ΠΑΝΕΠΙΣΤΗΜΙΟ ΣΤΕΡΕΑΣ ΕΛΛΑΔΑΣ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΜΕ ΕΦΑΡΜΟΓΕΣ ΣΤΗ ΒΙΟΪΑΤΡΙΚΗ

Εκπαίδευση Τεχνητών Νευρωνικών Δικτύων ανά Ομάδα Προτύπων Εισόδου

Γαλάνης Κ. Ηρακλής
ΑΜ: ΠΒ0013

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ
Υπεύθυνος
Πλαγιανάκος Βασίλειος
Επίκουρος Καθηγητής

Λαμία, 2008

Περίληψη

Σκοπός της εργασίας είναι να παρουσιαστεί η μεθοδολογία εκπαίδευσης των τεχνητών νευρωνικών δικτύων ανά ομάδα προτύπων (batch training). Σύμφωνα με αυτή τη μεθοδολογία θέλουμε να εκπαιδεύσουμε ένα τεχνητό νευρωνικό δίκτυο ως εξής: Θα θεωρήσουμε ένα σύνολο διανυσμάτων εκπαίδευσης και τα επιθυμητά αποτελέσματα. Στη συνέχεια θα τρέξουμε μια φορά όλα τα δεδομένα του συνόλου εκπαίδευσης, κι έπειτα θα υπολογίσουμε το ολικό μέσο τετραγωνικό σφάλμα. Το αποτέλεσμα του ολικού μέσου τετραγωνικού σφάλματος θα συγκριθεί με το επιθυμητό. Αυτή η διαδικασία θα επαναληφθεί έως ότου το ολικό μέσο τετραγωνικό σφάλμα να συγκλίνει στο μηδέν. Αφού θα υπολογιστεί για πρώτη φορά το ολικό μέσο τετραγωνικό σφάλμα για το σύνολο των προτύπων, θα ακολουθήσει η πρώτη τροποποίηση των βαρών. Όταν ολοκληρωθεί η διαδικασία της τροποποίησης των βαρών τότε θεωρούμε ότι τελειώνει η εκπαίδευση του τεχνητού νευρωνικού δικτύου. Επόμενο βήμα είναι η προσομοίωση του τεχνητού νευρωνικού δικτύου. Προσομοίωση είναι η διαδικασία κατά την οποία το τεχνητό νευρωνικό δίκτυο έχει εκπαιδευτεί κι απλά τρέχουμε νέα δεδομένα στο δίκτυο με σκοπό την εξαγωγή συμπερασμάτων. Ωστόσο τα τεχνητά νευρωνικά δίκτυα είναι ένα αντικείμενο προς μελέτη με ευρύ περιεχόμενο. Οπότε για να έχουμε μια σφαιρική εικόνα επί του θέματος αναφέρουμε ποιες τοπολογίες δικτύων υπάρχουν και ποιους τρόπους εκπαίδευσης γνωρίζουμε. Ένα βασικό εργαλείο για τη μελέτη των τεχνητών νευρωνικών δικτύων είναι το γνωστό λογισμικό Matlab. Οπότε και κάνουμε μια αναφορά σε κάποιες από τις συναρτήσεις της Matlab που χρησιμοποιούμε στα τεχνητά νευρωνικά δίκτυα. Τέλος ακολουθεί ένα παράδειγμα ενός τεχνητού νευρωνικού δικτύου με εκπαίδευση ανά ομάδα προτύπων γραμμένο σε κώδικα Matlab.

Λέξεις-κλειδιά: Εκπαίδευση ανά ομάδα προτύπων εισόδου, προσομοίωση, ολικό μέσο τετραγωνικό σφάλμα, όπισθεν διάδοση σφάλματος, εκπαίδευση, ρυθμός εκπαίδευσης, συνάρτηση απόδοσης σφάλματος, εποχή, σύγκλιση.

Abstract

The aim of this work is to present the methodology of artificial neural networks batch training. According to this methodology we want to train an artificial neural network as follows: We will consider a set of samples and the desirable results. Afterwards we will present all training samples and then we will calculate the mean square error. The mean square error will be compared against the desirable. This process will be repeated until the mean square error converges to zero. Then the first modification of weights follows and we consider that the training of the artificial neural network is completed. Next step is the simulation of the artificial neural network. Simulation is the process at which the trained artificial neural network is fed with new data. A tool for the study of the artificial neural network is the program Matlab. Therefore we present some functions of the Matlab software. Finally, an example of an artificial neural networks written in Matlab is presented.

key -Words: Neural networks, batch training, simulation, mean square error, training, convergence, epoch, learning rate.

Αφιερώνεται στη μνήμη του πατέρα μου Κώστα.

ΠΕΡΙΕΧΟΜΕΝΑ

1 Τεχνητά Νευρωνικά Δίκτυα

1.1 Εισαγωγή

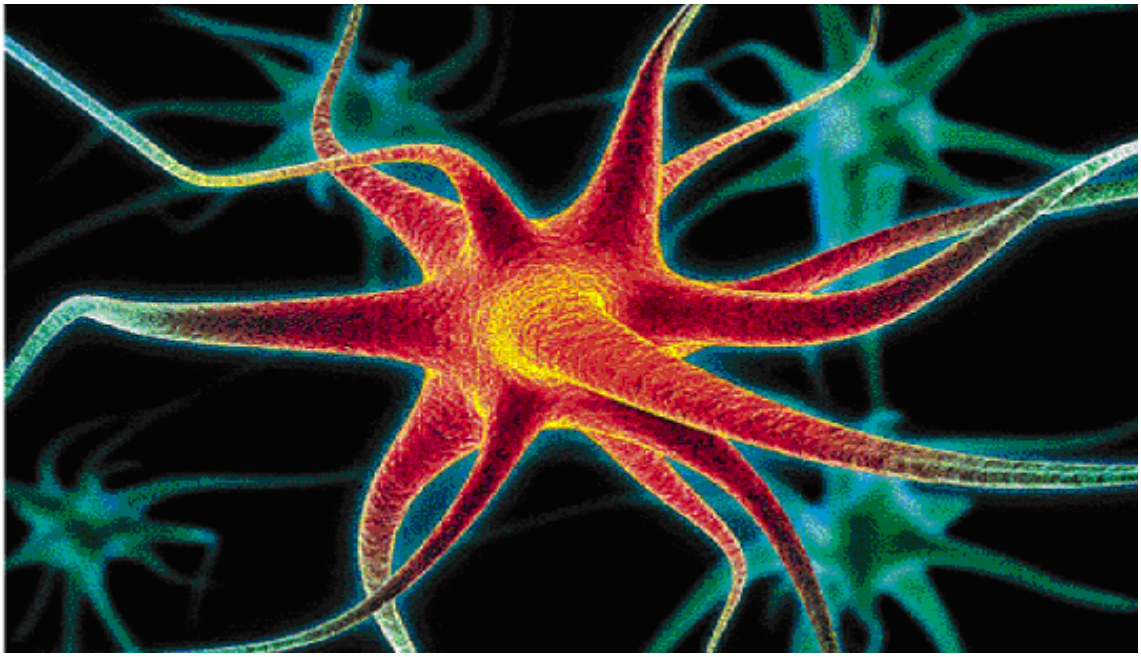
Τα τεχνητά νευρωνικά δίκτυα ως ιδέα προέκυψαν από τη δεκαετία του ’40, ωστόσο από τη δεκαετία του ’80 άρχισαν οι πρώτες μελέτες και υλοποιήσεις αυτών λόγω της προόδου που παρατηρήθηκε στο λογισμικό και στο υλικό στο τομέα των μηχανών υπολογισμού. Η παρατήρηση που ώθησε τους επιστήμονες να καλλιεργήσουν την ιδέα αυτή ήταν δύο χαρακτηριστικά του ανθρώπου. Ο άνθρωπος έχει την ικανότητα μέσω του εγκεφάλου του, να εκτελεί τους υπολογισμούς κατανεμημένα και παράλληλα, κι επιπλέον η ανάπτυξη ενός προγράμματος αντικαθίσταται από τη διαδικασία της μάθησης.

Ο όρος «τεχνητό» προκύπτει από το γεγονός ότι τα στοιχεία για να πραγματοποιηθούν οι υπολογισμοί είναι ηλεκτρονικά κι όχι βιολογικά. Κι έτσι ενώ ο υπολογιστής εκτελούσε με ακριβή δεδομένα κατά βήμα το πρόγραμμα, με τη κατάλληλη χρήση των αλγορίθμων εκπαίδευσης είναι δυνατό πλέον το τεχνητό νευρωνικό δίκτυο να εκπαιδευτεί και να προσαρμοστεί στο περιβάλλον του. Έτσι ενώ αρχικά η «προσαρμογή» παρατηρήθηκε ως χαρακτηριστικό του ανθρώπου άρχισε πλέον με μικρά αλλά πολύ σημαντικά βήματα να εφαρμόζεται στους υπολογιστές μας.

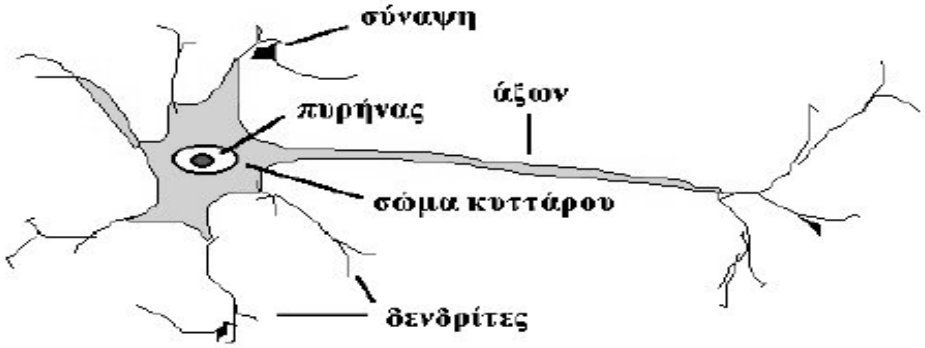
Η εκπαίδευση ενός τεχνητού νευρωνικού δικτύου περιλαμβάνει ένα καθορισμένο σύνολο δεδομένων. Βάσει αυτών των δεδομένων το δίκτυο μπορεί να εκπαιδευτεί. Το αποτέλεσμα της εκπαίδευσης θα είναι το δίκτυο αυτό να μπορεί να παράγει αποτελέσματα με μεγάλη πιθανότητα επιτυχίας. Έτσι μπορούμε να κάνουμε χρήση του δικτύου που εκπαιδεύσαμε σε παρόμοια, αλλά νέα προβλήματα. Το επίπεδο στο οποίο έχουν αναπτυχθεί μέχρι σήμερα τα τεχνητά νευρωνικά δίκτυα είναι αρκετά ικανοποιητικό και θα συνεχίσουν να αναπτύσσονται σε συνδυασμό αυτών με έμπειρα συστήματα και γενετικούς αλγορίθμους [6].

1.2 Βιολογικοί νευρώνες

Οι νευρώνες του εγκεφάλου αποτελούνται από τρία δομικά μέρη ανεξαρτήτως από το μέγεθος τους. Από το σώμα, τον άξονα και τους δενδρίτες. Οι παρακάτω εικόνες αναπαριστούν το βιολογικό νευρώνα.



Εικόνα 1.1 Μορφή βιολογικού νευρώνα



Εικόνα 1.2 Μέρη νευρώνα

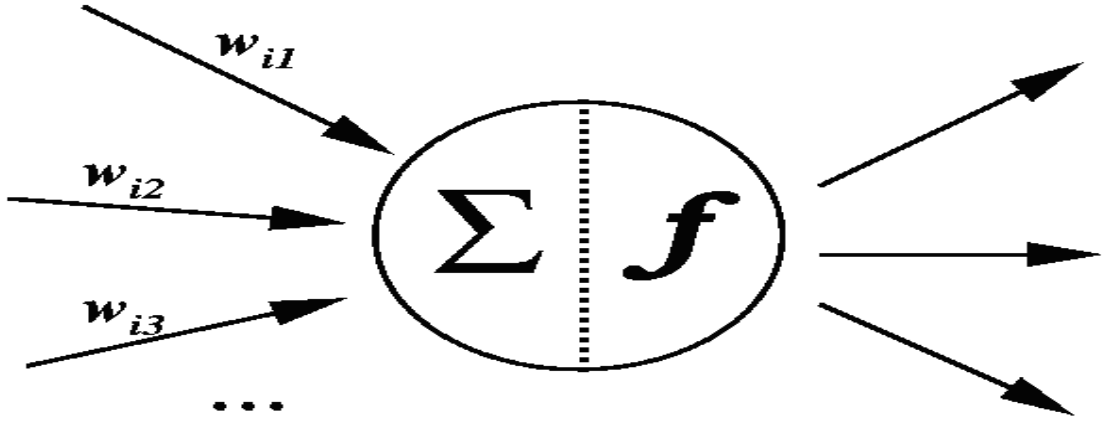
Οι δενδρίτες είναι αυτοί οι οποίοι λαμβάνουν ηλεκτροχημικά σήματα από τους άλλους νευρώνες μέσω των συνάψεων. Στη συνέχεια το σήμα μεταβιβάζεται διαμέσου του άξονα στον πυρήνα ή αλλιώς σώμα του νευρώνα. Στο σώμα εκτελείται η άθροιση των σημάτων. Αυτή η επεξεργασία έχει ως αποτέλεσμα, κάποια δεδομένη χρονική στιγμή, να παράγει ένα παλμό. Ο παλμός αυτός παράγεται όταν η τιμή του αποτελέσματος ξεπερνά μια τιμή κατωφλίου. Αφού παραχθεί ο παλμός στη συνέχεια μεταδίδεται μέσω του άξονα, των δενδριτών και των συνάψεων και στους υπόλοιπους νευρώνες.

Η εκπαίδευση του κάθε νευρώνα οφείλεται στην ηλεκτροχημική αντίσταση που έχει η κάθε σύναψη. Ο νευρώνας έχει την ικανότητα να αυξομειώνει την αγωγιμότητα των συνάψεων με αποτέλεσμα να ελέγχει ποια ερεθίσματα από τους γειτονικούς νευρώνες να περνάνε και ποια όχι. Έτσι οι διαφορές που παρατηρούνται στην αγωγιμότητα των συνάψεων και η επιλογή για το ποια σήματα τελικά θα περνάνε έχει ως αποτέλεσμα την εκπαίδευση και προσαρμογή του ανθρώπινου εγκεφάλου. Και θεωρείται ότι η γνώση έχει κατακτηθεί (από τον εγκέφαλο) όταν οι σχετικοί νευρώνες ενεργοποιούνται στα ανάλογα ερεθίσματα. Αυτός είναι και ο στόχος των τεχνητών νευρωνικών δικτύων.

1.3 Τεχνητοί νευρώνες

1.3.1 Αντιστοιχία των όρων

Όπως οι βιολογικοί έτσι και οι τεχνητοί νευρώνες αποτελούνται από το σώμα, τον άξονα και τους δενδρίτες. Η Εικόνα 1.3 αναπαριστά έναν τεχνητό νευρώνα.



Εικόνα 1.3 Τεχνητός νευρώνας

Ο κόμβος αντιστοιχεί στο σώμα του νευρώνα, οι συνδέσεις αντιστοιχούν μεταξύ του κόμβου και του άξονα και μεταξύ του κόμβου και των δενδριτών, τα βάρη αντιστοιχούν

στις συνάψεις w. Τα διανύσματα αθροίζονται, στο σχήμα η άθροιση απεικονίζεται με το γράμμα Σ, και στη συνέχεια η συνάρτηση ενεργοποίησης f παράγει ένα αποτέλεσμα, ως έξοδο του δικτύου.

1.3.2 Ορολογία και διαφορές

Στο παρακάτω πίνακα παρουσιάζεται η αντιστοιχία μεταξύ των όρων.

Πίνακας 1.1.α Ορολογία

Ορολογία βιολογίας	**Ορολογία Τεχνητών Νευρωνικών Δικτύων**
Νευρώνας (Neuron)	Μονάδα/Κόμβος/Νευρώνας (Unit/Node/Cell/Neuron)
Σύναψη (Synapse)	Σύνδεση/Σύναψη (Link/Synapse)
Αποτελεσματικότητα σύναψης (Synaptic Efficiency)	Συναπτικό βάρος (Synaptic Weight)
Συχνότητα διέγερσης (Firing Frequency)	Έξοδος κόμβου/νευρώνα (Node/Neuron Output)

Και στο πίνακα 1.1.β παρουσιάζονται οι διαφορές εγκεφάλου με υπολογιστή

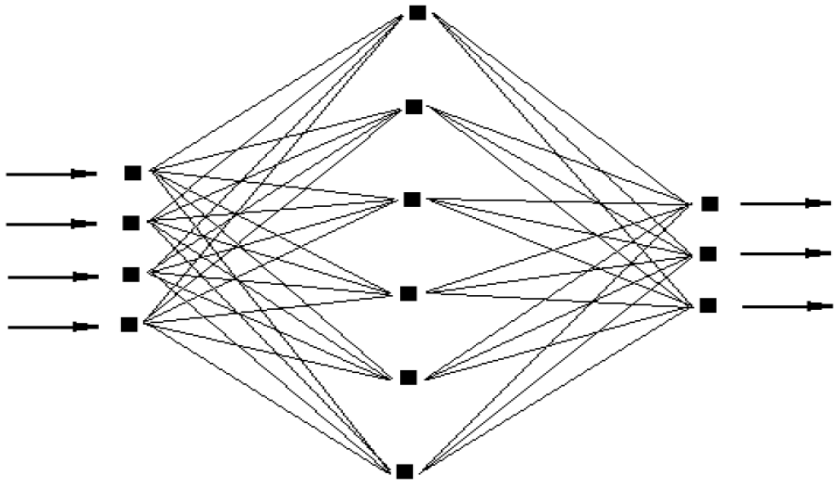
Πίνακας 1.1.β Εγκέφαλος – Υπολογιστής

Εγκέφαλος	**Υπολογιστής**
Πολλά απλά δομικά στοιχεία (νευρώνες)	Λίγοι πολύπλοκοι επεξεργαστές
Λίγα βήματα επεξεργασίας	Πολλά υπολογιστικά βήματα
Μάθηση με την εμπειρία – ικανότητα γενίκευσης	Αναλυτικός προγραμματισμός
Υψηλή συνεκτικότητα – κατανεμημένη αποθήκευση πληροφορίας	Τοπική αποθήκευση πληροφορίας
Ανεκτικότητα σε μερική καταστροφή	Αποτυχία σε περίπτωση μερικής καταστροφής

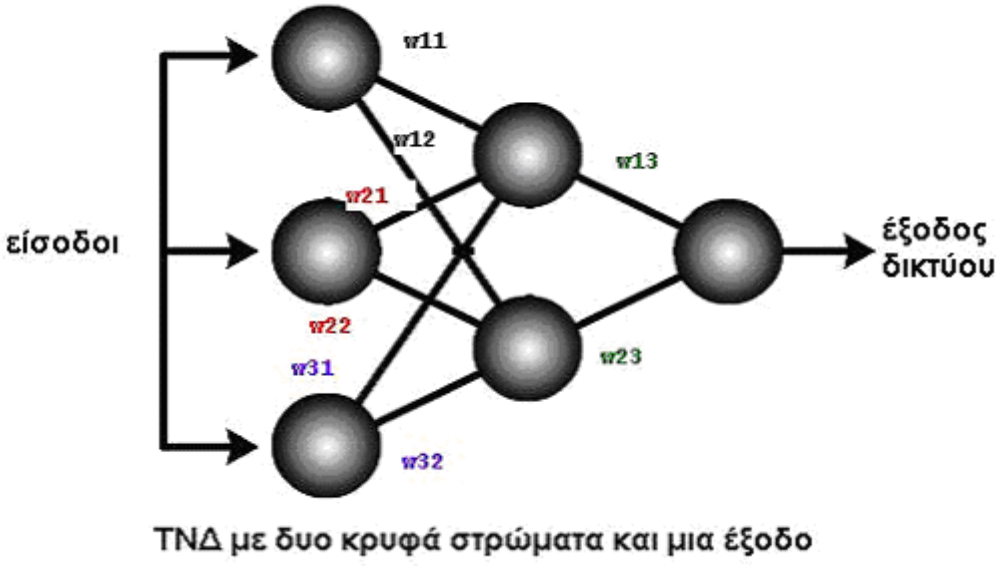
1.3.3 Ορισμός

Ένα τεχνητό νευρωνικό δίκτυο είναι ένα σύνολο διασυνδεδεμένων μονάδων που λαμβάνουν ως είσοδο ένα διάνυσμα, το οποίο σταθμίζεται με τη χρήση βαρών w και επιπλέον τροποποιείται με τη χρήση κάποιων τιμών που ονομάζονται πολώσεις b. Στη συνέχεια γίνεται χρήση μιας συνάρτησης ενεργοποίησης και βάσει της τιμής κατωφλίου παράγεται ένα αποτέλεσμα υπό τη μορφή διανύσματος. Πιο συγκεκριμένα, μεταξύ των διασυνδεδεμένων μονάδων υπάρχουν οι τιμές των βαρών που αποτελούν τη γνώση που είναι αποθηκευμένη στο δίκτυο και καθορίζουν τη λειτουργικότητα του. Η τιμή της εξόδου από κάθε μονάδα χαρακτηρίζεται από τη συνάρτηση ενεργοποίησης, τη διασύνδεση με τις υπόλοιπες μονάδες και τις τιμές των πολώσεων. Βασικό χαρακτηριστικό επίσης του τεχνητού νευρωνικού δικτύου είναι ο αλγόριθμος εκπαίδευσης που εφαρμόζεται. Συνοψίζοντας, η λειτουργικότητα του δικτύου είναι αποτέλεσμα της τοπολογίας, του αλγορίθμου εκπαίδευσης αλλά και των δεδομένων βάσει των οποίων γίνεται η εκπαίδευση. Κι επειδή ο αριθμός των νευρώνων μπορεί να είναι πολύ μεγάλος και λειτουργούν ταυτόχρονα (παράλληλα), τους δίνουν το χαρακτηριστικό του μαζικού παράλληλου προγραμματισμού.

Οι Εικόνες 1.4 και 1.5 απεικονίζουν δύο είδη τοπολογίας πολυεπίπεδων (ή πολυστρωματικών) τεχνητών νευρωνικών δικτύων.



Εικόνα 1.4 Πολυεπίπεδο ΤΝΔ



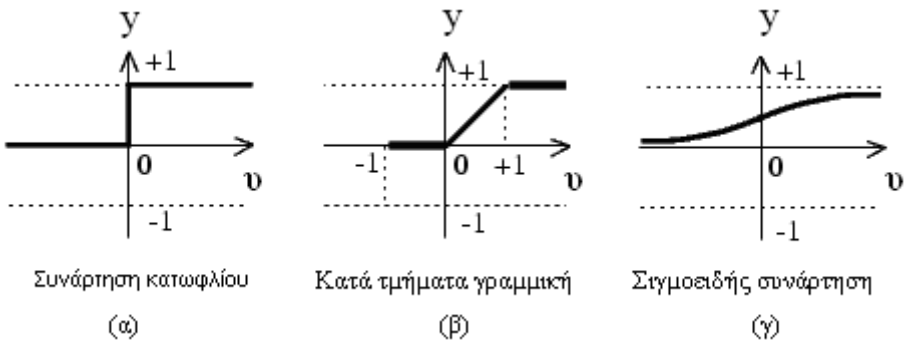
Εικόνα 1.5 Πολυεπίπεδο δίκτυο προσοτροφοδότησης

1.4 Βασικά χαρακτηριστικά των τεχνητών νευρωνικών δικτύων

Ο κάθε κόμβος υλοποιεί μια συνάρτηση ενεργοποίησης και το δίκτυο στο σύνολο του μια λειτουργία. Ο καθορισμός των τιμών στα βάρη καθορίζεται μέσω της διαδικασίας της εκπαίδευσης. Η εκπαίδευση επιτυγχάνεται με τη τροποποίηση των τιμών στα βάρη. Η γνώση, η εμπειρία και η εκπαίδευση του δικτύου αποθηκεύεται στις τιμές των βαρών των νευρώνων. Στόχος των τεχνητών νευρωνικών δικτύων είναι να αναπτύξουν μια καλή εσωτερική δομή, ώστε να αναγνωρίζουν πρότυπα, τα οποία όμως θα είναι παρόμοια με αυτά που εκπαιδεύτηκε.

1.5 Συναρτήσεις ενεργοποίησης

Η συνάρτηση ενεργοποίησης μπορεί να είναι η συνάρτηση κατωφλίου (Threshold function), η συνάρτηση κατά τμήματα γραμμική (Linear Function) ή μια σιγμοειδής συνάρτηση (Sigmoid Function). Γραφικά οι παραπάνω συναρτήσεις παρουσιάζονται στην Εικόνα 1.6.



Εικόνα 1.6 Συναρτήσεις κατωφλίωσης

Η συνάρτηση κατωφλίου (Εικόνα 1.6.α) δίνει τιμές στο διάστημα [0,1]

$$f(\upsilon)=\begin{cases}1,\upsilon\geq 0\\0,\upsilon<0\end{cases}$$

Η συνάρτηση κατά τμήματα γραμμική (Εικόνα 1.6.β) δίνει τιμές στο διάστημα [0,1]

$$f(\upsilon)=\begin{cases}1,\upsilon\geq 1\\\upsilon,0<\upsilon<1\\0,\upsilon\leq 0\end{cases}$$

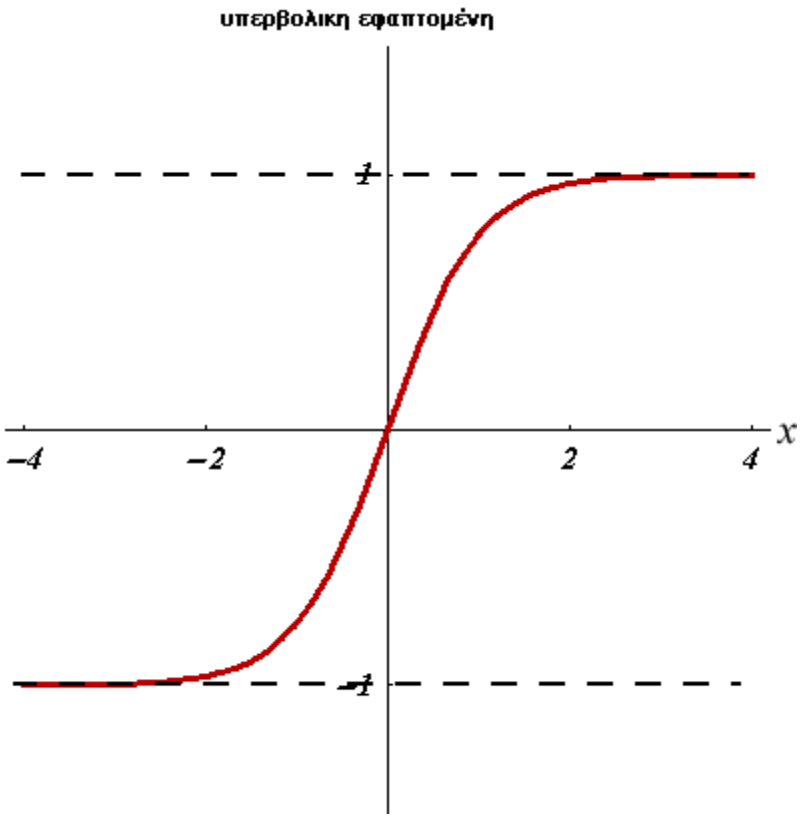
Και η σιγμοειδής συνάρτηση (Εικόνα 1.6.γ) δίνει τιμές στο διάστημα [0,1]

$f(\upsilon)=\dfrac{1}{1+\exp(-\lambda\upsilon)}$ όπου λ είναι μια παράμετρος η οποία ελέγχει τη κλίση της συνάρτησης.

Τέλος, υπάρχει και η συνάρτηση της υπερβολικής εφαπτομένης (Εικόνα 1.7)

$$f(u)=1-\exp(-u)\div 1+\exp(-u)$$

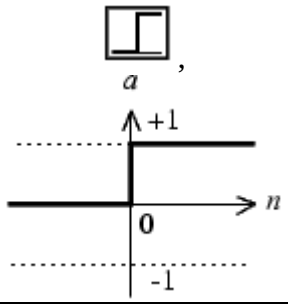
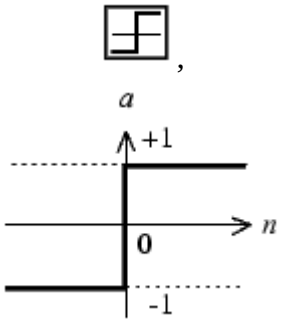
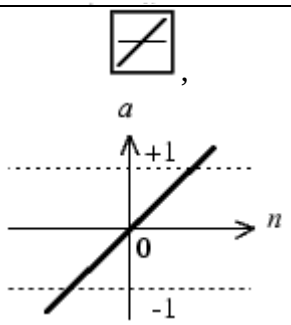
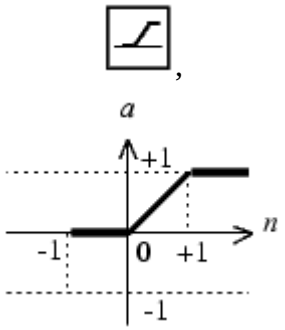
Που δίνει τιμές στο διάστημα [-1, 1]

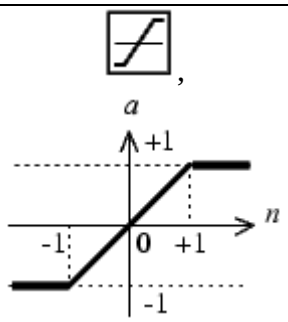
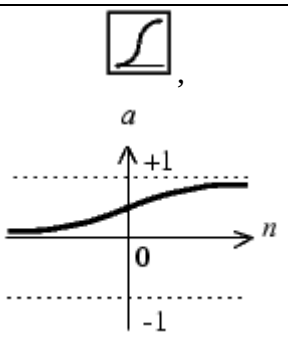
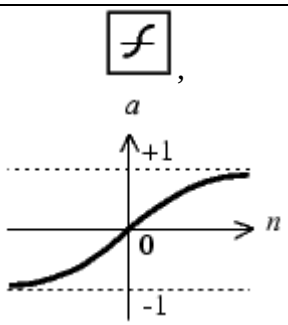
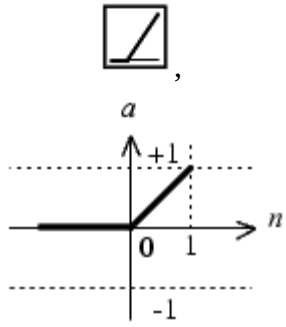
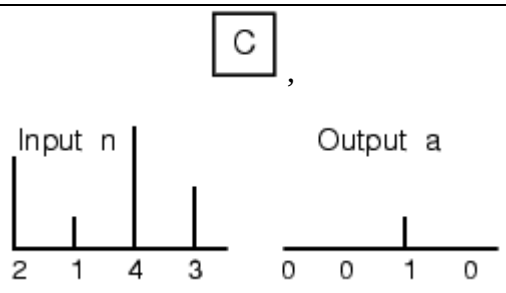


Εικόνα 1.7 Γραφική παράσταση της συνάρτησης υπερβολική εφαπτομένη

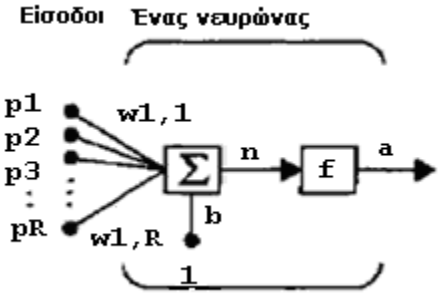
Παρακάτω ακολουθεί ο συνοπτικός Πίνακας 1.2 με κάποιες από τις συναρτήσεις ενεργοποίησης.

Πίνακας 1.2 Πίνακας συναρτήσεων ενεργοποίησης

Όνομα	Σχέση εισόδου/εξόδου	Συνάρτηση στη Matlab	Εικονίδιο/Γράφημα
Hard limit	a = 0, n < 0 a = 1, n ≥ 0	Hardlim	
Symmetrical hard limit	a = -1, n < 0 a = 1, n ≥ 0	Hardlims	
Linear	a = n	Purelin	
Saturating linear	a = 0, n < 0 a = n, 0 ≤ n ≤ 1 a = 1, n > 1	Satlin	

Symmetric saturating linear	a = -1, n < 0 a = n, -1 ≤ n ≤ 1 a = 1, n > 1	Satlins	
Log-sigmoid	a = 1 / (1 + exp(-n))	Logsig	
Hyperbolic tangent sigmoid	a = 2/(1+exp(-2*n))-1	Tansig	
Positive linear	a = 0, n < 0 a = n, n ≥ 0	Poslin	
Competitive	a = 1, ο νευρώνας με το μέγιστο n a = 0, όλοι οι υπόλοιποι νευρώνες	Compet	

Στο Πίνακα 1.2 στη πρωτελευταία στήλη αναγράφεται και η ονομασία της συνάρτησης αν θελήσουμε να εργαστούμε σε περιβάλλον Matlab. Η έξοδος σε ένα νευρώνα υπολογίζεται από τον τύπο a = f(**wp** + **b**)



Εικόνα 1.8 Απλή μορφή δικτύου

όπου f είναι η συνάρτηση ενεργοποίησης, w το διάνυσμα βαρών, p το διάνυσμα εισόδου και b η τιμή της πόλωσης.

2 Εκπαίδευση

Βασική ιδιότητα ενός τεχνητού νευρωνικού δικτύου είναι να μαθαίνει από το περιβάλλον του και να βελτιώνεται δια μέσου της εκπαίδευσης. Ένα τεχνητό νευρωνικό δίκτυο εκπαιδεύεται μέσω μιας διαδικασίας αλληλεπίδρασης για τον υπολογισμό των συναπτικών βαρών και των πολώσεων. Έτσι το δίκτυο γίνεται πιο καλά πληροφορημένο σχετικά με το περιβάλλον του μετά από την εκπαίδευσή του. Υπάρχουν πάρα πολλές δραστηριότητες που συνδέονται με την έννοια της εκπαίδευσης ενός δικτύου. Ο ορισμός που δίνεται στην έννοια της εκπαίδευσης στα τεχνητά νευρωνικά δίκτυα είναι ο ακόλουθος:
Εκπαίδευση είναι μια διαδικασία κατά την οποία οι ελεύθεροι παράμετροι ενός τεχνητού νευρωνικού δικτύου προσαρμόζονται μέσω μιας διαδικασίας διέργεσης από το περιβάλλον, την οποία το δίκτυο ενσωματώνει. Ο τύπος της εκπαίδευσης είναι καθορισμένος από τον τρόπο με τον οποίο οι αλλαγές των παραμέτρων πραγματοποιούνται. [17]

Ο καθορισμός της εκπαίδευσης υπονοεί την ακόλουθη σειρά βημάτων:

1) Το τεχνητό νευρωνικό δίκτυο διεγείρεται από το περιβάλλον.
2) Το δίκτυο υποβάλλεται στις αλλαγές στις ελεύθερες παραμέτρους του, ως αποτέλεσμα αυτής της διέργεσης.
3) Το δίκτυο αποκρίνεται με έναν νέο τρόπο στο περιβάλλον εξαιτίας των αλλαγών που έχουν εμφανιστεί στην εσωτερική δομή του.

Ο καθορισμός καλά ορισμένων-πεπερασμένων κανόνων που οδηγούν στη λύση του προβλήματος της εκπαίδευσης καλείται αλγόριθμος εκπαίδευσης. Και φυσικά δεν υπάρχει ένας και μοναδικός αλγόριθμος εκπαίδευσης τεχνητών νευρωνικών δικτύων. Βασικά, οι αλγόριθμοι εκπαίδευσης διαφέρουν μεταξύ τους με τον τρόπο με τον οποίο γίνεται η ρύθμιση σε ένα συναπτικό βάρος ενός νευρώνα [7], [17], [18].

2.1 Λάθος-Διόρθωση εκπαίδευση

Έστω η απλή περίπτωση ενός νευρώνα Κ που αποτελεί το μόνο υπολογιστικό κόμβο στο επίπεδο παραγωγής (εξόδου) ενός τεχνητού νευρωνικού δικτύου προσοτροφοδότησης (feedforward). Ένα ή περισσότερα επίπεδα κρυμμένων νευρώνων διεγείρονται από ένα διάνυσμα εισαγωγής που εφαρμόζεται στους κόμβους πηγής (δηλ., επίπεδο εισαγωγής) του τεχνητού νευρωνικού δικτύου. Το αποτέλεσμα του νευρώνα Κ

δίνεται από το διάνυσμα $y_k(n)$. Αυτό το διάνυσμα του νευρώνα παραγωγής αντιπροσωπεύει τη μόνη παραγωγή αποτελέσματος του τεχνητού νευρωνικού δικτύου. Ωστόσο το επιθυμητό διάνυσμα-στόχων είναι το $d_o(n)$. Συνεπώς, ένα διάνυσμα λάθους $e_k(n)$ προκύπτει. Εξ ορισμού το διάνυσμα λάθους που προκύπτει ισούται με **$e_k(n) = d_k(n) - y_k(n)$.** Το διάνυσμα λάθους $e_κ(n)$ ωθεί έναν μηχανισμό ελέγχου, με σκοπό να εφαρμοστεί μια ακολουθία διορθωτικών ρυθμίσεων στα συναπτικά βάρη του κάθε νευρώνα του δικτύου. Οι διορθωτικές ρυθμίσεις σχεδιάζονται για να προσαρμόσουν το διάνυσμα παραγωγής $y_k(n)$ του νευρώνα K να τείνει στη τιμή του επιθυμητού διανύσματος-στόχων $d_k(n)$. Αυτός ο στόχος επιτυγχάνεται με την ελαχιστοποίηση της συνάρτησης κόστους E(n), όπου $E(n) = 1/2e^2(n)$. Δηλαδή η συνάρτηση E(n) εκφράζει τη στιγμιαία αξία της ενέργειας του λάθους. Οι ρυθμίσεις στα συναπτικά βάρη του νευρώνα K συνεχίζονται έως ότου φθάσει το σύστημα σε μια σταθερή κατάσταση (δηλ., έως ότου τα συναπτικά βάρη σταθεροποιηθούν). Κι αφού προσεγγίσουμε την ελάχιστη τιμή της συνάρτησης λάθους E(n) η διαδικασία εκπαίδευσης ολοκληρώνεται. Η διαδικασία εκπαίδευσης που περιγράφεται αναφέρεται ως εκπαίδευση λάθους-διορθώσεων. Συγκεκριμένα, η ελαχιστοποίηση της συνάρτησης E(n) οδηγεί σε έναν κανόνα εκπαίδευσης που ονομάζεται κανόνας Δέλτα ή κανόνας widrow-Hoff. Η μεταβολή Δw που υφίσταται το συναπτικό βάρος τη δεδομένη χρονική στιγμή n προσδιορίζεται από τη σχέση $\Delta w = \eta * e_k(n) * x(n)$, όπου η είναι μια θετική σταθερά η οποία καθορίζει το ρυθμό εκπαίδευσης καθώς προχωράμε από ένα βήμα στη διαδικασία εκπαίδευσης στο επόμενο.

2.2 Εκπαίδευση που βασίζεται στη μνήμη

Στην εκπαίδευση που βασίζεται στη μνήμη όλα (ή τα περισσότερα) από τα πρότυπα αποθηκεύονται στη μνήμη ταξινομημένα, σύμφωνα με τη σχέση των εισόδων και εξόδων τους $\{(x_i,d_i)\}^N_{i=1}$, όπου x_i είναι το διάνυσμα εισόδου και d_i είναι η επιθυμητή έξοδος του δικτύου. Παραδείγματος χάριν, σε ένα δυαδικό πρόβλημα ταξινόμησης προτύπων υπάρχουν δύο κλάσεις, η C_1 και η C_2. Σε αυτό το παράδειγμα, το επιθυμητό διάνυσμα d_i παίρνει την τιμή 0 για την κατηγορία C_1 και τη τιμή 1 για την κατηγορία C_2. Στη συνέχεια εκπαιδεύουμε το δίκτυο χρησιμοποιώντας τα διανύσματα εκπαίδευσης. Συγκεκριμένα προσπαθούμε να ταξινομήσουμε το δίανυσμα x_{test} σε μια από τις δύο κλάσεις C_1 ή C_2. Ο αλγόριθμος αποκρίνεται ανακτώντας από τη μνήμη και αναλύωντας τα χαρακτηριστικά των αρχικά ταξινομημένων διανυσμάτων σε μια «τοπική γειτονιά» του x_{test}. Σημειώνουμε ότι το διάνυσμα x_{test} δεν έχει δοθεί ποτέ πριν στο δίκτυο. Όλοι οι αλγόριθμοι εκπαίδευσης που βασίζονται στη μνήμη περιλαμβάνουν δύο ουσιαστικά συστατικά:

1) Ένα κριτήριο που χρησιμοποιείται για τον καθορισμό της τοπικής γειτονιάς του διανύσματος δοκιμής $x_{test.}$

2) Ο αλγόριθμος εφαρμόζεται στη τοπική γειτονιά των διανυσμάτων που πρόκειται να ταξινομηθούν.

Ένα απλός αλγόριθμος εκπαίδευσης που βασίζεται στη χρήση μνήμης είναι αυτός του κοντινότερου γείτονα (neighbor rule). Η τοπική γειτονία καθορίζει στο παράδειγμα εκπαίδευσης ότι το διάνυσμα εκπαίδευσης x_{test} θα τοποθετηθεί στην αμέσως πιο κοντινή τοπική-περιοχή. Δηλαδή, έστω ότι έχουμε αποθηκεύσει το διάνυσμα x_N που ανήκει στο σύνολο $\{x_1, x_2, \ldots, x_N\}$ και θέλουμε να ταξινομήσουμε το διάνυσμα x_{test}. Σύμφωνα με το κανόνα του κοντινότερου (ή πλησιέστερου) γείτονα θα πρέπει να βρεθεί με πιο διάνυσμα από τα x_i, όπου i = 1 έως N του παραπάνω συνόλου είναι πιο κοντά το διάνυσμα x_{test}. Δηλαδή, θα κάνουμε χρήση της συνάρτησης της ελάχιστης απόστασης $\min d(x_i, x_{test}) = d(x_N, x_{test})$, όπου $d(x_i, x_{test})$ είναι η ευκλείδια απόσταση μεταξύ των διανυσμάτων x_i και x_{test}. Έτσι το διάνυσμα x_{test} τοποθετείται στη κλάση με την οποία το συνδέει η ελάχιστη απόσταση με το x_i. Δηλαδή ανήκει πλέον στην ίδια κλάση με το διάνυσμα x_i.

Οι Cover και Hart (1967) έχουν μελετήσει τυπικά το κανόνα του κοντινότερου γειτόνα ως εργαλείο για την ταξινόμηση προτύπων. Η ανάλυση που παρουσιάζεται είναι βασισμένη σε δύο υποθέσεις:

1) Τα ταξινομημένα παραδείγματα (x_i, d_i) είναι ανεξάρτητα και ομοιόμορφα κατανεμημένα σύμφωνα με τη κοινή κατανομή πιθανότητας (x, d).

2) Το μέγεθος N του δείγματος είναι οπωσδήποτε πάρα πολύ μεγάλο.

Σύμφωνα με τα παραπάνω και τη βιβλιογραφία αποδεικνύεται ότι η πιθανότητα του λάθους ταξινόμησης από το κανόνα του κοντινότερου γείτονα είναι οριακά δύο φορές η πιθανότητα του λάθους κατά Bayes. Μια παραλλαγή του ταξινομητή του κοντινότερου γείτονα είναι ο ταξινομητής των k κοντινότερων γειτόνων, ο οποίος συνεχίζει ως ακολούθως:

1) Προσδιορίστε τα k ταξινομημένα πρότυπα που βρίσκονται πλησιέστερα στο διάνυσμα δοκιμής $x_{test.}$

2) Αντιστοίχισε το διάνυσμα x_{test} στη κλάση που εμφανίζεται πιο συχνά στους k κοντινότερους γείτονες.

2.3 Hebbian εκπαίδευση

Το αξίωμα εκπαίδευσης του Hebb είναι το παλαιότερο και το διασημότερο όλων των κανόνων εκπαίδευσης. Ονομάζεται έτσι προς τιμή του νευροψυχολόγου Hebb (1949).

Ένας άξονας του κυττάρου Α είναι αρκετά κοντά ώστε να δειγείρει ένα κύτταρο Β. Αν το γεγονός συμβαίνει διαρκώς ή επανειλημμένα και παρατηρείται το φαινόμενο αύξησης της διεργασίας ή των μεταβολικών αλλαγών σε ένα ή και στα δύο κύτταρα τότε λέγεται ότι αυξάνεται η αποδοτικότητα του κυττάρου Α, ως το αποτέλεσμα της πυροδότησης του κυττάρου Β, (The Organization of Behavior, σελ.62) [17].

Ο παραπάνω κανόνας διαμορφώνεται ως εξής (Stent, 1973, Changeux and Danchin, 1976):

1) Εάν δύο νευρώνες από κάθε πλευρά μιας σύναψης (σύνδεση) διεργεθούν συγχρόνως, κατόπιν η δύναμη εκείνης της σύναψης αυξάνεται επιλεκτικά.

2) Εάν δύο νευρώνες από κάθε πλευρά μιας σύναψης διεργεθούν ασύγχρονα, κατόπιν δύναμη εκείνης της σύναψης αποδυναμώνεται επιλεκτικά.

Μια τέτοια σύναψη καλείται σύναψη του Hebb ή αλλιώς Hebbian, (η δεύτερη ερμηνεία δεν περιέχεται στο πρωτότυπο κανόνα του Hebb). Ακριβέστερα, καθορίζουμε μια σύναψη του Hebb ως σύναψη που χρησιμοποιεί έναν χρονικά και τοπικά εξαρτημένο μηχανισμό για να αυξήσει τη συναπτική αποδοτικότητα, ως λειτουργία του συσχετισμού μεταξύ των προσυναπτικών και μετασυναπτικών δραστηριοτήτων. Οι ακόλουθες τέσσερις βασικές ιδιότητες χαρακτηρίζουν μια σύναψη του Hebb:

1) Χρονικά εξαρτημένος μηχανισμός (Time-dependent mechanism). Αυτός ο μηχανισμός αναφέρεται στο γεγονός ότι οι τροποποιήσεις σε μια σύναψη Hebbian εξαρτώνται από τον ακριβή χρόνο που εμφανίζονται τα προσυναπτικά και μετασυναπτικά διανυσμάτα, δηλαδή αν εμφανίστηκαν συγχρόνως ή όχι.

2) Τοπικός μηχανισμός (Local mechanism). Από τη φύση της, μια σύναψη είναι η περιοχή μετάδοσης πληροφορίας που φέρουν τα και θα πρέπει να τα χαρακτηρίζει η χωροχρονική συνάφεια. Αυτές οι τοπικά διαθέσιμες πληροφορίες χρησιμοποιούνται από μια Hebbian σύναψη για να παραγάγουν μια τοπική συναπτική τροποποίηση.

3) Αλληλεπιδραστικός μηχανισμός (interactive mechanism). Η αλλαγή μιας Hebbian σύναψης εξαρτάται από τον τρόπο που αλληλεπιδρούν τα διανύσματα και στις δύο πλευρές της σύναψης.

4) Συνδετικός ή συσχετιστικός μηχανισμός (Conjunctional or correlational mechanism). Μια ερμηνεία της Hebbian εκπαίδευσης είναι ότι ο όρος για μια αλλαγή στη συναπτική αποδοτικότητα είναι η σύνδεση των προσυναπτικών και μετασυναπτικών δραστηριοτήτων. Για αυτόν τον λόγο μια Hebbian σύναψη αναφέρεται μερικές φορές ως συνδετική σύναψη. Ο συσχετισμός με την πάροδο του χρόνου μεταξύ των προσυναπτικών και μετασυναπτικών διανυσμάτων αποτελεί ένα μέτρο της συναπτικής μεταβολής. Συνεπώς, μια σύναψη Hebbian αναφέρεται επίσης ως συσχετιστική σύναψη. Το χαρακτηριστικό του συσχετισμού είναι η βάση της εκπαίδευσης (Eggermont, 1990).

Γενικά η θετικά συσχετισμένη δραστηριότητα έχει ως αποτέλεσμα τη συναπτική ενίσχυση, και η ασύνδετη ή η αρνητικά συσχετισμένη δραστηριότητα παράγει τη συναπτική αποδυνάμωση (Stent, 1973). Με τους όρους ενίσχυση και αποδυνάμωση εννοούμε ότι οι μεταβολές των των τιμών των βαρών είναι μεγάλες ή μικρές αντίστοιχα, ως το αποτέλεσμα της διέργεσης του νευρώνα. Οι τροποποιήσεις των Hebbian συνάψεων διαχωρίζονται σε Hebbian, αντι-Hebbian, και μη-Hebbian (palm, 1982). Σύμφωνα με αυτό το πρότυπο, μια Hebbian σύναψη αυξάνει τη τιμή της με τα θετικά συσχετισμένα προσυναπτικά και μετασυναπτικά διανύσματα, και μειώνει τη τιμή της όταν αυτά είναι είτε ασύνδετα είτε αρνητικά συσχετισμένα. Αντιθέτως, μια αντι-Hebbian σύναψη αποδυναμώνει τα θετικά συσχετισμένα προσυναπτικά και μετασυναπτικά διανύσματα, και ενισχύει τα αρνητικά συσχετισμένα. Και στις δύο πρώτες περιπτώσεις η τροποποίηση της συναπτικής αποδοτικότητας στηρίζεται σε έναν μηχανισμό που είναι χρονικά εξαρτημένος, ιδιαίτερα τοπικός, και έντονα αλληλεπιδραστικής φύσης. Υπό αυτή την έννοια, μια αντι-Hebbian σύναψη είναι ακόμα Hebbian στη φύση, αλλά όχι στη λειτουργία. Μια σύναψη μη-Hebbian, δεν περιλαμβάνει έναν Hebbian μηχανισμό κανενός είδους.

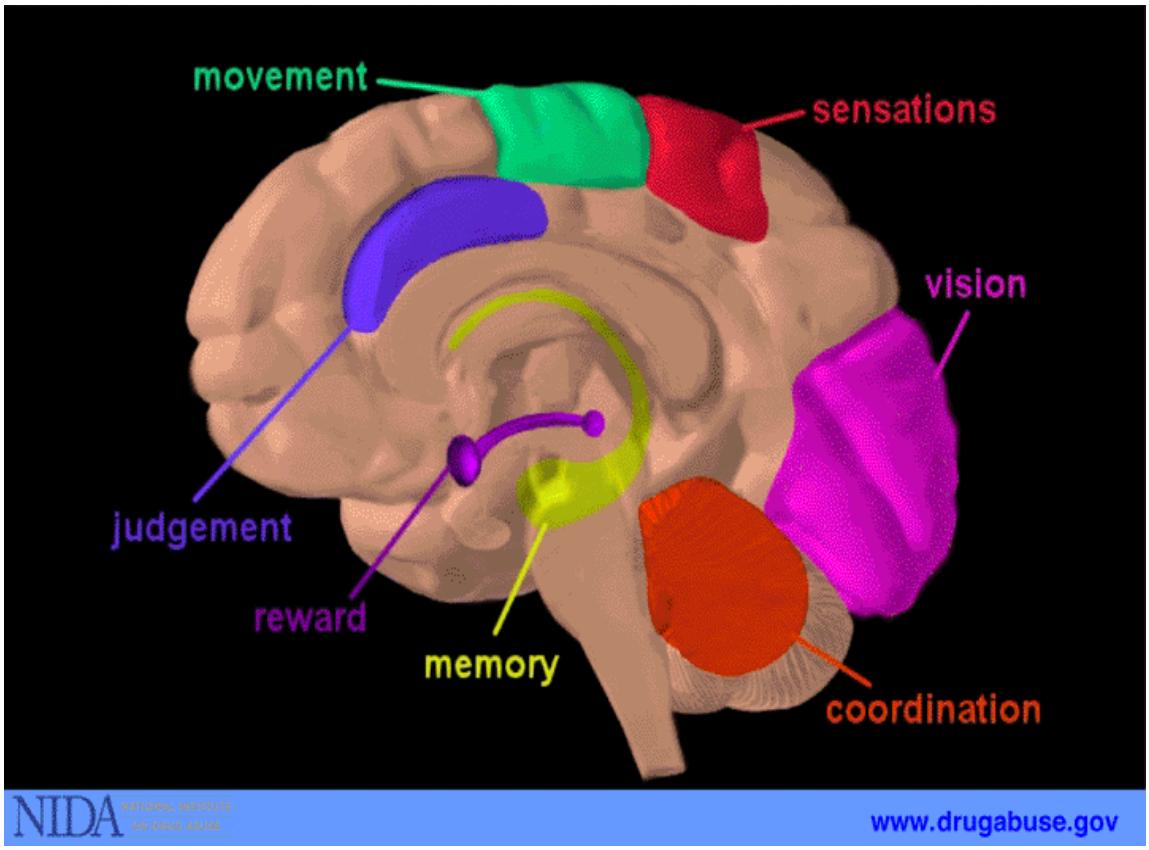
2.3.1 Μαθηματικά πρότυπα των τροποποιήσεων Hebbian

Έστω ένα συναπτικό βάρος w_{kj} του νευρώνα K με τα προσυναπτικά και μετασυναπτικά διανύσματα x_j και y_k αντίστοιχα. Η ρύθμιση που εφαρμόζεται στο συναπτικό βάρος w_{kj} τη χρονική στιγμή n εκφράζεται στη γενική μορφή:
$\Delta w_{kj}(n) = F(y_k(n), x_j(n))$ (1), όπου F(., .) είναι μια συνάρτηση των μετασυναπτικών και προσυναπτικών διανυσμάτων. Ο υπολογισμός των νέων βαρών γίνεται κάνοντας χρήση μιας εκ των δυο υποθέσεων παρακάτω:

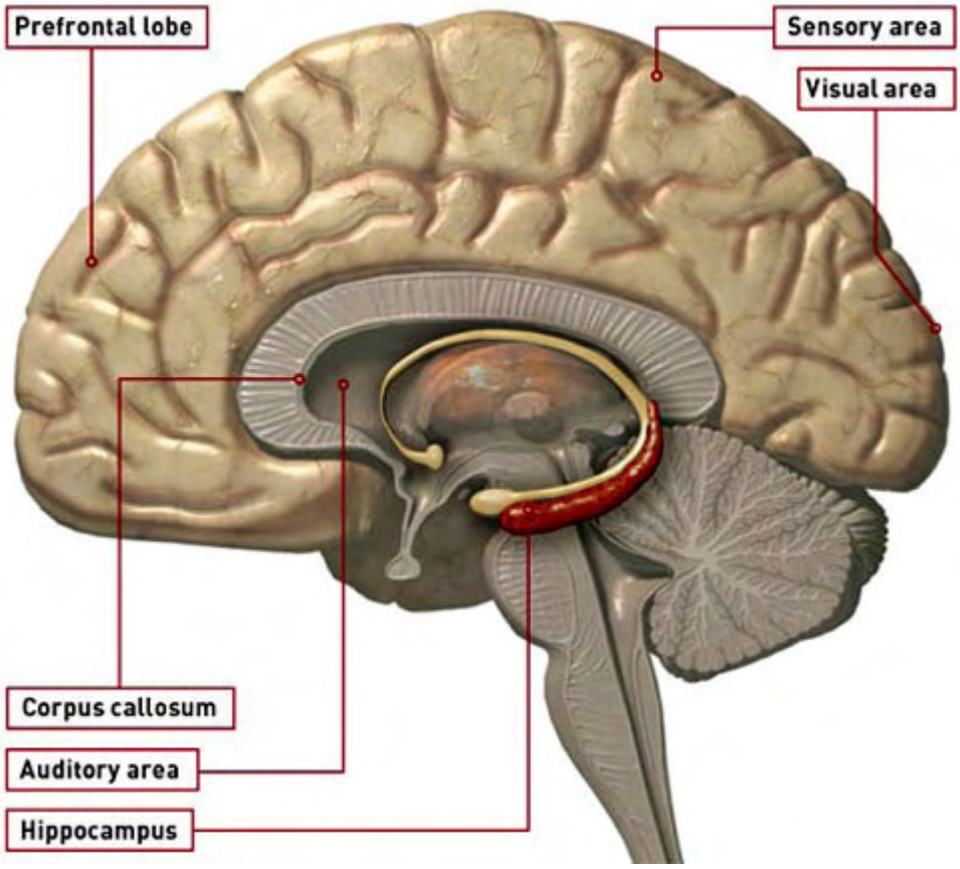
1) Υπόθεση Hebb. Η απλούστερη μορφή Hebbian εκπαίδευσης περιγράφεται από τη σχέση $\Delta w_{kj}(n) = \eta * y_k(n) * x_j(n)$ (2), όπου η μια θετική σταθερά που ορίζει το ρυθμό εκπαίδευσης. Ο τύπος (2) παραπάνω υπογραμμίζει τη συσχετιστική φύση μιας Hebbian σύναψης. Ο βασικός περιορισμός τη υπόθεσης του Hebb είναι το γεγονός ότι καθώς αυξάνει η μετασυναπτική δραστηριότητα φτάνει σε κορεσμό η τροποποίηση των βαρών.

2) Υπόθεση συνδιακύμανσης. Ένας τρόπος να ξεπεράσουμε το περιορισμό της υπόθεσης του Hebb είναι να κάνουμε χρήση της υπόθεσης της συνδιακύμανσης που εισήγαγε ο Sejnowski (1977). Σε αυτήν την περίπτωση, τα προσυναπτικά και μετασυναπτικά διανύσματα της ισότητας (2) αντικαθίσταται από τις αντίστοιχες μέσες τιμές τους για ένα χρονικό διάστημα. Σύμφωνα με την υπόθεση της συνδιακύμανσης, τα συναπτικά βάρη τροποποιούνται βάσει του τύπου $\Delta w_{kj}(n) = \eta*(y_\kappa - y_{median})*(x_j - x_{median})$ (3), όπου y_{median} και x_{median} οι μέσες τιμές των μετασυναπτικών και προσυναπτικών διανυσμάτων αντίστοιχα, για κάποιο χρονικό διάστημα. Ο αριθμός η είναι ο ρυθμός εκπαίδευσης του δικτύου. Οι μέσες τιμές αποτελούν τα προσυναπτικές και μετασυναπτικές κατώτατες τιμές που καθορίζουν τη συναπτική τροποποίηση.

Υπάρχουν ισχυρά φυσιολογικά στοιχεία της Hebbian εκπαίδευσης στη περιοχή του εγκεφάλου που καλείται ιππόκαμπος. Ο ιππόκαμπος διαδραματίζει έναν σημαντικό ρόλο σε ορισμένες πτυχές της εκπαίδευσης ή της μνήμης. Αυτά τα φυσιολογικά στοιχεία κάνουν την εκμάθηση του κανόνα του Hebb ακόμη πιο ελκυστική [5], [13], [17].



Εικόνα 2.1 Προσδιορισμός της μνήμης στον εγκέφαλο



Εικόνα 2.2 Προσδιορισμός του ιπποκάμπου στον εγκέφαλο

2.4 Ανταγωνιστική εκπαίδευση

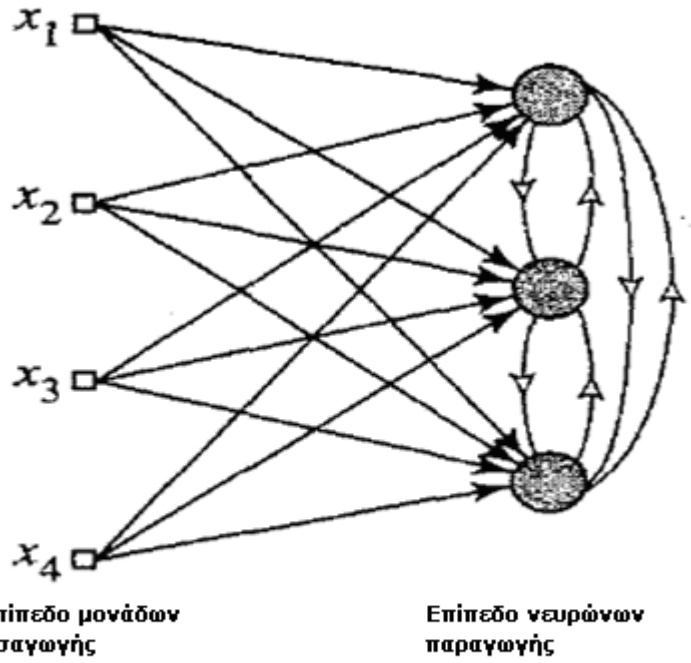
Στην ανταγωνιστική εκπαίδευση, οι νευρώνες παραγωγής ενός τεχνητού νευρωνικού δικτύου ανταγωνίζονται μεταξύ τους για να γίνουν ενεργοί καθώς μόνο ένας νευρώνας παραγωγής μπορεί να είναι ενεργός κάποια χρονική στιγμή. Είναι αυτό το χαρακτηριστικό γνώρισμα που καθιστά την ανταγωνιστική εκπαίδευση ιδιαίτερη ώστε να ανακαλύψει τα στατιστικά κατάλληλα χαρακτηριστικά γνωρίσματα που μπορούν να χρησιμοποιηθούν για να ταξινομήσουν ένα σύνολο προτύπων.

Υπάρχουν τρία βασικά χαρακτηριστικά στην ανταγωνιστική εκπαίδευση (Rumelhart and Zipser, 1985):

1) Υπάρχει ένα σύνολο νευρώνων που είναι όλοι οι ίδιοι εκτός από μερικά τυχαία προσδιορισμένα συναπτικά βάρη, και που επομένως αποκρίνονται διαφορετικά σε ένα δεδομένο σύνολο προτύπων εισαγωγής.
2) Επιβάλλεται ένα όριο στη «δύναμη» κάθε νευρώνα.

3) Υπάρχει ένας μηχανισμός που επιτρέπει στους νευρώνες να ανταγωνιστούν μεταξύ τους, ώστε να προκύψει ένας νευρώνας παραγωγής που ονομάζεται νικητής.

Έτσι οι νευρώνες του τεχνητού νευρωνικού δικτύου μαθαίνουν να ειδικεύονται σε σύνολα παρόμοιων προτύπων και με αυτές τις ιδιότητες που αποκτούν γίνονται ανιχνευτές χαρακτηριστικών γνωρισμάτων για τις διαφορετικές κατηγορίες προτύπων εισαγωγής. Η Εικόνα 2.3 παριστάνει ένα τέτοιο δίκτυο εκπαίδευσης με ανταγωνισμό.



Εικόνα 2.3 Συνδέσεις από τους κόμβους εισόδου στους νευρώνες, και πλευρικές συνδέσεις μεταξύ των νευρώνων, οι πλευρικές συνδέσεις δηλώνονται από τα ανοικτά βέλη. Ένας από τους τρεις νευρώνες θα είναι ο νικητής

Στην απλούστερη μορφή του, ένα τεχνητό νευρωνικό δίκτυο είναι ένα ενιαίο επίπεδο από νευρώνες παραγωγής, όπου κάθε ένας συνδέεται πλήρως με τους κόμβους εισαγωγής. Το δίκτυο μπορεί να περιλάβει ως σχέσεις ανατροφοδότησης τις συνδέσεις μεταξύ των νευρώνων. Στη παραπάνω αρχιτεκτονική του τεχνητού νευρωνικού δικτύου η σχέση ανατροφοδότησης έχει ως σκοπό την παρεμπόδιση του νευρώνα με τον οποίο συνδέεται πλευρικά. Αντίθετα, οι συναπτικές συνδέσεις προσο-τροφοδότησης στο δίκτυο είναι όλες ενισχυτικές. Ο νικητής νευρώνας παίρνει τη τιμή ένα κι όλοι υπόλοιποι τη τιμή μηδέν. Στη συνέχεια ανατροφοδοτείται με πρότυπα εισαγωγής ο νικητής νευρώνας. Το συνολικό άθροισμα των βαρών σε κάθε νευρώνα θα πρέπει να είναι ίσο με ένα. Αν ένας νευρώνας δεν αποκρίνεται σε ένα πρότυπο εισαγωγής τότε δε πραγματοποιείται σε αυτόν η διαδικασία της εκπαίδευσης. Ένας νευρώνας μαθαίνει να εκπαιδεύεται με τη

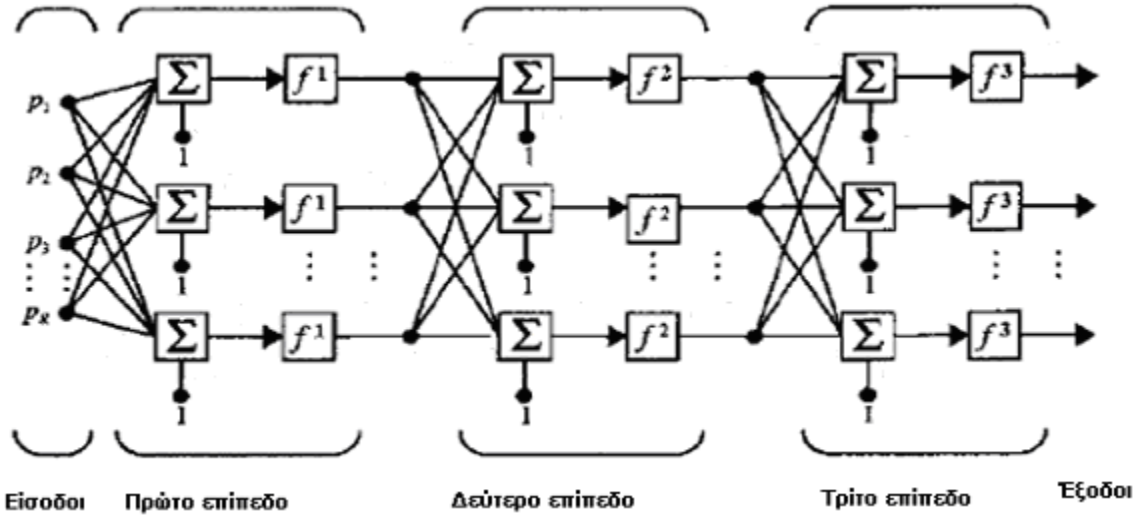
μετατόπιση των συναπτικών βαρών από τους ανενεργούς προς τους ενεργούς κόμβους εισαγωγής του. Εάν ένας νευρώνας κερδίσει τον ανταγωνισμό, κάθε κόμβος εισαγωγής εκείνου του νευρώνα σταματά σε κάποιο ποσοστό του συναπτικού βάρους του, και το υπόλοιπο διανέμεται έπειτα εξίσου μεταξύ των ενεργών κόμβων εισαγωγής σύμφωνα με τον ανταγωνιστικό κανόνα εκπαίδευσης. Αυτός ο κανόνας έχει τη γενική επίδραση της κίνησης του συναπτικού διανύσματος του βάρους, από το νευρώνα που κερδίζει προς τις συνδέσεις εισαγωγής [5], [13], [17].

3 Ταξινόμηση νευρωνικών δικτύων

3.1 Τοπολογία νευρωνικών δικτύων

Ως προς την τοπολογία τα τεχνητά νευρωνικά δίκτυα διακρίνονται σε δίκτυα με προς τα εμπρός ροή των δεδομένων και σε δίκτυα με ανατροφοδότηση. Αναλυτικότερα:

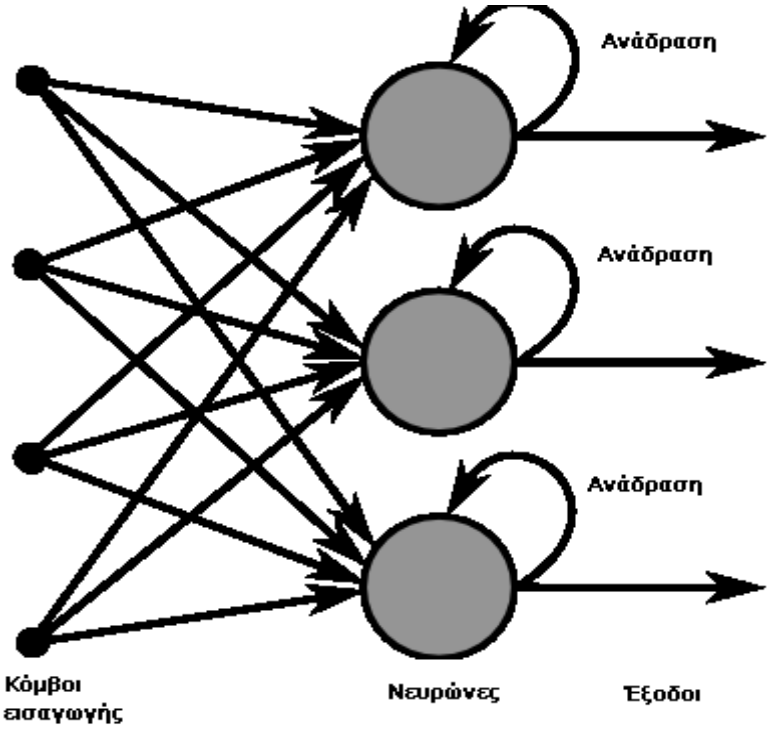
1) Νευρωνικά δίκτυα προσοτροφοδότησης (feed-forward networks): Η ροή των δεδομένων είναι αυστηρά από την είσοδο προς της έξοδο. Η επεξεργασία των δεδομένων μπορεί να επεκτείνεται σε πολλές μονάδες, χωρίς όμως να παρουσιάζονται συνδέσεις ανατροφοδότησης. Οι μονάδες μπορεί να είναι οργανωμένες σε επίπεδα. Ως επίπεδο ορίζεται το στρώμα στο οποίο γίνεται κάποιος υπολογισμός. Στη γενική περίπτωση περιέχει ένα ή περισσότερα κρυμμένα επίπεδα (hidden layers) των οποίων οι κόμβοι ονομάζονται κρυμμένοι νευρώνες. Τα δίκτυα αυτά ονομάζονται και πολυεπίπεδα (ή πολυστρωματικά) τεχνητά νευρωνικά δίκτυα (ΤΝΔ) προσοτροφοδότησης. Οι έξοδοι του προηγούμενου επιπέδου λειτουργούν ως είσοδοι στο επόμενο ως ότου φθάσουν στο τελευταίο επίπεδο το οποίο ονομάζεται επίπεδο εξόδου. Τα πολυεπίπεδα δίκτυα τα συμβολίζουμε ως εξής: 5-8-3-1, όπου 5 είναι οι κόμβοι εισόδου των παραμέτρων, 8 είναι οι κόμβοι του πρώτου κρυφού επιπέδου, 3 είναι οι κόμβοι του δεύτερου κρυφού επιπέδου και 1 είναι ο κόμβος στο επίπεδο της εξόδου.



Εικόνα 3.1 Δίκτυο προσοτροφοδότησης τριών επιπέδων

Στην Εικόνα 3.1 απεικονίζεται ένα δίκτυο προσοτροφοδότησης τριών επιπέδων. Περιγράφεται από την έκφραση (αν θεωρήσουμε ότι το διάνυσμα εισόδου έχει πέντε παραμέτρους, το πρώτο επίπεδο πέντε κόμβους, το δεύτερο τρεις και το τρίτο επίσης τρεις.) 5-5-3-3. Η έξοδος από τους κόμβους του πρώτου επιπέδου υπολογίζεται από τον τύπο $a_1 = f_1(w_1 p + b_1)$. Από το δεύτερο υπολογίζεται από τον τύπο $a_2 = f_2(w_2 a_1 + b_2)$ και από το τρίτο από τον τύπο $a_3 = f_3(w_3 a_2 + b_3)$. Γενικότερα, η έξοδος στο τελευταίο επίπεδου του δικτύου που περιγράψαμε υπολογίζεται από τον τύπο:
$\mathbf{a_3 = f_3(w_3 f_2(w_2 f_1(w_1 p + b_1) + b_2) + b_3)}$.

2) Δίκτυα με ανατροφοδότηση (recurrent networks): Τα δίκτυα αυτά περιέχουν συνδέσεις με ανατροφοδότηση. Αντίθετα με τα δίκτυα προσοτροφοδότησης οι δυναμικές ιδιότητες αυτών των δικτύων είναι σημαντικές. Ανατροφοδότηση σημαίνει ότι ανακυκλώνεται πληροφορία. Αποτέλεσμα της ανατροφοδότησης είναι να μην παράγεται ένα πρότυπο εξόδου σε ένα πεπερασμένο αριθμό χρονικών βημάτων, αλλά να δρα κυκλικά. Ως εκ τούτου τα ίδια στρώματα επαναλειτουργούν. Το σημαντικότερο εδώ κατά την εκπαίδευση είναι να βρούμε τις τιμές των συναπτικών βαρών για τις οποίες το δίκτυο σταθεροποιείται και δεν επαναλαμβάνει τη λειτουργία της ανάδρασης.



Εικόνα 3.2 Δίκτυο ανατροφοδότησης

Στην Εικόνα 3.2 αναπαρίσταται ένα δίκτυο με ανατροφοδότηση. Όπως μπορούμε να δούμε οι έξοδοι από το νευρώνα μπορούν να ανατροφοδοτήσουν το δίκτυο. Σημειώνεται ότι οι βρόχοι ανατροφοδότησης έχουν θετική επίδραση στην εκπαίδευση του δικτύου.

Παραδείγματα δικτύων προσοτροφοδότησης (feed-forward network) είναι οι τοπολογίες δικτύων perceptron και adaline. Αντίθετα, παραδείγματα δικτύων με ανατροφοδότηση (recurrent networks) είναι οι τοπολογίες kohonen και Hopfield [5], [17].

3.2 Εκπαίδευση τεχνητών νευρωνικών δικτύων

Ο τρόπος με τον οποίο τροποποιούμε τις τιμές στα βάρη σε ένα τεχνητό νευρωνικό δίκτυο ονομάζεται εκπαίδευση. Έτσι, ανάλογα με την τεχνική εκπαίδευσης τα τεχνητά νευρωνικά δίκτυα διακρίνονται σε δίκτυα με επίβλεψη, χωρίς επίβλεψη και σε δίκτυα με ενισχυτική μάθηση. Αναλυτικότερα:

1) Μάθηση με επίβλεψη (supervised learning): Οι είσοδοι γίνονται δεκτοί από το δίκτυο και οι έξοδοι συγκρίνονται με τα επιθυμητά αποτελέσματα. Ο κανόνας αυτός εκπαίδευσης χρησιμοποιείται για την ρύθμιση των βαρών w και των πολώσεων b, ώστε οι έξοδοι του δικτύου να τείνουν στα επιθυμητά αποτελέσματα.

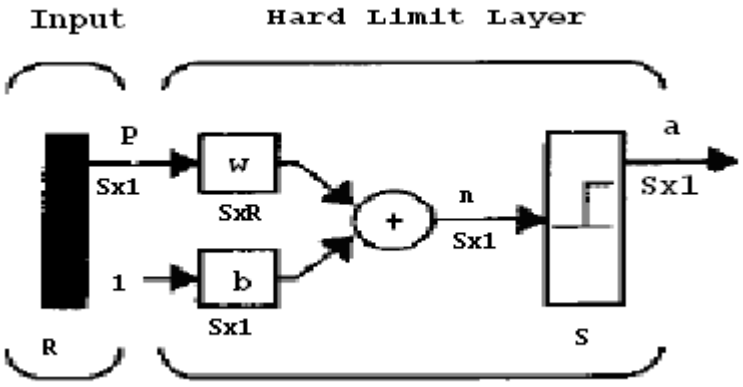
2) Μάθηση χωρίς επίβλεψη (unsupervised learning): Οι τιμές των βαρών w και των πολώσεων b τροποποιούνται σε σχέση με τις εισόδους. Αρχικά μοιάζει μη πρακτική μέθοδος, διότι τίθεται το ερώτημα πως μπορούμε να εκπαιδεύσουμε ένα δίκτυο χωρίς να ξέρουμε τι πρόκειται να κάνει. Η απάντηση είναι ότι οι περισσότεροι από αυτούς τους αλγορίθμους εκτελούν τη διαδικασία της ομαδοποίησης. Μαθαίνουν να κατηγοριοποιούν ένα πρότυπο εισόδου σε ένα πεπερασμένο αριθμό ομάδων.

3) Ενισχυτική μάθηση (reinforcement learning): Είναι μια παρόμοια μέθοδος με τη μέθοδο με επίβλεψη, μόνο που εδώ αντί να δίδεται η σωστή έξοδος του δικτύου για κάθε είσοδο, δίδεται ένας μόνο βαθμός (score). Ο βαθμός αυτός είναι ένα μέτρο της απόδοσης του δικτύου για μία ακολουθία εισόδων. Αυτός ο τύπος εκπαίδευσης είναι λιγότερο διαδεδομένος. Φαίνεται να είναι κατάλληλος ώστε να ελέγχει εφαρμογές συστημάτων [5], [17].

4 Perceptron και adaline

4.1 Δίκτυα με λειτουργίες ενεργοποίησης κατώτατων ορίων

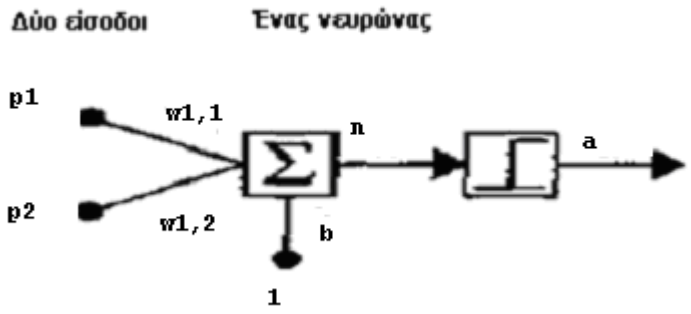
Ένα επίπεδο ενός τεχνητού νευρωνικού δικτύου προσοτροφοδότησης (feed-forward) περιέχει έναν ή περισσοτέρους νευρώνες που δίνουν μια έξοδο (o), ο καθένας από τους οποίους είναι συνδεδεμένος με όλες τις εισόδους (i) έχοντας κάποιο βάρος w(i). Η είσοδος στο νευρώνα είναι η σταθμισμένη τιμή των εισόδων προσθέτοντας και τη τιμή πόλωσης b. Το αποτέλεσμα της εξόδου διαμορφώνεται από τη συνάρτηση ενεργοποίησης.



Εικόνα 4.1 Χρήση της συνάρτησης hardlim της Matlab

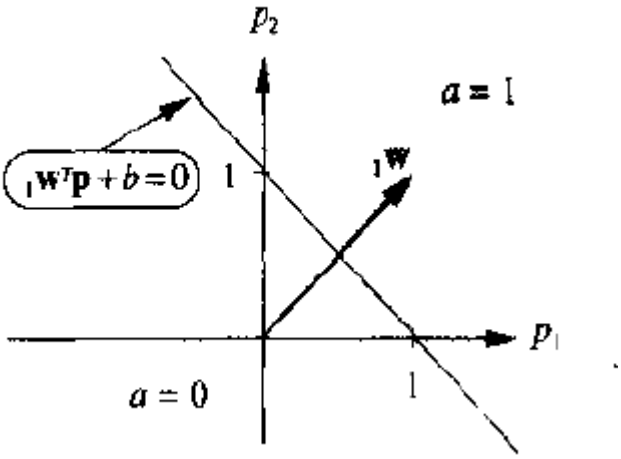
Στην Εικόνα 4.1 παρατηρούμε ότι το διάνυσμα είναι μεγέθους Rx1, ο πίνακας των βαρών w μεγέθους SxR και ο πίνακας των πολώσεων b μεγέθους Sx1. Οι τιμές των προτύπων σταθμίζονται και στη συνέχεια αθροίζονται. Η έξοδος του νευρώνα είναι ο πίνακας n μεγέθους Sx1. Και μέσω της συνάρτησης ενεργοποίησης f παράγεται η έξοδος a, που επίσης είναι διάνυσμα μεγέθους Sx1. Ο τύπος που δίνει την έξοδο είναι ο:
a = hardlim(wp + b).

Αν υποθέσουμε ότι το διάνυσμα έχει μέγεθος 2x1 κι έχουμε κι ένα νευρώνα, (Εικόνα 4.2),



Εικόνα 4.2 Δίκτυο δύο εισόδων

τότε η έξοδος a υπολογίζεται στο λογισμικό Matlab από τον παρακάτω τύπο:
a = hardlim(n) = hardlim(wp + b) = hardlim($w^T p + b$) = hardlim($w_{1,1}p_1 + w_{1,2}p_2 + b$)
Θα έχουμε f(n) = 1, αν n > 0 ή n = 0 και f(n) = 0, αλλιώς. Οπότε οι είσοδοι κατηγοριοποιούνται σε δυο κλάσεις. Η γεωμετρική αναπαράσταση είναι παρόμοια με αυτή της Εικόνας 4.3.



Εικόνα 4.3 Γραμμή διαχώρισης δεδομένων

Η γραμμή διαχώρισης δίδεται από τη σχέση $\Sigma w_i + b = 0 \Rightarrow wx_1 + wx_2 + b = 0$ (1)

Για να υπολογίσουμε τη κλίση της ευθείας απλά τροποποιούμε τον τύπο (1) και γράφουμε: $x_2 = -(w_1/w_2)x_1 - b/w_2$

Πως μπορούμε να τροποποιήσουμε τις τιμές των βαρών και των σταθερών εισόδων;

Χρησιμοποιώντας τον αλγόριθμο εκπαίδευσης perceptron και το κανόνα delta.

4.2 Αλγόριθμος εκπαίδευσης perceptron

Μοντέλο Perceptron: Είναι δυαδικός ταξινομητής και προτάθηκε από τον Rosenblatt το 1959. Τα βήματα που πρέπει να εκτελέσουμε στον αλγόριθμο εκπαίδευσης perceptron (perceptron learning rule) είναι τα ακόλουθα:
Έστω ότι έχουμε το διάνυσμα x ως είσοδο και ως έξοδο το διάνυσμα f(x). Οι τιμές που παίρνει η έξοδος είναι -1 ή 1.

1. Επιλέγουμε τυχαίες τιμές για τα βάρη.
2. Επιλέγουμε ένα διάνυσμα x από το σύνολο.
3. Αν το y που είναι η επιθυμητή τιμή διαφέρει από την f(x) που είναι η πραγματική τότε :
4. Υπολόγισε τα βάρη w_i σύμφωνα με το τύπο $\Delta w_i = f(x_i)x_i$.
5. Επανέλαβε το βήμα 2.

Οπότε η τιμή του νέου βάρους είναι $w_i(t + 1) = w_i(t) + \Delta w_i(t)$.
Ομοίως για τον υπολογισμό του Δb έχουμε ότι Δb = 0, αν ο perceptron ανταποκριθεί σωστά, αλλιώς Δb = f(x). Και η τιμή του νέου b υπολογίζεται από τον τύπο $b(t + 1) = b(t) + \Delta b(t)$ [5], [17].

4.2.1 Θεώρημα σύγκλισης

Εάν υπάρχει ένα σύνολο βαρών **w***, το οποίο είναι ικανό να εκτελέσει τη μετατροπή y = f(x), ο αλγόριθμος εκπαίδευσης perceptron θα συγκλίνει σε κάποιες λύσεις (οι οποίες μπορεί να είναι ίδιες ή όχι με το σύνολο **w***), μέσα σε ένα πεπερασμένο αριθμό βημάτων για οποιανδήποτε επιλογή βαρών.

4.2.2 Ο αρχικός perceptron

Ο όρος perceptron είναι μετάδοση του όρου perception που σημαίνει αντίληψη. Στην πιο απλή του μορφή είναι ένα επίπεδο n-στοιχείων εισόδου, τα οποία τροφοδοτούν ένα επίπεδο από m συνδεδεμένες μονάδες και μια μονάδα εξόδου. Οι αλγόριθμοι perceptron μπορούν να κατηγοριοποιηθούν σε διαφορετικές οικογένειες. Η μονάδα εξόδου ενός peceptron είναι ένα γραμμικό στοιχείο, αποτέλεσμα κατώτατων ορίων. Ο Rosenblatt το 1959 διατύπωσε ένα θεώρημα για τον αλγόριθμο εκπαίδευσης perceptron και το 1969 στην εργασία των Minsky και Papert’s Perceptrons ορίστηκαν αυστηροί περιορισμοί.

4.3 Adaline

Μοντέλο Adaline: Παρουσιάστηκε από τους Widrow και Hoff το 1960 Ο όρος adaline προέρχεται από τα αρχικά των όρων adaptive linear element. Ένας σημαντικός λοιπόν κανόνας εκπαίδευσης είναι αυτός του ελάχιστου τετραγωνικού σφάλματος (LMS = least mean square) ή αλλιώς γνωστός και ως κανόνας Δέλτα

4.3.1 Υλοποίηση του κανόνα Δέλτα

Έστω ότι πρόκειται για τεχνητό νευρωνικό δίκτυο ενός επιπέδου, με μια έξοδο και γραμμική συνάρτηση ενεργοποίησης. Έστω ότι η έξοδος είναι y. Τότε $y = \Sigma w_i x_i + b$. Για κάθε πρότυπο εισόδου x^p η έξοδος του δικτύου y^p διαφέρει από το επιθυμητό αποτέλεσμα d^p. Ο κανόνας Δέλτα χρησιμοποιεί τη συνάρτηση απόδοσης σφάλματος. Η συνάρτηση απόδοσης σφάλματος υπολογίζει το συνολικό τετραγωνικό σφάλμα. Ο τύπος είναι:

$E = \Sigma(E^p) = ½ \Sigma (d^p - y^p)^2$ για κάθε p,

όπου p το πρότυπο εισόδου και E^p το σφάλμα για το πρότυπο p. Ο κανόνας Δέλτα (ή LMS) υπολογίζει τα νέα βάρη που ελαχιστοποιούν τη συνάρτηση σφάλματος. Έτσι πρέπει να υπολογίσουμε τη παράγωγο της συνάρτησης E. Σύμφωνα λοιπόν με τη θεωρία

$\Delta_p w_j = - \gamma \partial E^p / \partial w_j$ (1), όπου το γ είναι μια σταθερά, (ο ρυθμός εκπαίδευσης).

$\partial E^p / \partial w_j = (\partial E^p / \partial y^p) (\partial y^p / \partial w_j)$, (2)

Ισχύει όμως ότι

$\partial y^p / \partial w_j = x_j$, (3) και

$\partial E^p / \partial y^p = - (d^p - y^p)$, (4)

Έτσι προκύπτει ότι ο κανόνας τροποποίησης των βαρών είναι ο παρακάτω:

$\mathbf{\Delta_p w_j = \gamma \delta^p x_j}$, όπου $\delta^p = d^p - y^p$

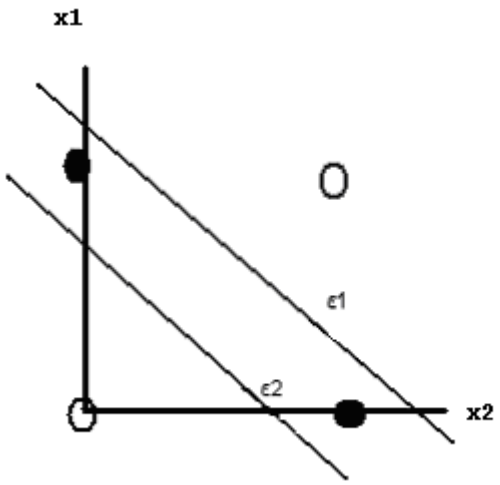
4.4 Περιορισμοί των perceptron και adaline

Στο βιβλίο των Minsky και Papert συζητείται ο περιορισμός των παραπάνω αλγορίθμων να αποδώσουν το πρόβλημα της συνάρτησης αποκλειστικού-είτε (x-or). Ο Πίνακας 4.1 είναι ο πίνακας αληθείας της πύλης x-or.

Πίνακας 4.1 Πίνακας αληθείας του τελεστή x-or

X_1	X_2	Y
0	0	0
0	1	1
1	0	1
1	1	0

Η έξοδος του perceptron είναι 0 αν $y < 0$ και $y = 1$ αν $y \geq 0$.
Το αποτέλεσμα του δικτύου θα είναι: (ε_1) $y = w_1x_1 + w_2x_2 + b$.
Γεωμετρικά περιγράφεται από την Εικόνα 4.4



Εικόνα 4.4 Διαχωρισμός δεδομένων του τελεστή x-or

Παρατηρούμε ότι ο αλγόριθμος δεν είναι ικανός να διαχωρίσει τις εξόδους σε δύο κλάσεις. Ο διαχωρισμός γίνεται σύμφωνα με την ευθεία (ε_1) ή σύμφωνα με την ευθεία

(ε_2). Αν ο διαχωρισμός γίνει με την ευθεία (ε_1) τότε πρόκειται για τον πίνακα αληθείας της πύλης and, ενώ αν γίνει σύμφωνα με την ευθεία (ε_2) τότε πρόκειται για τον πίνακα αληθείας της πύλης or. Αυτή είναι και η αδυναμία ενός τέτοιου απλού δικτύου. Το πρόβλημα της πύλης x-or μπορεί να λυθεί με τη χρήση κρυφών επιπέδων. Για δυαδικές μονάδες, ο perceptron με τη χρήση κρυφών επιπέδων μπορεί να βρει λύση δίνοντας τα σωστά βάρη και τους σωστούς μετασχηματισμούς.

Για παράδειγμα έχουμε το μετασχηματισμό y = d(x) και θέλουμε να διαχωρίσουμε ένα σύνολο εισόδων σε δύο κλάσεις. Το τεχνητό νευρωνικό δίκτυο θα κάνει χρήση ενός κρυφού επιπέδου και θα έχει μια έξοδο και N μονάδες εισόδου. Ο δυνατός πιθανός αριθμός εισόδων θα είναι 2^N, για κάθε διάνυσμα που ανήκει στο σύνολο των δεδομένων μας. Για κάθε κρυφή μονάδα h η συνάρτηση ενεργοποίησης δίδεται από τον τύπο:

$$Y_h^p = \text{sgn} \left(\sum_i w_{ih} x_i^p - N + ½ \right),$$

όπου αρχικά οι τιμές των βαρών $\mathbf{w}_{ih}$ είναι ίσες με τις τιμές του κάθε προτύπου, δηλαδή $\mathbf{w}_{ih} = \mathbf{x}^p$ και οι τιμές των b_h είναι ίσες με 1-N. Ο τύπος που δίνει το αποτέλεσμα της εξόδου του δικτύου είναι ο :

$$y_0^p = \text{sgn} \left(\sum_{h=1}^{M} y_h + M - ½ \right),$$

όπου M είναι ο συνολικός αριθμός των νευρώνων.

4.5 Σύνοψη

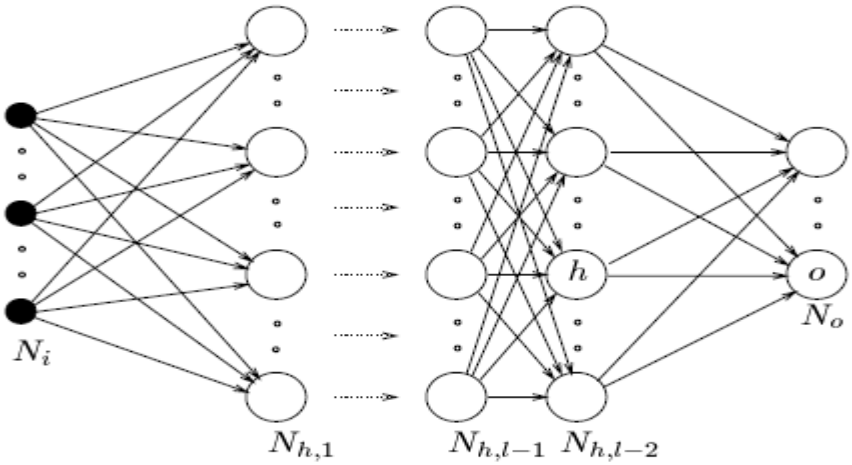
Συνοψίζοντας οι αλγόριθμοι perceptron και adaline εμφανίζουν μειονεκτήματα και πλεονεκτήματα. Το βασικότερο μειονέκτημα τους είναι η περιορισμένη υπολογιστική δυνατότητα. Ο διαχωρισμός των δεδομένων παρουσιάζεται γραμμικά. Αντίθετα, το βασικότερο πλεονέκτημα το οποίο και οφείλεται στη γραμμικότητα του προβλήματος, είναι ότι η εκπαίδευση του τεχνητού νευρωνικού δικτύου δίνει τη βέλτιστη λύση [1], [5], [13], [17].

5 Νευρωνικό δίκτυο με όπισθεν διάδοση του σφάλματος (Backpropagation neural network)

Στα μέσα της δεκαετίας του 1980 δημοσιεύτηκε ανεξάρτητα από τους David Rumelhort, Geoffrey Hinton και Ronald Williams, David Parter και τέλος τον Vann Le Cun το θέμα των Backpropagation τεχνητών νευρωνικών δικτύων (τεχνητά νευρωνικά δίκτυα με όπισθεν διάδοση του σφάλματος), με την απόδειξη ότι αυτά μπορούν να χειριστούν προβλήματα που η λύση τους είναι μη γραμμική. Γενικότερα, τα δίκτυα αυτά είναι επέκταση των adaline δικτύων. Η κεντρική ιδέα είναι η εξής: Τα λάθη στις τιμές των βαρών για τις μονάδες του κρυφού στρώματος καθορίζονται με τη πίσω διάδοση των λαθών των μονάδων του στρώματος παραγωγής των εξόδων. Στη παράγραφο αυτή οι όροι backpropagation και γενίκευση του κανόνα Δέλτα για μη γραμμικές συναρτήσεις ενεργοποίησης ταυτίζονται.

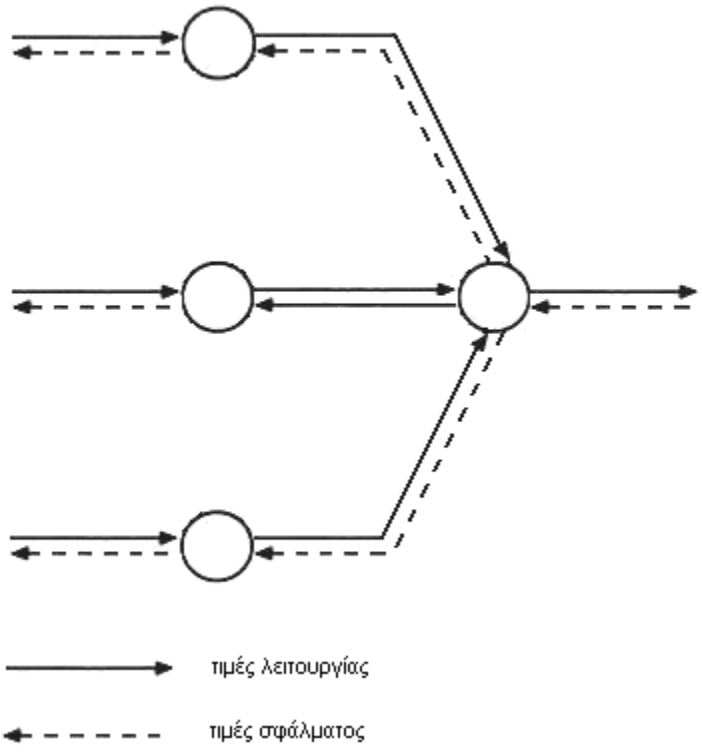
5.1 Χαρακτηριστικά Backpropagation

Η διαδικασία της εκπαίδευσης συνίσταται από δύο «περάσματα» σε όλα τα επίπεδα του δικτύου που καλούνται ευθύ και ανάδρομο πέρασμα. Έστω ότι έχουμε ένα σύνολο εκπαίδευσης που αποτελείται από N ζεύγη της μορφής [**x**,**t**], όπου x το διάνυσμα εισόδου και t το διάνυσμα των επιθυμητών εξόδων όταν εφαρμοστεί ως είσοδος το διάνυσμα x. Στην διαδικασία του ευθέως περάσματος (forward pass) όλα τα βάρη των συνδέσεων παραμένουν ως έχουν και υπολογίζονται οι έξοδοι όλων των νευρώνων. Το ευθύ πέρασμα ξεκινά από το πρώτο κρυμμένο επίπεδο με την εφαρμογή του διανύσματος εισόδου. Στη συνέχεια υπολογίζει όλα τα σήματα εξόδου των νευρώνων και προχωράει στα επόμενα επίπεδα του δικτύου, μέχρι να φτάσει τελικά στο επίπεδο εξόδου. Εκεί υπολογίζει το διάνυσμα εξόδου του δικτύου καθώς και το σφάλμα του κάθε νευρώνα αυτού του επιπέδου. Η Εικόνα 5.1 αναπαριστά το ευθύ πέρασμα.



Εικόνα 5.1 Πολυεπίπεδο τεχνητό νευρωνικό δίκτυο

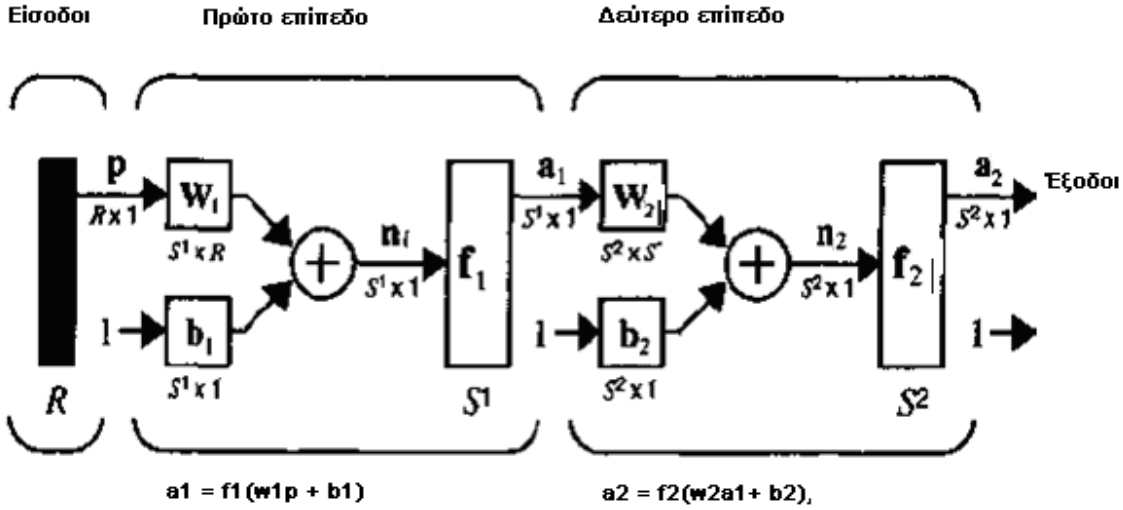
Το αντίστροφο πέρασμα (reverse pass) ξεκινά από το επίπεδο εξόδου περνώντας τις τιμές σφάλματος προς τα πίσω υπολογίζοντας αναδρομικά την τιμή της τοπικής κλίσης της εισόδου του νευρώνα για κάθε νευρώνα. Έτσι τροποποιούνται τα συναπτικά βάρη και οι τιμές των πολώσεων των νευρώνων, σύμφωνα με το γενικευμένο κανόνα δέλτα. Στην Εικόνα 5.2 αναπαρίσταται το ευθύ και το αντίστροφο πέρασμα.



Εικόνα 5.2 Τιμές λειτουργίας και σφάλματος

Τα δεδομένα εισόδου εμφανίζονται στην είσοδο του δικτύου και η ροή τους είναι από την είσοδο προς το επίπεδο εξόδου όπου και παράγεται το διάνυσμα εξόδου. Το διάνυσμα εισόδου χρησιμοποιείται κατά την κανονική λειτουργία του δικτύου. Δηλαδή κατά το πέρασμα των τιμών του διανύσματος από κάθε νευρώνα υλοποιείται ένας υπολογισμός. Το διάνυσμα σφάλματος δημιουργείται κατά τη φάση της εκπαίδευσης και μόνο στην έξοδο του δικτύου. Η κατεύθυνση που ακολουθεί είναι η ανάστροφη του διανύσματος λειτουργίας.

Μπορούμε να πούμε ότι τα Backpropagation δίκτυα, είναι feed-forward τεχνητά νευρωνικά δίκτυα δομημένα σε επίπεδα. Εκπαιδεύονται βασιζόμενα στη μέθοδο με επίβλεψη χρησιμοποιώντας τον αλγόριθμο ανάστροφης διάδοσης, που στηρίζεται στο κανόνα διόρθωσης σφάλματος. Οι υπολογισμοί όπως είπαμε γίνονται σε δύο στάδια το ευθύ και το αντίστροφο. Αυτού του είδους τα δίκτυα χρησιμοποιούν τουλάχιστον ένα κρυφό επίπεδο νευρώνων. Το πρώτο επίπεδο χρησιμοποιείται για την εισαγωγή των δεδομένων κι εκεί δε γίνεται καμία επεξεργασία. Απλά μεταφέρονται τα δεδομένα στο αμέσως επόμενο κρυφό επίπεδο. Εκεί χρησιμοποιείται από κάθε νευρώνα μία μη γραμμική συνάρτηση ενεργοποίησης η οποία και υπολογίζει τα νέα σταθμισμένα δεδομένα στο επόμενο επίπεδο νευρώνων. Οι συναρτήσεις που χρησιμοποιεί ο κάθε νευρώνας σε κρυφό επίπεδο μπορούν να είναι διαφορετικές μεταξύ τους, απαραίτητη προϋπόθεση να είναι παραγωγίσιμες. Αυτή η διαδικασία γίνεται σε κάθε κρυφό επίπεδο. Τελικά, τα δεδομένα φτάνουν στον επίπεδο παραγωγής (ή αλλιώς εξόδου), όπου κι εκεί μέσω μιας συνάρτησης ενεργοποίησης υπολογίζονται οι έξοδοι. Αυτές οι έξοδοι θα χρησιμοποιηθούν για την τροποποίηση των βαρών και των πολώσεων. Είναι σημαντικό να σημειώσουμε ότι δεν υπάρχουν συνδέσεις ανάμεσα στις μονάδες των επιπέδων. Η Εικόνα 5.3 εξηγεί πως θα υπολογιστεί το αποτέλεσμα της εξόδου κατά το ευθύ πέρασμα



Εικόνα 5.3 Υπολογισμός τιμών σε κάθε νευρώνα

Το αποτέλεσμα της εξόδου δίδεται από τον τύπο $\mathbf{a_2 = f_2(w_2 a_1 + b_2)}$, όπου $\mathbf{a_1 = f_1(w_1 p + b_1)}$. Το πρώτο επίπεδο είναι το κρυφό επίπεδο και το δεύτερο επίπεδο είναι το επίπεδο εξόδου. Οπότε και πρέπει τώρα να υπολογίσουμε τα νέα βάρη, βάσει του λάθους που προκύπτει στη έξοδο. Η εξήγηση ακολουθεί παρακάτω στη παράγραφο 5.3.

5.2 Συνοψίζοντας τα χαρακτηριστικά ενός πολυστρωματικού perceprton με όπισθεν διάδοση του σφάλματος.

1) Κάθε κρυμμένος νευρώνας έχει μια παραγωγίσιμη συνάρτηση ενεργοποίησης. Παράδειγμα τέτοιας συνάρτησης είναι η υπερβολική εφαπτομένη. Η ύπαρξη της μη γραμμικότητας προσδίδει σε αυτού του είδους τα δίκτυα τις επιθυμητές υπολογιστικές δυνατότητες.

2) Εκτός από τα επίπεδα εισόδου και εξόδου θα πρέπει να υπάρχει τουλάχιστον ένα επίπεδο κρυφών νευρώνων. Έτσι το δίκτυο γίνεται πιο ικανό στη μάθηση. Βασικό χαρακτηριστικό είναι η απουσία συνδέσεων μεταξύ νευρώνων του ίδιου επιπέδου. Συνήθως, ένα τέτοιο δίκτυο είναι πλήρως διασυνδεδεμένο. Δηλαδή, οι νευρώνες του προηγούμενου επιπέδου συνδέονται με όλους τους νευρώνες του επόμενου επιπέδου.

5.3 Ο Γενικευμένος Κανόνας Δέλτα

Η έξοδος δίνεται από τον τύπο

$y^p_k = F(s^p_k)$, (1) όπου

$s^p_k = \sum w_{jk} y_j^p + \theta_k$, (2)

Και $\theta_k = b_k$

Θέτουμε

$\Delta_p W_{jk} = -\gamma(\partial E^p / \partial w_{jk})$,

όπου E^p είναι το μέσο συνολικό τετραγωνικό σφάλμα του προτύπου P σε σχέση με τις μονάδες εξόδου. Ο τύπος που δίνει το μέσο τετραγωνικό σφάλμα είναι ο

$$E^p = 1/2 \sum_{0=1}^{N0} (d_0^p - y_0^p)^2,$$

όπου ο = 1 είναι η έξοδος στο πρώτο νευρώνα στο στρώμα παραγωγής και Νο είναι η έξοδος του Ν-οστού νευρώνα, dp είναι η επιθυμητή έξοδος από τη μονάδα εξόδου του πρότυπου Ρ και y^p είναι η πραγματική έξοδος. Οπότε το συνολικό τετραγωνικό σφάλμα για το σύνολο των προτύπων είναι ίσο με

$$E = \sum_p E^p$$

Άρα μπορούμε να γράψουμε

$$\partial E^p/\partial w_{jk} = (\partial E^p/\partial s_k^p)*(\partial s_k^p/\partial w_{jk}), \quad (3)$$

Ισχύει ότι

$$\partial s_k^p/\partial w_{jk} = y_j^p, \quad (4)$$

Θέτουμε

$$\delta_k^p = -(\partial E^p/\partial s_k^p), \quad (5)$$

Έτσι μπορούμε να υπολογίσουμε τις αλλαγές στα βάρη σύμφωνα με τον τύπο:

$$\mathbf{\Delta_p W_{jk} = \gamma^* \delta_k^p * y_j^p}, \quad (6)$$

Ο όρος δ_k^p είναι η τοπική κλίση.

Έτσι η διόρθωση $\Delta_p w_{jk}(n)$ του βάρους που συνδέει τον νευρώνα ***j*** με τον νευρώνα *k* του επόμενου επιπέδου, μετά την n-οστή επανάληψη, ορίζεται από το γενικευμένο κανόνα δέλτα (τύπος (6)). Το ενδιαφέρον είναι ο επαναλαμβανόμενος υπολογισμός των δ_k^p, που μπορεί να εφαρμοστεί με τη διάδοση σημάτων λάθους προς τα πίσω στο δίκτυο. Για να υπολογίσουμε τον όρο δ_k^p τον γράφουμε ως γινόμενο δύο παραγόντων:

$$\delta_k^p = -(\partial E^p/\partial s_k^p) = -[(\partial E^p/\partial y_k^p)*(\partial y_k^p/\partial s_k^p)], \quad (7)$$

Ο πρώτος όρος περιγράφει τη μεταβολή του λάθους κι ο δεύτερος περιγράφει την αλλαγή της εξόδου χρησιμοποιώντας κάποια συνάρτηση. Ο δεύτερος όρος ισούται με $\partial y_k^p/\partial s_k^p = F(s_k^p)$, (8)

Για να υπολογίσουμε τον πρώτο όρο, δηλαδή τον όρο $\partial y_k^p/\partial s_k^p$ εξετάζουμε δύο περιπτώσεις. Πρώτα απ' όλα αν η μονάδα k είναι μονάδα εξόδου του δικτύου. Δηλαδή να ισχύει ότι κ = ο (output). Σε αυτή τη περίπτωση υπολογίζεται από τον τύπο

$$\partial E^p/\partial y_0^p = -(d_0^p - y_0^p).$$

Οπότε ο τύπος (7) παίρνει τη μορφή

$\delta_0^p = (d_0^p - y_0^p)* F_0'(s_0^p)$, (9).

Η δεύτερη περίπτωση είναι η μονάδα k να είναι κρυφή μονάδα (hidden unit), δηλαδή k = h (hidden). Το μέτρο του λάθους μπορεί να γραφεί ως συνάρτηση των μονάδων εισόδου από το κρυφό προς το εξωτερικό επίπεδο. Δηλαδή, ως εξής
$E^p = E^p (s_1^p, s_2^p, \ldots, s_j^p, \ldots)$
Χρησιμοποιώντας το κανόνα της αλυσίδας γράφουμε το τύπο :

$$\partial E^p/\partial y_h^p = \sum_{o=1} (\partial E^p/\partial s_0^p) * (\partial s_0^p/\partial y_h^p) = \sum_{o=1}(\partial E^p/ \partial s_0^p)*(\partial E^p/d\partial y_h^p)*\sum_{j=1} w_{k0} y_j^p =$$

$$\sum^{N0} (\partial E^p/ \partial s_0^p * w_{h0}) = - \sum^{N0} \delta_0^p w_{h0}$$

Οπότε ο τύπος (7) παίρνει τη μορφή

$$\delta_h^p = F'(s_h^p)\sum_{0=1}^{N0} \delta_0^p * w_{h0}, \quad (10)$$

Έτσι καταλήξαμε στο να μάθουμε πως υπολογίζουμε την τοπική κλίση από κάποιον νευρώνα αν αυτός είναι νευρώνας κρυφού επιπέδου ή αν αυτός είναι νευρώνας του επιπέδου της εξόδου.

5.4 Μία συνοπτική παρουσίαση

Όλες οι παραπάνω εξισώσεις φαίνονται περίπλοκες, ωστόσο υπάρχει μια πιο σαφής εξήγηση. Το πρότυπο εισόδου μετασχηματίζεται μέσω των νευρώνων και καταλήγει ως έξοδος. Στη συνέχεια η κάθε έξοδος συγκρίνεται με τις επιθυμητές εξόδους και καταλήγουμε σε ένα λάθος για κάθε μια από τις μονάδες παραγωγής. Αυτό το λάθος το καλούμε e_o και αντιστοιχεί στη μονάδα εξόδου ο. Σκοπός είναι το e_o να μηδενιστεί. Προσπαθούμε να αλλάξουμε τις συνδέσεις στο δίκτυο κατά τέτοιο τρόπο ώστε το λάθος e_o να μηδενιστεί την επόμενη φορά. Ο μόνος τρόπος βέβαια να ελαττώσουμε το λάθος είναι να προσαρμόσουμε τα βάρη χρησιμοποιώντας τον τύπο $\Delta w_{ho} = (d_o - y_o)* y_h$. Αλλά δεν αρκεί διότι τα βάρη των κρυφών επιπέδων δεν αλλάζουν. Έτσι για να υπολογίσουμε τις τιμές των βαρών των μονάδων των κρυφών επιπέδων χρησιμοποιούμε το γενικευμένο κανόνα δέλτα. Το πρόβλημα λύνεται κάνοντας χρήση του κανόνα της αλυσίδας. Διανέμεται έτσι το λάθος των μονάδων εξόδου σε όλες τις κρυμμένες μονάδες με τις οποίες αυτές συνδέονται. Μια κρυφή μονάδα h λαμβάνει ένα δ από κάθε μονάδα εξόδου ο με την οποία συνδέεται, το οποίο δ που λαμβάνει ισούται με το σταθμισμένο δ της μονάδας εξόδου. Το δ σταθμίζεται βάσει της τιμής του βάρους που υπάρχει μεταξύ των

συνδεδεμένων μονάδων κάθε φορά. Συμβολίζεται $\delta_h = \Sigma\delta_o w_{ho}$. Βασικές προϋποθέσεις για να πραγματοποιηθεί η όπισθεν διάδοση του λάθους είναι η συνάρτηση ενεργοποίησης (η παράγωγος της συνάρτησης ενεργοποίησης) να αποδεχτεί το νέο δ.

5.5 Βήματα backpropagation (μέθοδος εκπαίδευσης ανά σύνολο προτύπων ή αλλιώς batch training)

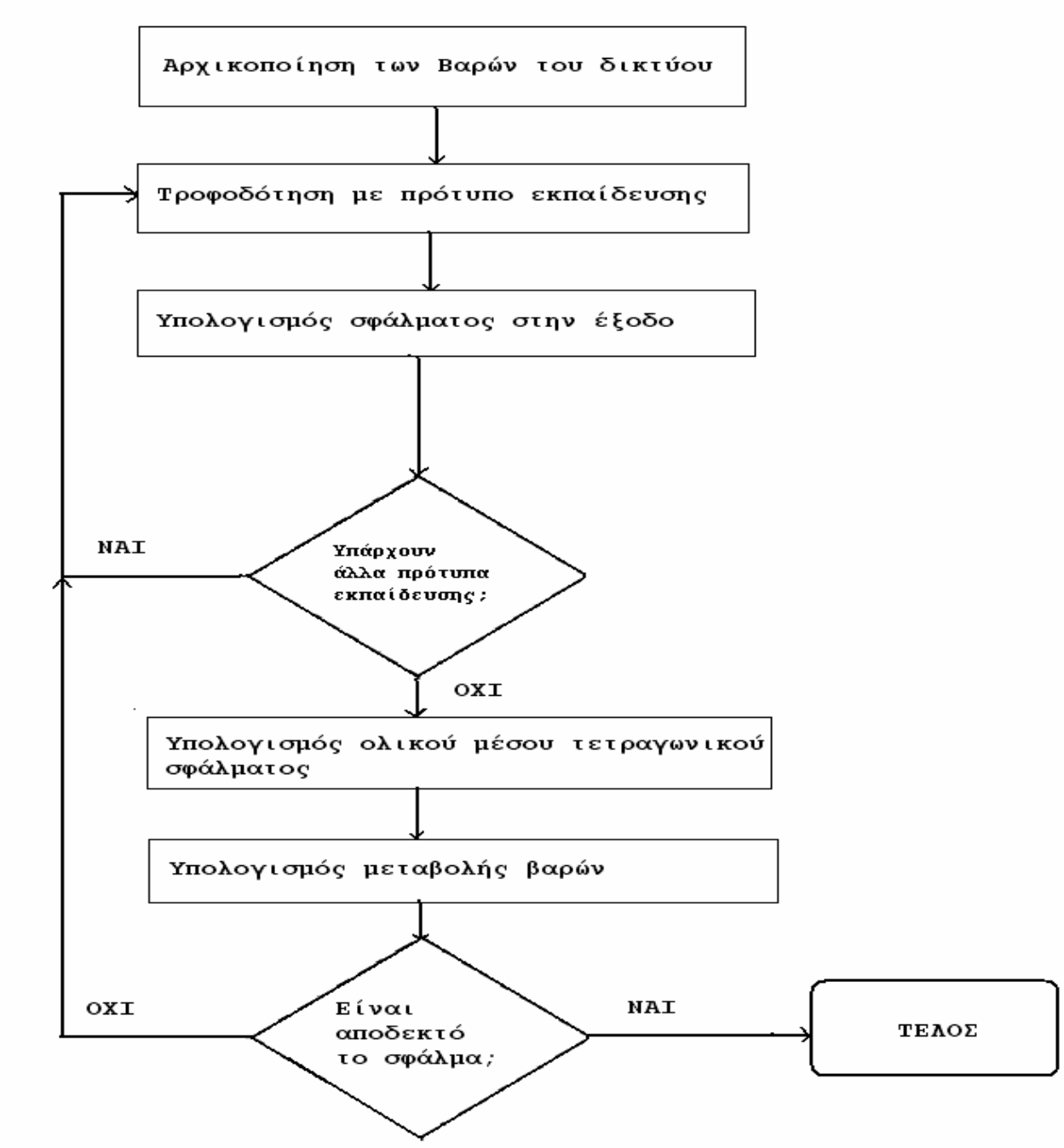
Ο αλγόριθμος υλοποιείται ως εξής:

1. Διάλεξε τα αρχικά βάρη και κατώφλια χρησιμοποιώντας μικρές θετικές τυχαίες τιμές.
2. Παρουσίασε στο ΤΝΔ το διάνυσμα εκπαίδευσης και το διάνυσμα των στόχων.
3. Υπολόγισε τα σήματα εξόδου όλων των νευρώνων του δικτύου νευρώνα-νευρώνα προς τα εμπρός χρησιμοποιώντας τις τρέχουσες τιμές των συναπτικών βαρών.
4. Υπολόγισε το τετραγωνικό σφάλμα και αποθηκευσέ το.
5. Επανέλαβε το βήμα 2 έως ότου τρέξουν στο δίκτυο όλα τα πρότυπα
6. Υπολόγισε το συνολικό μέσο τετραγωνικό σφάλμα.
7. Ανανέωσε (βελτίωσε) τα βάρη αρχίζοντας από τους νευρώνες της εξόδου και προχωρώντας ανάστροφα προς το στρώμα εισόδου, χρησιμοποιώντας τον κανόνα Backpropagation, ελέγχοντας αν ο νευρώνας ανήκει σε κρυφό επίπεδο ή είναι νευρώνας εξόδου, ώστε να χρησιμοποιηθεί ο κατάλληλος τύπος για τον υπολογισμό της τοπικής κλίσης, δηλαδή του δ.

5.6 Εκπαίδευση back-propagation

Δίνουμε αρχικά στο τεχνητό νευρωνικό δίκτυο δύο διανύσματα. Το πρώτο διάνυσμα είναι το διάνυσμα των δεδομένων που θα τρέξει το δίκτυο και το δεύτερο διάνυσμα, είναι το διάνυσμα με τις επιθυμητές τιμές που θέλουμε εμείς το δίκτυο να παρουσιάσει στις εξόδους. Όταν τρέξουν όλα τα πρότυπα στο δίκτυο το χρονικό διάστημα ονομάζεται εποχή. Η διαδικασία της εκπαίδευσης εκτελείται σε επαναλήψεις εποχών έως ότου οι τιμές των βαρών του δικτύου σταθεροποιηθούν σε συγκεκριμένες τιμές και το σφάλμα να είναι το ελάχιστο που μπορεί πλέον να υπολογιστεί. Δηλαδή το μέσο ολικό τετραγωνικό σφάλμα να συγκλίνει στην ελάχιστη τιμή. Έχει περισσότερο ενδιαφέρον τα πρότυπα σε κάθε εποχή να δίνονται τυχαία, γιατί έτσι γίνεται στοχαστικά η αναζήτηση των νέων τιμών των βαρών. Η διαδικασία της εκπαίδευσης μπορεί να πραγματοποιηθεί με δύο δυνατούς τρόπους. Ο πρώτος τρόπος ονομάζεται εκπαίδευση ανά ομάδα προτύπων (batch training), ενώ ο δεύτερος ονομάζεται εκπαίδευση ανά πρότυπο (on-line training).

Εκπαίδευση ανά ομάδα προτύπων (Εικόνα 5.4) σημαίνει ότι τα βάρη τροποποιούνται αφού τρέξουν όλα τα πρότυπα στο δίκτυο. Έτσι υπολογίζεται το ολικό μέσο τετραγωνικό σφάλμα για όλα τα πρότυπα και αυτό διαδίδεται προς τα πίσω κατά τα γνωστά. Όταν τρέξουν στο δίκτυο όλα τα πρότυπα και το σφάλμα διαδοθεί προς τα πίσω για τη μεταβολή των τιμών βαρών των, τότε ολοκληρώνεται μια εποχή [1], [2], [20].



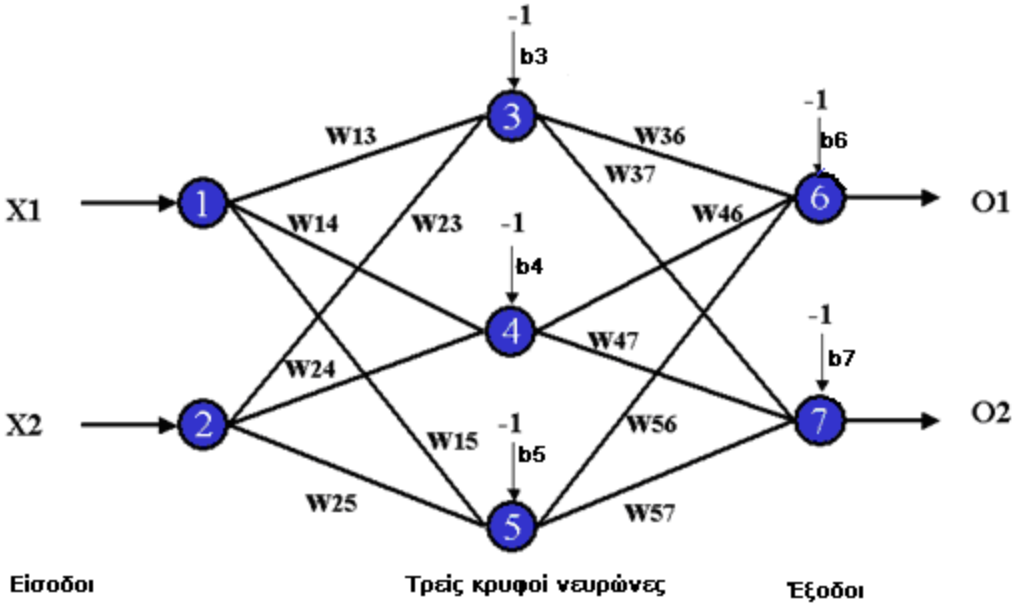
Εικόνα 5.4 Σχήμα επεξήγησης του υπολογισμού σφάλματος ανά ομάδα προτύπων

Εκπαίδευση ανά πρότυπο σημαίνει ότι για κάθε πρότυπο που τρέχει στο δίκτυο υπολογίζεται το μέσο τετραγωνικό σφάλμα και αυτό διαδίδεται προς τα πίσω επίσης κατά τα γνωστά. Όταν αυτή η διαδικασία πραγματοποιηθεί για όλα τα πρότυπα του συνόλου τότε ολοκληρώνεται μια εποχή. Και στις δυο παραπάνω περιπτώσεις που περιγράψαμε υπάρχουν θετικά και αρνητικά στοιχεία. Αυτό όμως κάθε φορά το κρίνει η φύση της εφαρμογής.

5.6.1 Παράδειγμα εκπαίδευσης δικτύου βάσει του αλγορίθμου back-propagation με τη μέθοδο batch training

Στην Εικόνα 5.5 δίνεται ένα πολυεπίπεδο ΤΝΔ τοπολογίας 2-3-2. Για την εκπαίδευσή του χρησιμοποιείται η μέθοδος όπισθεν-διάδοσης του σφάλματος με ρυθμό εκπαίδευσης n = 0.3. Η συνάρτηση ενεργοποίησης σε όλους τους νευρώνες είναι η σιγμοειδής συνάρτηση S, όπου:

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



Εικόνα 5.5 Υπολογισμός βαρών με παράδειγμα

Πίνακας 5.1 Τιμές βαρών

Βάρος	Τιμή	Βάρος	Τιμή
w13 =	0,2	w36 =	0,1
w14 =	0,2	w37 =	0,3
w15 =	0,2	w46 =	0,1
w23 =	0,3	w47 =	0,3
w24 =	0,3	w56 =	0,1
w25 =	0,3	w57 =	0,3
w30 = b3	0,4	w60 =	0,4
w40 = b4	0,4	w70 =	0,4
w_{50} = b5	0,4		

Έστω ότι θέλουμε να εκπαιδεύσουμε το δίκτυο. Η τιμή του διανύσματος εκπαίδευσης είναι [0.2,0.8] με επιθυμητή έξοδο [0.0, 1.0]. Τα βάρη των συνδέσεων έχουν πάρει τις τιμές που δίνονται στον Πίνακα 5.1.

Πρώτος στόχος είναι να υπολογίσουμε την έξοδο του δικτύου για το δοσμένο πρότυπο

Η μέθοδος επίλυσης είναι η ακόλουθη:

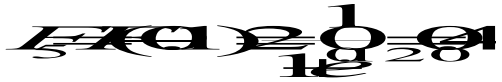
Άθροισμα στο κόμβο 3:
(w13*x1) + (w23* x2) +(-1)* w30 = 0.2*0.2+0.3*0.8-0.4 =-0.120

Ενεργοποίηση στον κόμβο 3: 

Άθροισμα στο κόμβο 4:
(w14*x1) + (w24* x2) +(-1)* w40 = 0.2*0.2+0.3*0.8-0.4 =-0.120

Ενεργοποίηση στον κόμβο 4: 

Άθροισμα στο κόμβο5:
(w15*x1) + (w25* x2) +(-1)* w50 = 0.2*0.2+0.3*0.8-0.4 =-0.120

Ενεργοποίηση στον κόμβο 5: 

Άθροισμα στο κόμβο 6: (w36* F3)+ (w46* F4) + (w56* F5) +(-1)* w60 =
0.1*0.470+0.1*0.470 +0.1*0.470 – 0.4 = -0.259
Ενεργοποίηση στον κόμβο 6: F6=F(-0.259) = 0.436

Άθροισμα στο κόμβο 7: (w37* F3)+ (w47* F4) + (w57* F5) +(-1)* w70 =
0.3*0.470+0.3*0.470 +0.3*0.470 – 0.4 = 0.023
Ενεργοποίηση στον κόμβο 7: F7=F(0.023) = 0.506

Τελικά η έξοδος του ΤΝΔ με τα δοθέντα βάρη είναι (0.436,0.506).

Δεύτερο βήμα είναι να κάνουμε χρήση του αλγορίθμου όπισθεν διάδοσης του σφάλματος (backpropagation), βάσει των τιμών που υπολογίσαμε στο πρώτο βήμα.

Ανανέωση των συνδέσεων μεταξύ κρυφού και επιπέδου εξόδου

Η έξοδος του ΤΝΔ έπρεπε να ήταν [0.0,1.0] για το πρότυπο εισόδου [0.2,0.8]. Άρα το σφάλμα στην έξοδο είναι:

Για τον κόμβο 6: 0-0.436= -0.436
Για τον κόμβο 7: 1-0.506= 0.494

Ο υπολογισμός των βαρών ακολουθεί την παρακάτω διαδικασία.

Για τους νευρώνες στο επίπεδο εξόδου ισχύει:

Για το πρόβλημά μας έχουμε:

 . Άρα:

 , όπου t = target =0.0 και o=output=0.436.

Ομοίως:  , όπου t = target =1.0 και o=output=0.506.

Για τον υπολογισμό των νέων τιμών w36, w37, w46, w47, w56 και w57 ισχύουν:

Ομοίως:

Όπου: n = 0.3 είναι ο ρυθμός εκπαίδευσης

F3 είναι η ενεργοποίηση της σιγμοειδούς στον κόμβο 3,
F4 είναι η ενεργοποίηση της σιγμοειδούς στον κόμβο 4 και
F5 είναι η ενεργοποίηση της σιγμοειδούς στον κόμβο 5.

Για τις νέες τιμές των κατωφλίων ισχύει:

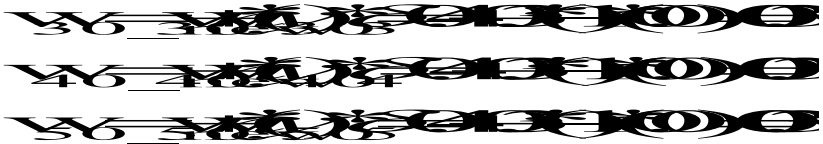
Ανανέωση των συνδέσεων μεταξύ κρυφού και επιπέδου εισόδου
Ο υπολογισμός των βαρών ακολουθεί την παρακάτω διαδικασία.

Για τους νευρώνες στο κρυφό επίπεδο ισχύει:

Οπότε για τον κόμβο 3 έχουμε:

δ3=F3*(1-F3)*[(δ6*w36)+(δ7*w37)] = 0.470*(1-0.470)* [(-0.107 *0.1)+(0.124*0.3)] =0.007.
Ομοίως:
δ4=F4*(1-F4)* [(δ6*w46)+(δ7*w47)] =0.007
δ5=F5*(1-F5)* [(δ6*w56)+(δ7*w57)] =0.007

Για τις νέες τιμές των κατωφλίων ισχύει:



Για τις νέες τιμές των βαρών w_{13}, w_{14}, w_{15}, w_{23}, w_{24}, w_{25} ισχύει:

$$w_{13\_new} = w_{13} + n*(x_1)*\delta_3 = 0.2 + 0.3*0.2*(0.007) = 0.200$$
$$w_{14\_new} = w_{14} + n*(x_1)*\delta_4 = 0.2 + 0.3*0.2*(0.007) = 0.200$$
$$w_{15\_new} = w_{15} + n*(x_1)*\delta_5 = 0.2 + 0.3*0.2*(0.007) = 0.200$$
$$w_{23\_new} = w_{23} + n*(x_2)*\delta_3 = 0.3 + 0.3*0.8*(0.007) = 0.302$$
$$w_{24\_new} = w_{24} + n*(x_2)*\delta_4 = 0.3 + 0.3*0.8*(0.007) = 0.302$$
$$w_{25\_new} = w_{25} + n*(x_2)*\delta_5 = 0.3 + 0.3*0.8*(0.007) = 0.302$$

όπου n = 0,3 6 ρυθμός εκπαίδευσης, x_2 και x_1 οι είσοδοι του τεχνητού νευρωνικού δικτύου [4], [12].

5.7 Ρυθμίσεις των βαρών χρησιμοποιώντας τη σιγμοειδής συνάρτηση ενεργοποίησης

Το βάρος μιας σύνδεσης ρυθμίζεται ανάλογα του σήματος λάθους δ. Η μονάδα k λαμβάνει την είσοδο και η έξοδος από τη μονάδα j στέλνεται κατά μήκος της σύνδεσης. Ο τύπος που περιγράφει τη διαδικασία είναι $\Delta_p w_{jk} = \gamma * \delta_k^{p*} y_j^p$. Αν η μονάδα είναι μονάδα εξόδου τότε το σήμα λάθους δίνεται από τον τύπο $\delta^P = (d^p - y^P) * F'(s^P)$ (1).

Έστω ότι $y^p = F(s^p) = 1/(1 + e^{-k})$, $k=s^p$

Αν παραγωγίσουμε την $F(s^p)$ θα προκύψει ότι:

$F'(k) = (\partial/\partial k) * (1/(1 + e^{-k}))$

$= (1/(1 + e^{-k})^2) * (-e^{-k})$

$= (1/(1 + e^{-k})) * ((-e^{-k})/(1 + e^{-k}))$

$= y^p * (1 - y^p)$

Οπότε η (1) εξίσωση γράφεται

$\delta^P = (d^p - y^P) * y^{P*} (1-y^P)$.

Αν πρόκειται για μονάδα κρυφού επιπέδου τότε ο υπολογισμός του δ γίνεται ως εξής:

$$\delta_h{}^p = F(s_h^p) \sum_{0=1}^{N0} \delta_0^p w_{h0} = y_h^p (1-y_h^p) \sum_{0=1}^{N0} \delta_0^p w_{h0},$$

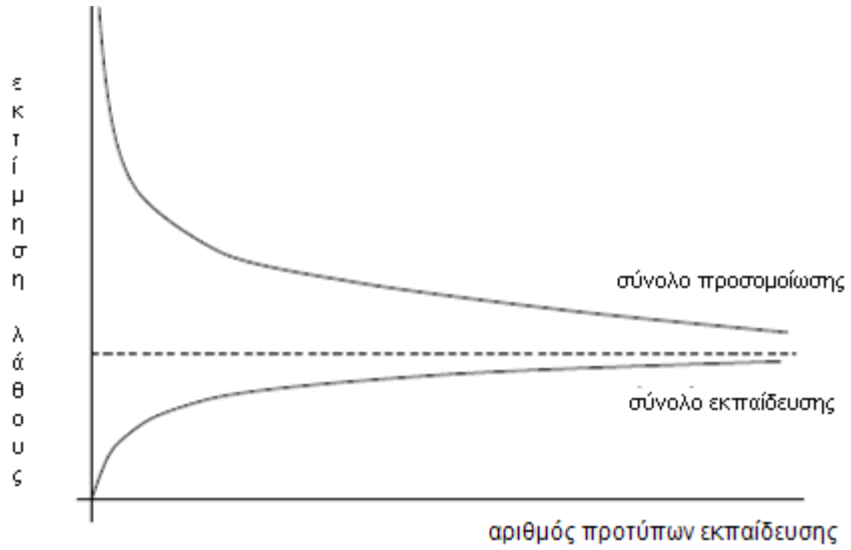
όπου h η κρυφή μονάδα.

5.8 Γενίκευση και ανεκτικότητα σε βλάβες

Δύο βασικά χαρακτηριστικά των back-propagation ΤΝΔ είναι η γενίκευση και η ανεκτικότητα σε βλάβες. Είναι δύο χαρακτηριστικά που δίνουν στις υπολογιστικές μηχανές περισσότερη ευελιξία και δυνατότητες. Θα μπορούσε να πει κάποιος ότι είναι δύο από τα πολλά χαρακτηριστικά του ανθρώπου. Δίνοντας αυτά τα χαρακτηριστικά και στους υπολογιστές καταφέρνουμε και να αυξήσουμε την υπολογιστική τους δύναμη αλλά και να τους κάνουμε να μοιάζουν έστω και λίγο στον άνθρωπο. Αυτή είναι και η τάση που παρατηρείται τα τελευταία χρόνια. Να φτιαχτούν μηχανές οι οποίες να ‘‘σκέφτονται’’ και να κινούνται όπως οι άνθρωποι. Τα τεχνητά νευρωνικά δίκτυα με τα χαρακτηριστικά της γενίκευσης και της ανεκτικότητας είναι ένα μέσο προς αυτή τη τάση [19].

5.8.1 Γενίκευση

Γενίκευση (generalization) είναι η δυνατότητα του τεχνητού νευρωνικού δικτύου να ταξινομεί πρότυπα τα οποία δε γνώρισε ποτέ, αλλά είναι παρόμοια με αυτά που το δίκτυο εκπαιδεύτηκε. Η γενίκευση επιτυγχάνεται λόγω των ιδιοτήτων των προτύπων εισόδου που έχουν κωδικοποιηθεί στους νευρώνες κατά την εκπαίδευση. Ένα παρόμοιο πρότυπο με τα πρότυπα εκπαίδευσης ταξινομείται σε σχέση με τα πρότυπα που έχει κοινές ιδιότητες. Επίσης, το ΤΝΔ έχει τη δυνατότητα να ταξινομεί σχετικά ικανοποιητικά ελλιπή δεδομένα ή δεδομένα με θόρυβο. Η γενίκευση ενός ΤΝΔ επηρεάζεται από το μέγεθος και την καταλληλότητα του συνόλου εκπαίδευσης, την αρχιτεκτονική του δικτύου και την πολυπλοκότητα του προβλήματος. Για να αντιμετωπίσουμε τα παραπάνω προβλήματα οι ενέργειες στις οποίες προβαίνουμε κατά αντιστοιχία είναι: πρώτα απ’ όλα επιλέγουμε όσο το δυνατόν περισσότερο αντιπροσωπευτικό (representative) σύνολο δεδομένων εκπαίδευσης. Και στη συνέχεια αφού μελετήσουμε προσεκτικά το πρόβλημα προσπαθούμε να επιλέξουμε την καλύτερη δυνατή αρχιτεκτονική δικτύου [17].



Εικόνα 5.6 Επίδραση του καθορισμένου μεγέθους εκπαίδευσης στο ποσοστό λάθους. Το μέσο ποσοστό λάθους και το μέσο ποσοστό λάθους δοκιμής ως λειτουργία του αριθμού δειγμάτων εκπαίδευσης.

5.8.2 Ανεκτικότητα σε βλάβες

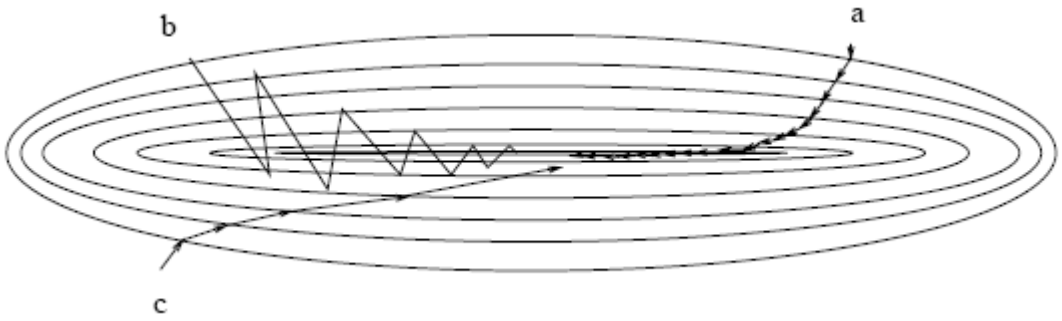
Γενικότερα ένα πολυεπίπεδο τεχνητό νευρωνικό δίκτυο προσοτροφοδότησης είναι ενδογενώς ανεκτικό σε βλάβες. Αυτό συμβαίνει διότι είναι ένα σύστημα παράλληλης επεξεργασίας, όπου το τελικό αποτέλεσμα είναι συνδυασμός όλων των εξόδων των υπολογιστικών κόμβων από κάθε επίπεδο. Οπότε ακόμη κι αν κάποιος κόμβος για κάποιο λόγο δε παράγει τιμή εξόδου ή τα συναπτικά βάρη υποστούν βλάβη, από το τελικό αποτέλεσμα θα μπορούμε να καταλήξουμε σε κάποιο συμπέρασμα. Βέβαια η ποιότητα της πληροφορίας που θα λάβουμε δεν θα είναι η ίδια αν το δίκτυο λειτουργούσε κανονικά. Όλα αυτά βέβαια αν το δίκτυο δεν έχει υποστεί ολοκληρωτική βλάβη. Επομένως η ποιότητα της παραγόμενης πληροφορίας μειώνεται σε αξία αλλά όχι καταστροφικά. Αν το δίκτυο τελικά υποστεί βλάβη ο τρόπος να διορθώσουμε αυτή τη βλάβη είναι να προβούμε σε επανεκπαίδευση του δικτύου [17].

5.9 Ρυθμός και ορμή εκπαίδευσης

Η διαδικασία εκπαίδευσης απαιτεί ότι η αλλαγή της τιμής του βάρους είναι ανάλογη του όρου $\partial E^p/\partial w$, όπου η κλίση πιο απότομης καθόδου απαιτεί να λαμβάνονται απειροελάχιστα βήματα. Το βήμα είναι ο ρυθμός εκπαίδευσης γ. Για πρακτικούς λόγους επιλέγουμε ο ρυθμός εκπαίδευσης να είναι όσο το δυνατόν μεγαλύτερος χωρίς να οδηγεί σε ταλάντωση. Για να αποφευχθεί η ταλάντωση πρέπει να καταστεί η αλλαγή της τιμής του βάρους εξαρτώμενη της προηγούμενης αλλαγής βάρους με την προσθήκη ενός όρου, του όρου ορμής, δηλαδή:

$\Delta w_{jk}(t+1) = \gamma \delta_k^p x_j^p + \alpha \Delta w_{jk}(t)$.

Όπου α είναι σταθερά που καθορίζει την επίδραση της προηγούμενης αλλαγής της τιμής του βάρους. Ο ρόλος του όρου ορμής παρουσιάζεται στην Εικόνα 5.7. Όταν δε χρησιμοποιείται ο όρος ορμής, παίρνει αρκετό χρονικό διάστημα ώστε να επιτευχθεί η ελάχιστη τιμή βάρους με ένα χαμηλό ρυθμό εκπαίδευσης, ενώ για υψηλούς ρυθμούς εκπαίδευσης η ελάχιστη τιμή του βάρους δεν επιτυγχάνεται ποτέ λόγω των ταλαντώσεων. Κατά προσθήκη του όρου ορμής, η ελάχιστη τιμή θα επιτευχθεί πιθανά γρηγορότερα.



Εικόνα 5.7 Το ελάχιστο στο διάστημα των βαρών, a: για μικρό ποσοστό εκπαίδευσης, b: για μεγάλο ποσοστό εκπαίδευησης: σημειώστε τις ταλαντώσεις, c: με μεγάλο ποσοστό εκπαίδευσης και τον όρο ορμής προστιθέμενο

Η προσθήκη αυτού του όρου συνήθως επιτρέπει ταχύτερη εκπαίδευση του τεχνητού νευρωνικού δικτύου. Η σταθερά α είναι ένας θετικός αριθμός στο διάστημα [0,1). Για την τιμή 0 έχουμε τον κλασσικό αλγόριθμο οπισθοδρομικής διάδοσης του σφάλματος. Ο όρος αυτός της ορμής επιτρέπει στον αλγόριθμο να κινηθεί με βάση τις πιο πρόσφατες τάσεις-κλίσεις της επιφάνειας του σφάλματος. Λειτουργώντας ως ένα κατωδιαβατό φίλτρο επιτρέπει στο δίκτυο να αγνοεί αμελητέα χαρακτηριστικά στην επιφάνεια του σφάλματος. Χωρίς αυτό τον όρο, ένα τεχνητό νευρωνικό δίκτυο μπορεί να παγιδευτεί σε κάποιο «ρηχό» τοπικό ελάχιστο (ένα ελάχιστο στη γειτονία του οποίου η βάθμωση έχει μικρή τιμή), ενώ με την προσθήκη, του δίνεται η δυνατότητα να «γλιστρήσει» από πάνω του.

Συνοψίζοντας,

1. Για να συγκλίνει ο Γενικευμένος Κανόνας Δέλτα θα πρέπει $0<|a|<1$. Όταν α = 0 τότε ο αλγόριθμος πίσω διάδοσης δεν χρησιμοποιεί τη σταθερά ορμής. Η σταθερά *α* μπορεί να πάρει και αρνητικές τιμές, όμως είναι απίθανο να χρησιμοποιηθεί στην πράξη.

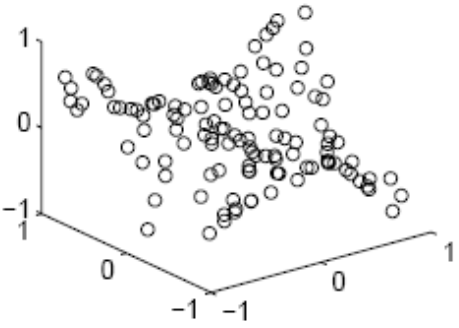
2. Η προσθήκη του όρου ορμής στον αλγόριθμο πίσω διάδοσης επιταχύνει τη σύγκλιση του αλγορίθμου σε περιπτώσεις που υπάρχουν σταθερές φθίνουσες κατευθύνσεις στο χώρο αναζήτησης.

3. Η προσθήκη του όρου ορμής στον αλγόριθμο πίσω διάδοσης σταθεροποιεί τη σύγκλιση του αλγορίθμου σε περιπτώσεις που υπάρχουν συνεχείς αλλαγές κατεύθυνσης (ανηφόρες και κατηφόρες) στο χώρο της αναζήτησης.

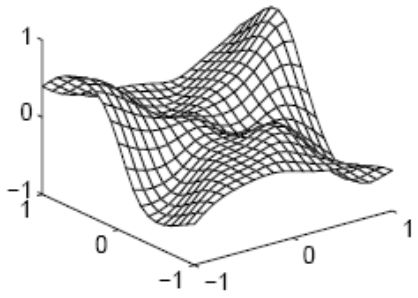
Η προσθήκη του όρου ορμής στον αλγόριθμο πίσω διάδοσης σφάλματος αποτελεί μια μικρή αλλαγή όσον αφορά την τροποποίηση των βαρών, όμως έχει πολλές θετικές επιδράσεις στην συμπεριφορά μάθησης του αλγορίθμου. Επίσης μπορεί να εμποδίσει τον πρόωρο τερματισμό της διαδικασίας σε ένα τοπικό ελάχιστο [8], [17].

5.10 Τοπικό ελάχιστο (ελάχιστη δυνατή τιμή του βάρους)

Η επιφάνεια λάθους ενός σύνθετου δικτύου είναι γεμάτη από λόφους και κοιλάδες. Λόγω της μεθόδου καθόδου της μέγιστης κλίσης, το δίκτυο μπορεί να παραμείνει παγιδευμένο σε ένα τοπικό ελάχιστο, ενώ υπάρχει εκεί κοντά ένα καλύτερο τοπικό ελάχιστο. Οι στοχαστικές μέθοδοι μπορούν να βοηθήσουν να αποφύγουν αυτήν την παγίδα, αλλά τείνουν να είναι αργές. Μια άλλη προτεινόμενη λύση είναι να αυξηθεί ο αριθμός κρυμμένων μονάδων. Αν και αυτό θα λειτουργήσει λόγω της υψηλότερης διαστατικότητας της συνάρτησης λάθους, αν και η πιθανότητα να πάρει παγιδευμένη τιμή είναι μικρότερη, φαίνεται να υπάρχει κάποιο ανώτερο όριο του αριθμού κρυμμένων μονάδων, που όταν ξεπερνιέται, οδηγεί πάλι το σύστημα να παγιδεύεται στα τοπικά ελάχιστα**.** Ένας άλλος τρόπος να αποφύγουμε το πρόβλημα των τοπικών ελάχιστων είναι να επαναπροσδιοριστούν οι τιμές των βαρών.



Εικόνα 5.8 Τιμές βαρών



Εικόνα 5.9 Επιφάνεια τιμών βαρών

5.11 Παράλυση δικτύου (network paralysis)

Ένα ή περισσότερα βάρη έχουν σταθερά υψηλές απόλυτες τιμές και δε τροποποιούνται σημαντικά σε κάθε επανάληψη. Αυτό έχει σαν αποτέλεσμα μεγάλο αριθμό κύκλων εκπαίδευσης. Μια λύση στο πρόβλημα αυτό είναι να αυξηθεί ο ρυθμός εκπαίδευσης. Υπάρχουν διάφορες «στρατηγικές» για τη μεταβολή του ρυθμού εκπαίδευσης, ανάλογα με την απόδοση του αλγορίθμου. Μία δυνατότητα είναι η παρακάτω:

1) Εάν το τετραγωνικό σφάλμα (υπολογισμένο για όλο το σύνολο εκπαίδευσης) αυξάνει πάνω από ένα προκαθορισμένο ποσοστό ζ (συνήθως 1%-5%) μετά την μεταβολή των βαρών, τα νέα βάρη απορρίπτονται, ο ρυθμός εκπαίδευσης πολλαπλασιάζεται με έναν παράγοντα $0<\rho<1$ και ο συντελεστής της ορμής (εάν χρησιμοποιείται) τίθεται ίσος με 0.

2) Εάν το τετραγωνικό σφάλμα μειώνεται μετά την μεταβολή των βαρών, τότε γίνεται αποδεκτή η μεταβολή των βαρών και ρυθμός εκπαίδευσης πολλαπλασιάζεται με έναν παράγοντα η >1, και ο συντελεστής της ορμής (εάν έχει μηδενιστεί προηγουμένως) τίθεται ίσος με την αρχική του τιμή.

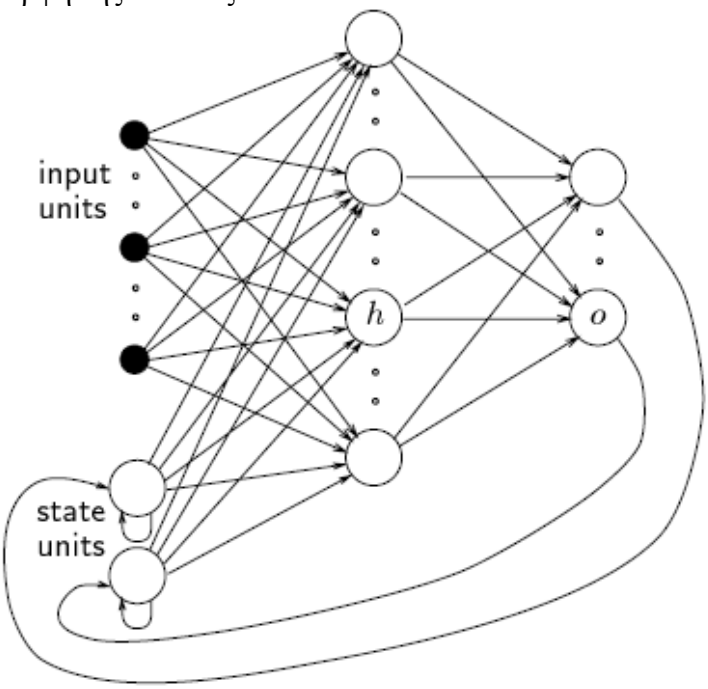
3) Εάν το τετραγωνικό σφάλμα αυξάνει αλλά όχι πάνω από το προκαθορισμένο ποσοστό ζ τα νέα βάρη γίνονται δεκτά, αλλά ο ρυθμός εκπαίδευσης και ο συντελεστής της ορμής παραμένουν αναλλοίωτοι [5], [17].

5.12 Εφαρμογές των backpropagation ΤΝΔ

Τα back-propagation ΤΝΔ έχουν χρησιμοποιηθεί επιτυχώς σε συστήματα πρόβλεψης (prediction), για την ταξινόμηση προτύπων (classification), την κατασκευή μοντέλων από δεδομένα (data fitting), για την επίλυση διαφορικών εξισώσεων, για τον έλεγχο συστημάτων. Ειδικότερα μπορούν να εφαρμοστούν σε ιατρικές εφαρμογές πρόβλεψης ασθενειών, σε μετεωρολογικές προβλέψεις, σε στρατιωτικές επιχειρήσεις καθώς και σε διαστημικές [9], [10], [11].

6 Δίκτυα με ανάδραση (recurrent networks)

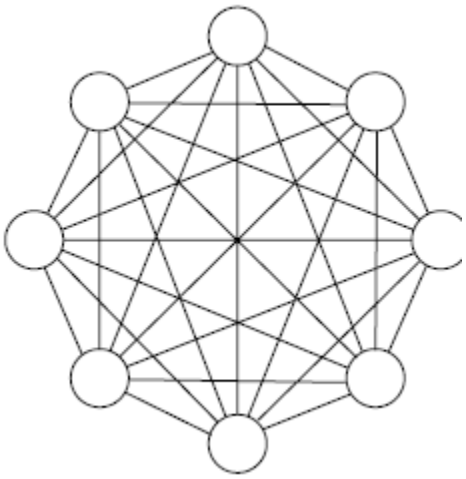
Αναφορικά θα αναφέρουμε κάποια τεχνητά νευρωνικά δίκτυα με ανάδραση. Δηλαδή δίκτυα τα οποία έχουν την ικανότητα να ανακυκλώνουν την πληροφορία. Τέτοια δίκτυα έχουν τη μορφή της Εικόνας 6.1.



Εικόνα 6.1 Παράδειγμα δικτύου με ανάδραση

6.1 Δίκτυα Hopfield

Τα δίκτυα Hopfield είναι ένα είδος τεχνητών νευρωνικών δικτύων που αναφέρθηκαν στη βιβλιογραφία και περιγράφηκαν ανεξάρτητα από τους Άντερσον (Άντερσον, 1977) και Kohonen (Kohonen, 1977) το 1977.
Ένα δίκτυο Hopfield αποτελείται από μια ομάδα νευρώνων με συνδέσεις μεταξύ κάθε μονάδας i και j, για κάθε i μονάδα διαφορετική από τη j (Εικόνα 6.2). Το 1982, ο Hopfield συγκεντρώνει διάφορες προηγούμενες ιδέες σχετικά με αυτά τα δίκτυα και παρουσιάζει μια πλήρη μαθηματική ανάλυση (Amit, Gutfreund και Sompolinsky, 1986). Έτσι τα δίκτυα αυτά ονομάστηκαν δίκτυα Hopfield [17].



Εικόνα 6.2 Όλοι οι νευρώνες είναι και είσοδοι και έξοδοι του δικτύου

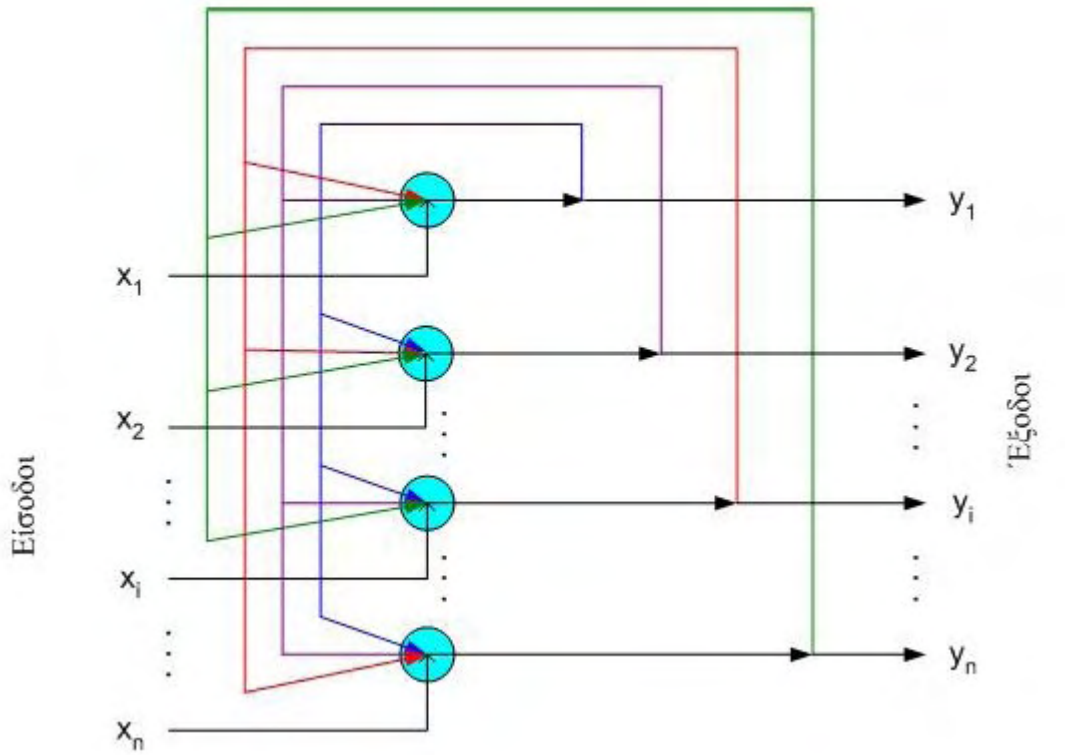
Τα δίκτυα Hopfield ανήκουν στην κατηγορία των αναδρομικών τεχνητών νευρωνικών δικτύων. Ένα αναδρομικό τεχνητό νευρωνικό δίκτυο έχει εκτός από τις προς τα εμπρός, και προς τα πίσω συνδέσεις. Έχει δηλαδή βρόγχους αναδρομής από τις εξόδους του προς τις εισόδους του. Η παρουσία τέτοιων βρόγχων έχει ισχυρή επίπτωση στην ικανότητα μάθησης του δικτύου.

Ένα αναδρομικό τεχνητό νευρωνικό δίκτυο εκπαιδεύεται ως εξής: Αφού εφαρμοστεί μια είσοδος υπολογίζεται η έξοδος του δικτύου η οποία στη συνέχεια ανατροφοδοτείται ως είσοδος στο δίκτυο. Υπολογίζεται η νέα έξοδος του δικτύου και η διαδικασία επαναλαμβάνεται έως ότου η έξοδος του δικτύου τείνει να είναι σταθερή.

Βέβαια δεν γίνεται πάντα η έξοδος του δικτύου σταθερή. Η παραπάνω διαδικασία δεν εξασφαλίζει πάντα ότι σε κάθε επανάληψη οι μεταβολές στην έξοδο του δικτύου θα είναι ολοένα και μικρότερες, έτσι ώστε σε κάποια χρονική στιγμή η έξοδος να πάψει να μεταβάλλεται. Αντιθέτως, είναι πολύ πιθανόν να οδηγήσει σε μια χαοτική συμπεριφορά του δικτύου. Στην περίπτωση αυτή η έξοδος του δικτύου δεν γίνεται ποτέ σταθερή και τότε λέμε ότι το δίκτυο είναι ασταθές.

Η ευστάθεια των αναδρομικών τεχνητών νευρωνικών δικτύων αποτέλεσε αντικείμενο έρευνας πολλών ερευνητών του χώρου στις δεκαετίες του 1960 και 1970. Παρόλα αυτά, η λύση ήρθε το 1982, όταν ο John Hopfield διατύπωσε τη φυσική αρχή της αποθήκευσης πληροφορίας σε ένα δυναμικά ευσταθές δίκτυο [Hopfield, 1982] [17].

Στην Εικόνα 6.3 απεικονίζεται ένα δίκτυο Hopfield ενός επιπέδου το οποίο αποτελείται από n υπολογιστικούς νευρώνες. Η έξοδος κάθε νευρώνα ανατροφοδοτείται ως είσοδος σε όλους τους υπόλοιπους νευρώνες (στα δίκτυα Hopfield δε συναντάμε το φαινόμενο της αυτό–ανατροφοδότησης).



Εικόνα 6.3 Δίκτυο hopfield, ενός επιπέδου με n νευρώνες

Η λειτουργία ενός νευρώνα *k* που χρησιμοποιεί τη συνάρτηση προσήμου περιγράφεται από τις ακόλουθες σχέσεις:

$$u_k = \sum_{j=0}^{m} w_{kj} x_j$$

$$\begin{aligned} y_k = \varphi(u_k) &= +1, \; u_k > 0, \\ &= -1, \; u_k < 0, \\ &= y_k, \; u_k = 0. \end{aligned}$$

Δηλαδή, η έξοδος του νευρώνα γίνεται +1 (ο νευρώνας μεταβαίνει στη κατάσταση +1) εάν το δυναμικό ενεργοποίησης u_k του νευρώνα είναι μεγαλύτερο του μηδενός, γίνεται -1 (μετάβαση σε κατάσταση -1) εάν το δυναμικό u_k είναι μικρότερο του μηδενός και παραμένει αμετάβλητη (ο νευρώνας παραμένει στην προηγούμενη του κατάσταση) εάν το δυναμικό u_k είναι ίσο με το μηδέν. Εάν στο δίκτυο εισάγουμε ένα διάνυσμα το οποίο είναι μερικώς λανθασμένο ή κατεστραμμένο τότε το δίκτυο θα συγκλίνει σε μία σταθερή κατάσταση μετά από έναν αριθμό επαναλήψεων. Οι σταθερές καταστάσεις πολλές φορές ονομάζονται και βασικές μνήμες (fundamental memories). Το δίκτυο Hopfield μπορεί να αποθηκεύσει ένα σύνολο από βασικές μνήμες (διανύσματα) εάν ο πίνακας των βαρών του είναι συμμετρικός και όλα τα στοιχεία της κύριας διαγωνίου του είναι ίσα με το

μηδέν. Τα βάρη υπολογίζονται μόνο μία φορά, κατά τη φάση της αποθήκευσης. Από τη στιγμή που υπολογιστούν δεν μεταβάλλονται και παραμένουν σταθερά. Αφού αποθηκεύσουμε τις βασικές μνήμες, στη συνέχεια, είναι απαραίτητο να επιβεβαιώσουμε ότι το δίκτυο Hopfield είναι ικανό να ανακαλέσει από τη «μνήμη» όλες τις βασικές μνήμες. Στη φάση της ανάκτησης, παρουσιάζουμε ένα άγνωστο n-διάστατο διάνυσμα X στο δίκτυο και ανακαλούμε μια σταθερή κατάσταση.

Ένα πρόβλημα που αντιμετωπίζουμε όταν εφαρμόζουμε τα δίκτυα Hopfield είναι η αποθηκευτική χωρητικότητα (storage capacity), ο μέγιστος αριθμός διανυσμάτων που μπορούν να αποθηκευτούν και να ανακληθούν χωρίς σφάλμα. Ο μέγιστος αριθμός βασικών μνημών M_{max} που μπορούν να αποθηκευτούν σε ένα αναδρομικό δίκτυο με n νευρώνες δίνεται από τη σχέση (δεν μας ενδιαφέρει αν στην ανάκληση έχουμε λάθη):

$M_{max} = 0.15\ n.$

Η μέγιστη αποθηκευτική χωρητικότητα των δικτύων Hopfield μπορεί να ορισθεί και στη βάση του ότι επιθυμούμε οι **περισσότερες** από τις βασικές μνήμες να μπορούν να ανακληθούν χωρίς σφάλμα [Amit, 1989; Negnevitsky, 2002]:

$M_{max} = n / (2\ln(n)).$

Στην περίπτωση, τώρα, που επιθυμούμε ***όλες*** οι βασικές μνήμες, που έχουν αποθηκευτεί σε ένα δίκτυο Hopfield με n νευρώνες, να μπορούν να ανακληθούν τέλεια (χωρίς κανένα σφάλμα) τότε $M_{max} = (n / 4\ln(n))$ [Amit, 1989; Negnevitsky, 2002].

7 Εκπαίδευση ανά ομάδα προτύπων σε σχέση με εκπαίδευση ανά πρότυπο

7.1 Γιατί η εκπαίδευση ανά ομάδα προτύπων μοιάζει καλύτερη

Όταν εκπαιδεύουμε με τη μέθοδο ανά ομάδα προτύπων (batch training), τότε η συσσωρευμένη αλλαγή βάρους δείχνει στην κατεύθυνση της αληθινής κλίσης λάθους. Αυτό σημαίνει ότι ένα αρκετά μικρό βήμα σε εκείνη την κατεύθυνση θα μειώσει το λάθος στο σύνολο εκπαίδευσης συνολικά, εκτός αν ένα τοπικό ελάχιστο έχει ήδη βρεθεί. Αντίθετα, κάθε αλλαγή βάρους που γίνεται κατά τη διάρκεια της εκπαίδευσης ανά πρότυπο θα μειώσει το λάθος για εκείνη την ιδιαίτερη περίπτωση, αλλά μπορεί να μειώσει ή να αυξήσει το λάθος επάνω το σύνολο εκπαίδευσης. Για αυτόν τον λόγο, η εκπαίδευση ανά ομάδα προτύπων θεωρείται μερικές φορές περισσότερο συμβατή με τη θεωρία βελτιστοποίησης σε σχέση με την εκπαίδευση ανά πρότυπο. Εντούτοις, κατά τη διάρκεια της εκπαίδευσης ανά πρότυπο η μέση αλλαγή βάρους τείνει στην κατεύθυνση της αληθινής κλίσης.

Επίσης μερικές φορές έχει υποστηριχτεί η άποψη ότι η εκπαίδευση ανά ομάδα προτύπων είναι «γρηγορότερη» από την εκπαίδευση ανά πρότυπο. Οι συγκρίσεις ταχύτητας μεταξύ της εκπαίδευσης ανά ομάδα προτύπων και της εκπαίδευσης ανά πρότυπο γίνονται μερικές φορές με τη σύγκριση της ακρίβειας (ή του λάθους της εκπαίδευσης) των δικτύων μετά από έναν δεδομένο αριθμό αναπροσαρμογών βαρών. Είναι αλήθεια ότι κάθε βήμα που λαμβάνεται στην εκπαίδευση ανά ομάδα προτύπων είναι πιθανό να φθάσει στο κριτήριο τερματισμού πιο γρήγορα διότι η εκτίμηση της κλίσης είναι ακριβέστερη. Ωστόσο, για να εκτελεστεί το επόμενο βήμα, οι μαθηματικοί υπολογισμοί κοστίζουν σε χρόνο.

Μια πρακτικότερη σύγκριση είναι να συγκριθεί η ακρίβεια των δικτύων μετά από ορισμένες εποχές εκπαίδευσης, δεδομένου ότι μια εποχή της εκπαίδευσης απαιτεί περίπου τον ίδιο χρόνο είτε πρόκειται να εκτελέσει εκπαίδευση ανά πρότυπο είτε ανά ομάδα προτύπων. Εντούτοις, οι εμπειρικές συγκρίσεις των δύο μεθόδων εκπαίδευσης έχουν πραγματοποιηθεί με πολύ μικρά σύνολα εκπαίδευσης. Με τέτοια μικρά σύνολα εκπαίδευσης, τα προβλήματα με την εκπαίδευση ανά ομάδα προτύπων δεν γίνονται προφανή, αλλά με τα μεγαλύτερα σύνολα εκπαίδευσης, γίνεται σύντομα προφανές ότι αυτή η εκπαίδευση είναι μη πρακτική [7].

7.2 Γιατί η εκπαίδευση ανά ομάδα προτύπων είναι πιο αργή

Σε ένα ορισμένο σημείο στο χώρο των βαρών η κλίση μπορεί να υπολογιστεί δαπανηρά με την εκπαίδευση ανά ομάδα προτύπων, οπότε σε αυτή την περίπτωση η κλίση είναι ακριβής, ή ανέξοδα με την εκπαίδευση ανά πρότυπο, οπότε σε αυτή την

περίπτωση η κλίση δεν είναι ακριβής. Ανεξάρτητα από το πόσο ακριβής είναι η κλίση που υπολογίζεται, δεν καθορίζει τη θέση του τοπικού ελάχιστου, ούτε επίσης την κατεύθυνση του. Δείχνει στην κατεύθυνση της πιο απότομης καθόδου από το τρέχον σημείο. Η λήψη ενός μικρού βήματος σε εκείνη την κατεύθυνση πιθανώς θα αποκαλύψει ότι η κατεύθυνση σε εκείνο το νέο σημείο είναι ελαφρώς διαφορετική από την κατεύθυνση στο αρχικό σημείο. Στην πραγματικότητα, η πορεία στο ελάχιστο είναι πιθανό να αλλάξει αρκετά, και όχι απαραίτητα ομαλά. Επομένως, αν και η κλίση μας προσδιορίζει την κατεύθυνση, δεν μας προσδιορίζει πόσο μακριά μπορούμε ακίνδυνα να ακολουθήσουμε αυτή την κατεύθυνση.

Ακόμα κι αν ξέραμε το βέλτιστο μέγεθος ενός βήματος που θα έπρεπε να πάρουμε προς αυτή τη κατεύθυνση, συνήθως δεν θα φτάναμε στο τοπικό ελάχιστο. Η εκπαίδευση ανά πρότυπο, λαμβάνει πολλά μικρά βήματα στη μέση κατεύθυνση της κλίσης. Αφότου έχουν παρουσιαστεί κάποια πρότυπα εκπαίδευσης στο δίκτυο, τα βάρη είναι σε διαφορετική θέση στο χώρο των βαρών, και η κλίση θα είναι πιθανώς ελαφρώς διαφορετική εκεί. Καθώς η εκπαίδευση ανά πρότυπο τρέχει μπορεί να ακολουθήσει τη μεταβαλλόμενη κλίση γύρω από τις αυθαίρετες καμπύλες και μπορεί έτσι να σημειώσει μεγαλύτερη πρόοδο κατά τη διάρκεια μιας εποχής, από οποιοδήποτε ενιαίο βήμα βασισμένο στην αρχική κλίση θα μπορούσε να πραγματοποιήσει η εκπαίδευση ανά ομάδα προτύπων.

Ένας μικρός, σταθερός ρυθμός εκπαίδευσης r (όπως r = 0.01) χρησιμοποιείται για να καθορίσει πόσο μεγάλο είναι ένα βήμα ώστε να υπολογιστεί η κατεύθυνση της κλίσης. Υποθέστε ότι η εκπαίδευση ανά ομάδα προτύπων (batch training) χρησιμοποιείται σε ένα μεγάλο σύνολο εκπαίδευσης 20.000 προτύπων. Οι αλλαγές βάρους που θα ακολουθήσουν θα χρειαστούν τουλάχιστον 20.000 βήματα. Μερικές από τις αλλαγές βάρους είναι σε διαφορετικές κατευθύνσεις και ακυρώνουν έτσι η μια την άλλη. Τα μικρότερα ποσοστά εκπαίδευσης απαιτούν αντίστοιχα περισσότερα περάσματα μέσω των προτύπων για να επιτύχουν τα ίδια αποτελέσματα. Κατά συνέπεια, καθώς το καθορισμένο μέγεθος εκπαίδευσης αυξάνεται, η εκπαίδευση ανά ομάδα προτύπων γίνεται πιο αργή όταν συγκρίνεται με την εκπαίδευση ανά πρότυπο. Εάν το μέγεθος του συνόλου εκπαίδευσης διπλασιάζεται, ο ρυθμός εκπαίδευσης για τη μέθοδο ανά ομάδα προτύπων πρέπει να διαιρεθεί αντίστοιχα στο μισό, ενώ στην εκπαίδευση ανά πρότυπο ο ρυθμός μπορεί να παραμείνει σταθερός. Κατά συνέπεια, όσο μεγαλύτερο το σύνολο εκπαίδευσης, τότε παρατηρείται η εκπαίδευση ανά ομάδα προτύπων να είναι πιο αργή. Χρησιμοποιώντας τη μέθοδο εκπαίδευσης ανά πρότυπο και αρχικοποιώντας τυχαία τα αρχικά βάρη σε ένα μεγάλο σύνολο εκπαίδευσης, τότε ένα μεγάλο μέρος του αρχικού λάθους θα μειωνόταν κατά τη διάρκεια της πρώτης εποχής. Λαμβάνοντας υπόψη αυτά τα ικανοποιητικά στοιχεία, η εκπαίδευση θα συνεχιζόταν μέχρι τα κριτήρια τερματισμού, χωρίς να πρέπει να εκτελέσει μια δεύτερη εποχή. Δηλαδή, σε αυτή τη περίπτωση η εκπαίδευση θα ολοκληρωνόταν πριν η εκπαίδευση ανά ομάδας προτύπων εκτελέσει τη πρώτη αναπροσαρμογή των βαρών.

Για το τρέχον παράδειγμά μας, η εκπαίδευση ανά ομάδα προτύπων εκτέλεσε τις τροποποιήσεις των βαρών βάσει των αρχικά τοποθετημένων βαρών. Έχοντας ορίσει ένα αρκετά μικρό ρυθμό εκπαίδευσης r, τα βάρη μετά από το πέρασμα των 20.000 προτύπων (1 εποχή) είναι ακόμα κοντά στις αρχικές τυχαίες τιμές τους.

Μια άλλη προσέγγιση που δοκιμάστηκε ήταν να υλοποιηθεί η εκπαίδευση ανά πρότυπο για τις πρώτες-πρώτες εποχές, ώστε να αποκτηθούν τα βάρη σε μια λογική σειρά και να ακολουθήσει εκπαίδευση ανά ομάδα προτύπων. Αν και αυτό βοηθά μέχρι ενός ορισμένου βαθμού, τα εμπειρικά και αναλυτικά αποτελέσματα δείχνουν ότι αυτή η

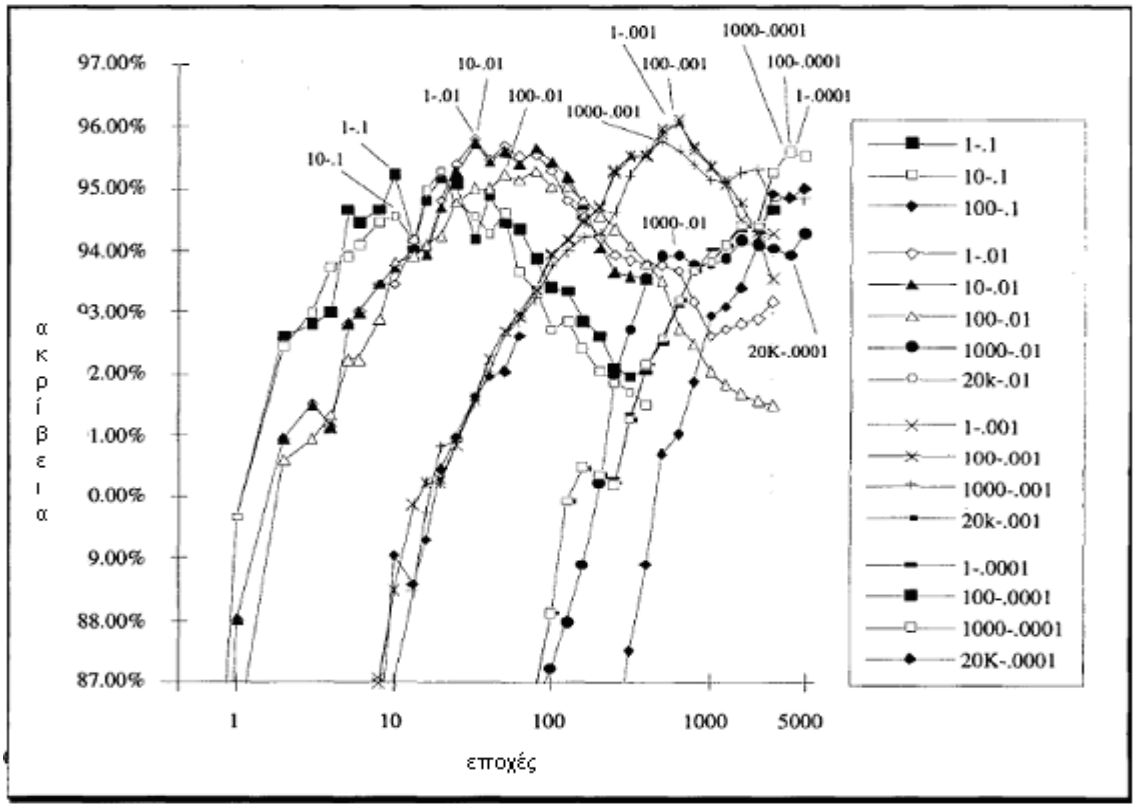
προσέγγιση απαιτεί ακόμα ένα μικρότερο ρυθμό εκπαίδευσης και είναι έτσι πολύ πιο αργή σε σχέση με την εκπαίδευση ανά πρότυπο, χωρίς κάποιο κέρδος στην ακρίβεια. Η τρέχουσα έρευνα και θεωρία καταδεικνύουν ότι για τα σύνθετα προβλήματα, τα μεγαλύτερα σύνολα εκπαίδευσης είναι καλύτερο να αποφευχθούν, ώστε να επιτευχθεί υψηλότερη ακρίβεια.

7.3 Εμπειρικά αποτελέσματα

Τα πειράματα οργανώθηκαν ώστε να κατανοήσουμε πως η συχνότητα αναπροσαρμογής των βαρών και ο ρυθμός εκπαίδευσης έχουν επιπτώσεις στην ταχύτητα με την οποία τα τεχνητά νευρωνικά δίκτυα μπορούν να εκπαιδευτούν. Επίσης άλλα θέματα είναι ο βαθμός της ακρίβειας και της γενίκευσης που μπορούν να επιτύχουν. Ένα σύστημα λεκτικής αναγνώρισης ψηφίων χρησιμοποιήθηκε για αυτά τα πειράματα. Ένα τεχνητό νευρωνικό δίκτυο με 130 εισόδους, 200 κρυμμένους κόμβους 178 εξόδους και 20000 πρότυπα εκπαιδεύθηκε. Η ακολουθία που παρήχθη από το σύστημα αναγνώρισης συγκρίθηκε με την αληθινή ακολουθία των ψηφίων. Ο αριθμός εισαγωγών I, διαγραφών D, αντικαταστάσεων S, και σωστών αντιστοιχιών C, αθροίστηκε σε όλες τις εκφράσεις. Η τελική ακρίβεια της λέξης υπολογίστηκε από το παρακάτω τύπο:
Accurancy = 100% (C - I)/T = 100% (T – (S + I + D))/T, όπου T είναι ο συνολικός αριθμός των λέξεων στόχων σε όλες τις εκφράσεις δοκιμής.

Η εκπαίδευση πραγματοποιήθηκε με συχνότητες αναπροσαρμογών U = 1 (δηλαδή εκπαίδευση ανά πρότυπο), 10, 100, 1000, και 20.000 (δηλαδή εκπαίδευση ανά ομάδα προτύπων). Οι ρυθμοί εκπαίδευσης ήταν r = 0.1, 0.01, 0.001, και 0.0001 και χρησιμοποιήθηκαν με τις κατάλληλες τιμές του U προκειμένου να δειχθεί πόσο μικρή έπρεπε να είναι η τιμή του ρυθμού εκπαίδευσης για την καλύτερη εκπαίδευση του δικτύου. Κάθε εκπαίδευση που πραγματοποιήθηκε χρησιμοποίησε το ίδιο σύνολο τυχαίων αρχικών βαρών, και την ίδια τυχαία σειρά παρουσίασης των προτύπων εκπαίδευσης για κάθε εποχή. Κατά τη χρήση του ρυθμού εκπαίδευσης r = 0.1, με συχνότητες αναπροσαρμογής U = 1 και U = 10 τα αποτελέσματα ήταν κατά προσέγγιση ισοδύναμα, αλλά με συχνότητα αναπροσαρμογής U = 100 το δίκτυο αντεπεξήλθε με δυσκολία. Αυτός ο ρυθμός εκπαίδευσης ήταν πολύ μεγάλος ακόμη και για την εκπαίδευση ανά πρότυπο, ενώ δεν έφθασε και σε αρκετά καλά επίπεδα ακρίβειας. Κατά τη χρήση του ρυθμού εκπαίδευσης r = 0.01, με συχνότητες αναπροσαρμογής των βαρών U = 1 και U = 10 το δίκτυο ανταποκρίθηκε σχεδόν όμοια, επιτυγχάνοντας ακρίβεια 96.31% μετά από 27 εποχές. Με συχνότητα αναπροσαρμογής U = 100 τα αποτελέσματα είχαν 95.76% ακρίβεια μετά από 46 εποχές και για συχνότητα αναπροσαρμογής U = 1000 το δίκτυο ανταπεξήλθε με δυσκολία (δηλαδή χρειάστηκαν πάνω από 1500 εποχές για να επιτύχει τη μέγιστη ακρίβειά που ήταν 95.02%). Η εκπαίδευση ανά ομάδα προτύπων δεν εκπαίδευσε καλύτερα για πάνω από 100 εποχές και δεν πήρε ποτέ ποσοστό ακρίβειας πάνω από 17.2%. Για ρυθμό εκπαίδευσης r = 0.001, με συχνότητες αναπροσαρμογής U = 1 και U = 100 το δίκτυο ανταποκρίθηκε σχεδόν όμοια, με ακρίβεια της τάξης του 96.49% μετά από περίπου 400 εποχές. Με συχνότητα αναπροσαρμογής U = 1000 προκλήθηκε μια μικρή πτώση στην ακρίβεια. Στην εκπαίδευση ανά ομάδα προτύπων (δηλαδή με συχνότητα αναπροσαρμογής U = 20.000) το δίκτυο αντεπεξήλθε με δυσκολία, παρατηρώντας μεγάλους χρόνους και με μέγιστη ακρίβεια να φθάνει στο ποσοστό του 81.55%.

Τέλος, ορίζοντας το ρυθμό εκπαίδευσης r ίσο με 0.0001, με συχνότητες αναπροσαρμογής U = 1, U = 100 και U = 1000 το δίκτυο ανταποκρίθηκε σχεδόν όμοια, επιτυγχάνοντας ακρίβεια της τάξης του 96.31% μετά από περίπου 4000 εποχές. Η εκπαίδευση ανά ομάδα προτύπων με ρυθμό εκπαίδευσης r = 0.0001 απαίτησε πάνω από 100 φορές περισσότερες εποχές εκπαίδευσης σε σχέση με το ρυθμό εκπαίδευσης r = 0.01, χωρίς οποιαδήποτε σημαντική βελτίωση στην ακρίβεια. Η Εικόνα 7.1 παρουσιάζει την ακρίβεια που παρατηρήθηκε για κάθε ένα από τους παραπάνω συνδυασμούς που περιγράφηκαν. Κάθε σειρά προσδιορίζεται από τις συχνότητες αναπροσαρμογής U των βαρών και του ρυθμού εκπαίδευσης r, π.χ. «1000-.01» σημαίνει συχνότητα αναπροσαρμογής U = 1000 και ρυθμός εκπαίδευσης r = 0.01. Ο οριζόντιος άξονας παρουσιάζει τον αριθμό εποχών εκπαίδευσης και χρησιμοποιεί ένα σύνολο σχημάτων για να καταστήσει τα στοιχεία ευκολότερα στο να απεικονίσουν [7].



D. Randall Wilson, Tony R. Martinez

Εικόνα 7.1 Η ακρίβεια που παρατηρήθηκε στο πέρασμα των εποχών

Ο Πίνακας 7.1 συνοψίζει τη περιγραφή που κάναμε παραπάνω. Αυτά τα αποτελέσματα υποστηρίζουν την υπόθεση ότι υπάρχει μια σχεδόν γραμμική σχέση μεταξύ του ποσοστού εκπαίδευσης και του αριθμού εποχών που απαιτούνται για την εκπαίδευση.

Πίνακας 7.1
Ρυθμός εκπαίδευσης – Συχνότητα αναπροσαρμογής – Ακρίβεια -
Εποχές εκπαίδευσης – Παρατήρηση

Learning Rate	Batch Size	Max Word Accuracy	Training Epochs	Rating
0.1	1	95.76%	21	OK
0.1	10	95.94%	41	OK
0.1	100	92.99%	43	Bad
0.01	1	96.31%	27	Good
0.01	10	96.31%	27	Good
0.01	100	95.76%	46	OK
0.01	1000	95.02%	1612	Poor
0.01	20,000	17.16%	381	Terrible
0.001	1	96.49%	402	Good
0.001	100	96.49%	468	Good
0.001	1000	96.13%	405	Good
0.001	20,000	81.55%	1966	OK
0.0001	1	96.31%	4294	Good
0.0001	100	96.31%	4306	Good
0.0001	1000	96.31%	4282	Good
0.0001	20,000	95.39%	3209	OK

D. Randall Wilson, Tony R. Martinez

7.4 Συνοψίζοντας

Η εκπαίδευση ανά ομάδα προτύπων πολυστρωματικών δικτύων έχει αναφερθεί στη βιβλιογραφία ως πιο ορθή μέθοδος σε σχέση με την εκπαίδευση ανά πρότυπο. Επίσης, έχει θεωρηθεί πως πραγματοποιείται σε παρόμοιους χρόνους. Εντούτοις, όταν χρησιμοποιούμε μεγάλα σύνολα εκπαίδευσης, η μέθοδος εκπαίδευσης ανά ομάδα προτύπων πρέπει να χρησιμοποιεί πολύ μικρότερο ρυθμό εκπαίδευσης, με συνέπεια μια γραμμική επιβράδυνση σε σχέση με την εκπαίδευση ανά πρότυπο. Η κλίση που υπολογίζεται στη μέθοδο εκπαίδευσης ανά ομάδα προτύπων είναι ακριβέστερη, ωστόσο η μέση κατεύθυνση της κλίσης με τη μέθοδο εκπαίδευσης ανά πρότυπο είναι στην κατεύθυνση της αληθινής κλίσης. Επιπλέον, η εκπαίδευση ανά πρότυπο έχει το ισχυρό πλεονέκτημα ότι ακολουθεί τις καμπύλες στην επιφάνεια λάθους μέσα σε κάθε εποχή. Στο παράδειγμα των 20000 προτύπων που περιγράψαμε φαίνεται πως η μέθοδος ανά πρότυπο υπερτερεί. Η εκπαίδευση ανά μέγεθος προτύπων εμφανίζεται να είναι μη πρακτική για μεγάλα σύνολα εκπαίδευσης. Η μελλοντική έρευνα θα εξετάσει αυτήν την υπόθεση περαιτέρω για διάφορα μεγέθη ώστε να ενισχυθεί η γενικότητα αυτών των συμπερασμάτων [7].

8 Νευρωνικά δίκτυα και matlab

8.1 ΔΗΜΙΟΥΡΓΙΑ, ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΠΡΟΣΟΜΟΙΩΣΗ ΤΟΥ ΔΙΚΤΥΟΥ

Για τη δημιουργία, εκπαίδευση και προσομοίωση του τεχνητού νευρωνικού δικτύου σε άγνωστα δεδομένα, έχουμε τη δυνατότητα να γράψουμε κώδικα καλώντας έτοιμες συναρτήσεις της Matlab ή εναλλακτικά να χρησιμοποιήσουμε το γραφικό περιβάλλον nntool, σχεδιασμένο με τέτοιο τρόπο ώστε ακόμη και ο μη εξοικειωμένος χρήστης να μπορεί να πειραματιστεί με τα νευρωνικά δίκτυα και να αρχίσει να κατανοεί τον τρόπο λειτουργία τους. Τα βήματα που ακολουθούμε είναι: 1) Αρχικά δημιουργούμε το δίκτυο χρησιμοποιώντας κάποια από τις συναρτήσεις της Matlab. Συγκεκριμένα η συνάρτηση newff() δημιουργεί ένα δίκτυο προσοτροφοδότησης. Παίρνει πέντε ορίσματα ως ορίσματα εισόδου και επιστρέφει το αντικείμενο του δικτύου. Το πρώτο όρισμα είναι ένας Rx2 πίνακας με τις ελάχιστες και μέγιστες τιμές για κάθε ένα από τα R στοιχεία του διανύσματος εισόδου (έστω ότι δίνουμε ως είσοδο ένα πίνακα P, καλούμε τη συνάρτηση minmax(P), για να υπολογίσουμε την ελάχιστη και τη μέγιστη τιμή). Το δεύτερο όρισμα είναι ένας πίνακας που περιέχει το μέγεθος του κάθε στρώματος (δηλαδή τον αριθμό των νευρώνων σε κάθε στρώμα). Για παράδειγμα γράφουμε [5 3 1], που σημαίνει το πρώτο κρυμμένο στρώμα έχει πέντε νευρώνες, το δεύτερο τρεις και το τρίτο είναι ο νευρώνας εξόδου. Το τρίτο όρισμα περιέχει τα ονόματα των συναρτήσεων ενεργοποίησης για καθένα από τα χρησιμοποιούμενα στρώματα. Το τέταρτο όρισμα είναι το όνομα του αλγόριθμου εκπαίδευσης που θέλουμε να χρησιμοποιήσουμε και το πέμπτο όρισμα είναι η συνάρτηση απόδοσης. Για παράδειγμα η εντολή:

net = newff(minmax(P),[5 3 1],{'tansig','tamsig', 'purelin'},'traingd', 'mse')

δημιουργεί το αντικείμενο ενός δικτύου προσοτροφοδότησης X-5-3-1, όπου X ο αριθμός του πεδίου των εισόδων. Το μέγεθος του διανύσματος εισόδου «διαβάζεται» από τον πίνακα των εισόδων P (αφού πρώτα πάρουμε τον ανάστροφο πίνακα του P δηλώνοντας P=P'), το πρώτο κρυμμένο στρώμα έχει 5 νευρώνες με συνάρτηση ενεργοποίησης την tansig, το δεύτερο στρώμα έχει 3 νευρώνες με συνάρτηση ενεργοποίησης την tansig και το τρίτο στρώμα (στρώμα εξόδου) έχει έναν νευρώνα και ως συνάρτηση ενεργοποίησης την purelin. Ο αλγόριθμος εκπαίδευσης που ορίζουμε για αυτό το δίκτυο είναι ο traingd και η συνάρτηση απόδοσης, είναι η συνάρτηση του μέσου ολικού τετραγωνικού σφάλματος (mse). Η newff εκτός από το ότι δημιουργεί το δίκτυο αρχικοποιεί και τις παραμέτρους του (βάρη και πολώσεις) δίνοντας τους μικρές τυχαίες τιμές. Ωστόσο μπορούμε να χρησιμοποιήσουμε τη συνάρτηση init(net) η οποία δίνει στο δίκτυο τυχαίες τιμές στα βάρη και τις πολώσεις (newcf(): με την κλήση της συνάρτησης αυτής δημιουργείται ένα Τεχνητό Νευρωνικό Δίκτυο Προώθησης με ρυθμό αλλαγής των συναπτικών βαρών και των πολώσεων μεγαλύτερο από εκείνο που παρέχει η newff() (cascade-forward neural network), και το οποίο εκπαιδεύεται βάσει του αλγορίθμου της όπισθεν διάδοσης του σφάλματος (backpropagation). 2) Στη συνέχεια πραγματοποιείται η εκπαίδευση του δικτύου χρησιμοποιώντας μια από τις συναρτήσεις της Matlab. Η συνάρτηση train() εκτελεί εκπαίδευση ανά ομάδα προτύπων και καλείται ως εξής:

[net,TR,Y,E,] = train(net,P,T)

Ως ορίσματα εισόδου παίρνει το αντικείμενο του δικτύου που μόλις δημιουργήσαμε, τον πίνακα με τις εισόδους των παραδειγμάτων εκπαίδευσης, (έστω P) και τον πίνακα με τους αντίστοιχους στόχους, (έστω T). Τα ορίσματα που επιστρέφει είναι το εκπαιδευμένο δίκτυο, η μεταβλητή TR με πληροφορίες για την απόδοση του δικτύου ανά εποχή και τους πίνακες Y και E με τις εξόδους που δίνει το εκπαιδευμένο δίκτυο στα παραδείγματα εκπαίδευσης και τα αντίστοιχα λάθη. Οι πίνακες P και T πρέπει να είναι μεγέθους (Αριθμός Εισόδων) * (Αριθμός Παραδειγμάτων Εκπαίδευσης) και (Αριθμός Εξόδων) * (Αριθμός Παραδειγμάτων Εκπαίδευσης) αντίστοιχα. Δηλαδή, τα στοιχεία των παραδειγμάτων εκπαίδευσης πρέπει να είναι τοποθετημένα στους πίνακες P, T κατά στήλες. 3) Το επόμενο βήμα είναι να γίνει η προσομοίωση του δικτύου. Σκοπός είναι το δίκτυο να ανταποκριθεί σε νέα δεδομένα. Η συνάρτηση της Matlab που καλούμε είναι η sim(). Ως ορίσματα δέχεται το αντικείμενο του δικτύου και το διάνυσμα που θέλουμε να τρέξει στο δίκτυο. Έτσι η συνάρτηση sim() μας επιστρέφει τελικά τις εξόδους για το νέο διάνυσμα [15], [18].

8.2 Αλγόριθμοι Εκπαίδευσης

Κάποιοι αλγόριθμοι λοιπόν που προσφέρει σαν έτοιμες συναρτήσεις για την εκπαίδευση πολυστρωματικών δικτύων perceptron η Matlab είναι οι ακόλουθοι:

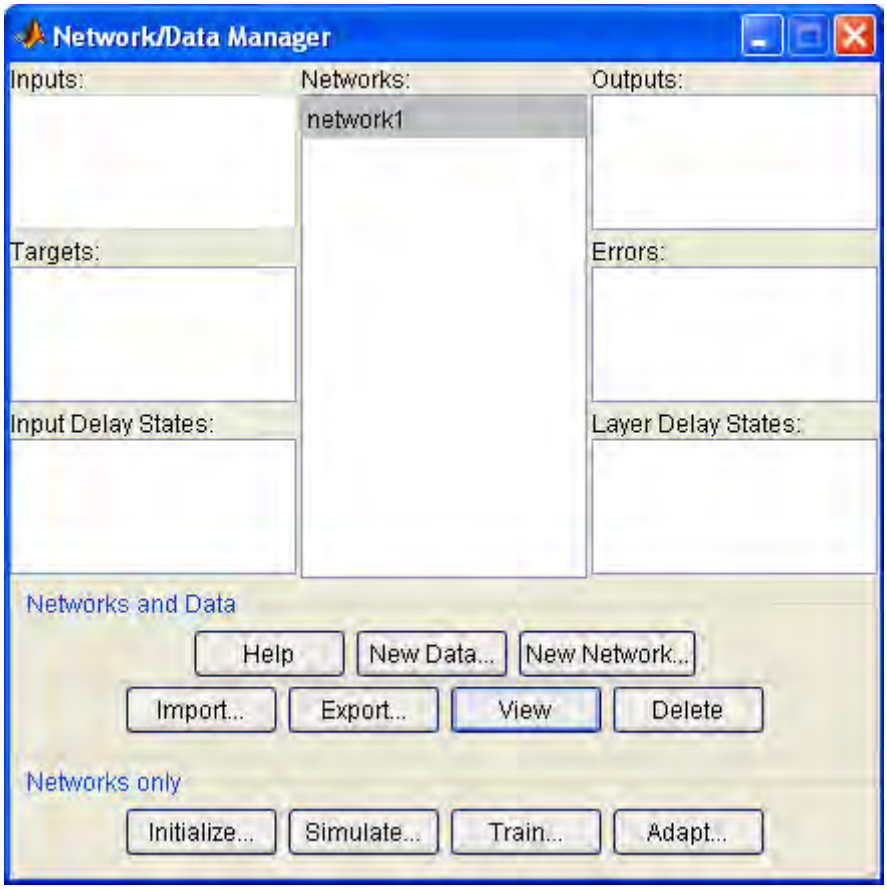
traingd, traingdm: Οι συναρτήσεις αυτές υλοποιούν τον βασικό αλγόριθμο ανάστροφης διάδοσης που περιγράψαμε. Η διαφορά της traingdm από την traingd είναι ότι συνυπολογίζει και τον όρο ορμής (momentum term). Ο αλγόριθμος traingd είναι γενικά πολύ αργός καθώς απαιτεί μικρές τιμές του ρυθμού εκπαίδευσης ώστε η διαδικασία να μην αποκλίνει. Ο αλγόριθμος traingdm είναι συνήθως γρηγορότερος αλλά και αυτός θεωρείται πολύ αργός για πρακτικά προβλήματα.

traingda, traingdx: Το ιδιαίτερο χαρακτηριστικό αυτών των αλγορίθμων είναι ο μεταβλητός ρυθμός εκπαίδευσης. Στον βασικό αλγόριθμο ανάστροφης διάδοσης (traingd ή traingdm) η τιμή του ρυθμού εκπαίδευσης διατηρείται σταθερή σε όλη τη διάρκεια της εκπαίδευσης με συνέπεια η απόδοση του αλγορίθμου να εξαρτάται σε μεγάλο βαθμό από την τιμή που θα επιλέξει ο χρήστης. Μεγάλη τιμή του ρυθμού εκπαίδευσης μπορεί να κάνει το σύστημα ασταθές ενώ από την άλλη μικρή τιμή οδηγεί σε πολλή αργή σύγκλιση. Αυτό ακριβώς το πρόβλημα προσπαθούν να λύσουν οι αλγόριθμοι traingda και traingdx. Μεταβάλλουν κατά τη διάρκεια της εκπαίδευσης το ρυθμό εκπαίδευσης προσπαθώντας να τον κρατήσουν στη μεγαλύτερη δυνατή τιμή που θα εξασφαλίζει ένα ευσταθές σύστημα. Η διαφορά της traingdx από την traingda είναι ότι συνυπολογίζει και τον όρο ορμής [3], [7], [15].

8.3 Neural network tool

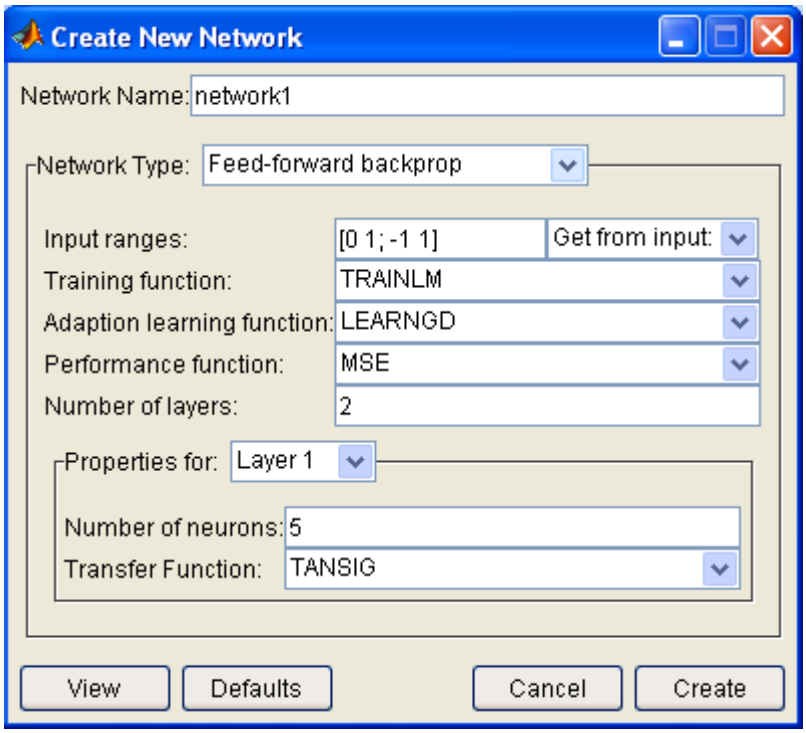
Ένας άλλος τρόπος να δημιουργήσουμε ένα νευρωνικό δίκτυο, να το εκπαιδεύσουμε και να του δώσουμε νέα δεδομένα προς υπολογισμό είναι να κάνουμε χρήση του γραφικού περιβάλλοντος της Matlab (neural network tool). Γράφοντας την

εντολή nntool στο περιβάλλον της Matlab προκύπτει ένα παράθυρο, όπως αυτό της Εικόνας 8.1 όπου μπορούμε να εκτελέσουμε διάφορες επιλογές, για παράδειγμα δημιουργία δικτύου, εισαγωγή δεδομένων και όποια άλλη εντολή είναι σχετική με το αντικείμενό μας.



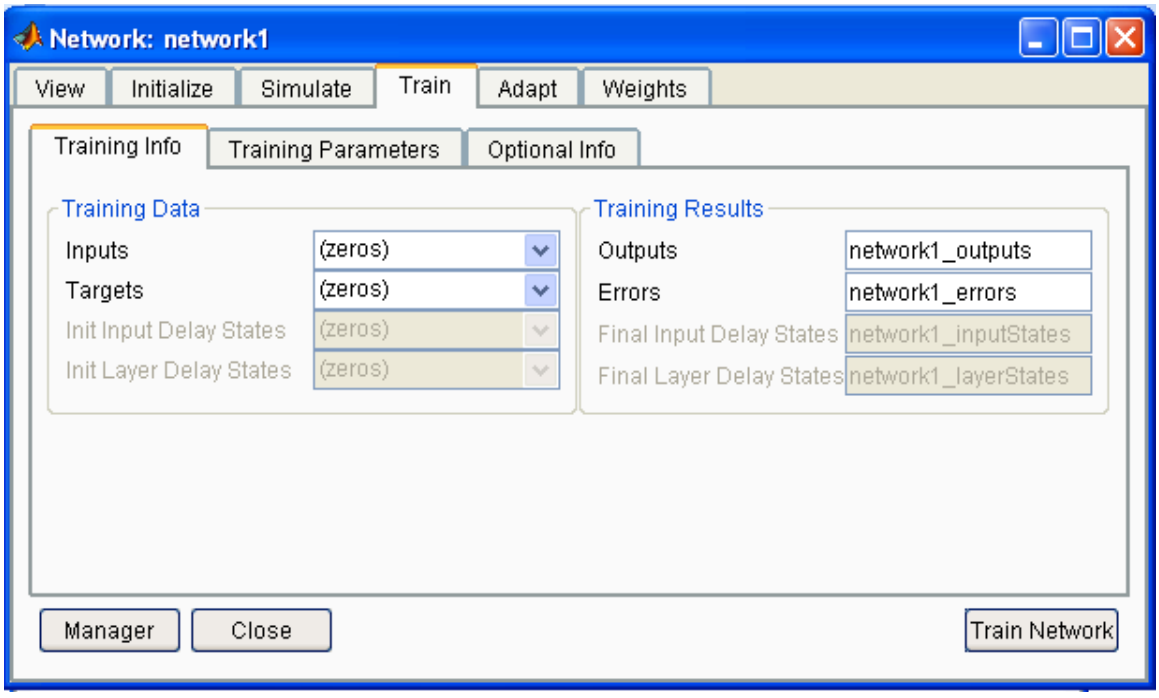
Εικόνα 8.1 Παράθυρο διαχείρισης τεχνητού νευρωνικού δικτύου και δεδομένων

Στη συνέχεια επιλέγοντας την εντολή newnetwork εμφανίζεται ένα παράθυρο όπως αυτό της Εικόνας 8.2. Σε εκείνο το παράθυρο εμφανίζονται διάφορα πεδία που σχετίζονται με το τύπο του δικτύου, τις συναρτήσεις ενεργοποίησης, τη συνάρτηση απόδοσης σφάλματος και τη διαχείριση των επιπέδων του δικτύου.



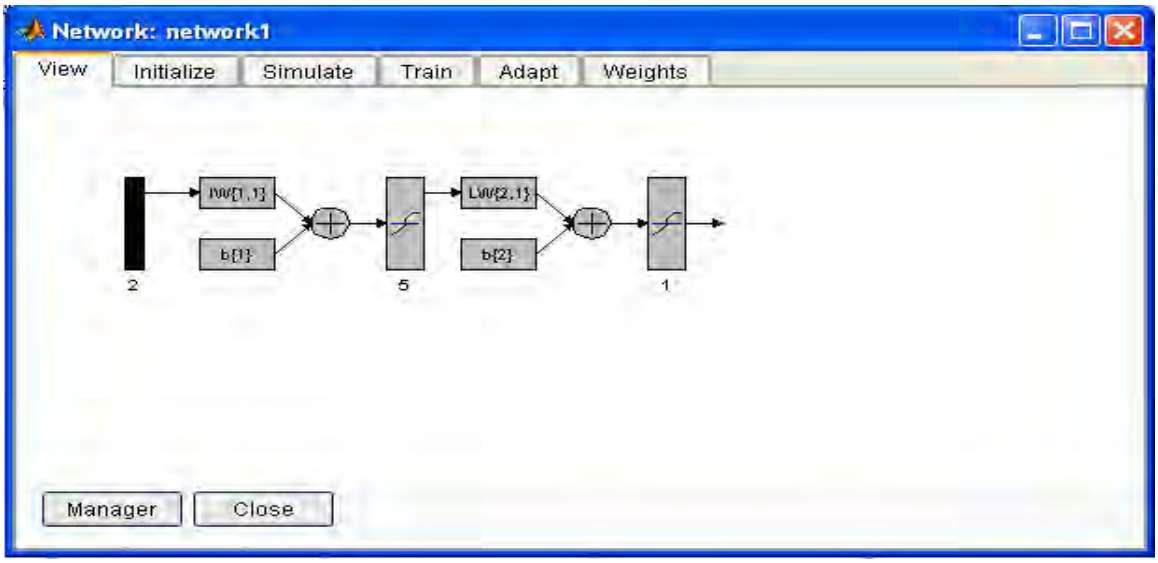
Εικόνα 8.2 Δημιουργία δικτύου

Αφού δημιουργήσουμε το τεχνητό νευρωνικό δίκτυο, στη συνέχεια μπορούμε να το εκπαιδεύσουμε πατώντας στην εντολή train. Στο νέο παράθυρο (Εικόνα 8.3) που εμφανίζεται δίνουμε ως εισόδους το δίανυσμα εκπαίδευσης και το διάνυσμα των στόχων.



Εικόνα 8.3 Εκπαίδευση τεχνητού νευρωνικού δικτύου

Τέλος επιλέγοντας την εντολή view, εμφανίζεται ένα νέο παράθυρο (Εικόνα 8.4) όπου δίνεται η σχηματική απεικόνιση του τεχνητού νευρωνικού δικτύου.



Εικόνα 8.4 Προβολή του τεχνητού νευρωνικού δικτύου

Το neural network tollbox είναι ένα αρκετά χρήσιμο εργαλείο που προσφέρει η Matlab, ώστε κάποιος να ασχοληθεί με την εκπαίδευση των τεχνητών νευρωνικών δικτύων και να έχει τη δυνατότητα να εξάγει κάποια συμπεράσματα. Το βασικό πλεονέκτημα είναι το γεγονός ότι δε χρειάζεται κάποιος να έχει άριστες γνώσεις στο χειρισμό της Matlab, ώστε να το χρησιμοποιήσει.

9 Υλοποίηση και αποτελέσματα κώδικα

Παρακάτω παρατίθεται ο κώδικας που υλοποιήθηκε με σκοπό τη δημιουργία αντικειμένου τεχνητού νευρωνικού δικτύου, εκπαίδευση, και προσομοίωση σε νέα δεδομένα. Επίσης ακολουθεί επεξήγηση του κώδικα και εξήγηση των αποτελεσμάτων.

Ο κώδικας

κώδικας εκπαίδευσης του νευρωνικού δικτύου

```
function [ net, mse_, correct_train, correct_test] =
neural1(train_function, goal_, epochs_, learning_rate, time_, Aset, t ,
ptest, real_t, ad, Cell_)
 T = t';
 s = Aset'; pp= ptest';
net=newff(minmax(s), ad, Cell_, train_function, 'learngdm', 'mse' );
 net=init(net);
 net.trainParam.epochs = epochs_;
 net.trainParam.min_grad=0.000000001;
 net.trainParam.goal =goal_;
 net.trainParam.lr =learning_rate;
 net.trainParam.show =1;
 net.trainParam.time = time_;
 net1 = train(net, s, T);
 Y=sim(net1, pp);
 mse_ = mse(real_t' - Y)

 figure;plot( real_t, 'go');
 grid on;
 hold on;
 plot(sign(Y' - 0.5)==1, 'r+')
 YY=sim(net1, s);
 correct_train = 100*(length(T) - sum(abs((sign(YY - 0.5)==1) - T))
)/length(T)
 correct_test  = 100*(length(real_t) - sum(abs((sign(Y' - 0.5)==1) -
real_t)))/length(real_t)
```

κώδικας δημιουργίας γραφικού περιβάλλοντος

```
function varargout = grafiko(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @grafiko_OpeningFcn, ...

                   'gui_OutputFcn',  @grafiko_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end


.
function grafiko_OpeningFcn(hObject, eventdata, handles, varargin)
handles.output = hObject;

guidata(hObject, handles);



function varargout = grafiko_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;



function edit6_Callback(hObject, eventdata, handles)


function edit6_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit7_Callback(hObject, eventdata, handles)



function edit7_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
```

```
end



function edit8_Callback(hObject, eventdata, handles)

function edit8_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit9_Callback(hObject, eventdata, handles)



function edit9_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function popupmenu2_Callback(hObject, eventdata, handles)

function popupmenu2_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit10_Callback(hObject, eventdata, handles)


function edit10_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit11_Callback(hObject, eventdata, handles)

function edit11_CreateFcn(hObject, eventdata, handles)
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))


set(hObject,'BackgroundColor','white');
end


function edit12_Callback(hObject, eventdata, handles)


function edit12_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit13_Callback(hObject, eventdata, handles)



function edit13_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit15_Callback(hObject, eventdata, handles)


function edit15_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function pushbutton1_Callback(hObject, eventdata, handles)
w=get(handles.edit6, 'string'); %σε αυτές τις γραμμές του κώδικα
συνδέουμε τα διάφορα αντικείμενα που εμφανίζονται στο γραφικό
περιβάλλον που δημιουργήσαμε
ww=char(w);
training_set= xlsread(ww);
Aset=training_set;

e=get(handles.edit7, 'string');
ee= char(e);
target_set= xlsread(ee);
```

```
t=target_set;

r= get(handles.edit8, 'string');
rr= char(r);
test_set= xlsread(rr);
ptest=test_set;

o= get(handles.edit9, 'string');
oo= char(o);
test_set= xlsread(oo);
real_t=test_set;

a= get(handles.popupmenu2, 'Value');
string_list= get(handles.popupmenu2, 'String')
train_function= string_list{a}


d= get(handles.edit10, 'string');
dd= char(d);
ad=str2num(dd);

g= get(handles.edit11, 'string');
epochs__= char(g);
epochs_=str2num(epochs__);

m= get(handles.edit12, 'string');
goal__= char(m);
goal_=str2num(goal__);

b= get(handles.edit13, 'string');
learning__rate= char(b);
learning_rate=str2num(learning__rate);

k= get(handles.edit15, 'string');
time__= char(k);
time_=str2num(time__);

hk= length(ad)
for i=1:hk-1
    Cell_(1,i)={'tansig'}
end
Cell_(1,hk)={'logsig'}
Cell_    %με αυτό το τρόπο όσα επίπεδα και αν βάλω μπορώ να υπολογίσω
σωστά το αντικείμενο net



[ net, mse_, correct_train, correct_test] = neural1(train_function,
goal_, epochs_, learning_rate, time_, Aset, t , ptest, real_t, ad,
Cell_)

function pushbutton1_CreateFcn(hObject, eventdata, handles)



function edit16_Callback(hObject, eventdata, handles)
```

```
 function edit16_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

9.2 Επεξήγηση κώδικα

Ορίσαμε, αρχικώς, το όνομα της συνάρτησης και τις μεταβλητές Aset και t, οι οποίες είναι το σύνολο των προτύπων που δίνονται για την εκπαίδευση του δικτύου και το σύνολο με τα επιθυμητά αποτελέσματα αντίστοιχα. Επιπλέον, για την προσομοίωση του δικτύου κάναμε χρήση των μεταβλητών p_test και real_t. Κατόπιν, κι εφόσον είχαμε δημιουργήσει το αντικείμενο του δικτύου, προχωρήσαμε στην αρχικοποίηση των τιμών των βαρών με μικρές τυχαίες τιμές. Στη συνεχεία, ορίσαμε κάποιες από τις παραμέτρους του δικτύου και οι οποίες είναι ο αριθμός των εποχών, ο μέγιστος χρόνος εκτέλεσης του προγράμματος, ο ρυθμός εκπαίδευσης καθώς και ο στόχος που θέλουμε να επιτύχουμε. Εν κατακλείδι, υπολογίσαμε τόσο την επιτυχία που είχε το πρόγραμμα να ταξινομήσει τα δεδομένα εκπαίδευσης όσο και τα δεδομένα προσομοίωσης έχοντας ως στόχο να διαπιστώσουμε κατά πόσο εκπαιδεύσαμε το τεχνητό νευρωνικό δίκτυο όσον το δυνατόν καλύτερα.

Ολοκληρώνοντας τη δημιουργία του βασικού προγράμματος, προτιμήθηκε, για πρακτικούς λόγους, να σχεδιάσουμε, χρησιμοποιώντας τον οδηγό δημιουργίας της Matlab, ένα γραφικό περιβάλλον, το οποίο και να διαχειρίζεται το πρόγραμμα. Το δυσκολότερο μέρος στον ανωτέρω σχεδιασμό ήταν να καταφέρουμε να αντιστοιχίσουμε τα αντικείμενα που εμφανίζονται στο παράθυρο του χρήστη με τις κατάλληλες εντολές. Ωστόσο αυτή η αντιστοιχία πραγματοποιήθηκε και πλέον το πρόγραμμα είναι κατάλληλο προς χρήση διαχειρίζοντάς το μέσω του γραφικού περιβάλλοντος. Το αρχείο Mammographic Mass Data Set περιέχει τα δεδομένα που χρησιμοποιήσαμε στο πρόγραμμα. Το κατεβάσαμε από τη σελίδα http://archive.ics.uci.edu/ml/ [14]. Τα χαρακτηριστρικά των δεδομένων ήταν:

1. BI-RADS assessment: 1 to 5 (ordinal, non-predictive!) δι-ακτινική εκτίμηση: από 1 έως 5 (αριθμητικό)
2. Ηλικία: σε χρόνια (ακέραιος)
3. Σχήμα: σχήμα μάζας: κυκλική = 1, οβάλ = 2, λοβός = 3, ακανόνιστη = 4
4. Όρια:όρια μάζας: περιορισμένη = 1, μικρολοβόδες = 2, ασαφής = 3, ασθενώς(ή ελαφρώς) καθορισμένη = 4, ακανθώδης(επεκτατική) = 5
5. Πυκνότητα: πυκνότητα μάζας: υψηλή = 1, ισοβαρής = 2, χαμηλή = 3 (περι)έχει πάχος =4
6. Σοβαρότητα: καλοήθης = 0, κακοήθης = 1 (δίτιμο, **πεδίο στόχος**!)

9.3 Πειραματικά αποτελέσματα

Πείραμα 1

Το πείραμα, που περιγράφεται κατωτέρω, είχε ως σκοπό να καταγράψει το ελάχιστο μέσο τετραγωνικό σφάλμα ως απόρροια των αλλαγών των τιμών των παραμέτρων. Επισημάνεται, επίσης, ότι οι αλλαγές των παραμέτρων κάθε φορά γίνονταν μέσα στο κώδικα, ενώ δημιουργήθηκε κι ο συγκεντρωτικός Πίνακας 9.1.

Εν αρχή, έγινε χρήση ενός πίνακα μεγέθους 726x6, όπου η τελευταία στήλη αντιπροσώπευε τις παρατηρήσεις τύπου 0 ή 1. Επίσης, ορίστηκαν ως κριτήρια τερματισμού η τιμή της συνάρτησης απόδοσης του ολικού μέσου τετραγωνικού σφάλματος, ο αριθμός των εποχών ή ο μέγιστος χρόνος κατά τον οποίο έτρεχε το πρόγραμμα. Εκτελέστηκαν διάφορες παραλλαγές του κώδικα και το αποτέλεσμα ήταν η δημιουργία ενός συγκεντρωτικού πίνακα με τις καταγεγραμμένες παρατηρήσεις οι οποίες καταλήγουν στο ότι, η τιμή του ελάχιστου τετραγωνικού σφάλματος που καταγράφηκε, έχοντας κάνει αρκετές δοκιμές, ήταν 0.0364921 αν και τελικός στόχος ήταν η τιμή 0.01. Εδώ πρέπει να σημειωθεί ότι στο πλαίσιο του πειράματος έγιναν διάφοροι συνδυασμοί των συναρτήσεων newff(), newcf(), traingdx(), trainrp(), traincfg().

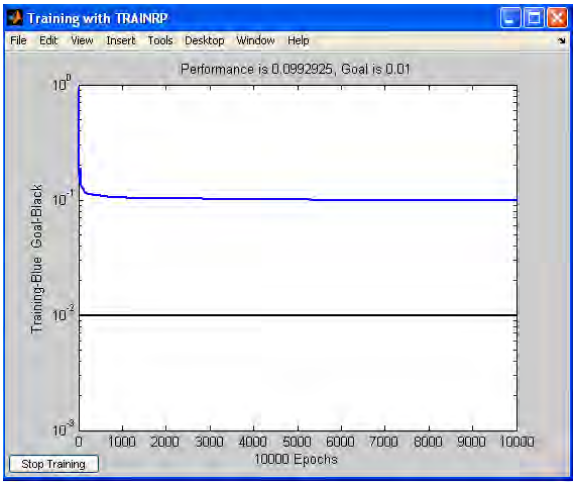
Αλλαγές, επίσης, πραγματοποιήθηκαν στη τιμή του ρυθμού εκπαίδευσης, καθώς και στο πλήθος των νευρώνων και των επιπέδων ενώ χρησιμοποιήθηκαν και οι συναρτήσεις μεταφοράς tansig() και purelin() (σε μια περίπτωση, σύμφωνα με το πίνακα). Τα χαρακτηριστικά του κώδικά για τη τιμή 0.0364921 της συνάρτησης απόδοσης ήταν: Χρησιμοποιήθηκε η συνάρτηση δημιουργίας τεχνητού νευρωνικού δικτύου newff(), με δομή 100-100-1, δηλαδή υπήρχαν δυο κρυφά επίπεδα με εκατό νευρώνες έκαστο. Η συνάρτηση μεταφοράς ήταν η tansig() και η συνάρτηση εκπαίδευσης η trainrp(), η οποία χρησιμοποιεί τη μέθοδο της αριθμητικής βελτιστοποίησης για την επίτευξη του ελάχιστου ολικού μέσου τετραγωνικού σφάλματος. Η τιμή του ρυθμού εκπαίδευσης ήταν πολύ μικρή, εξαιτίας και του πλήθους των δεδομένων, και ίση με 0.0001. Η τιμή που επετεύχθει δεν ήταν μικρότερη από 0.01, παρόλα αυτά είναι αρκετά μικρή. Αντίθετα, η μέγιστη τιμή που επετεύχθει ήταν: 0.0992925. Τα χαρακτηριστικά του κώδικά για τη τιμή αυτή της συνάρτησης απόδοσης ήταν τα κάτωθι: χρησιμοποιήθηκε η συνάρτηση δημιουργίας νευρωνικού δικτύου newff(), με δομή 5-1, δηλαδή υπήρχε ένα μόνο κρυφό επίπεδο με πέντε νευρώνες. Η συνάρτηση μεταφοράς ήταν η tansig() και η συνάρτηση εκπαίδευσης η trainrp().

Το σύνολο των χαρακτηριστικών των δεδομένων ήταν έξι. Το 80% του συνόλου των δεδομένων χρησιμοποιήθηκαν για την εκπαίδευση του δικτύου και το υπόλοιπο 20% χρησιμοποιήθηκε για τη προσομοίωση του δικτύου. Ακόμη και σε επιλογή χαρακτηριστικών που πραγματοποιήθηκε, και συγκεκριμένα αφαιρέθηκε η στήλη ηλικία από τα δεδομένα, το αποτέλεσμα της συνάρτησης απόδοσης δεν είχε μικρότερη τιμή από τη τιμή 0.0364921. Συγκεκριμένα η τιμή ήταν ίση με 0.0690473. Τα δεδομένα που χρησιμοποιήθηκαν αποτελούνταν από διακριτές τιμές.

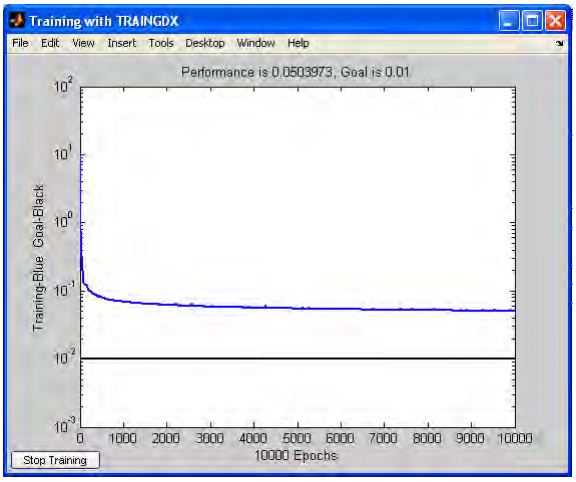
Πίνακας 9.1
Συγκεντρωτικός πίνακας μετρήσεων

Δομή δικτύου	Συνάρτηση εξόδου	Συνάρτηση εκπαίδευσης	Τεχνική εκπαίδευσης	Ρυθμός εκπαίδευσης	Τιμή συνά-ρτησης απόδοσης	Εικόνα
5-1	tansig	trainrp	Αριθμητική βελτιστοποίηση	0.001	0.0992925	9.14
150-100- 1	purelin	traingdx	ευρεστική	0.001	0.0503973	9.15
200-100-1	tansig	traingdx	ευρεστική	0.001	0.0484261	9.16
500-250-1	tansig	traingdx	ευρεστική	0.0001	0.0480565	9.17
500-250-1	tansig	trainrp	Αριθμητική βελτιστοποίηση	0.0001	0.0367559	9.18
500-250-1	tansig	traincgf	Αριθμητική βελτιστοποίηση	0.0001	0.0390309	9.19
1000-300-1	tansig	trainrp	Αριθμητική βελτιστοποίηση	0.0001	0.0373432	9.20
100-100-1	tansig	trainrp	Αριθμητική βελτιστοποίηση	0.0001	0.0365503	9.21
250-250-1	tansig	trainrp	Αριθμητική βελτιστοποίηση	0.0001	0.0365367	9.22
100-100-100-1	tansig	trainrp	Αριθμητική βελτιστοποίηση	0.0001	0.0364921	9.23
50-50-40-1	pureline	trainrp	Αριθμητική βελτιστοποίηση	0.0001	0.036539	9.24
153-140-1	tansig	trainrp	Αριθμητική βελτιστοποίηση	0.0001	0.036525	9.25
153-140-1	tansig	trainrp	Αριθμητική βελτιστοποίηση	0.0001	0.0365299	9.26
100-100-1(επιλογη χαρακτηριστικων)	tansig	trainrp	Αριθμητική βελτιστοποίηση	0.005	0.0690473	9.27

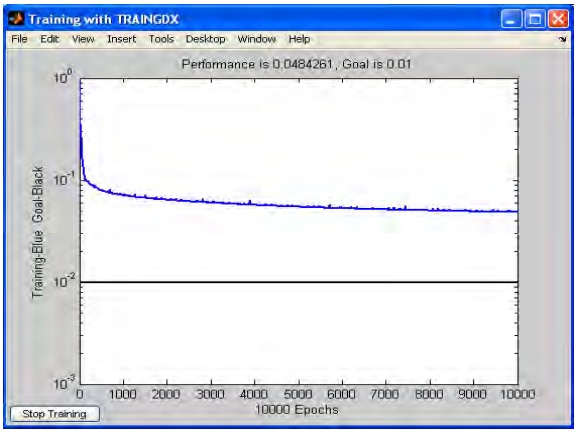
Παρακάτω ακολουθούν τα σχετικά διαγράμματα που αφορούν στον υπολογισμό του ολικού μέσου τετραγωνικού σφάλματος.



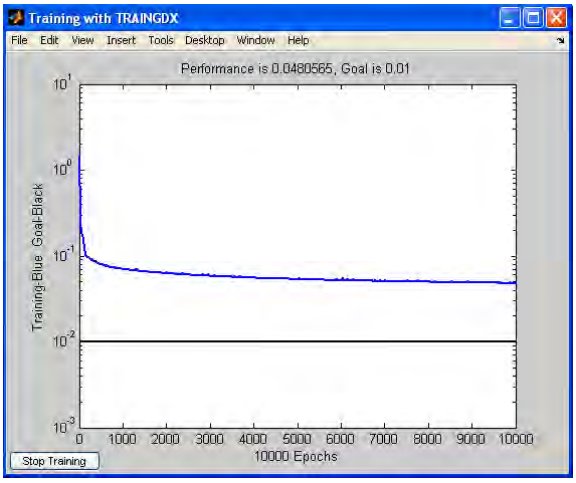
Εικόνα 9.14 Δομή δικτύου 5-1



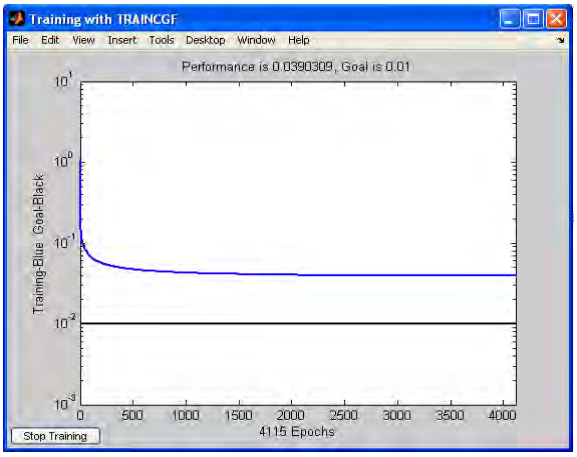
Εικόνα 9.15 Δομή δικτύου150-100- 1



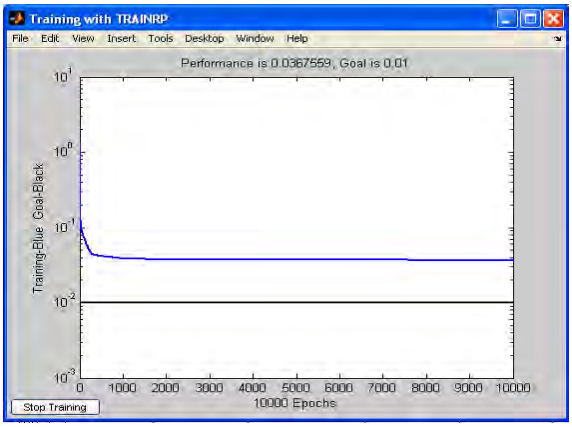
Εικόνα 9.16 Δομή δικτύου 200-100-1



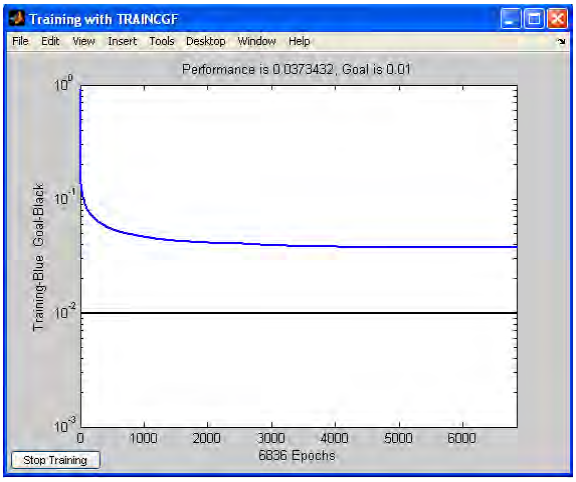
Εικόνα 9.17 Δομή δικτύου 500-250-1



Εικόνα 9.18 Δομή δικτύου 500-250-1

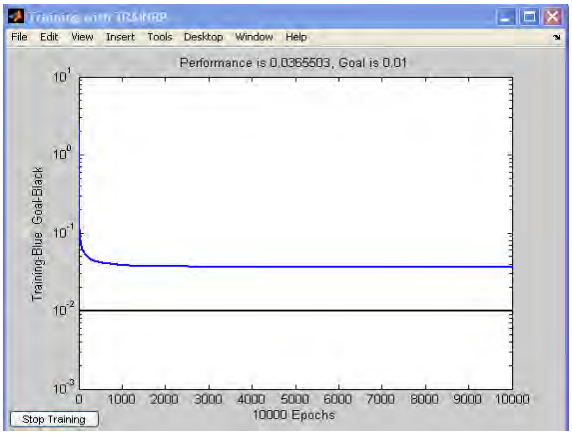


Εικόνα 9.19 Δομή δικτύου 500-250-1

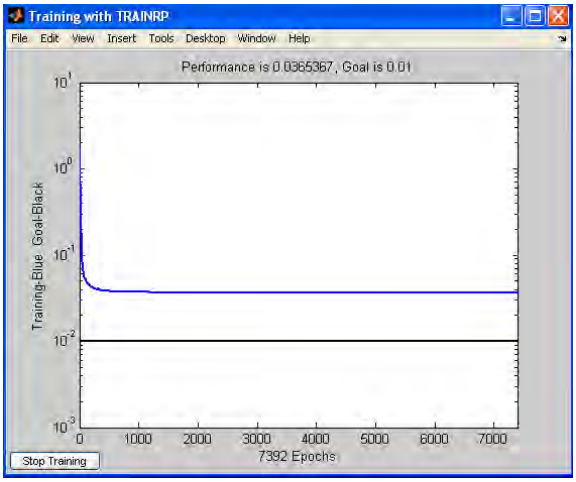


Εικόνα 9.20 Δομή δικτύου 1000-300-1

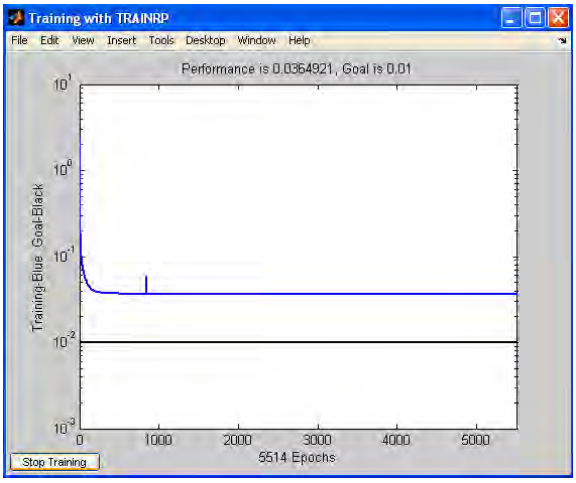
.



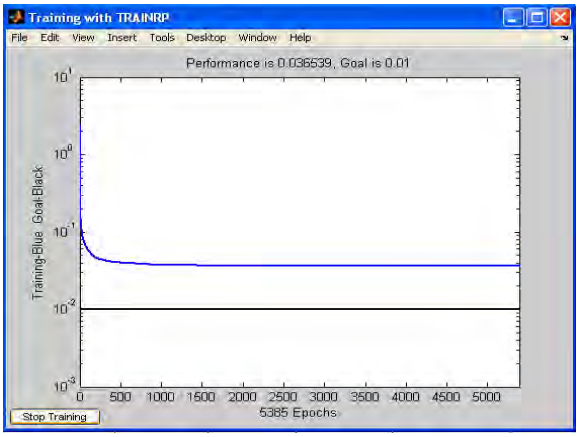
Εικόνα 9.21 Δομή δικτύου 100-100-1



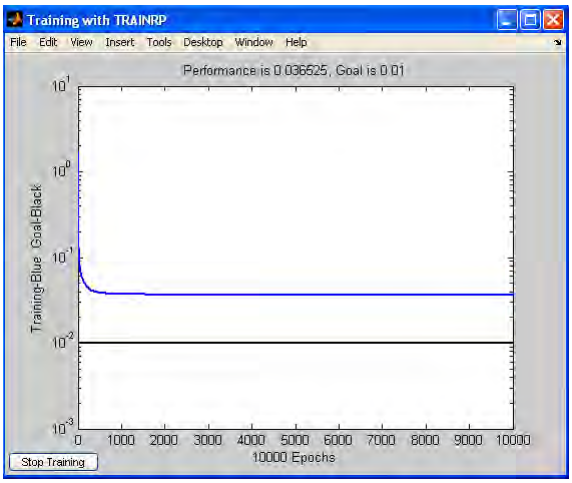
Εικόνα 9.22 Δομή δικτύου 250-250-1



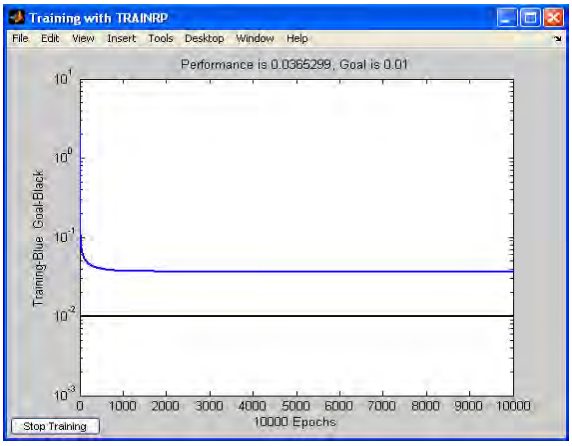
Εικόνα 9.23 Δομή δικτύου 100-100-100-1



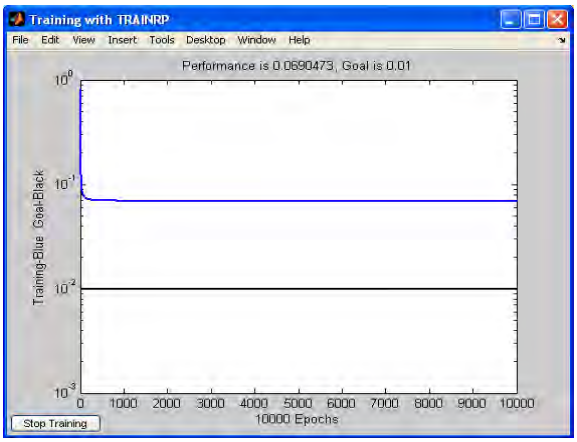
Εικόνα 9.24 Δομή δικτύου 50-50-40-1



Εικόνα 9.25 Δομή δικτύου 153-140-1



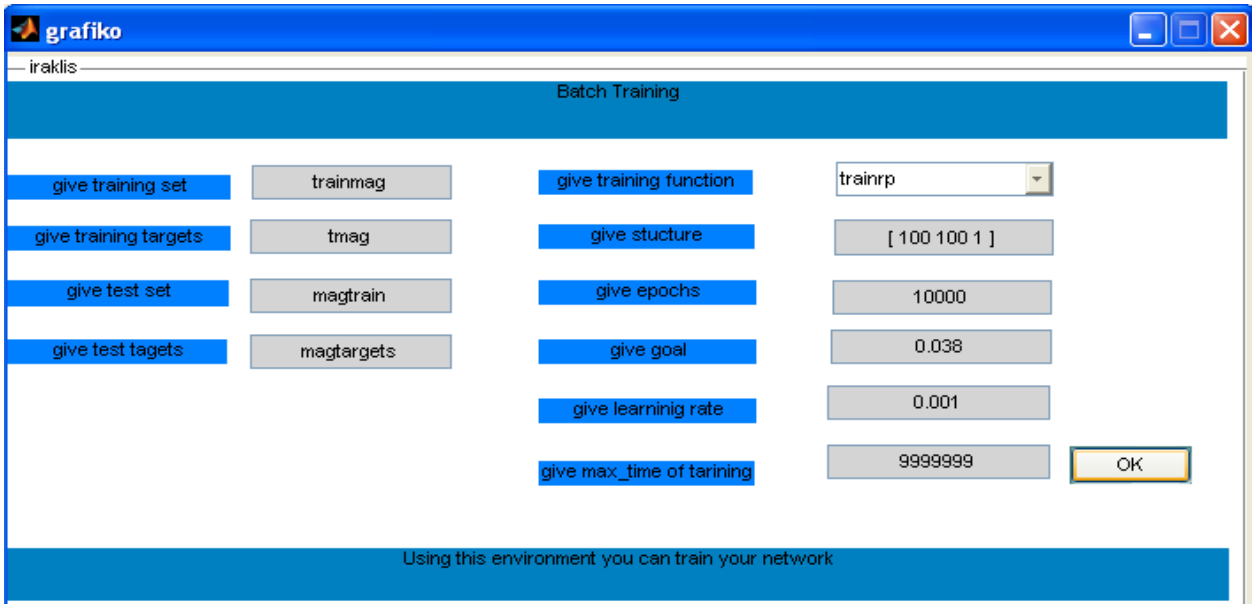
Εικόνα 9.26 Δομή δικτύου 153-140-1



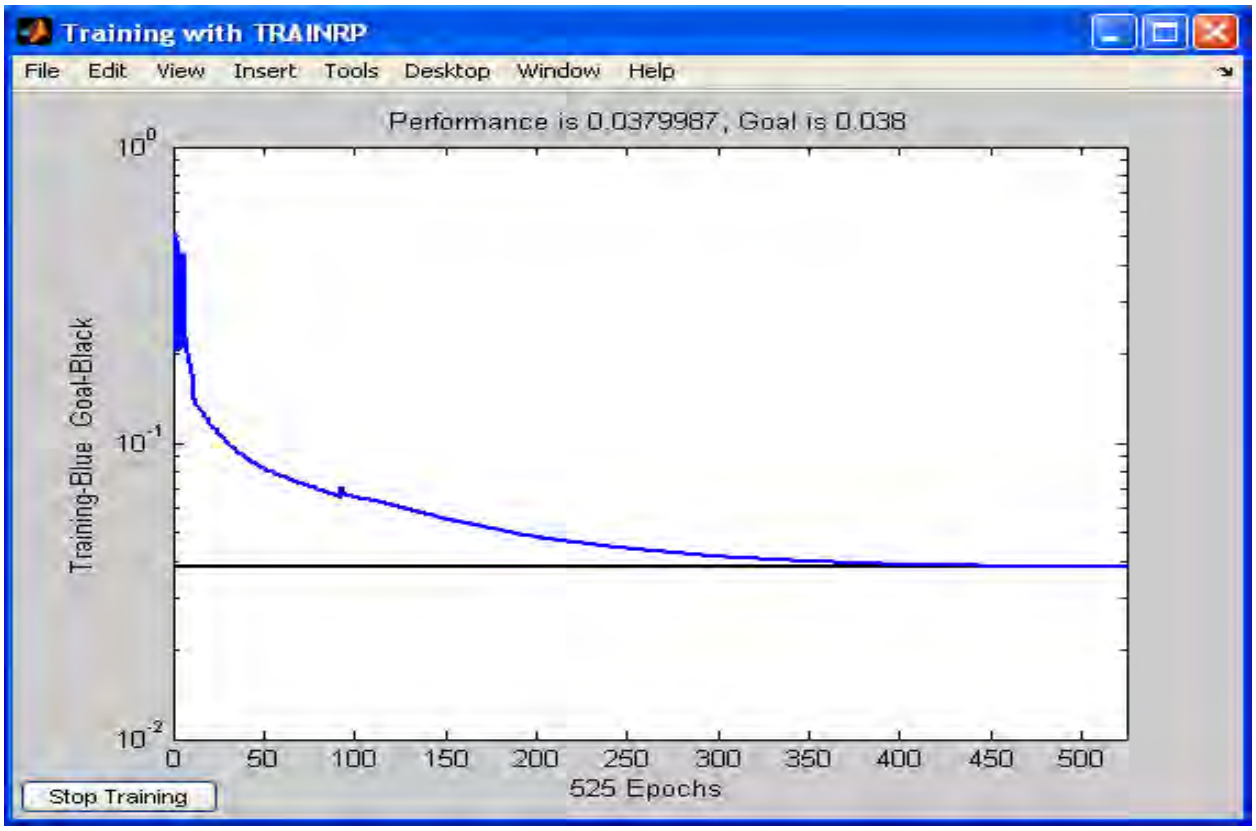
Εικόνα 9.27 Δομή δικτύου 100-100-1
(επιλογή χαρακτηριστικών)

Στο παράδειγμα της Εικόνας 9.27 και σε μια προσπάθεια να ελαττώσουμε το ολικό μέσο τετραγωνικό σφάλμα ακόμη περισσότερο, αφαιρέσαμε τη στήλη ηλικία από τις παρατηρήσεις μας. Παρόλα αυτά η τιμή του ολικού μέσου τετραγωνικού σφάλματος παρέμεινε στα ίδια επίπεδα.

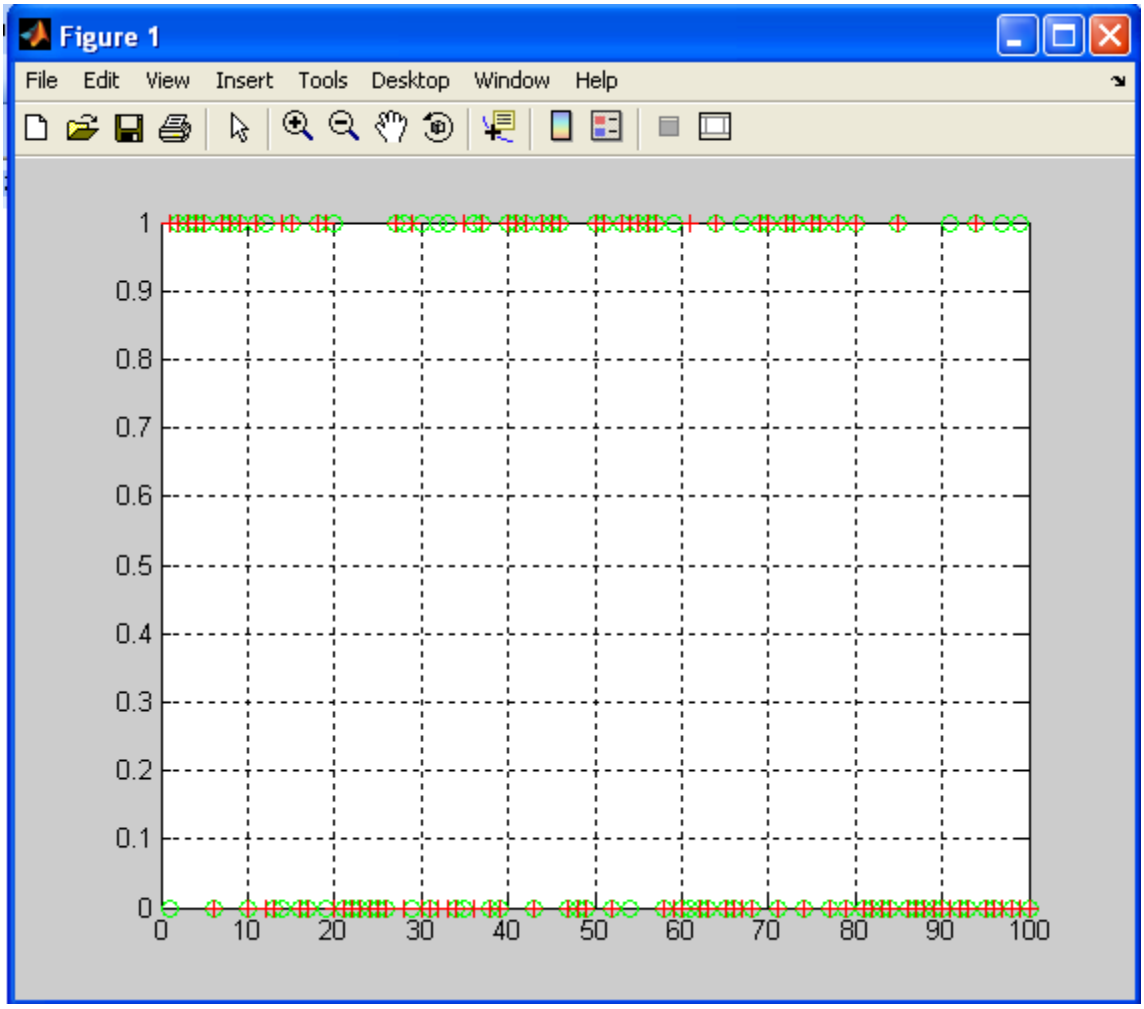
Σε αυτό το σημείο θα κάνουμε χρήση του γραφικού περιβάλλοντος που δημιουργήσαμε και θα παρουσιάσουμε τα αντίστοιχα αποτελέσματα κάθε φορά. Θα χρησιμοποιήσουμε τα ίδια δεδομένα όπως και πριν.



Εικόνα 9.1 Δίνουμε τιμές στο πρόγραμμα



Εικόνα 9.2 Εδώ παρατηρούμε ότι το δίκτυο συγκλίνει στη τιμή 0.0379987 στη 525η εποχή



Εικόνα 9.3 Οι τιμές οι οποίες μέσα στο πράσινο κύκλο έχουν κι ένα κόκκινο σταυρό σημαίνει ότι εκπαιδεύτηκαν σωστά

Τα αποτελέσματα που εξάγει το πρόγραμμα όσον αφορά στο υπολογισμό του μέσου ολικού τετραγωνικού σφάλματος, την απόδοση στην εκπαίδευση, την απόδοση στην προσομοίωση σε νέα δεδομένα καθώς και ιδιότητες του δικτύου παρουσιάζονται παρακάτω:

mse_ =

0.1612

correct_train =

93.9394

correct_test =

81

net =

Neural Network object:

architecture:

numInputs: 1
numLayers: 3
biasConnect: [1; 1; 1]
inputConnect: [1; 0; 0]
layerConnect: [0 0 0; 1 0 0; 0 1 0]
outputConnect: [0 0 1]
targetConnect: [0 0 1]

numOutputs: 1 (read-only)
numTargets: 1 (read-only)
numInputDelays: 0 (read-only)
numLayerDelays: 0 (read-only)

subobject structures:

inputs: {1x1 cell} of inputs
layers: {3x1 cell} of layers
outputs: {1x3 cell} containing 1 output
targets: {1x3 cell} containing 1 target
biases: {3x1 cell} containing 3 biases
inputWeights: {3x1 cell} containing 1 input weight
layerWeights: {3x3 cell} containing 2 layer weights

functions:

adaptFcn: 'trains'
initFcn: 'initlay'
performFcn: 'mse'
trainFcn: 'trainrp'

parameters:

adaptParam: .passes
initParam: (none)
performParam: (none)
trainParam: .epochs, .show, .goal, .time,
.min_grad, .max_fail, .delt_inc, .delt_dec,
.delta0, .deltamax, .lr

weight and bias values:

IW: {3x1 cell} containing 1 input weight matrix

LW: {3x3 cell} containing 2 layer weight matrices
b: {3x1 cell} containing 3 bias vectors

other:

userdata: (user stuff)

mse_ =

0.1612

correct_train =

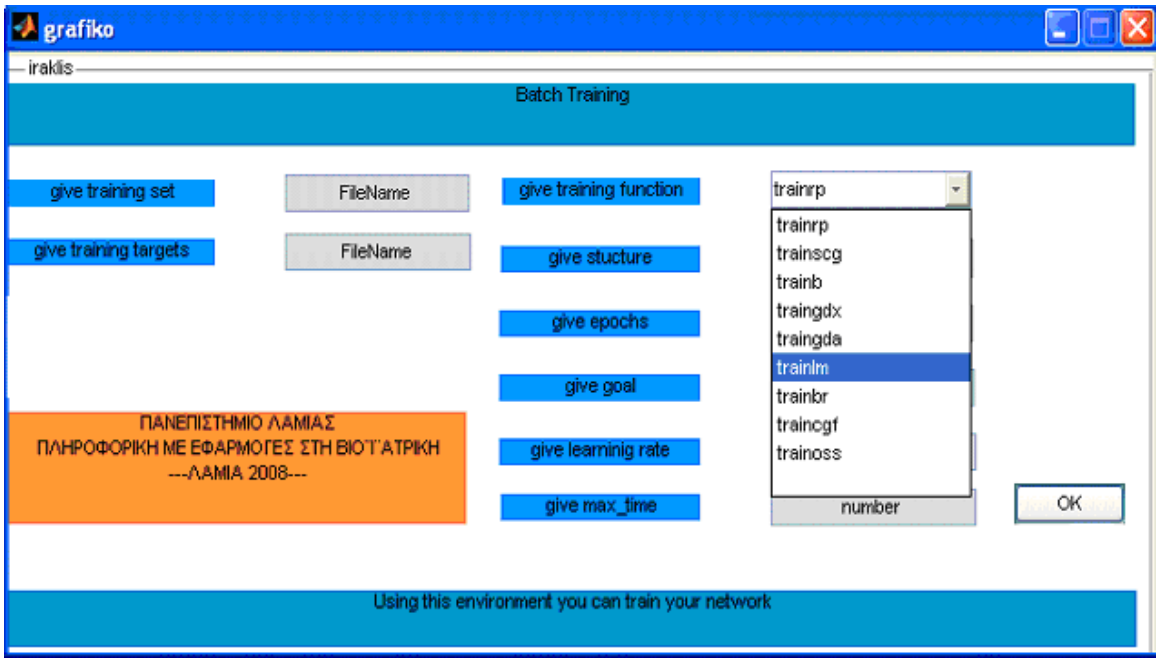
93.9394

correct_test =

81

Ακολουθώντας τη βιβλιογραφία και κάνοντας τις κατάλληλες δοκιμές όπως φαίνεται παραπάνω καταφέραμε να δημιουργήσουμε ένα πρόγραμμα το οποίο να είναι κατάλληλο στην εκπαίδευση τεχνητών νευρωνικών δικτύων ανά ομάδα προτύπων. Το γραφικό περιβάλλον του προγράμματος είναι ευανάγνωστο και σαφές ώστε ακόμη κι κάποιος που δε γνωρίζει την ύπαρξη και λειτουργία των τεχνητών νευρωνικών δικτύων ή που δεν έχει ευχέρεια στη Matlab να δύναται να χρησιμοποιήσει το παραπάνω πρόγραμμα και να εκπαιδεύσει ένα τεχνητό νευρωνικό δίκτυο. Ο χρήστης, συνεπώς, χωρίς δυσκολία, μπορεί, στο πλαίσιο μικρών πειραμάτων, να εισάγει κάθε φορά στο πρόγραμμα τις απαραίτητες παραμέτρους ή να τις αρχικοποιεί έχοντας ποσοστά επιτυχίας που είναι αρκετά ικανοποιητικά. Ωστόσο, τα δεδομένα που χρησιμοποιήσαμε ανωτέρω ήταν ακεραίου τύπου και για το λόγο αυτό ακολούθησε και δεύτερο πείραμα όπου τα δεδομένα είναι πραγματικοί αριθμοί και με σκοπό της ως άνω περιγραφόμενης αλλαγής να σχολιάσουμε τις διαφορές που παρατηρούνται.

Η γενική μορφή του γραφικού περιβάλλοντος ύστερα από αλλαγές είναι η εξής:



Εικόνα 9.4 Μορφή γραφικού περιβάλλοντος εκπαίδευσης νευρωνικών δικτύων ανά ομάδα προτύπων

Καθώς εισάγαμε, λοιπόν, τα δεδομένα στο πρόγραμμα, αυτό μας έδωσε τόσο τις επιθυμητές εξόδους σε μορφή γραφημάτων, συμφώνως και με τα ανωτέρω, όσο και την παρουσίαση του ολικού μέσου τετραγωνικού σφάλματος, την απόδοση της εκπαίδευσης του δικτύου και την απόδοση της προσομοίωσης σε νέα δεδομένα.

Στην αριστερή στήλη, λοιπόν, εισάγαμε τα ονόματα των αρχείων των δεδομένων. Το πρώτο αρχείο περιελάμβανε τις τιμές των παραμέτρων του διανύσματος, ενώ το δεύτερο τα αποτελέσματα των τιμών στόχων. Στη δεξιά στήλη προσδιορίσαμε τα χαρακτηριστικά της εκπαίδευσης του τεχνητού νευρωνικού δικτύου ανά ομάδα προτύπων. Κατόπιν εισάγαμε μια συνάρτηση εκπαίδευσης, εν προκειμένω ήταν η trainlm. Έπειτα ορίσαμε για πόσες εποχές θέλουμε να τρέξει το πρόγραμμα και προσδιορίσαμε τον αριθμό των επιπέδων και των νευρώνων, την ελάχιστη τιμή του ολικού μέσου τετραγωνικού σφάλματος που θέλουμε να επιτύχουμε και ποιος θα είναι ο ρυθμός εκπαίδευσης καθώς και ο μέγιστος χρόνος κατά τον οποίο θα γίνεται η εκπαίδευση. Το παραπάνω πρόγραμμα ανταποκρίθηκε στο κώδικα που είναι γραμμένος και παρουσιάζεται παρακάτω:

Νέος Κώδικας

```
function [ net, mse_, correct_train, correct_test] =
neural1(train_function, goal_, epochs_, learning_rate, time_, Aset, t ,
ptest, real_t, ad, Cell_)
 I = t';
 s = Aset';
 pp= ptest';
 mi=real_t;

[R,Q] = size(s)

iitst = 8:16:Q;
iival = 16:16:Q;
iitr = [1:2:Q 3:3:Q];
val.P = s(:,iival); val.T = I(:,iival);
test.P = s(:,iitst); test.T = I(:,iitst);
ptr = s(:,iitr); ttr = I(:,iitr);

 net=newff(minmax(ptr), ad, Cell_, train_function, 'learngdm', 'mse' );
 net=init(net);
 net.trainParam.epochs = epochs_;
 net.trainParam.min_grad=0.000000001;
 net.trainParam.goal =goal_;
 net.trainParam.lr =learning_rate;
 net.trainParam.show =1;
 net.trainParam.time = time_;

net.trainParam.max_fail=1000000000000000000000000000000000000000000000000
0000000000000000000000;
 [net,tr]=train(net,ptr,ttr,[],[],val,test);
 plot(tr.epoch,tr.perf,tr.epoch,tr.vperf,tr.epoch,tr.tperf)
 legend('Training','Validation','Test',-1);
 ylabel('Squared Error'); xlabel('Epoch')
 Y=sim(net,ptr);
 mse_ = mse(ttr - Y)
 YY=sim(net, test.P);

 k=randint()+5;
 if ((I(k)==1) | (I(k)==0))
     flag=1
 else flag=0
 end
     if flag==1
          figure;plot( test.T, 'bo');
           grid on;
           hold on;
           plot(sign(YY' - 0.5)==1, 'r+')
     else
           figure;plot( test.T, 'bo');
           grid on;
           hold on;
           plot(YY', 'r+')
     end

  correct_train = 100*(length(ttr) - sum(abs((sign(Y - 0.5)==1) - ttr))
)/length(ttr)
```

```
 correct_test  = 100*(length(test.T) - sum(abs((sign(YY - 0.5)==1) -
test.T)))/length(test.T)
```

Η διαφορά του δεύτερου κώδικα σε σχέση με τον πρώτο που παρουσιάσαμε είναι η εξής: Στον δεύτερο κώδικα ορίσαμε ως τιμές εισόδου τα δεδομένα με τις παρατηρήσεις και τους στόχους και αυτομάτως υπολογίστηκε το διάνυσμα της εκπαίδευσης και το διάνυσμα της προσομοίωσης. Αντιθέτως στον πρώτο κώδικα ορίζαμε επιπροσθέτως το διάνυσμα της εκπαίδευσης και το διάνυσμα της προσομοίωσης. Επιπλέον, την δεύτερη φορά το πρόγραμμα τροποποιήθηκε έτσι ώστε να εμφανίζεται η καμπύλη εκπαίδευσης και η καμπύλη προσομοίωσης.

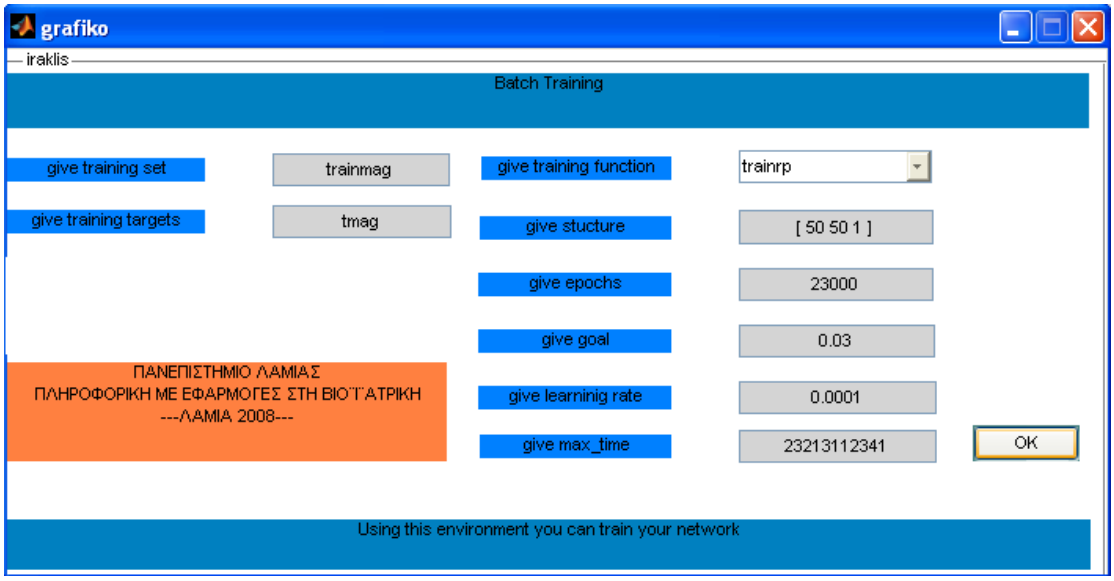
Παρακάτω θα ακολουθήσουν δοκιμές του δεύτερου προγράμματος όπως και πριν, μόνο που αυτή τη φορά θα χρησιμοποιηθούν και δεδομένα που δημιουργήσαμε εμείς χρησιμοποιώντας τη συνάρτηση rand() του εκπαιδευτικού λογισμικού Matlab.

Πείραμα 2

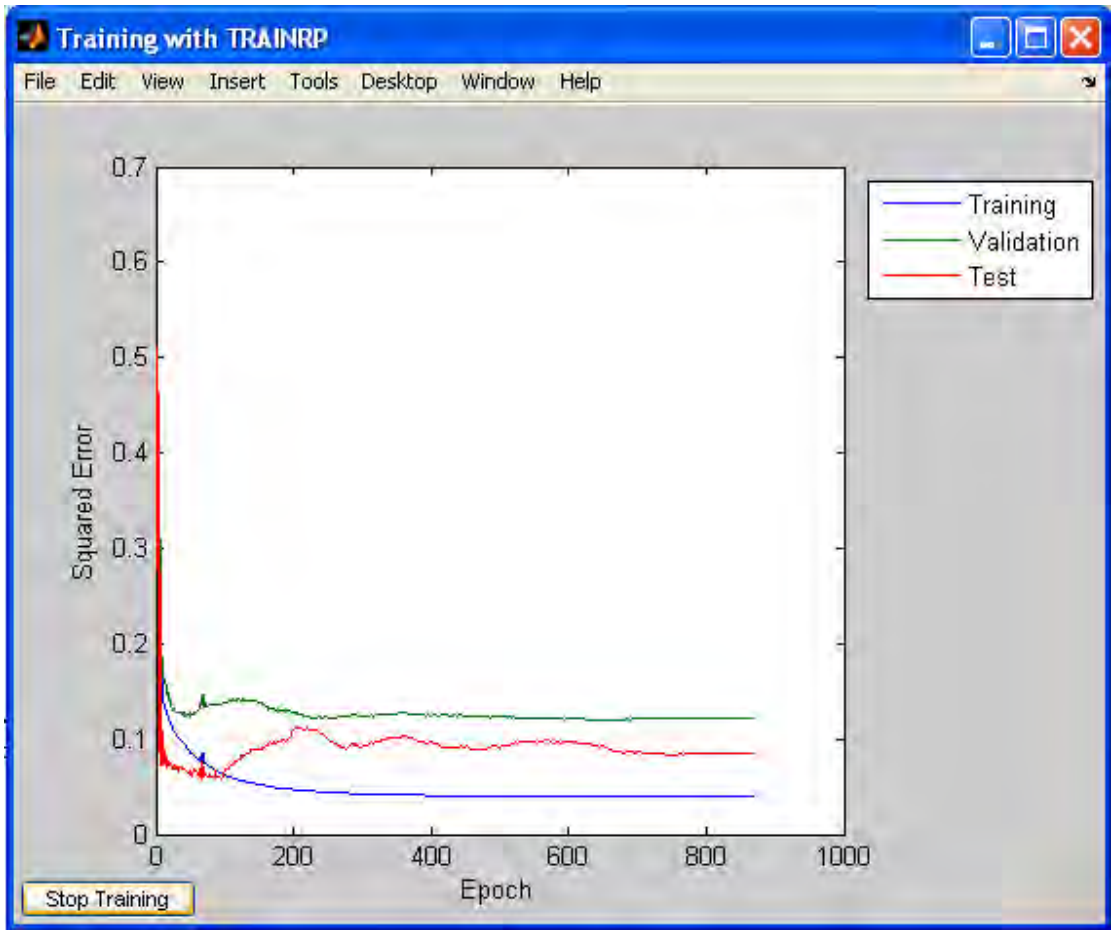
Το παρακάτω αναλυόμενο πείραμα ήταν παρεμφερές με το υπ' αριθμ. 1 πείραμα. Η διαφορά τους έγκειται ότι στο υπ' αριθμ. 2 πείραμα κάναμε σύγκριση των αποτελεσμάτων μεταξύ αρχείων με ακέραιο τύπο δεδομένων και μεταξύ αρχείων που τα δεδομένα τους ακολουθούν τυχαίες τιμές στο διάστημα [0,1). Συγκεκριμένα για το πρώτο αρχείο οι στόχοι πήραν τιμές 0 ή 1 και οι τιμές των παραμέτρων ήταν διακριτές. Αντίθετα, για το δεύτερο αρχείο οι στόχοι πήραν τιμές στο διάστημα [0,1) όπως επίσης και οι τιμές των παραμέτρων. Σκοπός ήταν να παρατηρήσουμε σε ποιες περιπτώσεις το πρόγραμμα ανταποκρινόταν καλύτερα. Παρακάτω ακολουθεί ο συγκριτικός Πίνακας 9.2 με τα αποτελέσματα που καταγράψαμε καθώς και τα αντίστοιχα διαγράμματα κάθε φορά. Επίσης ακολουθούν τα αποτελέσματα που εξήγαγε το πρόγραμμα σε μορφή διαγραμμάτων καθώς και η οπτικοποίηση των αποτελεσμάτων του πίνακα υπό τη μορφή γραφημάτων.

Πίνακας 9.2
Πίνακας δοκιμών – αποτελεσμάτων

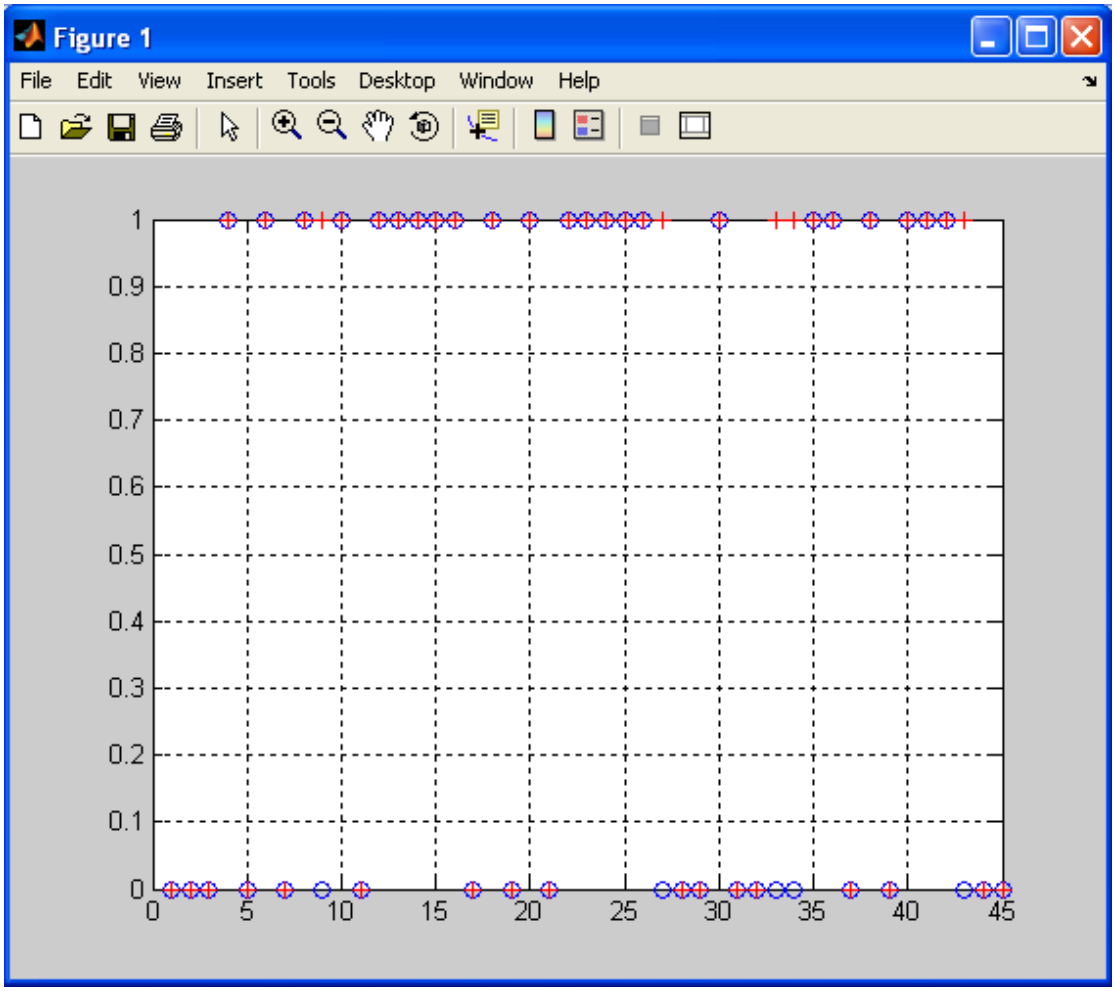
Τύπος δεδομένων	Συνάρτηση εκπαίδευσης	Δομή	Εποχές	Mse(προσομοίωσης)	Correct training%	Correct test%	Σχήμα
1) Ακέραοι	Trainrp	50-50-1	900	0.0394	94.2149	88.8889	9.5.α / 9.5.β / 9.5.γ
2) Πραγματικοί	Trainrp	50-50-1	14	0.0094	77.5458	77.8211	9.6.α / 9.6.β / 9.6.γ
3) Ακέραοι	Traingdx	50-50-1	10300	0.0999	87.2727	95.5556	9.7.α / 9.7.β / 9.7.γ
4) Πραγματικοί	Traingdx	50-50-1	610	0.0081	75.5755	78.2468	9.8.α / 9.8.β / 9.8.γ
5) Ακέραοι	Trainlm	20-20-1	390	0.08347	86.1157	93.333	9.9.α / 9.9.β/ 9.9.γ
6) Πραγματικοί	Trainlm	20-20-1	790	0.0067	77.6836	77.7426	9.10./ 9.10./ 9.10.γ
7) Ακέραοι	Traincgf	20-20-1	750	0.1038	86.7769	91.1111	9.11./ 9.11./ 9.11.γ
8) Πραγματικοί	Traincgf	20-20-1	732	0.0075	77.6979	77.7426	9.12./ 9.12./ 9.12.γ



Εικόνα 9.5.α Εισαγωγή δεδομένων και τιμών στο πρόγραμμα



Εικόνα 9.5.β Γραφική απεικόνιση των αποτελεσμάτων



Εικόνα 9.5.γ Απεικόνιση των τιμών στόχων (πραγματικές τιμές) και των υπολογιζόμενων τιμών

Αποτελέσματα προγράμματος:

mse_ =

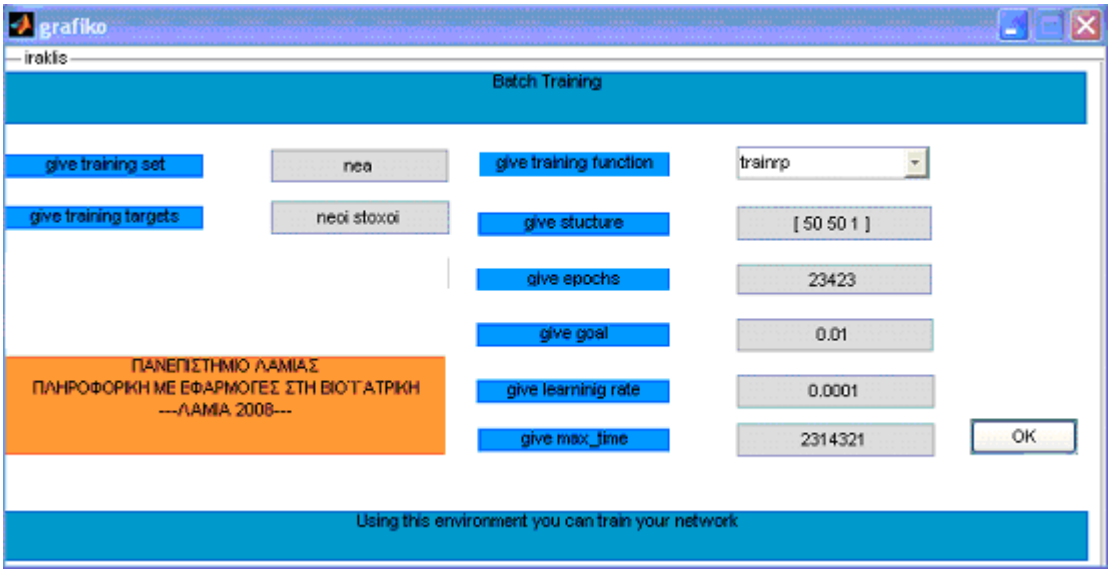
0.0394

correct_train =

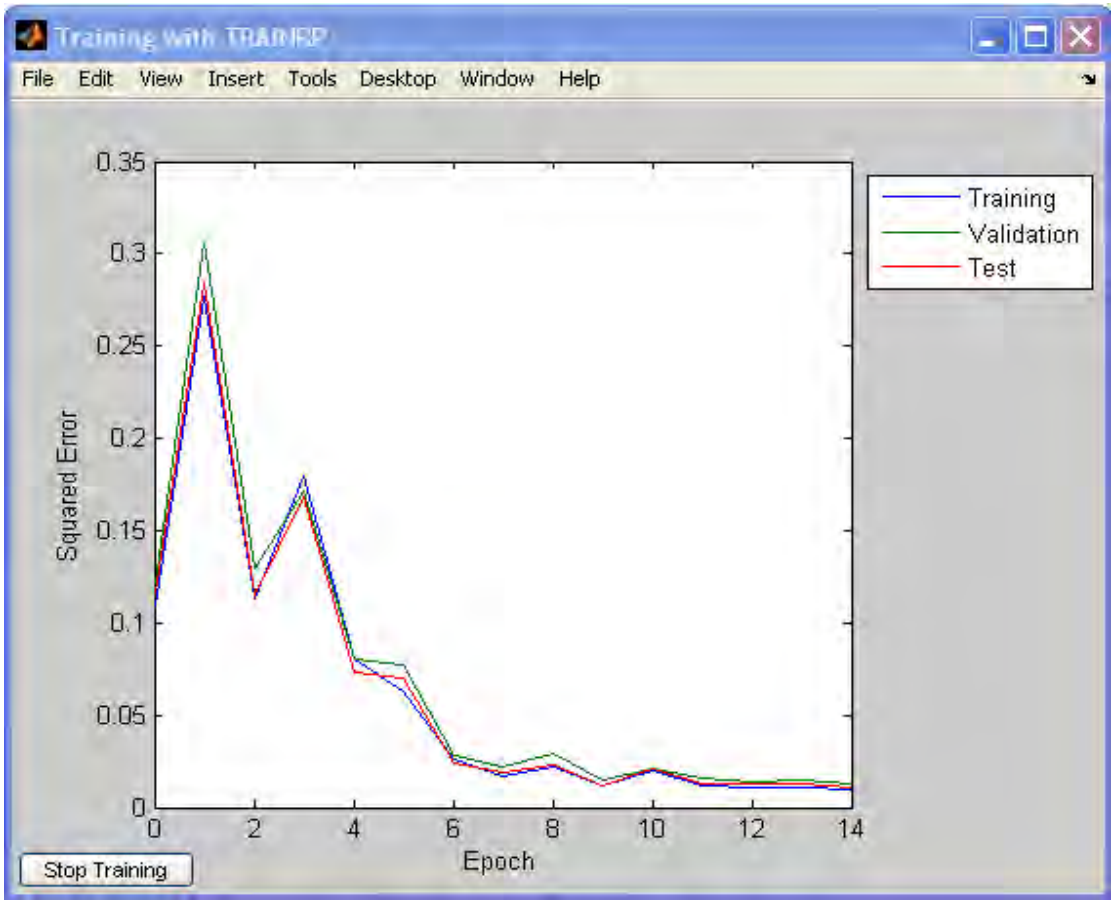
94.2149

correct_test =

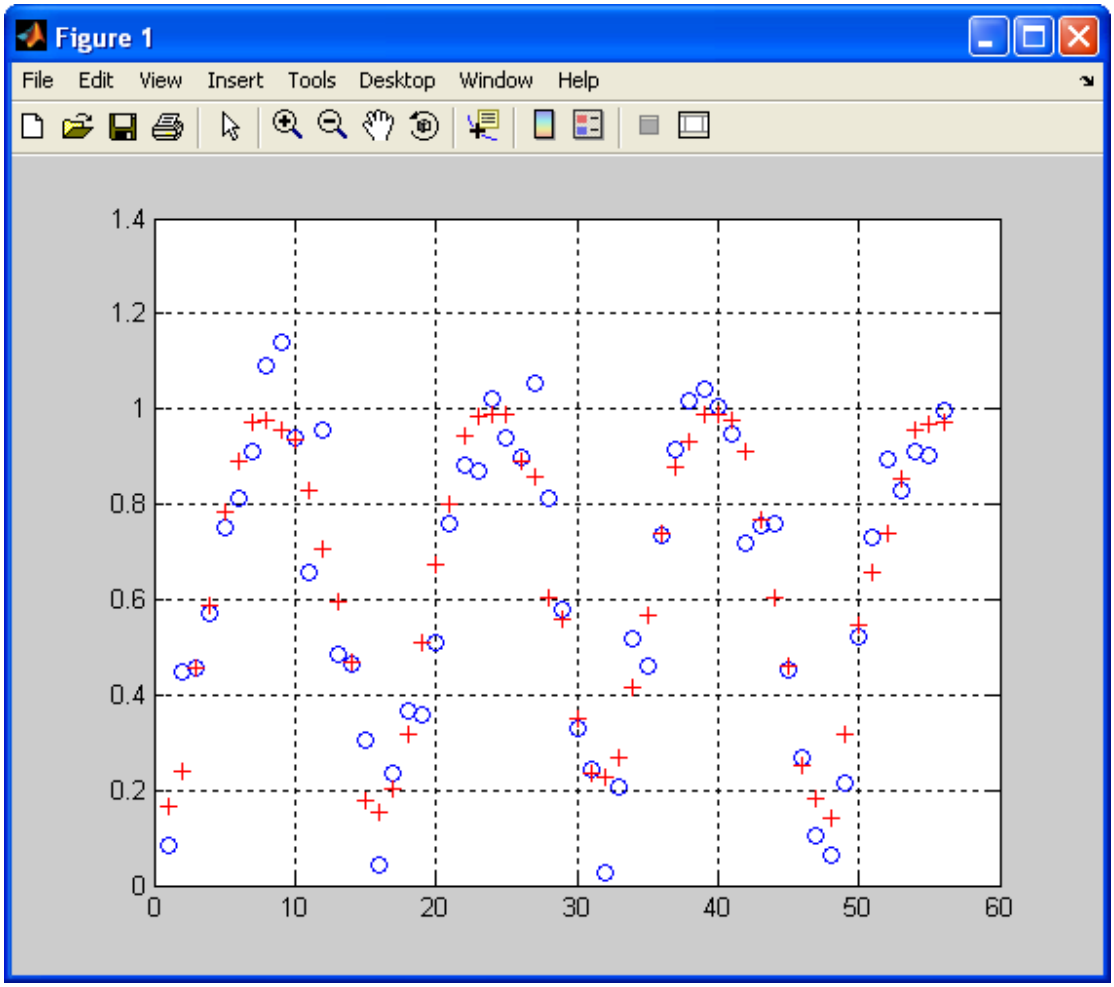
88.8889



Εικόνα 9.6.α Εισαγωγή δεδομένων και τιμών στο πρόγραμμα



Εικόνα 9.6.β Γραφική απεικόνιση των αποτελεσμάτων



Εικόνα 9.6.γ Απεικόνιση των τιμών στόχων (πραγματικές τιμές) και των υπολογιζόμενων τιμών

Αποτελέσματα προγράμματος:

mse_ =

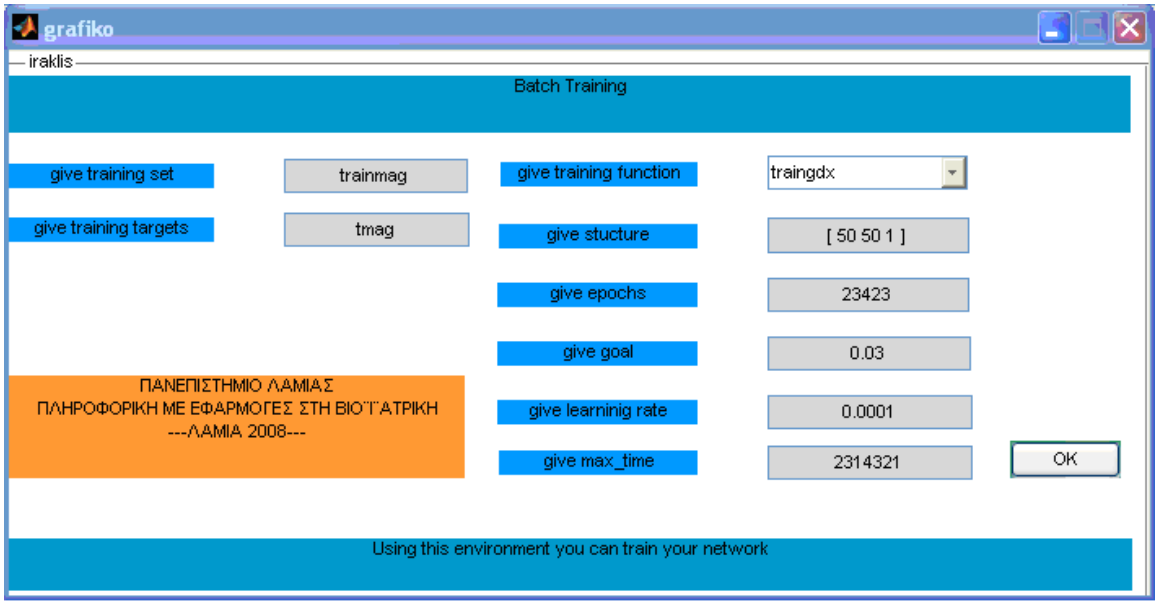
0.0094

correct_train =

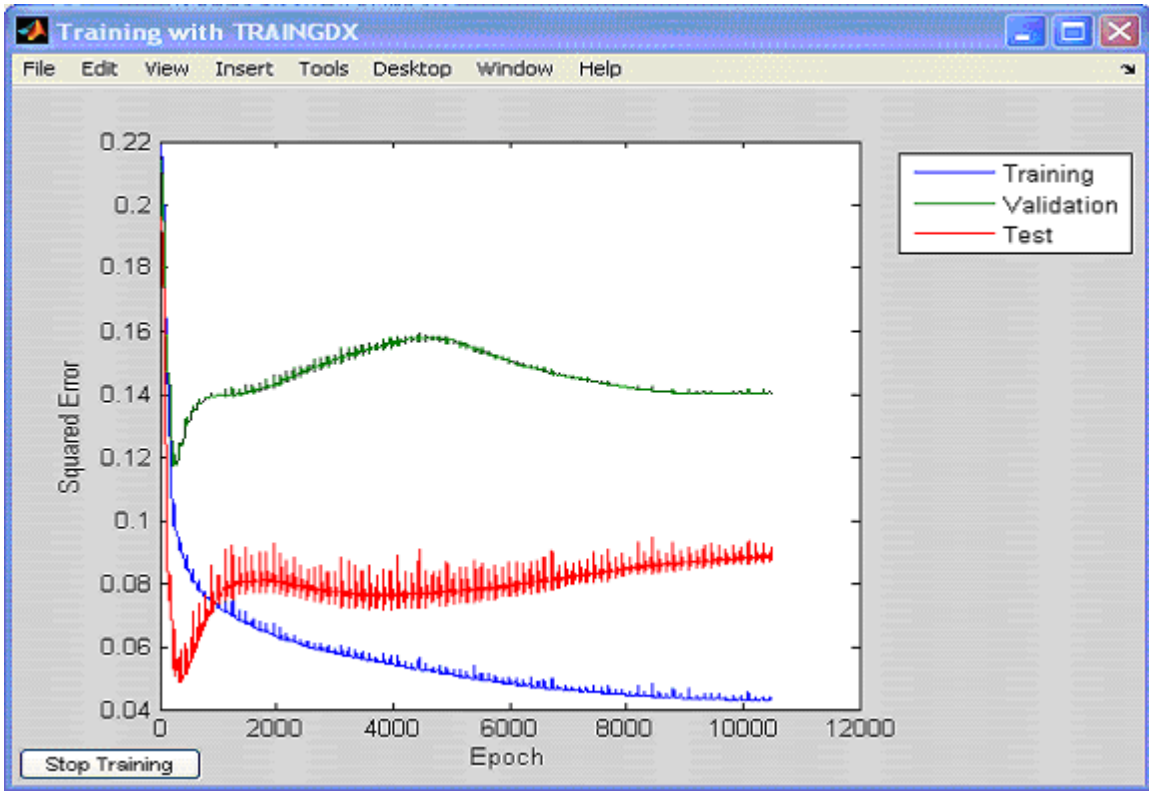
77.5458

correct_test =

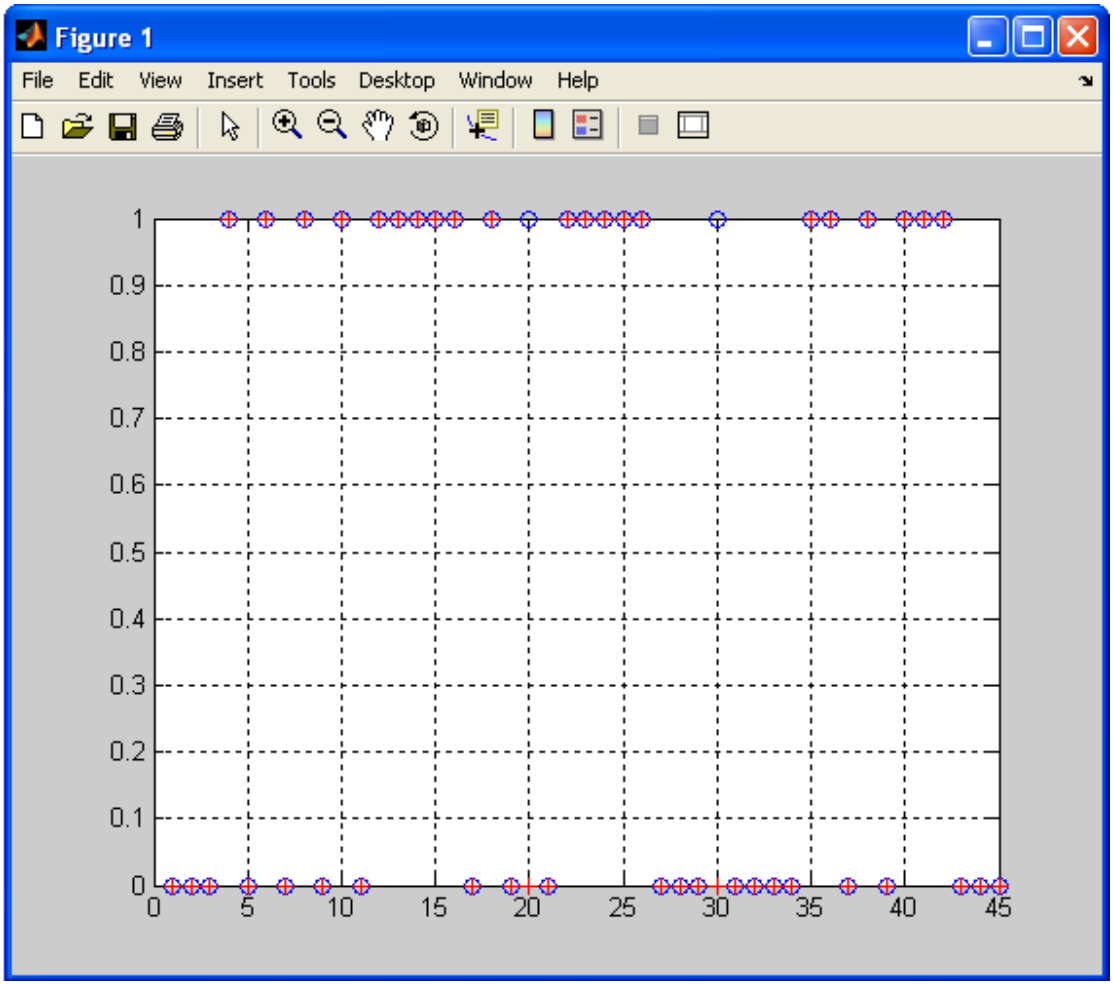
77.821



Εικόνα 9.7.α Εισαγωγή δεδομένων και τιμών στο πρόγραμμα



Εικόνα 9.7.β Γραφική απεικόνιση των αποτελεσμάτων



Εικόνα 9.7.γ Απεικόνιση των τιμών στόχων (πραγματικές τιμές) και των υπολογιζόμενων τιμών

Αποτελέσματα προγράμματος:

mse_ =

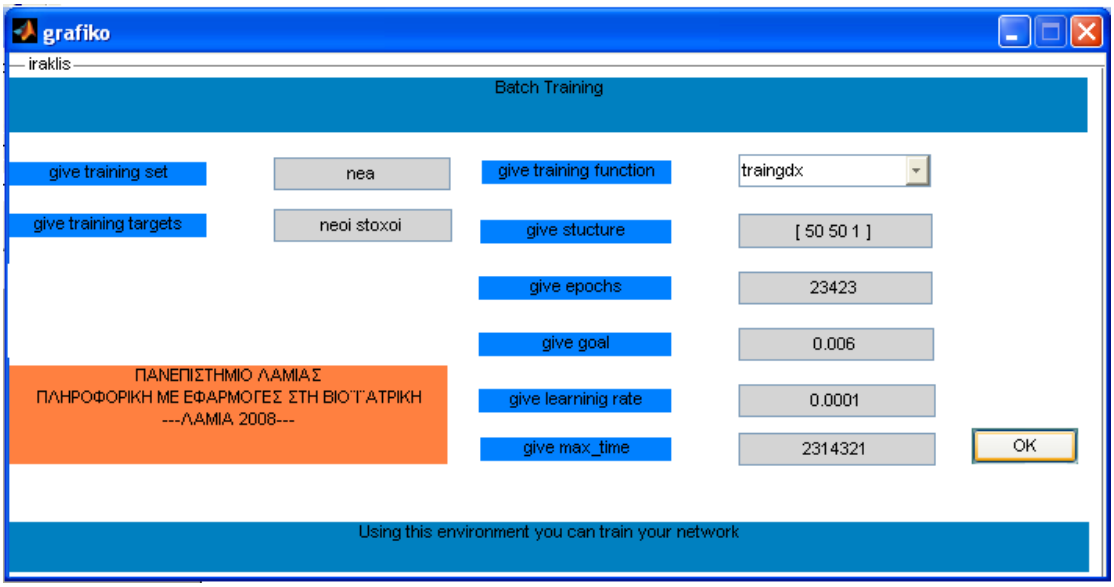
0.0999

correct_train =

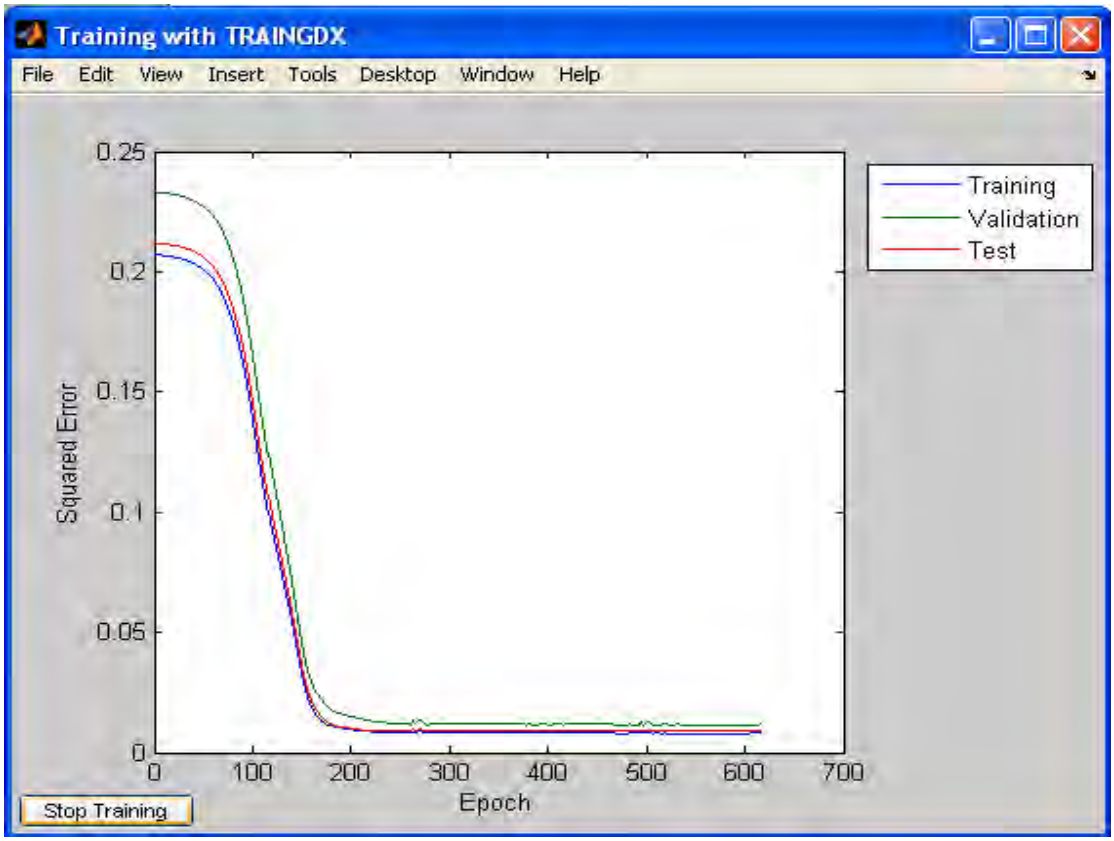
87.2727

correct_test =

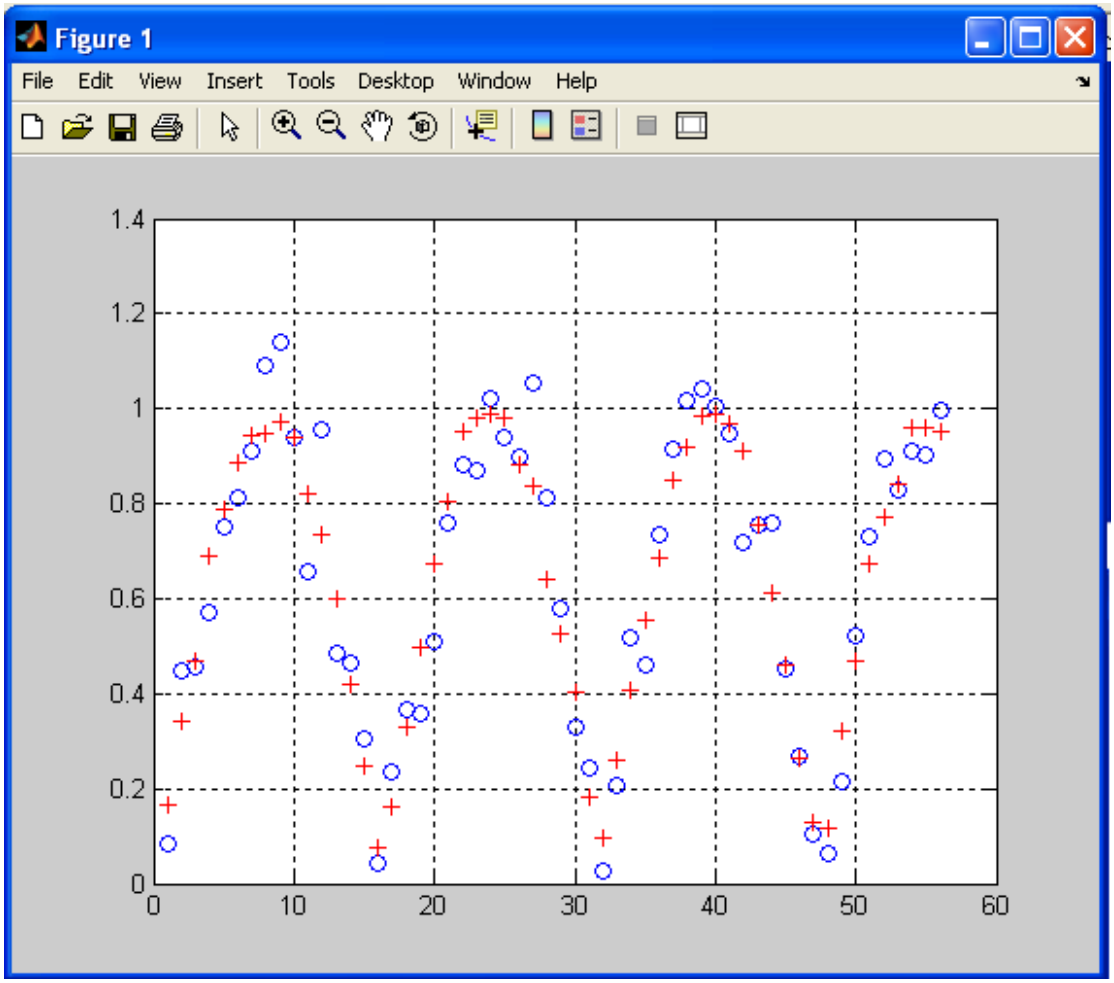
95.5556



Εικόνα 9.8.α Εισαγωγή δεδομένων και τιμών στο πρόγραμμα



Εικόνα 9.8.β Γραφική απεικόνιση των αποτελεσμάτων



Εικόνα 9.8.γ Απεικόνιση των τιμών στόχων (πραγματικές τιμές) και των υπολογιζόμενων τιμών

Αποτελέσματα προγράμματος:

mse_ =

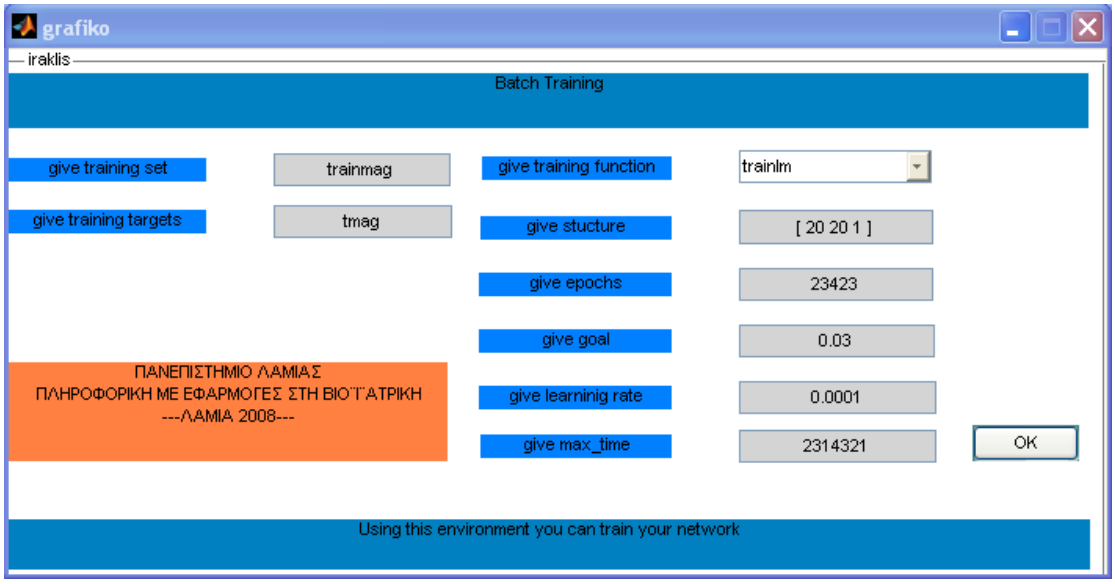
0.0081

correct_train =

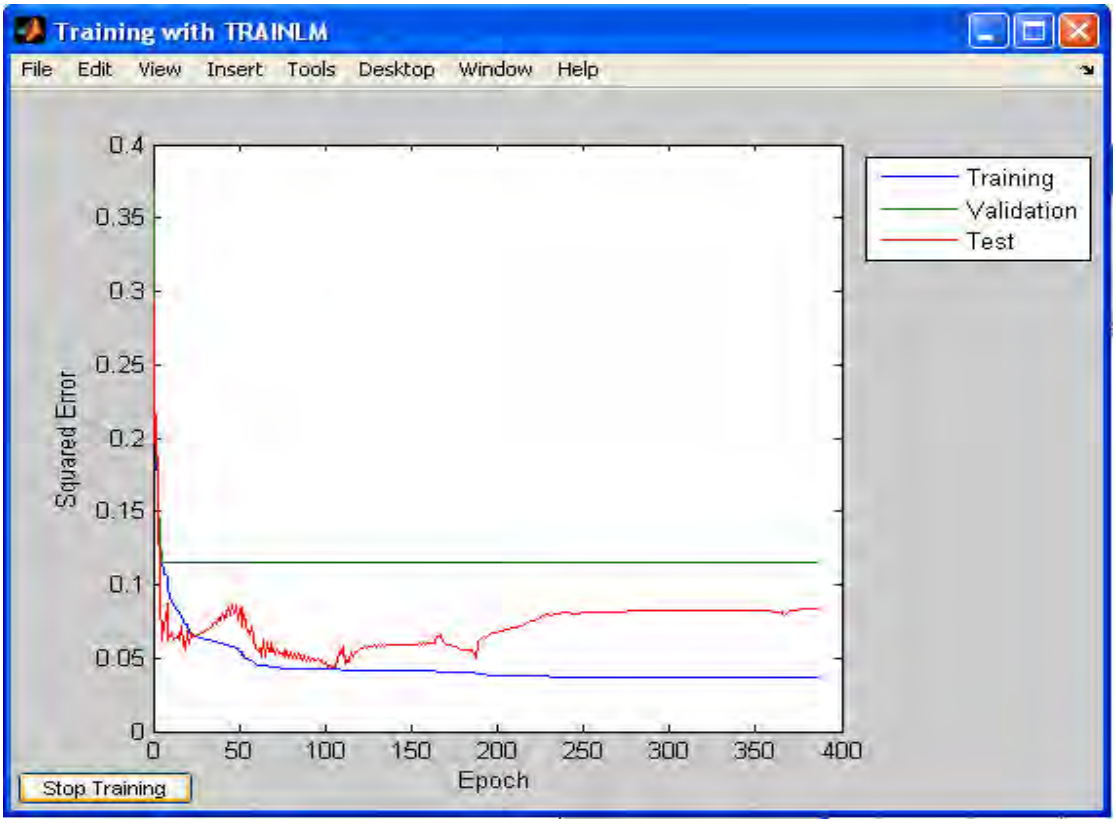
77.5755

correct_test =

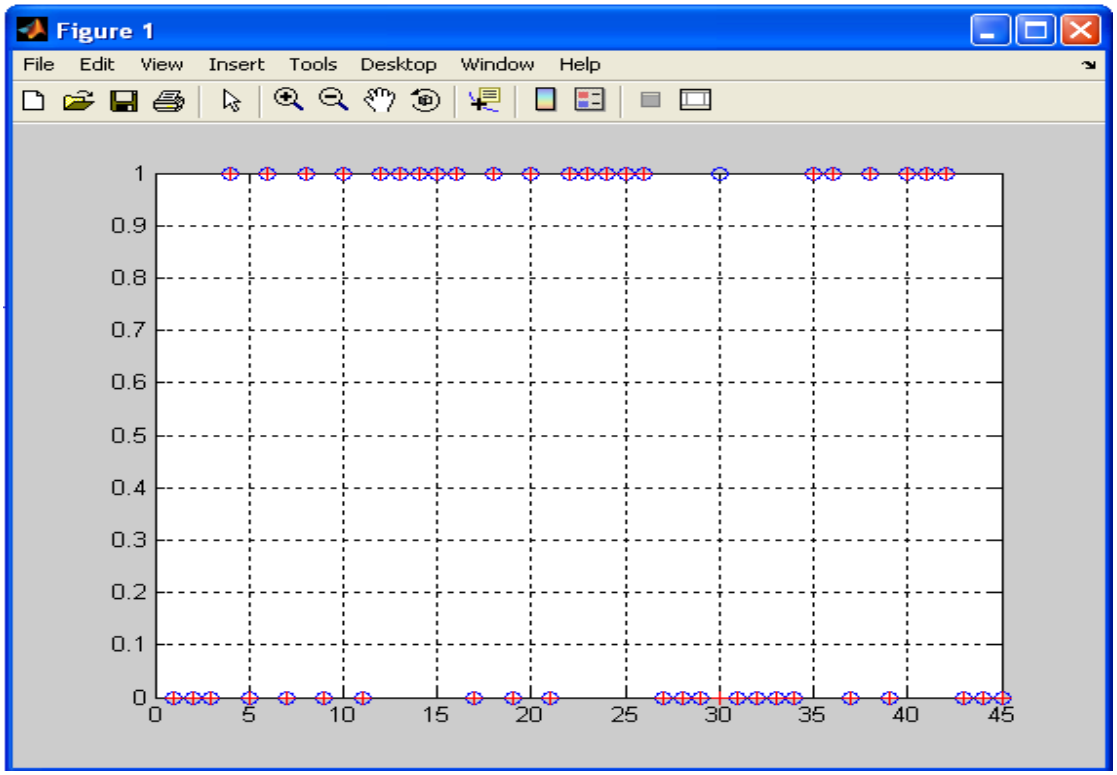
78.2468



Εικόνα 9.9.α Εισαγωγή δεδομένων και τιμών στο πρόγραμμα



Εικόνα 9.9.β Γραφική απεικόνιση των αποτελεσμάτων



Εικόνα 9.9.γ Απεικόνιση των τιμών στόχων (πραγματικές τιμές) και των υπολογιζόμενων τιμών

Αποτελέσματα προγράμματος:

mse_ =

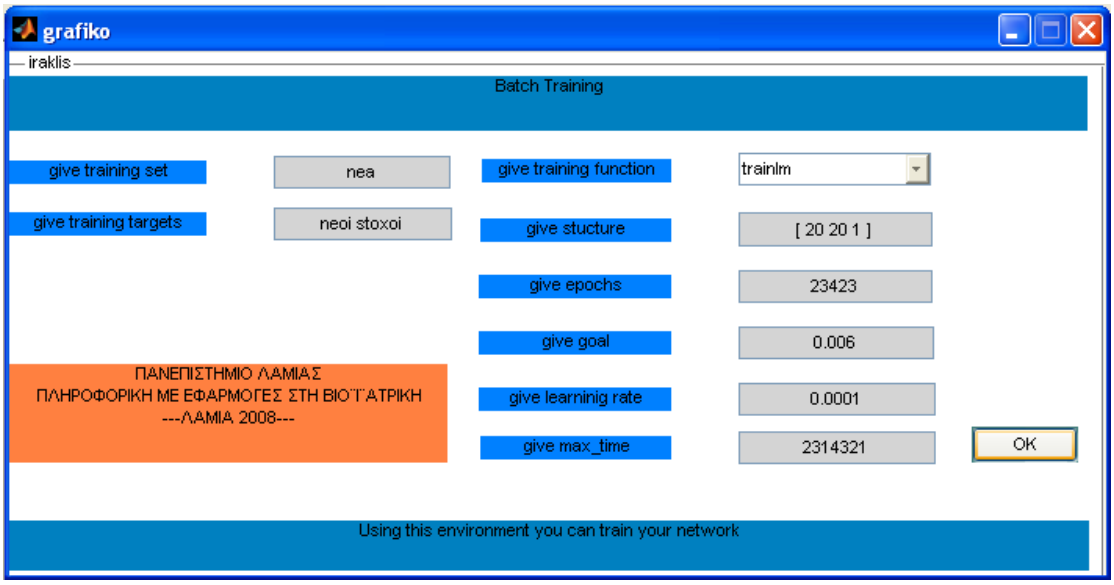
0.08347

correct_train =

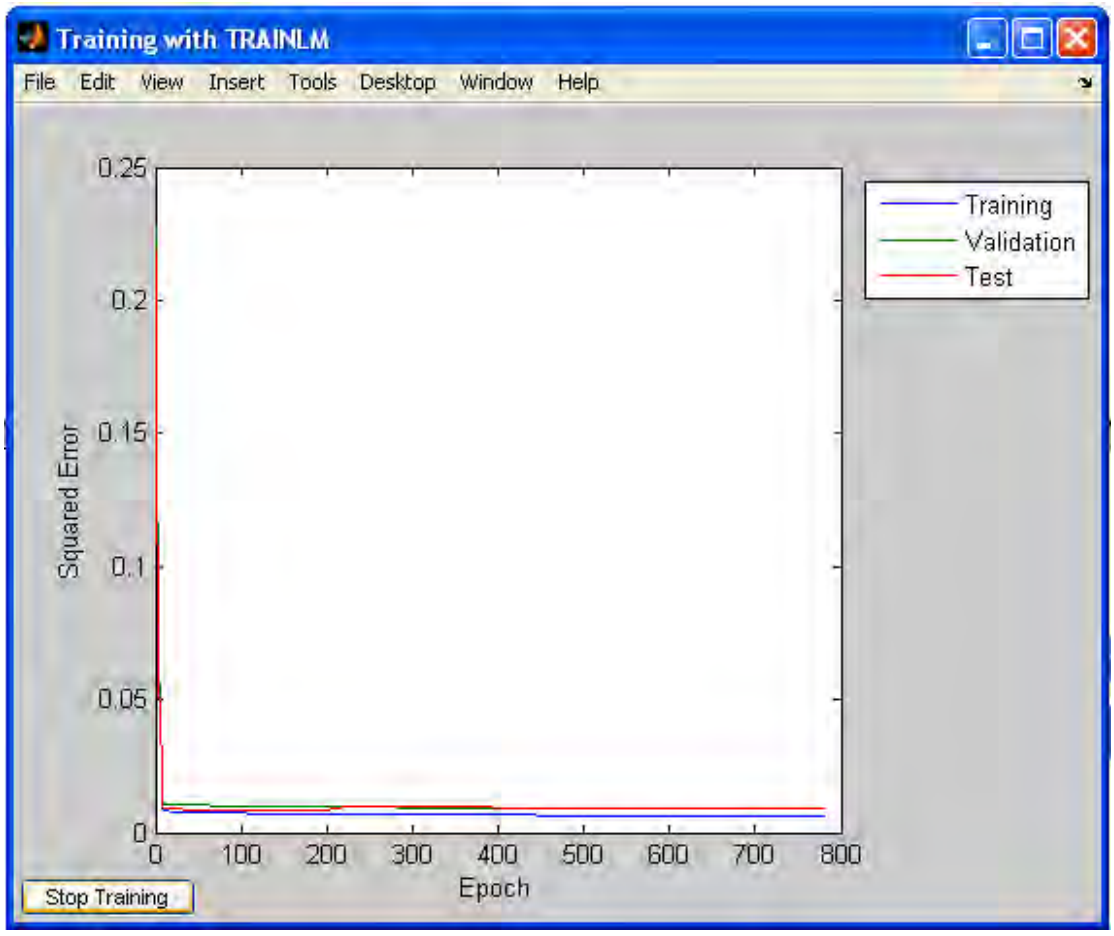
86.1157

correct_test =

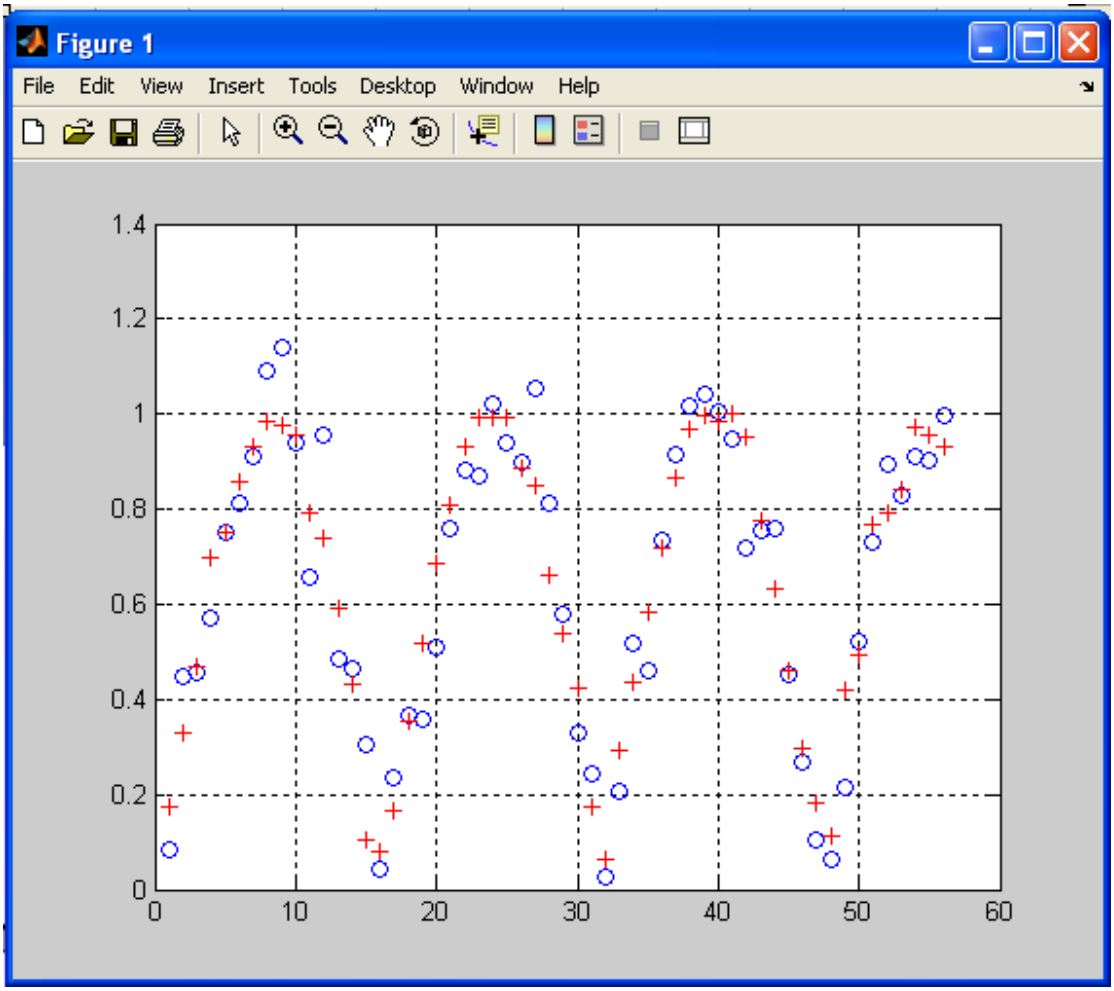
93.3333



Εικόνα 9.10.α Εισαγωγή δεδομένων και τιμών στο πρόγραμμα



Εικόνα 9.10.β Γραφική απεικόνιση των αποτελεσμάτων



Εικόνα 9.10.γ Απεικόνιση των τιμών στόχων (πραγματικές τιμές) και των υπολογιζόμενων τιμών

Αποτελέσματα προγράμματος:

mse_ =

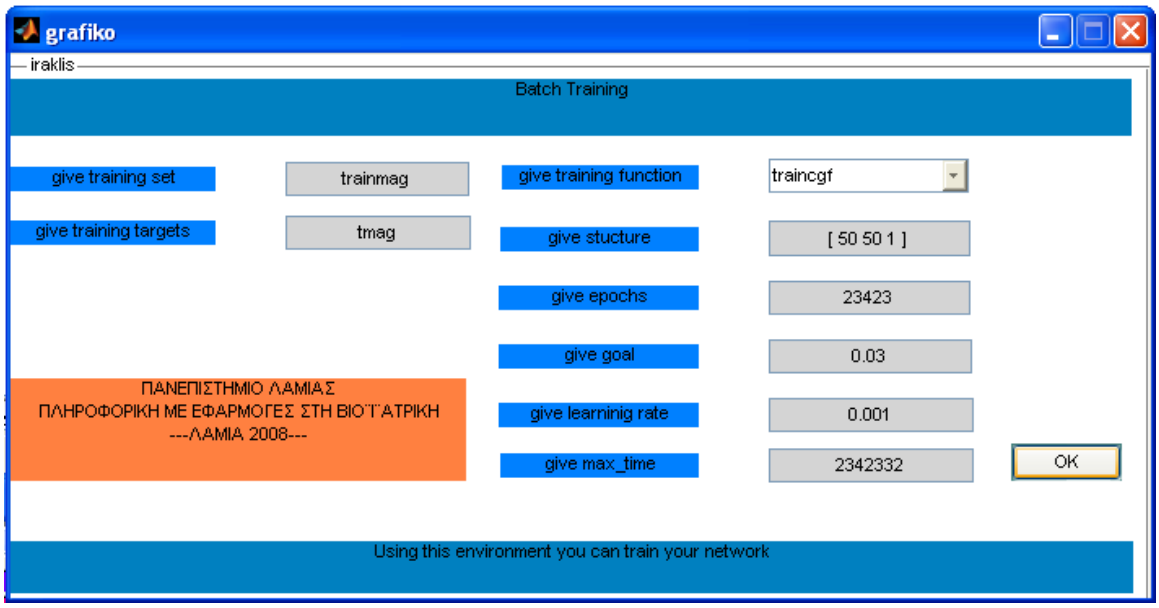
0.0067

correct_train =

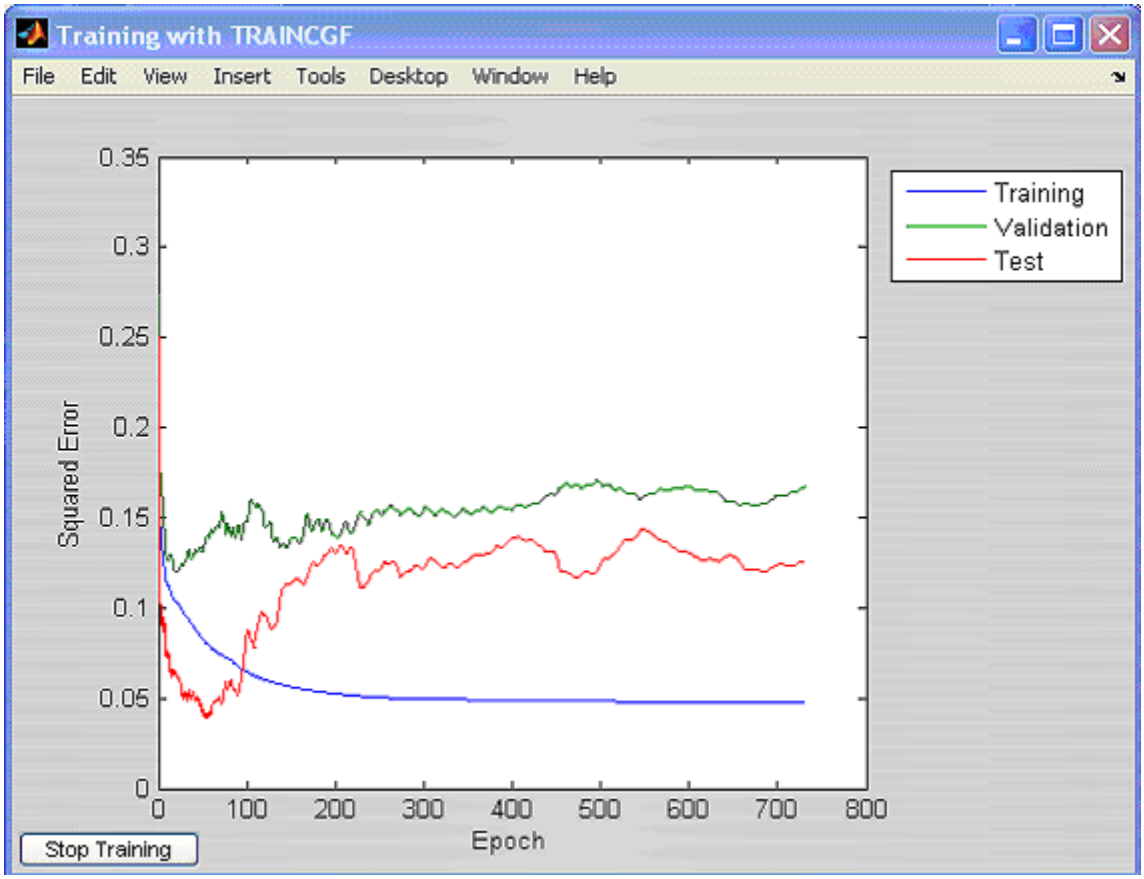
77.6836

correct_test =

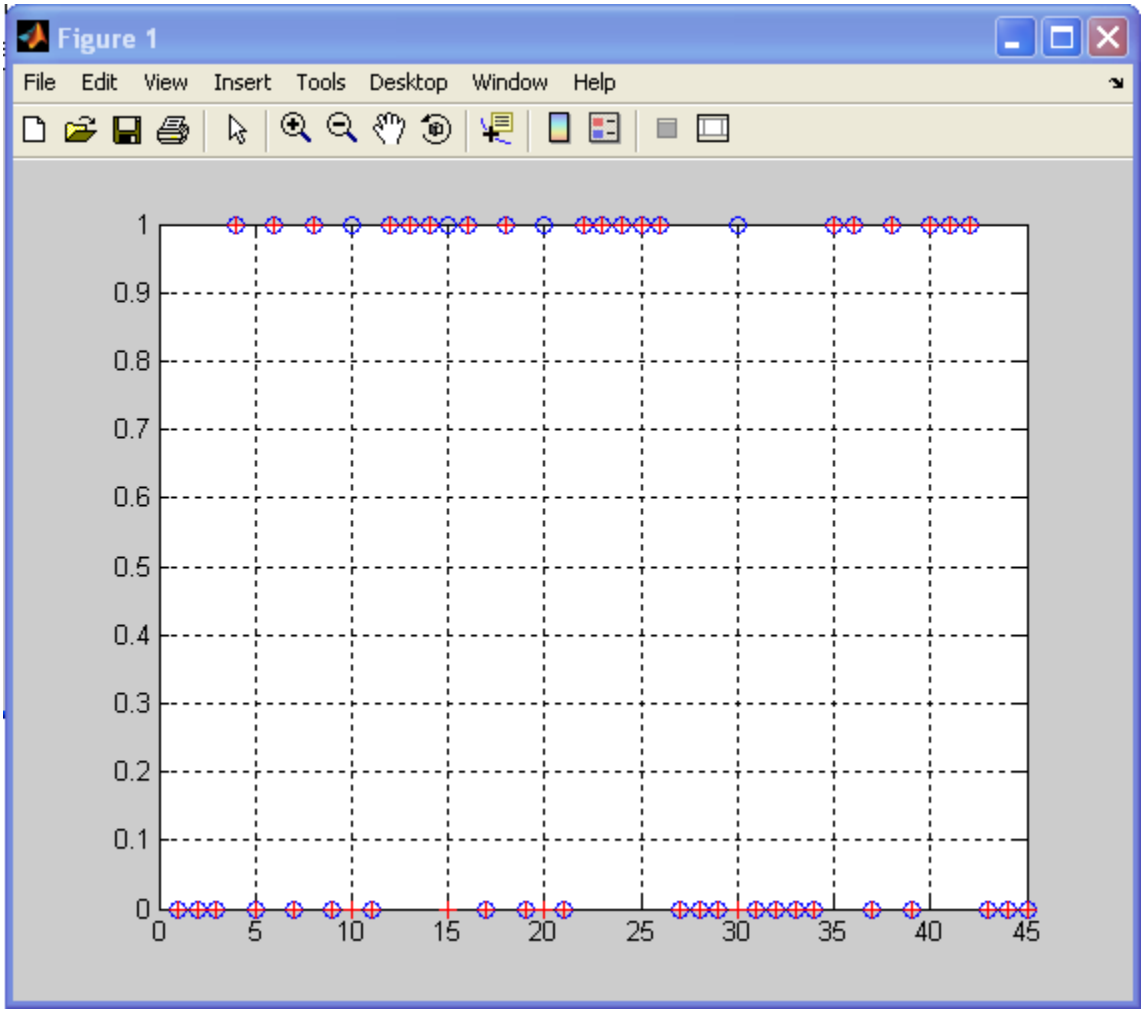
77.7426



Εικόνα 9.11.α Εισαγωγή δεδομένων και τιμών στο πρόγραμμα



Εικόνα 9.11.β Γραφική απεικόνιση των αποτελεσμάτων



Εικόνα 9.11.γ Απεικόνιση των τιμών στόχων (πραγματικές τιμές) και των υπολογιζόμενων τιμών

Αποτελέσματα προγράμματος:

mse_ =

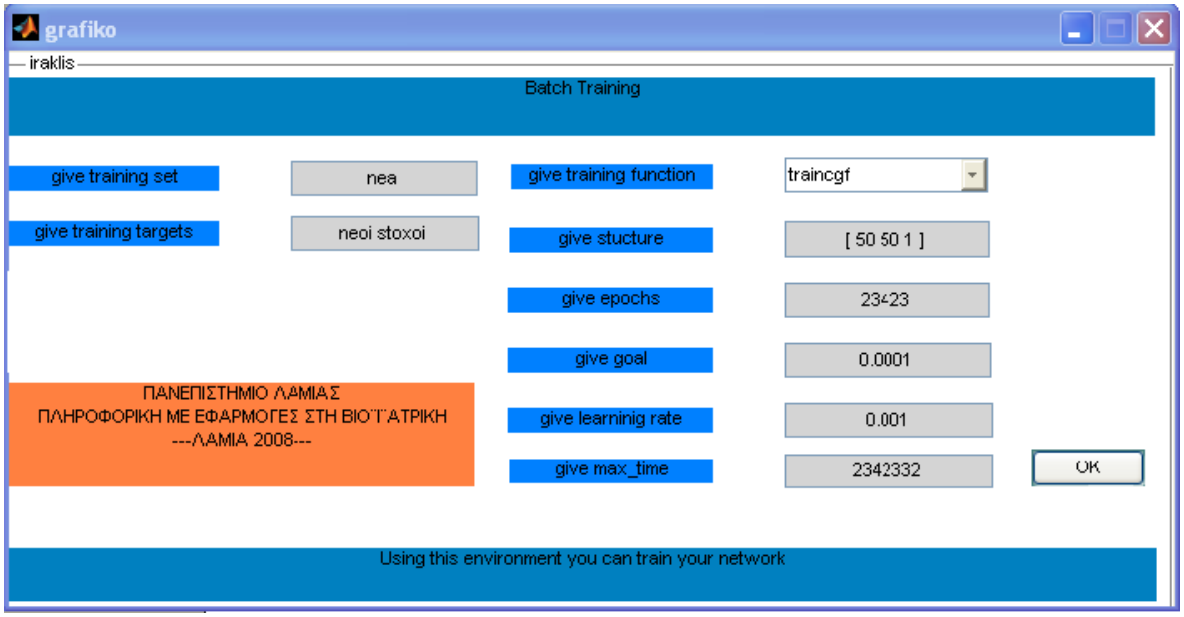
0.1038

correct_train =

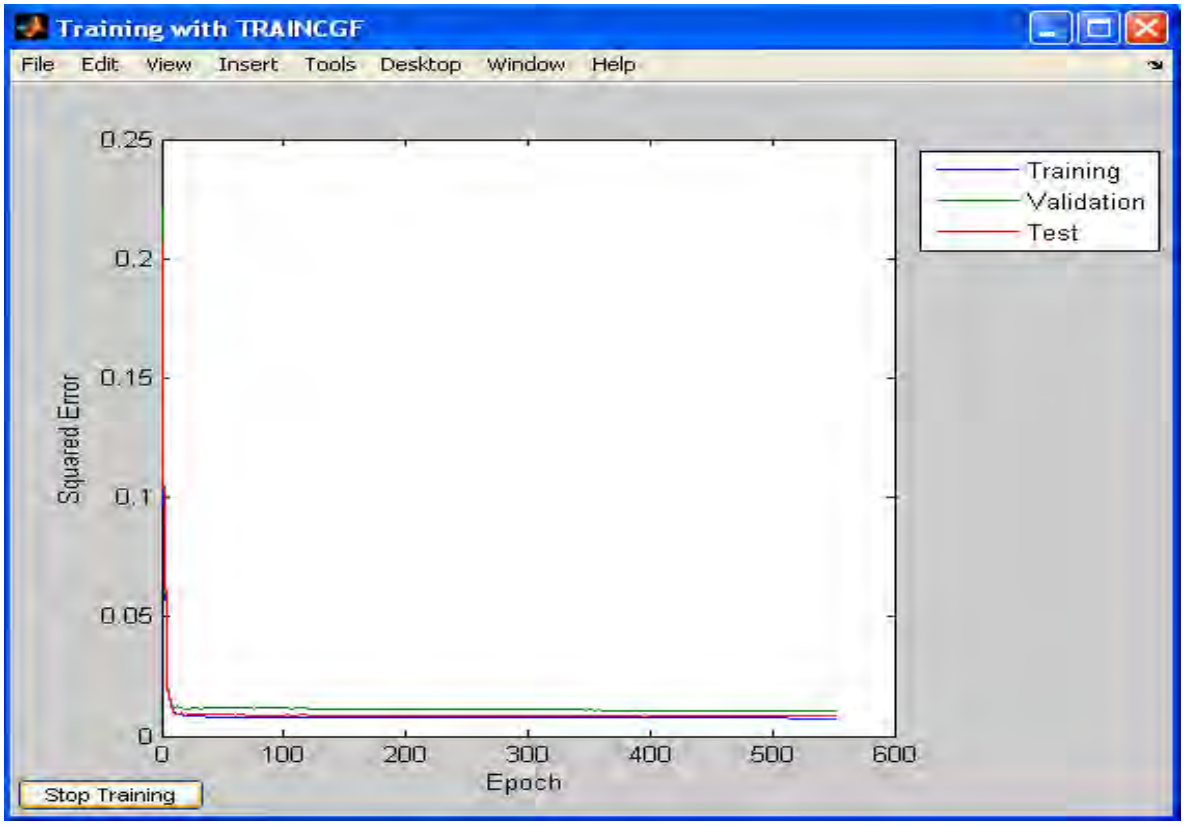
86.7769

correct_test =

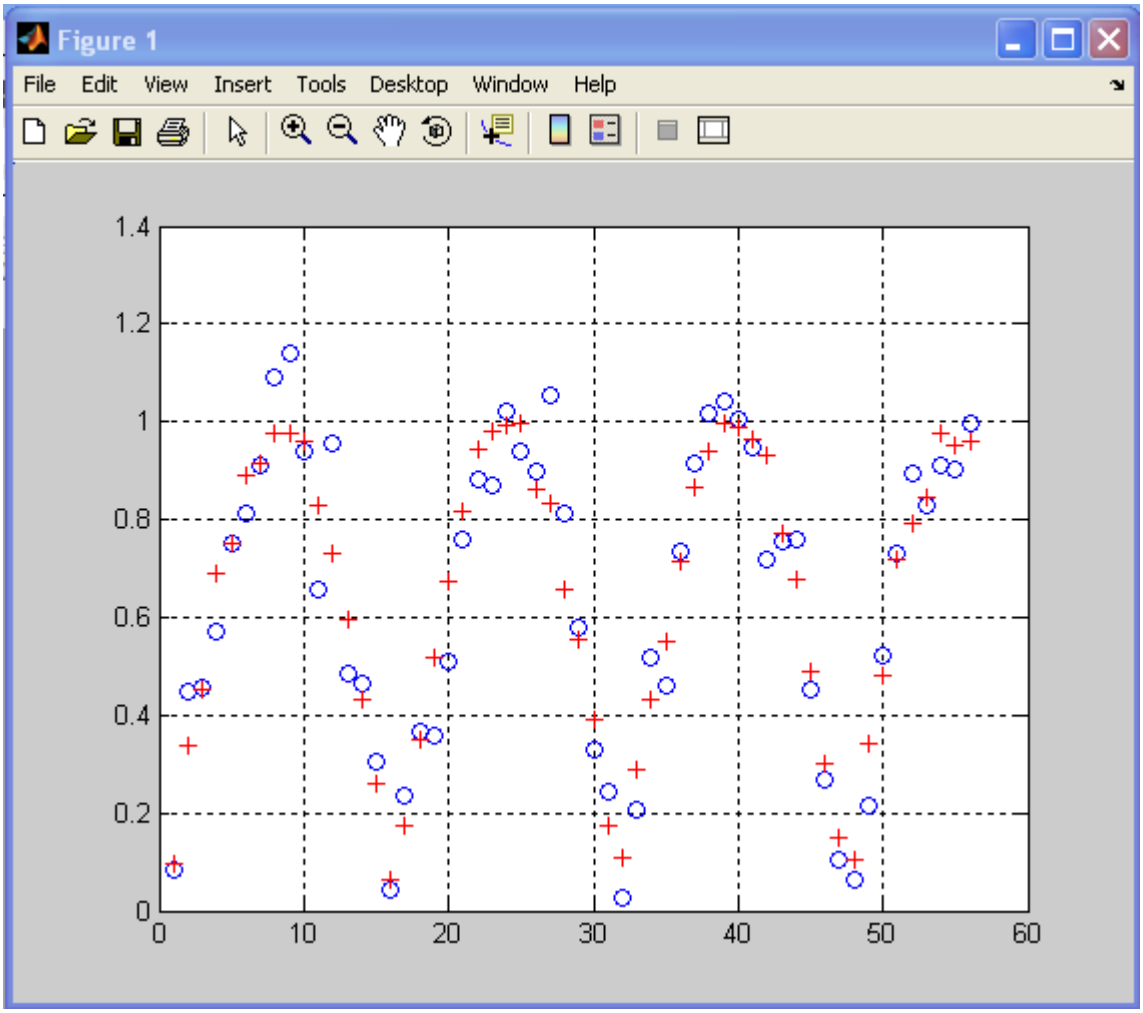
91.1111



Εικόνα 9.12.α Εισαγωγή δεδομένων και τιμών στο πρόγραμμα



Εικόνα 9.12.β Γραφική απεικόνιση των αποτελεσμάτων



Εικόνα 9.12.γ Απεικόνιση των τιμών στόχων (πραγματικές τιμές) και των υπολογιζόμενων τιμών

Αποτελέσματα προγράμματος:

mse_ =

0.0075

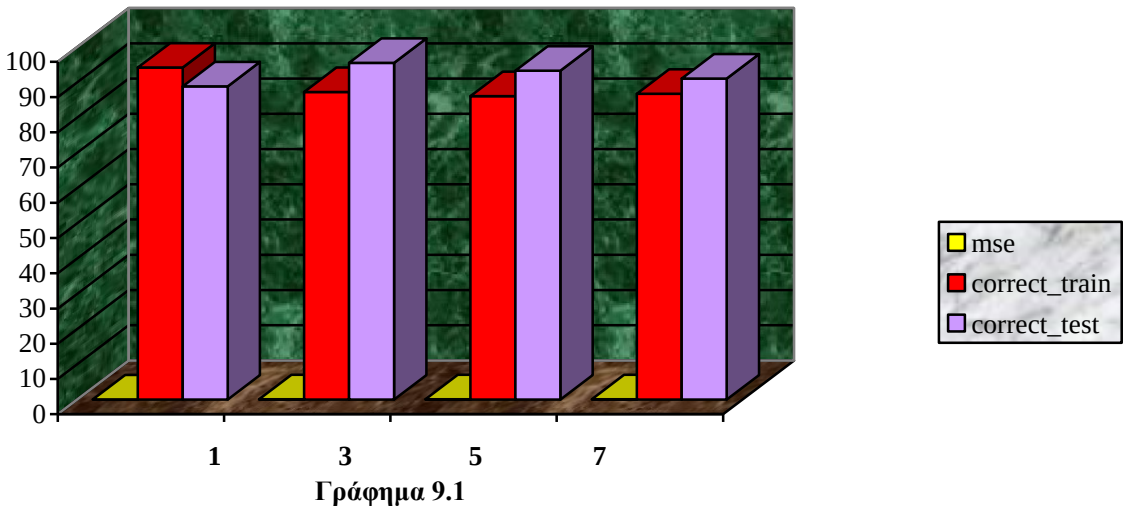
correct_train =

77.6979

correct_test =

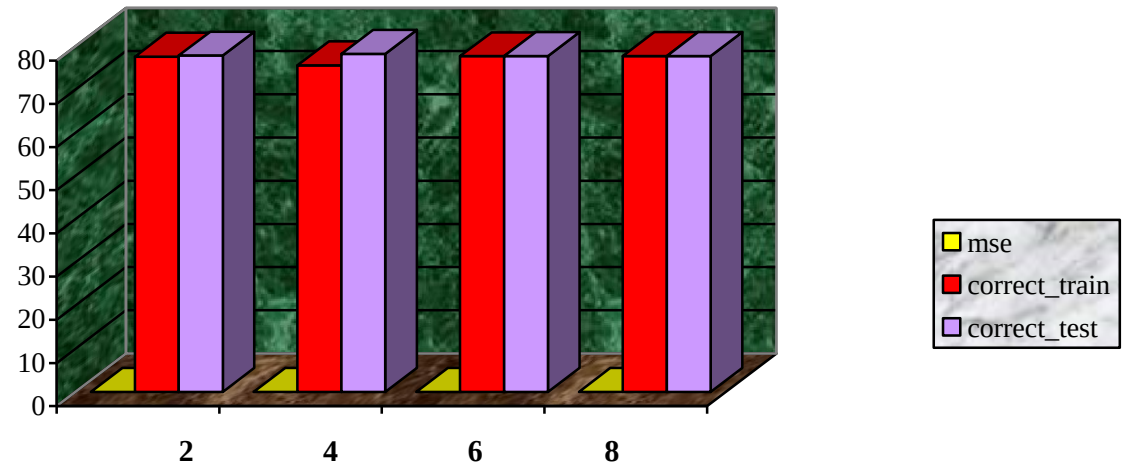
77.7426

ΑΠΕΙΚΟΝΙΣΗ ΤΙΜΩΝ ΓΙΑ ΑΚΕΡΑΙΟ ΤΥΠΟ ΔΕΔΟΜΕΝΩΝ



Γράφημα 9.1

ΑΠΕΙΚΟΝΙΣΗ ΤΙΜΩΝ ΓΙΑ ΠΡΑΓΜΑΤΙΚΟ ΤΥΠΟ ΔΕΔΟΜΕΝΩΝ



Γράφημα 9.2

Συμπεράσματα

Αποφασίσαμε να οπτικοποιήσουμε το πίνακα 9.2 χρησιμοποιώντας δυο γραφήματα: Το γράφημα 9.1 το οποίο παρουσιάζει πως ένα αρχείο ακεραίου τύπου δεδομένων ανταποκρίνεται στις τιμές που δίνουμε στο πρόγραμμα και το γράφημα 9.2 που παρουσιάζει πως ένα αρχείο πραγματικού τύπου δεδομένων ανταποκρίνεται στις τιμές των παραμέτρων του προγράμματος. Σκοπός των ανωτέρω γραφημάτων είναι να έχουμε μια καλύτερη απόδοση σε γραφικό περιβάλλον των τιμών που παρήχθησαν, ώστε να γίνει ο σχολιασμός και πιο εύκολα.

Αρχικά η απόδοση στον υπολογισμό των παραμέτρων της εκπαίδευσης και της προσομοίωσης διαφέρει ανάλογα τόσο με τα αρχεία δεδομένων όσο και με τον τύπο των δεδομένων που κάθε φορά εισαγάγουμε. Επίσης, αναλόγως αισθητή είναι η διαφορά της τιμής του ολικού μέσου τετραγωνικού σφάλματος. Όταν τα δεδομένα μας είναι ακεραίου τύπου, τότε παρατηρούνται πολύ καλά αποτελέσματα στην εκπαίδευση και στην προσομοίωση του δικτύου σε νέα δεδομένα. Αυτό το διαπιστώνουμε είτε από τα αποτελέσματα που εξάγει το πρόγραμμά μας είτε από τα διαγράμματα «απεικόνισης των τιμών στόχων (πραγματικές τιμές) και των υπολογιζόμενων τιμών». Η ελάχιστη όμως τιμή του ολικού μέσου τετραγωνικού σφάλματος είναι μεγάλη συγκριτικά με τη τιμή του ολικού μέσου τετραγωνικού σφάλματος όταν το αρχείο μας είναι πραγματικού τύπου δεδομένων με τιμές που να ανήκουν στο διάστημα [0,1). Συγκεκριμένα η μικρότερη τιμή ολικού μέσου τετραγωνικού σφάλματος που παρατηρήθηκε σε αυτό το πείραμα, και κάνοντας χρήση της συναρτήσεως εκπαίδευσης trainrp για αρχείο με ακέραιο τύπο δεδομένων, ήταν 0.0394. Αντίστοιχως, η μικρότερη τιμή σφάλματος που παρατηρήθηκε για το αρχείο με πραγματικό τύπο δεδομένων που να ανήκουν στο διάστημα [0,1) ήταν 0.0067, χρησιμοποιώντας όμως αυτή τη φορά ως συνάρτηση εκπαίδευσης την trainlm. Δηλαδή η τιμή του ολικού μέσου τετραγωνικού σφάλματος για το αρχείο με τον ακέραιο τύπο δεδομένων είναι περίπου έξι φορές μεγαλύτερη από τη τιμή του ολικού μέσου τετραγωνικού σφάλματος για το αρχείο με τον πραγματικό τύπο δεδομένων.

Ωστόσο, και στο αρχείο με το πραγματικό τύπο δεδομένων παρατηρήθηκε αρνητικό χαρακτηριστικό και το οποίο είναι ότι τα ποσοστά εκπαίδευσης και προσομοίωσης ήταν φτωχά. Αξίζει να σημειωθεί ότι η μέση τιμή για τα ποσοστά εκπαίδευσης είναι 77.1257% ενώ για τα ποσοστά προσομοίωσης 77.8882% εν αντιθέσει με το αρχείο με τον ακέραιο τύπο δεδομένων όπου η μέση τιμή για τα ποσοστά εκπαίδευσης και προσομοίωσης είναι 88.5950% και 92.2222% αντιστοίχως. Παρατηρώντας όμως καλύτερα τα διαγράμματα «απεικόνισης των τιμών στόχων (πραγματικές τιμές) και των υπολογιζόμενων τιμών» για το αρχείο με το πραγματικό τύπο δεδομένων παρατηρούμε την ύπαρξη κάποιων ακραίων τιμών οι οποίες είναι πολύ πιθανόν να διαμορφώνουν τα χαμηλά ποσοστά προσομοίωσης και εκπαίδευσης. Αν λοιπόν ο ερευνητής έχοντας υπόψη του τα ανωτέρω δεδομένα, κατορθώσει να εμποδίσει την εμφάνιση τέτοιων ακραίων τιμών, κανονικοποιώντας για παράδειγμα τις παρατηρήσεις, τότε ίσως το αποτέλεσμα της μέσης τιμής όσον αφορά τη προσομοίωση του δικτύου σε νέα δεδομένα να είναι πιο ενθαρρυντικό κρίνοντας πάντα από το αρχείο ακεραίου τύπου δεδομένων που χρησιμοποιήσαμε. Σε αυτό η μέση τιμή του ποσοστού προσομοίωσης του δικτύου είναι όχι μόνο κατά 15 μονάδες μεγαλύτερη, αλλά και πολύ κοντά στον επιδιωκόμενο στόχο μας. Εν κατακλείδι, ενώ τα αποτελέσματα των ποσοστών εκπαίδευσης και προσομοίωσης για το αρχείο με το πραγματικό τύπο δεδομένων δεν μοιάζουν αρχικώς ικανοποιητικά, εν συνεχεία και εξετάζοντας

προσεκτικότερα τα σχετικά διαγράμματα ‘απεικόνισης των τιμών στόχων (πραγματικές τιμές) και των υπολογιζόμενων τιμών’ και παρατηρώντας ότι οι τιμές της προσομοίωσης ακολουθούν τη καμπύλη των πραγματικών τιμών εύκολα συμπεραίνουμε ότι η εκπαίδευση του τεχνητού νευρωνικού δικτύου κρίνεται επιτυχής είτε το αρχείο είναι ακεραίου τύπου δεδομένων είτε το αρχείο είναι πραγματικού τύπου δεδομένων.

10 Βιβλιογραφία

[1] Βλαχάβας Ιωάννης, Κεφάλας Πέτρος,
Βασιλειάδης Νικόλαος,Κοκκορας Φώτης,
Σακελλαρίου Ηλίας,
‘Τεχνητή Νοημοσύνη’, Γ' Έκδοση, 2006.

[2] Πλαγιανάκος Βασίλειος, ‘Νέες Μέθοδοι Εκπαίδευσης ΤΝΔ, Βελτιστοποίησης και Εφαρμογές’, 2007.

[3] Πάνου Ελένης, ‘Υλοποίηση Αλγορίθμου Πρόβλεψης Ζήτησης με Χρήση Τεχνητών Νευρωνικών Δικτύων’, 2005.

[4] Ηλεκτρονική σελίδα του Ελληνικού Ανοιχτού Πανεπιστημίου (παραδείγματα-ασκήσεις).

[5] Ben Krose and
Patrick van der Smagt,
‘An introduction to Neural Networks Eighth edition’ 1996.

[6] Cowan J.D., ‘Neural networks: The early days’, 1990.

[7] Randall Wilson and
Tony R. Martinez
‘The Inefficiency of Batch Training for Large Training Sets’, 2003.

[8] Gogging, S.D.D., Johnson K.M and Gustafon K., ‘Primacy and recency effects due to momentum in back-propagation learning’, 1990.

[9] Guyon I., ‘Neural Networks and applications’, 1991.

[10] Hammerstrom D., ‘Neural Networks at work’, IEEE Spectrum, 1993.

[11] Hammerstrom D., ‘Working with Neural Networks’, 1992.

[12] Hawkins R.D and G.H Bower, ‘Computational Models’, 1989.

[13] Martin T. Hagan, Howard HB. Demuth Mark Beale,
‘Neural Network Design’, 1997.

[14] Matthias Elter
Fraunhofer Institute for Integrated Circuits (IIS)
Image Processing and Medical Engineering Department (BMT) ,
Prof. Dr. Rudiger Schulz-Wendtland
Institute of Radiology, Gynaecological Radiology, University Erlangen-Nuremberg , data set.

[15] Matlab help.

[16] Lowe D., 'What have neural networks to offer statistical processing?', 1991.

[17] Simon Haykin, McMaster University, Hamilton, Ontario, Canada,
'Neural Networks a Comprehesive Foundation', Second Edition, 1999.

[18] Ηλεκτρονική Εγκυκλοπαίδεια Wikipedia.

[19] Widrow B., 'Generalization and Information storage in networks of Adeline 'neurons' ', 1962.

[20] Williams R.J, 'Feature discovery through error-correction learning', 1985.