0,964στο προηγούμενο)

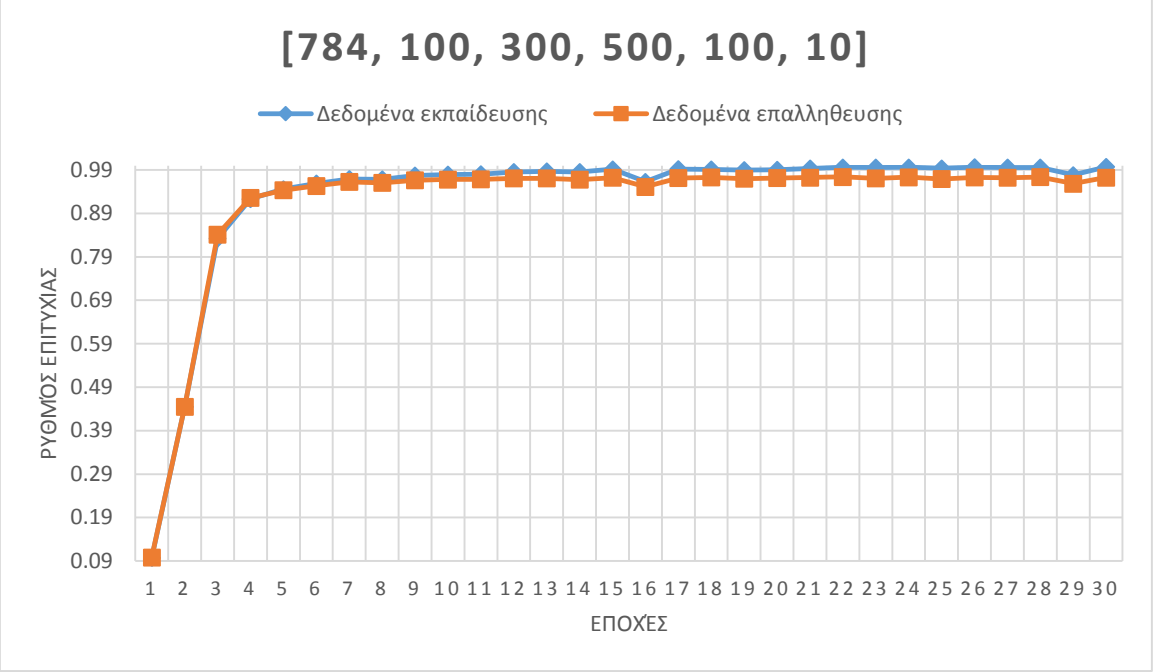
Παρά την βελτίωση των αποτελεσμάτων σε σχέση με το προηγούμενο, «φτωχότερο» τριστρωματικό δίκτυο τα αποτελέσματα παραμένουν ελάχιστα χαμηλότερα από το διστρωματικό δίκτυο [748, 150, 3001 10] των διαγραμμάτων (Διάγραμμα 4.5, Διάγραμμα 4.6). Αυτή η μικρή διαφορά αναδεικνύει το γεγονός ότι όσο αυξάνεται το βάθος του δικτύου η δυσκολία της εκπαίδευσης του αυξάνεται. Για να επιτευχθούν καλύτερα αποτελέσματα θα πρέπει να δοκιμασθούν άλλες τεχνικές βελτίωσης καθώς και άλλες τιμές των υπερπαραμέτρων του δικτύου.



Διάγραμμα 4.10: Εξέλιξη κόστους / χρονική διάρκεια εποχών σε [784, 100, 300, 100, 10] FFN

Στο Διάγραμμα 4.10 φαίνεται ότι ο χρόνος εκπαίδευσης για άλλη μια φορά αυξήθηκε σε σχέση με το προηγούμενο τρισεπίπεδο δίκτυο. Η εκπαίδευσης διήρκησε 4010,09 δευτερόλεπτα (66.83 λεπτά) με τον μέσω χρόνο ανά εποχή να ανέρχεται στα 133.48 δευτερόλεπτα. Τα κόστη στα δεδομένα εκπαίδευσης και επαλήθευσης την τριακοστή εποχή ισούνται με 0,0024 και 0,1831 αντίστοιχα.

Το τελευταίο δίκτυο που εκπαιδεύτηκε είναι ένα έμπροσθεν τροφοδοτούμενο δίκτυο 4 κρυφών στρωμάτων. Το πρώτο στρώμα αριθμεί 100 κρυφούς νευρώνες, το δεύτερο 300 το τρίτο 500 και το τέταρτο 100 νευρώνες [784, 100, 300, 500, 100, 10]. Αυτή είναι η μεγαλύτερη και πιο βαθιά αρχιτεκτονική που εκπαιδεύτηκε εξαιτίας του χρονοβόρου της διαδικασίας αλλά και εξαιτίας τις δυσκολίας που παρουσιάζει η εύρεση βέλτιστων υπερπαραμέτρων. Τα ακόλουθα διαγράμματα (Διάγραμμα 4.11, Διάγραμμα 4.12) ακολουθούν κοινή λογική με τα προηγούμενα του κεφαλαίου 4.1.1.

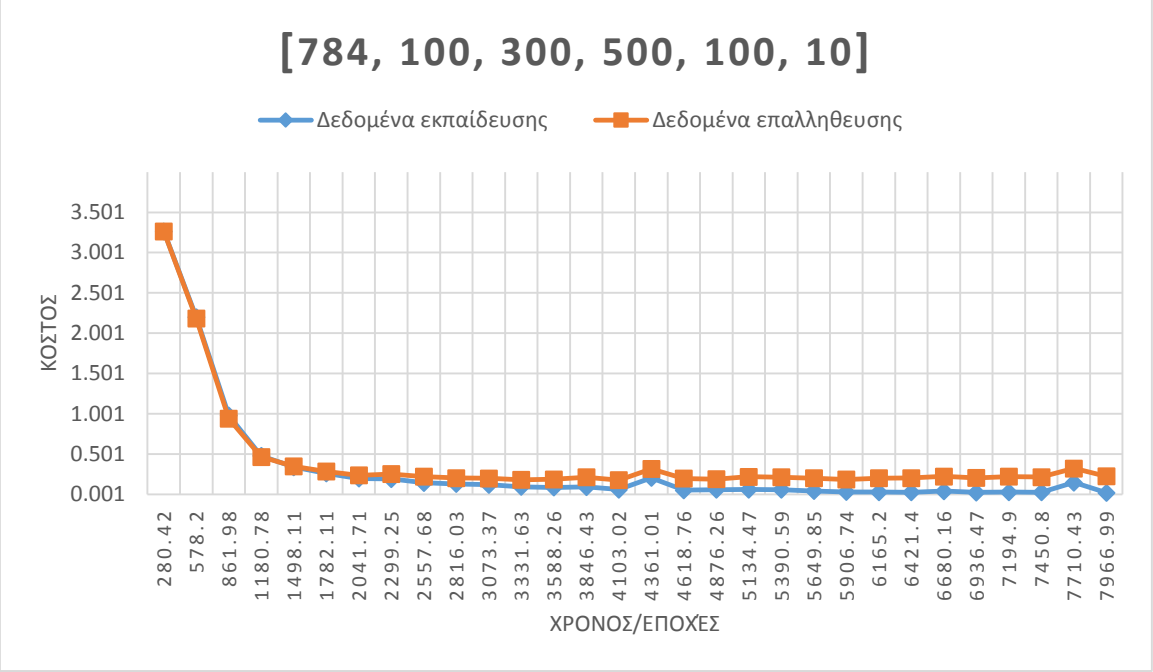


Διάγραμμα 4.11: Εξέλιξη ρυθμών επιτυχίας σε [784, 100, 300, 500, 100, 10] FFN

Στο Διάγραμμα 4.11 εμφανίζεται για πρώτη φόρα το εξής φαινόμενο: Τόσο ο ρυθμός επιτυχίας τόσο στα δεδομένα εκπαίδευσης όσο και στα δεδομένα επαλήθευσης αρχίζει από πολύ χαμηλά επίπεδα. Ο βασικότερος ύποπτος για αυτό το δυσμενές φαινόμενο είναι ότι έγινε λάθος επιλογή για την αρχικοποίηση βαρών. Παρόλα αυτά τα αποτελέσματα είναι αρκετά καλά (όχι τα καλυτέρα ως τώρα όμως)

- Ρυθμός επιτυχίας στα δεδομένα εκπαίδευσης: Ρυθμός την τριακοστή εποχή (καθώς και μέγιστος), 0.99728 έναντι του 0, 99996 μέγιστο στο προηγούμενο δίκτυο.
- Ρυθμός επιτυχίας στα δεδομένα επαλήθευσης: Μέγιστός 0.9736, ρυθμός την τριακοστή εποχή, 0.9725 (0,9783και 0, 9779 στο προηγούμενο)

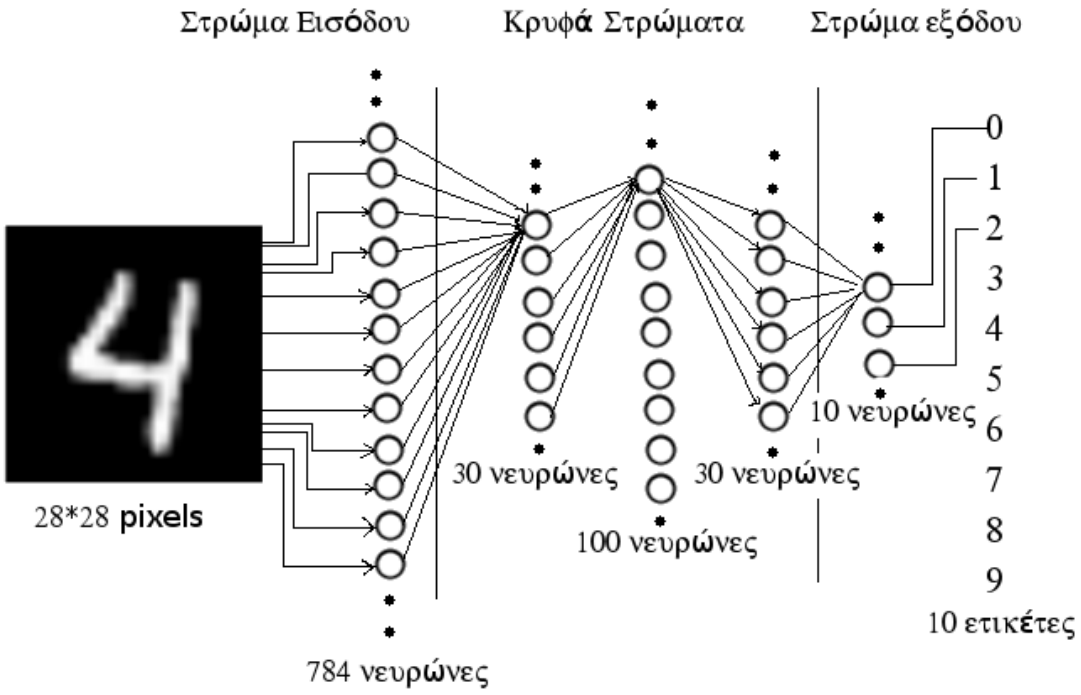
Το μέγιστο αυτό δίκτυο αποτυγχάνει να επιτύχει και τα βέλτιστα αποτελέσματα. Για άλλη μια φορά αναδεικνύεται η δυσκολία εκπαίδευσης βαθιών αρχιτεκτονικών.



Διάγραμμα 4.12: Εξέλιξη κόστους / χρονική διάρκεια εποχών σε [784, 100, 300, 500, 100, 10] FFN

Στο Διάγραμμα 4.12 φαίνεται ότι ο χρόνος εκπαίδευσης για άλλη μια φορά αυξήθηκε σε σχέση με το προηγούμενα δίκτυα. Η εκπαίδευση διήρκησε 7966,99 δευτερόλεπτα (132.78 λεπτά) με τον μέσω χρόνο ανά εποχή να ανέρχεται στα 263.88 δευτερόλεπτα (διπλάσιο του προηγούμενου). Τα κόστη στα δεδομένα εκπαίδευσης και επαλήθευσης την τριακοστή εποχή ισούνται με 0,0209 και 0,2259 αντίστοιχα.

Εκπαιδεύτηκε ένα βαθύ, έμπροσθεν τροφοδοτούμενο δίκτυο για την επίλυση του προβλήματος αναγνώρισης χειρόγραφων ψηφίων στο σετ δεδομένων MNIST. Το δίκτυο αυτό είχε σταθερό μέγεθος το οποίο διατηρήθηκε καθ' όλη την διάρκεια των πειραμάτων. Το δίκτυο είχε μέγεθος $[784, 30, 100, 30, 10]$ (Εικόνα 4.1) και για την εκπαίδευση του χρησιμοποιήθηκε ο αλγόριθμος ΟΔΣ με χρήση στοχαστικής απότομης καθόδου (stochastic gradient descent). Ακολουθούν τα αποτελέσματα πειραμάτων με διαφορετική παραμετροποίησής του δικτύου.



Εικόνα 4.1: Βαθύ Νευρωνικό Δίκτυο Πειραμάτων

Τα δεδομένα εισόδου είναι ασπρόμαυρες εικόνες 28*28 του σετ δεδομένων MNIST.

Για τα παρακάτω πειράματα χρησιμοποιήθηκαν εναλλάξ δύο συναρτήσεις κόστους

- Η τετραδική συνάρτηση κόστους (quadratic cost function/mean squared error)

$$C(w,b) = \frac{1}{2n}\left(\sum_i \left(F_{w,b}\left(x^{(i)}\right) - t^{(i)}\right)\right)^2$$

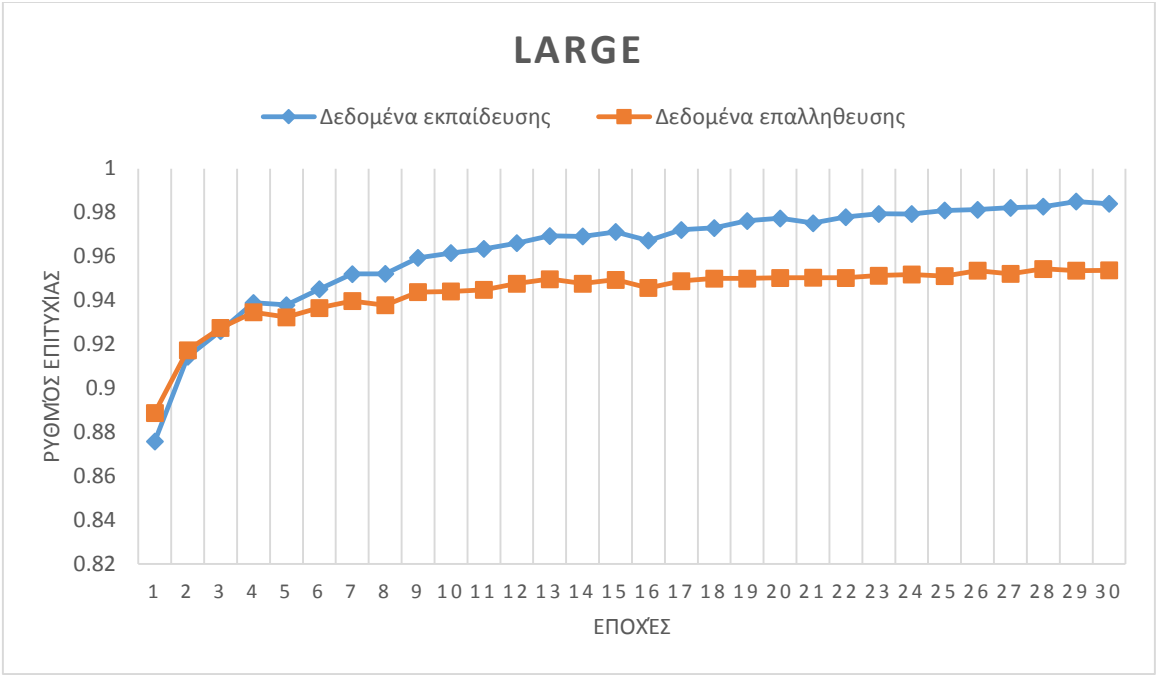
- Η συνάρτηση δια-εντροπίας (cross-entropy)

$$L_{\mathcal{H}}(x,z) = -\left(\sum_{k=1}^{d}[x_k \log z_k + (1-x_k)\log(1-z_k)]\right)$$

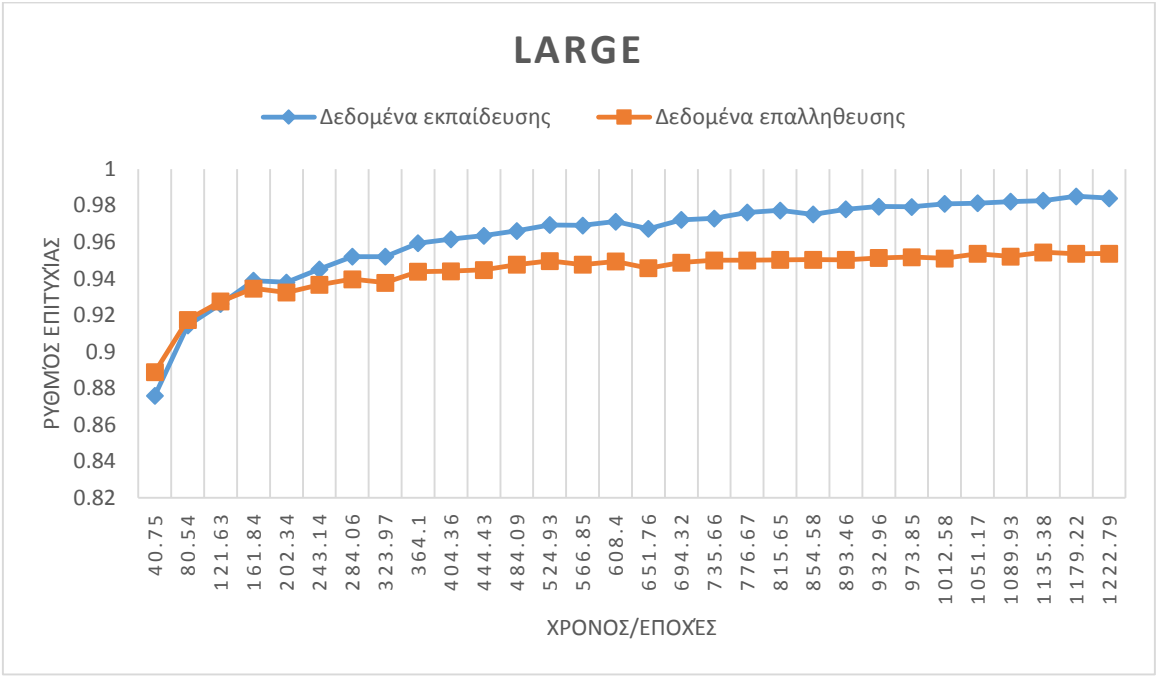
Στα παρακάτω διαγράμματα επιλέχτηκε μια σχετικά μικρή τιμή ρυθμού μάθησης $Learning\_Rate = 0{,}25$ και μέγεθος υπο-ομάδας 20 δειγμάτων εισόδου. Οι διαφοροποιήσεις έγιναν στην επιλογή αλγορίθμου αρχικοποίησης βαρών. Χρησιμοποιήθηκαν 3 τρόποι.

- All_zero: Αρχικοποίηση όλων των βαρών και σταθερών πόλωσης σε μηδέν
- Large: Αρχικοποίηση όλων των βαρών και σταθερών πόλωσης τυχαία, από μία κανονική κατανομή με μέση τιμή $\mu = 0$ και διακύμανση $\sigma^2 = 1$
- Default: Αρχικοποίηση όλων των βαρών και σταθερών πόλωσης τυχαία, από μία κανονική κατανομή με μέση τιμή $\mu = 0$ και διακύμανση $\sigma^2 = 1$. Στην συνέχεια, μόνο για τα βάρη, η τυχαία αυτή τιμή τους διαιρείται με την τιμή $\sqrt{x}$ με x να είναι η το πλήθος των συνδέσεων εισόδου του νευρώνα.

Ακολουθούν διαγράμματα που απεικονίζουν την εξέληξη του ρυθμού επιτυχείας στα δεδομένα εκπαίδευσης και στα δεδομένα επαλήθευσης στην παροδο του χρόνου. Ο τίτλος του κάθε διαγράμματος προσδιορίζει πια τεχνική αρχικοποίησης καθός και πια συνάρτηση κόσοτυς χρησιμοποιήθηκε.



Στο παρακάτω διάγραμμα οι άξονας x έχει αντικατασταθεί με τον χρόνο που απαιτήθηκε για να φτάσει στον συγκεκριμένο ρυθμό επιτυχίας το δίκτυο. Ο χρόνος αντιστοιχεί στο τέλος της εκάστοτε εποχής.



ΠΙΝΑΚΑΣ ΟΡΟΛΟΓΙΑΣ

Ξενόγλωσσος όρος	**Ελληνικός Όρος**
Recurrent Neural Networks	Αναδρομικά Νευρωνικά Δίκτυα
Back-Propagation Algorithm	Αλγόριθμος Οπισθοδρομικής Διάδοσης Σφάλματος
Feed-Forward Neural Networks	Έμπροσθεν Διαδιδόμενα Νευρωνικά Δίκτυα
Generative Models	Παραγωγικά Μοντέλα
Convolutional Neural Networks	Νευρωνικά Δίκτυα Συνέλιξης
Activation	Ενεργοποίηση
Activation Function	Συνάρτηση Ενεργοποίησης
Weight	Συναπτικό Βάρος / Βάρος
Bias	Σταθερά Πόλωσης
Deep Belief Networks	Δίκτυα Βαθιάς Πίστης
Autoencoder	Αυτόματοι Κωδικοποιητές
Receptive Fields	Δεκτικά Πεδία
Echo State Networks	Δίκτυο Κατάστασης Αντήχησης
Denoising Autoencoder	Αραιός αποθορυβοποίητής Autoencoder
Bi-directional RNN	Αμφίδρομα RNN
Feature Map	Χάρτης Χαρακτηριστικών
Sigmoid	Σιγμοειδής
Linear Regression	Γραμμική Παλινδρόμηση
Logistic Regression	Εφοδιαστική Παλινδρόμηση
Gradient Descent	Απότομή Κάθοδος
Stochastic Gradient Descent	Στοχαστική Απότομη Κάθοδος
Momentum	Ορμή
Mini-Batch	Υπο-παρτίδα
Energy Based Models EBMs	Ενεργειακά Μοντέλα
Time-Wrapping-Invariant Echo State Networks	Δίκτυα Κατάστασης Αντήχησης Αναλλοίωτης Αναδίπλωσης Χρόνου

ΣΥΝΤΜΗΣΕΙΣ – ΑΡΚΤΙΚΟΛΕΞΑ – ΑΚΡΩΝΥΜΙΑ

NN	Neural Networks (Νευρωνικά Δίκτυα)
ΝΔ	Νευρωνικά Δίκτυα
MLPs	Multi Layered Perceptrons
DNN/ΒΝΔ	Deep Neural Network/ Βαθύ Νευρωνικό Δίκτυο
ΟΔΣ	Οπισθοδρομικής Διάδοσης Σφάλματος (Αλγόριθμος)
FFN	Feed Forward Neural Networks (Έμπροσθεν Τροφοδοτούμενά ΝΔ)
RBF	Radical Basis Function Network (Δίκτυα Συναρτήσεως Ριζοσπαστικής Βάσης)
RNN	Recurrent Neural Networks (Αναδρομικά ΝΔ)
SCN	Symmetrically Connected Networks (Συμμετρικά Συνδεδεμένα Δίκτυα)
SCNwH	Symmetrically Connected Networks with Hidden units (Συμμετρικά συνδεδεμένα δίκτυα με κρυφούς νευρώνες)
AE/ANN	Autoencoders (Αυτόματοι Κωδικοποιητές)
CNN	Convolutional Neural Networks (ΝΔ Συνέλιξης)
dA/dAE	Denoising Autoencoders (Αποθορυβοποίοι Αυτόματοι Κωδικοποιητές)
SA/SAE	Stacked Autoencoders (Στοιβαγμένοι Αυτόματοι Κωδικοποιητές)
EdA	Emphasized Denoising Autoencoders (Αυτόματοι Κωδικοποιητές με Έμφαση)
BM	Boltzmann Machines (Μηχανές Boltzmann)
DBM	Deep Boltzmann Machines (Βαθιές Μηχανές Boltzmann)
RBM	Restricted Boltzmann Machines (Περιορισμένες Μηχανές Boltzmann)
DBN	Deep Belief Network (Δίκτυο Βαθιάς Πίστης)
RNN	Recurrent Neural Networks (Αναδρομικό ΝΔ)
DRNN	Deep Recurrent Neural Networks (Βαθύ Αναδρομικό ΝΔ)
DO-RNN	Deep Output - Recurrent Neural Networks (Αναδρομικό ΝΔ Βαθιάς Εξόδου)
DT-RNN	Deep Transition - Recurrent Neural Networks (Αναδρομικό ΝΔ Βαθιάς Μετάβασης)
DT(S)-RNN	Deep Transition (Shortcut) - Recurrent Neural Networks

	(Αναδρομικό ΝΔ Βαθιάς Μετάβασης με Συντόμευση)
s-RNN	Stacked Recurrent Neural Networks (Στοιβαγμένο Αναδρομικό ΝΔ)
DOT-RNN	Deep Transition, Output - Recurrent Neural Networks (Αναδρομικό ΝΔ Βαθιάς Μετάβασης Βαθιάς εξόδου)
BRNN	Bi-directional Recurrent Neural Networks (Αμφίδρομα Αναδρομικά ΝΔ)
LSTM	Long Short Term Memory (Δίκτυο μακράς βραχυπρόθεσμης μνήμης)
CTRNN	Continuous time Recurrent Neural Networks (Αμφίδρομα ΝΔ Συνεχούς Χρόνου)
ESN	Echo State Networks (Δίκτυα Κατάστασης Αντήχησης)
TWI-ESN/ TWIESN	Time-Wrapping-Invariant Echo State Networks (Δίκτυα Κατάστασης Αντήχησης Αναλλοίωτης Αναδίπλωσης Χρόνου)
Leaky-ESN	Leaky Echo State Networks (Διαρρέοντα Δίκτυα Κατάστασης Αντήχησης)
DBN-DNN	Deep Belief Network – Deep Neural Network
SGD	Stochastic Gradient Descent
EBMs	Energy Based Models

ΑΝΑΦΟΡΕΣ

[1] D. Harris, «gigaom xbox one voice search powered by deep learning,» gigaom , 11 November 2013. [Ηλεκτρονικό]. Available: https://gigaom.com/2013/11/19/xbox-one-voice-search-powered-by-deep-learning/. [Πρόσβαση 28 May 2015].

[2] M. Inc, «Azure Microsoft Media Indexer Product page,» Microsoft Inc, [Ηλεκτρονικό]. Available: http://azure.microsoft.com/en-gb/services/media-services/media-indexer/.

[3] l. Chen, J. Basilico και X. Mtriain, «Techblog Netflix DNN,» Netflix, Monday February 2014. [Ηλεκτρονικό]. Available: http://techblog.netflix.com/2014/02/distributed-neural-networks-with-gpus.html. [Πρόσβαση 28 May 2015].

[4] S. Dieleman, «Spotify CNN,» Spotify, 05 August 2014. [Ηλεκτρονικό]. Available: http://benanne.github.io/2014/08/05/spotify-cnns.html. [Πρόσβαση 28 May 2015].

[5] J. Wagner και H. Sonya, «Deep Learning: 6 Real World Use,» 2012 December 2014. [Ηλεκτρονικό]. Available: http://go.alchemyapi.com/deep-learning-6-use-cases. [Πρόσβαση 28 May 2015].

[6] enlitic, «Enlitic medical company using deep learning,» enlitic, [Ηλεκτρονικό]. Available: http://www.enlitic.com/tech.html. [Πρόσβαση 28 May 2015].

[7] D. Harris, «gigaom baibu builds super computer for deep learning,» gigaom , 15 January 2015. [Ηλεκτρονικό]. Available: https://gigaom.com/2015/01/14/baidu-has-built-a-supercomputer-for-deep-learning/. [Πρόσβαση 28 May 2015].

[8] D. Harris, «gigaom paypal uses deep learning to fight fraud,» gigaom, 06 March 2015. [Ηλεκτρονικό]. Available: https://gigaom.com/2015/03/06/how-paypal-uses-deep-learning-and-detective-work-to-fight-fraud/. [Πρόσβαση 28 May 2105].

[9] R. F. Molz, P. M. Engel και F. G. Morales, «Codesign of fully parallel neural network for a classification problem,» Université Montpellier.

[10] M. Minsky και S. Papert, Perceptrons: an introduction to computational geometry, 1969.

[11] Stanford, «Standford Neural Networks History,» Stanford, [Ηλεκτρονικό]. Available: http://cs.stanford.edu/people/eroberts/courses/soco/projects/neural-networks/History/history1.html.

[12] L. Yu και D. Dong, «Deep Learning Methods and Applications,» 2014.

[13] J. Schmidhuber, «Deep Learning in Neural Networks: An Overview,» The Swiss AI Lab IDSIA, Lugano, 2014.

[14] reingold, «3.0 History of Neural Networks Artificial Neural Networks Technology,» University of Toronto, [Ηλεκτρονικό]. Available: http://psych.utoronto.ca/users/reingold/courses/ai/cache/neural4.html.

[15] «AlanTuring.net,» [Ηλεκτρονικό]. Available:

http://www.alanturing.net/turing_archive/pages/reference%20articles/connectionism/Turing's%20neural%20networks.html.

[16] G. Hinton, «Coursera: Neural Networks for Machine Learning,» 2012. [Ηλεκτρονικό]. Available: https://class.coursera.org/neuralnets-2012-001.

[17] R. Kumar Srivastava, J. Masci, F. Gomez και J. Schmidhuber, «Understanding Locally Competitive Networks,» Manno–Lugano, Switzerland, 2014.

[18] M. Chai, Y. Shi και J. Liu, «DEEP MAXOUT NEURAL NETWORKS FOR SPEECH RECOGNITION,» Tsinghua National Laboratory for Information Science and Technology Department of Electronic Engineering, Tsinghua University, Beijing, 2014.

[19] I. J. Gooffellow, D. Warde-Farley, M. Mirza, A. Courville και B. Yoshua, «Maxout networks,» σε *International Conference on Machine Learning (ICML)*, 2013.

[20] M. Riedmiller και J. T. Springenberg, «Improving Deep Neural Networks with Probabilistic Maxout Units,» Department of Computer Science University of Freiburg, Freiburg. Germany, 2014.

[21] M. Nielsen, Neural Networks and Deep Learning, Determination Press, 2015.

[22] «UFLDL Tutorial,» Stanford, [Ηλεκτρονικό]. Available: http://ufldl.stanford.edu/tutorial/.

[23] Γ. Ρεφανίδης, *Νευρωνικά Δίκτυα διαφάνιες διάλεξης,* Πανεπιστήμιο Μακεδονίας Τμήμα Εφαρμοσμένης Πληροφορικής, 2013-2014.

[24] N. Qian, «On the momentum term in gradient descent learning algorithms,» *Neural Networks,* αρ. 12, pp. 145-151, 1991.

[25] Ι. Χατζηλάου, Π. Κουπατσιάρης και Γ. Τσεκούρας, «ΕΙΣΑΓΩΓΗ ΣΤΑ ΝΕΥΡΩΝΙΚΑ ΔΙΚΤΥΑ,» ΣΧΟΛΗ ΝΑΥΤΙΚΩΝ ΔΟΚΙΜΩΝ Εργαστήρια Ηλεκτροτεχνίας, 2008.

[26] Y. Bengio, Learning Deep Architectures for AI, NOW Publishers, 2009.

[27] Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato και F. J. Huang, «A Tutorial on Energy-Based Learning,» MIT Press, 2006, New York, 2006.

[28] P. Werbos, «Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences,» Harvard University, 1974.

[29] J. Schmidhuber, «Learning complex, extended sequences using the principle of history compression,» *Neural Computation,* αρ. 4, pp. 234-242, 1992.

[30] J. Schmidhuber, «An ancient experiment with credit assignment across 1200 time steps or virtual layers and unsupervised pre-training for a stack of recurrent NN,» Habilitation thesis, TUM, 1993.

[31] J. Martens, «standford.edu: Hessian Free Deep Learning,» 10 December 2010. [Ηλεκτρονικό]. Available: http://cs229.stanford.edu/proj2010/Iyengar-HessianFreeDeepLearning.pdf. [Πρόσβαση 02 June 2015].

[32] Y. Lee, S.-H. Oh και M. Won Kon, «An Analysis of Premature Saturation in Back Propagation Learning,» *Neural Networks,* τόμ. 6, pp. 719-728, 1993.

[33] J. Vitela και J. Reifman, «Premature saturation in backpropagation networks: Mechanism and necessary conditions,» 1997.

[34] H.-M. Lee, C.-M. Chen και T.-C. Huang, «Learning effciency improvement of back-propagation algorithm by error saturation prevention method,» *Neurocomputing,* αρ. 41, pp. 125-143, 2001.

[35] C. Charalambous, «Conjugate gradient algorithm for effcient training of artifcial neural networks,» *Circuits, Devices and Systems,* τόμ. 139, αρ. 3, pp. 301-310, 1992.

[36] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever και R. Salakhutdinov, «Dropout: A Simple Way to Prevent Neural Networks from Overfitting,» Department of Computer Science University of Toronto, Toronto, 2014.

[37] Y. Bengio, I. Goodfellow και A. Courville, DEEP LEARNING An MIT Press book in preparation, 2015.

[38] «deeplearning.net,» deeplearning.net, [Ηλεκτρονικό]. Available: http://deeplearning.net/tutorial/.

[39] «developer.apple: ConvolutionOperations,» [Ηλεκτρονικό]. Available: https://developer.apple.com/library/ios/documentation/Performance/Conceptual/vImage/ConvolutionOperations/ConvolutionOperations.html.

[40] cs231n, «github.io: convolutional-networks,» [Ηλεκτρονικό]. Available: http://cs231n.github.io/convolutional-networks/.

[41] B. Poxzos και A. Singh, *Introduction to Machine Learning: CMU-10701 Class slides,* Pittsburgh, Pennsylvania: Carnegie Mellon School of Computer Science, Machine Learning Departement, 2014.

[42] Y. Lecun, «LeNet-5, convolutional neural networks,» [Ηλεκτρονικό]. Available: http://yann.lecun.com/exdb/lenet/index.html. [Πρόσβαση 05 06 2015].

[43] A. Krizhevsky, i. Sutskever και G. Hinton, «ImageNet Classification with Deep Convolutional Neural Networks,» 2012.

[44] B. A. Olshausen και D. Field, «Emergence of simple-cell receptive field properties by learning a sparse code for natural Images,» Departement of Psychology, Cornell University, Ithaca, New York.

[45] Y. Bengio, P. Lambin, P. Dan και H. Larochelle, «Greedy Layer-Wise Training of Deep Networks,» Dept. IRO, University of Montreal, Montreal, 2006.

[46] G. Hinton, S. Osindero και Y.-W. Teh, «A Fast Learning Algorithm for Deep Belief Nets,» *Neural Computation,* τόμ. 18, αρ. 1527-1554, 2006.

[47] G. E. Hinton, N. Srivastava και R. R. Salakhutdinov, « Modeling Documents with a Deep Boltzmann Machine,» In Uncertainty in Artificial Intelligence (UAI 2013), 2013.

[48] G. E. Hinton, A. Graves και A. Mohamed, « Speech Recognition with Deep Recurrent

Neural Networks,» Department of Computer Science, University of Toronto, Toronto, 2013.

[49] P. Baldi και K. Hornik, «Neural networks and principal component analysis: Learning from examples without local minima,» *Neural Networks,* τόμ. 2, αρ. 1, pp. 53-58, 1989.

[50] M. Blondet, «BRAINPRINT: IDENTIFYING UNIQUE FEATURES OF NEURAL ACTIVITY,» [Ηλεκτρονικό]. Available: https://prezi.com/edgarbt5xcbu/brainprint-identifying-unique-features-of-neural-activity/.

[51] P. Vincent, H. Larochelle, Y. Bengio και P.-A. Manzgol, «Extracting and Composing Robust Features with Denoising Autencoders,» University of Montreal, Montreal.

[52] A. Ng, «CS294A Lecture notes: Sparse autoencoder,» stanford.edu.

[53] J. Liang και K. Kelly, «Stacked Denoising Autoencoders for Representation Learning».

[54] P. Vincent, H. Larochelle, Y. Bengio και P.-A. Manzagol, «Extracting and Composing Robust Features with Denoising Autoencoders,» University of Montreal, Dept. IRO, Montreal.

[55] A. Makhzani και B. Frey, «k-Sparse Autoencoders,» University of Toronto, Toronto, 2014.

[56] J. Xie, L. Xu και E. Chen, «Image Denoising and Inpainting with Deep Neural Networks,» School of Computer Science and Technology University of Science and Technology of China, Hefei, China, 2012.

[57] K. Cho, «Simple Sparsification Improves Sparse Denoising Autoencoders in Denoising Highly Noisy Images,» Department of Information and Computer Science, Aalto University School of Science, Finland, 2013.

[58] G. E. Hinton, A. Krizhevsky και S. D. Wang, «Transforming Auto-encoders,» Department of Computer Science, University of Toronto, Toronto, 2011.

[59] R. Pascanu, C. Gulcehre, K. Cho και Y. Bengio, «How to Construct Deep Recurrent Neural Networks,» 2014.

[60] M. Hermans και B. Schrauwen, «Training and Analyzing Deep Recurrent Neural Networks,» Ghent University, ELIS departement, Ghent, Belgium, 2013.

[61] A. Graves, A.-r. Mohamed και G. Hinton, «SPEECH RECOGNITION WITH DEEP RECURRENT NEURAL NETWORKS,» Department of Computer Science, University of Toronto, Toronto, 2013.

[62] M. Schuster και K. K. Paliwal, «Bidirectional Recurrent Neural Networks,» *IEEE TRANSACTIONS ON SIGNAL PROCESSING,* τόμ. 45, αρ. 11, 1997.

[63] A. Graves, M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke και J. Schmidhuber, «A Novel Connectionist System for Improved Unconstrained Handwriting Recognition,» *EEE Transactions on Pattern Analysis and Machine Intelligence,* τόμ. 31, αρ. 5, 2009.

[64] A. Mohamed, A. Graves και N. Jaitly, «Hybrid speech recognition with deep bidirectional LSTM,» σε *Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2013.

[65] H. Inman, P. Husbands και D. Cliff, «Seeing the light: Artificial evolution, real vision,» σε *Third international conference on Simulation of adaptive behavior: from animals to animats*, 1994.

[66] M. Quinn, «Evolving communication without dedicated communication channels,» *Advances in Artificial Life. Lecture Notes in Computer Science,* τόμ. 2159, p. 357–366, 2001.

[67] R. Beer, «"The dynamics of adaptive behavior: A research program,» *Robotics and Autonomous System ,* τόμ. 20, p. 257–289, 1997.

[68] S.-X. Lun, X.-S. Yao, H.-Y. Qi και H.-F. Hu, «A novel model of leaky integrator echo state network for time-series prediction,» *Neurocomputing,* τόμ. 159, p. 58–66, 2015.

[69] S. Haykin, Neural Networks and Learning Machines, PearsonPearson Education, Inc, 2009.

[70] R. Salakhutdinov και H. Geoffrey, «Deep Boltzmann Machines,» Toronto.

[71] A. Fischer και C. Igel, «An Introduction to Restricted Boltzmann Machines».

[72] T. Tieleman, «Training Restricted Boltzmann Machines using Approximations to the Likelihood Gradient,» Department of Computer Science, University of Toronto, Toronto, 2008.

[73] A.-r. Mohamed, G. Dahl και G. Hinton, «Deep Belief Networks for phone recognition,» Department of Computer Science University of Toronto, Toronto.

[74] G. E. Hinton, «Scholarpedia: Deep belief networks,» 2009. [Ηλεκτρονικό]. Available: http://www.scholarpedia.org/article/Deep_belief_networks.

[75] T. G. Dietterich και G. Bakiri, «Solving multiclass learning problems via errorcorrecting output codes,» *Journal of Artificial Intelligence Research,* τόμ. 2, p. 263–286, 1995.

[76] J. Garofolo, L. Lamel, W. Fisher, J. Fiscus, D. Pallett και N. Dahlgren, «3.4.1 Core Test Set,» σε *DARPA TIMIT: Acoustic-Phonetic Continuous Speech Corpus CD-ROM*, pp. 20-21.

[77] D. Erhan, B. Yoshua, A. Courville, P.-A. Manzagol και P. Vincent, «Why Does Unsupervised Pre-training Help Deep Learning?,» *Journal of Machine Learning Research,* αρ. 11, pp. 625-660, 2010.

[78] O. Yadan, K. Adams, Y. Taigman και M. Ranzato, «Multi-GPU Training of ConvNets,» Facebook AI Group, 2013 .

[79] Z. Zhang, M. Wang, T. Xiao, J. Li, J. Zhang και C. Hong, «Speech : Minerva: A scalable and highly efficient training platform for deep learning,» 2014.

[80] R. Collobert, K. Kavukcuoglu και C. Farabet, «Torch7: A Matlab-like Environment for

Machine Learing,» 2011.

[81] F. Bastien, P. Lamblin, J. Bergstra, I. Goodfellow, A. Bergeron, N. Bouchard, D. Warde-Farley και Y. Bengio, «Theano: new features and speed improvements,» University of Montreal, Montreal, 2012.

[82] t. d. team, «deeplearning.net/software/theano/,» theano, [Ηλεκτρονικό]. Available: http://deeplearning.net/software/theano/.

[83] Caffe, «caffe.berkeleyvision.org/,» Barkley Caffe, [Ηλεκτρονικό]. Available: http://caffe.berkeleyvision.org/.

[84] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama και T. Darrell, «Caffe: Convolutional Architecture for Fast Feature Embedding*,» UC Berkeley, Berkeley, 2014.

[85] ndd4j, «deeplearning4j.org/,» Skymind, 2015. [Ηλεκτρονικό]. Available: http://deeplearning4j.org/.

[86] V. Parmar και A. Candel, Deep Learning with H2O, LeanPub, 2015.

[87] NVidia, «developer.nvidia.com/cuDNN,» NVidia, [Ηλεκτρονικό]. Available: https://developer.nvidia.com/cuDNN.

[88] I. Goodfellow, D. Warde-Farley, P. Lamblin, V. Dumoulin, M. Mirza, R. Pascanu, J. Bergstra, F. Bastien και Y. Bengio, «Pylearn2: a machine learning research library,» Deparetment of Informatics of Operational Research, University of Montreal; Center of Theoretical Neuroscience, univesity of Waterloo.

[89] «Code.Google,» cuda-convnet, [Ηλεκτρονικό]. Available: https://code.google.com/p/cuda-convnet/source/browse/trunk/example-layers/layers-conv-local-11pct.cfg.

[90] Z. Zygmunt, «Object recognition in images with cuda-convnet,» FastML, 27 November 2013. [Ηλεκτρονικό]. Available: http://fastml.com/object-recognition-in-images-with-cuda-convnet/.

[91] J. Dean, G. S. Corrado, R. Monga, K. Chen, D. Matthieu, Q. V.Le, M. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang και A. Ng, «Large Scale Distributed Deep Networks,» Google.

[92] P. Balakrishnan, «corpocrat: machine-learning-using-restricted-boltzmann-machines,» 17 10 2014. [Ηλεκτρονικό]. Available: http://corpocrat.com/2014/10/17/machine-learning-using-restricted-boltzmann-machines/.

[94] «cuda-convnet2,» googlr, [Ηλεκτρονικό]. Available: https://code.google.com/p/cuda-convnet2/.

[95] M. Ramarlina, «github: Denoising AutoEncoder,» 09 February 2013. [Ηλεκτρονικό]. Available: https://github.com/ramarlina/DenoisingAutoEncoder.

[96] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio και P.-A. Manzagol, «Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a

Local Denoising Criterion,» *Journal of Machine Learning Research,* τόμ. 11, pp. 3371-3408, 2010.

[97] «Wikipedia: Artificial_neuron,» [Ηλεκτρονικό]. Available: https://en.wikipedia.org/wiki/Artificial_neuron.

[98] «sourceforge: Tutorial 2: Creating and training a simple digit classifier,» [Ηλεκτρονικό]. Available: http://eblearn.sourceforge.net/beginner_tutorial2_train.html.

[99] Y. Fan, Y. Qian, F. Xie και F. K. Soong, «TTS Synthesis with Bidirectional LSTM based Recurrent Neural Networks,» hanghai Jiao Tong University, Shanghai, China; Microsoft Research Asia, Beijing, China , Shanghai, 2014.

[100] «cs.toronto.edu: The CIFAR-10 dataset,» [Ηλεκτρονικό]. Available: http://www.cs.toronto.edu/~kriz/cifar.html

[101] G. E. Dahl, D. Yu, D. Li και A. Acero, «LARGE VOCABULARY CONTINUOUS SPEECH RECOGNITION WITH CONTEXT-DEPENDENT DBN-HMMS,» 2011.