ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ

ΠΡΟΓΡΑΜΜΑ ΣΠΟΥΔΩΝ ΕΚΠΑΙΔΕΥΣΗΣ ΑΠΟ ΑΠΟΣΤΑΣΗ

ΠΡΟΓΡΑΜΜΑ ΣΠΟΥΔΩΝ	**:**	*ΠΛΗΡΟΦΟΡΙΚΗ*
ΘΕΜΑΤΙΚΗ ΕΝΟΤΗΤΑ P-INF-003	**:**	*ΝΕΥΡΩΝΙΚΑ ΔΙΚΤΥΑ ΚΑΙ ΓΕΝΕΤΙΚΟΙ ΑΛΓΟΡΙΘΜΟΙ*

ΕΚΠΑΙΔΕΥΤΙΚΟ ΥΛΙΚΟ

ΤΡΙΤΟ ΚΕΦΑΛΑΙΟ

ΣΥΓΓΡΑΦΕΙΣ : ***Σ. ΛΥΚΟΘΑΝΑΣΗΣ***
ΕΠ. ΚΑΘΗΓΗΤΗΣ
ΤΜΗΜΑΤΟΣ ΜΗΧ/ΚΩΝ Η/Υ ΚΑΙ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΠΑΝΕΠΙΣΤΗΜΙΟΥ ΠΑΤΡΩΝ

Ε. ΓΕΩΡΓΟΠΟΥΛΟΣ
ΜΗΧΑΝΙΚΟΣ Η/Υ ΚΑΙ ΠΛΗΡΟΦΟΡΙΚΗΣ

- ΠΑΤΡΑ 1999 -

3. ΑΛΓΟΡΙΘΜΟΙ ΜΑΘΗΣΗΣ

Σκοπός

Στο δεύτερο κεφάλαιο αναφέραμε τα δομικά στοιχεία ενός Τεχνητού Νευρωνικού Δικτύου. Επίσης μελετήσαμε διάφορες συναρτήσεις, οι οποίες χρησιμοποιούνται σαν συναρτήσεις ενεργοποίησης των τεχνητών νευρώνων. Τέλος, παρουσιάσαμε τις γνωστές κατηγορίες και αρχιτεκτονικές των Τ.Ν.Δ., που χρησιμοποιούνται για πρακτικές εφαρμογές.

Σε αυτό το κεφάλαιο θα παρουσιάσουμε τους τρεις βασικούς αλγορίθμους μάθησης (εκπαίδευσης) Ν.Δ.. Θα ξεκινήσουμε την παρουσίαση, με τον αλγόριθμο εκπαίδευσης του απλού Perceptron (Αισθητήρα) και το θεώρημα της σύγκλισής του. Ακολουθεί ο αλγόριθμος Ελάχιστου Μέσου Τετραγωνικού (Ε.Μ.Τ.) λάθους, για την εκπαίδευση ενός απλού Ν.Δ.. Για την απόδειξη του αλγορίθμου, θα δανειστούμε ιδέες από το γραμμικό πρόβλημα φιλτραρίσματος. Θα παρουσιάσουμε πρώτα τις εξισώσεις των Wiener-Hopf και στη συνέχεια τις δύο μεθόδους επίλυσής τους. Αυτές είναι η μέθοδος Ταχύτερης Καθόδου και η μέθοδος του Ελάχιστου Μέσου Τετραγωνικού λάθους. Τέλος, θα παρουσιάσουμε το βασικό αλγόριθμο εκπαίδευσης για δίκτυα εμπρός τροφοδότησης πολλών επιπέδων, που είναι γνωστά σαν Perceptrons πολλών επιπέδων. Ο αλγόριθμος εκπαίδευσης αυτών των δικτύων είναι ο πολύ δημοφιλής αλγόριθμος Πίσω Διάδοσης (Π.Δ.) του λάθους. Αν και η παραγωγή του αλγορίθμου είναι αρκετά πολύπλοκη, ο ίδιος ο αλγόριθμος είναι εύκολο να υλοποιηθεί και έχει τύχει ευρείας εφαρμογής σε πολλά πρακτικά προβλήματα.

Συνοψίζοντας, μπορούμε να πούμε ότι σκοπός αυτού του κεφαλαίου είναι η παρουσίαση των βασικών αλγορίθμων εκπαίδευσης τόσο απλών όσο και πολυεπίπεδων Τ.Ν.Δ.. Έτσι ο αναγνώστης, μελετώντας τα τρία πρώτα κεφάλαια, θα έχει αποκτήσει μια γενική εικόνα για το τι είναι τα Τεχνητά Νευρωνικά Δίκτυα, που αποτελούν ένα σημαντικό τμήμα της Υπολογιστικής Νοημοσύνης και πως εκπαιδεύονται.

Προσδοκώμενα Αποτελέσματα:

Όταν θα έχετε τελειώσει τη μελέτη αυτού του κεφαλαίου, θα μπορείτε να:

- υλοποιήσετε τον αλγόριθμο εκπαίδευσης του απλού Perceptron,
- υλοποιήσετε τον αλγόριθμο εκπαίδευσης Ε.Μ.Τ. λάθους,
- υλοποιήσετε τον αλγόριθμο εκπαίδευσης Π.Δ. του λάθους, για Perceptrons πολλών επιπέδων,
- εξηγήσετε τη λειτουργία απλών και πολυεπίπεδων Τ.Ν.Δ..

Έννοιες Κλειδιά:

- αλγόριθμοι εκπαίδευσης
- κανόνας διόρθωσης του λάθους
- εκπαιδευτικό σύνολο
- γραμμικά διαχωριζόμενα πρότυπα
- αλγόριθμος του Perceptron
- θεώρημα σύγκλισης
- συνάρτηση κόστους
- αλγόριθμος Ε.Μ.Τ. λάθους
- μέθοδος ταχύτερης καθόδου
- αλγόριθμος Π.Δ. του λάθους
- λειτουργικά σήματα
- σήματα λάθους
- αλυσιδωτός κανόνας παραγώγισης

Εισαγωγικές Παρατηρήσεις:

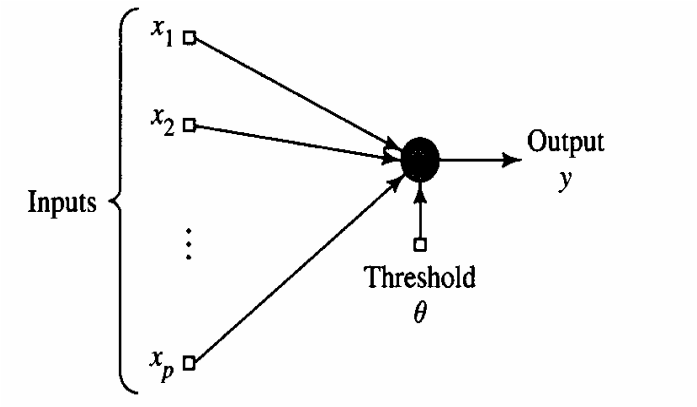
Στα δύο προηγούμενα κεφάλαια αναφέραμε ότι τα Ν.Δ. εκπαιδεύονται με τη βοήθεια παραδειγμάτων, έτσι ώστε να μαθαίνουν το περιβάλλον τους. Ένα παράδειγμα περιλαμβάνει την είσοδο και την επιθυμητή έξοδο σε αυτή. Το σύνολο των παραδειγμάτων αποτελεί το εκπαιδευτικό σύνολο. Για την εκπαίδευση χρησιμοποιούνται κανόνες, οι οποίοι βασίζονται στην ελαχιστοποίηση του λάθους στην έξοδο του δικτύου. Ακολουθεί η γενίκευση, δηλαδή τα Ν.Δ. μαθαίνουν παραδείγματα για τα οποία δεν έχουν εκπαιδευτεί. Όπως υπάρχουν πολλές κατηγορίες Ν.Δ., ανάλογα με την αρχιτεκτονική τους και τον τρόπο εκπαίδευσής τους έτσι

υπάρχει μεγάλη ποικιλία αλγορίθμων εκπαίδευσης, ανάλογα με τον κανόνα μάθησης και τον αλγόριθμο ελαχιστοποίησης που χρησιμοποιείται. Στις επόμενες ενότητες, θα παρουσιάσουμε δύο βασικές κατηγορίες εκπαίδευσης των Τ.Ν.Δ.. Πρώτα θα παρουσιάσουμε τους δύο αλγορίθμους εκπαίδευσης απλών Ν.Δ., ενός επιπέδου. Αυτά τα δίκτυα είναι κατάλληλα για την ταξινόμηση προτύπων, που είναι γραμμικά διαχωριζόμενα. Στη συνέχεια θα ασχοληθούμε με δίκτυα πολλών επιπέδων που είναι γνωστά και σαν Perceptrons πολλών επιπέδων. Αυτά τα δίκτυα εκπαιδεύονται με τον αλγόριθμο Πίσω Διάδοσης του λάθους και είναι κατάλληλα για την ταξινόμηση προτύπων που δεν είναι γραμμικά διαχωριζόμενα. Αυτός είναι ο λόγος που αυτή η κατηγορία Τ.Ν.Δ. έχει χρησιμοποιηθεί για την επίλυση μιας μεγάλης ποικιλίας πρακτικών προβλημάτων.

3.1 Ο αλγόριθμος μάθησης του Perceptron (Αισθητήρα)

Το Perceptron είναι η απλούστερη μορφή Νευρωνικού δικτύου, το οποίο χρησιμοποιείται για την ταξινόμηση ενός ειδικού τύπου προτύπων, που είναι γραμμικά διαχωριζόμενα (δηλαδή πρότυπα που βρίσκονται στις αντίθετες πλευρές ενός υπερεπιπέδου, το οποίο ορίζει τις περιοχές απόφασης).

Ένα τέτοιο δίκτυο φαίνεται στο παρακάτω σχήμα 1:

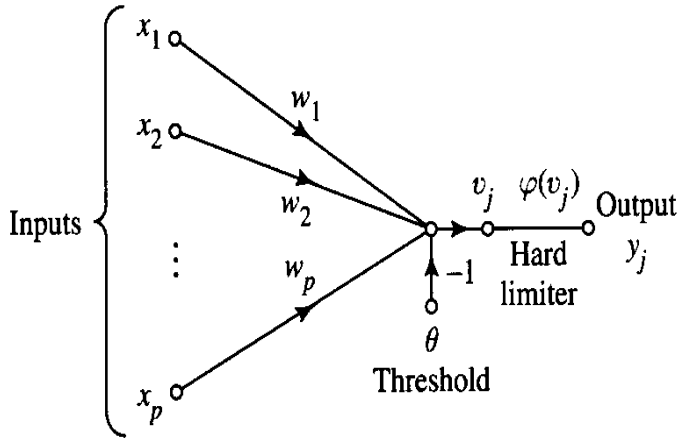


Σχήμα 1: Perceptron ενός επιπέδου

Προκειμένου να εκπαιδευτεί ένα τέτοιο Ν.Δ., σαν αλγόριθμος εκπαίδευσης χρησιμοποιείται ο γνωστός κανόνας του Rosenblatt[1]. Αυτός ο κανόνας εφαρμόζεται στο γνωστό μοντέλο Mc Culloch – Pitts, για το νευρώνα. Όπως είδαμε στο δεύτερο

κεφάλαιο αποτελείται από ένα γραμμικό συνδυαστή ακολουθούμενο από ένα στοιχείο κατωφλίου και η παραγόμενη έξοδος παίρνει με τιμές ± 1.

Θεωρούμε το διάγραμμα ροής σήματος του Perceptron, που φαίνεται στο σχήμα 2.



Σχήμα 2: Το διάγραμμα ροής σήματος του Perceptron.

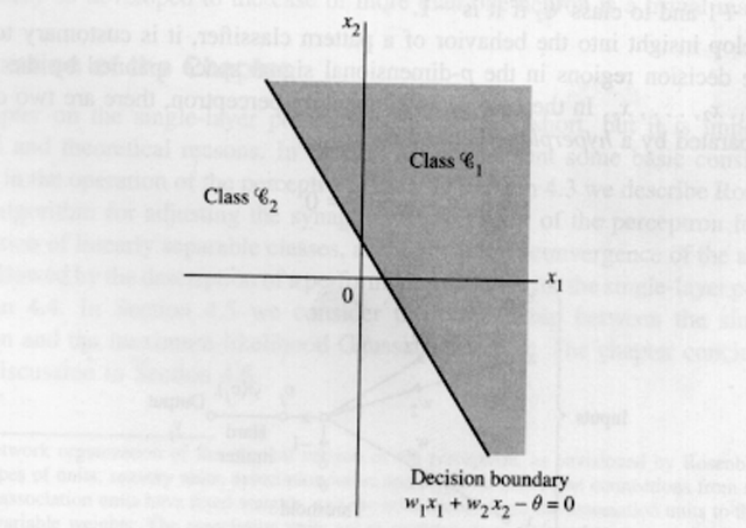
Η έξοδος του γραμμικού συνδυαστή υπολογίζεται εύκολα από το διάγραμμα του σχήματος 2 και είναι:

$$\upsilon = \sum_{i=1}^{p} w_i x_i - \theta \qquad (1)$$

Σκοπός του Perceptron είναι να ταξινομήσει ένα σύνολο εισόδων (προτύπων) x_1, x_2, … x_ρ σε μία από τις κλάσεις l_1 και l_2. Ο κανόνας απόφασης για την ταξινόμηση είναι: ανάθεσε το σημείο που αναπαριστούν x_1, x_2, … x_ρ τα στην κλάση l_1, αν y = +1 και στην κλάση l_2 αν y = -1. Οι περιοχές απόφασης διαχωρίζονται από το υπερεπίπεδο που ορίζεται από τη σχέση:

$$\upsilon = \sum_{i=1}^{p} w_i x_i - \theta = 0 \quad \Leftrightarrow \quad w_1 x_1 + w_2 x_2 - \theta = 0 \qquad (2)$$

Στο σχήμα 3 φαίνεται η γραμμική διαχωρισιμότητα για ένα δισδιάστατο πρόβλημα ταξινόμησης, με δύο κλάσεις.



Σχήμα 3: Το όριο και οι περιοχές απόφασης για ένα δισδιάστατο πρόβλημα ταξινόμησης δύο κλάσεων.

Από το παραπάνω σχήμα φαίνεται το αποτέλεσμα της εφαρμογής του κατωφλίου, το οποίο μετατοπίζει το όριο απόφασης από την αρχή των αξόνων. Τα συναπτικά βάρη του Perceptron, μπορούν να προσαρμοσθούν επαναληπτικά. Για την προσαρμογή του διανύσματος βαρών w, χρησιμοποιούμε έναν κανόνα διόρθωσης λάθους, που είναι γνωστός σαν κανόνας σύγκλισης του Perceptron και αναπτύσσεται στην επόμενη υποενότητα.

3.1.1 Το θεώρημα σύγκλισης του Perceptron

Για την παραγωγή του αλγορίθμου μάθησης διόρθωσης λάθους, για ένα απλό Perceptron ενός επιπέδου, θα εργαστούμε με το μοντέλο ροής σήματος του σχήματος 4. Θεωρούμε το κατώφλι θ(n) σαν ένα συναπτικό βάρος, που είναι συνδεδεμένο σε μια σταθερή είσοδο -1. ΄Αρα, το (p + 1) x 1 διάνυσμα εισόδου είναι:

$$\mathbf{x}(n) = [\, -1, x_1(n), x_2(n), \ldots, x_p(n) \,]^T \qquad (3)$$

και αντίστοιχα ορίζουμε το (p + 1) x 1 διάνυσμα βαρών:

$$\mathbf{w}(n) = [\, \theta(n), w_1(n), w_2(n), \ldots, w_p(n) \,]^T \qquad (4)$$

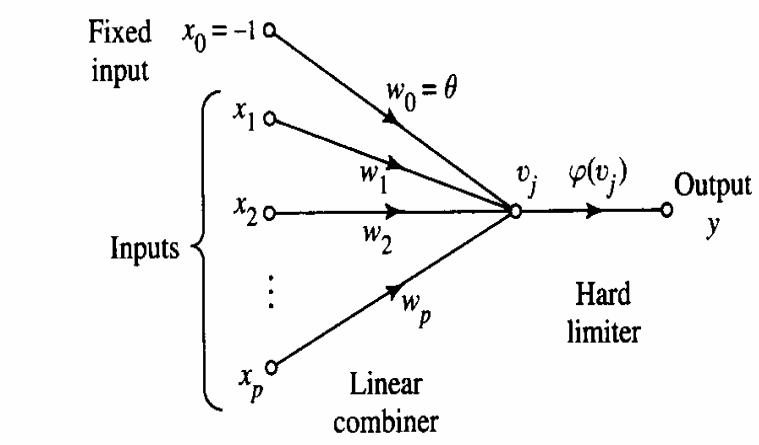
Η έξοδος του γραμμικού συνδυαστή είναι:

$$\upsilon(n) = \mathbf{w}^T(n)\, \mathbf{x}(n) \qquad (5)$$

Αν οι κλάσεις l_1 και l_2 είναι γραμμικά διαχωριζόμενες, τότε υπάρχει ένα διάνυσμα βαρών, για το οποίο μπορούμε να ορίσουμε ότι:

και
$$\left.\begin{array}{ll} \mathbf{w}^T\mathbf{x} \geq 0 & \forall\, x \in l_1 \\ \mathbf{w}^T\mathbf{x} < 0 & \forall\, x \in l_2 \end{array}\right\} \qquad (6)$$

Το πρόβλημα για το απλό Perceptron είναι να βρούμε το διάνυσμα βαρών w, το οποίο ικανοποιεί τις ανισότητες (6).



Σχήμα 4: Ισοδύναμο διάγραμμα ροής σήματος του Perceptron.

Ο αλγόριθμος προσαρμογής των βαρών μπορεί τώρα να διατυπωθεί ως εξής.

1. Αν το n - στό μέλος του εκπαιδευτικού διανύσματος **x**(n), ταξινομείται σωστά από το διάνυσμα βαρών στην η-στή επανάληψη του αλγορίθμου, δεν γίνεται καμία διόρθωση στο w(n), δηλαδή:

$\mathbf{w}(n+1) = \mathbf{w}(n)$ αν $\mathbf{w}^T(n)\,\mathbf{x}(n) \geq 0$ και $\mathbf{x}(n) \in l_1$

και $\mathbf{w}(n+1) = \mathbf{w}(n)$ αν $\mathbf{w}^T(n)\,\mathbf{x}(n) < 0$ και $\mathbf{x}(n) \in l_2$ (7)

2. Διαφορετικά, το διάνυσμα βαρών του Perceptron, ενημερώνεται σύμφωνα με τον κανόνα:

$\mathbf{w}(n+1) = \mathbf{w}(n) - \eta(n)\,\mathbf{x}(n)$ αν $\mathbf{w}^T(n)\,\mathbf{x}(n) \geq 0$ και $\mathbf{x}(n) \in l_2$

και $\mathbf{w}(n+1) = \mathbf{w}(n) + \eta(n)\,\mathbf{x}(n)$ αν $\mathbf{w}^T(n) < 0$ και $\mathbf{x}(n) \in l_1$ (8)

όπου η παράμετρος ρυθμού - μάθησης η(n) ελέγχει τις ρυθμίσεις, που εφαρμόζονται στο διάνυσμα βαρών στην επανάληψη η.

Αν $\eta(n) = \eta - ct > 0$, τότε έχουμε ένα κανόνα σταθερά αυξανόμενης προσαρμογής

(fixed increment adaptation rule) για το Perceptron. Για τη μελέτη της σύγκλισης αυτού του αλγορίθμου, ο αναγνώστης παραπέμπεται στην αναφορά [1, κεφάλαιο4]. Εκεί αποδεικνύεται ότι ο κανόνας εκπαίδευσης του απλού Perceptron, για γραμμικά διαχωριζόμενα πρότυπα, συγκλίνει σε πεπερασμένο αριθμό επαναλήψεων.

3.1.2 Ανακεφαλαίωση

Στον πίνακα 1, παρουσιάζεται η ανακεφαλαίωση του αλγόριθμου σύγκλισης του Perceptron [1].

Το σύμβολο sgn(·), που χρησιμοποιείται στο βήμα 3 του πίνακα, για τον υπολογισμό της πραγματικής απόκρισης του Perceptron, παριστάνει την συνάρτηση προσήμου:

$$\text{sgn}(\upsilon) = \begin{cases} +1 & \text{αν} \quad \upsilon > 0 \\ -1 & \text{αν} \quad \upsilon < 0 \end{cases}$$

ΠΙΝΑΚΑΣ 1: Αλγόριθμος Σύγκλισης του Perceptron

Μεταβλητές και Παράμετροι

$\mathbf{x}(n)$ = (p + 1) x 1 input vector

$[\ -1, x_1(n), x_2(n), \ldots x_p(n)\ ]^T$

$\mathbf{w}(n)$ = (p + 1) x 1 weight vector

$= [\ \theta(n), \mathbf{w}_1(n), \mathbf{w}_2(n), \ldots, \mathbf{w}_p(n)\ ]^T$

$\boldsymbol{\theta}(n)$ = threshold (κατώφλι)

$\mathbf{y}(n)$ = actual response (πραγματική έξοδος)

$\mathbf{d}(n)$ = desired response (επιθυμητή έξοδος)

$\boldsymbol{\eta}$ = learning - rate parameter, θετική σταθερά < 1

Step 1: ***Αρχικοποίηση***

Θέσε $\mathbf{w}(0) = 0$. Κατόπιν κάνε τους υπολογισμούς για η = 1, 2, …

Step 2: Ενεργοποίηση

Στο χρόνο n, ενεργοποίησε το Perceptron εφαρμόζονται το συνεχές διάνυσμα εισόδου x(n) και το $\mathbf{d}(n)$.

Step 3: ***Υπολογισμός πραγματικής απόκρισης***

Υπολόγισε την πραγματική απόκριση του Perceptron:

$\mathbf{y}(n) = \text{sgn}[\mathbf{w}^T(n)\ \mathbf{x}(n)\ ]$

Step 4: ***Προσαρμογή διανύσματος βαρών***

Προσάρμοσε τα βάρη του Perceptron:

$\mathbf{w}(n+1) = \mathbf{w}(n) + \boldsymbol{n}\,[\mathbf{d}(n) - \mathbf{y}(n)]\,\mathbf{x}(n)$ (E.C.L. rule)

όπου:

$$\mathbf{d}(n) = \begin{cases} +1, & \text{αν } \mathbf{x}(n) \text{ ανήκει στην κλάση } l_1 \\ -1, & \text{αν } \mathbf{x}(n) \text{ ανήκει στην κλάση } l_2 \end{cases}$$

Step 5: ***Αύξησε το χρόνο η κατά μια μονάδα και πήγαινε στο βήμα 2.***

Άσκηση αυτοαξιολόγησης 3.1/1:

1. Γραμμικά διαχωριζόμενα είναι τα πρότυπα
2. για τα οποία μπορούμε εύκολα να ορίσουμε περιοχές απόφασης
3. για τα οποία οι περιοχές απόφασης είναι γραμμικές
4. που βρίσκονται στις αντίθετες πλευρές ενός υπερεπιπέδου που ορίζει τις περιοχές απόφασης
5. τα οποία χρησιμοποιούνται για την εκπαίδευση του Perceptron.

Απάντηση: Σύμφωνα με τον ορισμό που δίνεται στην ενότητα 3.1, η σωστή απάντηση είναι η 3.

Άσκηση αυτοαξιολόγησης 3.1/2:

Να υποθέσετε ότι στο διάγραμμα ροής σήματος του Pereptron του σχήματος 4, η συνάρτηση ενεργοποίησης έχει τη μορφή:

$$\varphi(\upsilon) = \tanh(\frac{\upsilon}{2})$$

Όπου υ είναι η γραμμική έξοδος του νευρώνα. Οι αποφάσεις ταξινόμησης από το Perceptron, καθορίζονται από τον ακόλουθο κανόνα:

Το διάνυσμα παρατήρησης x ανήκει στην κλάση l_1 αν y>θ.

Διαφορετικά το x ανήκει στην κλάση l_2.

Απάντηση: Το σήμα εξόδου υπολογίζεται από τη σχέση

$$y = \tanh(\frac{\upsilon}{2}) = \tanh(\frac{\theta}{2} + \frac{1}{2}\sum_i w_i x_i)$$

Ισοδύναμα, μπορούμε να γράψουμε:

$$\theta + \sum_i w_i x_i = y' \qquad (1)$$

Όπου :

$$y' = 2\tanh^{-1}(y)$$

Η εξίσωση (1) είναι η εξίσωση ενός υπερεπιπέδου.

Άσκηση αυτοαξιολόγησης 3.1/3:

(α) Το Perceptron μπορεί να χρησιμοποιηθεί για να μάθει διάφορες λογικές συναρτήσεις. Να δείξετε την υλοποίηση των δυαδικών λογικών συναρτήσεων AND, OR και COMPLEMENT.

(β) Ένα βασικό μειονέκτημα του Perceptron είναι ότι δεν μπορεί να υλοποιήσει την συνάρτηση EXCLUSIVE OR. Να εξηγήσετε το λόγο για αυτόν τον περιορισμό.

Απάντηση:

(α) Να κατασκευάσετε τον πίνακα αλήθειας για κάθε μια συνάρτηση. Στη συνέχεια να υλοποιήσετε ένα Perceptron δύο εισόδων με μοναδιαία βάρη και κατώφλια –1.5, -0.5 και +0.5 αντίστοιχα. Να επαληθεύσετε ότι ικανοποιούνται οι αντίστοιχοι πίνακες.

(β) Από τον πίνακα αλήθειας της συναρτησης, προκύπτει εύκολα ότι δεν μπορούμε να κατασκευάσουμε ένα γραμμικό όριο απόφασης (όπως στο σχήμα 3).

3.2 Ο αλγόριθμος Ελάχιστου Μέσου Τετραγωνικού (ΕΜΤ) λάθους

Σε αυτό το κεφάλαιο θα ασχοληθούμε με μία ″πρωτόγονη″ κατηγορία νευρωνικών δικτύων που αποτελούνται από ένα απλό νευρώνα και λειτουργούν κάτω από την υπόθεση της γραμμικότητας. Αυτή η κατηγορία νευρωνικών δικτύων είναι σπουδαία για τρεις λόγους:

α. Αναπτύσσεται καλά η θεωρία των γραμμικών προσαρμοζόμενων φίλτρων που χρησιμοποιούν το μοντέλο ενός απλού γραμμικού νευρώνα, με πάρα πολλές εφαρμογές, όπως ο αυτόματος έλεγχος, τα ραντάρ, τα σόναρ, κ.λ.π..

β. Είναι ένα προϊόν της πρωτοποριακής δουλειάς που έγινε στα νευρωνικά δίκτυα τη δεκαετία του 1960.

γ. Μια μελέτη των γραμμικών προσαρμοζόμενων φίλτρων ανοίγει το δρόμο για τη θεωρητική ανάπτυξη της πιο γενικής περίπτωσης των perceptrons πολλών-επιπέδων, που περιλαμβάνει τη χρήση μη-γραμμικών στοιχείων.

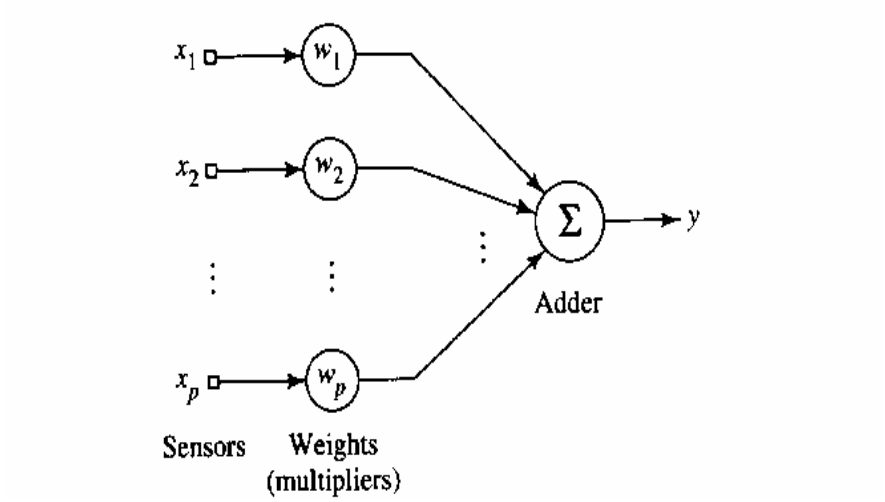
Θα αρχίσουμε τη μελέτη μας με μία σύντομη αναφορά στο πρόβλημα του βέλτιστου γραμμικού φιλτραρίσματος. Στη συνέχεια διατυπώνεται ο αλγόριθμος Ελαχίστων Μέσων Τετραγώνων (Least Mean Square – LMS), που είναι επίσης γνωστός σαν Delta-rule ή σαν ο κανόνας των Widrow και Hoff (1960).

Ο αλγόριθμος LMS λειτουργεί με το μοντέλο ενός απλού γραμμικού νευρώνα, και

έχει βρεί πολλές εφαρμογές. Πράγματι, ο LMS αλγόριθμος καθιερώθηκε σαν ένα σπουδαίο λειτουργικό κομμάτι στην συνεχώς επεκτεινόμενη περιοχή της προσαρμοζόμενης επεξεργασίας σημάτων [1].

3.2.1 Οι εξισώσεις των Wiener-Hopf

Θεωρείστε ένα σύνολο από p αισθητήρες, τοποθετημένους σε διαφορετικά σημεία στο χώρο, όπως φαίνεται στο σχήμα 5. Έστω $x_1,x_2, \ldots ,x_p$, τα σήματα που παράγονται από αυτούς τους αισθητήρες. Αυτά τα σήματα εφαρμόζονται σε ένα αντίστοιχο σύνολο βαρών $w_1,w_2, \ldots ,w_p$. Τα ζυγισμένα σήματα προστίθενται τότε , για να παράγουν την έξοδο y. Αν d, είναι η επιθυμητή έξοδος το ζητούμενο είναι να υπολογίσουμε τη βέλτιστη τιμή του w , έτσι ώστε να ελαχιστοποιεί το λάθος e=d-y. Η λύση σε αυτό το πρόβλημα βρίσκεται στις εξισώσεις των Wiener-Hopf .



Σχήμα 5: Χωρικό φίλτρο (Spatial filter)

Η σχέση εισόδου-εξόδου του παραπάνω φίλτρου είναι :

$$y = \sum_{k=1}^{p} w_k x_k \qquad (9)$$

και το σήμα λάθους : $e = d-y$ (10)

Ένα μέτρο επίδοσης ή συνάρτηση κόστους , είναι το μέσο τετραγωνικό λάθος (mean-squared error), που ορίζεται από τη σχέση:

$$J = ½ E[e^2] \qquad (11)$$

Μπορούμε τώρα να ορίσουμε το γραμμικό πρόβλημα φιλτραρίσματος ως εξής :

Ζητείται να καθοριστεί το βέλτιστο σύνολο βαρών $w_{o1}, w_{o2}, \ldots, w_{op}$, για το οποίο το μέσο τετραγωνικό λάθος J είναι ελάχιστο.

<u>Λύση</u> : Είναι γνωστή σαν φίλτρο Wiener, το οποίο παρουσιάζεται στη συνέχεια.

Αντικαθιστώντας τις εξισώσεις (9) και (10) στην(11) έχουμε:

$$J = ½ E[d^2] - E\left[\sum_{k=1}^{p} w_k x_k d\right] + ½ \left[\sum_{j=1}^{p} \sum_{k=1}^{p} w_j w_k x_j x_k\right] \qquad (12)$$

όπου το διπλό άθροισμα χρησιμοποιείται για να αναπαραστήσει το τετράγωνο του αθροίσματος. Επειδή ο τελεστής Ε είναι γραμμικός , μπορούμε να αλλάξουμε τη σειρά με το Σ, άρα έχουμε:

$$J = ½ E[d^2] - \sum_{k=1}^{p} w_k E[x_k d] + ½ \sum_{j=1}^{p} \sum_{k=1}^{p} w_j w_k E[x_j x_k] \qquad (13)$$

όπου τα w θεωρούνται σταθερές, άρα βγαίνουν έξω από το E[.] .

Ορισμοί:

1. Η αναμενόμενη τιμή $E[d^2]$ είναι η μέση τετραγωνική τιμή του d, άρα :
$$r_d = E[d^2] \qquad (14)$$
2. Η $E[dx_k]$ είναι η συνάρτηση ετεροσυσχέτισης (cross-correlation) μεταξύ του d και του x_k . Έστω
$$r_{dx}(k) = E[dx_k] \ , k=1,2, \ldots p \qquad (15)$$
3. Η $E[x_j x_k]$ είναι η συνάρτηση αυτοσυσχέτισης (autocorrelation) του συνόλου των σημάτων εισόδου . Έστω
$$r_x(j,k) = E[x_j, x_k] \qquad j,k=1,2 \ldots p \qquad (16)$$

Με βάση τους παραπάνω ορισμούς μπορούμε να απλοποιήσουμε την εξίσωση (13) ως εξής :

$$J = \tfrac{1}{2} r_d - \sum_{k=1}^{p} w_k r_{dx}(k) + \tfrac{1}{2} \sum_{j=1}^{p} \sum_{k=1}^{p} w_j w_k r_x(j,k) \qquad (17)$$

Μια σχεδίαση πολλών διαστάσεων της συνάρτησης κόστους J , ως προς τα βάρη $w_1, w_2, \ldots, w_p$, αποτελεί την επιφάνεια απόδοσης λάθους , ή απλώς την επιφάνεια λάθους του φίλτρου. Έχει κοίλο σχήμα , με πολύ καλά καθορισμένο πυθμένα, ή σημεία ολικού ελάχιστου. Αυτό το σημείο , είναι ακριβώς το βέλτιστο για το φίλτρο , με την έννοια ότι το μέσο τετραγωνικό λάθος παίρνει την ελάχιστη τιμή του J_{min} .

Για τον προσδιορισμό του βέλτιστου διαφορίζουμε τη συνάρτηση κόστους J ως προς w_k και μηδενίζουμε το αποτέλεσμα για κάθε k . Η μερική παράγωγος του J ως προς w_k , είναι η κλίση (gradient) της επιφάνειας λάθους ως προς το συγκεκριμένο w_k . Άρα :

$$\nabla_{wk} J = \frac{dJ}{dw_k} \quad , \quad \text{για} \quad k=1,2, \ldots, p$$

(18)

Παραγωγίζοντας την εξίσωση (17) ως προς w_k ,έχουμε :

$$\nabla_{wk} J = -r_{dx}(k) + \sum_{j=1}^{p} w_j r_x(j,k)$$

(19)

Άρα η βέλτιστη συνθήκη για το φίλτρο , ορίζεται από την εξίσωση:

$$\nabla_{wk} J = 0 \quad , \quad k=1,2, \ldots, p$$

(20)

Έστω ότι το w_{ok} , δηλώνει τη βέλτιστη τιμή του w_k . Τότε από την εξίσωση (19), βρίσκουμε ότι οι βέλτιστες τιμές των βαρών καθορίζονται από το ακόλουθο σύνολο εξισώσεων :

$$\sum_{j=1}^{p} w_{oj} r_x(j,k) = r_{xd}(k) \quad , \quad k=1,2, \ldots, p \qquad (21)$$

Αυτό το σύνολο εξισώσεων είναι γνωστό σαν εξισώσεις των Wiener-Hopf και το φίλτρο του οποίου τα βάρη ικανοποιούν τις εξισώσεις Wiener-Hopf καλείται φίλτρο

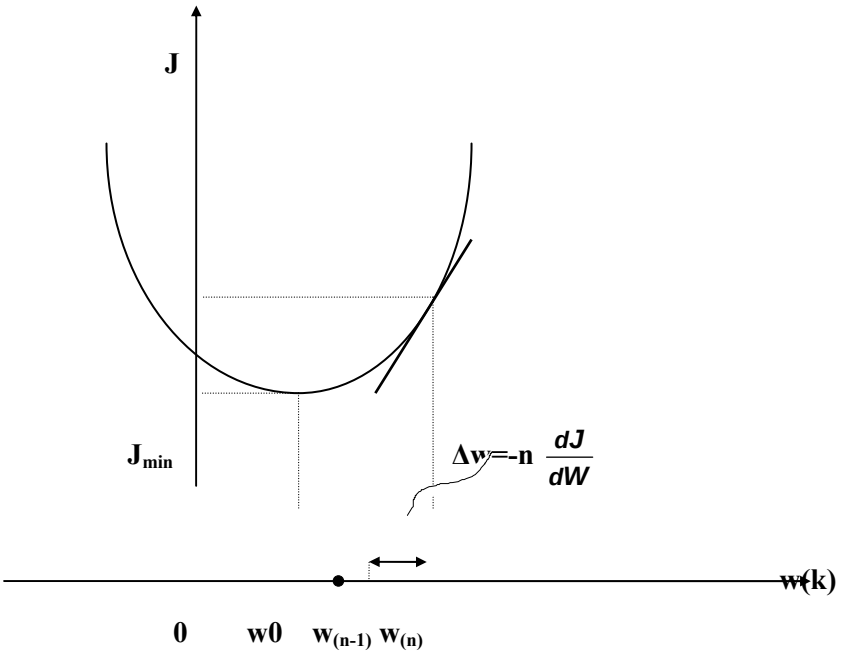
Wiener.

3.2.2 Η μέθοδος Ταχύτερης Καθόδου (Steepest Descent)

Για να λύσουμε τις εξισώσεις Wiener-Hopf , πρέπει να υπολογίσουμε τον αντίστροφο ενός (pxp) πίνακα, τον $r_x(j,k)$, για j, k=1, 2, ... , p. Μπορούμε να αποφύγουμε την αντιστροφή, αν χρησιμοποιήσουμε τη μέθοδο steepest descent. Σύμφωνα με αυτή τη μέθοδο, υποθέτουμε ότι τα βάρη του φίλτρου είναι χρονικά μεταβαλλόμενα και ότι οι τιμές τους διορθώνονται με ένα επαναληπτικό τρόπο κατά μήκος της επιφάνειας λάθους, μετακινώντας τα προοδευτικά προς τη βέλτιστη λύση. Η μέθοδος ταχύτερης καθόδου έχει σαν στόχο τη συνεχή αναζήτηση βέλτιστης λύσης .

Όπως φαίνεται και στο σχήμα 6, οι διορθώσεις για να είναι επιτυχείς πρέπει να γίνονται σε κατεύθυνση αντίθετη προς το διάνυσμα κλίσης, του οποίου τα στοιχεία καθορίζονται από τη σχέση :

$$\nabla_{wk} J \ , \ \text{για} \ \ k=1,2, ... ,p$$



Σχήμα 6: Το κριτήριο MSE για την προσαρμογή ενός βάρους w.

Έστω $w_k(n)$, η τιμή του βάρους w_k , που υπολογίζεται τη χρονική στιγμή n , με τη μέθοδο ταχύτερης καθόδου. Αντίστοιχα, η κλίση της επιφάνειας λάθους, ως προς τα βάρη , παίρνει τη χρονικά μεταβαλλόμενη μορφή :

$$\nabla_{wk}J(n) = - r_{dx}(k) + \sum_{j=1}^{p} w_j(n)r_x(j,k)$$

(22)

δηλαδή, οι δείκτες k, j αναφέρονται σε θέσεις των διαφορετικών αισθητήρων στο χώρο, ενώ ο δείκτης n , αναφέρεται σε αριθμό επανάληψης. Σύμφωνα με τη μέθοδο ταχύτερης καθόδου, η διόρθωση που εφαρμόζεται στο βάρος $w_k(n)$, στην επανάληψη n, δίνεται από τη σχέση:

$$\Delta w_k(n) = - n\nabla_{wk}J(n) \quad , \quad k=1,2, ..., p$$

(23)

όπου n είναι μια θετική σταθερά που ονομάζεται παράμετρος μάθησης (learning-rate). Δοσμένης της παλιάς τιμής του k-τάξεως στοιχείου $w_k(n)$, στην επανάληψη n, η ενημερωμένη τιμή του βάρους την επόμενη χρονική στιγμή n+1, υπολογίζεται από τη σχέση :

$$w_k(n+1) = w_k(n) + \Delta w_k(n) = w_k(n) - n \nabla_{wk}J(n) \quad , \quad k=1,2, ... p$$

(24)

Άρα , μπορούμε να ορίσουμε τη μέθοδο ταχύτερης καθόδου ως εξής :

Η ενημερωμένη τιμή του k-οστού βάρους ενός φίλτρου Wiener (που έχει σχεδιαστεί βάσει του MSE), ισούται με την παλιά τιμή του βάρους συν μια διόρθωση , η οποία είναι ανάλογη της αρνητικής κλίσης της επιφάνειας λάθους , ως προς αυτό το συγκεκριμένο βάρος .

Αντικαθιστώντας την εξίσωση (22) στην (24) , μπορούμε να τυποποιήσουμε την μέθοδο ταχύτερης καθόδου, σαν συνάρτηση των $r_x(j,k)$, $r_{dx}(k)$ ως εξής :

$$w_k(n+1) = w_k(n) + n[r_{dx}(k) - \sum_{j=1}^{p} w_j(n)r_x(j,k)] \quad , \quad k=1,2, ... ,p$$

(25)

Η μέθοδος ταχύτερης καθόδου είναι ακριβής με την έννοια ότι δεν κάνει προσεγγίσεις στην παραγωγή της, η οποία βασίζεται στην ελαχιστοποίηση του MSE, που ορίζεται σαν :

$$J(n) = ½ E[e^2(n)]$$

(26)

Η παραπάνω συνάρτηση κόστους είναι ένας μέσος συνόλου, που παίρνεται σε μία συγκεκριμένη στιγμή n και πάνω σε ένα σύνολο χωρικών φίλτρων με παρόμοια σχεδίαση , αλλά διαφορετικές εισόδους , που παίρνονται από τον ίδιο πληθυσμό.

- Η μέθοδος ταχύτερης καθόδου μπορεί να προκύψει και από την ελαχιστοποίηση του αθροίσματος των τετραγώνων του λάθους :

$$\mathrm{E}_{total}(n) = \sum_{i=1}^{n} \mathrm{E}(i) = \frac{1}{2} \sum_{i=1}^{n} e^2(i) \qquad (27)$$

όπου η ολοκλήρωση παίρνεται τώρα πάνω σε όλες τις επαναλήψεις του αλγορίθμου, αλλά για συγκεκριμένη υλοποίηση του φίλτρου. Αυτή η δεύτερη προσέγγιση δίνει ίδια αποτελέσματα με την εξίσωση (25), αλλά με διαφορετική ερμηνεία των συναρτήσεων συσχέτισης. Συγκεκριμένα, η συνάρτηση αυτοσυσχέτισης $\mathbf{r_x}$, και η συνάρτηση ετεροσυσχέτισης $\mathbf{r_{dx}}$ ορίζονται τώρα σαν χρονικές μέσες τιμές παρά σαν μέσες τιμές συνόλου. Αν η φυσική διαδικασία που παράγει τις εισόδους και την επιθυμητή απόκριση είναι από κοινού εργοδικές, τότε μπορούμε να αντικαταστήσουμε τις χρονικές μέσες τιμές με τις μέσες τιμές συνόλου.

Παρατηρήσεις:

1. Ανεξάρτητα από ποια προσέγγιση θα χρησιμοποιήσουμε, για να δουλέψει η μέθοδος ταχύτερης καθόδου, πρέπει να δώσουμε ιδιαίτερη προσοχή στην επιλογή της παραμέτρου μάθησης.
2. Ένας πρακτικός περιορισμός της μεθόδου είναι ότι απαιτεί τη γνώση των χωρικών συναρτήσεων συσχέτισης $\mathbf{r_{dk}(k)}$ και $\mathbf{r_x(j,k)}$. Όταν το φίλτρο λειτουργεί σε ένα άγνωστο περιβάλλον αυτές οι συναρτήσεις δεν είναι διαθέσιμες και σε αυτή την περίπτωση αναγκαζόμαστε να χρησιμοποιήσουμε τις εκτιμήσεις τους.
3. Ο αλγόριθμος Ελάχιστου Μέσου Τετραγωνικού Λάθους (LMS) που περιγράφεται στη συνέχεια , προκύπτει από ένα απλό και συγχρόνως αποδοτικό τρόπο υπολογισμού αυτών των εκτιμήσεων.

3.2.3 Η απόδειξη του αλγορίθμου LMS

Ο αλγόριθμος LMS βασίζεται στη χρήση στιγμιαίων εκτιμήσεων της συνάρτησης αυτοσυσχέτισης $\mathbf{r_x(j,k)}$ και της συνάρτησης ετεροσυσχέτισης $\mathbf{r_{dk}(k)}$. Αυτές οι εκτιμήσεις συνάγονται απ' ευθείας από τις εξισώσεις ορισμού (15) και (16)

ως εξής :

$$\hat{r}_k(j,k) = x_j(n)x_k(n) \tag{28}$$

και $$\hat{r}_{dx}(k;n) = x_k(n)d(n) \tag{29}$$

Οι ορισμοί που δίνονται από τις (28) και (29) έχουν γενικευθεί για να περιλαμβάνουν ένα μη στάσιμο περιβάλλον. Σε αυτή την περίπτωση τόσο τα σήματα των αισθητήρων, όσο και οι επιθυμητές αποκρίσεις είναι χρονικά μεταβαλλόμενες. Άρα, αντικαθιστώντας τις $\mathbf{r_x(j,k)}$ και $\mathbf{r_{dk}(k)}$ στην (25) με τις εκτιμήσεις τους έχουμε :

$$\hat{w}_k(n+1) = \hat{w}_k(n) + \eta\left[x_k(n)d(n) - \sum_{j=1}^{p}\hat{w}_j(n)x_j(n)x_k(n)\right]$$

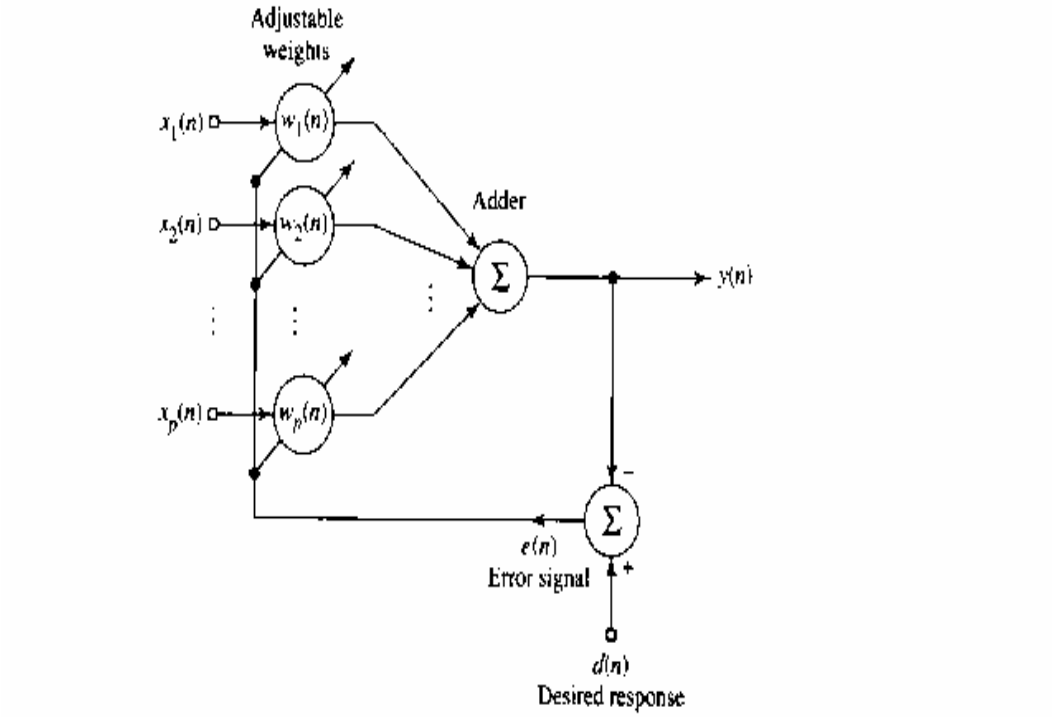
$$= \hat{w}_k(n) + \eta\left[d(n) - \sum_{j=1}^{p}\hat{w}_j(n)x_j(n)\right]x_k(n)$$

$$= \hat{w}_k(n) + \eta\left[d(n) - y(n)\right]x_k(n)\ ,\ \text{k=1,2,...p} \tag{30}$$

όπου y(n) είναι η έξοδος του χωρικού φίλτρου που υπολογίζεται στη n-στή επανάληψη σύμφωνα με τον αλγόριθμο LMS, δηλαδή :

$$y(n) = \sum_{j=1}^{p}\hat{w}_j(n)x_j(n) \tag{31}$$

Σημειώστε ότι στην εξίσωση (30) χρησιμοποιούμε $\hat{w}_k(n)$ αντί του $w_k(n)$, για να δώσουμε έμφαση στο γεγονός ότι η εξίσωση (30) περιλαμβάνει ‘’ εκτιμήσεις ‘’ των βαρών του χωρικού φίλτρου.



Σχήμα 7: Προσαρμοζόμενο χωρικό φίλτρο.

- Το σχήμα 7 δείχνει το λειτουργικό περιβάλλον του αλγορίθμου LMS , ο οποίος περιγράφεται πλήρως από τις εξισώσεις (44) και (45).

Μία σύνοψη του αλγορίθμου LMS , φαίνεται στον παρακάτω πίνακα 2, από τον οποίο φαίνεται και η απλότητα του αλγορίθμου.

ΠΙΝΑΚΑΣ 2: Ο αλγόριθμος LMS.

1. Initialization. Set

$$\hat{w}_k(1) = 0 \quad \text{for } k = 1, 2, \ldots, p$$

2. Filtering. For time n= 1, 2, ... , compute:

$$y(n) = \sum_{j=1}^{p} \hat{w}_j(n) x_j(n)$$

$$e(n) = d(n) - y(n)$$

$$\hat{w}_k(n+1) = w_k(n) + \eta e(n) x_k(n) \quad \text{for } k = 1, 2, \ldots, p$$

- Όπως φαίνεται στον πίνακα , για την αρχικοποίηση του αλγορίθμου , συνηθίζεται

να βάζουμε τα βάρη σε αρχικές τιμές ίσες με το μηδέν.

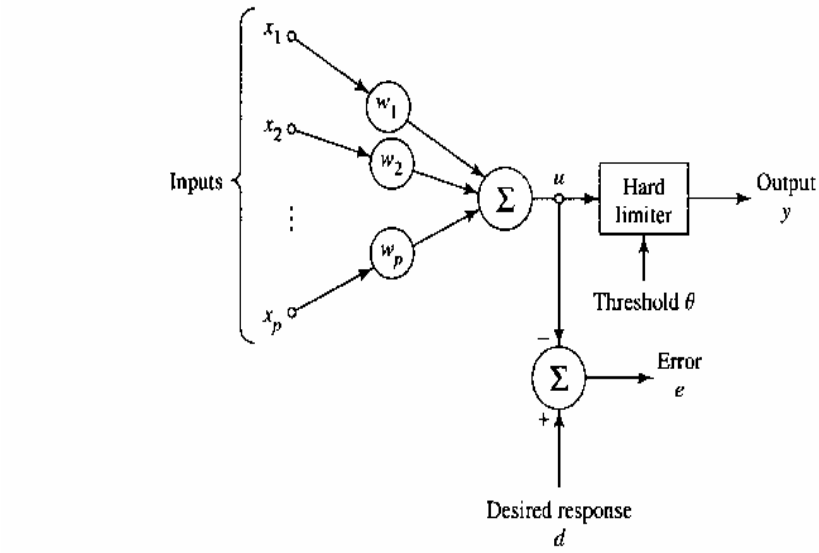
- Στη μέθοδο ταχύτερης καθόδου, που εφαρμόζεται σε ένα ‘’ γνωστό ‘’ περιβάλλον το διάνυσμα βαρών w(n), που αποτελείται από τα βάρη $w_1(n)$, $w_2(n)$,...., $w_p(n)$, αρχίζει με αρχική τιμή w(0) , και μετά ακολουθεί μία ακριβώς καθορισμένη τροχιά (πάνω στην επιφάνεια λάθους) , η οποία πράγματι τελειώνει πάνω στη βέλτιστη λύση w_0 , δεδομένου ότι η παράμετρος μάθησης $\boldsymbol{\eta}$ έχει εκλεγεί κατάλληλα. Αντίθετα , στον αλγόριθμο LMS , που εφαρμόζεται σε ένα ‘’ άγνωστο ‘’ περιβάλλον , το διάνυσμα βάρους $\hat{w}(n)$, που είναι μία εκτίμηση του w(n) , ακολουθεί μία τυχαία τροχιά. Γι’ αυτό το λόγο , ο αλγόριθμος LMS μερικές φορές αναφέρεται σαν στοχαστικός αλγόριθμος κλίσης (stochastic gradient algorithm). Καθώς ο αριθμός των επαναλήψεων , στον αλγόριθμο LMS , πλησιάζει το άπειρο , το $\hat{w}(n)$ εκτελεί ένα τυχαίο περίπατο (Brownian motion) , γύρω από την βέλτιστη λύση w_0 .
- Σε κάθε επανάληψη n , η μέθοδος ταχύτερης καθόδου ελαχιστοποιεί το μέσο τετραγωνικό λάθος J(n). Αυτή η συνάρτηση κόστους περιλαμβάνει μέση τιμή συνόλου , πράγμα που αυξάνει την ακρίβεια καθώς αυξάνει το n. Ενώ ο αλγόριθμος LMS , ελαχιστοποιεί τη στιγμιαία εκτίμηση της J(n) , άρα το διάνυσμα κλίσης του LMS είναι ‘’τυχαίο’’, και η ακρίβεια του βελτιώνεται κατά μέση τιμή, καθώς αυξάνει το n.
- Επειδή η μέθοδος ταχύτερης καθόδου ελαχιστοποιεί το άθροισμα των τετραγώνων του λάθους E_{total} (n), για όλες τις προηγούμενες επαναλήψεις, περιλαμβανομένης και της n, απαιτεί αποθήκευση μεγάλου όγκου πληροφορίας. Ο LMS ελαχιστοποιώντας το στιγμιαίο λάθος E(n), ελαχιστοποιεί τις απαιτήσεις μνήμης.

Ο αλγόριθμος LMS λειτουργεί τόσο σε στάσιμο όσο και σε μη-στάσιμο περιβάλλον. Άρα ο LMS όχι μόνο αναζητά αλλά ανιχνεύει το βέλτιστο. Από αυτή την άποψη, όσο μικρότερη είναι η τιμή του $\boldsymbol{\eta}$, τόσο καλύτερη είναι η σύγκλιση του αλγόριθμου. Όμως η βελτίωση της απόδοσης έχει σαν κόστος χαμηλό ρυθμό προσαρμογής.

3.2.4 Γραμμικό προσαρμοζόμενο στοιχείο - ADALINE (Adaptive Linear Element)

Το Adaline (Adaptive linear element - γραμμικό προσαρμοζόμενο στοιχείο), αρχικά

χρησιμοποιήθηκε από τους Widrow και Hoff και είναι μία προσαρμοζόμενη μηχανή ταξινόμησης προτύπων, που χρησιμοποιεί τον αλγόριθμο LMS για τη λειτουργία του. Ένα μπλοκ διάγραμμα του Adaline φαίνεται στο σχήμα 8. Αποτελείται από ένα γραμμικό συνδυαστή , μία συσκευή κατωφλίου και ένα μηχανισμό προσαρμογής των βαρών. Οι είσοδοι $x_1,x_2,...,x_p$ παίρνουν την τιμή ± 1. Μια μεταβλητή κατωφλίου θ ($(\theta \in [0,1])$ εφαρμόζεται στη συσκευή κατωφλίου. Η επιθυμητή έξοδος d παίρνει



Σχήμα 8: Το λειτουργικό διάγραμμα του Adaline.

επίσης τιμές ± 1. Τα βάρη $w_1,w_2,...,w_p$ και το κατώφλι θ προσαρμόζονται σύμφωνα με τον αλγόριθμο LMS , χρησιμοποιώντας το λάθος e=d-u . Η έξοδος του Adaline y παίρνεται , περνώντας την έξοδο u του γραμμικού συνδυαστή μέσα από τη συσκευή κατωφλιού. Έτσι έχουμε :

$$y = \begin{cases} +1 & u \geq \theta \\ -1 & u < \theta \end{cases} \qquad (32)$$

Αν e_α είναι το πραγματικό λάθος e_α =d-y , ο στόχος της προσαρμοζόμενης διαδικασίας στο Adaline είναι ο εξής:

Δοσμένου ενός συνόλου προτύπων εισόδου και των σχετικών επιθυμητών εξόδων , να βρεθεί το βέλτιστο σύνολο των συναπτικών βαρών και του κατωφλίου θ , έτσι ώστε να ελαχιστοποιηθεί το MSE του πραγματικού λάθους e_α .Επειδή είναι $d=\pm 1$ και $y=\pm 1$ $\Rightarrow e_a = \pm 2$. Άρα, η ελαχιστοποίηση της MSE τιμής των e_α είναι ισοδύναμη με την ελαχιστοποίηση του μέσου αριθμού των πραγματικών βαρών.

Κατά την εκπαίδευση, η μηχανή μαθαίνει κάτι από κάθε πρότυπο και ως εκ τούτου,

από αυτή την εμπειρία πραγματοποιεί μια αλλαγή στη σχεδίαση της. Η συνολική εμπειρία που αποκτάται από τη μηχανή αποθηκεύεται στις τιμές των βαρών και του θ. Το Adaline, μπορεί επίσης να εκπαιδευτεί από μία ακολουθία προτύπων με θόρυβο, στη βάση ενός περάσματος , έτσι ώστε η διαδικασία συγκλίνει κατά ένα στατιστικό τρόπο. Όταν η εκπαίδευση την Adaline τελειώσει , μπορεί να χρησιμοποιηθεί για να ταξινομήσει αυθεντικά πρότυπα και θορυβώδεις ή παραμορφωμένες εκδόσεις αυτών.

Άσκηση αυτοαξιολόγησης 3.2/4:

Ποιά είναι τα μειονεκτήματα της μεθόδου ταχύτερης καθόδου.

Απάντηση:

1. Για να δουλέψει αυτή η μέθοδος πρέπει να δώσουμε ιδιαίτερη προσοχή στην επιλογή της παραμέτρου μάθησης.
2. Απαιτεί τη γνώση των χωρικών συναρτήσεων συσχέτισης, οι οποίες είναι συνήθως άγνωστες.
3. Ελαχιστοποιεί το άθροισμα των τετραγώνων του λάθους, για όλες τις επαναλήψεις, άρα απαιτεί την αποθήκευση μεγαλύτερου όγκου πληροφορίας.

Άσκηση αυτοαξιολόγησης 3.2/5:

Ποιά είναι τα πλεονεκτήματα του αλγορίθμου LMS.

Απάντηση:

1. Υπολογίζει εκτιμήσεις των χωρικών συναρτήσεων συσχέτισης με ένα απλό και συγχρόνως αποδοτικό τρόπο.
2. Ελαχιστοποιεί το στιγμιαίο λάθος E(n), άρα ελαχιστοποιεί τις απαιτήσεις μνήμης.
3. Λειτουργεί τόσο σε στάσιμο όσο και σε μη-στάσιμο περιβάλλον.

Άσκηση αυτοαξιολόγησης 3.2 / 6:

Ο κανονικοποιημένος (normalized) LMS περιγράφεται από την ακόλουθη εξίσωση ενημέρωσης του διανύσματος των βαρών:

$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) + \frac{\tilde{\eta}}{\|\mathbf{x}(n)\|^2} \cdot e(n) \cdot \mathbf{x}(n)$$

όπου $\tilde{\eta}$ είναι μια θετική σταθερά και $\|\mathbf{x}(n)\|$ είναι η Ευκλείδεια νόρμα του

διανύσματος εισόδου **x**(n).Δείξτε ότι προκειμένου ο κανονικοποιημένος LMS να συγκλίνει στο μέσο τετράγωνο (mean square) θα πρέπει: $0 < \tilde{\eta} < 2$

Απάντηση: Για τον «συμβατικό» LMS ισχύει:

$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) + \eta \cdot e(n) \cdot \mathbf{x}(n) \qquad (1)$$

Όπως γνωρίζουμε από τη θεωρία, προκειμένου να έχουμε σύγκλιση στο μέσο τετράγωνο πρέπει να ισχύει:

$$0 < \eta < \frac{2}{\|\mathbf{x}(n)\|^2} \qquad (2)$$

Για τον κανονικοποιημένο LMS ισχύει:

$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) + \frac{\tilde{\eta}}{\|\mathbf{x}(n)\|^2} \cdot e(n) \cdot \mathbf{x}(n) \qquad (3)$$

Από τις σχέσεις (1) και (3) προκύπτει:

$$\tilde{\eta} = \eta \cdot \|\mathbf{x}(n)\|^2 \text{ ή } \eta = \frac{\tilde{\eta}}{\|\mathbf{x}(n)\|^2}$$

Χρησιμοποιώντας το αποτέλεσμα αυτό στην σχέση (2) προκύπτει οτι για να συγκλίνει ο κανονικοποιημένος LMS θα πρέπει: $0 < \tilde{\eta} < 2$

Άσκηση αυτοαξιολόγησης 3.2 / 7:

Θεωρείστε ένασύστημα γραμμικής πρόβλεψης (Linear Predictor) όπου το διάνυσμα εισόδου του αποτελείται από τα δείγματα x(n-1), x(n-2), … , x(n-m), όπου το m είναι το βήμα πρόβλεψης (prediction order). Χρησιμοποιήστε τον LMS αλγόριθμο για να κάνετε μια πρόβλεψη $\hat{x}(n)$ του δείγματος εισόδου x(n). Υλοποιήστε την αναδρομή που απαιτείται για να υπολογίσετε το διάνυσμα των βαρών w_1, w_2, … ,w_m του predictor.

Απάντηση:

Το διάνυσμα εισόδου είναι:

$$\boldsymbol{x}(n-1) = [x(n-1), x(n-2), \ldots, x(n-m)]^T$$

Η επιθυμητή απόκριση είναι d(n) = x(n).

Οπότε οι εξισώσεις του LMS αλγορίθμου για τον one-step predictor (predictor ενός

βήματος) είναι οι εξής:

$$\hat{x}(n) = \mathbf{w}^T(n) \cdot \mathbf{x}(n-1)$$
$$e(n) = x(n) - \hat{x}(n)$$
$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta \cdot e(n) \cdot \mathbf{x}(n-1)$$

3.3 Ο αλγόριθμος Πίσω Διάδοσης (Π.Δ.) του λάθους

Σε αυτή την ενότητα θα μελετήσουμε μία σπουδαία κλάση νευρωνικών δικτύων, τα δίκτυα εμπρός τροφοδότησης πολλών επιπέδων. Τυπικά ένα τέτοιο δίκτυο αποτελείται από ένα σύνολο αισθητήρων (πηγαίοι κόμβοι), που αποτελούν το επίπεδο εισόδου, ένα ή περισσότερα κρυφά επίπεδα (hidden layers) υπολογιστικών κόμβων και ένα επίπεδο υπολογιστικών κόμβων εξόδου. Το σήμα εισόδου διαδίδεται μέσα στο δίκτυο σε μία προς τα εμπρός κατεύθυνση, από επίπεδο σε επίπεδο. Αυτά τα νευρωνικά δίκτυα αναφέρονται σαν Perceptrons πολλών επιπέδων (Multi Layer Perceptrons- MLPs) τα οποία είναι μια γενίκευση του απλού Perceptron.

Τα MLPs έχουν εφαρμοστεί με επιτυχία στην επίλυση δύσκολων και ποικίλων προβλημάτων, εκπαιδεύοντας τα με έναν επιβλεπόμενο τρόπο (supervised manner), με ένα πολύ δημοφιλή αλγόριθμο γνωστό σαν αλγόριθμο πίσω διάδοσης του λάθους (error Back Propagation algorithm - BP). Αυτός ο αλγόριθμος βασίζεται στον κανόνα μάθησης διόρθωσης του λάθους (error correction learning rule).

Βασικά η διαδικασία της πίσω διάδοσης του λάθους αποτελείται από δυο περάσματα διαμέσου των διαφορετικών επιπέδων του δικτύου ένα προς τα εμπρός πέρασμα (forward pass) και ένα προς τα πίσω πέρασμα (backward pass).

- Στο εμπρός πέρασμα ένα διάνυσμα εισόδου (input vector) εφαρμόζεται στους νευρώνες εισόδου του δικτύου, και η επίδραση του διαδίδεται μέσα στο δίκτυο από επίπεδο σε επίπεδο (layer by layer). Τελικά ένα σύνολο από εξόδους παράγεται ως η πραγματική απόκριση του δικτύου. Κατά τη διάρκεια του εμπρός περάσματος τα βάρη του δικτύου είναι σταθερά.

- Από την άλλη μεριά κατά τη διάρκεια της πίσω διάδοσης τα βάρη προσαρμόζονται σε συμφωνία με τον κανόνα διόρθωσης λάθους.

Πιο συγκεκριμένα, η πραγματική απόκριση του δικτύου αφαιρείται από την

επιθυμητή απόκριση για την παραγωγή ενός σήματος λάθους, που διαδίδεται προς τα πίσω στο δίκτυο, αντίθετα από την κατεύθυνση των συνδέσεων, από το οποίο προκύπτει και το όνομα πίσω διάδοσης του λάθους.

Τα συναπτικά βάρη προσαρμόζονται έτσι ώστε να κάνουν την πραγματική απόκριση του δικτύου να πλησιάσει την επιθυμητή απόκριση.

Στην βιβλιογραφία ο αλγόριθμος πίσω διάδοσης του λάθους συχνά αναφέρεται και σαν αλγόριθμος πίσω διάδοσης (Back Propagation Algorithm) ή πιο απλά σαν Back Prop. Από δω και στο εξής θα αναφερόμαστε σε αυτόν σαν αλγόριθμο πίσω διάδοσης ή Π.Δ.. Η διαδικασία μάθησης που εκτελείται με αυτόν τον αλγόριθμο ονομάζεται μάθηση πίσω διάδοσης.

Ένα Perceptron πολλών επιπέδων έχει τρία διακριτικά χαρακτηριστικά:

1. Το μοντέλο κάθε νευρώνα στο δίκτυο περιλαμβάνει μια μη γραμμικότητα στην έξοδο. Ένα σημαντικό σημείο στο οποίο πρέπει να δώσουμε έμφαση εδώ, είναι ότι η μη γραμμικότητα είναι 'εξομαλισμένη' (smooth), δηλαδή είναι παντού παραγωγίσιμη. Μία συνηθισμένη μορφή μη γραμμικότητας που ικανοποιεί αυτήν την προϋπόθεση είναι μια σιγμοειδής μη γραμμικότητα (sigmoidal nonlinearity) που ορίζεται από την παρακάτω λογιστική συνάρτηση:

$$y_j = \frac{1}{1+\exp(-\upsilon_j)} \tag{33}$$

όπου υ_j : η τιμή ενεργοποίησης του νευρώνα j

και y_j : η έξοδος του νευρώνα j

Η παρουσία μη γραμμικοτήτων είναι σημαντική, διότι διαφορετικά η σχέση εισόδου-εξόδου του δικτύου μπορούσε να ελαττωθεί σ'αυτή του perceptron ενός επιπέδου. Επιπλέον η χρήση της λογιστικής συνάρτησης έχει βιολογικά κίνητρα μιας και προσπαθεί να δικαιολογήσει την επίμονη φάση των πραγματικών νευρώνων. (χαρακτηριστικό των πραγματικών βιολογικών νευρώνων είναι ότι δεν έχουν δυαδικές εξόδους, αλλά η έξοδος τους έχει συνεχώς κάποια τιμή).

2.Το δίκτυο περιέχει ένα ή περισσότερα κρυφά επίπεδα από νευρώνες τα οποία δεν είναι τμήμα της εισόδου ή της εξόδου του δικτύου. Αυτοί οι κρυφοί νευρώνες

δίνουν την δυνατότητα στο δίκτυο να μάθει πολύπλοκες εργασίες με το να εξάγουν προοδευτικά τα πιο σημαντικά χαρακτηριστικά από τα διανύσματα εισόδου.

3.Το δίκτυο επιδεικνύει έναν υψηλό βαθμό διασύνδεσης (connectivity) που καθορίζεται από τις συνδέσεις (συνάψεις) του δικτύου. Μία αλλαγή στον τρόπο διασυνδέσεις του δικτύου απαιτεί αλλαγή στον πληθυσμό των συνδέσεων ή στα βάρη τους.

Πράγματι το Perceptron πολλών επιπέδων αντλεί την υπολογιστική του ισχύ μέσω του συνδυασμού αυτών των χαρακτηριστικών μαζί με την ικανότητα να μαθαίνει από την εμπειρία διαμέσου της εκπαίδευσης. Αυτά τα ιδιοχαρακτηριστικά όμως είναι επίσης υπεύθυνα για της ελλείψεις στην παρούσα κατάσταση της γνώσης μας πάνω στη συμπεριφορά του δικτύου.

- Πρώτον η παρουσία μιας κατανεμημένης μορφής μη γραμμικότητας και η υψηλή διασύνδεση του δικτύου κάνουν την θεωρητική ανάλυση ενός Perceptron πολλών επιπέδων, πολύ δύσκολο να επιχειρηθεί.
- Δεύτερον η χρήση κρυφών νευρώνων κάνει την διαδικασία μάθησης πιο δύσκολη στο να κατανοηθεί. Κατά μια έννοια η διαδικασία μάθησης πρέπει να αποφασίσει ποια χαρακτηριστικά των διανυσμάτων εισόδου πρέπει να παρασταθούν από τους κρυφούς νευρώνες. Επομένως η διαδικασία μάθησης γίνεται πιο δύσκολη επειδή η έρευνα πρέπει να διεξαχθεί σε ένα πολύ μεγαλύτερο χώρο από πιθανές συναρτήσεις και πρέπει να γίνει μια επιλογή μεταξύ εναλλακτικών αναπαραστάσεων του διανύσματος εισόδου.

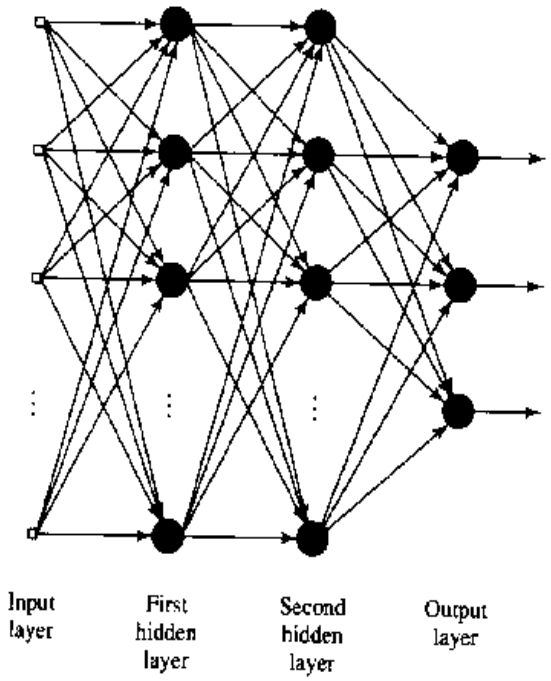
Η ανάπτυξη του αλγόριθμου πίσω διάδοσης αποτελεί ένα σταθμό στα νευρωνικά δίκτυα γιατί παρέχει μια υπολογιστικά αποδοτική μέθοδο για την εκπαίδευση πολυεπίπεδων Perceptrons. Αν και δεν μπορεί να παρέχει λύσεις για όλα τα προβλήματα που επιδέχονται επίλυση, είναι δίκαιο να πούμε ότι έβαλε στην άκρη την αρνητική προκατάληψη, για την μάθηση σε πολυεπίπεδες μηχανές που μπορεί να είχε συναχθεί από το βιβλίο των Minsky και Papert (1969).

Στο σχήμα 9 φαίνεται η γραφική αναπαράσταση ενός πολυεπόπεδου Perceptron με δύο κρυφά επίπεδα (hidden layers). Το δίκτυο που φαίνεται εδώ είναι πλήρως διασυνδεδεμένο (fully connected), πράγμα που σημαίνει ότι ένας νευρώνας

οποιουδήποτε επιπέδου, είναι συνδεδεμένος με όλους τους νευρώνες του προηγουμένου επιπέδου. Η ροή του σήματος στο δίκτυο προχωρά σε μια προς τα εμπρός κατεύθυνση, από τα αριστερά προς τα δεξιά από επίπεδο σε επίπεδο.

Στο σχήμα 10 απεικονίζεται ένα τμήμα ενός MLP. Σ'αυτό το δίκτυο αναγνωρίζονται δυο είδη σημάτων .

Λειτουργικά σήματα: Ενα λειτουργικό σήμα (function signal) είναι ένα σήμα εισόδου (ερέθισμα) που εισέρχεται από την απόληξη εισόδου του δικτύου και διαδίδεται προς τα εμπρός διαμέσου του δικτύου και εξέρχεται από την έξοδο του δικτύου σαν ένα σήμα εξόδου. Αναφερόμαστε σε ένα τέτοιο σήμα σαν "function signal" για δυο λόγους:

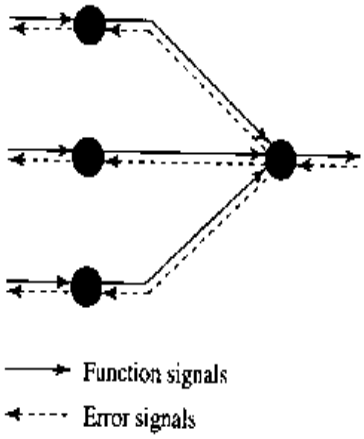


Σχήμα 9: Ο αρχιυεκτονικός γράφος ενός πολυεπίπεδου perceptron με δυο κρυφά επίπεδα.

- Πρώτον, υποτίθεται ότι επιτελεί μια χρήσιμη συνάρτηση στην έξοδο του δικτύου.
- Δεύτερον, σε κάθε νευρώνα του δικτύου, μέσω του οποίου περνά ένα λειτουργικό σήμα, το σήμα υπολογίζεται σαν μία συνάρτηση των εισόδων και των συσχετιζόμενων βαρών, που εφαρμόζονται στο νευρώνα.

2. ***Σήμα λάθους:*** Ένα σήμα λάθους (error signal) δημιουργείται σε έναν νευρώνα εξόδου του δικτύου και διαδίδεται προς τα πίσω (layer by layer) διαμέσου του δικτύου. Αναφερόμαστε σ'αυτό σαν "error signal" επειδή ο υπολογισμός του από κάθε νευρώνα του δικτύου εμπεριέχει μια συνάρτηση εξαρτώμενη από το λάθος στην μια ή στην άλλη μορφή.

Οι νευρώνες εξόδου αποτελούν το επίπεδο εξόδου του δικτύου. Οι υπόλοιποι νευρώνες σχηματίζουν τα κρυφά επίπεδα του δικτύου. Οι κρυφές μονάδες δεν ανήκουν στο επίπεδο εισόδου ή εξόδου του δικτύου για αυτό ονομάζονται και κρυφές (hidden). Το πρώτο κρυφό επίπεδο τροφοδοτείται από το επίπεδο εισόδου που αποτελείται από τις αισθητήριες μονάδες, οι έξοδοι που προκύπτουν από το πρώτο κρυφό επίπεδο εφαρμόζονται με τη σειρά τους στο επόμενο κρυφό επίπεδο και ούτω καθεξής για το υπόλοιπο του δικτύου.



Σχήμα 10: Απεικόνιση των διευθύνσεων των δυο βασικών σημάτων ροής σε ένα πολυεπίπεδο Perceptron.

Κάθε κρυφός νευρώνας ή νευρώνας εξόδου του πολυεπίπεδου Perceptron σχεδιάζεται έτσι ώστε να επιτελεί δυο υπολογισμούς:

1. Ο υπολογισμός του λειτουργικού σήματος που εμφανίζεται στην έξοδο ενός νευρώνα, το οποίο εκφράζεται σαν μια συνεχής μη γραμμική συνάρτηση των σημάτων εισόδου και των συναπτικών βαρών που σχετίζονται με τον νευρώνα.

2. Ο υπολογισμός μιας στιγμιαίας εκτίμησης του διανύσματος κλίσης, ο οποίος χρειάζεται για την πίσω διάδοση μέσω του δικτύου.

Η παραγωγή του αλγόριθμου πίσω διάδοσης είναι πολύπλοκη. Για να διευκολύνουμε την μαθηματική επιβάρυνση που εμπεριέχεται σ'αυτή τη διαδικασία παρουσιάζουμε μια σύνοψη από συμβολισμούς που χρησιμοποιούνται σ'αυτή την παραγωγή.

- Τα i,j και k αντιστοιχούν σε διαφορετικούς νευρώνες, με τα σήματα να διαδίδονται μέσα από το δίκτυο από τα αριστερά προς τα δεξιά, ο νευρώνας j βρίσκεται ένα επίπεδο αριστερά από τον νευρώνα i και ο νευρώνας k ένα επίπεδο αριστερά από τον νευρώνα j, όταν ο j είναι μια κρυφή μονάδα.
- Η επανάληψη n αντιστοιχεί στο ν-οστό διάνυσμα εκπαίδευσης που δόθηκε σαν είσοδος στο δίκτυο.
- Το σύμβολο $\mathbf{E}$(n) είναι το στιγμιαίο άθροισμα των τετραγωνικών λαθών στην επανάληψη n. Ο μέσος όρος του $\mathbf{E}$(n) όλων των τιμών του n είναι το μέσο τετραγωνικό λάθος $\mathbf{E}_{av.}$
- Το σύμβολο e_j (n) αντιστοιχεί στο σήμα λάθους στην έξοδο του νευρώνα j για την επανάληψη n.
- Το σύμβολο d_j (n) αντιστοιχεί στην επιθυμητή απόκριση για τον νευρώνα j και χρησιμοποιείται στον υπολογισμό του e_j (n) .
- Το σύμβολο y_j (n) αντιστοιχεί στο λειτουργικό σήμα στην έξοδο του νευρώνα j για την επανάληψη n.
- Το σύμβολο w_{ij} (n) είναι το συναπτικό βάρος που συνδέει τον νευρώνα i, με τον νευρώνα j κατά την διάρκεια της επανάληψης n. Η ποσότητα κατά την οποία διορθώνεται το βάρος της σύναψης στη επανάληψη n συμβολίζεται με Δw_{ij} (n) .
- Η τιμή ενεργοποίησης του νευρώνα j στην επανάληψη n συμβολίζεται με υ_j (n).
- Η συνάρτηση ενεργοποίησης του νευρώνα j συμβολίζεται με $\varphi_j(\cdot)$.
- Το κατώφλι το οποίο εφαρμόζεται στον νευρώνα j συμβολίζεται με ϑ_j (n). Συνήθως αναπαριστάται με μια σύναψη με βάρος $w_{j0} = \vartheta_j$ συνδεδεμένο σε μια σταθερή

είσοδο που ισούται με -1.

- Το i-οστο στοιχείο του διανύσματος εισόδου συμβολίζεται με x_i(n).
- Το k-οστο στοιχείο του συνολικού διανύσματος εξόδου συμβολίζεται με o_k (n).
- Η παράμετρος μάθησης συμβολίζεται με $\boldsymbol{\eta}$

3.3.1 Παραγωγή του αλγορίθμου Πίσω Διάδοσης.

Το σήμα λάθους στην έξοδο του νευρώνα j στην επανάληψη n ορίζεται από την σχέση:

$$e_j = d_j - y_j \quad \text{όπου ο νευρώνας j είναι κόμβος εξόδου} \qquad (34)$$

Ορίζουμε την στιγμιαία τιμή του τετραγωνικού λάθους για τον νευρώνα j σαν $\frac{1}{2}e_j^2(n)$. Έτσι το στιγμιαίο άθροισμα των τετραγωνικών λαθών του δικτύου γράφεται ως εξής:

$$\mathbf{E}(\mathrm{n}) = \frac{1}{2}\sum_{j \in C}^{N} e_j^2(n) \qquad (35)$$

Όπου το σύνολο C περιλαμβάνει όλους τους νευρώνες του επιπέδου εξόδου του δικτύου. Έστω N ο συνολικός αριθμός διανυσμάτων στο σύνολο εκπαίδευσης. Το μέσο τετραγωνικό λάθος για όλο το σύνολο εκπαίδευσης είναι:

$$\mathbf{E}_{\mathrm{av}} = \frac{1}{N}\sum_{n=1}^{N} \mathbf{E}(\mathrm{n}) \qquad (36)$$

Το στιγμιαίο άθροισμα των τετραγωνικών λαθών $\mathbf{E}$(n), και κατά συνέπεια και το μέσο τετραγωνικό λάθος $\mathbf{E}_{\mathrm{av}}$, είναι μια συνάρτηση όλων των ελεύθερων παραμέτρων (π.χ συναπτικά βάρη και κατώφλια) του δικτύου. Για ένα δοσμένο εκπαιδευτικό σύνολο (training set), η $\mathbf{E}_{\mathrm{av}}$ αντιπροσωπεύει την συνάρτηση κόστους (cost function) σαν το

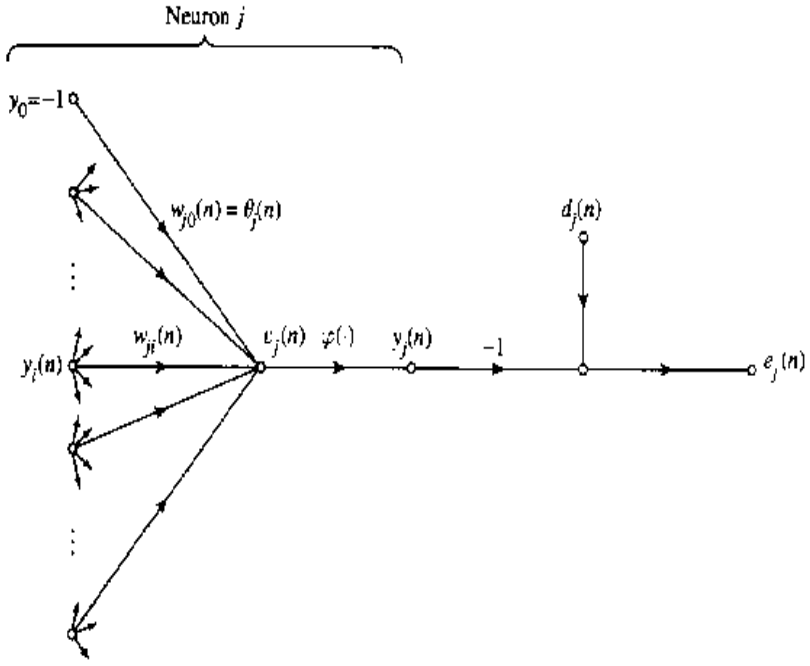
μέτρο για την απόδοση μάθησης του συνόλου εκπαίδευσης. Ο αντικειμενικός σκοπός της διαδικασίας μάθησης είναι να προσαρμόσει τις ελεύθερες παραμέτρους του δικτύου έτσι ώστε να ελαχιστοποιήσει το $\mathbf{E}_{av}$.

3.3.2 Διαδικασία Μάθησης.

Θεωρούμε μια απλή μέθοδο εκπαίδευσης στην οποία τα βάρη ενημερώνονται σε μια πρότυπο προς πρότυπο (pattern by pattern) βάση. Οι προσαρμογές (ρυθμίσεις) των βαρών γίνονται σε συμφωνία με τα αντίστοιχα λάθη που υπολογίζονται για κάθε πρότυπο που παρουσιάζεται στην είσοδο.

Ο αριθμητικός μέσος όρος αυτών των ατομικών αλλαγών στα βάρη, πάνω στο σύνολο εκπαίδευσης είναι λοιπόν μια εκτίμηση της πραγματικής αλλαγής στα βάρη που θα συνέβαινε, από την μεταβολή (ρύθμιση) των βαρών για την ελαχιστοποίηση της συνάρτησης κόστους $\mathbf{E}_{av}$ πάνω στο συνολικό σύνολο εκπαίδευσης.

Στο σχήμα 11 απεικονίζεται ο νευρώνας j ο οποίος δέχεται ένα σύνολο από λειτουργικά σήματα που παράγονται από το επίπεδο στα αριστερά του.



Σχήμα 11: Γράφος που δείχνει με λεπτομέρεια τη ροή των σημάτων στον νευρώνα

εξόδου j.

Για τον νευρώνα j έχουμε:

$$\upsilon_j(n) = \sum_{i=1}^{p} w_{ji}(n) y_i(n) \qquad (37)$$

όπου p είναι ο συνολικός αριθμός εισόδων (εξαιρούμε το κατώφλι) που εφαρμόζονται στον νευρώνα j. Επίσης, θέτουμε $w_{j0} = \theta_j$.

Επομένως το λειτουργικό σήμα $y_j(n)$ στην έξοδο του νευρώνα j θα είναι

$$y_j(n) = \varphi_j(\upsilon_j(n)) \qquad (38)$$

Με τρόπο παρόμοιο με τον LMS αλγόριθμο ο αλγόριθμος πίσω διάδοσης εφαρμόζει μια διόρθωση $\Delta w_{ji}(n)$ στο συναπτικό βάρος $w_{ji}(n)$, η οποία είναι ανάλογη της στιγμιαίας κλίσης $\partial \mathbf{E}(n)/\partial w_{ji}(n)$. Σύμφωνα με τον αλυσιδωτό κανόνα (Chain Rule) μπορούμε να εκφράσουμε την κλίση ως εξής:

$$\frac{\partial \mathbf{E}(n)}{\partial w_{ji}(n)} = \frac{\partial \mathbf{E}(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial \upsilon_j(n)} \frac{\partial \upsilon_j(n)}{\partial w_{ji}(n)} \qquad (39)$$

Η κλίση $\frac{\partial \mathbf{E}(n)}{\partial w_{ji}(n)}$ αντιπροσωπεύει ένα παράγοντα ευαισθησίας, καθορίζοντας την κατεύθυνση έρευνας στο χώρο των βαρών για το συναπτικό βάρος w_{ji}. Παραγωγίζοντας και τις δύο πλευρές της εξίσωσης (35) με το $e_j(n)$ παίρνουμε:

$$\frac{\partial \mathbf{E}(n)}{\partial w_{ji}(n)} = e_j(n) \qquad (40)$$

Παραγωγίζοντας τώρα και τις δυο πλευρές της εξίσωσης (34) με το $y_j(n)$ παίρνουμε:

$$\frac{\partial e_j(n)}{\partial y_j(n)} = -1 \qquad (41)$$

Τελικά παραγωγίζοντας την (38) και (37) με το $\upsilon_j(n)$ και $w_{ji}(n)$ αντίστοιχα παίρνουμε:

$$\frac{\partial y_j(n)}{\partial \upsilon_j(n)} = \varphi'_j(\upsilon_j(n)) \qquad (42)$$

$$\frac{\partial \upsilon_j(n)}{\partial w_{ji}(n)} = y_i(n) \qquad (43)$$

Αντικαθιστώντας τις εξισώσεις (40),(41),(42),(43) στην (39) παίρνουμε:

$$\frac{\partial E(n)}{\partial w_{ji}(n)} = -e_j(n)\varphi'_j(\upsilon_j(n))y_i(n) \qquad (44)$$

Η διόρθωση $\Delta w_{ji}(n)$ που εφαρμόζεται στο $w_{ji}(n)$ καθορίζεται από τον δέλτα κανόνα:

$$\Delta w_{ji}(n) = -\eta \frac{\partial E(n)}{\partial w_{ji}(n)} \qquad (45)$$

όπου το η ονομάζεται παράμετρος ρυθμού μάθησης (learning rate parameter). Η χρήση του αρνητικού συμβόλου (-) στην (45) ερμηνεύεται σαν πτώση της κλίσης στο χώρο των βαρών. Από τις (44) και (45) παίρνουμε:

$$\Delta w_{ji}(n) = \eta \delta_j(n) y_i(n) \qquad (46)$$

όπου η τοπική κλίση $\delta_j(n)$ ορίζεται από την σχέση:

$$\ddot{a}_j(n) = -\frac{\partial \text{Æ}(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial \tilde{o}_j(n)} = e_j(n)\varphi'_j(\upsilon_j(n)) \quad (47)$$

Η τοπική κλίση δείχνει τις απαιτούμενες αλλαγές στα βάρη. Σύμφωνα με την εξίσωση (47) η τοπική κλίση $\delta_j(n)$ για τον νευρώνα εξόδου j είναι ίσο με το γινόμενο του αντίστοιχου σήματος λάθους $e_j(n)$ και της παραγώγου $\varphi'_j(\upsilon_j(n))$ της συνάρτησης ενεργοποίησης του. Από τις εξισώσεις (46) και (47) φαίνεται ότι ένας παράγοντας κλειδί που εμπλέκεται στον υπολογισμό της προσαρμογής (του βάρους) $\Delta w_{ji}(n)$ είναι το σήμα στην έξοδο του νευρώνα j. Στο σημείο αυτό μπορούμε να διακρίνουμε δυο περιπτώσεις, ανάλογα με το που είναι τοποθετημένος ο νευρώνας j, στο δίκτυο. Στην περίπτωση (I) ο j είναι ένας νευρώνας εξόδου. Την περίπτωση αυτή είναι απλό να την χειριστούμε, διότι κάθε νευρώνας εξόδου στο δίκτυο εφοδιάζεται με την επιθυμητή απόκρισή του, κάνοντας τον υπολογισμό του αντίστοιχου σήματος λάθους, μια εύκολη υπόθεση. Στην περίπτωση (II) ο νευρώνας j είναι ένας κρυφός νευρώνας .

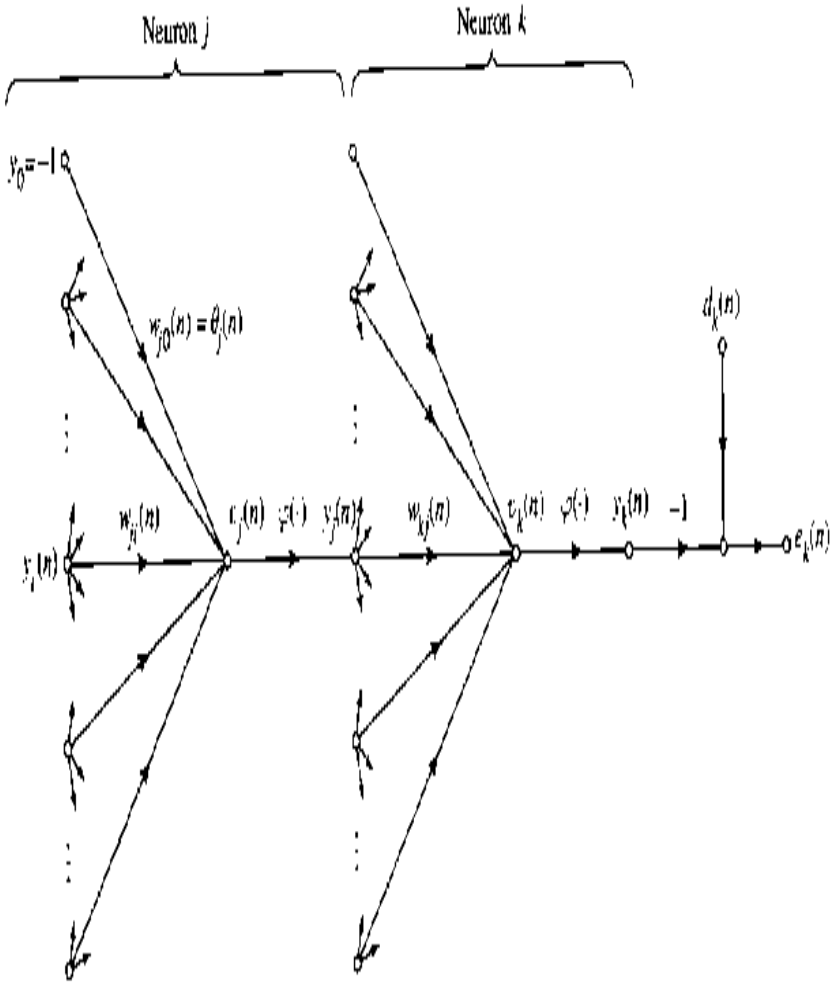
Αν και οι κρυφοί νευρώνες δεν είναι άμεσα προσπελάσιμοι, μοιράζονται ευθύνη για κάθε λάθος που συμβαίνει στην έξοδο του δικτύου. Το ζήτημα, όμως είναι να γνωρίζουμε πως να επιβάλλουμε ποινή (penalize) ή να επιβραβεύσουμε (reward) κρυφούς νευρώνες για το μερίδιο της ευθύνης τους. Αυτό το πρόβλημα είναι το πρόβλημα της επιβράβευσης (Credit-Assignment) [1]. Οπως θα δούμε στη συνέχεια, λύνεται με έναν κομψό τρόπο, με την πίσω διάδοση των σημάτων λάθους στο δίκτυο.
Στην συνέχεια θεωρούμε τις περιπτώσεις I και II.

Περίπτωση I : *Ο νευρώνας j είναι ένας κόμβος εξόδου.*

Όταν ο νευρώνας j βρίσκεται στο επίπεδο εξόδου του δικτύου, τροφοδοτείται με την επιθυμητή του έξοδο. Επομένως μπορούμε να χρησιμοποιήσουμε την εξίσωση (34) για να υπολογίσουμε το σήμα λάθους $e_j(n)$ που σχετίζεται με τον νευρώνα. Αφού έχουμε καθορίσει το σήμα λάθους, μετά είναι μια απλή διαδικασία να υπολογίσουμε την τοπική κλίση $\delta_j(n)$ χρησιμοποιώντας την εξίσωση (47).

Περίπτωση II : *Ο νευρώνας j είναι ένας κρυφός κόμβος.*

Όταν ένας νευρώνας j βρίσκεται σε ένα κρυφό επίπεδο του δικτύου, δεν υπάρχει κάποια καθορισμένη επιθυμητή απόκριση γι'αυτόν τον νευρώνα. Ανάλογα, το σήμα λάθους για ένα κρυφό νευρώνα θα έπρεπε να καθοριστεί επαναληπτικά σε όρους των σημάτων λάθους από όλους τους νευρώνες με τους οποίους αυτός ο κρυφός νευρώνας συνδέεται άμεσα. Εδώ είναι που περιπλέκεται η ανάπτυξη του αλγορίθμου πίσω διάδοσης . Θεωρούμε την κατάσταση που απεικονίζεται στο σχήμα 12, όπου θεωρούμε έναν νευρώνα j σαν ένα κρυφό νευρώνα του δικτύου.



Σχήμα 12: Γράφος που δείχνει με λεπτομέρεια τη ροή των σημάτων στον νευρώνα εξόδου k ο οποίος συνδέεται με τον κρυφό νευρώνα j.

Σύμφωνα με την εξίσωση (47) μπορούμε να ορίσουμε πάλι την τοπική κλίση $\delta_j(n)$ για κρυφό νευρώνα j ως εξής:

$$\delta_j(n) = -\frac{\partial E(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial \upsilon_j(n)}$$

$$= -\frac{\partial \mathrm{E}(\mathrm{n})}{\partial \mathrm{y}_\mathrm{j}(\mathrm{n})} \varphi'_j(\upsilon_j(n)) \text{ , όπου ο j είναι κρυφός νευρώνας} \quad (48)$$

Για να υπολογίσουμε την μερική παράγωγο $\partial \mathbf{E}(\mathrm{n}) / \partial \mathbf{y}_\mathrm{j}(\mathrm{n})$ μπορούμε να προχωρήσουμε ως εξής :

Από το σχήμα 12 βλέπουμε ότι το στιγμιαίο άθροισμα των τετραγώνων του λάθους στην *επανάληψη n, δίνεται από τη σχέση:*

$$\mathbf{E}(\mathrm{n}) = \frac{1}{2} \sum_{\mathrm{k} \in \mathrm{C}} \mathrm{e}_\mathrm{k}^2(\mathrm{n}), \qquad \text{όπου ο νευρώνας k είναι ένας κόμβος εξόδου} \quad (49)$$

Ας θυμηθούμε εδώ, ότι η στιγμιαία τιμή του τετραγωνικού λάθους**,** γιά το νευρώνα k ορίζεται σαν 1/2 $\mathbf{e}_\kappa^2(\mathrm{n})$ και η $\boldsymbol{\varepsilon}(\mathrm{n})$ προκύπτει αθροίζοντας τα 1/2 $\mathbf{e}_\kappa^2(\mathrm{n})$ για όλους τους νευρώνες εξόδου. Άρα, από την εξίσωση (49) έχουμε:

$$\frac{\partial \mathbf{E}(\mathrm{n})}{\partial \mathrm{y}_\mathrm{j}(\mathrm{n})} = \sum_\mathrm{k} \mathrm{e}_\mathrm{k} \frac{\partial \mathrm{e}_\mathrm{k}(\mathrm{n})}{\partial \mathrm{y}_\mathrm{j}(\mathrm{n})} \qquad (50)$$

Στη συνέχεια χρησιμοποιούμε τον αλυσιδωτό κανόνα για τον υπολογισμό της μερικής παράγωγου. Δηλαδή η $\partial \mathbf{e}_\mathrm{k}(\mathrm{n}) / \partial \mathbf{y}_\mathrm{j}(\mathrm{n})$ και η σχέση (50) ξαναγράφεται ως εξής:

$$\frac{\partial \mathbf{E}(\mathrm{n})}{\partial \mathrm{y}_\mathrm{j}(\mathrm{n})} = \sum_\mathrm{k} \mathrm{e}_\mathrm{k} \frac{\partial \mathrm{e}_\mathrm{k}(\mathrm{n})}{\partial \upsilon_\mathrm{k}(\mathrm{n})} \frac{\partial \upsilon_\mathrm{k}(\mathrm{n})}{\partial \mathrm{y}_\mathrm{j}(\mathrm{n})} \qquad (51)$$

Αλλά από την εξίσωση (37) έχουμε ότι:

$\mathrm{e}_\kappa(\mathrm{n}) = \mathrm{d}_\kappa(\mathrm{n}) - \mathrm{y}_\kappa(\mathrm{n})$ ή

$\mathrm{e}_\kappa(\mathrm{n}) = \mathrm{d}_\mathrm{K}(\mathrm{n}) - \varphi_\kappa(\upsilon_\kappa(\mathrm{n}))$, αν ο νευρώνας κ είναι κόμβος εξόδου (52)

Επομένως: $\frac{\partial e_\mathrm{k}(\mathrm{n})}{\partial \upsilon_\mathrm{k}(\mathrm{n})} = -\varphi'(\upsilon_\mathrm{k}(\mathrm{n}))$ (53)

Από το σχήμα 12, παρατηρούμε επίσης ότι για τον νευρώνα k, το εσωτερικό επίπεδο ενεργοποίησης του δικτύου είναι:

$$\upsilon_k(n) = \sum_{j=0}^{q} w_{kj}(n) - y_j(n) \qquad (54)$$

όπου q είναι ο συνολικός αριθμός εισόδων (εξαιρούμε το κατώφλι), που εφαρμόζονται στον νευρώνα k. Και εδώ, επίσης το $w_{k0}(n)$ ισούται με το $\theta_k(n)$, που εφαρμόζεται στον νευρώνα k και η αντίστοιχη είσοδος y_0 έχει σταθερή τιμή -1.

Από την εξίσωση 37: $$\frac{\partial \upsilon_k(n)}{\partial y_j(n)} = w_{kj}(n) \qquad (55)$$

Έτσι, χρησιμοποιώντας τις εξισώσεις (53) και (55) στην (51) παίρνουμε την επιθυμητή μερική παράγωγο.

$$\frac{\partial \mathcal{E}(n)}{\partial y_j(n)} = \sum_k e_k(n)\varphi'_k(\upsilon_k(n))w_{kj}(n) = \sum_k \delta_k(n)w_{kj}(n) \qquad (56)$$

Εδώ έχουμε χρησιμοποιήσει τη σχέση $\delta_k(n) = e_k(n)\, \varphi'_k(\upsilon_k(n))$ (δηλ. τη σχέση (47)), όπου ο k είναι νευρώνας εξόδου.

Τελικά χρησιμοποιώντας την εξίσωση (56) στην (48), παίρνουμε την τοπική κλίση δj(n), για ένα κρυφό νευρώνα j, αφού επαναδιατάξουμε τους όρους, ως εξής:

$$\delta_j(n) = \varphi'_j(\upsilon_j(n))\sum_k \delta_k(n)w_{kj}(n) \quad \text{όπου ο νευρώνας j είναι κρυφός} \qquad (57)$$

Ο παράγοντας $\varphi_j'(\upsilon_j(n))$ στην (57) εξαρτάται αποκλειστικά από την συνάρτηση ενεργοποίησης που σχετίζεται με τον κρυφό νευρώνα j.

Ο παράγοντας του απομένει στο υπολογισμό του δj(n), δηλαδή ο $\Sigma\, \delta_k(n)w_{kj}(n)$, δηλαδή η πρόσθεση γιά όλα τα k, εξαρτάται από δύο σύνολα όρων :

- Το πρώτο σύνολο όρων, το $\delta_k(n)$, απαιτεί γνώση των σημάτων λάθους $e_k(n)$, για όλους εκείνους τους νευρώνες που βρίσκονται στο αμέσως δεξιό επίπεδο, από τον

κρυφό νευρώνα j, οι οποίοι είναι άμμεσα συνδεδεμένοι με το νευρώνα j (όπως στο σχήμα 12).

- Το δεύτερο σύνολο όρων, το w_{kj} (n), αποτελείται από τα συναπτικά βάρη που σχετίζονται μ' αυτές τις συνδέσεις.

Μπορούμε τώρα να συνοψίσουμε τις σχέσεις, που έχουμε παράγει για τον αλγόριθμο Πίσω-Διάδοσης.

Πρώτον, η διόρθωση $\Delta w_{ji}(n)$ που εφαρμόζεται στο συναπτικό βάρος, που συνδέει τον νευρώνα i στο νευρώνα j καθορίζεται από τον δέλτα κανόνα:

$$\begin{pmatrix} \text{Weight} \\ \text{correction} \\ \Delta w_{ji}(n) \end{pmatrix} = \begin{pmatrix} \text{learning} \\ \text{parameter} \\ \eta \end{pmatrix} \bullet \begin{pmatrix} \text{local} \\ \text{gradient} \\ \delta_j(n) \end{pmatrix} \bullet \begin{pmatrix} \text{input signa} \\ \text{of neuron j} \\ y_i(n) \end{pmatrix} \quad (58)$$

ή

$$\Delta w_{ji}(n) = \eta\, \delta_j(n) \cdot y_i(n)$$

Δεύτερον, η τοπική κλίση $\delta_j(n)$ εξαρτάται από το εάν ο νευρώνας j είναι ένας κόμβος εξόδου ή ένας κρυφός κόμβος.

1. Εάν ο νευρώνας j είναι ένας κόμβος εξόδου, $\delta_j(n)$ ισούται με το γινόμενο της παραγώγου $\varphi_j'(\upsilon_j(n))$ και του σήματος λάθους $e_j(n)$ και τα δύο εκ των οποίων σχετίζονται με το νευρώνα (βλέπε εξίσωση (48)).

Δηλαδή αν ο j είναι κόμβος εξόδου, τότε: $\delta_j(n) = e_j(n)\, \varphi_j'(\upsilon_j(n))$

2. Εάν ο νευρώνας j είναι ένας κρυφός νευρώνας το $\delta_j(n)$ ισούται με το γινόμενο της σχετιζόμενης παραγωγού $\varphi_j'(\upsilon_j(n))$ και του ζυγισμένου αθροίσματος (weighted sum) των δ, που υπολογίζονται για τους νευρώνες, στο επόμενο κρυφό ή επίπεδο εξόδου, που είναι συνδεδεμένοι στον j (βλέπε εξίσωση (57)).

Δηλαδή, αν ο νευρώνας j είναι κρυφός κόμβος τότε**:**

$$\delta_j(n) = \varphi_j'(\upsilon_j(n)) \sum_k \delta_k(n) w_{kj}(n)$$

3.3.3. Τα δύο περάσματα του υπολογισμού.

Στην εφαρμογή του αλγόριθμου BP, μπορούμε να διακρίνουμε δύο ξεχωριστά περάσματα του υπολογισμού. Το πρώτο περάσμα αναφέρεται σαν προς τα εμπρός (forward pass) και το δεύτερο σαν προς τα πίσω (backward pass).

Στο **forward pass** (προς τα εμπρός πέρασμα) τα συναπτικά βάρη παραμένουν αμετάβλητα, μέσα στο δίκτυο και τα λειτουργικά σήματα του δικτύου υπολογίζονται σε μια νευρώνα – προς – νευρώνα βάση.

Συγκεκριμένα, το λειτουργικό σήμα που εμφανίζεται στην έξοδο του νευρώνα j υπολογίζεται ως εξής:

$$y_j(n) = \varphi(\upsilon_j(n)) \tag{59}$$

όπου $\upsilon_j(n)$ είναι το εσωτερικό επίπεδο ενεργοποίησης του νευρώνα j, που ορίζεται από τη σχέση:

$$\upsilon_j(n) = \sum_{i=1}^{p} w_{ji}(n) y_i(n) \tag{60}$$

όπου p είναι ο συνολικός αριθμός εισόδων (εξαιρουμένου του κατωφλιού ενεργοποίησης), που εφαρμόζονται στον νευρώνα j και $w_{ji}(n)$ είναι το συναπτικό βάρος (της σύνδεσης), που συνδέει τον νευρώνα i στον νευρώνα j και $y_i(n)$ είναι το σήμα εξόδου του νευρώνα j ή ισοδύναμα, το λειτουργικό σήμα που εμφανίζεται στην έξοδο του νευρώνα i.

Αν ο νευρώνας j βρίσκεται στο πρώτο κρυφό επίπεδο του δικτύου, τότε ο δείκτης i αναφέρεται στο I-οστό άκρο εισόδου του δικτύου, για το οποίο γράφουμε:

$$y_i(n) = x_i(n) \tag{61}$$

όπου $x_i(n)$ είναι το I-οστό στοιχείο του διανύσματος εισόδου (pattern).

Αν απ' την άλλη μεριά, ο j βρίσκεται στο επίπεδο εξόδου του δικτύου, ο δείκτης j αναφέρεται στο j-στό άκρο της εξόδου του δικτύου, για το οποίο γράφουμε:

$$y_j(n) = o_j(n) \tag{62}$$

όπου $o_j(n)$ είναι το j-οστό στοιχείο του διανύσματος εξόδου (pattern).

Αυτή η έξοδος συγκρίνεται με την επιθυμητή απόκριση dj(n), παρέχοντας το σήμα λάθους $e_j(n)$ για το j-στό νευρώνα. Ετσι, η ***προς τα εμπρός φάση*** του υπολογισμού,

ξεκινά στο πρώτο κρυφό επίπεδο προσφέροντάς του το διάνυσμα εξόδου και τερματίζει στο επίπεδο εξόδου υπολογίζοντας το σήμα λάθους για κάθε νευρώνα αυτού του επιπέδου. Το ***προς τα πίσω πέρασμα***, από την άλλη μεριά, ξεκινά στο επίπεδο εξόδου, περνώντας τα σήματα λάθους προς τα αριστερά μέσω του δικτύου, επίπεδο προς επίπεδο και υπολογίζοντας το δ (δηλαδή την τοπική κλίση) επαναληπτικά, για κάθε νευρώνα. Αυτή η επαναληπτική διαδικασία, επιτρέπει στα συναπτικά βάρη του δικτύου, να υφίστανται αλλαγές (μεταβολές) σε σύμφωνα με τον κανόνα Δέλτα, δηλαδή την εξίσωση (58).

Για ένα νευρώνα τοποθετημένο στο επίπεδο εξόδου, το δ είναι απλά, ίσο με το σήμα λάθους γι' αυτόν το νευρώνα πολλαπλασιασμένο με την πρώτη παράγωγο της μη-γραμμικότητάς του. Επομένως, χρησιμοποιούμε την εξίσωση (58) για να υπολογίσουμε τις αλλαγές στα βάρη όλων των συνδέσεων, που τροφοδοτούν το (καταλήγουν στο) επίπεδο εξόδου..

Αφού έχουμε υπολογίσει τα δ για τους νευρώνες του επιπέδου εξόδου, στη συνέχεια χρησιμοποιούμε την εξίσωση (59) για να υπολογίσουμε τα δ για όλους τους νευρώνες στο προτελευταίο επίπεδο και επομένως τις αλλαγές στα βάρη για όλες τις συνδέσεις που καταλήγουν σ' αυτό. Ο επαναληπτικός υπολογισμός συνεχίζεται, επίπεδο προς επίπεδο, διαδίδοντας τις αλλαγές, που έγιναν σε όλα τα συναπτικά βάρη.

Σημειώστε ότι για την παρουσίαση κάθε εκπαιδευτικού παραδείγματος (training example), το πρότυπο εισόδου είναι σταθερό καθ' όλη την διάρκεια της διαδικασίας ταξιδιού μετ' επιστροφής, συμπεριλαμβάνοντας το προς τα εμπρός πέρασμα, ακολουθούμενο από το προς τα πίσω πέρασμα.

3.3.4 Σιγμοειδής μη-γραμμικότητα

Ο υπολογισμός του δ για κάθε νευρώνα του perceptron πολλών επιπέδων απαιτεί την γνώση της παραγώγου της συνάρτησης ενεργοποίησης του αντίστοιχου νευρώνα. Για να υπάρχει αυτή η παράγωγος απαιτείται, η συνάρτηση να είναι συνεχής. Σε βασικές γραμμές, η παραγωγισημότητα είναι η μόνη συνθήκη που πρέπει να ικανοποιεί μια συνάρτηση ενεργοποίησης. Ένα παράδειγμα παραγωγίσιμων συνεχών μη γραμμικών συναρτήσεων ενεργοποίησης, οι οποίες χρησιμοποιούνται συχνά στα πολυεπίπεδα

perceptrons είναι οι σιγμοειδείς μη γραμμικές συναρτήσεις. Μια τέτοια συνάρτηση, η οποία έχει οριστεί εδώ για τον νευρώνα j είναι η λογιστική συνάρτηση (logisitic function).

$$y_j(n)= \varphi_j(\upsilon_j(n))$$

$$= \frac{1}{1+\exp(-\upsilon_j(n))} \qquad -\infty<\upsilon_j(n)<\infty \qquad (63)$$

όπου $\upsilon_j(n)$ είναι η τιμή ενεργοποίησης του νευρώνα j. Εδώ η τιμή της εξόδου, βρίσκεται στη περιοχή $0<y_j<1$. Μια άλλη τυπική σιγμοειδής συνάρτηση είναι η υπερβολική εφαπτομένη της οποίας οι τιμές εξόδου βρίσκονται στη περιοχή $-1<y_j<1$. Παραγωγίζοντας τις δυο πλευρές της εξισώση (63) με το $\upsilon_j(n)$, έχουμε:

$$\frac{\partial y_j(n)}{\partial \upsilon_j(n)} = \varphi'(\upsilon_j(n))$$

$$= \frac{\exp(-\upsilon_j(n))}{\left[1+\exp(-\upsilon_j(n))\right]^2} \qquad (64)$$

Χρησιμοποιώντας την εξίσωση (63) για να απαλείψουμε τον ενθετικό όρο $\exp(-\upsilon_j(n))$ από την σχέση (64), μπορούμε να εκφράσουμε την παράγωγο $\varphi'_j(\upsilon_j(n))$ σαν:

$$\varphi'_j(\upsilon_j(n))=y_j(n)[1- y_j(n)] \qquad (65)$$

Για κάθε νευρώνα j ο οποίος ανήκει στο επίπεδο εξόδου, ισχύει ότι

$$y_j(n)=o_j(n) \qquad (66)$$

Επομένως μπορούμε να εκφράσουμε την τοπική κλίση για τον νευρώνα j σαν :

$$\delta_j(n)=e_j(n)\varphi'_j(u_j(n))$$

$$=[d_j(n) - o_j(n)]o_j(n)[1 - y_j(n)] \quad \text{ο νευρώνας j ανήκει στο επίπεδο εξόδου} \qquad (67)$$

όπου $o_j(n)$ είναι το λειτουργικό σήμα στην έξοδο του νευρώνα j, και $d_j(n)$ η επιθυμητή απόκριση. Ενώ για έναν νευρώνα j ο οποίος βρίσκεται σε ένα κρυφό επίπεδο, μπορούμε να εκφράσουμε την τοπική κλίση ως εξής:

$$\delta_j(n)=\varphi'_j(\upsilon_j(n))\sum\delta_k(n)w_{kj}$$

$$= y_j(n)[1 - y_j(n)] \sum_k \delta_k(n) w_{kj}(n) \quad \text{ο νευρώνας j ανήκει στο κρυφό επίπεδο} \quad (68)$$

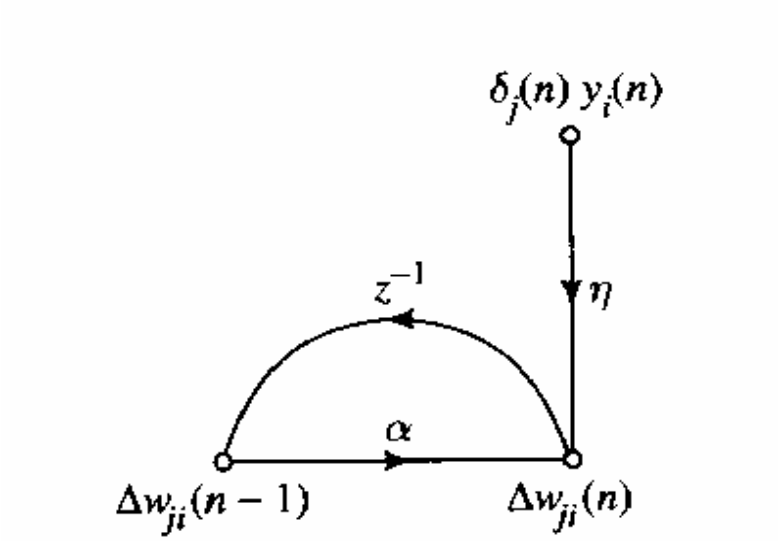
Σημειώστε ότι στην εξίσωση (65) η παράγωγος $\varphi'_j(n)$ έχει μέγιστη τιμή για $y_j(n)=0.5$ και την ελάχιστη (μηδέν) για $y_j(n)=1.0$. Αφού η αλλαγή των συναπτικών βαρών του δικτύου είναι ανάλογη της παραγώγου $\varphi'_j(u_j(n))$ συμπεραίνουμε ότι για μια σιγμοειδή συνάρτηση ενεργοποίησης τα συναπτικά βάρη θα τροποποιηθούν πιο πολύ για αυτούς τους νευρώνες, οι οποίοι έχουν τα λειτουργικά σήματα να βρίσκονται στη μέση τιμή τους. Σύμφωνα με τον Rummelhatt [1] είναι αυτή η ιδιότητα της μάθησης πίσω διάδοσης η οποία συνεισφέρει στη σταθερότητα του στον αλγόριθμο μάθησης.

3.3.5 Ρυθμός μάθησης

Ο αλγόριθμος πίσω-διάδοσης δίνει μια προσέγγιση της φθίνουσας καμπύλης προς το ελάχιστο στο υπερεπίπεδο του χώρου των βαρών, χρησιμοποιώντας της μέθοδο της απότομης μεταβολής. Όσο πιο μικρή είναι η παράμετρος μάθησης η τόσο πιο μικρές θα είναι οι αλλαγές στα βάρη του δικτύου σε κάθε επανάληψη του συνόλου εκπαίδευσης, και τόσο πιο ομαλή θα είναι η φθίνουσα καμπύλη στο υπερεπίπεδο, δηλαδή έχουμε πιο σταθερή σύγκλιση. Όμως όλες αυτές οι βελτιώσεις έχουν σαν αντίτιμο πιο αργό ρυθμό μάθησης. Αν όμως η παράμετρος μάθησης γίνει πολύ μεγάλη, έτσι ώστε να επιταχύνει τον ρυθμό μάθησης, οι μεγάλες αλλαγές που θα υπάρξουν στα βάρη μπορεί να δημιουργήσουν ένα ασταθές δίκτυο. Μια απλή μέθοδος να αυξήσουμε το ρυθμό μάθησης και ταυτόχρονα να αποφύγουμε τον κίνδυνο της αστάθειας, είναι να τροποποιήσουμε τον δέλτα κανόνα της εξίσωσης (45) συμπεριλαμβάνοντας ένα όρο ορμής (momentum term):

$$\Delta w_{ji}(n) = \alpha \Delta w_{ji}(n-1) + \eta \delta_j(n) y_i(n) \quad (69)$$

όπου α είναι συνήθως ένας θετικός αριθμός που ονομάζεται σταθερά ορμής (momentum constant), και καθορίζει πόσο μεγάλη είναι η αλλαγή του βάρους στον επόμενο υπολογισμό, όπως φαίνεται στο σχήμα 13, όπου z^{-1} είναι ένα μοναδιαίο στοιχείο καθυστέρησης.



Σχήμα 13 Γραφική παράσταση που δείχνει την επίδραση της σταθεράς ορμής α

Η εξίσωση (69) ονομάζεται ***Γενικευμένος Δέλτα Κανόνας (Generalized Delta Rule)***, απ΄όπου ο Δέλτα Κανόνας (Delta Rule) προκύπτει σαν ειδική περίπτωση, με α=0.
Για να δούμε την επίδραση των ακολουθιών από παρουσιάσεις προτύπων, πάνω στα βάρη μέσω της σταθεράς α, θα εκφράσουμε την σχέση (69) σαν χρονοσειρά με δείκτη το t. Ο δείκτης t παίρνει τιμές από 0 έως η. Τη σχέση (69) μπορούμε να την δούμε σαν διαφορική εξίσωση πρώτης τάξης του παράγοντα διόρθωσης βάρους $\Delta w_{ji}(n)$, άρα έχουμε:

$$\Delta w_{jt}(n) = -\eta \sum_{t=0}^{n} \alpha^{n-\tau} \delta_j(t) y_t(t) \qquad (70)$$

η οποία είναι μια σειρά μεγέθους n+1. Από τις σχέσεις (45), (47) συμπεραίνουμε ότι το γινόμενο $\delta_j(n)y_i(n)$ ισούται με $-\partial E(n)/\partial w_{ji}(n)$. Επομένως μπορούμε να ξαναγράψουμε την σχέση (70) ως εξής:

$$\Delta w_{ji}(n) = -\eta \sum_{t=0}^{n} \alpha^{n-t} \frac{\partial E(t)}{\partial w_{ji}(t)} \qquad (71)$$

Βασισμένοι σε αυτή τη σχέση μπορούμε να κάνουμε τις εξής παρατηρήσεις.

1. Ο παράγοντας διόρθωσης βάρους $\Delta w_{ji}(n)$ αποτελείται από άθροισμα εκθετικών όρων. Για να συγκλίνει αυτή η σειρά πρέπει $0<|\alpha|<1$. Όταν $\alpha=0$ 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. Όταν η μερική παράγωγος $\partial \mathbf{E}(t)/\partial w_{ji}(t)$ έχει το ίδιο πρόσημο σε συνεχόμενες επαναλήψεις , το εκθετικό άθροισμα $\Delta w_{ji}(n)$ αυξάνει, και έτσι το βάρος $w_{ji}(n)$ τροποποιείται κατά μεγάλο ποσό. Συνεπώς η συμπερίληψη της ορμής στον αλγόριθμο πίσω διάδοσης επιταχύνει σε περιπτώσεις που υπάρχουν σταθερές φθίνουσες κατευθύνσεις.
3. Όταν η μερική παράγωγος $\partial \mathbf{E}(t)/\partial w_{ji}(t)$ έχει αντίθετο πρόσημο σε διαδοχικές επαναλήψεις, το εκθετικό άθροισμα $\Delta w_{ji}(n)$ ελαττώνεται και έτσι το βάρος $w_{ji}(n)$ τροποποιείται κατά ένα μικρό πόσο. Συνεπώς η συμπερίληψη της ορμής στον αλγόριθμο πίσω διάδοσης σταθεροποιεί σε περιπτώσεις που υπάρχουν ταλαντώσεις πρόσημων.

Η συμπερίληψη της ορμής στον αλγόριθμο πίσω διάδοσης 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 τερματισμό της διαδικασίας σε ένα τοπικό ελάχιστο, το οποίο δεν είναι το ολικό ελάχιστο.

Πρόσθετες παρατηρήσεις.

Αναλύοντας τον αλγόριθμο πίσω διάδοσης υποθέσαμε ότι η παράμετρος μάθησης είναι σταθερά και την ονομάσαμε σαν $\boldsymbol{\eta}$. Στην πραγματικότητα θα έπρεπε να οριστεί ως $\boldsymbol{\eta_{ji}}$ δηλαδή η παράμετρος μάθησης θα έπρεπε να είναι διαφορετική ανάλογα με την σύνδεση. Πράγματι πολλά ενδιαφέροντα πράγματα θα μπορούσαν να γίνουν, κάνοντας την παράμετρο μάθησης διαφορετική για διάφορα μέρη του δικτύου.

Είναι ακόμα αξιοσημείωτο ότι στον αλγόριθμο πίσω διάδοσης μπορούμε να ορίσουμε όλα τα βάρη να τροποποιούνται ή μερικά από αυτά να παραμένουν σταθερά κατά τη διάρκεια της πίσω διάδοσης. Σε αυτή την περίπτωση τα σήματα λάθους διαδίδονται πίσω στο δίκτυο, αλλά τα βάρη δεν τροποποιούνται. Αυτό μπορεί να επιτευχθεί θέτοντας την παράμετρο μάθησης η_{ji} για το σημαντικό βάρος w_{ji} ίση με μηδέν.

Άλλο ένα ενδιαφέρον σημείο είναι ο τρόπος με το οποίο τα διάφορα επίπεδα του δικτύου είναι συνδεδεμένα. Στην ανάπτυξη του αλγορίθμου πίσω διάδοσης ο οποίος

παρουσιάστηκε εδώ, εμείς βασιστήκαμε πάνω στην υπόθεση ότι κάθε νευρώνας ενός επιπέδου δέχεται εισόδους από νευρώνες του προηγούμενου επιπέδου όπως φαίνεται στο σχήμα 9. Όμως δεν υπάρχει λόγος για τον οποίο ένας νευρώνας να μην δέχεται εισόδους από μονάδες άλλων, προηγουμένων επιπέδων. Στον χειρισμό ενός τέτοιου νευρώνα πρέπει να λάβουμε υπόψη δυο σήματα λάθους: (1) ένα σήμα λάθους το οποίο προκύπτει από την σύγκρουση του σήματος εξόδου με την επιθυμητή απόκριση και (2) ένα σήμα λάθους το οποίο διαδίδεται μέσα από τα άλλες μονάδες στις οποίες επιδρά . Σε αυτή την περίπτωση απλά προσθέτουμε τις αλλαγές στα βάρη που πρέπει να γίνουν σύμφωνα με την απευθείας σύγκριση με αυτές οι οποίες διαδίδονται πίσω από τις άλλες μονάδες.

3.3.6 Τρόποι εκπαίδευσης του δικτύου

Στο αλγόριθμο πίσω διάδοσης, η μάθηση επιτυγχάνεται εφαρμόζοντας ένα σύνολο από διανύσματα εκπαίδευσης σαν είσοδο στο πολυεπίπεδο perceptron. ***Η προβολή όλων των διανυσμάτων εκπαίδευσης στο δίκτυο λέγεται ΄κύκλος΄ (epoch)***. Η διαδικασία μάθησης προχωράει από epoch σε epoch, μέχρι να σταθεροποιηθούν τα βάρη και τα κατώφλια του δικτύου και το μέσο τετραγωνικό λάθος όλων των διανυσμάτων εκπαίδευσης τείνει σε κάποια ελάχιστη τιμή . Είναι καλή πρακτική να θέτουμε τα διανύσματα εκπαίδευσης σε μια τυχαία σειρά από ένα epoch στο άλλο . Αυτή η τυχαιότητα τείνει να κάνει το ψάξιμο στο χώρο των βαρών σε κάθε κύκλο μάθησης σε στοχαστική διαδικασία, έτσι αποφεύγουμε τον κίνδυνο να γίνουν λιγότεροι κύκλοι από ότι πρέπει.

Για ένα δεδομένο σύνολο διάδοσης μπορούμε να ακολουθήσουμε ένα από τους εξής δύο τρόπους:

1. ***Τρόπος Προτύπων (Pattern Mode)***. Στο pattern mode η τροποποίηση των βαρών γίνεται με την προβολή κάθε διανύσματος του συνόλου εκπαίδευσης. Για αυτόν τον τρόπο λειτουργίας αναλύσαμε τον αλγόριθμο πίσω διάδοσης που παρουσιάσαμε εδώ. Συγκεκριμένα κάθε epoch αποτελείται από N διανύσματα (patterns) τοποθετημένα στη σειρά [x(1),d(1),...,x(n),d(n)]. Το πρώτο διάνυσμα [x(1),d(1)] σε ένα epoch, προβάλλεται στο δίκτυο και εκτελείται η ακολουθία των προς στο μπροστά (forward) και προς τα πίσω (backward) υπολογισμών που παρουσιάστηκαν στα προηγούμενα

κεφάλαια, τροποποιώντας τα βάρη και τα κατώφλια του δικτύου. Όταν το δεύτερο διάνυσμα [x(2),d(2)] σε ένα epoch, προβάλλεται στο δίκτυο τότε όλη η ακολουθία επαναλαμβάνεται και έχει σαν αποτέλεσμα την περαιτέρω τροποποίηση των βαρών και κατωφλιών. Αυτή η διαδικασία συνεχίζεται μέχρι να δοθεί το διάνυσμα [x(n),d(n)]. Το $\Delta w_{ji}(n)$ είναι η τροποποίηση του βάρους w_{ji} μετά από την προβολή του διανύσματος n. Τότε η μέση αλλαγή του βάρους $\Delta \tilde{w}$ πάνω από όλα τα διανύσματα δίνεται από την σχέση:

$$\Delta\tilde{w} = \frac{1}{N}\sum_{n=1}^{N}\Delta w_{ji}(n)$$

$$= -\frac{\eta}{N}\sum_{n=1}^{N}\frac{\partial E(n)}{\partial w_{ji}(n)}$$

$$= -\frac{\eta}{N}\sum_{n=1}^{N}e_j(n)\frac{\partial e_j(n)}{\partial w_{ji}(n)} \qquad (72)$$

όπου στη δεύτερη γραμμή και τρίτη σειρά κάναμε χρήση των εξισώσεων (45) και (35).

2. ***Σωρηδόν Τρόπος (Batch Mode)***. Στο batch mode η τροποποίηση γίνεται μετά από την προβολή όλων των διανυσμάτων εκπαίδευσης που αποτελούν ένα epoch, στο δίκτυο. Για ένα epoch ορίζουμε την συνάρτηση κόστους (cost function) σαν το μέσο τετραγωνικό λάθος των σχέσεων (35) και (36), έχοντας εδώ πάλι αντικαταστήσει την (35) στην (36):

$$E_{av} = \frac{1}{2N}\sum_{n=1}^{N}\sum_{j\in C}e_j^2(n) \qquad (73)$$

όπου το σήμα λάθους $e_j(n)$ αντιστοιχεί στην έξοδο του νευρώνα j για το διάνυσμα εκπαίδευσης n, και ορίζεται από την σχέση (34). Το $e_j(n)$ ισούται με τη διαφορά μεταξύ των τιμών $d_j(n)$ και $y_j(n)$, οι οποίες αναπαριστούν το j-οστό στοιχείο του επιθυμητού διανύσματος d(n) και την αντίστοιχη τιμή του διανύσματος εξόδου. Στην (73) το εσωτερικό άθροισμα σε σχέση με το j συμπεριλαμβάνει όλα τα διανύσματα εκπαίδευσης σε ένα epoch. Έχοντας σαν παράμετρο μάθησης το η, η τροποποίηση στο βάρος w_{ji}, που συνδέει τον νευρώνα i με τον νευρώνα j, ορίζεται από τον δέλτα κανόνα και ισούται με:

$$\Delta w_{ji} = -\eta \frac{\partial E_{av}}{\partial w_{ji}}$$

$$= -\frac{\eta}{N} \sum_{n=1}^{N} e_j(n) \frac{\partial e_j(n)}{\partial w_{ji}} \qquad (74)$$

Για να υπολογιστεί η μερική παράγωγος $\partial e_j(n)/\partial w_{ji}$ προχωράμε όπως προηγουμένως. Σύμφωνα με την (74) στο batch mode η τροποποίηση Δw_{ji} γίνεται μόνο αφού έχει γίνει προβολή όλων των διανυσμάτων εκπαίδευσης στο δίκτυο. Συγκρίνοντας τις (72) και (74) βλέπουμε καθαρά ότι ο μέσος όρος τροποποίησης βαρών $\Delta \tilde{w}$ στο pattern mode είναι διαφορετικός από την αντίστοιχη τιμή Δw_{ji} στο batch mode. Πράγματι το $\Delta \tilde{w}$ του pattern mode είναι μια εκτίμηση του Δw_{ji} του batch mode.

Από μια "on-line" (σε πραγματικό χρόνο) οπτική γωνία το pattern mode είναι προτιμότερο ενάντια στο batch mode, επειδή χρειάζεται λιγότερη τοπική μνήμη για κάθε σύναψη. Ακόμα έχοντας σαν δεδομένο ότι τα διανύσματα προβάλλονται στο δίκτυο με μια τυχαία σειρά, η χρήση της πρότυπο προς πρότυπο τροποποίησης όπως γίνεται στο pattern mode, μετατρέπει το ψάξιμο στο χώρο των βαρών σε στοχαστική διαδικασία, η οποία δεν επιτρέπει εύκολα την παγίδευση του αλγορίθμου σε τοπικά ελάχιστα. Όμως η χρήση του batch mode δίνει μια πιο σωστή εκτίμηση του διανύσματος κλίσης. Σε τελική ανάλυση, η αποδοτικότητα των δυο τρόπων εκπαίδευσης εξαρτάται από το πρόβλημα.

3.3.7 Κριτήρια τερματισμού

Ο αλγόριθμος πίσω-διάδοσης γενικά δεν συγκλίνει ούτε υπάρχουν σαφώς ορισμένα κριτήρια για να σταματούν την λειτουργία του, αλλά έχει λογικά κριτήρια που μπορούν να χρησιμοποιηθούν για να τερματίσουν της ρυθμίσεις τον βαρών. Για να διαμορφώσουμε ένα τέτοιο κριτήριο το λογικό είναι να σκεφτούμε σε σχέση με τις μοναδικές ιδιότητες του τοπικού ή ολικού ελαχίστου της επιφάνειας λάθους. Έστω ότι το διάνυσμα βαρών **w*** δηλώνει ένα ελάχιστο, τοπικό ή ολικό. Για να είναι το **w*** ένα ελάχιστο πρέπει το διάνυσμα κλίσης **g(w)** της επιφάνειας λάθους σε σχέση με το διάνυσμα βαρών **w** να είναι μηδέν όταν **w = w*.** Αντίστοιχα, μπορούμε να

διατυπώσουμε ένα λογικό κριτήριο σύγκλισης για την μάθηση πίσω-διάδοσης όπως παρακάτω (Kramer and Sangiovanni-Vincentelli, 1989 [1]):

- *Ο αλγόριθμος πίσω-διάδοσης συγκλίνει όταν η Ευκλείδεια νόρμα του διανύσματος κλίσης φτάσει σ΄ένα αρκετά μικρό κατώφλι κλίσης.*

Το μειονέκτημα από αυτό το κριτήριο σύγκλισης είναι ότι ο χρόνος μάθησης μπορεί να είναι μεγάλος και χρειάζεται το υπολογισμό του διανύσματος κλίσης **g(w)**.

Άλλη μοναδική ιδιότητα ενός ελαχίστου είναι ότι η συνάρτηση κόστους ή μέτρο λάθους $\mathbf{E}_{av}\mathbf{(w)}$ είναι στάσιμο στο σημείο $\mathbf{w=w^*}$ και μπορούμε ως εκ τούτου να προτείνουμε ένα διαφορετικό κριτήριο σύγκλισης:

- *Ο αλγόριθμος πίσω-διάδοσης συγκλίνει όταν ο απόλυτος ρυθμός μεταβολής στο μέσο τετραγωνικό λάθος ανά κύκλο είναι αρκετά μικρός.*

Τυπικά, ο ρυθμός της μεταβολής στο μέσο τετραγωνικό λάθος θεωρείται αρκετά μικρός εάν βρίσκεται στο διάστημα 0.1 έως 1 % ανά κύκλο εκπαίδευσης , ενώ μερικές φορές χρησιμοποιείται μια τιμή που είναι αρκετή μικρή έως 0.01 % ανά κύκλο.

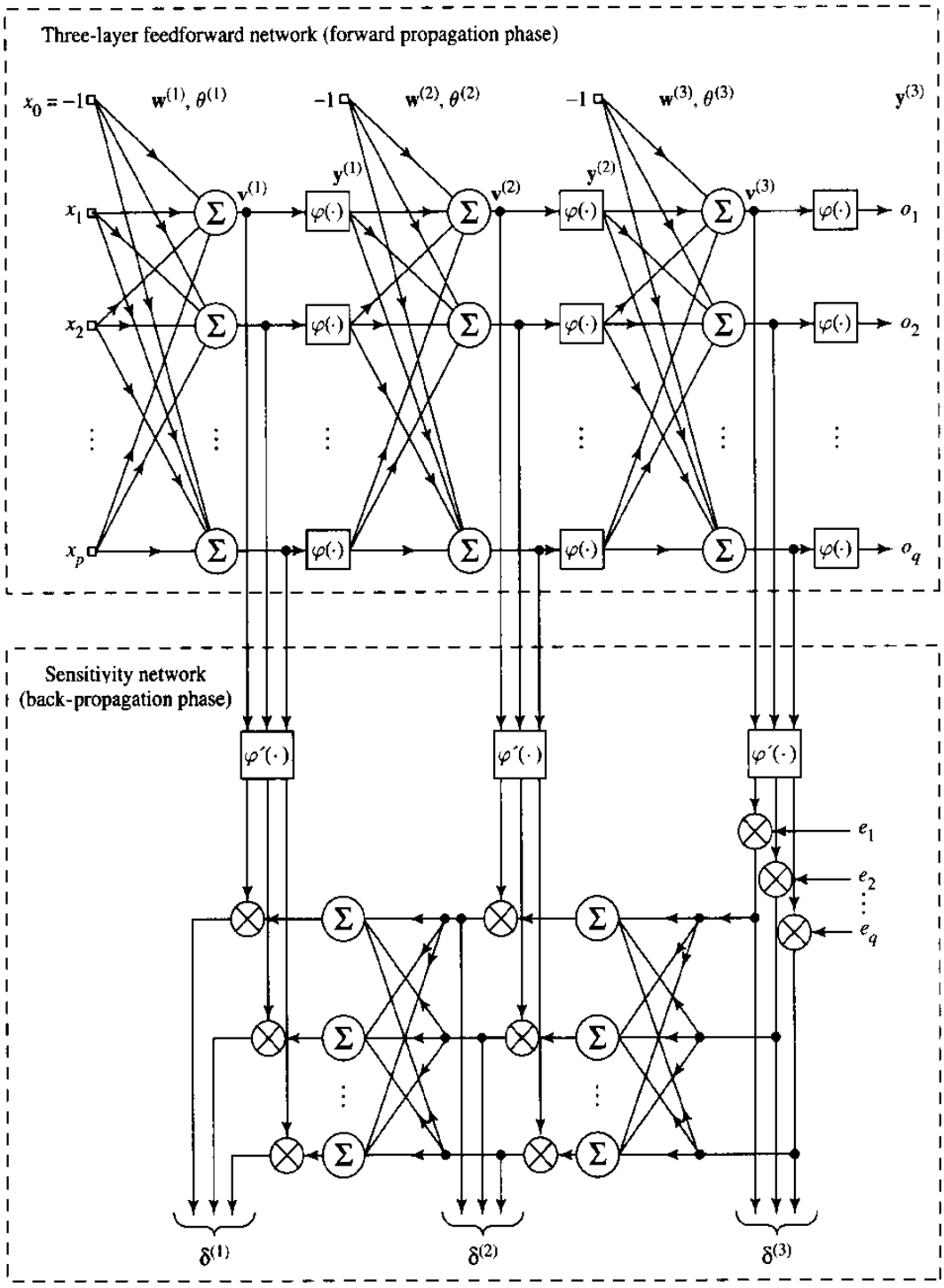
Ένα άλλο κριτήριο σύγκλισης του αλγορίθμου, παραλλαγή του προηγούμενου ,είναι να απαιτούμε η μέγιστη τιμή του μέσου τετραγωνικού λάθους $\mathbf{E}_{av}\mathbf{(w)}$ να είναι ίση ή μικρότερη από ένα αρκετά μικρό κατώφλι . Οι Kramer και Sangiovanni-Vincentelli (1989) πρότειναν ένα υβριδικό κριτήριο σύγκλισης που συνίσταται απ' αυτό το τελευταίο κατώφλι και ένα κατώφλι κλίσης, όπως δηλώνεται παρακάτω:

- *Ο αλγόριθμος πίσω-διάδοσης τερματίζεται στο διάνυσμα βαρών* $\boldsymbol{w_{final}}$ *όταν* $\|\boldsymbol{g(w_{final})}\| <= \boldsymbol{\varepsilon}$*, όπου* $\boldsymbol{\varepsilon}$ *είναι ένα αρκετά μικρό κατώφλι κλίσης, ή όταν* $\boldsymbol{E_{av}(w_{final})} <= \boldsymbol{\tau}$*, όπου* $\boldsymbol{\tau}$ *είναι ένα αρκετά μικρό κατώφλι ενέργειας λάθους.*

Άλλο χρήσιμο κριτήριο σύγκλισης είναι το παρακάτω. Μετά από κάθε επανάληψη μάθησης, το δίκτυο δοκιμάζεται για την γενική του απόδοση, και αν η γενική απόδοση είναι αρκετή ή έχει κορυφωθεί τότε σταματάμε την διαδικασία μάθησης.

3.3.8 Σύνοψη του αλγορίθμου Πίσω-Διάδοσης

Στο σχήμα 8 παρουσιάσαμε το αρχιτεκτονικό πλάνο από ένα πολυεπίπεδο perceptron. Η αντίστοιχη αρχιτεκτονική για την πίσω-διάδοσης μάθηση παρουσιάζεται στο σχήμα 14, που ενσωματώνει τις προς τα εμπρός και προς τα πίσω φάσεις των υπολογισμών που περιλαμβάνονται στη διαδικασία μάθησης. Το πολυεπίπεδο δίκτυο που δείχνουμε στο πάνω μέρος του σχήματος δείχνει την προς τα εμπρός φάση.



Σχήμα 14: Αρχιτεκτονικό διάγραμμα ενός τριών επιπέδων feedforward δικτύου

και ένα συνεργαζόμενο δίκτυο με τα σήματα πίσω-διάδοσης λάθους.

Η σημειογραφία που χρησιμοποιήθηκε σ' αυτό το κομμάτι του σχήματος είναι η παρακάτω:

$\mathbf{w}^{(l)}$ = διάνυσμα συναπτικών βαρών (synaptic weight vector) από το νευρώνα στο επίπεδο l

$\boldsymbol{\theta}^{(l)}$ = κατώφλι ενός νευρώνα στο επίπεδο l

$\mathbf{v}^{(l)}$ = διάνυσμα των εσωτερικών επιπέδων δραστηριότητας του δικτύου (net internal activity levels) των νευρώνων στο επίπεδο l

$\mathbf{y}^{(l)}$ = διάνυσμα από σήματα λειτουργίας των νευρώνων στο επίπεδο l

Ο δείκτης επιπέδου l εκτείνεται από το επίπεδο εισόδου ($l = 0$) μέχρι το επίπεδο εξόδου ($l = L$). Στο σχήμα 14 έχουμε το L = 3, το L είναι το βάθος του δικτύου. Το κατώτερο κομμάτι του σχήματος δείχνει τη πίσω φάση, που αναφέρεται ως ένα δίκτυο ευαισθησίας (sensitivity) για υπολογισμό των τοπικών κλίσεων στον αλγόριθμο πίσω-διάδοσης. Η σημειογραφία που χρησιμοποιήθηκε σ' αυτό το δεύτερο κομμάτι του σχήματος είναι η παρακάτω:

$\boldsymbol{\delta}^{(l)}$ = διάνυσμα των τοπικών κλίσεων των νευρώνων στο επίπεδο l

$\mathbf{e}$ = διάνυσμα λάθους (error vector) που συμβολίζεται από τα e1,e2, . . . ,eq σαν στοιχεία

Έχουμε αναφέρει ότι η pattern by pattern μέθοδος ενημέρωσης των βαρών προτιμάται για on-line υλοποίηση του αλγορίθμου πίσω-διάδοσης. Για αυτόν τον τρόπο λειτουργίας, ο αλγόριθμος κάνει κύκλους διαμέσου των δεδομένων εκπαίδευσης $\{[\mathbf{x}(n),\mathbf{d}(n)]; n = 1, 2, \ldots, N\}$ όπως παρακάτω:

1. *Αρχικοποίηση*. Ξεκίνα με μία λογική διαμόρφωση του δικτύου, και θέσε σ' όλα τα συναπτικά βάρη και τα επίπεδα κατωφλίου του δικτύου μικρούς τυχαίους αριθμούς που είναι ομοιόμορφα κατανεμημένοι.

2. *Παρουσίαση Παραδειγμάτων Εκπαίδευσης*. Παρουσίασε στο δίκτυο ένα κύκλο από

παραδείγματα εκπαίδευσης. Για κάθε παράδειγμα ταξινομημένο με κάποιο τρόπο μέσα στο σύνολο, εκτέλεσε την παρακάτω σειρά από μπρος και πίσω υπολογισμούς του 3 και 4 αντίστοιχα.

3. *Εμπρός Υπολογισμός*. Έστω ότι ένα παράδειγμα εκπαίδευσης του κύκλου δηλώνεται από το [**x**(n),**d**(n)], με το διάνυσμα εισόδου **x**(n) να εφαρμόζεται στο επίπεδο εισόδου των αισθητήρων κόμβων και το επιθυμητό διάνυσμα απόκρισης **d**(n) να παρουσιάζεται στο επίπεδο εξόδου των κόμβων υπολογισμού. Υπολόγισε τα επίπεδα ενεργοποίησης και τα λειτουργικά σήματα του δικτύου, προχωρώντας προς τα εμπρός δια μέσω του δικτύου, επίπεδο ανά επίπεδο. Το εσωτερικό επίπεδο ενεργοποίησης του δικτύου $\mathbf{v}(l)$ για το νευρώνα j στο επίπεδο *l* είναι:

$$u_j^{(l)}(n) = \sum_{i=0}^{p} w_{ji}^{l}(n) y_i^{(l-1)}(n)$$

όπου $y_i^{(l-1)}$ (n) είναι το λειτουργικό σήμα του νευρώνα i στο προηγούμενο επίπεδο *l*-1 στην επανάληψη n και το $w_{ji}^{(l)}$ είναι το συναπτικό βάρος του νευρώνα j στο επίπεδο *l* που τροφοδοτείται από το νευρώνα j στο επίπεδο *l*-1. Για *i* = 0, έχουμε $y_0^{(l-1)}$ (n) = -1 και $w_{j0}^{(l)}$ (n) = $\theta_j^{(l)}$(n), όπου το $\theta_j^{(l)}$ είναι το κατώφλι που εφαρμόζεται στη νευρώνα j στο επίπεδο *l*. Υποθέτοντας τη χρησιμοποίηση μιας λογιστικής συνάρτησης για τη σιγμοειδή μη γραμμικότητα, το λειτουργικό σήμα (έξοδος) από το νευρώνα j στο επίπεδο *l* είναι:

$$y_j^{(l)}(n) = \frac{1}{1+e^{(-\upsilon_j^{(l)}(n))}}$$

Αν ο νευρώνας j είναι στο πρώτο κρυμμένο επίπεδο (δηλαδή, *l*=1) βάλε:

$y_j^{(0)}$ (n) = x_j(n)

όπου x_j(n) είναι το j-στό στοιχείο του διανύσματος εισόδου **x**(n). Αν ο νευρώνας j είναι στο επίπεδο εξόδου, βάλε:

$y_j^{(L)}$ (n) = o_j(n)

Ετσι, υπολόγισε το σήμα λάθους:

$e_j(n) = d_j(n) - o_j(n)$

όπου $d_j(n)$ είναι το j-στό στοιχείο του επιθυμητού διανύσματος απόκρισης **d**(n).

4. *Πίσω Υπολογισμός.* Υπολόγισε τα δ (δηλαδή τις τοπικές κλίσεις) του δικτύου, προχωρώντας προς τα πίσω , επίπεδο ανά επίπεδο:

$\delta_j^{(L)}(n) = e_j^{(L)}(n)\, o_j(n)\left[1 - o_j(n)\right]$ για το νευρώνα j στο επίπεδο εξόδου L

$\delta_j^{(l)}(n) = y_j^{(l)}(n)\left[1 - y_j^{(l)}(n)\right]\sum_k \delta_j^{(l+1)}(n) w_{kj}^{(l+1)}(n)$ για το νευρώνα j στο κρυφό επίπεδο *l*

Έτσι, ρύθμισε τα συναπτικά βάρη του δικτύου στο επίπεδο *l* με βάση τον γενικό κανόνα δέλτα:

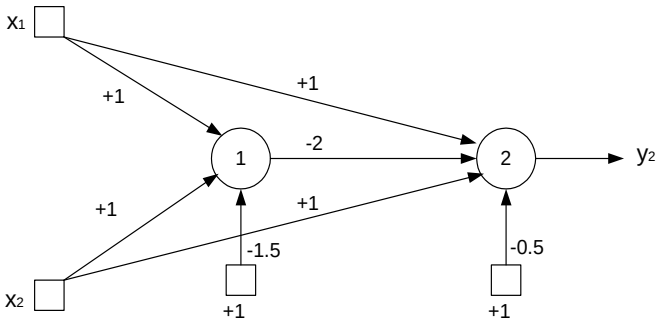
$w_{ji}^{(l)}(n+1) = w_{ji}^{(l)}(n) + \delta\left[w_{ji}^{(l)}(n) - w_{ji}^{(l)}(n-1)\right] + \eta\delta_j^{(l)}(n) y_i^{(l-1)}(n)$

όπου η είναι ο ρυθμός μάθησης και α είναι η σταθερά ορμής.

5. *Επανάληψη.* Επανέλαβε τον υπολογισμό παρουσιάζοντας νέους κύκλους παραδειγμάτων εκπαίδευσης μέχρι που οι ελεύθερες παράμετροι του δικτύου σταθεροποιούν και το μέσο τετραγωνικό λάθος $\mathbf{E}_{av}(\mathbf{w})$ που υπολογίσαμε πάνω στο συνολικό set εκπαίδευσης να είναι σε ένα ελάχιστο ή σε μια αποδεκτά μικρή τιμή. Η σειρά παρουσίασης των παραδειγμάτων εκπαίδευσης θα πρέπει να είναι τυχαία από κύκλο σε κύκλο. Καθώς αυξάνεται ο αριθμός των επαναλήψεων μάθησης η ορμή και ο ρυθμός μάθησης μεταβάλλονται τυπικά (και συνήθως μειώνονται).

Άσκηση αυτοαξιολόγησης 3.3 / 8:

Στο παρακάτω σχήμα φαίνεται ένα νευρωνικό δίκτυο για την επίλυση του XOR προβλήματος. Δείξτε ότι όντως το δίκτυο επιλύει το XOR πρόβλημα κατασκευάζοντας τα (α) τις περιοχές απόφασης και (β) έναν πίνακα αλήθειας του δικτύου.



Απάντηση:
Υποθέτουμε ότι κάθε νευρώνας ακολουθεί το μοντέλο McCulloch – Pitts.

(α) Για να κατασκευάσουμε τις περιοχές απόφασης εργαζόμαστε όπως στην παράγραφο που περιγράφει το XOR πρόβλημα. Ετσι κατασκευάζουμε πρώτα τις περιοχές απόφασης που σχηματίζει ο νευρώνας 1 και στη συνέχεια τις περιοχές που σχηματίζει ο νευρώνας 2 που είναι και οι περιοχές απόφασης του δικτύου.

(β)Για τον νευρώνα 1 έχουμε:

$$u_1 = x_1 + x_2 - 1.5$$

Ετσι μπορούμε να κατασκευάσουμε τον ακόλουθο πίνακα:

x_1	0	0	1	1
x_2	0	1	0	1
u_1	-1.5	-0.5	-0.5	0.5
y_1	0	0	0	1

Για τον νευρώνα 2 έχουμε:

$$u_2 = x_1 + x_2 - 2 \cdot y_1 - 0.5$$

x_1	0	0	1	1
x_2	0	1	0	1
y_1	0	0	0	1
u_2	-0.5	0.5	0.5	-0.5
y_2	0	1	1	0

Από τον πίνακα αυτό παρατηρούμε ότι η έξοδος του δικτύου y_2 είναι 0 εάν x_1 και x_2 είναι 0 ή 1, και 1 αν κάποιο από τα δύο είναι 0 και το άλλο 1. Άρα το δίκτυο έχει μάθει τη συνάρτηση XOR.

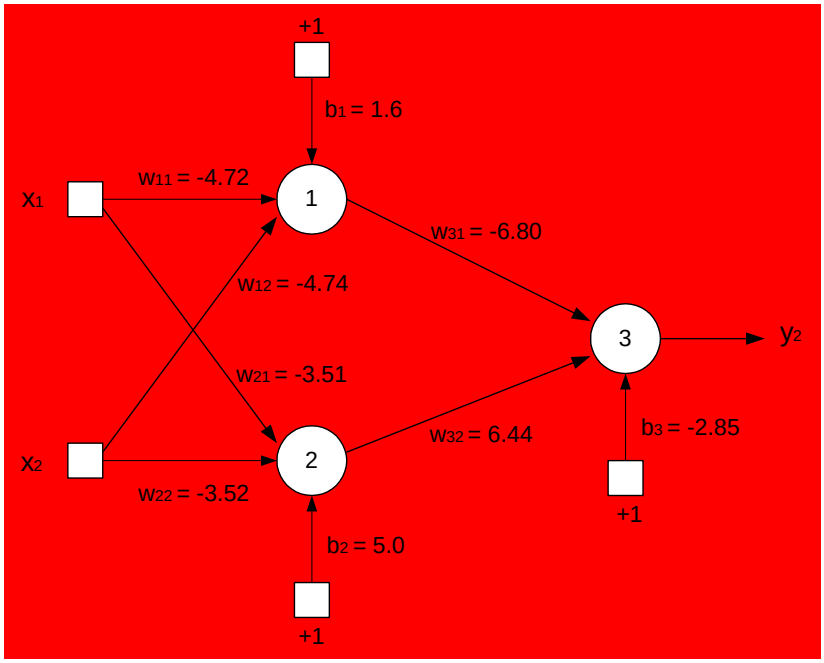
Άσκηση αυτοαξιολόγησης 3.3 / 9:

Χρησιμοποιήστε τον Αλγόριθμο Πίσω Διάδοσης του Λάθους για να υπολογίσετε ένα σύνολο από πολώσεις (bias) και βάρη για ένα δίκτυο με δύο εισόδους, δύο κρυφούς νευρώνες και μία έξοδο, προκειμένου να επιλύει το XOR πρόβλημα. Σαν συνάρτηση ενεργοποίησης χρησιμοποιήστε την λογιστική απεικόνιση.

Απάντηση:

Επειδή γενικά η λογιστική συνάρτηση παίρνει τιμές από 0.1 έως 0.9 σαν επιθυμητή απόκριση του δικτύου θα χρησιμοποιήσουμε την 0.1 για την περίπτωση του 0 και την 0.9 για την περίπτωση του 1. Σαν αρχικές τιμές στα βάρη και στις πολώσεις δίνουμε τιμές στο διάστημα (-1.0, 1.0).

Τρέχοντας τον Αλγόριθμο Πίσω Διάδοσης του Λάθους για 2500 επαναλήψεις προκύπτει το παρακάτω δίκτυο που έχει μάθει τη συνάρτηση XOR. Εδώ θα πρέπει να σημειώσουμε ότι οι τιμές των ελευθέρων παραμέτρων (βάρη και πολώσεις) του δικτύου που προέκυψε είναι ενδεικτικές και οτι δεν σημαίνει πως οι δικές σας θα είναι αναγκαία ίδιες με αυτές.



Άσκηση αυτοαξιολόγησης 3.3 / 10:

Στην παράμετρο ορμής δίνουμε συνήθως θετικές τιμές στο διάστημα [0, 1). Ερευνήστε

τι επίδραση θα είχε στη συμπεριφορά της εξίσωσης

$$\Delta w_{ji}(n) = -\eta \sum_{t=0}^{n} a^{n-t} \frac{\partial \mathrm{E}(t)}{\partial w_{ji}(t)}$$

η χρήση αρνητικών τιμών για το momentum α από το διάστημα (-1, 0].

Απάντηση:

Αν το momentum α πάρει αρνητικές τιμές θα έχουμε:

$$\Delta w_{ji}(n) = -\eta \sum_{t=0}^{n} a^{n-t} \cdot \frac{\partial \mathrm{E}(t)}{\partial w_{ji}(t)}$$
$$-\eta \sum_{t=0}^{n} (-1)^{n-t} \cdot |a|^{n-t} \cdot \frac{\partial \mathrm{E}(t)}{\partial w_{ji}(t)}$$

Από την παραπάνω εξίσωση μπορούμε να δούμε ότι αν το $\partial E(t) / \partial w_{ji}(t)$ έχει το ίδιο πρόσημο σε συνεχόμενες επαναλήψεις του αλγορίθμου το μέγεθος του $\Delta w_{ji}(n)$ ελαττώνεται. Το αντίθετο ισχύει εάν το $\partial E(t) / \partial w_{ji}(t)$ αλλάζει πρόσημο σε διαδοχικές επαναλήψεις του αλγορίθμου. Αρα η επίδραση του momentum α είναι η αντίθετη από αυτή που έχει όταν παίρνει θετικές τιμές.

Άσκηση αυτοαξιολόγησης 3.3 / 11:

Θεωρήστε ένα απλό παράδειγμα δικτύου με ένα βάρος και συνάρτηση κόστους:

$$E(w) = k_1(w-w_0)^2 + k_2$$

Όπου τα w_0, k_1 και k_2 είναι σταθερές. Για την ελαχιστοποίηση του E(w) χρησιμοποιούμε τον αλγόριθμο Πίσω Διάδοσης του Λάθους με παράμετρο ορμής. Ερευνήστε πως η χρήση ορμής επηρεάζει τη διαδικασία μάθησης με ιδιαίτερη αναφορά στον αριθμό των κύκλων (epochs) που απαιτούνται για σύγκληση συναρτήσει του αλγορίθμου.

Απάντηση:

Από τη θεωρία γνωρίζουμε ότι:

$$\Delta w_{ji}(n) = -\eta \sum_{t=1}^{n} a^{n-t} \frac{\partial \mathrm{E}(t)}{\partial w_{ji}(t)} \qquad (1)$$

Στην περίπτωση που έχουμε ένα μόνο βάρος η συνάρτηση κόστους είναι:

$$E(w) = k_1(w-w_0)^2 + k_2 \qquad (2)$$

Οπότε η εξίσωση (1) λόγω της (2) γίνεται:

$$\Delta w(n) = -2 \cdot k_1 \cdot \eta \sum_{t=1}^{n} a^{n-t} \cdot (w(t) - w_0) \qquad (3)$$

Στην περίπτωση αυτή η μερική παράγωγος $\partial E(t)/\partial w(t)$ έχει το ίδιο πρόσημο σε διαδοχικές επαναλύψεις. Επομένως για $0 \le \alpha < 1$ το $\Delta w(n)$ αυξάνει σε μέγεθος. Αυτό σημαίνει ότι το βάρος διορθώνεται με μεγάλες ποσότητες. Αρα η χρήση του momentum α στην εξίσωση έχει ως συνέπεια την επιτάχυνση της «καθόδου» προς το βέλτιστο.

3.4 Σύνοψη κεφαλαίου

Για να χρησιμοποιηθεί ένα ΤΝΔ πρέπει πρώτα να εκπαιδευτεί για να μάθει. Η μάθηση συνίσταται στον προσδιορισμό των κατάλληλων συντελεστών βάρους, ώστε το Τ.Ν.Δ. να εκτελεί τους επιθυμητούς υπολογισμούς, και πραγματοποιείται με τη βοήθεια αλγορίθμων που είναι γνωστοί ως κανόνες μάθησης ή εκπαίδευσης. Ο ρόλος των συντελεστών βάρους μπορεί να ερμηνευτεί ως αποθήκευση γνώσης, η οποία παρέχεται μέσω παραδειγμάτων. Με αυτόν τον τρόπο τα Ν.Δ. μαθαίνουν το περιβάλλον τους, δηλαδή το φυσικό μοντέλο που παρέχει τα δεδομένα.

Σε αυτό το κεφάλαιο παρουσιάσαμε τους τρεις βασικούς αλγορίθμους μάθησης (εκπαίδευσης) Ν.Δ.. Αρχίσαμε την παρουσίαση, με τον αλγόριθμο εκπαίδευσης του απλού Perceptron (Αισθητήρα) και το θεώρημα της σύγκλισής του. Ακολούθησε η παρουσίαση του αλγορίθμου Ελάχιστου Μέσου Τετραγωνικού (Ε.Μ.Τ.) λάθους, για την εκπαίδευση ενός απλού Ν.Δ.. Για την απόδειξη του αλγορίθμου, δανειστήκαμε ιδέες από τη λύση του γραμμικού προβλήματος φιλτραρίσματος. Παρουσιάσαμε πρώτα τις εξισώσεις των Wiener-Hopf και στη συνέχεια τις δύο μεθόδους επίλυσής τους. Αυτές είναι η μέθοδος Ταχύτερης Καθόδου και η μέθοδος του Ελάχιστου Μέσου Τετραγωνικού λάθους. Είδαμε ότι με τη χρήση του αλγορίθμου LMS, παρακάμπτουμε το πρόβλημα της εκ των προτέρων γνώσης των χωρικών συναρτήσεων συσχέτισης,

χρησιμοποιώντας τις εκτιμήσεις τους. Έτσι ο αλγόριθμος LMS προκύπτει από ένα απλό και συγχρόνως αποδοτικό τρόπο υπολογισμού αυτών των εκτιμήσεων.

Τέλος, παρουσιάσαμε το βασικό αλγόριθμο εκπαίδευσης για δίκτυα εμπρός τροφοδότησης πολλών επιπέδων, που είναι γνωστά σαν Perceptrons πολλών επιπέδων. Ο αλγόριθμος εκπαίδευσης αυτών των δικτύων είναι ο πολύ δημοφιλής αλγόριθμος Πίσω Διάδοσης (Π.Δ.) του λάθους, ο οποίος βασίζεται στον υπολογισμό της κλίσης της συνάρτησης λάθους στην έξοδο του δικτύου και τον προς τα πίσω υπολογισμό των τοπικών κλίσεων. Κατά την παραγωγή του αλγορίθμου, είδαμε ότι διακρίνουμε δύο κατηγορίες νευρώνων, τους νευρώνες που είναι στο επίπεδο εξόδου και τους νευρώνες που είναι στα κρυφά επίπεδα. Αν και η παραγωγή του αλγορίθμου είναι αρκετά πολύπλοκη, ο ίδιος ο αλγόριθμος είναι εύκολο να υλοποιηθεί και έχει τύχει ευρείας εφαρμογής σε πολλά πρακτικά προβλήματα. Έγινε επίσης η παρουσίαση μιας τροποποίησης του αλγορίθμου Π.Δ., που είναι γνωστή σαν Γενικευμένος Δέλτα κανόνας. Με αυτόν τον αλγόριθμο επιτυγχάνεται η αύξηση του ρυθμού μάθησης, ενώ εξασφαλίζεται η σύγκλιση του αλγορίθμου. Η ενότητα αυτή ολοκληρώθηκε με την παρουσίαση των δύο τρόπων υπολογισμού, δηλαδή την κατεργασία ανά πρότυπο και την σωρηδόν κατεργασία. Σε κάθε τρόπο λειτουργίας του αλγορίθμου, είδαμε τα πλεονεκτήματα και τα μειονεκτήματά του. Τέλος, το κεφάλαιο ολοκληρώθηκε με την συζήτηση των κριτηρίων τερματισμού της εκτέλεσης του αλγορίθμου. Για περισσότερες λεπτομέρειες σχετικά με τα παραπάνω θέματα, ο αναγνώστης παραπέμπεται στην αναφορά [1]. Για πρακτική εξάσκηση, στους αλγορίθμους εκπαίδευσης, μπορεί να χρησιμοποιηθεί το δεύτερο βιβλίο το οποίο περιέχει εκτελέσιμο κώδικα, για τους αλγορίθμους που παρουσιάσαμε σε αυτό το κεφάλαιο.

Συνοψίζοντας, μπορούμε να πούμε ότι σκοπός αυτού του κεφαλαίου ήταν η παρουσίαση των βασικών αλγορίθμων εκπαίδευσης τόσο απλών όσο και πολυεπίπεδων Τ.Ν.Δ.. Εδώ πρέπει να αναφέρουμε ότι η τιμή της παραμέτρου μάθησης έχει ιδιαίτερη σημασία για τη σύγκλιση του αλγορίθμου. Επίσης, ο αλγόριθμος Π.Δ. του λάθους, απαιτεί να είναι εκ των προτέρων γνωστή η τοπολογία του δικτύου. Αυτό όμως αποτελεί ένα βασικό πρόβλημα των Τ.Ν.Δ., για το οποίο δεν έχει ακόμη δοθεί μια γενική λύση. Αυτά τα θέματα όμως θα τα συζητήσουμε αναλυτικά στο επόμενο κεφάλαιο, στα πλαίσια μιας εφαρμογής.

3.5 Βιβλιογραφία

1. “NEURAL NETWORKS: A Comprehensive Foundation’’, S. Haykin, Macmillan Publishing Company, N.Y., 1994 (ISBN 0-02-352761-7)

2. ‘’NEURAL NETWORK DESIGN’’, M. T. Hagan, H. B. Demuth and m. Beal, PWS Publishing Company, Boston, 1996 (ISBN 0-534-94332-2)