



ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ  
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ  
Μ.Δ.Ε «ΕΠΙΣΤΗΜΗ ΚΑΙ ΤΕΧΝΟΛΟΓΙΑ ΗΛΕΚΤΡΟΛΟΓΟΥ ΜΗΧΑΝΙΚΟΥ ΚΑΙ ΜΗΧΑΝΙΚΟΥ  
ΥΠΟΛΟΓΙΣΤΩΝ»

**Ταχεία Εκπαίδευση Νευρωνικών Δικτύων Βασισμένη σε Έννοιες  
της Θεωρίας Δικτύων**  
**Fast Network Science – Based Training of Neural Networks**

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ  
του  
**ΝΙΚΟΛΑΟΥ ΣΑΚΕΛΛΑΡΙΟΥ**

**Επιβλέπων :** Δημήτριος Κατσαρός  
Αν. Καθηγητής ΤΗΜΜΥ Πανεπιστημίου Θεσσαλίας

Βόλος, Φεβρουάριος 2019



## **Τριμελής Επιτροπή**

Αντωνόπουλος Χρήστος

Αν. Καθηγητής ΤΗΜΜΥ Πανεπιστημίου Θεσσαλίας

Κατσαρός Δημήτριος

Αν. Καθηγητής ΤΗΜΜΥ Πανεπιστημίου Θεσσαλίας

Τουσίδου Ελένη

Ε.ΔΙ.Π. ΤΗΜΜΥ Πανεπιστημίου Θεσσαλίας



## Περίληψη

Σκοπός της παρούσας διπλωματικής εργασίας είναι η έρευνα και υλοποίηση τεχνικών βελτίωσης του χρόνου εκπαίδευσης και εκτέλεσης βαθιών νευρωνικών δικτύων παραμετροποιώντας τα βάρη των ακμών τους διατηρώντας υψηλή ακρίβεια. Συγκεκριμένα έγινε πειραματισμός και τροποποίηση του Multilayer Perceptron αλγορίθμου με αφαίρεση βαρών κατά την διαδικασία της εκπαίδευσης. Αρχικά δοκιμάστηκε η αραιοποίηση του δικτύου αφαιρώντας ένα σταθερό αριθμό βαρών και κρατώντας το πλήθος τους σταθερό καθ' όλη τη διαδικασία της εκπαίδευσης(μέθοδος DRP) και εν συνεχεία δοκιμάστηκε μια δεύτερη μέθοδος(D-NIB) η οποία αφαιρεί τυχαία ένα μικρό πλήθος βαρών σε κάθε epoch. Και οι δύο μέθοδοι παρουσιάζουν ενδιαφέροντα αποτελέσματα διατηρώντας υψηλή ακρίβεια precision και recall με αυξημένο όμως Loss, ενώ για κάποιες τοπολογίες παρατηρήθηκε ταχύτερη εκπαίδευση του αλγορίθμου.

**Λέξεις Κλειδιά:** <<Multilayer Perceptron, νευρωνικά δίκτυα, βελτίωση >>

## **Abstract**

Scope of this thesis is the research and development of technics that minimize the training time of a deep neural network while keeping good accuracy precision and recall. The means to achieve that was the parameterization of the weights on the edges of the network. Specifically, the Multilayer Perceptron algorithm was used and experiments were made on removing multiple weights from the network during the algorithm's training phase. For that purpose, two techniques were used in order to find an efficient way for removing network weights. DRP and D-NIB are proposed. The first removes a fix percentage of the artificial neural network's weights while the second removes weights analogous to the number of samples and the batch size parameter of the MLP model. Both methods produce interesting results in terms of accuracy and training time.

**Keywords:** <<Multilayer Perceptron, Deep Neural Network, Deep Learning, >>

## Πίνακας περιεχομένων

### Contents

Ευχαριστίες .....	9
Εισαγωγή .....	10
Μηχανική Μάθηση .....	10
Τεχνητά Νευρωνικά Δίκτυα και Βαθιά Μάθηση(Deep Learning) .....	11
Αντικείμενο διπλωματικής.....	11
Οργάνωση κειμένου.....	12
Κεφάλαιο 1 - Τεχνητά Νευρωνικά Δίκτυα .....	13
1.1 Εισαγωγή στα Νευρωνικά Δίκτυα .....	13
1.2 Ιστορική Αναδρομή .....	16
1.3 Βιολογικά Νευρωνικά Δίκτυα .....	19
1.4 Διαδικασίες μάθησης Νευρωνικών Δικτύων .....	20
1.5 Μέθοδος Οπισθοδιάδοσης του Λάθους (Backpropagation) .....	22
Κεφάλαιο 2 - Multilayer Perceptron .....	24
Κεφάλαιο 3 – Σχετικές Εργασίες.....	25
3.1 Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science.....	25
3.2 Dropout .....	25
3.3 meProp: Sparsified Back Propagation for Accelerated Deep Learning with Reduced Overfitting ..	26
Κεφάλαιο 4 – Μέθοδοι .....	28
<b>4.1 Drop Random edges at fix Percentage (DRP) .....</b>	<b>28</b>
4.1.1 Περιγραφή DRP .....	28
4.1.2 Υλοποίηση .....	28
<b>4.2 Drop based on Number of Instances per Batch size(D-NIB) .....</b>	<b>35</b>
4.2.1 Περιγραφή D-NIB.....	35
4.2.2 Υλοποίηση .....	35
Κεφάλαιο 5 – Πειραματική Αξιολόγηση .....	42
Αποτελέσματα Digits Dataset(MNIST).....	42
Πίνακας 1 – Digits Dataset DRP.....	43
Πίνακας 2 – Digits Dataset D-NIB .....	46
Πίνακας 3 – Digits Dataset Vanilla.....	48
Συμπεράσματα .....	49

Κεφάλαιο 6 - Επίλογος .....	50
Βιβλιογραφία .....	51



## Ευχαριστίες

Ευχαριστώ θερμά τον καθηγητή και μέντορα μου Δημήτριο Κατσαρό καθώς και τον συνάδελφο και φίλο μου Παναγιώτη Γουλιδάκη για τη βοήθεια τους στη περάτωση της παρούσας διπλωματικής εργασίας.

“Time, time is the only thing in life you can’t get most of, and it’s the one thing that will mercilessly tear you up when it’s gone”  
Sherrilyn Kenyon

# Εισαγωγή

## Μηχανική Μάθηση

Μηχανική μάθηση είναι υποπεδίο της επιστήμης των υπολογιστών που αναπτύχθηκε από τη μελέτη της αναγνώρισης προτύπων και της υπολογιστικής θεωρίας μάθησης στην τεχνητή νοημοσύνη. Το 1959, ο Άρθουρ Σάμουελ ορίζει τη μηχανική μάθηση ως "Πεδίο μελέτης που δίνει στους υπολογιστές την ικανότητα να μαθαίνουν, χωρίς να έχουν ρητά προγραμματιστεί". Η μηχανική μάθηση διερευνά τη μελέτη και την κατασκευή αλγορίθμων που μπορούν να μαθαίνουν από τα δεδομένα και να κάνουν προβλέψεις σχετικά με αυτά. Τέτοιοι αλγόριθμοι λειτουργούν κατασκευάζοντας μοντέλα από πειραματικά δεδομένα, προκειμένου να κάνουν προβλέψεις βασιζόμενες στα δεδομένα ή να εξάγουν αποφάσεις που εκφράζονται ως το αποτέλεσμα.

Η μηχανική μάθηση είναι στενά συνδεδεμένη και συχνά συγχέεται με υπολογιστική στατιστική, ένας κλάδος, που επίσης επικεντρώνεται στην πρόβλεψη μέσω της χρήσης των υπολογιστών. Έχει ισχυρούς δεσμούς με την μαθηματική βελτιστοποίηση, η οποία παρέχει μεθόδους, τη θεωρία και τομείς εφαρμογής. Η Μηχανική μάθηση εφαρμόζεται σε μια σειρά από υπολογιστικές εργασίες, όπου τόσο ο σχεδιασμός όσο και ο ρητός προγραμματισμός των αλγορίθμων είναι ανέφικτος. Παραδείγματα εφαρμογών αποτελούν τα φίλτρα spam (spam filtering), η οπτική αναγνώριση χαρακτήρων (OCR), οι μηχανές αναζήτησης και η υπολογιστική όραση. Η Μηχανική μάθηση μερικές φορές συγχέεται με την εξόρυξη δεδομένων, όπου η τελευταία επικεντρώνεται περισσότερο στην εξερευνητική ανάλυση των δεδομένων, γνωστή και ως μη επιτηρούμενη μάθηση.

Στο πεδίο της ανάλυσης δεδομένων, η μηχανική μάθηση είναι μια μέθοδος που χρησιμοποιείται για την επινόηση πολύπλοκων μοντέλων και αλγορίθμων που οδηγούν στην πρόβλεψη. Τα αναλυτικά μοντέλα επιτρέπουν στους ερευνητές, τους επιστήμονες δεδομένων, τους μηχανικούς και τους αναλυτές να παράγουν αξιόπιστες αποφάσεις και αποτελέσματα και να αναδείξουν αλληλοσυσχετίσεις μέσω της μάθησης από ιστορικές σχέσεις και τάσεις στα δεδομένα.

### **Επιβλεπόμενη μηχανική μάθηση ή μάθηση με επίβλεψη(supervised learning)**

Το υπολογιστικό πρόγραμμα δέχεται τις παραδειγματικές εισόδους καθώς και τα επιθυμητά αποτελέσματα από έναν «δάσκαλο», και ο στόχος είναι να μάθει έναν γενικό κανόνα προκειμένου να αντιστοιχίσει τις εισόδους με τα αποτελέσματα.

### **Μη επιβλεπόμενη μηχανική μάθηση ή μάθηση χωρίς επίβλεψη(unsupervised learning)**

Χωρίς να παρέχεται κάποια εμπειρία στον αλγόριθμο μάθησης, πρέπει να βρεί την δομή των δεδομένων εισόδου. Η μη επιβλεπόμενη μάθηση μπορεί να είναι αυτοσκοπός (ανακαλύπτοντας κρυμμένα μοτίβα σε δεδομένα) ή μέσο για ένα τέλος (χαρακτηριστικό της μάθησης).

## Τεχνητά Νευρωνικά Δίκτυα και Βαθιά Μάθηση(Deep Learning)

Τα τεχνητά νευρωνικά δίκτυα ανήκουν στον τομέα της μηχανικής μάθησης και είναι ένα από τα πιο όμορφα προγραμματιστικά παραδείγματα που δημιουργήθηκαν ποτέ. Στη συμβατική προσέγγιση του προγραμματισμού λέμε στον υπολογιστή τι να κάνει, σπάζοντας μεγάλα προβλήματα σε πολύ μικρά καλώς ορισμένα tasks που ο υπολογιστής μπορεί να αντιληφθεί και να υλοποιήσει εύκολα. Σε αντίθεση με τα νευρωνικά δίκτυα όπου δεν λέμε στον υπολογιστή πως να λύσει το πρόβλημα. Αντ' αυτού, μαθαίνει παρατηρώντας δεδομένα, βρίσκοντας τις δικές του λύσεις για το εκάστοτε πρόβλημα.

Η αυτόματη αυτή μάθηση από δεδομένα ακούγεται πολύ υποσχόμενη. Ωστόσο μέχρι το 2006 δεν γνωρίζαμε πως να εκπαιδεύσουμε νευρωνικά δίκτυα ώστε να ξεπεράσουν πιο παραδοσιακές προσεγγίσεις εκτός μερικών ειδικών προβλημάτων. Αυτό που άλλαξε το 2006 ήταν η ανακάλυψη τεχνικών μάθησης γνωστών ως βαθιά νευρωνικά δίκτυα. Αυτές οι τεχνικές είναι πλέον γνωστές ως βαθιά μάθηση(deep learning). Αναπτύχθηκαν περαιτέρω και σήμερα τα βαθιά νευρωνικά δίκτυα και η βαθιά μάθηση πετυχαίνουν εξαιρετική απόδοση σε πολλά σημαντικά προβλήματα στους τομείς της όρασης υπολογιστή, αναγνώρισης φωνής και επεξεργασία φυσικής γλώσσας. Χρησιμοποιούνται σε μεγάλη κλίμακα από εταιρίες όπως η Google, Microsoft και Facebook.

### Αντικείμενο διπλωματικής

Αντικείμενο της παρούσας διπλωματικής είναι η αναζήτηση μηχανισμών βελτίωσης του χρόνου εκπαίδευσης και εκτέλεσης του Multilayer Perceptron αλγορίθμου ο οποίος ανήκει στους αλγορίθμους βαθιάς μηχανικής μάθησης. Σκοπός είναι να βελτιώνεται ο χρόνος εκπαίδευσης διατηρώντας υψηλή ακρίβεια με το δυνατόν μικρότερο λάθος(loss) στη συνάρτηση σφάλματος.

Το Multilayer Perceptron(MLP) είναι μία κλάση τεχνητών νευρωνικών δικτύων πρόσθιας τροφοδότησης. Ένα MLP αποτελείται από τουλάχιστον τρία επίπεδα κόμβων(hidden layers): το επίπεδο εισόδου δεδομένων, ένα κρυφό επίπεδο και το επίπεδο εξόδου. Εκτός από τους κόμβους εισόδου, κάθε κόμβος είναι ένας νευρώνας που χρησιμοποιεί μία μη γραμμική συνάρτηση ενεργοποίησης. Ο MLP χρησιμοποιεί μία τεχνική επιβλεπόμενης μάθησης γνωστή ως οπισθοδρόμηση(backpropagation) για τη διαδικασία μάθησής του. Τα πολλά επίπεδα(layers) και οι μη γραμμικές συναρτήσεις ενεργοποίησης διαφοροποιούν τον MLP από ένα γραμμικό perceptron. Δηλαδή ο MLP μπορεί να διακρίνει και κατηγοριοποιήσει δεδομένα τα οποία δεν είναι γραμμικά διαχωρίσιμα.

Για τον εν λόγω αλγόριθμο λοιπόν έγινε έλεγχος και πειραματισμός για διάφορες τοπολογίες και πλήθος κόμβων. Δοκιμάστηκαν από ένα έως τρία κρυφά επίπεδα και από 50 έως 1000 κρυφοί κόμβοι για κάθε επίπεδο. Υλοποιήθηκαν και εφαρμόστηκαν σε κάθε τοπολογία δύο τεχνικές.

Τη πρώτη τεχνική ας την ονομάσουμε «**Drop Random edges at fix Percentage(DRP)**». Στον DRP ορίζουμε ένα όριο(threshold) το οποίο αντιπροσωπεύει το ποσοστό των βαρών που θα αφαιρεθούν από την αρχική τοπολογία. Εν συνεχεία στο στάδιο της αρχικοποίησης του MLP αφαιρούνται με τυχαίο τρόπο βάρη(μηδενίζονται) από τον πίνακα βαρών μέχρι το πλήθος των αναιρεμένων βαρών να ισούται με το προκαθορισμένο όριο. Για παράδειγμα αν έχουμε ορίσει

όριο 0.9, το 90% του πίνακα των βαρών του MLP θα μηδενιστεί και θα παραμείνει έτσι καθ' όλη τη διάρκεια εκπαίδευσης του δικτύου. Ουσιαστικά θα διαμορφώνεται σε κάθε εποχή μόνο το 10% των βαρών. Ο στόχος αυτής της τεχνικής είναι εφόσον υπάρχει πλέον ένας πολύ αραιός πίνακας βαρών, να μειωθεί πολύ ο χρόνος εκτέλεσης της διαδικασίας πρόσθιας τροφοδότησης. Πιθανά μειονεκτήματα αυτής της μεθόδου είναι εν τέλη η αύξηση του χρόνου εκτέλεσης λόγω του χρόνου μετατροπής του πίνακα των βαρών σε αραιό καθώς και η πιθανότητα overfitting του μοντέλου.

Τη δεύτερη τεχνική ας την ονομάσουμε «**Drop based on Number of Instances per Batch size(D-NIB)**». Ο D-NIB αφαιρεί με μια πιθανότητα βάρη από τον πίνακα των βαρών σε κάθε εποχή. Το συνολικό πλήθος των βαρών που αφαιρούνται μέχρι το τέλος της εκπαίδευσης εξαρτάται από το πλήθος των δειγμάτων εκπαίδευσης(training set size), το *batch size* και το πλήθος των εποχών. Σκοπός αυτής της υλοποίησης είναι επίσης να μειωθεί ο χρόνος εκτέλεσης εφόσον αφαιρούμε βάρη αλλά αυτό να γίνεται με λίγες μόνο παραπάνω προγραμματιστικές εντολές σε επίπεδο κώδικα ώστε να μην εντάσσεται επιπλέον πολυπλοκότητα στην *vanilla* υλοποίηση του αλγορίθμου εξαιρώντας τη πιθανότητα να έχει το αντίθετο αποτέλεσμα όσον αφορά τη βελτίωση του χρόνου εκτέλεσης.

### Οργάνωση κειμένου

Η εργασία οργανώνεται ως εξής:

- Στο πρώτο κεφάλαιο παρουσιάζεται μια εισαγωγή στα τεχνητά νευρωνικά δίκτυα.
- Στο κεφάλαιο 2 περιγράφεται ο αλγόριθμος Multilayer Perceptron.
- Στο τρίτο κεφάλαιο παρουσιάζονται σχετικές εργασίες(δημοσιεύσεις) και πηγές έμπνευσης για τη προσέγγιση υλοποίησης αυτής της εργασίας.
- Στο κεφάλαιο 4 αναλύονται οι τρόποι που προέκυψαν πειραματικά για την επίλυση του προβλήματος.
- Στο κεφάλαιο 5 παρουσιάζονται τα αποτελέσματα.
- Τέλος, το κεφάλαιο 6 αποτελεί τη σύνοψη.

## Κεφάλαιο 1 - Τεχνητά Νευρωνικά Δίκτυα

### 1.1 Εισαγωγή στα Νευρωνικά Δίκτυα

Το νευρωνικό δίκτυο είναι ένα δίκτυο από απλούς υπολογιστικούς κόμβους (νευρώνες, νευρώνια), διασυνδεδεμένους μεταξύ τους. Είναι εμπνευσμένο από το Κεντρικό Νευρικό Σύστημα (ΚΝΣ), το οποίο προσπαθεί να προσομοιώσει.

Οι νευρώνες είναι τα δομικά στοιχεία του δικτύου. Κάθε τέτοιος κόμβος δέχεται ένα σύνολο αριθμητικών εισόδων από διαφορετικές πηγές (είτε από άλλους νευρώνες, είτε από το περιβάλλον), επιτελεί έναν υπολογισμό με βάση αυτές τις εισόδους και παράγει μία έξοδο. Η εν λόγω έξοδος είτε κατευθύνεται στο περιβάλλον, είτε τροφοδοτείται ως είσοδος σε άλλους νευρώνες του δικτύου. Υπάρχουν τρεις τύποι νευρώνων: οι νευρώνες εισόδου, οι νευρώνες εξόδου και οι υπολογιστικοί νευρώνες ή κρυμμένοι νευρώνες. Οι νευρώνες εισόδου δεν επιτελούν κανέναν υπολογισμό, μεσολαβούν απλώς ανάμεσα στις περιβαλλοντικές εισόδους του δικτύου και στους υπολογιστικούς νευρώνες. Οι νευρώνες εξόδου διοχετεύουν στο περιβάλλον τις τελικές αριθμητικές εξόδους του δικτύου. Οι υπολογιστικοί νευρώνες πολλαπλασιάζουν κάθε είσοδό τους με το αντίστοιχο *συναπτικό βάρος* και υπολογίζουν το ολικό άθροισμα των γινομένων. Το άθροισμα αυτό τροφοδοτείται ως όρισμα στη *συνάρτηση ενεργοποίησης*, την οποία υλοποιεί εσωτερικά κάθε κόμβος. Η τιμή που λαμβάνει η συνάρτηση για το εν λόγω όρισμα είναι και η έξοδος του νευρώνα για τις τρέχουσες εισόδους και βάρη.

Εάν  $x_{ki}$  είναι η  $i$ -οστή είσοδος του  $k$  νευρώνα,  $w_{ki}$ : το  $i$ -οστό συναπτικό βάρος του  $k$  νευρώνα και  $\phi(\cdot)$  η συνάρτηση ενεργοποίησης του νευρωνικού δικτύου, τότε η έξοδος  $y_k$  του  $k$  νευρώνα δίνεται από την εξίσωση:

$$y_k = \phi \left( \sum_{i=0}^N x_{ki} w_{ki} \right)$$

Στον  $k$ -οστό νευρώνα υπάρχει ένα συναπτικό βάρος  $w_{k0}$  με ιδιαίτερη σημασία, το οποίο καλείται **πόλωση** ή **κατώφλι** (bias, threshold). Η τιμή της εισόδου του είναι πάντα η μονάδα,  $x_{k0} = 1$ . Εάν το συνολικό άθροισμα από τις υπόλοιπες εισόδους του νευρώνα είναι μεγαλύτερο από την τιμή αυτή, τότε ο νευρώνας ενεργοποιείται. Εάν είναι μικρότερο, τότε ο νευρώνας παραμένει ανενεργός. Η ιδέα προέκυψε από τα βιολογικά νευρικά κύτταρα.

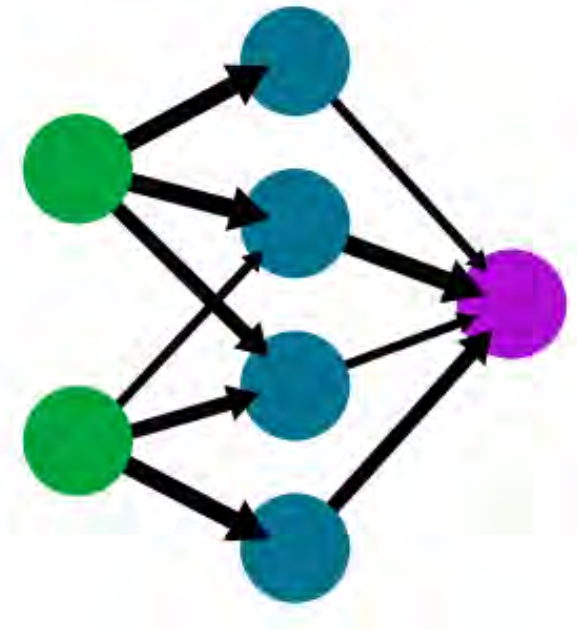


Figure 1 Παράδειγμα Τεχνητού Νευρωνικού Δικτύου

Όπως είναι φανερό, οι αριθμοί οι οποίοι συναποτελούν το διάνυσμα εισόδου (κάθε στοιχείο του διανύσματος τροφοδοτείται κατά τη λειτουργία του δικτύου σε έναν νευρώνα εισόδου), αλλά και οι αριθμοί οι οποίοι συναποτελούν το διάνυσμα εξόδου (κάθε στοιχείο του οποίου εμφανίζεται, μετά το πέρας του ολικού υπολογισμού, σε έναν νευρώνα εξόδου), περιγράφουν χαρακτηριστικά του προς επίλυση προβλήματος. Συνήθως αυτό που μας ενδιαφέρει είναι το δίκτυο να απεικονίζει με ορθό τρόπο διανύσματα εισόδου σε κατάλληλα διανύσματα εξόδου, το πρόβλημα δηλαδή είναι η υλοποίηση μίας συνάρτησης πολλαπλών μεταβλητών, κατά κανόνα περίπλοκης και με άγνωστο ακριβή τύπο. Τέτοιες απεικονίσεις έχουν εφαρμογή σε ποικιλία τομέων της επιστήμης και της τεχνολογίας, αφού λειτουργούν ως αριθμητικά μοντέλα για πολλά διαφορετικά ζητήματα. Το ίδιο δίκτυο μπορεί να υλοποιήσει άπειρες διαφορετικές απεικονίσεις, μία για κάθε διαφορετική επιλογή συνόλου συναπτικών βαρών.

Το κύριο χαρακτηριστικό των νευρωνικών δικτύων είναι η εγγενής ικανότητα μάθησης. Ως μάθηση μπορεί να οριστεί η σταδιακή βελτίωση της ικανότητας του δικτύου να επιλύει κάποιο πρόβλημα (π.χ. η σταδιακή προσέγγιση μίας συνάρτησης). Η μάθηση επιτυγχάνεται μέσω της **εκπαίδευσης**, μίας επαναληπτικής διαδικασίας σταδιακής προσαρμογής των παραμέτρων του δικτύου (συνήθως των βαρών και της πόλωσής του) σε τιμές κατάλληλες ώστε να επιλύεται με επαρκή επιτυχία το προς εξέταση πρόβλημα. Αφού ένα δίκτυο εκπαιδευτεί, οι παράμετροί του συνήθως «παγώνουν» στις κατάλληλες τιμές και από εκεί κι έπειτα είναι σε λειτουργική κατάσταση. Το ζητούμενο είναι το λειτουργικό δίκτυο να χαρακτηρίζεται από μία ικανότητα **γενίκευσης**: αυτό σημαίνει πως δίνει ορθές εξόδους για εισόδους καινοφανείς και διαφορετικές από αυτές με τις οποίες εκπαιδεύτηκε.

## Ιδιότητες

Τα τελευταία χρόνια έχει υπάρξει μία έκρηξη ενδιαφέροντος για τα νευρωνικά δίκτυα καθώς εφαρμόζονται με μεγάλη επιτυχία σε ένα ασυνήθιστα μεγάλο φάσμα τομέων της επιστήμης και της τεχνολογίας, όπως τα χρηματοοικονομικά, η ιατρική, η επιστήμη μηχανικού, η γεωλογία, η φυσική, η ρομποτική, η επεξεργασία σήματος κτλ. Στην πραγματικότητα, τα νευρωνικά δίκτυα εισάγονται οπουδήποτε τίθεται θέμα πρόβλεψης, ταξινόμησης ή ελέγχου. Η σαρωτική αυτή επιτυχία, μπορεί να αποδοθεί σε δύο βασικά στοιχεία: την ισχύ και την ευχρηστία.

**Ισχύς:** Τα νευρωνικά δίκτυα είναι πολύ εξελιγμένες τεχνικές μη γραμμικής μοντελοποίησης, ικανές να μοντελοποιήσουν εξαιρετικά πολύπλοκες λειτουργίες. Η γραμμική μοντελοποίηση υπήρξε ευρέως διαδεδομένη για πολύ καιρό, δεδομένου ότι στα γραμμικά μοντέλα εφαρμόζονται πολύ γνωστές στρατηγικές βελτιστοποίησης. Στις συνήθειες, όμως, περιπτώσεις όπου η γραμμική προσέγγιση δεν ήταν έγκυρη, τα μοντέλα αυτά αποτύγχαναν αναλόγως. Τα νευρωνικά δίκτυα βέβαια, αν και επιτρέπουν τη μη γραμμικότητα μέσω χρήσης μη γραμμικών συναρτήσεων ενεργοποίησης, μεταθέτουν με τη σειρά τους το πρόβλημα στο ζήτημα της διάστασης (του πλήθους των διαφορετικών εισόδων και εξόδων), το οποίο αποτελεί αγκάθι στις προσπάθειες μοντελοποίησης μη γραμμικών συναρτήσεων με μεγάλο αριθμό μεταβλητών.

**Ευχρηστία:** Τα νευρωνικά δίκτυα εκπαιδεύονται με παραδείγματα. Ο χρήστης συγκεντρώνει αντιπροσωπευτικά δεδομένα και στη συνέχεια, καθώς τα τροφοδοτεί συστηματικά στο δίκτυο μέσω των κατάλληλων αλγορίθμων εκπαίδευσης, το δίκτυο «αντιλαμβάνεται» αυτομάτως τη δομή των δεδομένων και η «γνώση» αυτή εκφράζεται ως κατάλληλες επιλογές συναπτικών βαρών. Επομένως το τελικό αποτέλεσμα της εκπαίδευσης με ένα συγκεκριμένο σύνολο παραδειγμάτων είναι ο προσδιορισμός των κατάλληλων βαρών του δικτύου. Ο χρήστης χρειάζεται να έχει κάποιες ουσιώδεις γνώσεις σχετικά με τον τρόπο επιλογής και προετοιμασίας των δεδομένων, τον τρόπο εκλογής του κατάλληλου νευρωνικού δικτύου και στο πως θα ερμηνευτούν τα αποτελέσματα. Παρά ταύτα, το επίπεδο των γνώσεων του χρήστη που απαιτούνται για μια επιτυχημένη εφαρμογή των νευρωνικών δικτύων, είναι πολύ χαμηλότερο συγκριτικά με κάποια περίπτωση που θα χρησιμοποιούνταν ορισμένες πιο παραδοσιακές, μη γραμμικές στατιστικές μέθοδοι.

## Εφαρμογές

Τα νευρωνικά δίκτυα είναι εφαρμόσιμα σχεδόν σε κάθε κατάσταση στην οποία ισχύει μια σχέση μεταξύ μεταβλητών πρόβλεψης (ανεξάρτητες, εισροές) και προβλεπόμενες μεταβλητές (εξαρτημένες, εκροές), ακόμα και όταν αυτή η σχέση είναι πολύ περίπλοκη για να αποδοθεί με τους συνηθισμένους όρους της «συσχέτισης» ή των «διαφόρων ομάδων». Ενδεικτικά αντιπροσωπευτικά παραδείγματα προβλημάτων στα οποία η ανάλυση των νευρωνικών δικτύων έχει εφαρμοστεί με επιτυχία είναι τα εξής:

- **Ιατρική διάγνωση:** Ένα ευρύ φάσμα ιατρικά συσχετιζόμενων ενδείξεων, όπως ο συνδυασμός της καρδιακής συχνότητας, τα επίπεδα των διαφόρων ουσιών στο αίμα, ο ρυθμός της αναπνοής μπορούν να παρακολουθηθούν. Η εκδήλωση μιας συγκεκριμένης ιατρικής κατάστασης, γίνεται να συσχετιστεί με ένα πολύπλοκο συνδυασμό μεταβολών σε



ένα υποσύνολο μεταβλητών που παρακολουθούνται. Τα νευρωνικά δίκτυα έχουν χρησιμοποιηθεί για την αναγνώριση αυτού του προτύπου πρόβλεψης, ώστε να χορηγηθεί η κατάλληλη θεραπεία.

- Χρηματιστηριακές προβλέψεις: Οι διακυμάνσεις των τιμών των μετοχών και των χρηματιστηριακών δεικτών είναι ακόμα ένα παράδειγμα ενός πολύπλοκου, πολυδιάστατου, αλλά και σε ορισμένες περιπτώσεις εν μέρει ντετερμινιστικού φαινομένου. Τα νευρωνικά δίκτυα χρησιμοποιούνται από πολλούς τεχνικούς αναλυτές, ώστε να κάνουν προβλέψεις σχετικά με τις τιμές των μετοχών, βασιζόμενοι σε ένα μεγάλο αριθμό παραγόντων, όπως δηλαδή, τις προηγούμενες επιδόσεις άλλων αποθεμάτων και διαφορών οικονομικών δεικτών.
- Πιστωτική ανάθεση: Μια ποικιλία από κομμάτια πληροφοριών, τα οποία είναι συνήθως γνωστά για ένα απαιτούμενο δάνειο. Για παράδειγμα, η ηλικία του αιτούντος, η εκπαίδευση, το επάγγελμα και πολλά άλλα στοιχεία που μπορεί να είναι διαθέσιμα. Μετά την εκπαίδευση ενός νευρωνικού δικτύου σε ιστορικά δεδομένα η ανάλυση μπορεί να εκτοπίσει τα πιο κατάλληλα και σχετικά χαρακτηριστικά και να τα χρησιμοποιήσει για την ταξινόμηση των αιτούντων ως χαμηλού ή υψηλού κινδύνου.
- Παρακολούθηση της κατάστασης των μηχανημάτων: Τα νευρωνικά δίκτυα μπορούν να συμβάλλουν στη μείωση του κόστους με την εξασφάλιση της πρόσθετης εμπειρογνωμοσύνης για τον προγραμματισμό προληπτικής συντήρησης των μηχανημάτων. Ένα νευρωνικό δίκτυο, λοιπόν, μπορεί να εκπαιδευτεί με τέτοιο τρόπο, ώστε να διακρίνει από τους ήχους τους οποίους παράγει μια μηχανή είτε αν εκτελεί κανονικά τις λειτουργίες της, είτε βρίσκεται στα πρόθυρα εμφάνισης οποιασδήποτε δυσλειτουργίας. Μετά από αυτήν την περίοδο εκπαιδευτικής κατάρτισης, η εμπειρία του ίδιου δικτύου είναι δυνατό να χρησιμοποιηθεί με σκοπό την προειδοποίηση ενός τεχνικού για κάποια επικείμενη βλάβη προτού συμβεί και ενδεχομένως προκαλέσει πολυδάπανες και απρόβλεπτες χρονικές καθυστερήσεις.
- Συστήματα διαχείρισης κινητήρα: Τα νευρωνικά δίκτυα έχουν χρησιμοποιηθεί για την ανάλυση των εισροών που δέχονται οι αισθητήρες ενός κινητήρα. Το νευρωνικό δίκτυο ελέγχει μια ποικιλία παραμέτρων με τις οποίες λειτουργεί ο κινητήρας, προκειμένου να επιτευχθεί ένας συγκεκριμένος στόχος. Για παράδειγμα, το δίκτυο αυτό επιχειρεί την ελαχιστοποίηση της κατανάλωσης των καυσίμων.

## 1.2 Ιστορική Αναδρομή

Καθ' ότι η περιοχή των νευρωνικών δικτύων είναι σχετικά μία νέα περιοχή, δεν έχει ουσιαστικά μεγάλη προϊστορία, όπως άλλες παραδοσιακές επιστήμες. Ξεκίνησε μόλις πριν 50 χρόνια, αλλά η μεγάλη ώθηση σ' αυτά δόθηκε μετά το 1980. Αξίζει λοιπόν τον κόπο να κάνουμε μία σύντομη αναδρομή και να δούμε πως φτάσαμε στις τελευταίες εξελίξεις. Η ανάπτυξη των νευρωνικών δικτύων πέρασε από πολλές φάσεις, και εξελίξεις.



Το πρώτο μοντέλο νευρωνικού δικτύου το οποίο προτείνει ότι οι νευρώνες είναι η βασική μονάδα του δικτύου παρουσιάστηκε το 1943 από τους McCulloch και Pitts . Σε μία πρώτη εργασία τους παρουσίασαν για πρώτη φορά την ιδέα ότι ένα νευρωνικό δίκτυο αποτελείται από μία συλλογή ενός μεγάλου αριθμού νευρώνων, και έδειξαν πως θα μπορούσαν να λειτουργούν οι νευρώνες με τις διασυνδέσεις τους. Αυτή θεωρείται ιστορικά ότι είναι η πρώτη εικόνα ενός νευρωνικού δικτύου. Μάλιστα οι συγγραφείς θεώρησαν ότι οι νευρώνες και οι συνδέσεις τους είναι ένα πρότυπο, ανάλογο ενός ηλεκτρικού κυκλώματος. Ο McCulloch ήταν νευροφυσιολόγος και ο Pitts ένας 18χρονος πρωτοετής φοιτητής των Μαθηματικών. Οι ίδιοι συγγραφείς προχώρησαν το 1947 σε πιο εξελιγμένο πρότυπο για την αναγνώριση σχημάτων . Το πρότυπο αυτό περιέχει πολλά χαρακτηριστικά από τα μεταγενέστερα πρότυπα. Ο νευρώνας θεωρείται ότι μπορεί να έχει δύο μόνον καταστάσεις. Μπορεί να δέχεται πολλές εισόδους αλλά δίνει μία μόνον έξοδο. Οι έξοδοι από διαφορετικούς νευρώνες δεν επιτρέπεται να ενώνονται, αλλά πρέπει υποχρεωτικά να οδηγούν σε είσοδο άλλου νευρώνα. Οι απολήξεις των νευρώνων είναι δύο ειδών: διεγερτικές και ανασταλτικές. Οι δύο καταστάσεις του νευρώνα είναι ότι είτε πυροδοτεί ή βρίσκεται σε ηρεμία. Η ροή της πληροφορίας μέσα στον νευρώνα ελέγχεται από πύλες, οι οποίες επίσης είναι διεγερτικές ή ανασταλτικές. Όταν πυροδοτεί στέλνει ένα παλμό. Οι λειτουργίες αυτές πάντα γίνονται σε διάκριτο χρόνο, και υποτίθεται ότι όλοι οι νευρώνες αποκρίνονται ταυτόχρονα, δηλ. το σύστημα δρα συγχρονισμένα. Η κατάσταση ενός νευρώνα σε χρόνο  $t+1$  εξαρτάται από την κατάστασή του σε χρόνο  $t$  και από τις εισόδους που εισέρχονται στην χρονική αυτή στιγμή. Στα δίκτυα McCulloch-Pitts μπορούμε να θεωρήσουμε ότι ο μηχανισμός μνήμης μπορεί να είναι η ύπαρξη κλειστών διαδρομών του σήματος μέσα στο δίκτυο. Αν δεν υπάρχει καμία τέτοια διαδρομή και χωρίς νέο εξερχόμενο σήμα, τότε το δίκτυο θα μείνει μόνιμα σε κατάσταση ηρεμίας. Έτσι, μια ίνα ενώνει την έξοδο ενός κυττάρου με το σημείο εισόδου στο ίδιο κύτταρο, δημιουργώντας έτσι έναν μηχανισμό ανάδρασης (feedback). Μόλις πυροδοτεί ένα τέτοιο κύτταρο θα συνεχίσει να πυροδοτεί μέχρι να έλθει σήμα από ανασταλτική ίνα. Καθ' όλη τη διάρκεια της λειτουργίας αποστέλλονται παλμοί στην πύλη των κυττάρων και μεταδίδεται το σήμα και η πληροφορία. Ο κύκλος αυτός του σήματος είναι ο μηχανισμός μνήμης.

Οι εργασίες αυτές πιθανόν να χάνονταν στην βιβλιογραφία αν δεν τις χρησιμοποιούσε ο J. Von Neumann ως παράδειγμα για υπολογιστικές μηχανές την δεκαετία που διαδόθηκε ο ηλεκτρονικός υπολογιστής, δηλ. την δεκαετία του πενήντα. Τότε έγιναν και οι πρώτες προσπάθειες να αντλήσουμε πληροφορίες από τα βιολογικά δίκτυα και να δημιουργηθούν τα πρώτα τεχνητά δίκτυα. Ένα άλλο έργο της πρώτης αυτής εποχής που αφήνει ακόμα και σήμερα την επιρροή του είναι το βιβλίο του D. Hebb, "The organisation of behavior" (1949), το οποίο εισάγει τον κανόνα μάθησης του Hebb. Το μοντέλο του Hebb έχει ως κεντρική ιδέα τις συνδέσεις μεταξύ μονάδων του συστήματος, δηλαδή τους νευρώνες. Έφτασε στα συμπεράσματα αυτά μετά από σωρεία πειραμάτων νευροφυσιολογίας. Ο κανόνας αυτός λέγει ότι κάθε φορά που το δίκτυο χρησιμοποιεί τις νευρωνικές του συνδέσεις, οι συνδέσεις αυτές ενισχύονται και το δίκτυο πλησιάζει περισσότερο στο να μάθει το πρότυπο το οποίο παρουσιάζεται. Όταν ο νευρώνας  $i$  επανειλημμένα διεγείρει τον νευρώνα  $j$ , τότε συμβαίνει να αναπτύσσεται μια μεταβολική σύνδεση στον ένα ή και στους δύο νευρώνες, έτσι ώστε η απόδοση του φαινομένου (το  $i$  διεγείρει το  $j$ ) να αυξάνεται. Αν  $w_{ij}$  είναι το βάρος της σύνδεσης μεταξύ  $i$  και  $j$ ,  $x_i$  η είσοδος στον νευρώνα  $j$  από τον νευρώνα  $i$ ,  $y_i$  η έξοδος του νευρώνα  $j$ , τότε ισχύει ότι:  $w_{ij}(\text{new}) = w_{ij}(\text{old}) + a x_i x_j$

Εδώ  $a$  είναι μία θετική σταθερά που λέγεται παράμετρος του ρυθμού εκπαίδευσης. Ο κανόνας αυτός έχει τοπικό χαρακτήρα, ισχύει δηλαδή μόνο για την σύνδεση του νευρώνα  $i$  και  $j$  και όχι για άλλες συνδέσεις του δικτύου.

Το μοντέλο του αισθητήρα (perceptron) παρουσιάστηκε για πρώτη φορά το 1957 από τον F. Rosenblatt, ο οποίος αρχικά έφτιαξε το πρώτο δίκτυο με hardware, και το οποίο μπορούσε να κάνει πολλές και διάφορες διεργασίες. Είναι ένα πολύ απλό μοντέλο (οι λεπτομέρειες θα παρουσιαστούν στα επόμενα κεφάλαια) που έχει μόνο δύο επίπεδα, της εισόδου και της εξόδου. Το σήμα προχωρά μονοδρομικά από την είσοδο στην έξοδο. Το μοντέλο αυτό στην αρχή είχε πολλές επιτυχίες, δημιούργησε μεγάλο ενθουσιασμό, και μάλιστα ήδη αρχίζει να συζητείται η ιδέα ότι πιθανόν τα νευρωνικά δίκτυα να είναι η πιο ανώτερη τεχνική που λύνει όλα τα προβλήματα που μέχρι τότε παρέμεναν άλυτα. Οι πρώτες λοιπόν επιτυχίες μεγαλοποιήθηκαν, αλλά γρήγορα φάνηκε ότι τα μοντέλα αυτά είχαν πολλούς περιορισμούς. Μια συνολική και 4 εμπεριστατωμένη εικόνα του προτύπου αυτού παρουσιάστηκε το 1969 στο βιβλίο "Perceptrons" των Minsky και Papert. Στο βιβλίο αυτό γίνεται μία συνολική εκτίμηση της χρησιμότητας του προτύπου του αισθητήρα και όλων των διεργασιών για τα οποία είναι χρήσιμο. Αποδεικνύεται με αναλυτικά μαθηματικά ότι υπάρχουν συγκεκριμένοι περιορισμοί στο πρότυπο αυτό. Έτσι, δεν μπορεί να λύσει, π.χ. το σχετικά απλό πρόβλημα του X-OR. Οι αρχικές προσδοκίες που είχαν δημιουργηθεί ήδη φαίνεται ότι δεν επαληθεύονται, και προς το παρόν τα νευρωνικά δίκτυα χάνουν την δημοτικότητα τους, και ο κόσμος στρέφεται σε μία νέα παρεμφερή περιοχή που τότε άρχισε να γίνεται γνωστή, την Τεχνητή Νοημοσύνη. Την ίδια περίπου εποχή με την ανάπτυξη του μοντέλου του αισθητήρα οι Widrow και Hoff ανέπτυξαν το 1959 δύο νέα μοντέλα, το Adaline και το Madaline. Μάλιστα αυτά τα μοντέλα ήταν τα πρώτα μοντέλα που χρησιμοποιήθηκαν επιτυχώς για πρακτικά προβλήματα: Χρησιμοποιήθηκαν ως φίλτρα για να εξαλείψουν την ηχώ σε τηλεφωνικές γραμμές.

Ένα μνημιώδες έργο παρουσιάστηκε το 1982 από τον J. Hopfield, ο οποίος είναι βιολόγος, και το οποίο έδωσε μεγάλη ώθηση στην ανάπτυξη των δικτύων. Σε μία εργασία του μόλις 5 σελίδων ο Hopfield έδειξε με αυστηρά μαθηματική απόδειξη πως ένα νευρωνικό δίκτυο μπορεί να χρησιμοποιηθεί ως αποθηκευτικός χώρος (storage device), και πως επίσης μπορεί ένα δίκτυο να επανακτήσει όλη την πληροφορία ενός συστήματος έστω και του δοθούν μερικά τμήματα μόνο, και όχι ολόκληρο το σύστημα. Αμέσως εκτιμήθηκε η σπουδαιότητα της ιδιότητας αυτής, και ως εκ τούτου η εργασία αυτή έγινε έμπνευση για πολλές άλλες ιδέες που ακολούθησαν. Το 1986 δημοσιεύεται ένα άλλο σημαντικό έργο από τους McClelland και Rumelhart, το "Parallel Distributed Processing" το οποίο ανοίγει νέους δρόμους στην εκπαίδευση των νευρωνικών δικτύων. Παρουσιάζεται η ιδέα πως ένα νευρωνικό δίκτυο μπορεί να θεωρηθεί και χρησιμοποιηθεί ως παράλληλος επεξεργαστής. Το έργο αυτό κάνει ένα σημαντικό βήμα πέραν από το Perceptron, με το να επιτρέπει την ύπαρξη και άλλων επιπέδων νευρώνων, εκτός από την είσοδο και την έξοδο, που αποτελούν την εσωτερική δομή του δικτύου. Προτείνουν μία νέα διαδικασία εκπαίδευσης, την μέθοδο της οπισθοδιάδοσης (back-propagation), η οποία κατέληξε να είναι η πιο χρήσιμη σήμερα τεχνική εκπαίδευσης δικτύων. Η μέθοδος αυτή είχε συζητηθεί και

από άλλους νωρίτερα, αλλά για πρώτη φορά το 1986 παρουσιάστηκε ολοκληρωμένα και με αυστηρό μαθηματικό τρόπο.

## **Η κατάσταση σήμερα**

Μετά την πρόοδο σε τόσα πολλά σημεία που παρουσιάστηκε ιδιαίτερα την δεκαετία του 1980, τα τελευταία δέκα χρόνια παρατηρούμε ότι αρχίζουν να εμφανίζονται πολλά σημεία που δείχνουν ότι η περιοχή των νευρωνικών δικτύων έχει πλέον αναπτυχθεί σε ένα ανεξάρτητο πεδίο της επιστήμης με δικά του στοιχεία, δικό του χαρακτήρα σαφώς καθορισμένο, και τέλος με μεγάλο αριθμό επιστημόνων που ασχολούνται αποκλειστικά τώρα με την νέα αυτή περιοχή. Τα στοιχεία αυτά είναι: Από το 1985 και μετά αρχίζουν τα πρώτα συνέδρια που είναι αφιερωμένα αποκλειστικά σε νευρωνικά δίκτυα, από την American Physical Society, και από την IEEE. Παρακολουθούνται από περισσότερους από χίλιους συνέδρους. Ταυτόχρονα δημιουργούνται ειδικές επαγγελματικές εταιρίες νευρωνικών δικτύων με χιλιάδες μέλη, όπως η International Neural Network Society με τρεις πόλους: Αμερική (με διευθυντή τον Grossberg), Ευρώπη (Kohonen), και Ιαπωνία (Amari). Προς τα τέλη της δεκαετίας του ογδόντα παρουσιάζονται τουλάχιστον πέντε νέα περιοδικά αφιερωμένα αποκλειστικά στα νευρωνικά δίκτυα, ενώ πριν λίγα χρόνια δεν υπήρχε ούτε ένα. Τα τελευταία χρόνια μετά το 1990 εκδίδονται και άλλα 3- 4 νέα, με συνέπεια να υπάρχουν σήμερα περίπου 10 επιστημονικά περιοδικά αφιερωμένα στα νευρωνικά δίκτυα. Φυσικά, και τα γνωστά περιοδικά της Επιστήμης Υπολογιστών, της Φυσικής, και των Ηλεκτρολόγων Μηχανικών επίσης περιλαμβάνουν πλειάδα άρθρων με νέα αποτελέσματα. Κάθε μήνα πλέον δημοσιεύονται εκατοντάδες εργασίες με αποκλειστικό θέμα κάποια άποψη των νευρωνικών δικτύων.

## **1.3 Βιολογικά Νευρωνικά Δίκτυα**

Τα νευρωνικά δίκτυα αναπτύχθηκαν μέσα από τις διεξαγωγές ερευνών της Τεχνητής Νοημοσύνης. Προσπάθειες, δηλαδή, μίμησης της ανοχής σε βλάβες και τη δυνατότητα εξόρυξης γνώσης μέσα από βιολογικά νευρωνικά συστήματα, μοντελοποιώντας τη δομή των χαμηλών επιπέδων του εγκεφάλου. Στη συνέχεια, από το κύριο μέρος της έρευνας της Τεχνητής Νοημοσύνης που έγινε το 1960-1980, προήλθαν τα Έμπειρα\_Συστήματα (Expert Systems). Τα συστήματα αυτά έχουν βασιστεί σε ένα μοντέλο λογικής διεργασίας υψηλού επιπέδου, το οποίο δημιουργήθηκε από τον τρόπο με τον οποίο έχει δομηθεί η συλλογιστική διεργασία των ανθρώπων σχετικά με το χειρισμό των συμβόλων. Έγινε γρήγορα εμφανές πως αυτά τα συστήματα αν και ήταν πολύ χρήσιμα σε κάποιους τομείς, απέτυχαν να συλλάβουν καίριες πτυχές της ανθρώπινης νοημοσύνης. Σύμφωνα με μια πτυχή της μελέτης, το γεγονός αυτό οφειλόταν στην αποτυχία τους να μιμηθούν τη βασική δομή του εγκεφάλου. Προκειμένου να αναπαραχθεί νοημοσύνη, κρίνεται απαραίτητη η δημιουργία συστημάτων με παρόμοια τεχνική.

Ο ανθρώπινος εγκέφαλος αποτελείται κατά κύριο λόγο από ένα ευρύ φάσμα νευρώνων (86.000.000.000 κατά προσέγγιση), οι οποίοι είναι μαζικά διασυνδεδεμένοι με ένα μέσο όρο από διάφορες χιλιάδες διασυνδέσεις ανά νευρώνα. Κάθε νευρώνας είναι ένα εξειδικευμένο κύτταρο το οποίο έχει τη δυνατότητα μετάδοσης ενός ηλεκτροχημικού σήματος. Ο νευρώνας έχει μια διακλαδωτική διάρθρωση εισροών, τους δενδρίτες (dendrites), ένα κυτταρικό σώμα και μια

διακλαδωτική δομή εκροών (τον άξονα). Οι άξονες ενός κυττάρου συνδέονται με τους δενδρίτες ενός άλλου, μέσω μιας σύναψης. Όταν, λοιπόν, ένας άξονας ενεργοποιηθεί, πυροδοτεί ένα ηλεκτροχημικό σήμα κατά μήκος του άξονα. Ένας νευρώνας εκτελεί αυτή τη διαδικασία μόνο όταν το συνολικό σήμα το οποίο λήφθηκε από τους δενδρίτες, υπερβεί ένα συγκεκριμένο επίπεδο, δηλαδή, το ουδό ενεργοποίησης (firing threshold).

Η ισχύς ενός σήματος που λαμβάνεται από ένα νευρώνα, εξαρτάται από την αποτελεσματικότητα των συνάψεων. Κάθε σύναψη περιέχει ένα κενό με νευροδιαβιβαστές χημικών ουσιών (neurotransmitter chemicals) που είναι σε ετοιμότητα για μετάδοση ενός μηνύματος. Ο Donald Hebb, ένας από τους πιο σημαντικούς ερευνητές στα νευρολογικά συστήματα, έθεσε ως ζήτημα πως η μάθηση συνιστάται κυρίως από τη μεταβολή της ισχύος των συναπτικών συνδέσμων.

Ως παράδειγμα τίθεται το πείραμα του Pavlov για την Κλασική Εξάρτηση. Η κλασική εξάρτηση είναι μια μορφή συνειρμικής μάθησης, η οποία παρουσιάστηκε για πρώτη φορά από τον Ivan Pavlov και περιλαμβάνει την παρουσίαση ενός ουδέτερου ερεθίσματος μαζί με κάποιο σημαντικό ερέθισμα. Ο Pavlov πειραματιζόμενος με σκύλους, παρατήρησε πως ορισμένα ερεθίσματα, όπως ο ήχος των βημάτων του εκτροφέα που πλησίαζε ή ο ήχος ενός κουδουνιού κατά την προσφορά της τροφής ενεργοποιούσε την έκκριση σιέλου, όπως ακριβώς την ενεργοποιούσε η διατροφή. Με την επανάληψη του πειράματος δημιουργήθηκε ένα καινούριο ανακλαστικό. Ο νέος αυτός τρόπος διασύνδεσης μεταξύ δύο ερεθισμάτων έγινε γνωστός ως εξαρτημένο ανακλαστικό (conditioned reflex) και η διαδικασία ονομάστηκε κλασική εξάρτηση και μέσω αυτής επέρχεται σημαντική αλλαγή της συμπεριφοράς.

Πρόσφατες έρευνες στη γνωσιακή επιστήμη και ιδιαίτερα στον τομέα της ασυνείδητης επεξεργασίας πληροφοριών, απέδειξαν περαιτέρω την τεράστια ικανότητα του ανθρώπινου μυαλού να καταλήγει σε απλές συνδιακυμάνσεις εισροών-εκροών, από εξαιρετικά πολύπλοκα ερεθίσματα. Επομένως, από ένα τεράστιο αριθμό ιδιαίτερα απλών μονάδων εργασίας, ο εγκέφαλος κατορθώνει την εκτέλεση εξαιρετικά πολύπλοκων καθηκόντων. Παρουσιάζει, μάλιστα, μεγάλο ενδιαφέρον το γεγονός ότι τα τεχνητά νευρωνικά δίκτυα έχουν την ικανότητα να επιτύχουν τόσο αξιόλογα αποτελέσματα χρησιμοποιώντας ένα μοντέλο όχι και τόσο πολύπλοκο.

## 1.4 Διαδικασίες μάθησης Νευρωνικών Δικτύων

Μια από τις πιο βασικές ιδιότητες των Νευρωνικών Δικτύων είναι η ικανότητά τους για εκπαίδευση. Η εκπαίδευση αυτή επιτυγχάνεται μέσω της ανταλλαγής τιμών και βαρών, που αποσκοπεί στη βαθμιαία σύλληψη της πληροφορίας η οποία στη συνέχεια θα είναι διαθέσιμη προς ανάκτηση. Υπάρχουν, βέβαια, πολλοί αλγόριθμοι που η εφαρμογή τους έχει στόχο την προσαρμογή των τιμών των βαρών ενός Τεχνητού Νευρωνικού Δικτύου. Όλες οι μέθοδοι μάθησης μπορούν να καταταχτούν σε δύο κατηγορίες : τη **μάθηση με επίβλεψη** (supervised learning) και τη **μάθηση χωρίς επίβλεψη** (unsupervised learning).

**Μάθηση με επίβλεψη:** Η μάθηση αυτή είναι μια διαδικασία η οποία συνδυάζει έναν εξωτερικό εκπαιδευτή και τη συνολική ή γενικευμένη πληροφορία. Κάποιες από τις μεθόδους οι οποίες συγκαταλέγονται σε αυτή την κατηγορία είναι η μάθηση με διόρθωση σφάλματος, η στοχαστική μάθηση. Παραδείγματα τα οποία αντιπροσωπεύουν την μάθηση με επίβλεψη συμπεριλαμβάνουν

αποφάσεις για το πότε θα πρέπει να σταματήσει η διαδικασία εκπαίδευσης, αποφάσεις αναφορικά με τη συχνότητα παρουσίας στο δίκτυο τα πρότυπα εκπαίδευσης και η παρουσίαση προόδου του δικτύου. Η μάθηση με επίβλεψη χωρίζεται σε δύο ακόμα κατηγορίες: στη **δομική** (structural) και στην **προσωρινή** (temporal) εκμάθηση. Οι αλγόριθμοι οι οποίοι βρίσκονται στην πρώτη κατηγορία, χρησιμοποιούνται για την εύρεση της βέλτιστης σχέσης μεταξύ εισόδων και εξόδων για κάθε ξεχωριστό ζευγάρι προτύπων. Παραδείγματα της δομικής εκμάθησης αποτελούν η αναγνώριση και η κατηγοριοποίηση προτύπων, ενώ παραδείγματα της προσωρινής εκμάθησης η πρόβλεψη και ο έλεγχος.

**Μάθηση χωρίς επίβλεψη:** Οι αλγόριθμοι της εν λόγω μάθησης αναφέρονται ως αυτό-οργανώμενοι (self-organized) και είναι διαδικασίες οι οποίες δεν απαιτούν να είναι παρών ένας «εξωτερικός» δάσκαλος ή επιβλέπων. Βασίζονται, μάλιστα, μόνο σε τοπική πληροφορία καθ' όλη τη διάρκεια της εκπαίδευσης του Τεχνητού Νευρωνικού Δικτύου. Οι συγκεκριμένοι αλγόριθμοι οργανώνουν τα δεδομένα και ανακαλύπτουν τις σημαντικές συλλογικές ιδιότητες. Για παράδειγμα, αλγόριθμοι εκπαίδευσης χωρίς επίβλεψη είναι ο αλγόριθμος Hebbian, ο διαφορικός αλγόριθμος Hebbian και ο Min-Max αλγόριθμος.

Κατά κύριο λόγο οι περισσότερες διαδικασίες εκπαίδευσης είναι off line. Όταν χρησιμοποιείται όλο το δείγμα προτύπων για την τροποποίηση των τιμών των βαρών, πριν την τελική χρήση του δικτύου ως εφαρμογή, τότε ονομάζεται off line εκπαίδευση. Οι αλγόριθμοι εκπαίδευσης off line έχουν την απαίτηση να βρίσκονται στην εκπαίδευση του δικτύου παρόντα όλα τα πρότυπα. Το γεγονός αυτό αποκλείει την πιθανότητα εισαγωγής νέων πληροφοριών μέσω νέων προτύπων. Βέβαια, υπάρχουν και Τεχνητά Νευρωνικά Δίκτυα τα οποία δεν αποκλείουν την εισαγωγή νέας πληροφορίας, μετά την τελική τους μοντελοποίηση. Αν παρουσιαστεί ανάγκη εισαγωγής νέου προτύπου στο δίκτυο, μπορεί να γίνει απευθείας χωρίς τον κίνδυνο να χαθεί κανένα μέρος της αρχικής πληροφορίας. Το πλεονέκτημα των δικτύων που χρησιμοποιούν off line διαδικασίες εκπαίδευσης επικεντρώνεται κυρίως στη δυνατότητα να δίνουν καλύτερες λύσεις σε δύσκολα προβλήματα

Ένα εμπροσθοτροφοδοτούμενο δίκτυο έχει μια πολυεπίπεδη δομή. Κάθε επίπεδο αποτελείται από μονάδες (νευρώνες) που λαμβάνουν την είσοδό τους από μονάδες στο αμέσως προηγούμενο επίπεδο και στέλνουν την έξοδό τους σε μονάδες του αμέσως επόμενου επιπέδου. Δεν υπάρχουν ενώσεις μέσα στο ίδιο επίπεδο. Οι  $N_i$  είσοδοι τροφοδοτούνται στο πρώτο στρώμα από  $N_{h,1}$  κρυφές μονάδες (πρώτο κρυφό επίπεδο). Η ενεργοποίηση μιας κρυφής μονάδας είναι μια συνάρτηση  $F_i$  των εισόδων και ενός δυναμικού. Η έξοδος των κρυφών μονάδων διανέμεται στο επόμενο επίπεδο που αποτελείται από  $N_{h,2}$  κρυφές μονάδες, μέχρι το τελευταίο κρυφό στρώμα, του οποίου οι έξοδοι τροφοδοτούνται σε ένα στρώμα από  $N_o$  μονάδες εξόδου.

Παρόλο που ο αλγόριθμος Back-Propagation μπορεί να εφαρμοστεί σε δίκτυα με οσοδήποτε μεγάλο αριθμό επιπέδων, έχει δείχθει ότι μόνο ένα κρυφό επίπεδο αρκεί για να προσεγγίσουμε οποιαδήποτε συνάρτηση με όσο μεγάλη ακρίβεια θέλουμε, με την προϋπόθεση οι συναρτήσεις ενεργοποίησης των κρυφών επιπέδων να είναι μη γραμμικές. Στην πλειονότητα των περιπτώσεων χρησιμοποιείται ένα δίκτυο πρόσθιας τροφοδότησης με μόνο ένα επίπεδο από κρυμμένες μονάδες

με στιγμοειδή συνάρτηση ενεργοποίησης για τις μονάδες. Ένα δίκτυοπολλών επιπέδων φαίνεται στο επόμενο σχήμα:

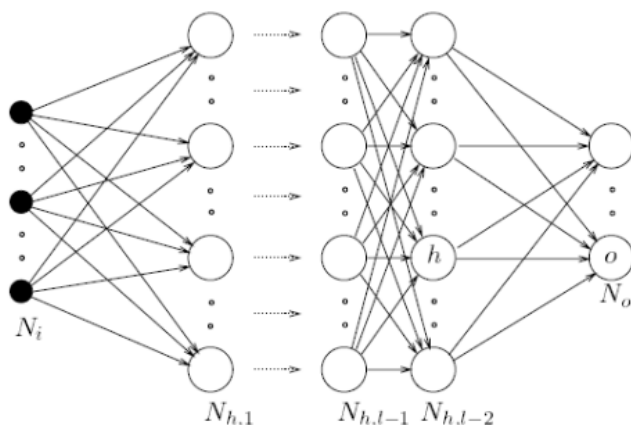


Figure 2 Multi-layer δίκτυο με  $l$  στρώματα μονάδων

### 1.5 Μέθοδος Οπισθοδιάδοσης του Λάθους (Backpropagation)

Η μέθοδος οπισθοδιάδοσης του λάθους (error backpropagation) είναι η πιο δημοφιλής μέθοδος σήμερα για την εκπαίδευση ενός πολυστρωματικού νευρωνικού δικτύου. Ιστορικά, πρώτα αναπτύχθηκαν δίκτυα ενός και δύο επιπέδων, όπως ο στοιχειώδης αισθητήρας (perceptron). Τα δίκτυα όμως αυτά γρήγορα φάνηκε ότι έχουν αρκετά περιορισμένες ικανότητες και έτσι σύντομα εγκαταλείφθηκαν.

Επομένως ακολούθησαν τα δίκτυα πολλών επιπέδων που αναπτύχθηκαν αργότερα και για τα οποία αρχικά δεν υπήρχαν θεωρητικοί τρόποι για την εκπαίδευσή τους, μέχρι που εμφανίστηκε η μέθοδος οπισθοδιάδοσης. Η μέθοδος αυτή αναπτύχθηκε ανεξάρτητα σε διάφορες παραλλαγές από τους Bryson και Ho, Werbos, Parker, αλλά προωθήθηκε από το έργο Parallel Distributed Processing: Explorations in the Microstructure of Cognition των Rumelhart και McClelland. Πρόκειται για μία τεχνική που βασίζεται σε καθαρά μαθηματική θεώρηση με αυστηρά τεκμηριωμένες αποδείξεις. Το νευρωνικό δίκτυο στο οποίο εφαρμόζεται είναι ένα δίκτυο πολλαπλών επιπέδων(πολυστρωματικό) πρόσθιας τροφοδότησης (feedforward), ενώ κάθε επίπεδο μπορεί να έχει πολλούς νευρώνες. Οι νευρώνες μέσα στο ίδιο επίπεδο δεν συνδέονται μεταξύ τους, αλλά οι νευρώνες που ανήκουν σε διαφορετικά επίπεδα συνδέονται ως συνήθως με τις γνωστές συνάψεις. Υπάρχουν λοιπόν πολλές σειρές με τα βάρη  $w$  μεταξύ των επιπέδων αυτών και όχι μία μόνο σειρά. Η καινοτομία που εισάγεται στα δίκτυα αυτά είναι ότι μπορούμε να επιφέρουμε τις κατάλληλες μεταβολές στα βάρη στα ενδιάμεσα επίπεδα, εκεί όπου δεν υπάρχει στόχος και άρα

δεν μπορεί να χρησιμοποιηθεί μια απλή τεχνική, όπως π.χ. ο κανόνας Δέλτα. Η κεντρική ιδέα της δομής και λειτουργίας τέτοιων δικτύων είναι σχετικά απλή: ένα δίκτυο ξεκινά την διαδικασία μάθησης από τυχαίες τιμές των βαρών του. Εάν δώσει λάθος απάντηση (που είναι και το πιο πιθανό), τότε τα βάρη διορθώνονται έτσι ώστε το λάθος να γίνει μικρότερο. Η ίδια διαδικασία επαναλαμβάνεται πολλές φορές έτσι ώστε σταδιακά το λάθος ελαττώνεται μέχρις ότου γίνει πολύ μικρό και ανεκτό. Στο σημείο αυτό λέμε ότι το δίκτυο έχει μάθει τα παραδείγματα που του διδάξαμε με την ακρίβεια που θέλαμε να μάθει. Το επίπεδο μεταξύ του σήματος εισόδου και του επιπέδου εξόδου ονομάζεται κρυμμένο επίπεδο και μπορεί να δημιουργήσει την εσωτερική αναπαράσταση των σημάτων εισόδου. Ο λόγος που ονομάζουμε το επίπεδο αυτό κρυμμένο, είναι επειδή δεν “βλέπει” κατευθείαν ούτε την είσοδο ούτε την έξοδο του δικτύου αλλά μόνο το εσωτερικό του.

## Κεφάλαιο - 2 Multilayer Perceptron

Το MLP είναι μία υλοποίηση Τεχνητού Νευρωνικού Δικτύου (ΤΝΔ) πολλαπλών στρωμάτων εμπρόσθιας τροφοδότησης, αποτελούμενα από μη γραμμικούς νευρώνες εσωτερικού γινομένου μαζί με τα επίπεδα εισόδου και εξόδου.

Το MLP διαθέτει τουλάχιστον ένα κρυφό επίπεδο με μη γραμμικούς νευρώνες (με συνάρτηση ενεργοποίησης, συνήθως σιγμοειδή) για να μπορεί να λύσει προβλήματα που είναι μη γραμμικά διαχωρίσιμα. Αλλιώς αν όλοι οι νευρώνες του MLP είχαν γραμμική συνάρτηση ενεργοποίησης τότε το σύνολο του Τεχνητού Νευρωνικού Δικτύου θα υλοποιούσε μία γραμμική απεικόνιση. Ο αριθμός των κρυμμένων επιπέδων είναι ένα ή περισσότερα (αυξάνοντας την πολυπλοκότητα του MLP) και εξαρτάται κάθε φορά από το πρόβλημα που θέλουμε να λύσουμε.

Έχει αποδειχθεί θεωρητικά ότι ένα MLP με τουλάχιστον ένα κρυφό επίπεδο μπορεί να προσεγγίσει με μεγάλη ακρίβεια οποιαδήποτε συνάρτηση, αυξάνοντας αναλόγως των αριθμό των νευρώνων στο κρυφό επίπεδο. Η ιδιότητα αυτή όμως δεν μας γνωστοποιεί και την αρχιτεκτονική του MLP. Έτσι μας είναι άγνωστο εξ αρχής πόσους νευρώνες θα χρησιμοποιήσουμε για ένα δεδομένο σύνολο εκπαίδευσης και είναι μέχρι σήμερα βασικό ερευνητικό ζήτημα η γνώση της αρχιτεκτονικής ενός MLP δεδομένου κάποιου συνόλου εκπαίδευσης, γενικότερα στα MLP. Τα MLP έχουν χρησιμοποιηθεί σε πάρα πολλά και μάλιστα δύσκολα προβλήματα μάθησης με επίβλεψη, σε ταξινομήσεις και προσεγγίσεις συναρτήσεων, με μεγάλη επιτυχία. Εκπαιδεύονται συνήθως χρησιμοποιώντας των αλγόριθμο οπισθοδιάδοσης του σφάλματος (error back propagation) που ουσιαστικά εφαρμόζει τη μέθοδο (βελτιστοποίησης) της καθόδου με βάση την κλίση (gradient descent) για την ελαχιστοποίηση του τετραγωνικού σφάλματος εκπαίδευσης.

Η διαδικασία της εκπαίδευσης του αλγορίθμου οπισθοδιάδοσης του σφάλματος αποτελείται από υπολογισμούς που γίνονται σε δύο περάσματα από τα επίπεδα του MLP. Το πρώτο πέρασμα γίνεται κατά την ευθεία φορά, δηλαδή από την είσοδο προς της έξοδο, και το δεύτερο ανάποδα, από την έξοδο προς την είσοδο. Έτσι το MLP μπορεί να επιλύσει προβλήματα που δεν είναι γραμμικά διαχωρίσιμα αντίθετα με το perceptron που επιλύει μόνο γραμμικά διαχωρίσιμα προβλήματα. Ουσιαστικά το MLP στις περιπτώσεις ταξινόμησης ορίζει υπερεπίπεδα στο χώρο των δεδομένων, δημιουργώντας περιοχές απόφασης από τις τομές αυτών των υπερεπιπέδων.

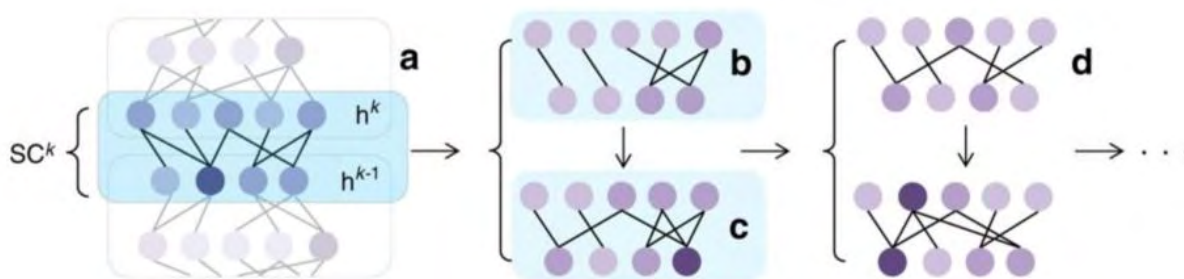


## Κεφάλαιο 3 – Σχετικές Εργασίες

Οι τεχνικές που χρησιμοποιήθηκαν στη παρούσα διπλωματική εργασία επηρεάστηκαν και εμπνεύστηκαν από εργασίες όπως η [1], [2], [3] που περιγράφονται συνοπτικά παρακάτω.

### 3.1 Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science

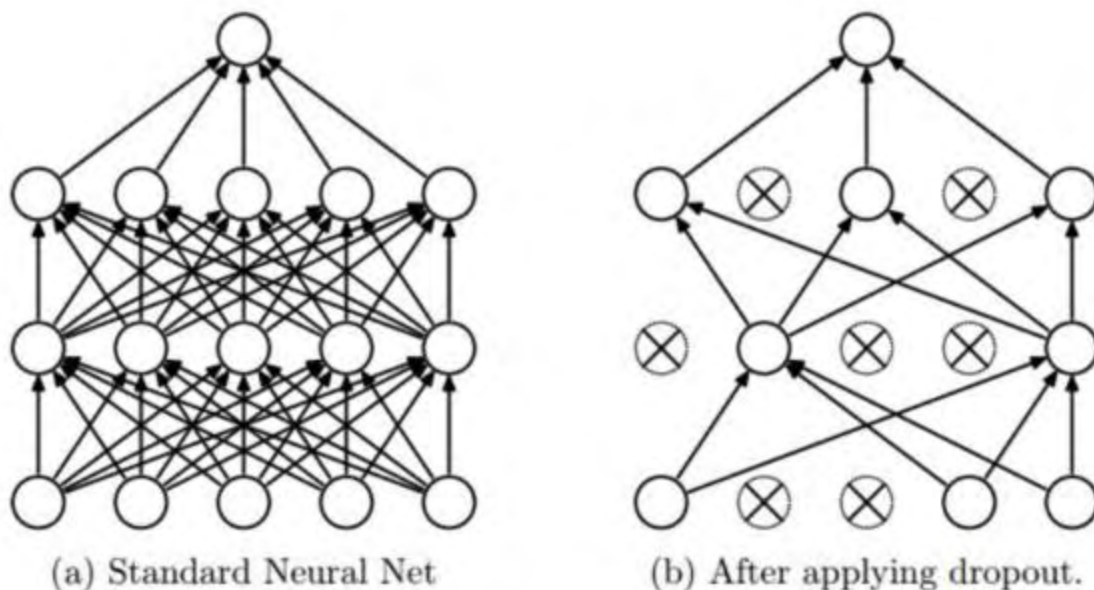
Σε αυτή την εργασία οι συγγραφείς υποστηρίζουν ότι τα τεχνητά νευρωνικά δίκτυα δεν χρειάζεται να έχουν πλήρως συνδεδεμένα επίπεδα(layers). Προτείνουν μια αραιή εξελικτική εκπαίδευση των τεχνητών νευρωνικών δικτύων(sparse evolutionary training - **SET**). Έναν αλγόριθμο ο οποίος εξελίσσει μια αρχική αραιή τοπολογία (τυχαίο γράφημα Erdős-Rényi) δύο διαδοχικών στρωμάτων(layers) νευρώνων σε μια τοπολογία *scale-free* κατά τη διάρκεια της μάθησης. Η μέθοδος αυτή αντικαθιστά τα πλήρως συνδεδεμένα στρώματα των τεχνητών νευρωνικών δικτύων με αραιά ακριβώς πριν την φάση της εκπαίδευσης μειώνοντας τετραδικά τον αριθμό των παραμέτρων χωρίς μείωση της ακρίβειας.



Στην παραπάνω εικόνα φαίνεται μία φάση της διαδικασίας SET. Για κάθε αραιό συνδεδεμένο στρώμα  $SC^k$  (a) ενός τεχνητού νευρωνικού δικτύου, στο τέλος της εκπαίδευσης μιας εποχής, ένα ποσοστό των βαρών, αυτά που είναι κοντινότερα στο μηδέν αφαιρούνται. (b)Εν συνεχεία νέα βάρη προστίθενται τυχαία ίδιου πλήθους με αυτά που αφαιρέθηκαν προηγουμένως(c). Προχωράει η εκπαίδευση με νέα εποχή (d) και η διαδικασία για την αφαίρεση και πρόσθεση βαρών επαναλαμβάνεται. Η διαδικασία συνεχίζεται για ένα πεπερασμένο αριθμό εποχών όπως συνήθως στα τεχνητά νευρωνικά δίκτυα.

### 3.2 Dropout

Τα βαθιά νευρωνικά δίκτυα με ένα μεγάλο αριθμό παραμέτρων είναι πολύ ισχυρά συστήματα μηχανικής μάθησης. Ωστόσο το overfitting είναι ένα σοβαρό πρόβλημα. Τα μεγάλα δίκτυα είναι επίσης αργά στη χρήση, κάνοντας δύσκολο το χειρισμό του overfitting συνδυάζοντας τις προβλέψεις από πολλά μεγάλα και διαφορετικά νευρωνικά δίκτυα κατά τη φάση του τεστιγκ. Το Dropout είναι μια τεχνική για την επίλυση αυτού του προβλήματος.



Η κύρια ιδέα είναι η με τυχαίο τρόπο αφαίρεση κόμβων μαζί με τις συνδέσεις τους κατά τη διαδικασία της εκπαίδευσης. Αυτό αποτρέπει τους κόμβους από το να υπερ-προσαρμοστούν στην λύση. Στην φάση της εκπαίδευσης ο Dropout δειγματοληπτεί από ένα εκθετικό αριθμό από διαφορετικά «λεπτά» δίκτυα. Στη φάση του testing, είναι εύκολο να προσεγγίσουμε την επίδραση της κατά μέσον όρο μέτρησης των προβλέψεων όλων αυτών των αραιωμένων δικτύων χρησιμοποιώντας απλά ένα ενιαίο δίκτυο που έχει μικρότερο βάρος.

Αυτό μειώνει σημαντικά το overfitting και προσφέρει σημαντικές βελτιώσεις σε σχέση με άλλες μεθόδους νομιμοποίησης(regularization). Το Dropout έχει αποδειχθεί ότι βελτιώνει την απόδοση των νευρωνικών δικτύων σε προβλήματα επιβλεπόμενης μάθησης στην όραση, την αναγνώριση ομιλίας, την ταξινόμηση εγγράφων και την υπολογιστική βιολογία, επιτυγχάνοντας κορυφαία αποτελέσματα σε πολλά σύνολα δεδομένων αναφοράς(benchmark datasets).

### 3.3 meProp: Sparsified Back Propagation for Accelerated Deep Learning with Reduced Overfitting

Οι συγγραφείς προτείνουν μια απλή αλλά αποτελεσματική τεχνική για τη μάθηση νευρωνικών δικτύων. Ο εμπρόσθιος πολλαπλασιασμός υπολογίζεται ως συνήθως. Κατά τη φάση του *backpropagation*, υπολογίζεται μόνο ένα μικρό υποσύνολο της πλήρους κλίσης για την ενημέρωση των παραμέτρων του μοντέλου. Τα διανύσματα κλίσης διαχωρίζονται με τέτοιο τρόπο ώστε να διατηρούνται μόνο τα μέγιστα- $k$  στοιχεία. Ως αποτέλεσμα, μόνο οι σειρές ή οι στήλες  $k$  (ανάλογα με τη διάταξη) του πίνακα βαρών τροποποιούνται, οδηγώντας σε γραμμική μείωση ( $k$  διαιρούμενο με τη διάσταση του διανύσματος) του υπολογιστικού κόστους. Παραδόξως, τα πειραματικά αποτελέσματα καταδεικνύουν ότι μπορούμε να ενημερώσουμε μόνο το 1-4% των βαρών σε κάθε διέλευση του *backpropagation*. Αυτό δεν έχει ως αποτέλεσμα μεγαλύτερο αριθμό

εκπαιδευτικών επαναλήψεων. Τέλος, η ακρίβεια των μοντέλων που προκύπτουν βελτιώνεται μάλλον παρά υποβαθμίζεται.

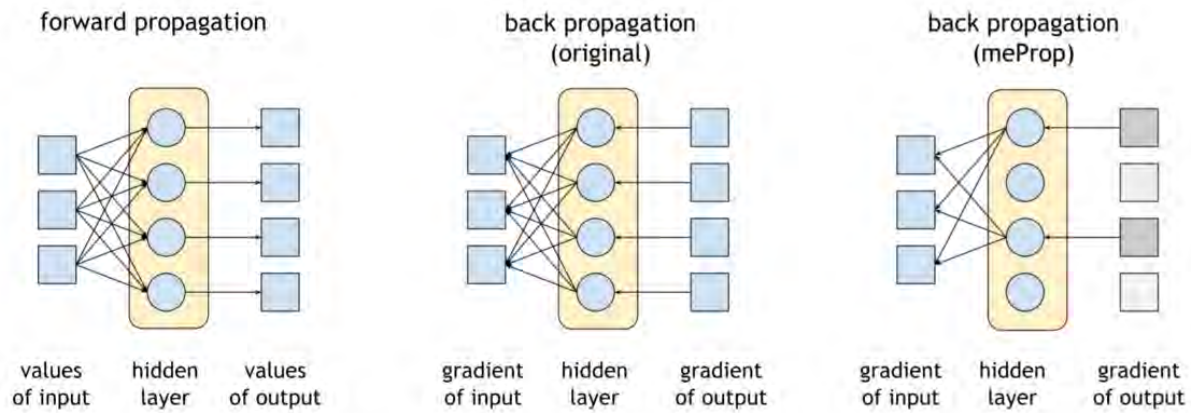


Figure 3 meProp

## Κεφάλαιο 4 – Μέθοδοι

Σε αυτό το κεφάλαιο θα δούμε τις δύο μεθόδους που προτείνονται (**DRP** και **D-NIB**) στα πλαίσια της παρούσας διπλωματικής εργασίας με σκοπό τη βελτίωση του χρόνου εκπαίδευσης του MLP αλγορίθμου. Η υλοποίηση τους βασίστηκε στη vanilla υλοποίηση του MLP του Ritchie Vink ([https://github.com/ritchie46/vanilla-machine-learning/blob/master/vanilla\\_mlp.py](https://github.com/ritchie46/vanilla-machine-learning/blob/master/vanilla_mlp.py)). Το προγραμματιστικό περιβάλλον είναι python 3.6 και χρησιμοποιήθηκαν οι βιβλιοθήκες numpy και matplotlib για την γραφική αναπαράσταση των αποτελεσμάτων.

Στα παραδείγματα αυτού του κεφαλαίου χρησιμοποιήθηκε το lung.mat dataset (<https://github.com/dcmocanu/sparse-evolutionary-artificial-neural-networks/tree/master/SET-MLP-Sparse-Python-Data-Structures/data>) που είναι και το dataset που χρησιμοποιήθηκε στην πειραματική αξιολόγηση του [1]. Για το γενικό σύνολο των πειραμάτων εκτέλεσης σε αυτή την διπλωματική χρησιμοποιήθηκε ένα υποσύνολο του MNIST dataset όπως εξηγείται στο κεφάλαιο 5.

### 4.1 Drop Random edges at fix Percentage (DRP)

#### 4.1.1 Περιγραφή DRP

Η DRP υλοποίηση και εφαρμογή σε ένα MultiLayer Perceptron μοντέλο χαρακτηρίζεται από ένα **όριο** βάση του οποίου αφαιρούνται βάρη από τον αρχικό πίνακα βαρών του MLP. Η αφαίρεση των βαρών πρακτικά σημαίνει τη μετατροπή των τιμών που θέλουμε να αφαιρεθούν σε μηδενικές. Αυτό γίνεται με άπληστο τρόπο διαλέγοντας τυχαία θέσεις του πίνακα και κάνοντάς τις μηδέν μέχρι να ικανοποιηθεί το τεθέν όριο. Αυτό είναι το στάδιο της αρχικοποίησης και το στάδιο που κοστίζει το μέγιστο για τη συνολική φάση εκπαίδευσης του αλγορίθμου. Εν συνεχεία σε κάθε εποχή αλλάζουν μόνο τα βάρη τις θέσεις των οποίων δεν είχαμε αρχικοποιήσει ως μηδέν.

Για παράδειγμα αν θέσουμε όριο το 0.9 τότε ο DRP μετατρέπει τον πίνακα των βαρών σε έναν αραιό πίνακα όπου το 90% του αποτελείται από μηδενικά. Σε μια πιο αφηρημένη θεώρηση, καταστρέφουμε το 90% των ακμών ενός multilayer perceptron και παρατηρούμε ικανοποιητική ακρίβεια για διάφορες τοπολογίες αλλά με έντονο κόστος τον χρόνο εκτέλεσης και τάξεις μεγέθους διαφορά στη loss function.

#### 4.1.2 Υλοποίηση

Στην απλή υλοποίηση κατασκευάζεται ένας πίνακας βαρών για κάθε επίπεδο(layer). Η μορφή του πίνακα στον κώδικα είναι μια δομή τύπου λεξικού(dictionary) όπου το κλειδί(key) είναι το επίπεδο και η τιμή είναι ένας numpy array με τις τιμές των βαρών. Π.χ `self.w[layer_2] = [[0.45 0.3 0. 0.1] [0.12 0.32 0.9] [...]]`

Για κάθε layer δημιουργούμε έναν νέο πίνακα(`w_removed[index]`) ο οποίος έχει τις ίδιες διαστάσεις με τον αντίστοιχο πίνακα βαρών(`self.w[layer_2].size = self.w_removed[layer_2].size`)

και αρχικοποιείται με άσσους σε όλες τις θέσεις του. Το επόμενο βήμα είναι να προσθέσουμε μηδενικά στον πίνακα με τυχαίο τρόπο ώσπου να αποτελείται κατά κάποιο fixed ποσοστό από μηδενικά.

Π.χ Στον κώδικα αρχικοποιούμε την μεταβλητή `self.threshold = 0.9` το οποίο σημαίνει ότι το 90% των τιμών του πίνακα `w_removed[layer_index]` θα μετατραπεί από άσσους σε μηδενικά. Αυτό γίνεται με ένα βίαιο τρόπο όπως φαίνεται στην παρακάτω εικόνα και εξαιτίας αυτού κοστίζει πάρα πολύ σε χρόνο το οποίο είναι και το σημαντικότερο μειονέκτημα αυτής της τεχνικής.

```
# Activations are also initiated by index. For the example we will have activations[2] and activations[3]
self.activations = {}
for i in range(len(dimensions) - 1):
    self.w[i + 1] = np.random.randn(dimensions[i], dimensions[i + 1]) / np.sqrt(dimensions[i])
    self.b[i + 1] = np.zeros(dimensions[i + 1])
    self.activations[i + 2] = activations[i]

#nsak
self.w_removed[i+1] = np.ones(self.w[i+1].shape)
print "Fixed percentage of zero weights"
print "zeros in w_removed", np.count_nonzero(self.w_removed[i+1]==0)
print "zeros to be left", self.threshold * self.w_removed[i+1].size
while np.count_nonzero(self.w_removed[i+1]==0) < self.threshold * self.w_removed[i+1].size:
    r = random.randint(0, self.w_removed[i+1].shape[0]-1)
    j = random.randint(0, self.w_removed[i+1].shape[1]-1)
    self.w_removed[i+1][r][j] = 0
```

Εν συνεχεία πολλαπλασιάζουμε τον πίνακα των βαρών με τον πίνακα `w_removed` στοιχείο προς στοιχείο ώστε αντίστοιχα το 90% των στοιχείων του πίνακα των βαρών(`self.w[layer_index]`) να γίνουν μηδέν. Αυτόν τον πολλαπλασιασμό τον κάνουμε μετά από κάθε τρέξιμο του back propagation αλγορίθμου ώστε παρόλες τις αλλαγές που επιφέρει ο backpropagation στον πίνακα των βαρών, να ξαναμετατραπεί στον πίνακα βαρών με 90% μηδενικά στις αρχικές random θέσεις και ίσως αλλαγμένο το υπόλοιπο 10% από τον back propagation(`learning rate * dw`).

```
def _update_w_b(self, index, dw, delta):
    """
    Update weights and biases.
    :param index: (int) Number of the layer
    :param dw: (array) Partial derivatives
    :param delta: (array) Delta error.
    """
    self.w[index] -= self.learning_rate * dw
    self.b[index] -= self.learning_rate * np.mean(delta, 0)
    #nsak
    #w_removed contains zeros and ones.
    self.w[index] = self.w_removed[index] * self.w[index]
```

Αυτό φυσικά λαμβάνει χώρα για κάθε epoch όπου σταθερά τα υπολειπόμενο 10% των βαρών προσαρμόζεται ώστε να μειωθεί το `loss_rate` και να φτάσουμε σε σύγκλιση.

Ο πίνακας `w_removed[layer_2]` είναι αυτής της μορφής (figure 4) όπου με μπλε φαίνονται τα μηδέν και με κόκκινο οι άσσοι. Αποτελείται από 250 στοιχεία από τα οποία τα 225 είναι μηδενικά.



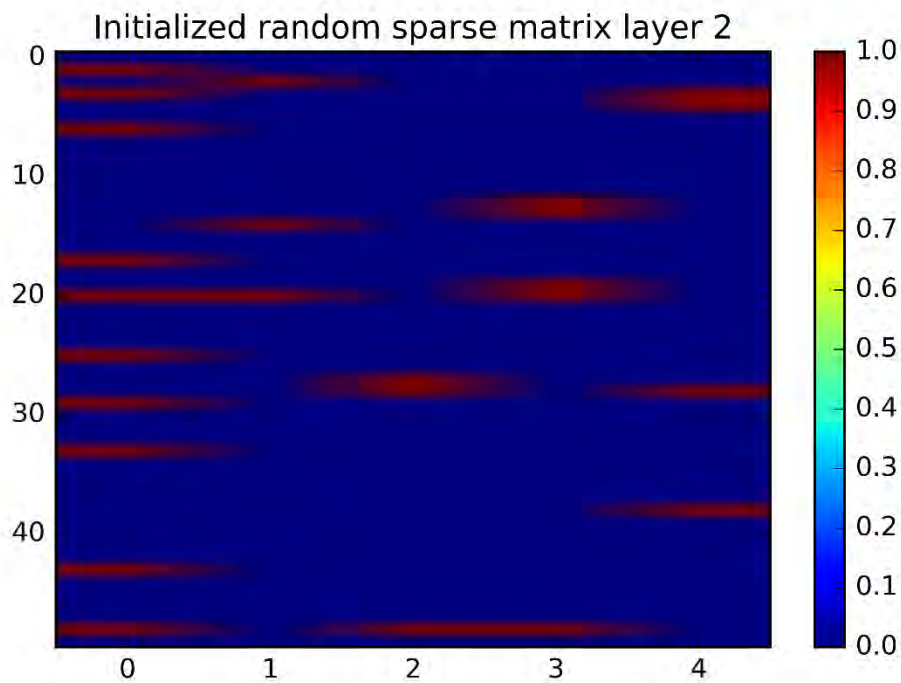


Figure 4 Πίνακας  $w\_removed[layer\_2]$

Στο Figure 3 βλέπουμε τον πίνακα ο οποίος θα πολλαπλασιαστεί στοιχείο-στοιχείο με τα βάρη που υπάρχουν ανάμεσα στο hidden layer και το output layer. Δηλαδή ο  $self.w\_removed[2]$  για τη παρακάτω τοπολογία:

```
Network((X_train.shape[1], 50, Y_train.shape[1]), ( Relu, Sigmoid))
```

```
nn.fit(x,y, loss=MSE, epochs=EPOCHS, batch_size=2, learning_rate=0.01)
```

Ο πίνακας των βαρών στη τελευταία εποχή(20) φαίνεται στην επόμενη εικόνα(figure 5)

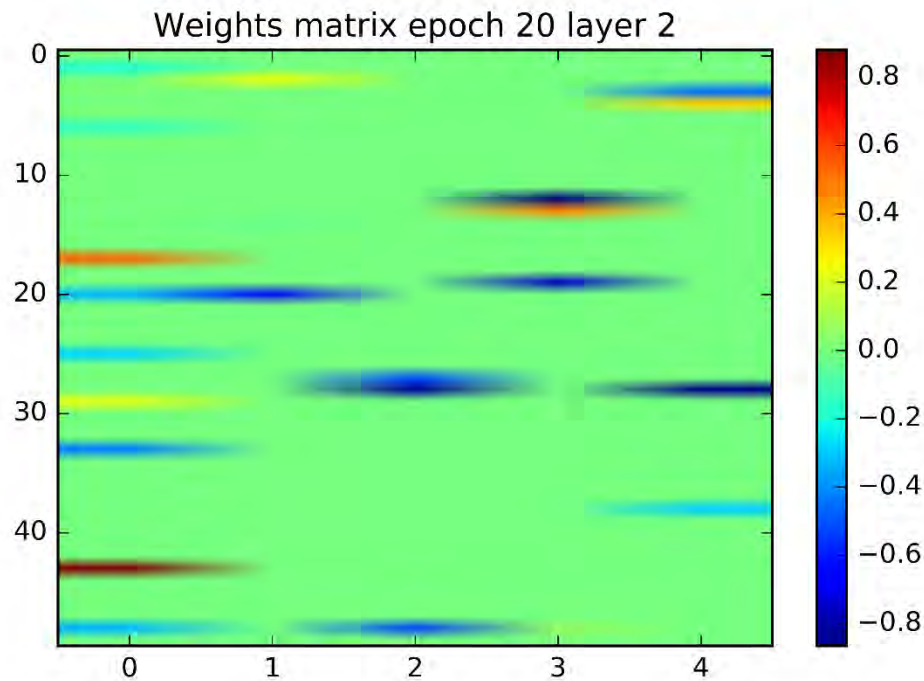


Figure 5

Προφανώς σε αυτή τη τεχνική το πλήθος των μηδενικών βαρών για κάθε επίπεδο ανά εποχή παραμένει σταθερό. Παρ όλα αυτά παραθέτουμε δυο ακόμα εικόνες(figure 6, figure 7) που δείχνουν το πλήθος των μηδενικών ανά EPOCH για σύγκριση με την τεχνική 2 που θα παρουσιαστεί στην επόμενη ενότητα. Υπενθυμίζουμε ότι μηδενίζουμε σε κάθε επίπεδο το 90% των βαρών και έχουμε ένα κρυφό επίπεδο με 50 κόμβους σε αυτή την εκτέλεση.

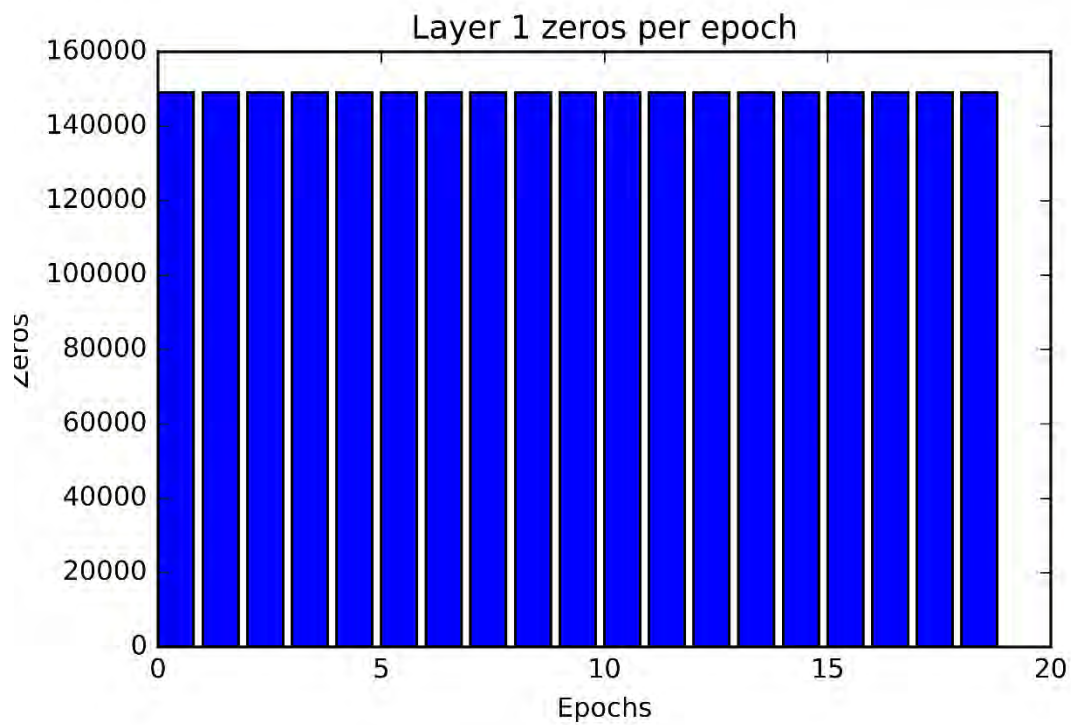


Figure 6 Πλήθος μηδενικών για το επίπεδο 1 σε κάθε εποχή

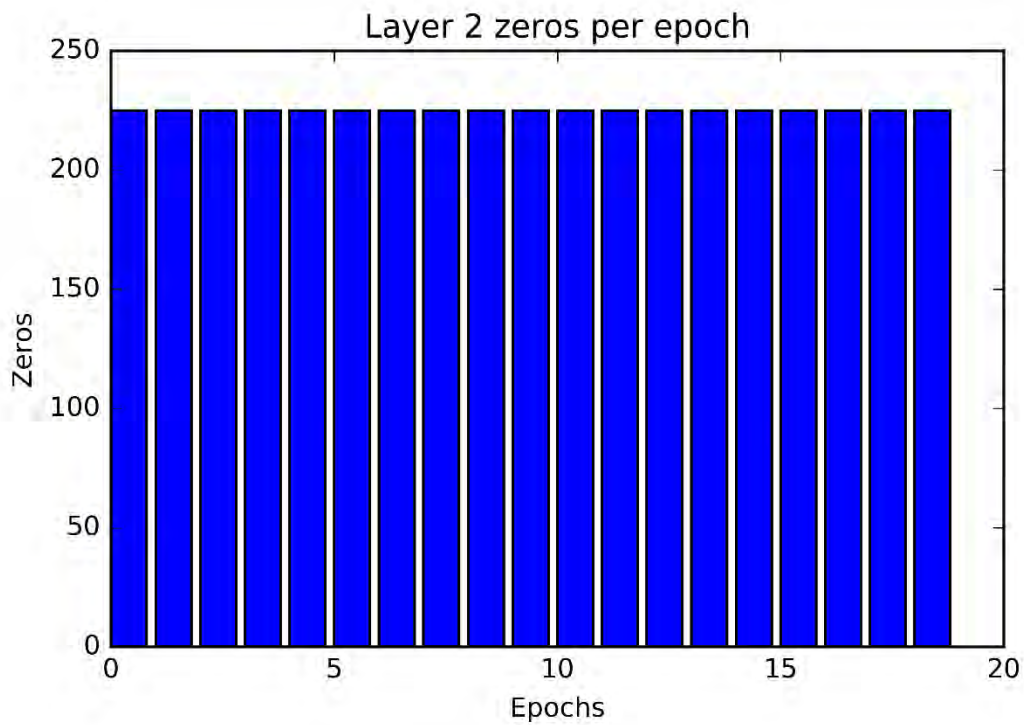


Figure 7 Πλήθος μηδενικών για το επίπεδο 2 σε κάθε εποχή



Τέλος παρακάτω φαίνονται και οι εικόνες(figure8,9) του πίνακα των βαρών( $w[\text{layer\_1}]$ ) καθώς και των άσων( $w_{\text{removed}}[\text{layer\_1}]$ ) για το πρώτο επίπεδο. Δηλαδή των βαρών που είναι ανάμεσα στο επίπεδο εισόδου και το κρυφό. Υπενθυμίζουμε ότι στη τοπολογία μας έχουμε μόνο ένα κρυφό επίπεδο των 50 κόμβων.

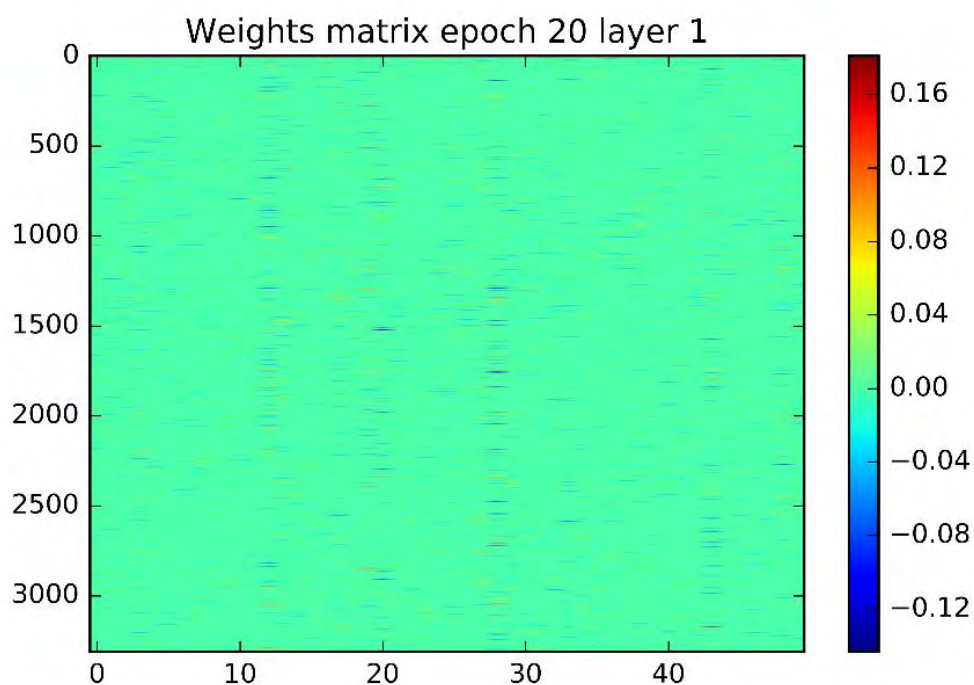


Figure 8

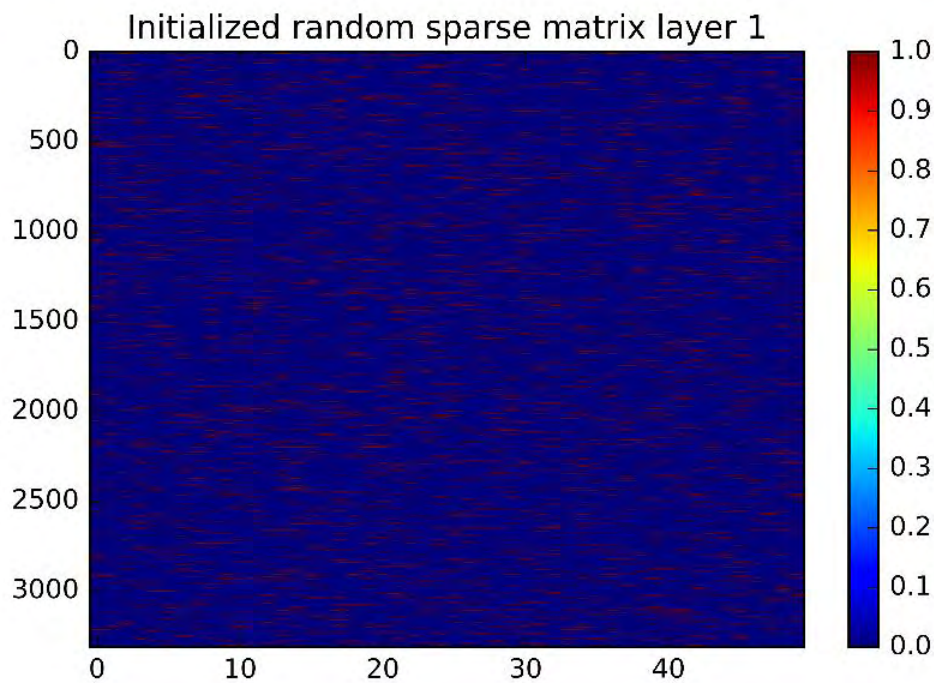


Figure 9

### Απόδοση DRP

Η απόδοση αυτού του μοντέλου είναι σαφώς καθαρά μειωμένη σε σχέση με τη *vanilla* υλοποίηση σε όρους ακρίβειας αλλά και χρόνου(λόγω της άπληστης αρχικοποίησης του πίνακα των μηδενικών) αλλά μπορούμε να παρατηρήσουμε ότι παρόλο που το 90% των βαρών είναι 0, στα 20 EPOCHS με 50 κόμβους έχουμε μεγαλύτερη από 0.9 accuracy, precision and recall. Φυσικά το Loss είναι τάξης μεγέθους μεγαλύτερο από τη *vanilla* υλοποίηση αλλά παρόλα αυτά βλέπουμε μια ικανοποιητική ακρίβεια.

Θα δούμε συγκεντρωτικά τα αποτελέσματα στο επόμενο κεφάλαιο.

## 4.2 Drop based on Number of Instances per Batch size(D-NIB)

### 4.2.1 Περιγραφή D-NIB

Η τεχνική D-NIB προέκυψε πειραματικά σε μια προσπάθεια να αφαιρεθούν ακμές του δικτύου με τυχαίο τρόπο και με μικρή πολυπλοκότητα. Σε κάθε εποχή, για κάθε επίπεδο(layer) επιλέγονται τυχαία ακμές πλήθους **αριθμός δειγμάτων/batch\_size** και αφαιρούνται. Για παράδειγμα αν έχουμε 135 δείγματα για το *training set*, επιλέξουμε παράμετρο *batch\_size*=2 και έχουμε ένα κρυφό επίπεδο τότε θα επιλεγθούν τυχαία  $135/2 = 67$  ακμές από αυτές που ενώνουν το επίπεδο εισόδου με το κρυφό επίπεδο και άλλες 67 ακμές που βρίσκονται ανάμεσα στο κρυφό επίπεδο και το επίπεδο εξόδου και θα αφαιρεθούν σε κάθε εποχή. Δηλαδή για κάθε εποχή θα μηδενιστούν μέχρι 135 νέα βάρη και στο σύνολο των εποχών μπορούν να μηδενιστούν μέχρι  $135 * \text{τον αριθμό των εποχών}$  για τη συγκεκριμένη τοπολογία.

Η λογική του αλγορίθμου αυτού είναι απλή αλλά μη διαισθητική. Στην επόμενη υποενότητα εξηγείται η υλοποίησή της.

### 4.2.2 Υλοποίηση

Όπως και στον DRP αρχικά κατασκευάζεται ένας πίνακας βαρών για κάθε layer. Η μορφή του πίνακα στον κώδικα είναι μια δομή τύπου dictionary όπου το key είναι το layer και το value είναι ένας numpy array με τις τιμές των βαρών. Π.χ *self.w[layer\_2] = [[0.45 0.3 0. 0.1] [0.12 0.32 0.9] [...]]*

Για κάθε layer δημιουργούμε έναν νέο πίνακα(*w\_removed[index]*) ο οποίος έχει τις ίδιες διαστάσεις με τον αντίστοιχο πίνακα βαρών(*self.w[layer\_2].size = self.w\_removed[layer\_2].size*) και αρχικοποιείται με άσσους σε όλες τις θέσεις του. Αφού τελειώσουν και οι υπόλοιπες τυπικές αρχικοποιήσεις της vanilla υλοποίησης ξεκινάει η φάση της εκπαίδευσης.

Ο αλγόριθμος ξεκινάει κανονικά την εκπαίδευση όπως και στην *vanilla* υλοποίηση με τη διαφορά ότι σε κάθε κάλεσμα της συνάρτησης *\_back\_prop()* (αλγόριθμος *back propagation*-οπισθοδιάδοσης) εισάγεται ένα μηδενικό σε τυχαία θέση στον πίνακα *w\_removed* (ο πίνακας των άσσων) του κάθε επιπέδου. Η συνάρτηση *\_back\_prop()* θα κληθεί συνολικά σε κάθε εποχή **πλήθος\_δειγμάτων/batch\_size** φορές. Στην παρακάτω εικόνα κώδικα φαίνεται η εισαγωγή μηδενικών στον πίνακα *w\_removed* σε τυχαίες θέσεις.

```
for k, v in update_params.items(): #ta items einai 2 gia ena hidden
    #nsak begin- update a weight to zero for every update for each layer.
    #!! D-NIB
    i = random.randint(0, self.w_removed[k].shape[0]-1)
    j = random.randint(0, self.w_removed[k].shape[1]-1)
    self.w_removed[k][i][j] = 0
    #nsak end
    self._update_w_b(k, v[0], v[1])
```

Στο τέλος της εκτέλεσης της *\_back\_prop()* καλείται η *\_update\_w\_b()* η οποία ενημερώνει τον πίνακα των βαρών του κάθε layer με τα αποτελέσματα της διαδικασίας οπισθοδιάδοσης(*learning\_rate \* dw*) όπως φαίνεται και στο απόκομα κώδικα στη παρακάτω εικόνα.

```

def _update_w_b(self, index, dw, delta):
    """
    Update weights and biases.
    :param index: (int) Number of the layer
    :param dw: (array) Partial derivatives
    :param delta: (array) Delta error.
    """
    self.w[index] -= self.learning_rate * dw
    self.b[index] -= self.learning_rate * np.mean(delta, 0)
    #nsak
    #w_removed contains zeros and ones.
    self.w[index] = self.w_removed[index] * self.w[index]

```

Στο τέλος της κλήσης της και αφού έχει αλλάξει τον πίνακα των βαρών για κάθε επίπεδο, πολλαπλασιάζουμε για κάθε επίπεδο τον πίνακα βαρών στοιχείο προς στοιχείο με τους αντίστοιχους πίνακες `w_removed[index]`.

Με αυτόν τον τρόπο, οι θέσεις που είχαν γίνει μηδέν στον `w_removed` γίνονται μηδέν και στον κύριο πίνακα των βαρών του MLP μοντέλου. Συνεπώς στην επόμενη φάση πρόσθιας διάδοσης, πολλά βάρη θα είναι μηδενικά. Με αυτόν τον τρόπο ευελπιστούμε να μειωθεί το κόστος της συνολικής διαδικασίας εκπαίδευσης του μοντέλου δεδομένου ότι σε κάθε εποχή ο πίνακας βαρών είναι πλουσιότερος σε μηδενικά και θεωρώντας ότι οι πράξεις με το μηδέν κοστίζουν λιγότερο.

Ας δούμε και ένα αντίστοιχο παράδειγμα με κοντινή τοπολογία όπως αυτή που εξετάσαμε στον DRP.

Δεδομένου μια τοπολογίας με ένα κρυφό επίπεδο αποτελούμενο από 100 κόμβους, 20 εποχές, `batch_size=2` και `learning_rate=0.01` θα παρουσιάσουμε τις γραφικές απεικονίσεις του πίνακα των βαρών(`self.w[layer_i]`) για κάθε επίπεδο, των αντίστοιχων πινάκων `w_removed` και γραφήματα με το πλήθος των αναιρούμενων ακμών(ή εισαχθέντων μηδενικών) ανά εποχή.

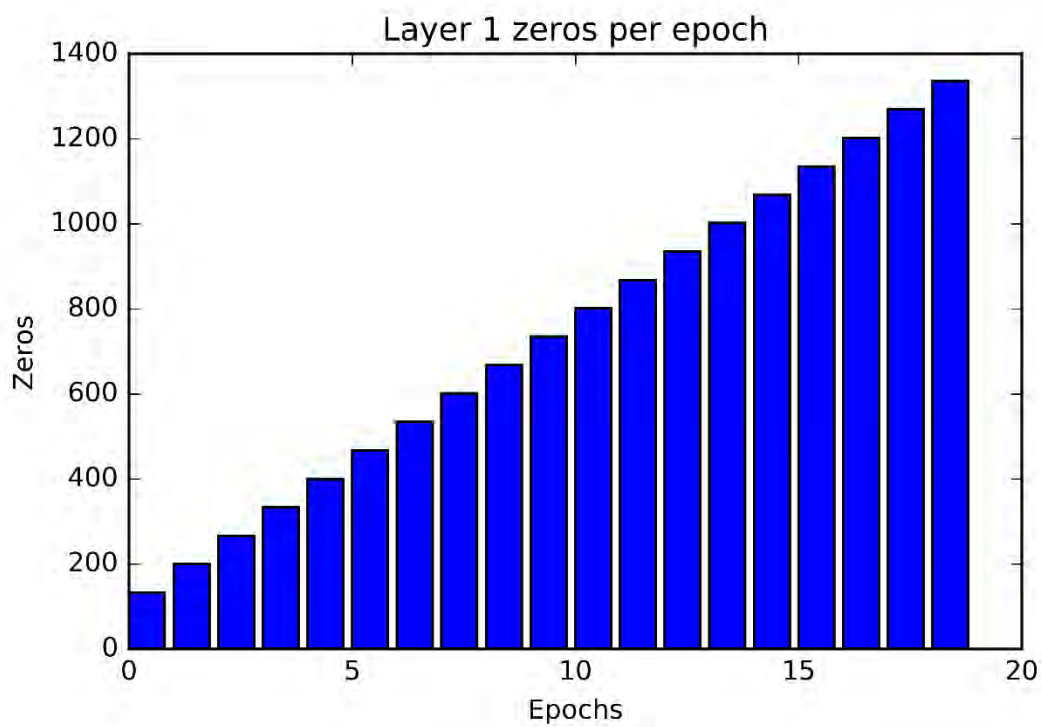


Figure 10 Πλήθος μηδενικών για το επίπεδο 1 σε κάθε εποχή D-NIB

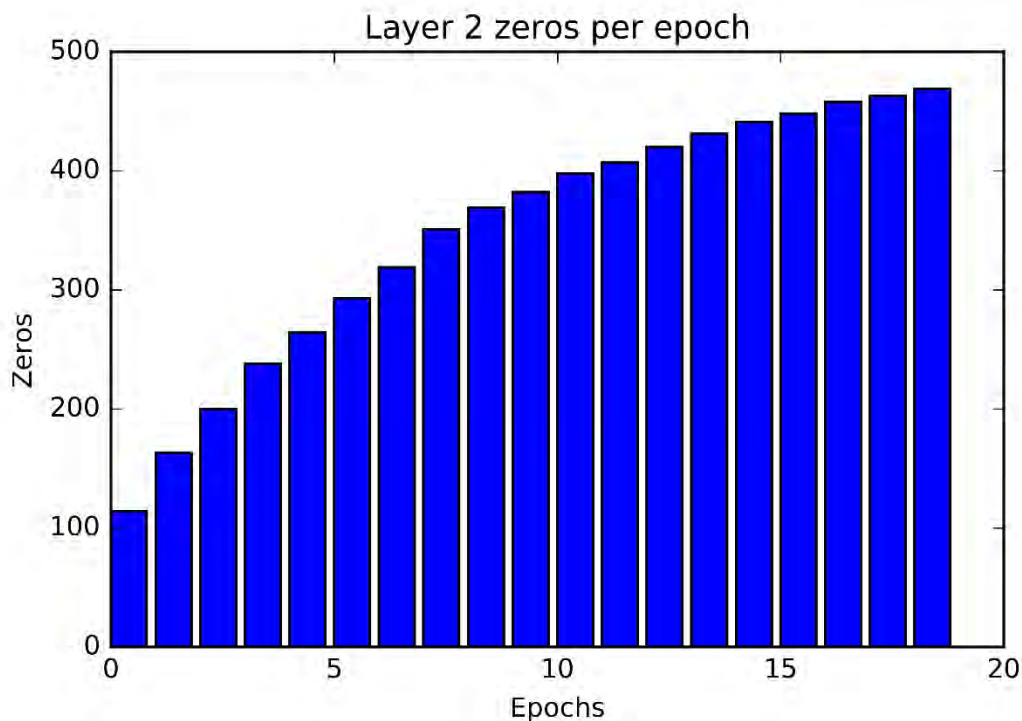


Figure 11 Πλήθος μηδενικών για το επίπεδο 2 σε κάθε εποχή D-NIB

Στη figure 10 βλέπουμε το πλήθος των εισαχθέντων μηδενικών στον πίνακα βαρών του πρώτου επιπέδου ανά εποχή. Παρατηρούμε μια γραμμικότητα στην αύξηση των μηδενικών σε αντίθεση με τον αντίστοιχο πίνακα (figure 11) για το δεύτερο επίπεδο. Αυτό οφείλεται στο γεγονός ότι κάθε επίπεδο στο συγκεκριμένο παράδειγμα σε κάθε εποχή μπορεί να αποκτήσει μέχρι 135 νέα μηδενικά στον πίνακα βαρών του. Το πρώτο επίπεδο αποτελείται από 331200 βάρη οπότε έχει μεγαλύτερη πιθανότητα σε κάθε εποχή να μπαίνουν 135 νέα μηδενικά. Αντίθετα το επίπεδο 2 αποτελείται από 500 βάρη οπότε ήδη στην 5<sup>η</sup> εποχή τα μισά του βάρη έχουν γίνει μηδέν.



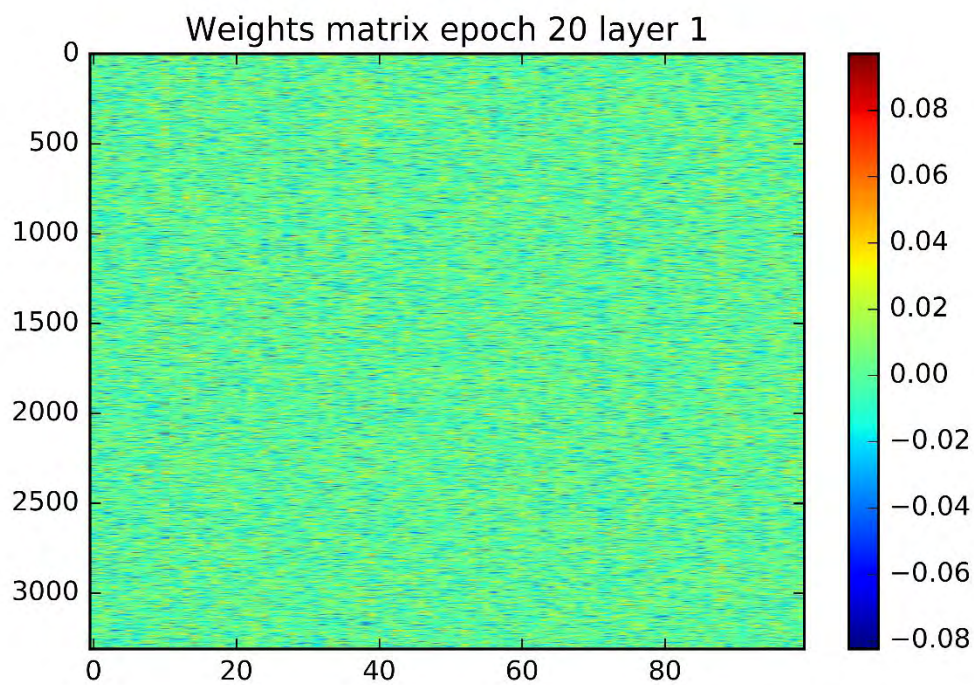


Figure 12 Πίνακας βαρών του layer1

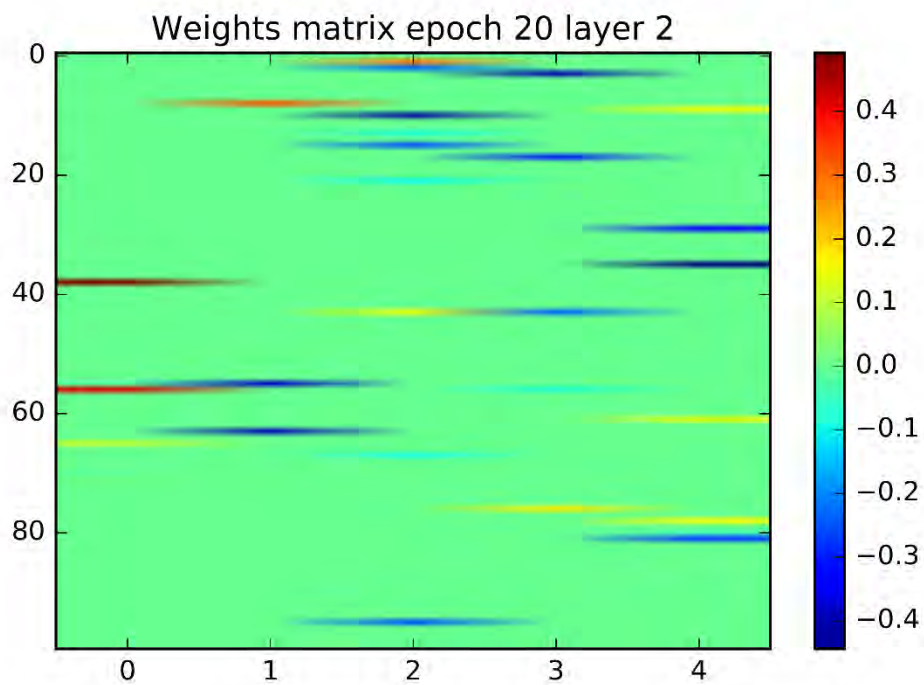


Figure 13 Πίνακας βαρών του layer2

Στις figure 12 και 13 βλέπουμε τις τιμές των πινάκων βαρών για το πρώτο και δεύτερο επίπεδο αντίστοιχα. Στην figure 1 φαίνεται ότι ο πίνακας έχει αρκετά τυχαίες τιμές με τις καθαρά μηδενικές να καταλαμβάνουν 1336 από τα 331200 βάρη. Αντίθετα ο πίνακας των βαρών του επιπέδου 2 είναι εμφανώς γεμάτος με μηδενικά με τις μη μηδενικές τιμές του να είναι μόνο 31 από τις 500.

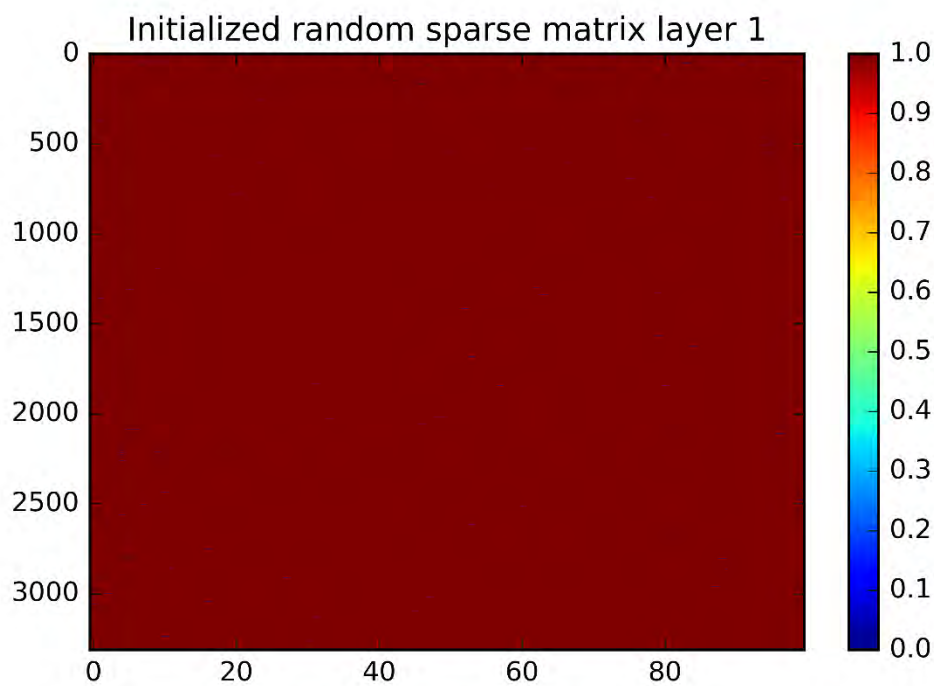


Figure 14  $w\_removed[layer1]$  περιέχει μόνο 0 ή 1



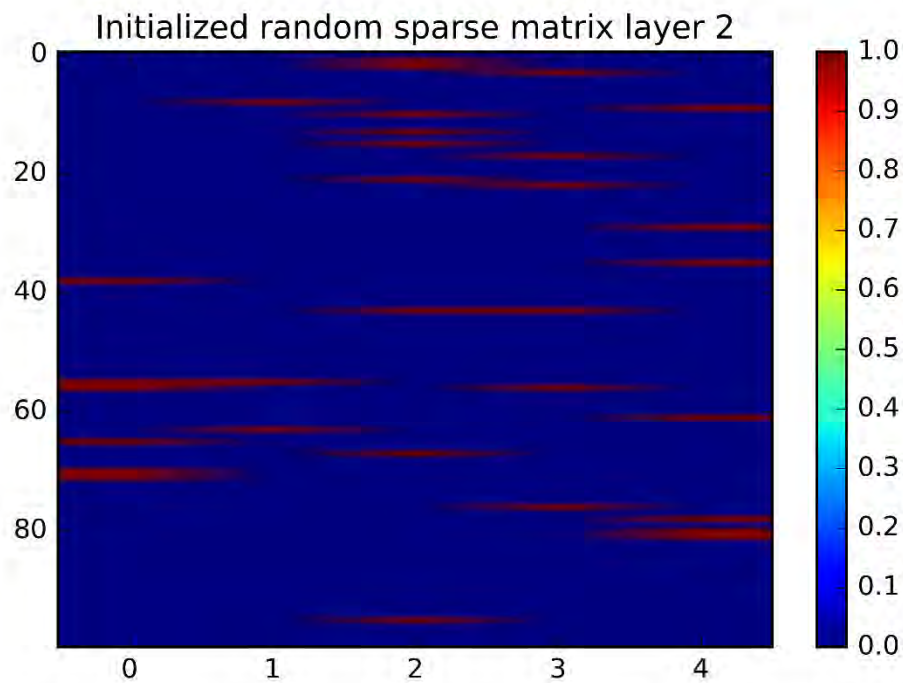


Figure 15  $w\_removed[layer2]$  περιέχει μόνο 0 ή 1

Οι figures 14 και 15 είναι οι πίνακες  $w\_removed$  που περιέχουν τιμές 0 ή 1. Κάνουν εμφανείς τις θέσεις που έχουν μηδενικά οι αντίστοιχοι πίνακες βαρών.

## Κεφάλαιο 5 – Πειραματική Αξιολόγηση

Σε αυτό το κεφάλαιο θα παρουσιαστούν οι πίνακες των αποτελεσμάτων των αλγορίθμων που προτάθηκαν σε σύγκριση με τη vanilla υλοποίηση. Για κάθε υλοποίηση (vanilla, DRP, D-NIB) εκτελέστηκε ένας αριθμός πειραμάτων για διαφορετικές τοπολογίες δικτύου στα μοντέλα και παρατίθεται ένας συγκεντρωτικός πίνακας που περιέχει την τοπολογία του δικτύου για κάθε πείραμα με τις μετρικές απόδοσης.

Χρησιμοποιήθηκαν δύο datasets. Ένα υποσύνολο του MNIST που περιέχεται στη βιβλιοθήκη sklearn.datasets και το lung dataset. Παρακάτω παρουσιάζονται τα αποτελέσματα από τη χρήση του MNIST dataset στα τρία μοντέλα. Το συγκεκριμένο dataset αριθμεί 1797 δείγματα. Το 90% αυτών χρησιμοποιήθηκαν σαν training set ενώ το υπόλοιπο 10% για testing set.

### Αποτελέσματα Digits Dataset(MNIST)

Σε αυτή την ενότητα παρουσιάζονται τα αποτελέσματα από την εκτέλεση των μοντέλων για διάφορες τοπολογίες και παραμέτρους. Για κάθε μοντέλο παρουσιάζεται αρχικά ένας πίνακας με τα αποτελέσματα των πειραμάτων και ακολουθείται από ένα γράφημα που δείχνει τους χρόνους εκτέλεσης των μοντέλων για κάθε τοπολογία.

#### *Επεξήγηση στηλών των πινάκων*

Ο κάθε πίνακας διαθέτει 10 στήλες. Η πρώτη στήλη περιέχει τον αριθμό των επιπέδων για τη συγκεκριμένη τοπολογία. Η δεύτερη στήλη περιέχει τον αριθμό των κόμβων για κάθε επίπεδο. Αν για παράδειγμα η στήλη #Nodes περιέχει το 1000,500,500 σημαίνει ότι έχουμε 3 κρυφά επίπεδα με 1000 κόμβους το πρώτο, 500 το δεύτερο και 500 το τρίτο. Η Τρίτη στήλη περιγράφει τον αριθμό των μηδενικών βαρών που έχει το κάθε επίπεδο. Η τέταρτη στήλη περιγράφει τον συνολικό χρόνο εκπαίδευσης του μοντέλου, η Πέμπτη τον χρόνο αρχικοποίησης του μοντέλου, η έκτη τον χρόνο εκπαίδευσης και η έβδομη τον χρόνο που κάνει το μοντέλο αφού έχει εκπαιδευτεί για να κάνει μία πρόβλεψη κατηγοριοποίησης. Η 8<sup>η</sup> περιέχει το loss κάθε μοντέλου ανά 10 εποχές. Για παράδειγμα αν περιέχει τις τιμές [0.0366,0.0316] τότε το μοντέλο έτρεξε για 20 εποχές όπου το σφάλμα στη 10<sup>η</sup> εποχή είναι 0.0366 ενώ στην 20<sup>η</sup> 0,0316. Τέλος η 9<sup>η</sup> στήλη περιέχει τις μετρικές precision, recall, f1\_score και η δέκατη το πλήθος των εποχών του μοντέλου.

Πίνακας 1 – Digits Dataset DRP

L	N	#Zeros/Layer	Total_time	Init_time	Fitting Time	Prediction_Time	Loss	Average precision,recall,f1_score	#epochs
1	50	2880,450	4.379	0.156	4.222	0.01563	[0.0366,0.0316]	[0.84, 0.83, 0.83]	20
1	100	5760,900	5.282	0.337	4.945	0.01562	[0.0275, 0.0241]	[0.89, 0.88, 0.88]	20
2	100,100	5760,9000,900	11.640	0.981	10.658	0.03125	[0.0123, 0.0073]	[0.97, 0.97, 0.97]	20
2	500,100	28800,45000,900	49.863	8.828	41.034	0.06685	[0.0048, 0.0025]	[0.99, 0.99, 0.99]	20
2	500,500	28800,225000,4500	928.949	447.397	481.551	0.387	[0.0051, 0.0019]	[0.99, 0.99, 0.99]	20
3	1000,500,500	57600, 450000, 225000, 4500	3101.7840	2156.979	944.804	0.857	[0.0042, 0.0010]	[1.0, 1.0, 1.0,]	20
3	1000,1000,1000	57600, 900000, 900000, 9000	10187.212	8590.682	1596.530	1.171	[0.0039, 0.0012]	[1.0, 1.0, 1.0]	20
1	50	2880, 450	12.810	0.245	12.565	0.011	[0.0498, 0.0475, 0.0464, 0.0458]	[0.76, 0.76, 0.76]	40
1	100	[5760, 900]	14.766	0.407	14.359	0.015	[0.0270, 0.0235, 0.0235, 0.0224]	[0.9, 0.9, 0.9]	40
2	100, 100	5760, 9000, 900	26.510	1.059	25.451	0.028	[0.0182, 0.0097, 0.0094, 0.0057]	[0.98, 0.98, 0.98]	40
2	500, 100	28800, 45000, 900	160.364	12.759	147.605	0.086	[0.0060, 0.0025, 0.0010, 0.0004]	[1.0, 1.0, 1.0]	40
2	500, 500	28800, 225000, 4500	771.188	294.202	476.985	0.215	[0.0058, 0.0019, 0.0004, 0.0001]	[1.0, 1.0, 1.0]	40
3	1000, 500, 500	57600, 450000, 225000, 4500]	2871.192	1514.965	1356.226	0.532	[0.0033, 0.0009, 0.0002, 5.4593e-05]	[1.0, 1.0, 1.0]	40
3	[1000, 1000, 1000]	[57600, 900000, 900000, 9000]	13357.925	9847.550	3510.375391	1.301	[0.0040, 0.0012, 0.0003, 9.6489e-05]	[1.0, 1.0, 1.0]	40

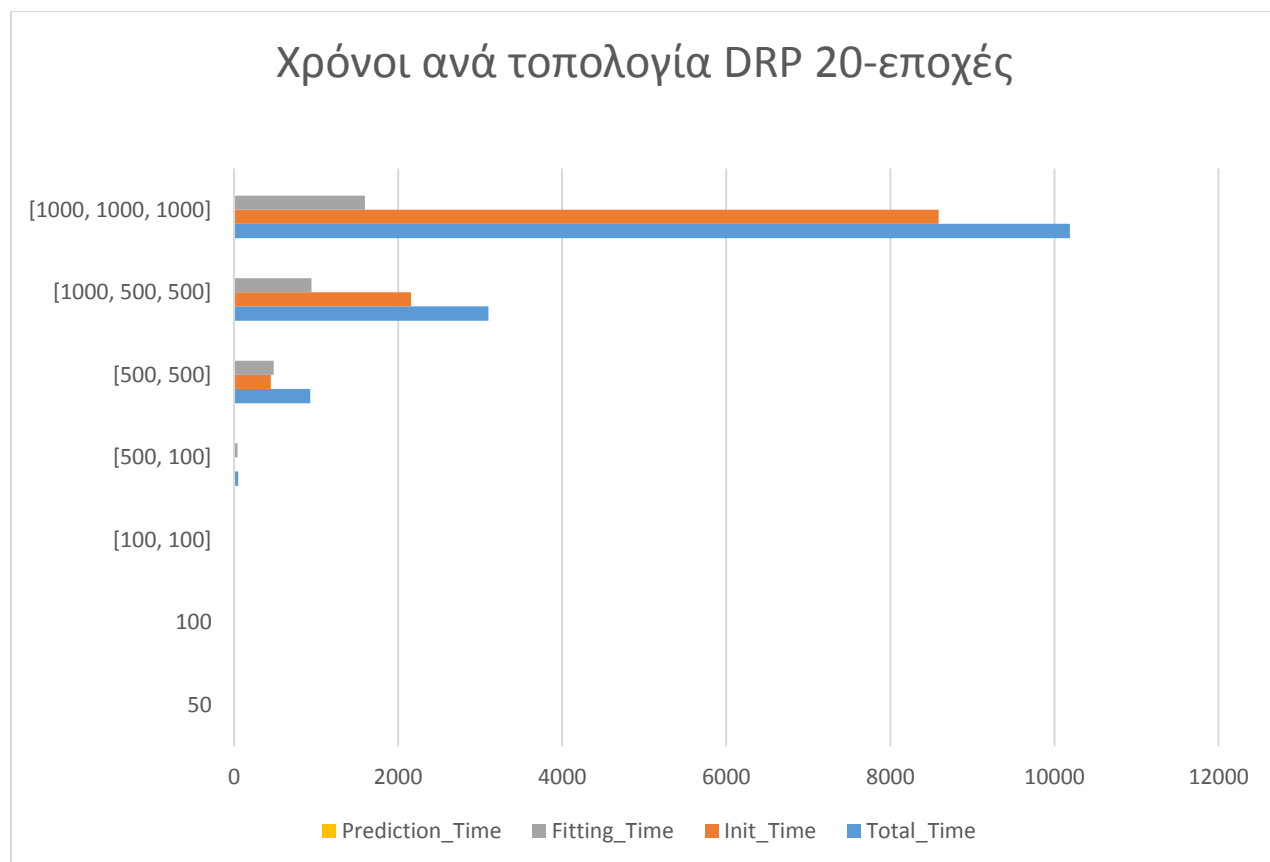


Figure 16 Χρόνοι εκτέλεσης για τεχνική DRP για 20 εποχές

Στη figure 15 παρατηρούμε τον εξωφρενικά μεγάλο χρόνο αρχικοποίησης του δικτύου ειδικά για μεγάλες τοπολογίες.

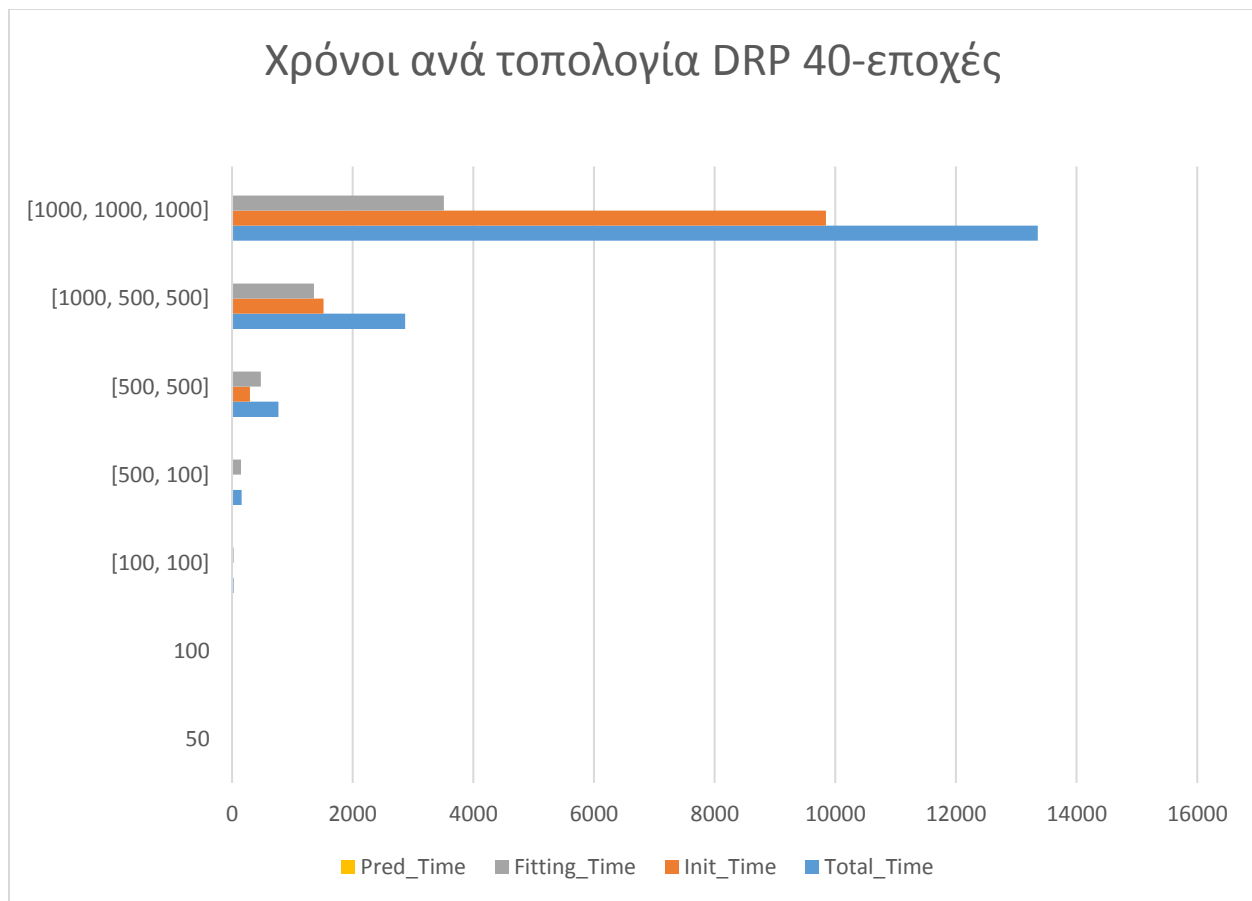


Figure 17 Χρόνοι εκτέλεσης για τεχνική DRP για 40 εποχές

Στη figure 16 βλέπουμε ότι παρόλο που τρέχει ο DRP για 40 εποχές, ο χρόνος εκπαίδευσης αυξάνεται μόνο κατά περίπου 30%. Αυτό μπορεί να είναι αποτέλεσμα των πολλών πράξεων με μηδενικά όσο περνάνε οι εποχές.

## Πίνακας 2 – Digits Dataset D-NIB

Ακολουθούν πίνακας αποτελεσμάτων του D-NIB για πλήθος 20 και 40 εποχών

L	N	Zeros per L	Total_Time	Init_Time	Fitting_Time	Pred_Time	Loss	avg(pre_rec_f1)
1	50	[3189, 500]	6.5466	0.0010	6.5456	0.0080	[0.0906, 0.0900]	[0.01, 0.1, 0.02]
1	100	[6023, 1000]	7.0183	0.0010	7.0173	0.0110	[0.0959, 0.0900]	[0.01, 0.1, 0.02]
2	[100, 100]	[6034, 8291, 1000]	13.2638	0.0020	13.2618	0.0270	[0.0953, 0.0900]	[0.01, 0.1, 0.02]
2	[500, 100]	[13729, 15065, 1000]	60.0133	0.0080	60.0053	0.0780	[0.0946, 0.0900]	[0.01, 0.1, 0.02]
2	[100, 500]	[6006, 15097, 4857]	38.6255	0.0050	38.6205	0.0630	[0.0077, 0.0135]	[0.95, 0.95, 0.95]
2	[500, 1000]	[13754, 17648, 8340]	658.7124	0.0540	658.6584	0.4711	[0.0017, 0.0004]	[1.0, 1.0, 1.0]
2	[500, 500]	[13821, 17342, 4873]	329.2914	0.0290	329.2624	0.2661	[0.0014, 0.0035]	[1.0, 1.0, 1.0]
3	[50, 100, 500]	[3188, 4842, 15111, 4881]	57.7750	0.0050	57.7700	0.0730	[0.0214, 0.0917]	[0.1, 0.18, 0.08]
3	[100, 500]	[6025, 15080, 17657, 8313]	514.9933	0.0540	514.9393	0.4151	[0.0026, 0.0054]	[0.98, 0.98, 0.98]
3	[500, 1000]	[13719, 17637, 17842, 10462]	1854.2052	0.2021	1854.0031	1.1793	[0.0002, 0.0001]	[1.0, 1.0, 1.0]
3	[500, 500]	[13833, 17335, 17628, 8310]	680.1739	0.0800	680.0939	0.5521	[0.0002, 0.0028]	[0.99, 0.99, 0.99]
3	[1000, 1000]	[15628, 17783, 17780, 8291]	1650.4858	0.1981	1650.2878	1.2273	[0.0001, 0.0001]	[1.0, 1.0, 1.0]

Figure 18 Πίνακας αποτελεσμάτων του D-NIB για πλήθος εποχών 20

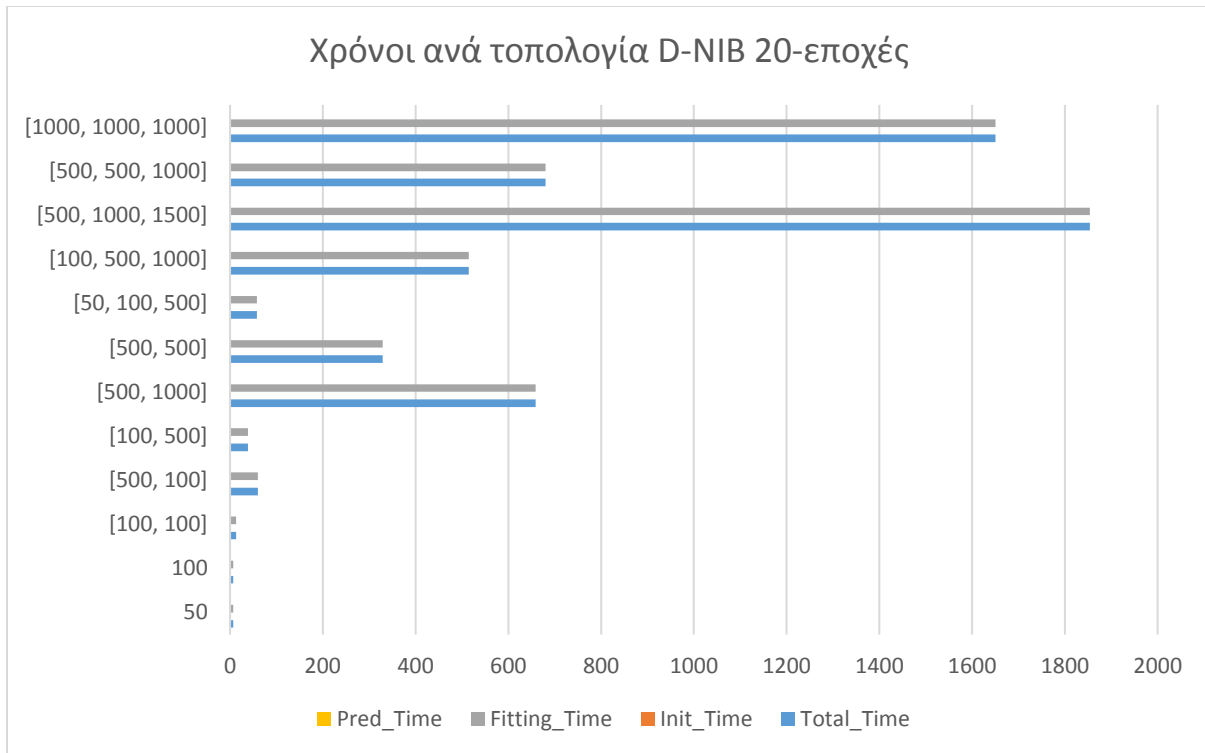


Figure 19 Χρόνοι εκτέλεσης για τεχνική D-NIB για 20 εποχές

L	N	Zeros per L	Total_Time	Init_Time	Fitting_Time	Pred_Time	Loss	avg(pre_rec_f1)
1	50	[3200, 500]	9.8859	0.0000	9.8859	0.0060	[0.0906, 0.0900, 0.0899, 0.0899]	[0.01, 0.1, 0.02]
1	100	[6377, 1000]	11.4209	0.0000	11.4209	0.0100	[0.0929, 0.0900, 0.0899, 0.0899]	[0.01, 0.1, 0.02]
2	[100, 100]	[6379, 9729, 1000]	25.1339	0.0020	25.1319	0.0200	[0.0959, 0.0900, 0.0899, 0.0899]	[0.01, 0.1, 0.02]
2	[100, 500]	[6375, 25663, 4995]	74.4889	0.0050	74.4839	0.0610	[0.0094, 0.0171, 0.0692, 0.0920]	[0.09, 0.18, 0.11]
2	[500, 100]	[21642, 25650, 1000]	86.1396	0.0070	86.1326	0.0770	[0.0935, 0.0900, 0.0899, 0.0899]	[0.01, 0.1, 0.02]
2	[500, 500]	[21648, 33426, 4996]	493.3176	0.0310	493.2866	0.2181	[0.0014, 0.0082, 0.0512, 0.0787]	[0.21, 0.3, 0.22]
3	[50, 100, 500]	[3200, 4998, 25603, 4994]	79.7435	0.0060	79.7375	0.0580	[0.0243, 0.0744, 0.0920, 0.0917]	[0.01, 0.1, 0.02]
3	[100, 500, 1000]	[6375, 25566, 34679, 9709]	1286.3355	0.0630	1286.2725	0.5479	[0.0025, 0.0059, 0.0123, 0.0587]	[0.6, 0.57, 0.55]
3	[500, 500, 1000]	[21578, 33421, 34653, 9745]	1917.5059	0.0980	1917.4079	0.7742	[0.0005, 0.0002, 0.0002, 0.0005]	[1.0, 1.0, 1.0]
3	[1000, 500, 500]	[27400, 34665, 33368, 4997]	1534.5512	0.1270	1534.4242	0.5781	[0.0003, 0.0028, 0.0341, 0.0743]	[0.32, 0.4, 0.33]
3	[1000, 1000, 1000]	[27448, 35318, 35275, 9701]	5140.9353	0.2201	5140.7152	1.3974	[9.2560e-05, 9.2166e-05, 9.5230e-05, 0.0005]	[1.0, 1.0, 1.0]

Figure 20 Πίνακας αποτελεσμάτων του D-NIB για πλήθος εποχών 40

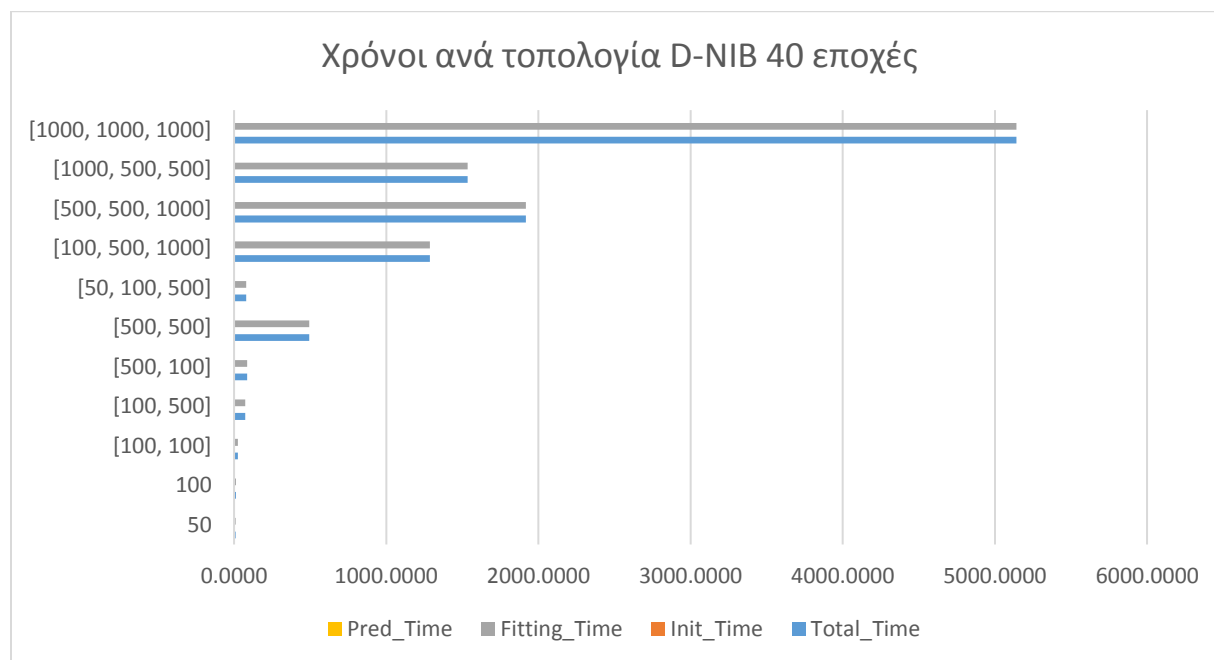


Figure 21 Χρόνοι εκτέλεσης για τεχνική D-NIB για 40 εποχές

Παρατηρούμε ότι οι χρόνοι εκτέλεσης του D-NIB είναι σαφώς μικρότεροι από ότι του DRP. Στις μεγάλες τοπολογίες βέβαια ο DRP καταφέρνει καλύτερους χρόνους εκπαίδευσης όπως φαίνεται για τη τοπολογία [1000,1000,1000] όπου ο D-NIB στις 40 εποχές χρειάζεται κάτι παραπάνω από 5000 δευτερόλεπτα ενώ αντίθετα ο DRP έχει χρόνο εκπαίδευσης 3700 δευτερόλεπτα με συνολικό όμως χρόνο εκτέλεσης πάνω από 13000.

Ενδιαφέρον επίσης παρουσιάζει η διαφορά στο χρόνο εκτέλεσης των 2 τοπολογιών [1000,500,500],[500,500,1000] καθώς και ίσως και της [100,500,1000] σε σύγκριση με τις άλλες του D-NIB στις 40 εποχές.

### Πίνακας 3 – Digits Dataset Vanilla

Ακολουθούν οι πίνακες αποτελεσμάτων για τη vanilla υλοποίηση

N	Zeros per L	Total_Time	Init_Time	Fitting_Time	Pred_Time	Loss	avg(pre_rec_f1)
50	[0, 0]	4.7772	0.0000	4.7772	0.0080	[0.0011, 8.8430e-05]	[1.0, 1.0, 1.0]
100	[0, 0]	5.7474	0.0010	5.7464	0.0120	[0.0009, 3.1382e-05]	[1.0, 1.0, 1.0]
[100, 100]	[0, 0, 0]	12.2701	0.0020	12.2681	0.0260	[0.0001, 1.1356e-05]	[1.0, 1.0, 1.0]
[500, 100]	[0, 0, 0]	40.7568	0.0080	40.7488	0.0800	[1.6954e-05, 2.2188e-06]	[1.0, 1.0, 1.0]
[500, 500]	[0, 0, 0]	191.5687	0.0313	191.5375	0.2085	[8.1135e-06, 1.2961e-06]	[1.0, 1.0, 1.0]
[1000, 500, 500]	[0, 0, 0, 0]	584.1837	0.0781	584.1056	0.5433	[1.7334e-06, 3.0768e-07]	[1.0, 1.0, 1.0]
[1000, 1000, 1000]	[0, 0, 0, 0]	1913.6559	0.1849	1913.4710	1.7004	[1.0688e-06, 2.2253e-07]	[1.0, 1.0, 1.00]

Figure 22 Πίνακας αποτελεσμάτων vanilla υλοποίησης για πλήθος εποχών 20

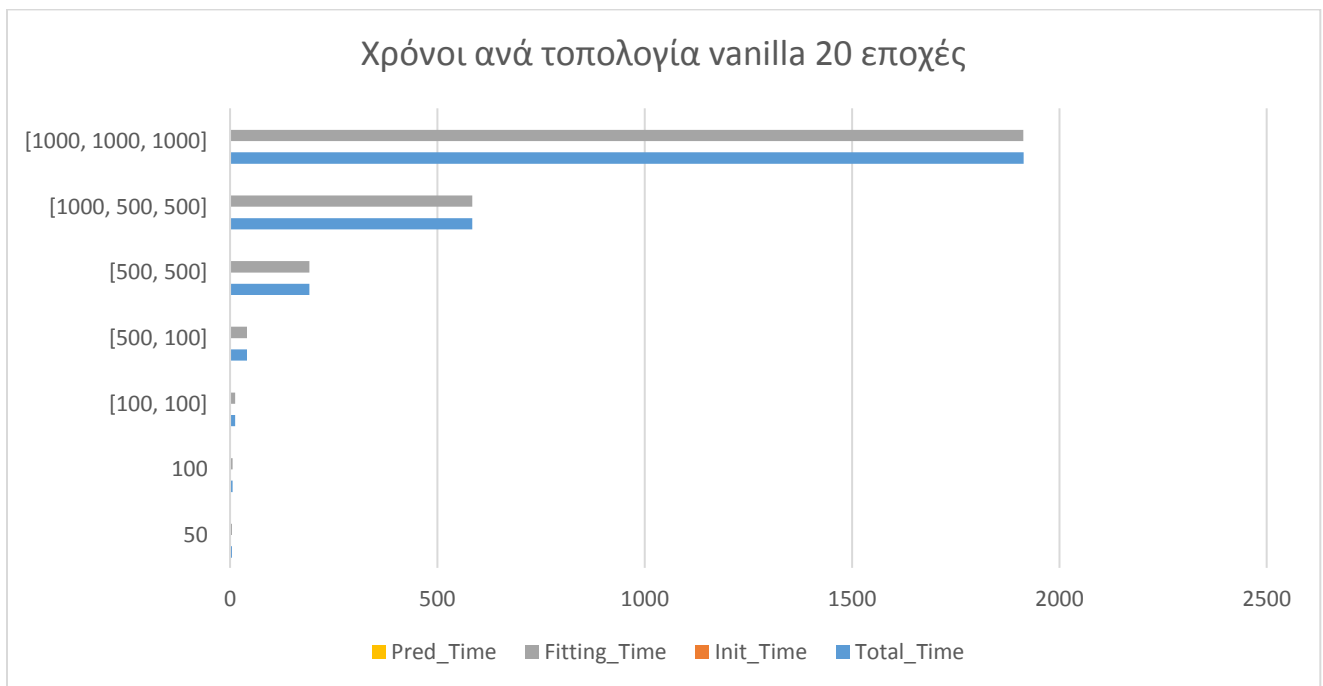


Figure 23 Χρόνοι εκτέλεσης για vanilla υλοποίηση για 20 εποχές

N	Zeros per L	Total_Time	Init_Time	Fitting_Time	Pred_Time	Loss	avg(pre_rec_f1)
50	[0, 0]	9.6159	0.0000	9.6159	0.0000	[0.0011, 8.8430e-05, 1.9414e-05, 7.3749e-06]	[1.0, 1.0, 1.0]
100	[0, 0]	9.9248	0.0000	9.9248	0.0101	[0.0009, 3.1382e-05, 9.1262e-06, 5.8163e-06]	[1.0, 1.0, 1.0]
[100, 100]	[0, 0, 0]	16.3156	0.0000	16.3156	0.0202	[0.0001, 1.1356e-05, 3.9013e-06, 1.5116e-06]	[1.0, 1.0, 1.0]
[500, 100]	[0, 0, 0]	105.4392	0.0000	105.4392	0.0960	[1.6954e-05, 2.2188e-06, 8.7237e-07, 4.6312e-07]	[1.0, 1.0, 1.0]
[500, 500]	[0, 0, 0]	618.1071	0.0340	618.0731	0.3151	[8.1135e-06, 1.2961e-06, 5.1157e-07, 2.7909e-07]	[1.0, 1.0, 1.0]
[1000, 500, 500]	[0, 0, 0, 0]	1580.5094	0.1300	1580.3794	0.6482	[1.7334e-06, 3.0768e-07, 1.2413e-07, 6.8229e-08]	[1.0, 1.0, 1.0]
[1000, 1000, 1000]	[0, 0, 0, 0]	3318.6393	0.2581	3318.3813	1.3753	[1.068e-06, 2.2253e-07, 9.4106e-08, 5.0871e-08]	[1.0, 1.0, 1.0]

Figure 24 Πίνακας αποτελεσμάτων vanilla υλοποίησης για πλήθος εποχών 40



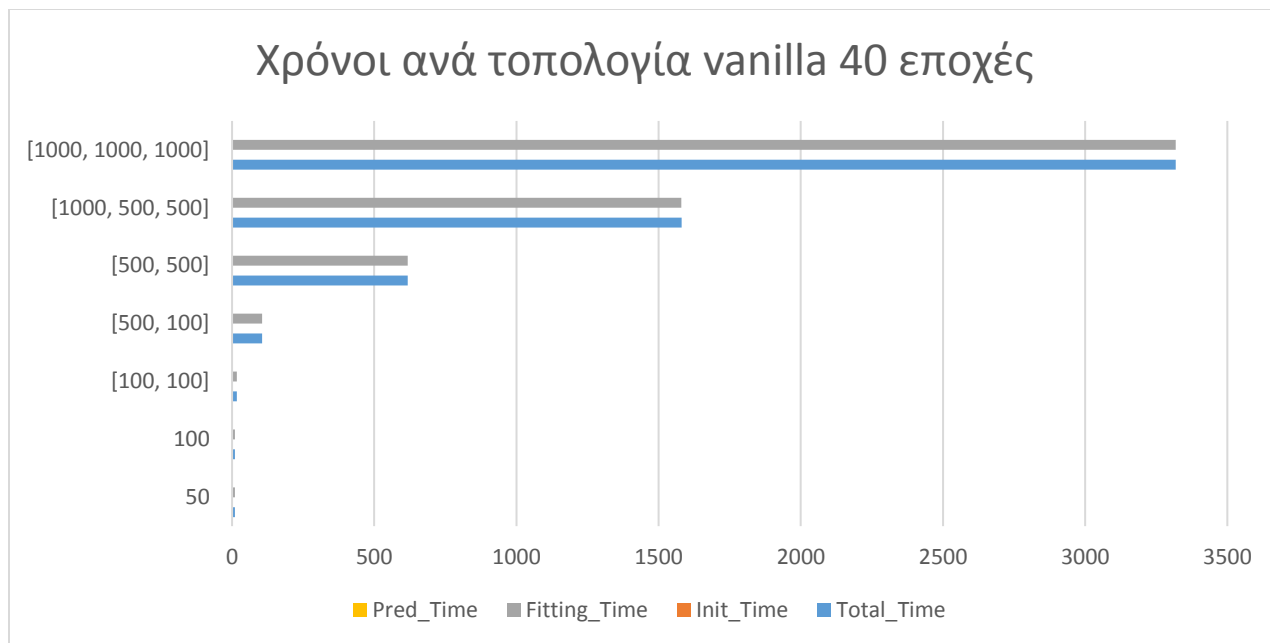


Figure 25 Χρόνοι εκτέλεσης για vanilla υλοποίηση για 40 εποχές

Αξίζει να παρατηρήσουμε εδώ ότι παρόλη τη πρόσθεση μιας μικρής βέβαια αλλά υπαρκτής πολυπλοκότητας στον κώδικα, σε κάποιες τοπολογίες βλέπουμε ο D-NIB να καταφέρνει καλύτερους χρόνους από την «καθαρή» vanilla υλοποίηση όπως για παράδειγμα στις 20 εποχές για τη τοπολογία [1000,1000,1000] όπου ο D-NIB καταφέρνει χρόνους περίπου 1630 δευτερολέπτων σε αντίθεση με τη vanilla των άνω των 18000 δευτερολέπτων. Βέβαια να σημειωθεί ότι το loss του D-NIB είναι 0.0001 ενώ του vanilla  $\sim 2.2e-7$ .

## Συμπεράσματα

Από τους 2 τρόπους που δοκιμάστηκαν φαίνεται πιο αποτελεσματικός ο D-NIB χωρίς όμως να έχουν γίνει βελτιστοποιήσεις σε επίπεδο κώδικα ώστε να καταλάβουμε τις δυνατότητες του DRP. Είναι σημαντικό και εντυπωσιακό που ο DRP καταφέρνει να κρατήσει αξιοπρεπή ακρίβεια για διάφορες τοπολογίες έχοντας μηδενικά στο 90% των βαρών του. Αυτό πρακτικά σημαίνει ότι ενδεχομένως και πολλοί κόμβοι του να μην είναι ενεργοποιημένοι.

Επίσης ελπίζαμε ότι θα παρατηρήσουμε και κάποια διαφορά στους χρόνους πρόβλεψης από τις 2 μεθόδους σε σχέση με τη vanilla λόγω των πολλών μηδενικών στη διαδικασία του *feedforward* χωρίς εν τέλει κάποιο αξιοσημείωτο αποτέλεσμα. Η μη παρατήρηση τους δεν σημαίνει απαραίτητα ότι δεν υπάρχει καθώς μπορεί να οφείλεται σε θέματα υλοποίησης της τεχνολογίας και του hardware που χρησιμοποιήσαμε (python σε συμβατικά laptop).

## Κεφάλαιο 6 - Επίλογος

Συνοψίζοντας, δύο μέθοδοι βελτιστοποίησης προτάθηκαν.

Η μέθοδος DRP μετέτρεψε το ΤΝΔ μας σε ένα πολύ αραιό δίκτυο διατηρώντας καλή ακρίβεια(>90% στη μικρότερη τοπολογία που δοκιμάστηκε) αλλά με τάξεις μεγέθους αυξημένο σφάλμα της συνάρτησης σφάλματος(Loss function) σε σχέση με τη vanilla υλοποίηση καθώς και πολύ υψηλό κόστος χρόνου. Εντυπωσιακό πάντως είναι το γεγονός ότι ο πίνακας βαρών του δικτύου αποτελείται κατά 90% από μηδενικά.

Ο D-NIB έδειξε ότι για κάποιες τοπολογίες βελτιώνεται ο χρόνος εκπαίδευσης χωρίς όμως να αυξάνεται ιδιαίτερα το σφάλμα της συνάρτησης σφάλματος διατηρώντας αντίστοιχη ακρίβεια με τη vanilla υλοποίηση.

## Βιβλιογραφία

- [1] Mocanu, D. C., Mocanu, E., Stone, P., Nguyen, P. H., Gibescu, M., & Liotta, A. (2018). Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science. *Nature Communications*, 9(1), 2383.
- [2] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1), 1929-1958.
- [3] Sun, X., Ren, X., Ma, S., & Wang, H. (2017). meprop: Sparsified back propagation for accelerated deep learning with reduced overfitting. *arXiv preprint arXiv:1706.06197*.
- Phil Simon (March 18, 2013). *Too Big to Ignore: The Business Case for Big Data*. Wiley, σελ. 89. ISBN 978-1-118-63817-0.
- Ron Kohavi; Foster Provost (1998). «Glossary of terms». *Machine Learning* 30: 271–274.
- Wernick, Yang, Brankov, Yourganov and Strother, *Machine Learning in Medical Imaging*, IEEE Signal Processing Magazine, vol. 27, no. 4, July 2010, pp. 25-38
- Mannila, Heikki (1996). «Data mining: machine learning, statistics, and databases». *Int'l Conf. Scientific and Statistical Database Management*. IEEE Computer Society.
- Friedman, Jerome H. (1998). «Data Mining and Statistics: What's the connection?». *Computing Science and Statistics* 29 (1): 3–9.
- Haykin, S. (1999) *Neural Networks: A Comprehensive Foundation*, Prentice Hall, ISBN 0-13-273350-1
- Διαμαντάρας, Κ. (2007) *Τεχνητά Νευρωνικά Δίκτυα*, Κλειδάριθμος, ISBN 9604610805
- Ματσατσίνης Ν., *Συστήματα Υποστήριξης Αποφάσεων*, Εκδόσεις Νέων Τεχνολογιών, 2010
- W. S. McCulloch and W. Pitts, A logical calculus of ideas immanent in nervous activity, *Bullettin of Mathematical Biophysics*, 5, 115(1943).
- W. Pitts and W. S. McCulloch, *Bullettin of Mathematical Biophysics*, 9, 127(1947).
- J. von Neumann, *The computer and the brain*, Yale University Press, New Haven (1958).
- D. Hebb, *The organisation of behavior*, (1949).

## URLS

[https://github.com/ritchie46/vanilla-machine-learning/blob/master/vanilla\\_mlp.py](https://github.com/ritchie46/vanilla-machine-learning/blob/master/vanilla_mlp.py)

<http://www.britannica.com/EBchecked/topic/1116194/machine-learning>

<http://neuralnetworksanddeeplearning.com>

<https://github.com/dcmocanu/sparse-evolutionary-artificial-neural-networks/tree/master/SET-MLP-Sparse-Python-Data-Structures/data>