Τμήμα Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ Τ.Ε

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Υλοποίηση FPGA ενός Δυαδικού Νευρωνικού Δικτύου

Αναγνωστοπούλου Μαρία, 2394

Επιβλέποντες: Παρασκευάς Κίτσος, Επίκουρος Καθηγητής

Περιεχόμενα

1.0 Ιστορική αναδρομή

Τεχνητή ή υπολογιστική νοημοσύνη ονομάζεται ένα σύνολο από ενέργειες που εκτελούνται από υπολογιστές οι οποίες θυμίζουν την ανθρώπινη νοημοσύνη. Μπορεί στο άκουσμα της πολλοί να φανταζόμαστε υπερσύγχρονα ανθρωποειδή και ρομπότ του μέλλοντος. Όμως η πραγματικότητα είναι αρκετά διαφορετική. Η ιδέα για μηχανές με νοημοσύνη ξεκινάει ήδη από το 1950. Όπου μεταξύ άλλων, ο Alan Turing, στο άρθρο του “Computing Machinery and Intelligence” κάνει λόγο για μηχανές που μαθαίνουν και θέτει ερωτήματα όπως το αν μπορούν οι μηχανές να σκεφτούν. Στο ίδιο άρθρο παρουσιάζει και το διάσημο “Παιχνίδι της Μίμησης”.

Το παιχνίδι της μίμημης χρειάζεται τρία άτομα, α) έναν άνδρα, β) μία γυναίκα και έναν ανακριτή ανεξαρτήτου φύλου.Οι κανόνες είναι απλοί. Ο ανακριτής δεν γνωρίζει το φύλο των α και β και δεν έχει οπτική ή ακουστική επαφή μαζί τους. Σκοπός του είναι να ανακαλύψει το φύλο τους μέσα από ένα σύνολο ερωτήσεων. Ο α έχει σκοπό να μπερδέψει τον ανακριτή, ενώ ο β θέλει να βοηθήσει να ανακαλύψει ο ανακριτής σωστά τα φύλα τους. Το ερώτημα του Turing είναι: Τι θα γινόταν αν αντικαθιστούσαμε τον έναν συμμετέχοντα με έναν υπολογιστή ικανό να δώσει γραπτές απαντήσεις. Το ποσοστό επιτυχίας του ανακριτή θα παρέμενε το ίδιο;

Στο πέρασμα του χρόνου, το παιχνίδι απλοποιήθηκε και σήμερα είναι γνωστό ως “Turing Test” και αποτελείται από έναν άνθρωπο και μία μηχανή και για πολλούς αποτελεί το κριτήριο για το επίπεδο νοημοσύνης μιας μηχανής. Η έννοια τεχνητή νοημοσύνη εδραιώθηκε ως έννοια, το 1956 στο συνέδριο “Dartmout Summer Research Project on Artificial Intelligence - (DSRPAI)” που διοργανώθηκε από τους John McCarthy και Marvin Minsky, στο πανεπιστήμιο του Darmout των Ηνωμένων Πολιτειών. Το 1970, ο Minsky δήλωσε πως μέσα σε τρία με οχτώ χρόνια θα έχόυμε το πρώτο μηχάνημα γενικής τεχνητής νοημοσύνης. Όπως φανταζόμαστε, ο Minsky έπεσε λίγο έξω και η μοίρα της τεχνητής νοημοσύνης, της επιφύλασε πολλές εκπλήξεις.

Το 1974, ο τομέας της τεχνητής νοημοσύνης κατάφερε να αποσπάσει μεγάλη χρηματοδότηση, όμως οι υπολογιστές τότε ήταν ακόμα πολύ αδύναμοι και τα αποτελέσματα που υπόσχονταν οι ερευνητές δεν κατάφεραν να πραγματοποιηθούν. Το 1974 λοιπόν, η χρηματοδότηση στα περισσότερα ερευνητικά έργα, σταμάτησε και ήρθε η εποχή που αργότερα ονομάστηκε πρώτος χειμώνας της τεχνητής νοημοσύνης. Στο μεταξύ, οι αλγόριθμοι και τα μαθηματικά μοντέλα που εφαρμόζονται στην τεχνητή νοημοσύνη εξελίσσονται και η χρηματοδότηση ξαναξεκινάει κάπου το 1980, με τους John Hopfield και David Rumelhart να δημοσιεύουν τεχνικές βαθιάς μάθησης, ή αλλιώς deep learning, οι οποίες κάνουν τους υπολογιστές να μαθαίνουν βάσει των εμπειριών τους. Το Ιαπωνικό κράτος χρηματοδότησε με τεράστια ποσά την έρευνα της τεχνητής νοημοσύνης από το 1982 έως και το 1990. Παρόλα αυτά, οι υπολογιστές δεν ήταν ακόμα με θέση να αποπληρώσουν το κόστος της έρευνας και έτσι στις αρχές του ‘90 έχουμε τον δεύτερο χειμώνα τεχνητής νοημοσύνης και διήρκεσε μέχρι το 1997.

Εκείνη τη χρονιά όμως, έχουμε και τα πρώτα ελπιδοφόρα αποτελέσματα, με την δημιουργία του πρώτου λογισμικού αναγνώρισης φωνής, και τον υπολογιστή της IBM:

Deep Blue, να κερδίζει τον πρωταθλητή του σκακιού Garry Kasparov, στο ίδιο του το παιχνίδι.Έκτοτε, οι δυνατότητες των υπολογιστών αυξάνονται με εκθετικό ρυθμό και όλο και περισσότερες εφαρμογές κάνουν χρήση τεχνικών υπολογιστικής νοημοσύνης σε πάρα πολλά επιστημονικά πεδία.

Και φτάνουμε στο σήμερα ενώ λοιπόν είμαστε ακόμα μακριά από την δημιουργία ενός συστήματος γενικής Τεχνητής Νοημοσύνης, ένα σύστημα που θα λειτουργεί χωρίς την ανθρώπινη παρέμβαση για ένα πολύ μεγάλο εύρος προβλημάτων. Τα μέχρι σήμερα συστήματα τεχνητής νοημοσύνης μας έχουν ξεπεράσει κατά πολύ στην αντιμετώπιση στοχευμένων προβλημάτων, όπως οι προβλέψεις μελλοντικών γεγονότων βάσεις στατιστικής ανάλυσης δεδομένων, αυτόνομη οδήγηση, βίντεο-παιχνίδια κτλ.

Τί είναι όμως η μηχανική μάθηση; Μηχανική μάθηση είναι ένας τρόπος με τον οποίο, ένα υπολογιστικό σύστημα μπορεί να αποκτήσει τεχνητή νοημοσύνη. Ονομάζεται μάθηση, γιατί θυμίζει τον τρόπο με τον οποίο, οι άνθρωποι μαθαίνουμε παρατηρώντας μία κατάσταση και πρακτικά έχουμε υπολογιστές που παίρνουν αποφάσεις, για τις οποίες δεν προγραμματίστηκαν επακριβώς. Η μηχανική μάθηση, έχει δείξει εξαιρετικές επιδόσεις σε προβλήματα τύπου ταξινόμησης ή ακόμα και πρόβλεψης. Καταλήγοντας, είναι εκπληκτικό το πως τεχνικές που αναπτύχθηκαν πριν δεκαετίες, σήμερα βρίσκουν διαρκώς νέες εφαρμογές, ξεκλειδώνοντας νέες δυνατότητες για τον άνθρωπο και το περιβάλλον του.

1.1 Τι είναι τα νευρωνικά δίκτυα

Τα Τεχνητά Νευρωνικά Δίκτυα αναπτύχθηκαν κατά τις τελευταίες δεκαετίες και είναι από τις πιο σημαντικές τεχνολογίες που χρησιμοποιούνται στις θετικές επιστήμες. Ένα νευρωνικό δίκτυο είναι ένα διασυνδεδεμένο συγκρότημα απλών στοιχείων επεξεργασίας, μονάδων ή κόμβων του οποίου η λειτουργικότητα βασίζεται στον βιολογικό νευρώνα. Η επεξεργαστική δυνατότητα αυτού του δικτύου βασίζεται στις δυνάμεις συνδέσεων ή αλλιώς βάρη που λαμβάνονται κάτω από συγκεκριμένα σύνολα εκπαίδευσης.

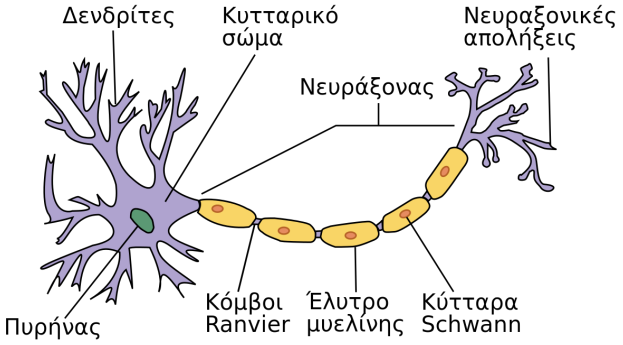
Τα τεχνητά νευρωνικά δίκτυα αποτελούν μία προσπάθεια προσέγγισης της λειτουργίας του ανθρώπινου εγκεφάλου. Ο ανθρώπινος εγκέφαλος αποτελείται από περίπου 10^11 (100 δις) νευρικά κύτταρα. Οι νευρώνες επικοινωνούν μεταξύ τους μέσω ηλεκτρικών σημάτων ή παλμών. Οι μεταξύ τους συνδέσεις προκαλούνται από ηλεκτροχημικές διασταυρωσεις που ονομάζονται συνάψεις, οι οποίες βρίσκονται στα κλαδιά του κυττάρου που ονομάζονται δενδρίτες.

Σκοπός των Τεχνητών Νευρωνικών Δικτύων είναι να δειχθεί αν τα μοντέλα που βασίζονται σε αυτές τις απλές αρχές μπορούν να εκτελέσουν τους υπολογισμούς που εκτελεί ο ανθρώπινος εγκέφαλος. Δηλαδή, αν μπορούν να μεταδώσουν πληροφορίες σχετικά με τα ερεθίσματα που λαμβάνουν.

Τα δίκτυα εκπαιδεύονται ώστε να αναγνωρίζουν και να επιτελούν μία συγκεκριμένη διεργασία. Η εκπαίδευση τους γίνεται με την αλλαγή των τιμών των βαρών τους. Έχουν πετύχει εντυπωσιακά αποτελέσματα αλλά έχουν και κάποιους περιορισμούς. Οι περιορισμοί δημιουργούνται συχνά όταν το μέγεθος και η πολυπλοκότητα του συστήματος αυξάνουν. Τα νευρωνικά δίκτυα δεν είναι αποτελεσματικά σε υπολογισμούς όπου η αριθμητική ακρίβεια είναι σημαντικός παράγοντας.

1.2 Μοντέλο Βιολογικού Νεύρωνα

Νεύρωνας*:* Η δομική μονάδα του εγκεφάλου
Σώμα*:* Ο πυρήνας του νεύρωνα
Δενδρίτες*:* Μεταφέρουν ηλεκτρικά σήματα από γειτονικούς νεύρωνες (σημεία εισόδου)
Αξονας*:* Υπό κατάλληλες συνθήκες, εξάγει ηλεκτρικά σήματα προς γειτονικούς νεύρωνες.



Στο άκρο κάθε δενδρίτη υπάρχει ένα απειροελάχιστο κενό που ονομάζεται ***σύναψη.*** Η ικανότητα μάθησης και μνήμης που παρουσιάζει ο εγκέφαλος οφείλεται στην ικανότητα των συνάψεων να μεταβάλλουν την αγωγιμότητα τους. Τα σήματα που εισέρχονται στο σώμα μέσω των δενδριτών, συνδυάζονται και αν το αποτέλεσμα ξεπερνά κάποιο κατώφλι, διαδίδεται μέσω του άξονα προς άλλους νεύρωνες.

1.3 Η αντιστοιχία των Τεχνητών Νευρωνικών Δικτύων με τα βιολογικά δίκτυα

Η έμπνευση για κάθε νευρωνικό δίκτυο ξεκινά από την βιολογία. Οι ζώντες οργανισμοί, από τους πιο απλούς μέχρι τον άνθρωπο, έχουν ένα νευρικό σύστημα το οποίο είναι υπεύθυνο για μία πλειάδα από διεργασίες, όπως είναι η επαφή με τον εξωτερικό κόσμο, η μάθηση, η μνήμη, κ.τ.λ. Η κεντρική μονάδα του νευρικού συστήματος είναι ο εγκέφαλος ο οποίος επίσης αποτελείται από νευρωνικά δίκτυα. Κάθε νευρωνικό δίκτυο αποτελείται από έναν μεγάλο αριθμό μονάδων που λέγονται νεύρωνες ή νευρώνια (neurons). Ο νεύρωνας είναι μία μικρή ανεξάρτητη ομάδα του δικτύου όπως π.χ το άτομο είναι η πιο μικρή μονάδα της ύλης. Οι νεύρωνες συνεχώς

και ασταμάτητα επεξεργάζονται πληροφορίες, παίρνοντας και στέλνοντας ηλεκτρικά σήματα σε άλλους νεύρωνες.

Οι διεργασίες που επιτελούνται από τα βιολογικά νευρωνικά δίκτυα στους ζώντες οργανισμούς είναι πολύ περίπλοκες αλλά και τόσο χρήσιμες στην καθημερινή ζωή του ανθρώπου. Μερικές από αυτές είναι εργασίες ρουτίνας, τις οποίες ο ανθρώπινος εγκέφαλος εκτελεί με ελάχιστη ή μηδαμινή προσπάθεια όπως για παράδειγμα η αναγνώριση μιας εικόνας.

Πολλά απ' τα πιο απλά πράγματα, όπως η αναγνώριση φωνής ή εικόνας που το μυαλό κάνει πολύ εύκολα, οι υπολογιστές δεν μπορούν εύκολα να τα κάνουν με επιτυχία. Και αυτό βέβαια δεν οφείλεται στην έλλειψη ταχύτητας, καθ΄ότι οι υπολογιστές είναι χιλιάδες φορές γρηγορότεροι απ΄τον ανθρώπινο εγκέφαλο. Ο λόγος είναι στην διαφορά της δομής μεταξύ τους. Αυτή η διαφορά έχει οδηγήσει στο να γίνουν κάποιες πρώτες σκέψεις μήπως είναι δυνατόν να δημιουργηθούν κάποια πρότυπα (μοντέλα) του νευρωνικού συστήματος του ανθρώπου, τα οποία θα περιέχουν όλα τα χαρακτηριστικά που είναι γνωστά μέχρι σήμερα, τα οποία από μόνα τους θα μπορούσαν να επιτελέσουν τις εργασίες αυτές με τον ίδιο τρόπο που γίνονται στα βιολογικά νευρωνικά δίκτυα. Τα δίκτυα αυτά ονομάζονται Τεχνητά Νευρωνικά Δίκτυα (Artificial Neural Networks).

Η βασική τους διαφορά από τα βιολογικά νευρωνικά δίκτυα είναι ότι αυτά τα δίκτυα παίρνουν γνώσεις με την εξάσκηση και την εμπειρία, όπως ακριβώς και οι άνθρωποι, αλλά διαφέρουν στο ότι δεν ακολουθούν ορισμένους προκαθορισμένους κανόνες που είναι χαρακτηριστικό των υπολογιστών.

Οι τεχνητοί νεύρωνες είναι συχνά απλοποιημένες εκδόσεις των βιολογικών νευρώνων και πολλοί νευροεπιστήμονες είναι σκεπτικοί σχετικά με την δύναμη αυτών "φτωχών" μοντέλων, επιμένοντας ότι υπάρχουν περισσότερες λεπτομέρειες επί της λειτουργίας του εγκεφάλου που πρέπει να εξηγηθούν. Αντλώντας γνώσεις για το πως συνδέονται οι πραγματικοί νεύρωνες, έχουν σημειωθεί σημαντικές επιπτώσεις στην μοντελοποίηση της λειτουργικότητας του εγκεφάλου.

1.4 Τι είναι ένας νευρώνας: ορισμός και λειτουργία

Οι νευρώνες είναι τα δομικά στοιχεία του δικτύου. Κάθε τέτοιος κόμβος δέχεται ένα σύνολο αριθμητικών εισόδων από διαφορετικές πηγές (είτε από άλλους νεύρωνες, είτε από το περιβάλλον), επιτελεί έναν υπολογισμό με βάση αυτές τις εισόδους και παράγει μία έξοδο. Η εν λόγω έξοδος είτε κατευθύνεται στο περιβάλλον, είτε τροφοδοτείται ως είσοδος σε άλλους νευρώνες του δικτύου. Υπάρχουν τρεις τύποι οι *νευρώνες εισόδου*, οι *νευρώνες εξόδου* και οι *υπολογιστικοί νευρώνες* ή *κρυμμένοι νευρώνες*. Οι νευρώνες εισόδου δεν επιτελούν κανέναν υπολογισμό, μεσολαβούν απλώς ανάμεσα στις περιβαλλοντικές εισόδους του δικτύου και στους υπολογιστικούς

νευρώνες. Οι νευρώνες εξόδου διοχετεύουν στο περιβάλλον τις τελικές αριθμητικές εξόδους του δικτύου. Οι υπολογιστικοί νευρώνες πολλαπλασιάζουν κάθε είσοδό τους με το αντίστοιχο *συναπτικό βάρος* και υπολογίζουν το ολικό άθροισμα των γινομένων. Το άθροισμα αυτό τροφοδοτείται ως όρισμα στη *συνάρτηση ενεργοποίησης*, την οποία υλοποιεί εσωτερικά κάθε κόμβος. Η τιμή που λαμβάνει η συνάρτηση για το εν λόγω όρισμα είναι και η έξοδος του νευρώνα για τις τρέχουσες εισόδους και βάρη.

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

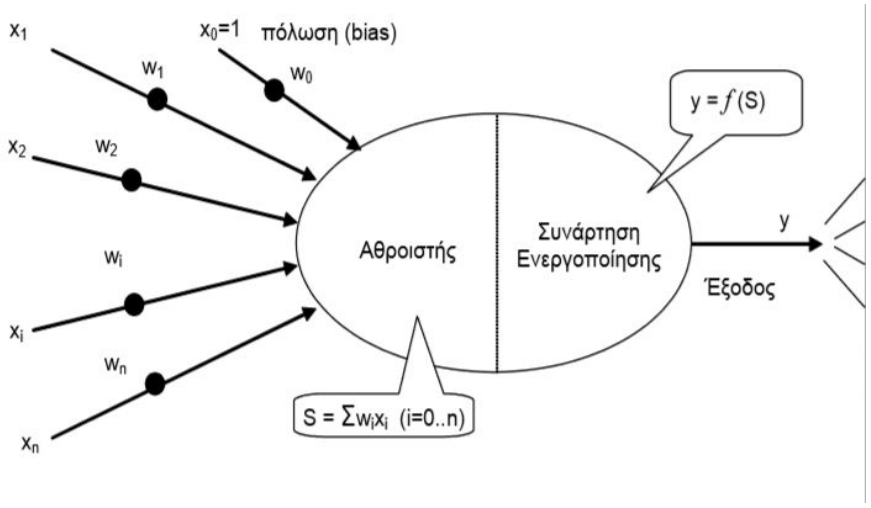
$$y_k = \phi(\sum_{i=0}^{N} x_{ki} \cdot w_{ki})$$

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

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

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

1.5 Μοντέλο Τεχνητού Νευρώνα



-Δέχεται ***σήματα εισόδου*** $x_1, x_2, \ldots, x_n$: Συνεχείς μεταβλητές (αντί ηλεκτρικών παλμών)

-Κάθε σήμα εισόδου μεταβάλλεται από μία αρνητική ή θετική ***τιμή βάρους*** w_i *(weight):* Αντίστοιχο των συνάψεων

-***Πολωση(bias)***: Ειδική περίπτωση βάρους (w_0) που επιδρά σε τιμή εισόδου $x_0 = 1$.Πρόκειται για εξωτερικό ερέθισμα. Μπορεί να παρέχει έμμεσο έλεγχο στην συνάρτηση ενεργοποίησης.

Το σώμα του τεχνητού νευρώνα αποτελείται από:

-Τον ***αθροιστή (sum)***: Προσθέτει τα επηρεασμένα από τα βάρη σήματα εισόδου και παράγει την ποσότητα S.

-Την ***συνάρτηση ενεργοποίησης ή κατωφλίου (activation ή threshold function)***: Μη γραμμικό φίλτρο που διαμορφώνει το σήμα εξόδου y, σε συνάρτηση με την ποσότητα S.

-***Έξοδος*** (έξοδος τιμή): Μπορεί να αποτελεί είσοδο σε άλλους νεύρωνες.

1.6 Συναρτήσεις ενεργοποίησης

Ξεκινώντας με την ***βηματική συνάρτηση***, αυτά που γνωρίζουμε είναι τα εξής:

-Δίνει στην έξοδο αποτέλεσμα (συνήθως 1) μόνο αν η τιμή που υπολογίζεται είναι μεγαλύτερη από την τιμή κατωφλίου.

-Χαρακτηρίζεται από δύο τιμές a και b

-Αν x<0 τότε g(x)=a και εάν x>0 τότε g(x)=b. Συνήθως χρησιμοποιούνται οι τιμές a=0 και b=1 είτε a=-1 και b=1.

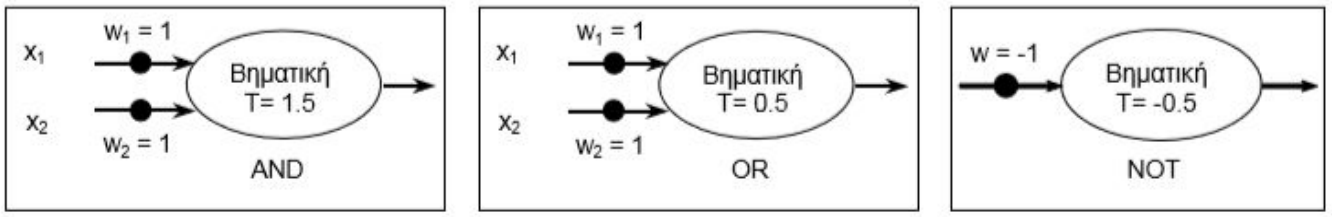
-Για x=0 υπάρχει ασυνέχεια και μπορούμε να αναθέσουμε είτε τιμή a είτε τιμή b.

-Η βηματική συνάρτηση δεν θεωρείται βολική ως συνάρτηση ενεργοποίησης των νευρώνων των τεχνητών νευρωνικών δικτύων.

Μεταβαίνοντας στις επόμενες δύο συναρτήσεις που είναι η ***λογιστική*** και η ***υπερβολική εφαπτομένη***, ανήκουν στην ομάδα των ***σιγμοειδών συναρτήσεων***. Λέγονται σιγμοειδής διότι έχουν μορφή τελικού σίγμα και αποτελούν συνεχείς και παραγωγίσιμες προσεγγίσεις της βηματικής.

Λογιστική συνάρτηση: σ(x)=1/(1+exp(-ax))
Υπερβολική εφαπτομένη: tanh(x)= e^ax-e^-ax/ e^ax+e^-ax

1.7 Υλοποίηση Λογικών Συναρτήσεων με Τεχνητό Νευρώνα

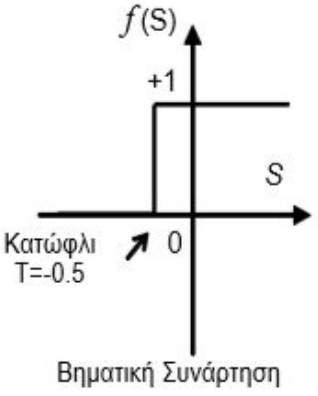


Παράδειγμα: Υλοποίηση της NOT

Σχεδίαση:

→Συνάρτηση Ενεργοποίησης: Βηματική Συνάρτηση με κατώφλι T=-0.5

→Οι τιμές εισόδου μπορούν να είναι 0 (ψευδές) ή 1 (αληθές)



Λειτουργία:

→Αν η είσοδος του νεύρωνα είναι 0 τότε πολλαπλασιαζόμενη με το βάρος w=-1 δίνει S=0. Η τιμή αυτή ξεπερνά το κατώφλι του -0.5, οπότε στην έξοδο παράγεται 1.

→Στην περίπτωση που η τιμή εισόδου είναι 1 τότε S=-1, τιμή που βρίσκεται κάτω του κατωφλίου -0.5, με αποτέλεσμα να παράγεται στην έξοδο 0.

1.8 Ο αισθητήρας Perceptron

Το μοντέλο του αισθητήρα Perceptron είναι από τα πρώτα μοντέλα νευρωνικών δικτύων που αναπτύχθηκαν και έδωσαν μεγάλη ώθηση χάρη στις επιτυχίες που είχε από την αρχή. Καθ' όσον οι γνώσεις μας για το νευρικό σύστημα του ανθρώπου προόδευαν, οι πρώτες αυτές προσπάθειες φαίνεται ότι ήταν πολύ απλοϊκές. Σήμερα υπάρχουν πολλές παραλλαγές με διαφορετικές και περίπλοκες νευρωνικές δομές. Η πιο απλή είναι ο λεγόμενος στοιχειώδης αισθητήρας (elementary perceptron) γιατί αποτελείται από ένα μόνο νευρώνα και είναι το πιο απλό αυτοδύναμο σύστημα που υπάρχει και επιτελεί μία ορισμένη διεργασία. Ο μοναδικός νεύρωνας του συστήματος έχει έναν ορισμένο αριθμό συνδέσεων που προέρχονται από άλλους νεύρωνες, τους οποίους όμως δεν εξετάζουμε. Ο νευρώνας είναι ο κύκλος και οι συνδέσεις του οι διάφορες γραμμές. Έχει ένα ορισμένο αριθμό εισόδων, αλλά μία μόνο έξοδο. Αυτό σημαίνει ότι η μονάδα αυτή δέχεται πολλές εισόδους s1, s2, s3, κ.λ.π. αλλά παράγει μία μόνο έξοδο που όπως φαίνεται στο σχήμα είναι στα δεξιά του κύκλου. Κάθε εισερχόμενο σήμα, συνδέεται με τον κεντρικό νευρώνα, το w είναι η επίδραση του εισερχόμενου σήματος με τον νευρώνα αυτό. Αυτό που έχει σημασία δεν είναι η τιμή του βάρους w από μόνη της, ούτε η τιμή του σήματος s, αλλά είναι το γινόμενο $s_i \cdot w_i$ και έτσι κάθε s_i πολλαπλασιάζεται επί το βάρος w_i που έχει η σύνδεση i και τελικά αυτό που παρουσιάζεται στον νεύρωνα από κάθε εισερχόμενο σήμα είναι το γινόμενο $s_i \cdot w_i$.

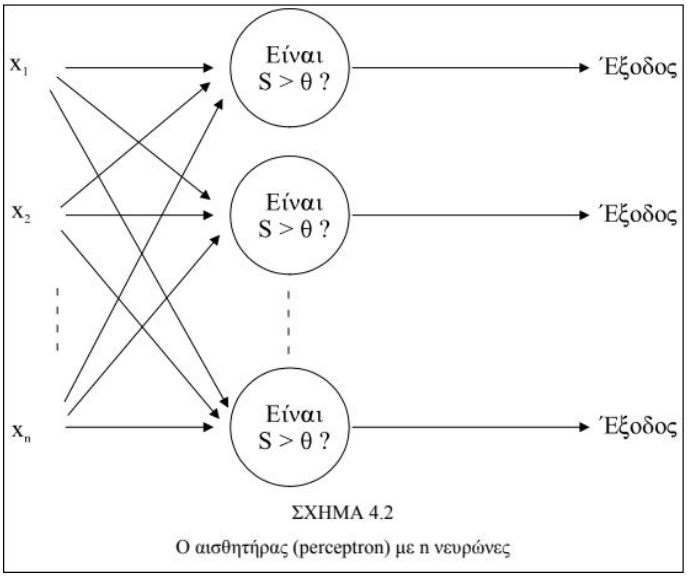
Ο αισθητήρας κατόπιν αθροίζει τα γινόμενα αυτά και θεωρούμε λοιπόν ότι λαμβάνει ένα συνολικό σήμα με τιμή:

$$\mathbf{S} = \sum_i s_i \cdot w_i$$

Ακολούθως εφαρμόζουμε την συνάρτηση κατωφλίου με ένα συγκεκριμένο κατώφλι θ. Συγκρίνουμε το θ με το άθροισμα S. Εάν S>θ, τότε ο αισθητήρας ενεργοποιείται και θεωρούμε ότι πυροδοτεί. Εάν S<θ, τότε το άθροισμα S μηδενίζεται και ο αισθητήρας παραμένει αδρανής . Αυτό συνοψίζεται ως:

Εάν S>θ	τότε έξοδος=1
Εάν S<θ	τότε έξοδος=0

Αντιλαμβανόμαστε λοιπόν ότι η ενεργητικότητα του αισθητήρα εξαρτάται από τα βάρη των συνδέσεων, τις τιμές των εισόδων και την τιμή του κατωφλίου. Θεωρούμε ότι αυτό που μαθαίνει το σύστημα μας αποθηκεύεται στα βάρη των συνδέσεων, τα οποία όπως θα δούμε, μεταβάλλονται συνεχώς κατά την διάρκεια που το σύστημα "μαθαίνει" κάποια πληροφορία.



ΣΧΗΜΑ 4.2

Ο αισθητήρας (perceptron) με n νευρώνες

Ένας αισθητήρας με πιο περίπλοκη δομή δίνεται στο σχήμα 4.2. Εδώ έχουμε n νεύρωνες, αντί για έναν μόνο που έχει ο στοιχειώδης αισθητήρας. Στην πιο γενική περίπτωση έχουμε πλήρη συνδεσμολογία, δηλαδή κάθε εισερχόμενο σήμα x_i παρουσιάζεται και στους n νεύρωνες με διαφορετικό βάρος κάθε φορά. Η διαδικασία σύγκρισης με το κατώφλι θ είναι η ίδιο όπως και στο απλό μοντέλο, αλλά εδώ έχουμε μία πλειάδα από εξόδους των οποίων ο αριθμός είναι n, όσο δηλαδή και ο αριθμός των νευρώνων. Υπάρχουν και διάφορα άλλα παρόμοια μοντέλα που ονομάζονται επίσης συλλογικά αισθητήρες και μερικά είναι πιο περίπλοκα από το παραπάνω αλλά και ο μηχανισμός λειτουργίας τους είναι ίδιος όπως αυτόν που είδαμε.

1.9 Το πρόβλημα της XOR

Ένα από τα πιο γνωστά προβλήματα των νευρωνικών δικτύων είναι το πρόβλημα της συνάρτησης XOR (exclusive-or), συνάρτηση της αποκλειστικής διάζευξης. Η συνάρτηση αυτή δέχεται δύο εισόδους και δίνει μία έξοδο. Οι είσοδοι και η έξοδος μπορει να είναι 0 ή 1 με τον εξής περιορισμό: Εάν και οι δύο είσοδοι είναι ίδιες τότε η έξοδος είναι 0, εάν είναι διαφορετικές τότε η έξοδος είναι 1.

Η συνάρτηση X-OR

Είσοδος 1	Είσοδος 2	Έξοδος
0	0	0
0	1	1
1	0	1
1	1	0

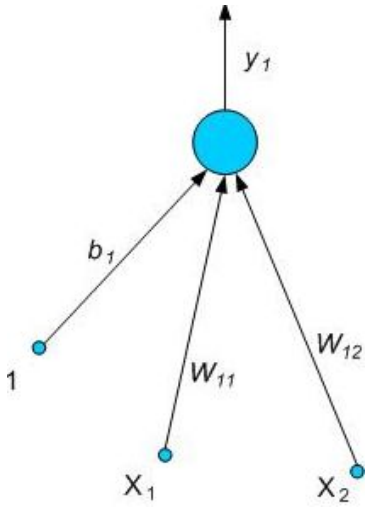
Ας φανταστούμε νεύρωνες που έχουν τα ακόλουθα χαρακτηριστικά:

-Ορίζονται σε ένα στρώμα
-Κάθε ένας έχει τα δικά του βάρη που προέρχονται από τις εισόδους x.
Αυτή η δομή των νευρώνων με τα χαρακτηριστικά αυτά σχηματίζει ένα μονοστρωματικό νευρωνικό δίκτυο. Οι παραπάνω παράμετροι καθορίζονται στην διαδικασία εκμάθησης ενός δικτύου. Αυτός ο τύπους δικτύου έχει περιορισμένες δυνατότητες. Και εδώ έρχεται το πρόβλημα με την εφαρμογή λειτουργίας XOR.
Η δυνατότητα εκμάθησης ενός νευρωνικού δικτύου ορίζεται από την γραμμική διαχωρισιμότητα των δεδομένων διδασκαλίας (μία γραμμή διαχωρίζει ένα σύνολο δεδομένων που αντιπροσωπεύει το u=1 και αυτό αντιπροσωπεύει το u=0). Αυτές οι συνθήκες πληρούνται από λειτουργίες όπως η OR ή η AND.

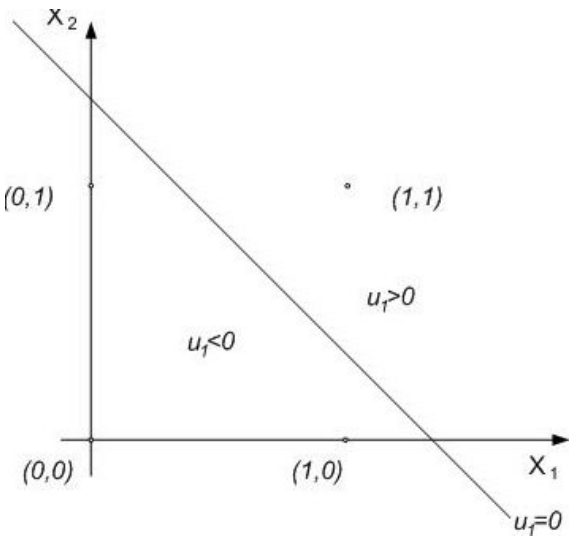
x1	x2	u
0	0	0
0	1	0
1	0	0
1	1	1

Η συνάρτηση AND

Το νευρωνικό δίκτυο που υλοποιεί μία τέτοια λειτουργία αποτελείται από έναν νεύρωμα εξόδου με δύο εισόδους x1, x2 και b1.



Κατά την διάρκεια της εκπαίδευσης έχουμε κατά νου ότι: y1= f(w11 x1 + w12 x2 + b1)= u1



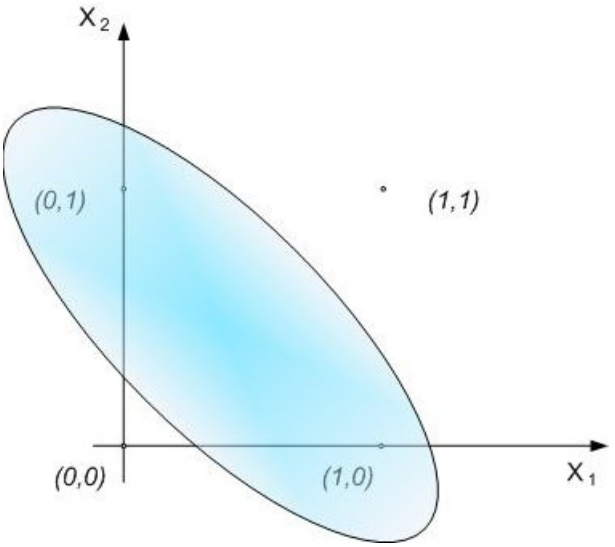
Όπως φαίνεται, θα πρέπει να λάβουμε σήμα εξόδου “1” μόνο στο σημείο (1,1). Η εξίσωση της γραμμής που υλοποιεί την γραμμική διαχωρισιμότητα είναι u1= w11x1 + w12x2 + b1. Έτσι μπορούμε να ταιριάξουμε αυτή την γραμμή και να λάβουμε γραμμικό διαχωρισμό βρίσκοντας κατάλληλους συντελεστές της γραμμής (w11, w12, ib1). Όπως μπορούμε να δούμε στο σχήμα, δεν υπάρχει πρόβλημα για την λειτουργία AND.

Η γραμμική διαχωρισιμότητα δεν μπορεί να χρησιμοποιηθεί με την λειτουργία XOR.

x1	x2	u
0	0	0
0	1	1
1	0	1
1	1	0

Η συναρτηση XOR

Αυτό σημαίνει ότι δεν είναι δυνατόν να βρεθεί μία γραμμή που χωρίζει τον χώρο δεδομένων στο διάστημα με σήμα εξόδου -0 και χώρο με σήμα εξόδου -1. Μέσα στο σχήμα της οβάλ περιοχής, η έξοδος είναι “1”. Εκτός αυτής της περιοχής το σήμα εξόδου ισούται με το “0”. Αυτό δεν είναι δυνατό να συμβεί με μία γραμμή.
Οι συντελεστές αυτής της γραμμής και τα βάρη w11, w12 και b1 δεν επηρεάζουν την αδυναμία χρήσης γραμμικού διαχωρισμού. Επομένως δεν μπορούμε να εφαρμόσουμε την λειτουργία XOR σε έναν perceptron.



1.10 Η διαδικασία εκμάθησης ενός Τεχνητού Νευρωνικού Δικτύου

Μόλις δομηθεί ένα δίκτυο για μια συγκεκριμένη εφαρμογή, το δίκτυο είναι έτοιμο να εκπαιδευτεί. Για να ξεκινήσει η διαδικασία, τα βάρη επιλέγονται τυχαία. Στην συνέχεια αρχίζει η εκπαίδευση ή αλλιώς εκμάθηση. Υπάρχουν δύο προσεγγίσεις, **υπό επίβλεψη** και **χωρίς επίβλεψη.** Η επιτηρούμενη εκπαίδευση περιλαμβάνει έναν μηχανισμό παροχής δικτύου με την επιθυμητή έξοδο είτε με χειροκίνητη “ταξινόμηση” της απόδοσης του δικτύου είτε με παροχή των επιθυμητών εξόδων με τις εισόδους. Η μη επιτηρούμενη εκπαίδευση είναι εκεί όπου το δίκτυο πρέπει να έχει υπόψη τις εισροές χωρίς εξωτερική βοήθεια.

Ο τεράστιος όγκος των δικτύων χρησιμοποιεί επιτηρούμενη εκπαίδευση. Η μη επιτηρούμενη εκπαίδευση χρησιμοποιείται για την εκτέλεση κάποιου αρχικού χαρακτηρισμού στις εισόδους.

Εκπαίδευση Υπό Επίβλεψη: Στην επιτηρούμενη εκπαίδευση, παρέχονται τόσο οι είσοδοι όσο και οι έξοδοι. Το δίκτυο επεξεργάζεται τις εισόδους και συγκρίνει τις προκύπτουσες εξόδους με τις επιθυμητές εξόδους. Στην συνέχεια, τα σφάλματα επιστρέφουν πίσω στην αρχή του δικτύου, προκαλώντας το σύστημα να προσαρμόσει τα βάρη του. Αυτή η διαδικασία λαμβάνει μέρος ξανά και ξανά και τα βάρη τροποποιούνται συνεχώς. Το σύνολο δεδομένων που ενεργοποιούν την εκπαίδευση ονομάζεται “σύνολο εκπαίδευσης”. Κατά την διάρκεια της εκπαίδευσης ενός δικτύου, το σύνολο δεδομένων επεξεργάζεται πολλές φορές καθώς τα βάρη βελτιώνονται συνεχώς.

Τα υπάρχοντα πακέτα ανάπτυξης δικτύου παρέχουν εργαλεία για τεχνητά νευρωνικά δίκτυα για την παρακολούθηση της ικανότητας πρόβλεψης της σωστής απάντησης. Αυτά τα εργαλεία επιτρέπουν την διαδικασία εκπαίδευσης να συνεχίζεται για μέρες,

σταματώντας μόνο όταν το σύστημα φτάσει σε επιθυμητό σημείο ή ακρίβεια. Ωστόσο, μερικά δίκτυα δεν μαθαίνουν ποτέ. Αυτό μπορεί να οφείλεται στο γεγονός ότι τα δεδομένα εισόδου δεν περιέχουν τις συγκεκριμένες πληροφορίες από τις οποίες προέρχεται η επιθυμητή έξοδος. Τα δίκτυα επίσης, δεν συγκλίνουν εάν δεν υπάρχουν αρκετά δεδομένα για να επιτρέψουν την πλήρη μάθηση. Στην ιδανική περίπτωση, θα πρέπει να υπάρχουν αρκετά δεδομένα ώστε ένα μέρος των δεδομένων να μπορεί να συγκρατηθεί για δοκιμή. Πολλά πολυεπίπεδα δίκτυα με πολλαπλούς κόμβους είναι ικανά να απομνημονεύουν δεδομένα. Για να παρακολουθηθεί το δίκτυο για να διαπιστωθεί εάν το σύστημα απομνημονεύει τα δεδομένα του με κάποιο τρόπο, η επιτηρούμενη εκπαίδευση πρέπει να συγκρατήσει ένα σύνολο δεδομένων που θα χρησιμοποιηθεί για την δοκιμή του συστήματος μετά την εκπαίδευσή του.

Εάν ένα δίκτυο δεν μπορεί να λύσει το πρόβλημα, τότε ο σχεδιαστής πρέπει να αναθεωρήσει τις εισόδους και τις εξόδους, των αριθμό των στρώσεων, τον αριθμό των στοιχείων ανά στρώμα, τις συνδέσεις μεταξύ των επιπέδων, τις λειτουργίες άθροισης, μεταφοράς και εκπαίδευσης, ακόμα και τα ίδια τα βάρη. Αυτές οι αλλαγές που απαιτούνται για την δημιουργία ενός επιτυχημένου δικτύου, αποτελούν μια διαδικασία που ονομάζεται “τέχνη” της νευρωνικής δικτύωσης.

Υπάρχουν πολλοί αλγόριθμοι που χρησιμοποιούνται για την εφαρμογή της προσαρμοστικής ανάδρασης που απαιτείται για της προσαρμογή των βαρών κατά την διάρκεια της εκπαίδευσης. Η πιο συνηθισμένη τεχνική είναι η διάδοση του σφάλματος προς τα πίσω (backpropagation algorithm). Ωστόσο, η εκπαίδευση δεν είναι απλώς μία τεχνική. Περιλαμβάνει μία συνειδητή ανάλυση για να διασφαλίσει ότι το δίκτυο δεν είναι πιο εκπαιδευμένο απ ότι θα έπρεπε. Αρχικά, ένα τεχνητό νευρωνικό δίκτυο διαμορφώνεται με τις γενικές στατιστικές τάσεις των δεδομένων. Αργότερα, συνεχίζει να “μαθαίνει” για άλλες πτυχές των δεδομένων που μπορεί να είναι ψευδείς από γενική άποψη.

Όταν τελικά το σύστημα έχει εκπαιδευτεί σωστά και δεν απαιτείται περαιτέρω μάθηση, τα βάρη μπορούν, εάν είναι επιθυμητό να “παγώσουν”. Σε ορισμένα συστήματα, αυτό το οριστικοποιημένο δίκτυο μετατρέπεται σε υλικό έτσι ώστε να μπορεί να είναι γρήγορο. Άλλα συστήματα δεν κλειδώνουν, αλλά συνεχίζουν να μαθαίνουν και να χρησιμοποιούνται στην παραγωγή.

Εκπαίδευση Χωρίς Επίβλεψη: Ο άλλος τύπος εκπαίδευση ονομάζεται εκπαίδευση χωρίς επίβλεψη. Σε μη επιτηρούμενη εκπαίδευση, το δίκτυο διαθέτει εισόδους αλλά όχι με τις επιθυμητές εξόδους. Το ίδιο το σύστημα πρέπει να αποφασίσει ποια χαρακτηριστικά θα χρησιμοποιήσει για να ομαδοποιήσει τα δεδομένα εισόδου. Αυτό συχνά αναφέρεται ως αυτοοργάνωση ή προσαρμογή.

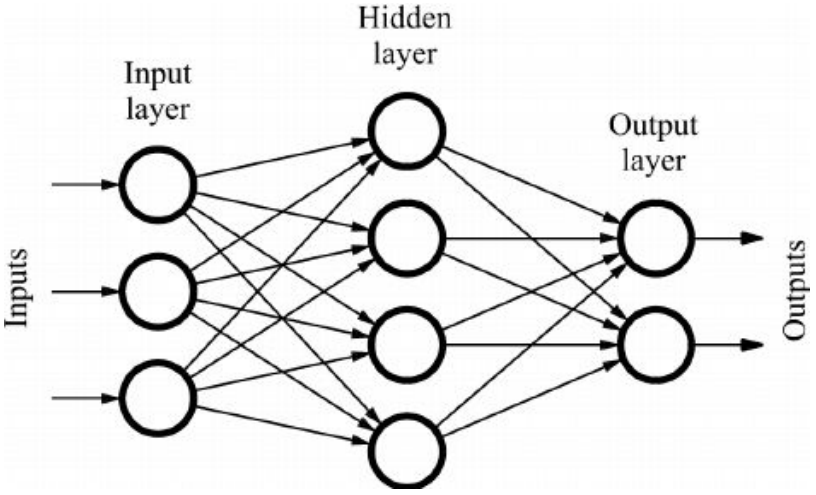
Προς το παρόν, η μάθηση χωρίς επίβλεψη δεν είναι πλήρως κατανοητή. Ένας από τους κορυφαίους ερευνητές στην μάθηση χωρίς επίβλεψη είναι ο Tuevo Kohonen, ηλεκτρολόγος μηχανικό στο Τεχνολογικό Πανεπιστήμιο του Ελσίνκι. Έχει αναπτύξει ένα αυτό οργανωτικό δίκτυο, που μαθαίνει χωρίς το όφελος της γνώσης της σωστής απάντησης. Είναι ένα ασυνήθιστο δίκτυο που φαίνεται ότι περιέχει ένα μόνο στρώμα με πολλές συνδέσεις. Τα βάρη για τις συνδέσεις αυτές πρέπει να αρχικοποιηθούν και οι

είσοδοι πρέπει να εξομαλυνθούν καθώς οι νευρώνες δημιουργούνται για να ανταγωνίζονται.

Ο Kohonen συνεχίζει την έρευνά του σε δίκτυα τα οποία είναι δομημένα διαφορετικά από τα πρότυπα του feedforward και του backpropagation. Το έργο του Kohonen ασχολείται με την ομαδοποίση νευρώνων σε πεδία. Οι νεύρωνες μέσα σε ένα πεδίο είναι “τοπολογικά διατεταγμένοι”. Η τοπολογία είναι ένας κλάδος των μαθηματικών που μελετά τον τρόπο χαρτογράφησης από έναν χώρο στον άλλο χωρίς να αλλάζει την γεωμετρική διαμόρφωση. Οι τρισδιάστατες ομαδοποιήσεις που βρίσκονται συχνά στους εγκεφάλους των θηλαστικών, είναι ένα παράδειγμα τοπολογίας.

Ο Kohonen έχει επισημάνει ότι η έλλειψη τοπολογίας στα μοντέλα νευρωνικών δικτύων κάνει τα σημερινά νευρωνικά δίκτυα απλές αφαιρέσεις των πραγματικών νευρωνικών δικτύων μέσα στον εγκέφαλο. Καθώς αυτή η έρευνα συνεχίζεται, υπάρχουν πιθανότητες για περισσότερα ισχυρά δίκτυα με αυτοδιδασκαλία.

2.1 Δίκτυα Feedforward

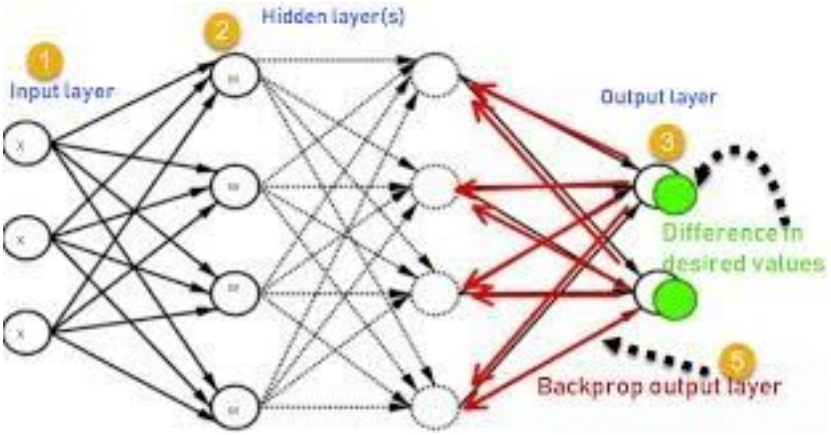


Ο στόχος μας στην χρήση ενός νευρωνικού δικτύου είναι να φτάσουμε στο σημείου του ελάχιστου σφάλματος όσο το δυνατόν γρηγορότερα. Κάνουμε έναν “αγώνα” και ο αγώνας είναι σε μία πίστα, έτσι περνάμε τα ίδια σημεία επανειλημμένα σε έναν βρόγχο. Η γραμμή εκκίνησης για τον αγώνα είναι η κατάσταση στην οποία αρχικοποιούνται τα βάρη μας και η γραμμή τερματισμού είναι η κατάσταση αυτών των παραμέτρων όταν θα είναι σε θέση να παράγουν επαρκώς ακριβείς ταξινομήσεις και προβλέψεις.

Ο ίδιος ο αγώνας περιλαμβάνει πολλά βήματα και κάθε ένα από αυτά τα βήματα, μοιάζει με τα βήματα του πριν και του μετά. Ακριβώς όπως ένας δρομέας, θα αναλάβουμε την επαναλαμβανόμενη πράξη ξανά και ξανά για να φτάσουμε στον τερματισμό. Κάθε βήμα για ένα νευρωνικό δίκτυο περιλαμβάνει μία εικασία, μία μέτρηση σφάλματος και μία μικρή ενημέρωση των βαρών του, μία βαθμιαία προσαρμογή στους συντελεστές καθώς μαθαίνει αργά να δίνει προσοχή στα πιο σημαντικά χαρακτηριστικά.

Αυτό οφείλεται στο γεγονός ότι ένα νευρωνικό δίκτυο είναι εξ’ αρχής στην “‘άγνοια”. Δεν γνωρίζει ποια βάρη και biases θα μεταφράσουν καλύτερα την είσοδο για να κάνουν σωστές εικασίες. Πρέπει να ξεκινήσει μία εικασία, και στην συνέχεια να προσπαθήσει να κάνει καλύτερες εικασίες διαδοχικά, καθώς μαθαίνει από τα λάθη του.

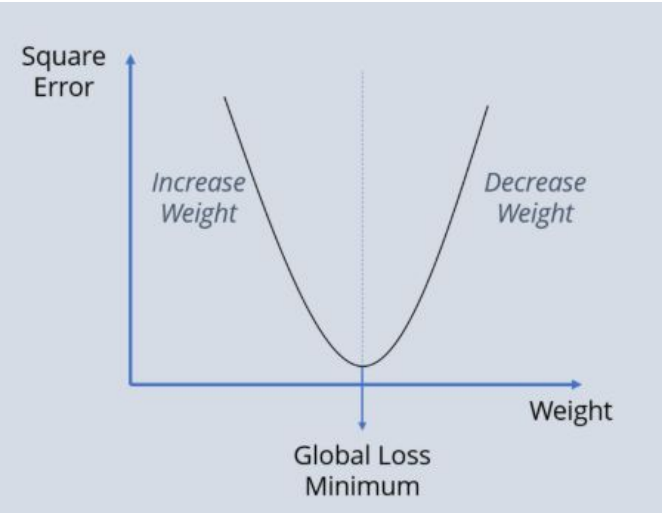
2.2 Αλγόριθμος Back-Propagation



Ένα νευρωνικό δίκτυο μεταδίδει το σήμα των δεδομένων εισόδου προς τα εμπρός μέσω των παραμέτρων του και στην συνέχεια, πηγαίνοντας προς τα πίσω, παράγει πληροφορίες σχετικά με το σφάλμα αντίστροφα μέσω του δικτύου, έτσι ώστε να μπορεί να αλλάξει παραμέτρους. Αυτό συμβαίνει βήμα βήμα:

1) Το δίκτυο κάνει μία εικασία σχετικά με τα δεδομένα χρησιμοποιώντας τους παραμέτρους του.
2) Το δίκτυο μετράται με λειτουργία απώλειας.
3) Το σφάλμα επαναπροσανατολίζεται για να προσαρμόσει τις παραπλανητικές παραμέτρους.

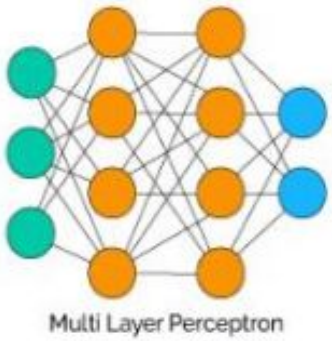
Έτσι προσπαθούμε να πάρουμε την τιμή του βάρους έτσι ώστε το σφάλμα να γίνει ελάχιστο. Βασικά πρέπει να υπολογίσουμε αν πρέπει να αυξήσουμε ή να μειώσουμε την τιμή του βάρους. Μόλις το ξέρουμε αυτό, συνεχίζουμε να ενημερώνουμε την τιμή του βάρους προς αυτήν την κατεύθυνση έως ότου το σφάλμα να γίνει ελάχιστο. Ενδέχεται να φτάσουμε σε ένα σημείο, όπου εάν ενημερώσετε περαιτέρω το βάρος, το σφάλμα να αυξηθεί. Τότε πρέπει να σταματήσουμε και αυτή θα είναι η τελική αξία του βάρους.



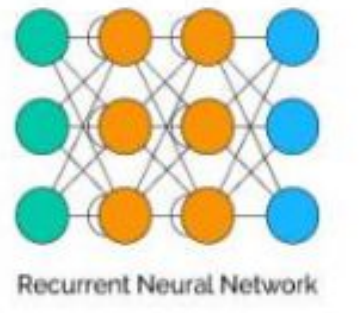
2.3 Τύποι Νευρωνικών Δικτύων στην Τεχνητή Νοημοσύνη

●**Perceptron:** Το συγκεκριμένο δίκτυο χρησιμοποιεί δύο εισόδους και μια έξοδο, χωρίς κρυφά επίπεδα. Χαρακτηρίζεται ως “single layer perceptron” και “linear binary classifier”

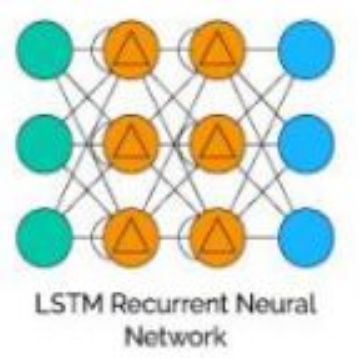
●**Multilayer Perceptron:** Ονομάζεται επίσης και Feed Forward Neural Network. Στα multilayer perceptrons υπάρχουν περισσότερα από έν γραμμικά στρώματα (συνδυασμοί νευρώνων). Αν πάρουμε το απλό παράδειγμα του δικτύου με τρία στρώματα, το πρώτο στρώμα θα είναι το στρώμα εισόδου, το τελευταίο το στρώμα εξόδου και το μεσαίο θα αποκαλείται κρυμμένο στρώμα. Τροφοδοτούμε τα δεδομένα εισόδου και παίρνουμε την έξοδο από το στρώμα εξόδου. Μπορούμε να αυξήσουμε τον αριθμό των κρυμμένων στρωμάτων όσο θέλουμε, έτσι ώστε το μοντέλο να γίνει πιο περίπλοκο.


Multi Layer Perceptron

●**Recurrent Neural Network:** Η ιδέα πίσω από τα RNNs είναι να κάνουν χρήση διαδοχικών πληροφοριών. Σε ένα παραδοσιακό νευρωνικό δίκτυο υποθέτουμε ότι όλες οι είσοδοι (και οι έξοδοι) είναι ανεξάρτητες μεταξύ τους. Τα RNNs ονομάζονται recurrent (επαναλαμβανόμενα) επειδή εκτελούν την ίδια εργασία για κάθε στοιχείο μιας ακολουθίας, με την έξοδο να εξαρτάται από τους προηγούμενους υπολογισμούς και ήδη γνωρίζουμε ότι έχουν μία “μνήμη” που κρατάει πληροφορίες για το τι έχει υπολογιστεί μέχρι τώρα.



●**Long Short-Term Memory Neural Network (LSTM):** Τα LSTM δίκτυα είναι ένα ειδικό είδος RNN, ικανό να μαθαίνει μακροπρόθεσμες εξαρτήσεις. Εισήχθησαν από τον Hochreiter και Schmidhuber το 1997. Δουλεύουν εξαιρετικά καλά σε μία μεγάλη ποικιλία προβλημάτων και τώρα χρησιμοποιούνται ευρέως. Τα LSTM έχουν σχεδιαστεί ειδικά για να αποφευχθεί το πρόβλημα της μακροχρόνιας εξάρτησης. Η ικανότητα “ανάμνησης” πληροφοριών για μεγάλες χρονικές περιόδους είναι η προεπιλεγμένη συμπεριφορά τους. Όλα τα επαναλαμβανόμενα νευρωνικά δίκτυα έχουν την μορφή επαναλαμβανόμενων ενοτήτων του δικτύου. Στα RNNs, αυτή η επαναλαμβανόμενη ενότητα θα έχει μία πολύ απλή δομή. Τα LSTM έχουν επίσης αυτή την αλυσίδα, αλλά η επαναλαμβανόμενη ενότητα έχει διαφορετική δομή. Αντό να έχουμε ένα ενιαίο στρώμα νευρωνικού δικτύου, υπάρχουν τέσσερις, που αλληλεπιδρούν με πολύ ειδικό τρόπο.



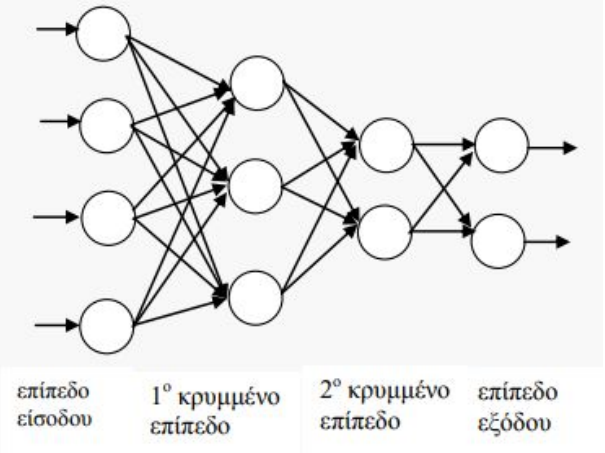
●**Hopfield Network:** Ένα δίκτυο Hopfield είναι ένας ειδικός τύπος επαναλαμβανόμενου τεχνητού νευρωνικού δικτύου που βασίζεται στην έρευνα του John Hopfield στην δεκαετία του ‘80 σε μοντέλα συνδυαστικών νευρωνικών δικτύων.

Τα δίκτυα Hopfield συνδέονται με την έννοια της προσομοίωσης της ανθρώπινης μνήμης μέσω της αναγνώρισης και της αποθήκευσης των σχεδίων.

2.4 Το Πολυεπίπεδο Perceptron (Multilayer Perceptron - MLP)

Τα Multilayer Perceptrons εφαρμόζονται συχνά σε επιβλεπόμενα προβλήματα μάθησης. Εκπαιδεύουν σε ένα σύνολο ζευγών εισόδου-εξόδου και μαθαίνουν να μοντελοποιούν την συσχέτιση μεταξύ αυτών των εισόδων και εξόδων. Η εκπαίδευση περιλαμβάνει την προσαρμογή των παραμέτρων ή των βαρών ώστε να ελαχιστοποιηθεί το σφάλμα. Ο αλγόριθμος Back-propagation χρησιμοποιείται για να κάνει αυτές τις ρυθμίσεις σε σχέση με το σφάλμα το οποίο μπορεί να μετρηθεί με διάφορους τρόπους συμπεριλαμβανομένου του μέσου τετραγωνικού σφάλματος ρίζας (RMSD - Root Mean Square Deviation)

Τα απλούστερα βαθιά δίκτυα ονομάζονται Multilayer Perceptrons και αποτελούνται από πολλά στρώματα νευρώνων, κάθε ένα από τα οποία είναι πλήρως συνδεδεμένο με αυτά που βρίσκονται στο κάτω στρώμα (από το οποίο λαμβάνουν την είσοδο) και εκείνα που είναι από πάνω (τα οποία με την σειρά τους επηρεάζουν). Όταν εκπαιδεύουμε μοντέλα μεγάλης χωρητικότητας διατρέχουμε τον κίνδυνο υπερφόρτωσης. Εάν θέλουμε να ξεπεράσουμε τους περιορισμούς της γραμμικής διαχωρισιμότητας και να διευρύνουμε το φάσμα των λειτουργιών, μπορούμε να ενσωματώσουμε ένα ή περισσότερα κρυμμένα στρώματα. Ο ευκολότερος τρόπος για να γίνει αυτό είναι να στοιβάζονται πολλά πλήρως συνδεδεμένα στρώματα το ένα πάνω στο άλλο. Κάθε στρώμα τροφοδοτεί το στρώμα πάνω από αυτό, μέχρι να δημιουργήσουμε μία έξοδο



Το πρώτο στρώμα που λαμβάνει δεδομένα από το σύνολο των δεδομένων ονομάζεται ορατό επίπεδο (επίπεδο εισόδου) επειδή είναι το εκτεθειμένο τμήμα του δικτύου. Συχνά, ένα νευρωνικό δίκτυο σχεδιάζεται με ένα ορατό επίπεδο με έναν νευρώνα ανά είσοδο. Οι συγκεκριμένοι νεύρωνες περνούν την τιμή εισόδου στο επόμενο στρώμα.

Τα επίπεδα μετά το στρώμα εισόδου ονομάζονται κρυμμένα στρώματα (hidden layers) επειδή δεν είναι άμεσα εκτεθειμένα στην είσοδο. Η απλούστερη δομή του δικτύου είναι να έχουμε ένα μόνο κρυφό στρώμα που εκπέμπει απευθείας την τιμή.

Δεδομένης της αύξησης της υπολογιστικής ισχύος και των αποδοτικών βιβλιοθηκών, μπορούν να κατασκευαστούν πολύ βαθιά νευρωνικά δίκτυα. Η βαθιά εκμάθηση (deep learning) μπορεί να αναφέρεται στην ύπαρξη πολλών κρυφών επιπέδων στο νευρωνικό δίκτυο.

Το τελικό κρυφό επίπεδο ονομάζεται στρώμα εξόδου και είναι υπεύθυνο για την εξαγωγή μία τιμής ή διανύσματος τιμών που απαιτείται για το πρόβλημα. Η επιλογή της λειτουργίας ενεργοποίησης σε αυτό το επίπεδο εξόδου περιορίζεται έντονα από το είδος του προβλήματος που διαμορφώνεται.

Υπάρχει πλήρης διασύνδεση μεταξύ των νευρώνων δύο διαδοχικών επιπέδων. Συνήθως δεν επιτρέπονται συνδέσεις μεταξύ των νευρώνων που ανήκουν σε επίπεδα που δεν είναι διαδοχικά.

Τα δίκτυα feedforward όπως τα MLP είναι σαν τενις ή πινγκ πονγκ. Συμμετέχουν κυρίως σε δύο κινήσεις, μία σταθερή μπροστά και πίσω. Μπορούμε να σκεφτούμε αυτό το πινγκ πονγκ των εικασιών και των απαντήσεων ως ένα είδος επιταχυνόμενης επιστήμης, αφού κάθε εικασία είναι μία δοκιμή για το τι πιστεύουμε ότι γνωρίζουμε και κάθε απάντηση είναι η ανατροφοδότηση που μας ενημερώνει πόσο λάθος είμαστε.

Στο πέρασμα του δικτύου με αλγόριθμο Feedforward, η ροή μετακινείται από το στρώμα εισόδου μέσω των κρυφών στρωμάτων στο στρώμα εξόδου και η απόφαση του στρώματος εξόδου μετράται.

Στο πέρασμα του δικτύου με αλγόριθμο Backpropagation, τα διάφορα βάρη και biases πολλαπλασιάζονται εκ νέου μέσω του MLP. Αυτή η πράξη διαφοροποίησης μας δίνει μία κλίση ή ένα τοπίο σφάλματος, κατά μήκος του οποίου οι παράμετροι μπορούν να ρυθμιστούν καθώς μετακινούν το MLP ένα βήμα πιο κοντά στο ελάχιστο λάθος. Αυτό μπορεί να γίνει με οποιονδήποτε αλγόριθμο βασισμένο σε κλίση, όπως ο αλγόριθμος Gradient Descent.

Το δίκτυο εξακολουθεί να παίζει αυτό το παιχνίδι τένις μέχρι το σφάλμα να μην μειώνεται άλλο. Αυτή η κατάσταση είναι γνωστή ως σύγκλιση.

2.5 Εκπαίδευση Πολυεπίπεδων Νευρωνικών Δικτύων

Προετοιμασία Δεδομένων

Πρέπει πρώτα να προετοιμάσουμε τα δεδομένα μας για εκπαίδευση σε ένα νευρωνικό δίκτυο.

Τα δεδομένα πρέπει να είναι αριθμητικά, για παράδειγμα πραγματικές τιμές. Αν έχουμε δεδομένα όπως ένα χαρακτηριστικό φύλου με τις τιμές “αρσενικό” και “θηλυκό”, μπορούμε να τα μετατρέψουμε σε μία αντιπροσωπευτική αναπαράσταση που ονομάζεται κωδικοποίηση. Εδώ προστίθεται μία νέα στήλη για κάθε τιμή κλάσης (δύο στήλες στην περίπτωση φύλου αρσενικού και θηλυκού) και ένα 0 ή 1 προστίθεται για κάθε σειρά ανάλογα με την τιμή κλάσης για την συγκεκριμένη σειρά.

Αυτή η κωδικοποίηση μπορεί να χρησιμοποιηθεί στην μεταβλητή εξόδου σε προβλήματα ταξινόμησης με περισσότερες από μία κλάση. Αυτό θα δημιουργούσε ένα δυαδικό διάνυσμα από μία μονή στήλη που θα ήταν εύκολο να συγκριθεί άμεσα με την έξοδο του νευρώνα στο επίπεδο του δικτύου, που όπως περιγράψαμε παραπάνω, θα εξάγει μία τιμή για κάθε κατηγορία.

Επανακαθορισμός Βαρών

Τα βάρη στο δίκτυο μπορούν να ενημερωθούν από τα σφάλματα που υπολογίζονται για κάθε εκπαιδευτικό παράδειγμα και αυτό ονομάζεται ηλεκτρονική μάθηση. Μπορεί α οδηγήσει σε γρήγορες αλλά και χαοτικές αλλαγές στο δίκτυο.

Εναλλακτικά τα σφάλματα μπορούν να αποθηκευτούν σε όλα τα παραδείγματα εκπαίδευσης και το δίκτυο μπορεί να ενημερωθεί στο τέλος. Αυτό ονομάζεται μαζική μάθηση και συχνά είναι πιο σταθερό.

Τυπικά, επειδή τα σύνολα δεδομένων είναι μεγάλα λόγω υπολογιστικής αποτελεσματικότητας, το μέγεθος της παρτίδας, ο αριθμός των παραδειγμάτων του δικτύου εμφανίζεται προτού μία ενημέρωση μειωθεί συχνά σε μικρό αριθμό, όπως δεκάδες ή εκατοντάδες παραδείγματα. Το ποσό που ενημερώνονται τα βάρη ελέγχεται από παραμέτρους που ονομάζονται ρυθμός εκμάθησης. Ονομάζεται επίσης το μέγεθος βήματος και ελέγχει το βήμα ή την αλλαγή που έγινε στο βάρος του δικτύου για ένα δεδομένο σφάλμα. Συχνά χρησιμοποιούνται μικρά μεγέθη βαρους όπως 0,1 ή 0,11 ή και μικρότερα.

Πρόβλεψη

Μόλις εκπαιδευτεί ένα νευρωνικό δίκτυο, μπορεί να χρησιμοποιηθεί για να κάνει προβλέψεις. Μπορούμε να κάνουμε προβλέψεις σχετικά με δεδομένα δοκιμής ή επικύρωσης, προκειμένου να εκτιμήσουμε την ικανότητα του μοντέλου σε δεδομένα

που δεν βλέπονται. Μπορούμε επίσης να αναπτύξουμε λειτουργικά και να το χρησιμοποιήσουμε για να κάνουμε προβλέψεις συνεχώς.

Η τοπολογία του δικτύου και το τελικό σύνολο βαρών είναι όλα όσο πρέπει να αποθηκεύσουμε από το μοντέλο. Οι προβλέψεις πραγματοποιούνται παρέχοντας την είσοδο στο δίκτυο και εκτελώντας ένα forward-pass που επιτρέπει να παράγει μία έξοδο την οποία μπορούμε να χρησιμοποιήσουμε ως πρόβλεψη.

2.6 Μέσο τετραγωνικό σφάλμα ρίζας (RMSD - Root Mean Squared Deviation)

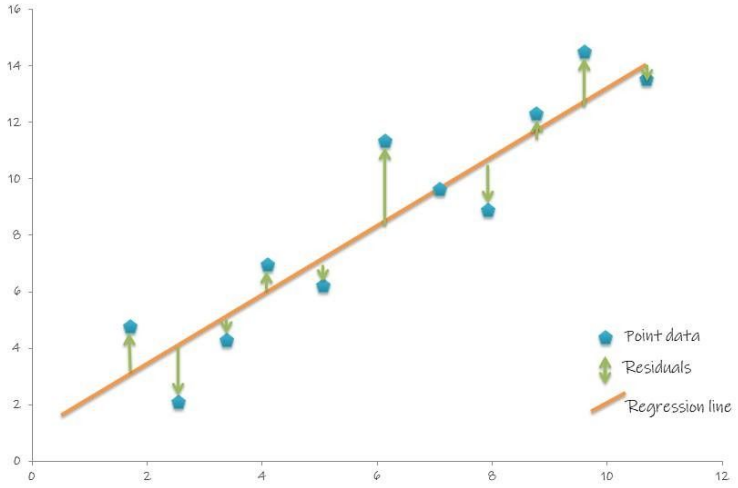
Το μέσο τετραγωνικό σφάλμα ρίζας είναι μία συχνά χρησιμοποιούμενη μέτρηση των διαφορών τιμών. Το RMSD αντιπροσωπεύει την τετραγωνική ρίζα του δείγματος των διαφορών μεταξύ των προβλεπόμενων τιμών και των υπάρχων τιμών ή του τετραγωνικού μέσου αυτών. Αυτές οι αποκλίσεις ονομάζονται υπολείμματα (residuals) όταν οι υπολογισμοί εκτελούνται πάνω στο δείγμα δεδομένων που χρησιμοποιήθηκε για την εκτίμηση και ονομάζονται σφάλματα (errors). Το RMSD χρησιμεύει για την συγκέντρωση των μεγεθών των σφαλμάτων στις προβλέψεις.

Τυπικώς είναι γνωστό με τον τύπο:

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}$$

Όπου: y^i, οι προβλεπόμενες τιμές
και yi, οι υπάρχουσες τιμές
n, ο αριθμός των υπάρχον τιμών

Θα πρέπει επίσης να εξηγήσουμε για την διαίρεση με το n κάτω από την τετραγωνική ρίζα στο RSMD: μας επιτρέπει να υπολογίσουμε την τυπική απόκλιση του σφάλματος για μια τυπική ενιαία παρατήρηση παρά ένα είδος συνολικού σφάλματος. Διαχωρίζοντας με το n διατηρούμε αυτό το μέτρο σφάλματος συνεπές, καθώς μεταβαίνουμε από μία μικρή συλλογή παρατηρήσεων σε μία μεγαλύτερη συλλογή. Για να το εξηγήσουμε με διαφορετικό τρόπο, το RMSD είναι ένας καλός τρόπος για να απαντήσουμε στην ερώτηση του πόσο μακριά μπορεί να φτάσει το μοντέλο μας για να βγάλει την επόμενη πρόβλεψή του.



3.1 Συνελικτικά Νευρωνικά Δίκτυα (Convolutional Neural Networks)

Τα Συνελικτικά Νευρωνικά Δίκτυα ή Δίκτυα Βαθιάς Συνέλιξης (CNN) είναι νευρωνικά δίκτυα που χρησιμοποιούνται κυρίως για την ταξινόμηση εικόνων, την ομαδοποίηση εικόνων κατά ομοιότητα (αναζήτηση φωτογραφιών) και την αναγνώριση αντικειμένων. Για παράδειγμα, τα συνελικτικά νευρωνικά δίκτυα χρησιμοποιούνται για τον εντοπισμό προσώπων, ατόμων, οδών, όγκων και πολλών άλλων πτυχών των οπτικών δεδομένων.

Η αποτελεσματικότητα των συνελικτικών δικτύων στην αναγνώριση εικόνας είναι ένας από τους κύριους λόγους για τους οποίους ο κόσμος έχει αφυπνηστεί για την αποτελεσματικότητα της βαθιάς μάθησης (deep learning). Κατά μία έννοια, τα CNN είναι ο λόγος για τον οποίο η βαθιά μάθηση είναι διάσημη. Η επιτυχία μιας βαθιάς συνελικτικής αρχιτεκτονικής, που ονομάζεται AlexNet στον διαγωνισμό ImageNet του 2012, ήταν αυτή που ακούστηκε σε όλο τον κόσμο. Τα CNN ενισχύουν σημαντικές εξελίξεις στην υπολογιστική όραση (computer vision) , η οποία έχει προφανείς εφαρμογές για αυτοκίνητα, ρομποτική, drones, ασφάλεια, ιατρικές διαγνώσεις και θεραπείες για άτομα με προβλήματα όρασης.

Τα συνελικτικά δίκτυα μπορούν επίσης να εκτελέσουν πιο καθαρές και πιο κερδοφόρες επιχειρηματικές εργασίες,όπως η οπτική αναγνώριση χαρακτήρων (optical character recognition - OCR) για την ψηφιοποίηση κειμένου και την δυνατότητα επεξεργασίας γλώσσας σε αναλογικά και χειρόγραφα έγγραφα, όπου οι εικόνες είναι σύμβολα που πρέπει να μεταφραστούν.

Ωστόσο, τα CNN δεν περιορίζονται στην αναγνώριση εικόνας. Έχουν εφαρμοστεί απευθείας στην ανάλυση κειμένου. Εφαρμόζονται στον ήχο όταν απεικονίζεται οπτικά σε φασματογράφημα και δεδομένα γραφήματος.

3.2 Πως λειτουργούν τα Συνελικτικά Νευρωνικά Δίκτυα

Το πρώτο πράγμα που πρέπει να γνωρίζουμε για τα συνελικτικά δίκτυα είναι ότι δεν αντιλαμβάνονται εικόνες όπως οι άνθρωποι. Επομένως, θα πρέπει να σκεφτούμε με διαφορετικό τρόπο τι σημαίνει μία εικόνα καθώς τροφοδοτείται και υποβάλλεται σε επεξεργασία από ένα συνελικτικό δίκτυο.

Τα συνελικτικά δίκτυα αντιλαμβάνονται τις εικόνες ως κομμάτια. Δηλαδή, τρισδιάστατα αντικείμενα αντί για για επίπεδους καμβάδες που πρέπει να μετρώνται μόνο κατά πλάτος και ύψος. Αυτό συμβαίνει επειδή οι ψηφιακές έγχρωμες εικόνες έχουν κωδικοποίηση κόκκινου-μπλε-πράσινου (RGB), συνδυάζοντας τρία χρώματα για να παράγουν το φάσμα χρωμάτων που αντιλαμβάνονται οι άνθρωποι. Ένα συνελικτικό δίκτυο απορροφά εικόνες όπως τρία ξεχωριστά στρώματα χρώματος που στοιβάζονται το ένα πάνω από το άλλο.

Έτσι, ένα συνελικτικό δίκτυο λαμβάνει μία έγχρωμη εικόνα ως ένα ορθογώνιο κουτί του οποίου το πλάτος και το ύψος μετριούνται από των αριθμό των pixels κατά μήκος αυτών των διαστάσεων και του οποίου το βάθος είναι τρία επίπεδα βαθιά, ένα για κάθε γράμμα, σε RGB. Αυτά τα επίπεδα βάθους αναφέρονται ως κανάλια.

Καθώς οι εικόνες κινούνται μέσω ενός συνελικτικού δικτύου, θα τις περιγράψουμε με όρους όγκου εισόδου και εξόδου, εκφράζοντας τα μαθηματικά ως πίνακες πολλαπλών διαστάσεων σε αυτή την μορφή: 30x30x3. Από επίπεδο σε επίπεδο οι διαστάσεις αλλάζουν.

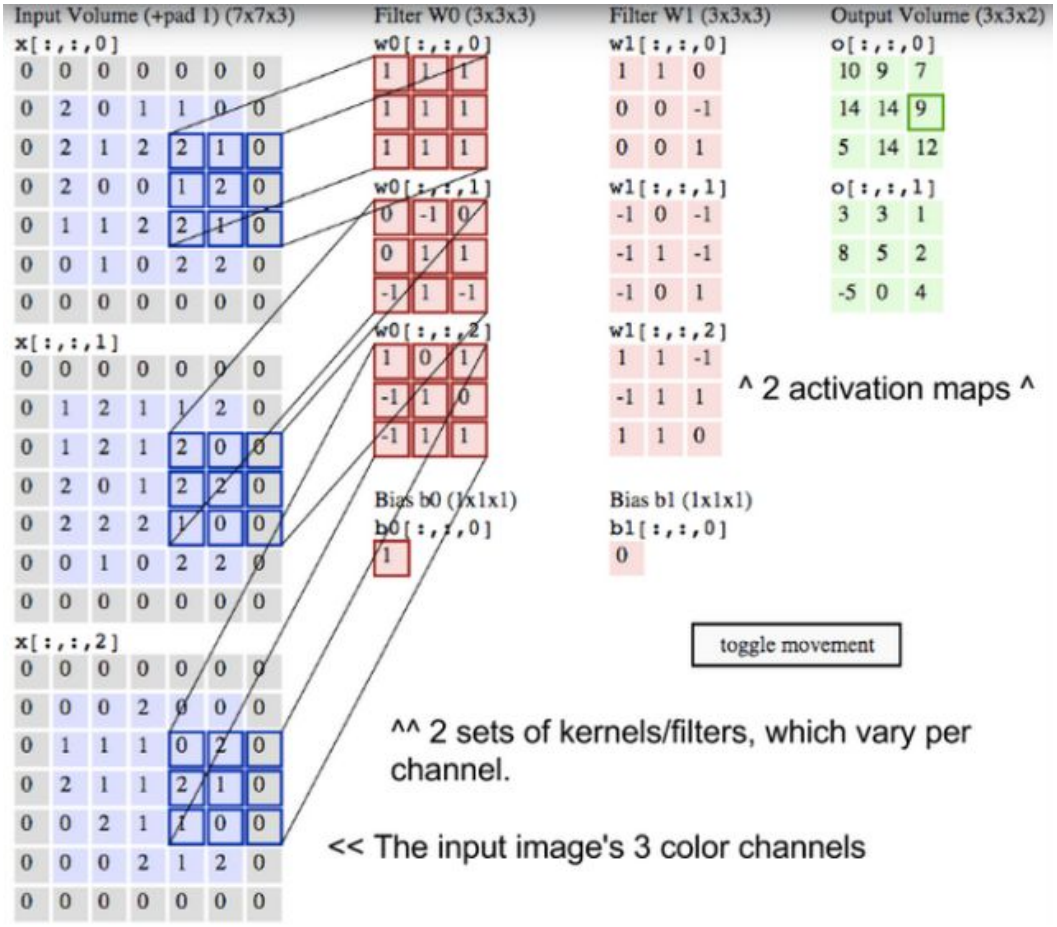
Θα πρέπει να δώσουμε ιδιαίτερη προσοχή στα ακριβή μέτρα κάθε διάστασης του όγκου της εικόνας, επειδή αποτελούν το θεμέλιο γραμμικών λειτουργιών άλγεβρας που χρησιμοποιούνται για την επεξεργασία εικόνων.

Τώρα για κάθε pixel μιας εικόνας, η ένταση των R, G και B θα εκφράζεται από έναν αριθμό, και αυτός ο αριθμός θα είναι ένα στοιχείο σε ένα από τα τρία, στοίβες δισδιάστατων πινάκων, οι οποίες μαζί σχηματίζουν τον όγκο της εικόνας.

Αυτοί οι αριθμοί είναι τα αρχικά, ακατέργαστα αισθητήρια χαρακτηριστικά που τροφοδοτούνται στο συνελικτικό δίκτυο και ο σκοπός του είναι να βρει ποιοι από αυτούς τους αριθμούς είναι σημαντικά σήματα που πραγματικά βοηθούν την ταξινόμηση των εικόνων με μεγαλύτερη ακρίβεια.

Αντί να επικεντρώνεται σε ένα pixel κάθε φορά, ένα συνελικτικό δίκτυο παίρνει τετράγωνα τμήματα pixel και τα περνά μέσα από ένα φίλτρο. Αυτό το φίλτρο είναι επίσης ένα τετράγωνο πλέγμα μικρότερο από την ίδια την εικόνα. Ονομάζεται επίσης πυρήνας (kernel) και η δουλειά του φίλτρο είναι να βρει μοτίβα στα pixel.

Μπορούμε να φανταστούμε δύο πίνακες. Ο ένας είναι 30x30 και ο άλλος 3x3. Δηλαδή το φίλτρο μπορεί να καλύψει το ένα εκατοστό της επιφάνειας της εικόνας.



Εάν δύο πίνακες έχουν υψηλές τιμές στις ίδιες θέσεις, η έξοδος θα είναι υψηλή. Αν δεν είναι υψηλές, η έξοδος θα είναι χαμηλή. Με αυτό τον τρόπο, μία μόνο τιμή δηλαδή η έξοδος μπορεί να μας πει εάν το μοτίβο των pixel στην εικόνα ταιριάζει με το μοτίβο των pixel που εκφράζεται από το φίλτρο μας.

Ας υποθέσουμε ότι το φίλτρο μας εκφράζει μία οριζόντια γραμμή με υψηλές τιμές κατά την δεύτερη σειρά και χαμηλές τιμές στη πρώτη και τρίτη σειρά. Τώρα, φανταζόμαστε ότι ξεκινάμε στην επάνω αριστερή γωνία της εικόνας και μετακινούμε το φίλτρο κατά μήκος της εικόνας βήμα προς βήμα μέχρι να φτάσει στην επάνω δεξια γωνία. Το μέγεθος του βηματος είναι γνωστό ως βήμα (stride). Μπορούμε να μετακινήσουμε το φίλτρο στην δεξιά στήλη ανά φορά ή μπορούμε να επιλέξουμε μεγαλύτερα βήματα.

Σε κάθε βήμα παίρνουμε ένα άλλο προϊόν και τοποθετούμε τα αποτελέσματα αυτού του προϊόντος σε έναν τρίτο πίνακα γνωστό ως χάρτη ενεργοποίησης. Το πλάτος ή ο αριθμός των στηλών του χάρτη ενεργοποίησης είναι ίσος με τον αριθμό των βημάτων που κάνει το φίλτρο για να διασχίσει την υποκείμενη εικόνα. Δεδομένου ότι τα μεγαλύτερα strides οδηγούν σε λιγότερα βήματα, ένα μεγάλο stride θα παράγει μικρότερο χάρτη ενεργοποίησης. Αυτό είναι σημαντικό διότι το μέγεθος των πινάκων που επεξεργάζονται και παράγουν συνελικτικά δίκτυα σε κάθε στρώμα είναι άμεσα ανάλογο με το όσο υπολογιστικά είναι ακριβό και πόσο χρόνο χρειάζονται για να εκπαιδευτούν. Ένα μεγάλο βήμα σημαίνει λιγότερο χρόνο και υπολογισμό.

Ένα φίλτρο που τοποθετείται στις τρεις πρώτες σειρές θα γλιστρήσει και θα ξεκινήσει ξανά με σειρές 4-6 της ίδιας εικόνας. Εάν έχει βλημα τριών, τότε θα παράγει έναν πίνακα που θα είναι 10x10. Το ίδιο φίλτρο που αντιπροσωπεύει μία οριζόντια γραμμή μπορεί να εφαρμοστεί και στα τρία κανάλια της υποκείμενης εικόνας R, G και B. Και οι τρεις χάρτες ενεργοποίησης 10x10 μπορούν να προστεθούν μαζί, έτσι ώστε ο συνολικός χάρτης ενεργοποίησης για μία οριζόντια γραμμή και στα τρία κανάλια της υποκείμενης εικόνας είναι 10x10.

Τώρα, επειδή οι εικόνες έχουν γραμμές προς πολλές κατευθύνσεις και περιέχουν πολλά διαφορετικά είδη σχημάτων και μοτίβων pixel, θα θελήσουμε να σύρουμε άλλα φίλτρα κατά μήκος της υποκείμενης εικόνας για αναζήτηση αυτών των μοτίβων. Θα μπορούσαμε, για παράδειγμα, να αναζητήσουμε 96 διαφορετικά μοτίβα στα pixel. Αυτά τα 96 μοτίβα θα δημιουργήσουν μία στοίβα 96 χαρτών ενεργοποίησης με αποτέλεσμα ένα νέο κομμάτι που είναι 10x10x96.

Ένα από τα κύρια προβλήματα με τις εικόνες είναι ότι είναι υψηλής διάστασης (high-dimensional) πράγμα που σημαίνει ότι κοστίζουν πολύ χρόνο και υπολογιστική ισχύ για επεξεργασία. Τα συνελικτικά δίκτυα έχουν σχεδιαστεί για να μειώσουν την διάσταση των εικόνων με πολλούς τρόπους. Το φιλτράρισμα είναι ένας τρόπος μείωσης των διαστάσεων. Ένας άλλος τρόπος είναι μέσω δειγματοληψίας.

Παράρτημα

Νευρωνικά Δίκτυα και ανάπτυξη συστήματος σε VHDL και FPGA

Τα Νευρωνικά δίκτυα είναι συστήματα που μαθαίνουν μέσα από μία σειρά δεδομένων. Αποτελούν πλέον μία πολύ σημαντική λογική στις επιστήμες του Machine Learning, της τεχνητής νοημοσύνης και του Data Mining.

Τα Νευρωνικά Δίκτυα μπορούν να χρησιμοποιηθούν ώστε να παράγονται μοντέλα ομαδοποίησης (classification) που με βάση σειρά από στοιχεία, αποφασίζουν.

Στην περίπτωσή μας προτείνεται η λειτουργία ενός συστήματος όπου ένα FPGA θα ελέγχει μία σειρά από διακόπτες (π.χ φωτισμός σε ένα δωμάτιο, κουζίνα, θερμοσίφωνας, κλπ). Το νευρωνικό δίκτυο αποτελεί βασικό σύστημα μέσω του οποίου θα ανοίξει ένας άλλος διακόπτης.

Η εφαρμογή των νευρωνικών δικτύων στην τεχνολογία των Smart homes, smart cars και γενικά έξυπνων συστημάτων είναι αρκετά συχνή και ιδιαίτερα αποτελεσματική.

Έτσι ουσιαστικά θα έχουμε ένα νευρωνικό δίκτυο που μετά την Νστή φορά που θα συμβούν γεγονότα τότε θα μπορεί να υπολογίζει αν ο χρήστης άνοιξε τον επόμενο διακόπτη (αυτό για παράδειγμα θα ήταν η ερώτηση αν ανοίξει το φως και το ψυγείο τότε κατά μεγάλη πιθανότητα θα ανοίξει και την τηλεόραση που θα αποτελεί τον τελικό διακόπτη). Το σύστημα αυτό μπορεί να προσαρμοστεί σε οποιαδήποτε περίπτωση συσκευών όπως σε αυτοκίνητα, σπίτια, εταιρείες, εργοστάσια, κλπ.

Αρχικά θα γίνει ένας σχεδιασμός του νευρωνικού δικτύου σε ένα περιβάλλον ΄πως το WEKA που έχει έτοιμα νευρωνικά συστήματα με samples data που είτε θα βρεθούν είτε θα δημιουργηθούν από εμάς ώστε να καταλήξουμε στον καλύτερο σχεδιασμό του νευρωνικού δικτύου και στην συνέχεια θα γίνει εφαρμογή σε VHDL.
Η εφαρμογή αυτή μπορεί να αποτελέσει πρότυπο σε εφαρμογές Internet Of Things.

Βήμα Πρώτο: Συλλογή στοιχείων

Αρχικά καταγράφουμε το άνοιγμα και κλείσιμο της σειράς από διακόπτες καθώς και δύο αισθητήρες που είναι της θερμοκρασίας και της φυσικής φωτεινότητας με στόχο τον έλεγχο ενός χώρου.
Έτσι έχουμε τον αισθητήρα κίνησης στον χώρο που ελεγχουμε, τον αισθητήρα φυσικού φωτισμού που ειναι σε εξωτερικό χώρο, τον αισθητήρα θερμοκρασίας, τον διακόπτη που αφορα το ψυγείο και την ώρα καταγραφής. Στόχος μας είναι να ελέγχουμε δύο διακόπτες που είναι το φώς καθώς και τον θερμοσίφωνα και την τηλεόραση.
Οι διακόπτες δίνουν ένα νούμερο 1,0 (on, off) οι αισθητήρες δίνουν συγκεκριμένες τιμές θερμοκρασίας και φωτισμού, για παράδειγμα η θερμοκρασία είναι σε βαθμούς Κελσίου ενώ ο φωτισμός σε 0 έως 1 με 1 μεγάλη φωτεινότητα (ηλιοφάνεια) και 0 σκοτάδι (βράδυ). Ο αισθητήρας κίνησης δίνει και αυτός 1 ή 0 που σημαίνει κίνηση ή όχι στον χώρο. Επίσης έχουμε και την ώρα που αφορά μόνο τον χρόνο σε δεκαδική μορφή π.χ 1:15 είναι 1.25 δεκαδικά.

Έτσι έχουμε την παρακάτω μορφή σε αρχείο CSV:

Time	Movement	Temp	brightness	Refrigerator	Light	TV	Heater
8,083581499	1	13,68048	0,8	0	0	0	0
9,000654954	0	13,31594	0,8	1	0	0	0
10,00090949	0	11,23471	0,8	0	0	0	0
11,00737422	0	17,46423	0,9	0	0	0	0
12,00226243	0	16,78073	1	1	0	0	0
13,00306566	1	19,0622	1	0	0	0	0
14,00417875	1	17,63038	1	1	0	0	0
15,00683491	1	15,30759	1	1	0	1	0
16,00105714	1	19,85282	0,9	0	0	1	0
17,00017134	0	13,74276	0,8	1	0	0	0
18,00464128	0	12,82111	0,6	1	0	1	0
19,00136896	1	10,09186	0,3	0	1	1	1
20,00545084	0	10,99898	0	1	1	1	0
21,00167251	1	12,18637	0	0	1	1	0
22,00096301	1	13,3872	0	0	1	1	0
23,03659085	0	10,55821	0	0	1	1	0
24,00766934	1	11,32828	0	1	1	1	0

Τα δεδομένα βρέθηκαν από την διεύθυνση για data διαθέσιμα για ανάλυση: https://data.world/

Διαδικασία για την εξαγωγή του Νευρωνικού Δικτύου

Τα μοντέλα μηχανικής μάθησης λειτουργουν με την λογική “Μαθαίνω από τα δεδομένα”. Έτσι στην περίπτωσή μας έχουμε μια σειρά καταγραφών από το σύστημά μας. Κάθε μοντέλο μάθησης έχει στόχο να δημιουργήσει μία μηχανή που θα έχει το βέλτιστο αποτέλεσμα μεταξύ των τιμών που προβλέπει με αυτές που έχουν εκτιμηθεί από τους ειδικούς.

Έτσι σε κάθε μοντέλο υπάρχει ένα σύνολο μάθησης και ένα ή περισσότερα σύνολα τεστ με τα οποία ελέγχουμε κατά πόσο το σύστημά μας έχει “μάθει σωστά”. Τα μοντέλα αυτά παράγονται έχοντας σαν είσοδο τους παράγοντες μαζί με τις εκτιμήσεις.

Στην συνέχεια κάθε μοντέλο ακολουθεί μία μεθοδολογία με στόχο να καταλήξει στο σημείο τερματισμού, που για κάθε μοντέλο είναι το μεγαλύτερο δυνατό γι αυτό το ποσοστό πρόβλεψης.

Η μηχανική μάθηση δεν απαιτεί προηγούμενες παραδοχές σχετικά με τις σχέσεις μεταξύ των μεταβλητών. Στα μοντέλα της μηχανικής μάθησης πρέπει απλά να δώσουμε όλα τα δεδομένα που έχουμε συλλέξει και ο αλγόριθμος επεξεργάζεται τα δεδομένα και ανακαλύπτει τα πρότυπα με τα οποία μπορούμε να κάνουμε προβλέψεις για το νέο σύνολο δεδομένων (test set). Η μηχανική μάθηση αντιμετωπίζει έναν αλγόριθμό σαν ένα μαύρο κουτί (black box) για όσο διάστημα αυτό λειτουργεί. Γενικά εφαρμόζεται σε σύνολα με πολλές διαστάσεις (δηλαδή πολλές μεταβλητές) ή σε σύνολα με πολλές καταγραφές (παρατηρήσεις), όσο περισσότερα είναι τα δεδομένα, τόσο πιο ακριβής θα είναι η πρόβλεψη του μοντέλου.

Η μηχανική μάθηση λειτουργεί με διαφορετικό τρόπο από την στατιστική. Στην στατιστική τα μοντέλα που αναζητούνται απαιτούν να ξέρει κάποιος πως επιλέχθηκαν τα δεδομένα, τις στατιστικές ιδιότητες των εκτιμητών (p-value, αμερόληπτες εκτιμήτριες κ.α), την υποκείμενη κατανομή του πληθυσμού που μελετάμε π.χ στην περίπτωση των καταγραφών κατολισθήσεων και τα είδη των ιδιοτήτων για κάθε μεταβλητή. Θα πρέπει να ξέρει ακριβώς τι κάνει για να καταλήξει σε παραμέτρους που θα παρέχουν την ικανότητα της πρόβλεψης. Οι τεχνικές στατιστικής μοντελοποίησης εφαρμόζονται συνήθως σε σύνολα χαμηλών διαστάσεων των δεδομένων, δηλαδή λιγότερες μεταβλητές και παρατηρήσεις.

Φυσικά και τα μοντέλα μηχανικής μάθησης χρησιμοποιούν στατιστική μέχρι να καταλήξουν στα αποτελέσματά τους, παρολα αυτα το τελικό μοντέλο είναι κάτι αυτόνομο και πολλές φορές η φυσική σημασία δεν έχει ακριβώς τόση έννοια αφού ουσιαστικά προσπαθούν να αποδώσουν την σύνθεση των παραγόντων μεταξύ τους με τέτοιο τρόπο που απλά να ικανοποιεί τα δεδομένα και τα αποτελέσματα αυτών. Ειδικά σε πολυπαραγοντικά συστήματα τα μοντέλα αυτά δίνουν πολλαπλές σχέσεις μεταξύ των παραγόντων με τέτοιο τρόπο που είναι σχεδόν αδύνατον να καταλάβει κανείς το γιατί. Λειτουργούν δηλαδή σαν ένα απλό φυσικό σύστημα που μαθαίνει,

όπως ένα παιδί που μεγαλώνει μαθαίνει να κάνει πράξεις. Ο τρόπος που οι νευρώνες λειτουργούν μεταξύ τους, ώστε να δοθεί αποτέλεσμα, είναι πρακτικά ακατανόητος. (Shai Shalev-Shwartz and Shai Ben-David - Understanding Machine Learning)

Νευρωνικά δίκτυα μέσω του προγράμματος WEKA

Το WEKA είναι ένα λογισμικό για μηχανική μάθηση και εξόρυξη δεδομένων γραμμένο σε Java. Αναπτύχθηκε στο πανεπιστήμιο του Waikato της Νέας Ζηλανδίας και διατίθεται ως ελεύθερο λογισμικό. Πήρε το όνομα του από το Weka, ένα μικρό και υπό εξαφάνιση πτηνό της Νέας Ζηλανδίας. Η μεγάλη ποικιλία μεθόδων εξόρυξης δεδομένων που περιλαμβάνει, η συνεχής υποστήριξη και εξέλιξη από μία διεθνή ομάδα προγραμματιστών, η ελεύθερη διανομή του πηγαίου κώδικα και η δυνατότητα εγκατάστασης του σε διαφορετικές υλικού και λογισμικού είναι ορισμένοι από τους παράγοντες που συμβάλλουν στην ευρύτερη αποδοχή και στην μεγάλη διάδοσή του. Επίσης, η γραφική διεπαφή που διαθέτει επιτρέπει τη χρήση του από χρήστες, οι οποίοι δεν έχουν ικανότητες προγραμματισμού.

Οι αλγόριθμοι και τα εργαλεία κατηγοριοποίησης που διαθέτει το WEKA είναι αξιοσημείωτα. Παρέχονται υλοποιήσεις όλων των κύριων μεθόδων κατηγοριοποίησης, όπως Δέντρα Αποφάσεων, Νευρωνικά Δίκτυα, Μηχανές Διανυσμάτων Υποστήριξης, Λογιστική Παλινδρόμηση, k-Πλησιέστεροι Γείτονες κλπ. Για κάθε μέθοδο υπάρχουν πολλές δυνατότητες παραμετροποίησης. Επίσης, διατίθενται πολλές παραλλαγές των βασικών μεθόδων, αλλά και εργαλεία για τη δημιουργία σύνθετων κατηγοριοποιητών bagging and boosting, κατηγοριοποιητών ευαίσθητων στο κόστος, κατηγοριοποιητών που χρησιμοποιούν ανάλυση συστάδων κλπ. Ο χρήστης μπορεί να επικυρώσει τα μοντέλα του εφαρμόζοντας τη μέθοδο cross validation, τη μέθοδο holdout ή χρησιμοποιώντας ένα ανεξάρτητο σύνολο δεδομένων. Για κάθε μοντέλο παρουσιάζονται αναλυτικά στοιχεία για τις επιδόσεις και τη δομή του (π.χ τα βάρη των συνδέσεων ενός δικτύου Multilayer Perceptron). Το WEKA περιλαμβάνει αρκετούς αλγορίθμους Ανάλυσης Συστάδων, όπως τον k-Means, τη Συσσωρευτική Ιεραρχική ΑΣ και το DBSCAN. Κάθε αλγόριθμος μπορεί να παραμετροποιηθεί.

Επίσης, υπάρχει δυνατότητα οπτικής αναπαράστασης της κατανομής των παρατηρήσεων στις συστάδες. Το tab ″Associate″ περιλαμβάνει αλγορίθμους για ανάλυση Κανόνων Συσχέτισης, μεταξύ των οποίων και τον βασικό αλγόριθμο Apriori. Υπάρχει η δυνατότητα εξόρυξης κανόνων συσχέτισης σε δεδομένα με πεδίο κλάσης. Οι κανόνες αυτοί θα έχουν στο δεξιό τμήμα τους μια τιμή κλάσης. Στο tab ″Select attributes″ ο χρήστης μπορεί να πειραματιστεί με διάφορες μεθόδους επιλογής χαρακτηριστικών και να συνδυάσει μεθόδους αναζήτησης με μεθόδους αξιολόγησης χαρακτηριστικών. Τέλος, στο tab ″Visualize″ υπάρχει ένας πίνακας διαγραμμάτων διασποράς. Ο χρήστης, κάνοντας κλικ σε ένα διάγραμμα, μπορεί να το προβάλει σε ξεχωριστό παράθυρο.

Το νευρωνικό δίκτυο που προτείνει το WEKA είναι ένας ταξινομητής που χρησιμοποιεί πολλαπλά επίπεδα με το λεγόμενο νευρώνα perceptron με backpropagation (κανόνας ελαχιστοποίησης του μέσου τετραγωνικού σφάλματος για όλα τα πρότυπα) και την σιγμοειδή συνάρτηση με στόχο την ταξινόμηση των εγγραφών μας.

Στην περίπτωση των νευρωνικών σημαντικοί παράμετροι είναι :
Hidden Layers: Κρυφά επίπεδα νευρώνων που χρησιμοποιούμε
Number of Nodes: Αριθμός νευρώνων ανά επίπεδο
Learning Rate: Ρυθμός μάθησης (ρυθμός με τον οποίο το σύστημα μαθαίνει)
Momentum: Ορμή όπου δείχνει το τι θα παράγουμε

Μεθοδολογία

Αρχικά έχουμε ένα CSV αρχείο που προκύπτει από το σύστημα μας το οποίο μπορούμε να το επεξεργαστούμε με το EXCEL και να κρατήσουμε τις στήλες που θέλουμε.

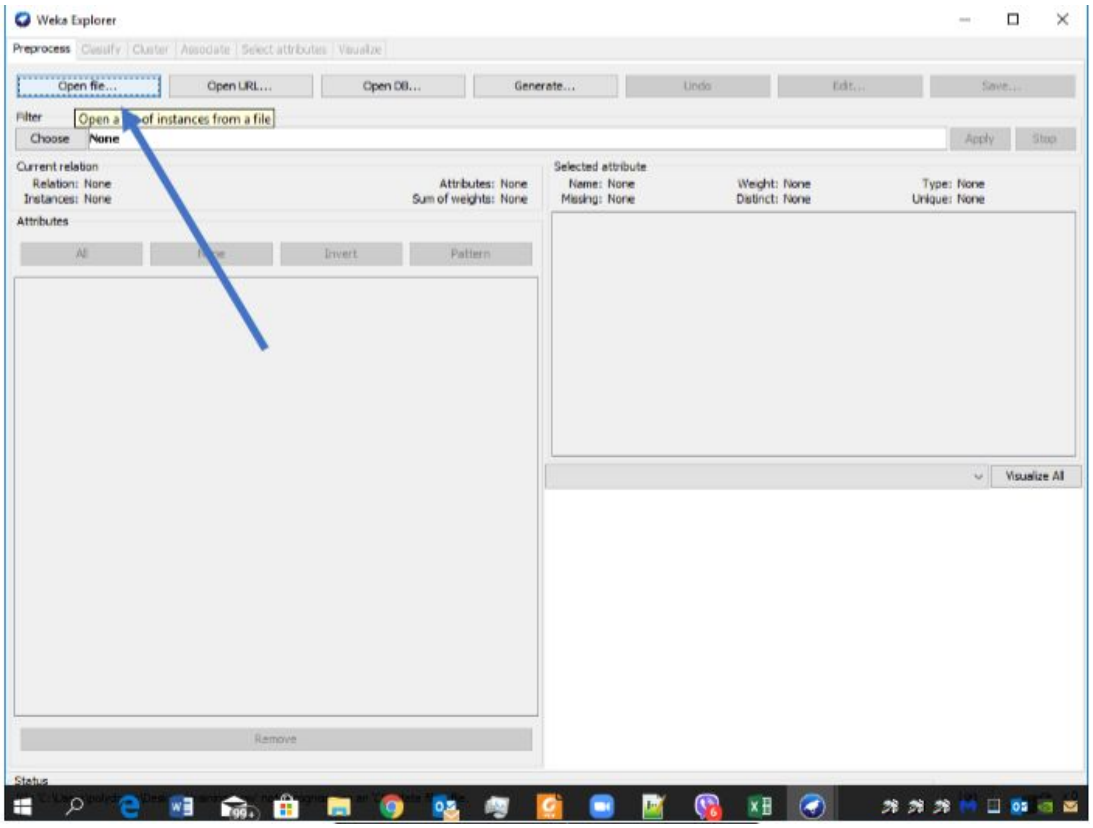
Το αρχείο αυτό μπορεί να περιλαμβάνει όλες τις εγγραφές με τις οποίες θα κάνουμε την επεξεργασία δηλαδή και τις εγγραφές του συνόλου Training και τις εγγραφές του συνόλου Testing.

Σημασία έχει το σύνολο Training να έχει όσο το δυνατόν περισσότερες εγγραφές. Τα σύνολα test παίζουν ρόλο για να επιβεβαιώσουμε το ποσοστό επιτυχίας του συστήματος μας. Πιο συγκεκριμένα το σύστημα μας θα μας δώσει ποσοστά επιτυχίας από το Training δηλαδή το κάθε μοντέλο θα εντοπίσει την μέγιστη απόδοση που μπορεί να έχει στο σύνολο Training και στην συνέχεια επιβεβαιώνει με ένα ή περισσότερα σύνολα testing ότι η απόδοση του δεν ήταν τυχαία αλλά δίνει σχετικά ίδιες αποδόσεις και στο testing.

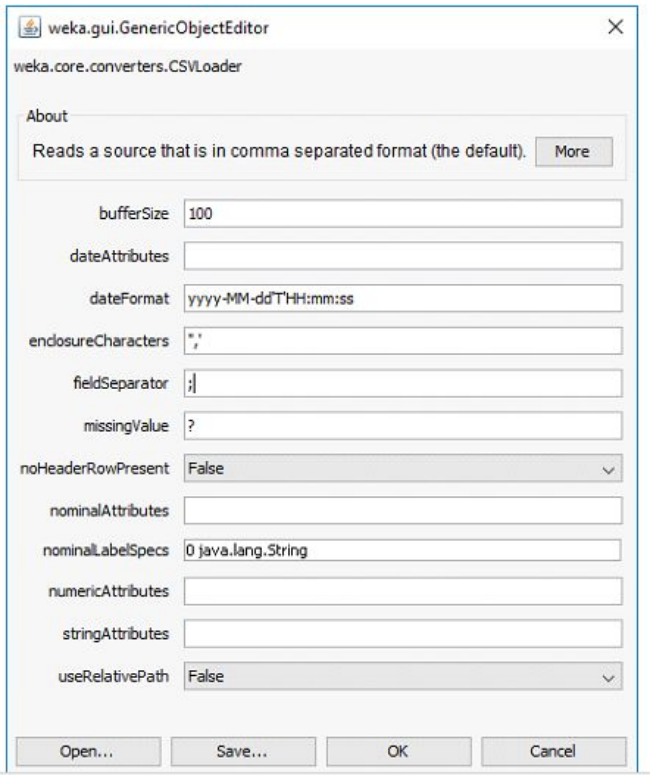
Το ποσοστό που συνήθως χρησιμοποιείται όταν έχουμε ένα σύνολο δεδομένων από 100 έως 500 εγγραφές είναι το 30% των εγγραφών αυτού να χρησιμοποιείται για
Training . Σε μεγαλύτερα σύνολα δηλαδή αν έχουμε 1000 εγγραφές τότε το ποσοστό εγγραφών για το Testing μπορεί να είναι και το 10% των εγγραφών ή και

μικρότερο.

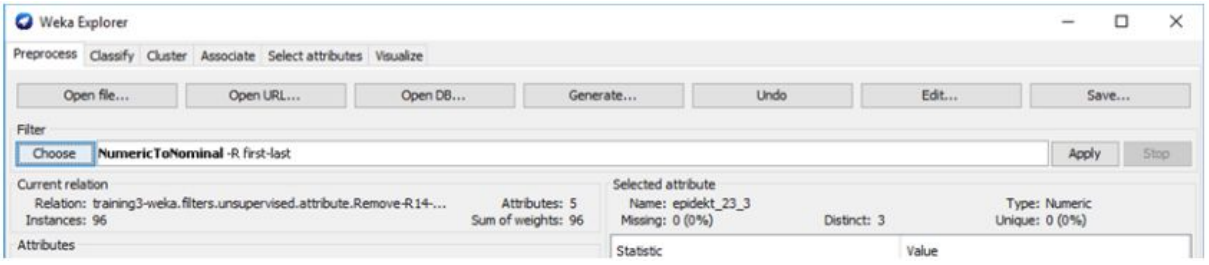
Το αρχείο αυτό το εισάγουμε στο WEKA με τον παρακάτω τρόπο:



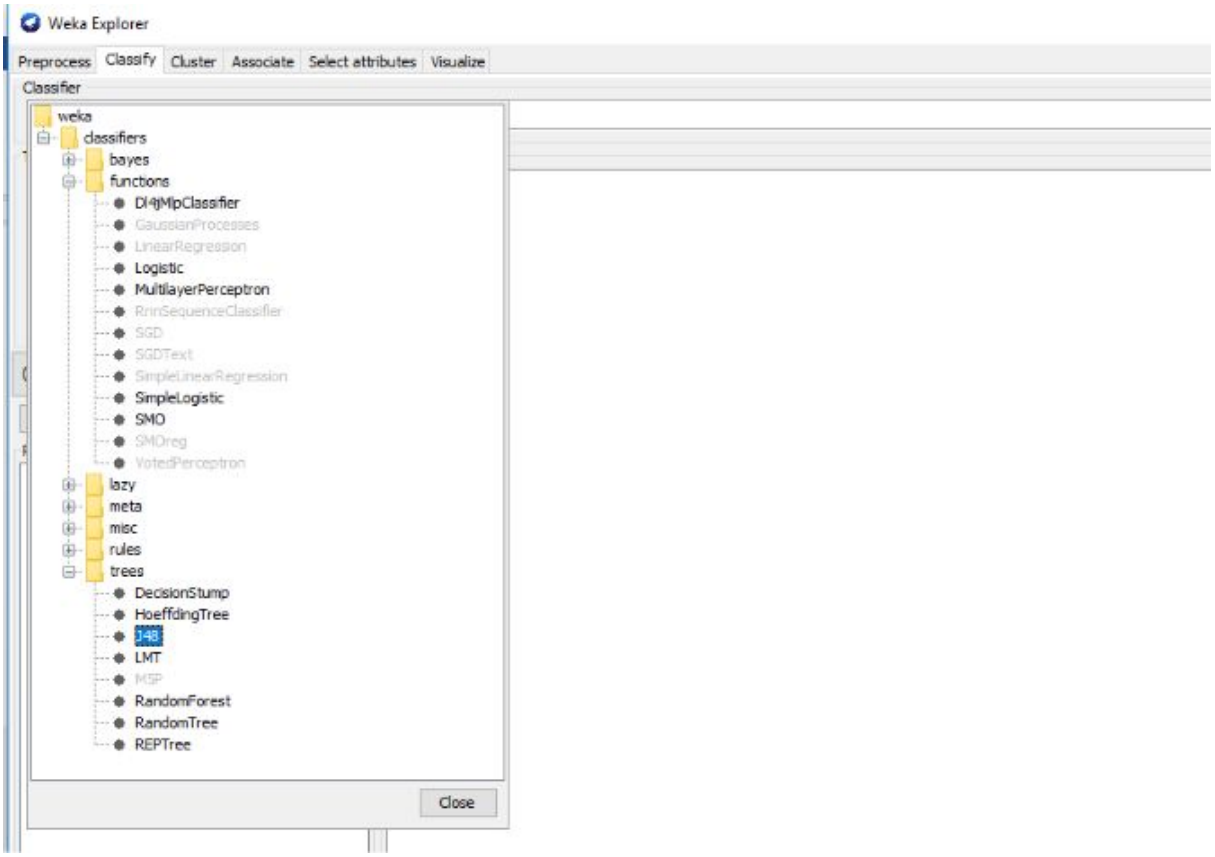
Πατάμε Open Files και επιλέγουμε το αρχείο CSV που έχουμε
Προσέχουμε να έχουμε επιλέξει και το Invoke options dialog ώστε να μπορούμε να ορίσουμε ποιος θα είναι ο διαχωριστικός χαρακτήρας του CSV. Τις περισσότερες φορές είναι η χαρακτήρας «κόμμα ‘,’ » αλλά και κάποιες το χαρακτήρα «ερωτηματικό ‘;’ ».



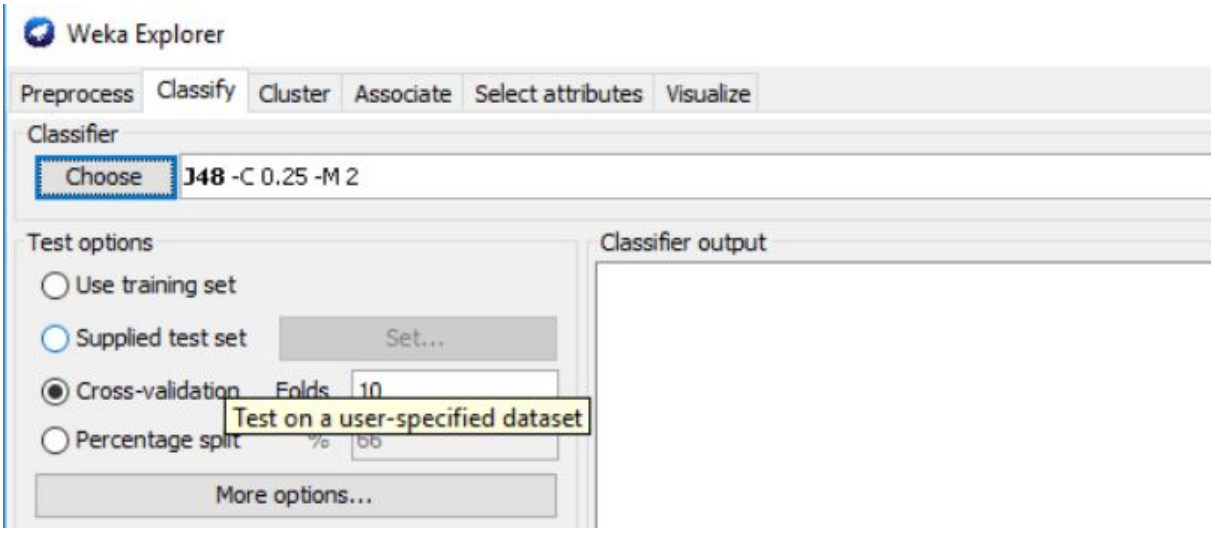
Το WEKA εισάγει το αρχείο μας.
Στην συνέχεια επιλέγουμε το φίλτρο Numeric to Nominal με στόχο να μετατρέψει σε κατηγορίες όλα τα αριθμητικά δεδομένα των διακοπτών.



Στην συνέχεια επιλέγουμε στο tab classify και επιλέγουμε το αλγόριθμο που θέλουμε μέσα από τους αλγόριθμους κατηγοριοποίησης που υπάρχουν.



Η επιλογή Multilayer Perceptron αφορά τα νευρωνικά δίκτυα Επίσης σημαντική είναι και η επιλογή Test Options όπου επιλέγουμε πως θα προκύψει το test σύνολο



Όπως βλέπουμε έχουμε τις επιλογές:

Use training set: που σημαίνει θα εξετάσουμε μόνο την απόδοση
Training set: δηλαδή την μέγιστη δυνατή απόδοση του συστήματος.
Supplied test set: που θα ορίσουμε ένα δικό μας σύνολο test που θα εφαρμοστεί μετά την εκτέλεση του αλγόριθμου με το training set, δηλαδή αφού έχουμε μια

πρώτη εκτέλεση με το training set.
Cross-validation: που αποτελεί την λεγόμενη διασταυρούμενη επικύρωση, μια τυποποιημένη τεχνική αξιολόγησης, που είναι ένας συστηματικός τρόπος εκτέλεσης επαναλαμβανόμενων ποσοστών διαχωρισμού. Δηλαδή ακολουθεί αν έχουμε πχ.. ένα cross-validation 10 την διαδικασία διαχωρίστε ένα σύνολο δεδομένων σε 10 κομμάτια (“Folds”), στη συνέχεια κρατήστε το κάθε κομμάτι με τη σειρά του και να δοκιμάσετε να εκπαιδεύσεις με τα υπόλοιπα 9 μαζί και κάνε χρήση το πρώτο κομμάτι σαν τεστ. Στην συνέχεια κάνει το ίδιο για ένα διαφορετικό κομμάτι από τα 10 και εκπαίδευση με τα άλλα 9 κ.ο.κ. Στο τέλος παίρνει σαν αποτέλεσμα την μέση απόδοση.

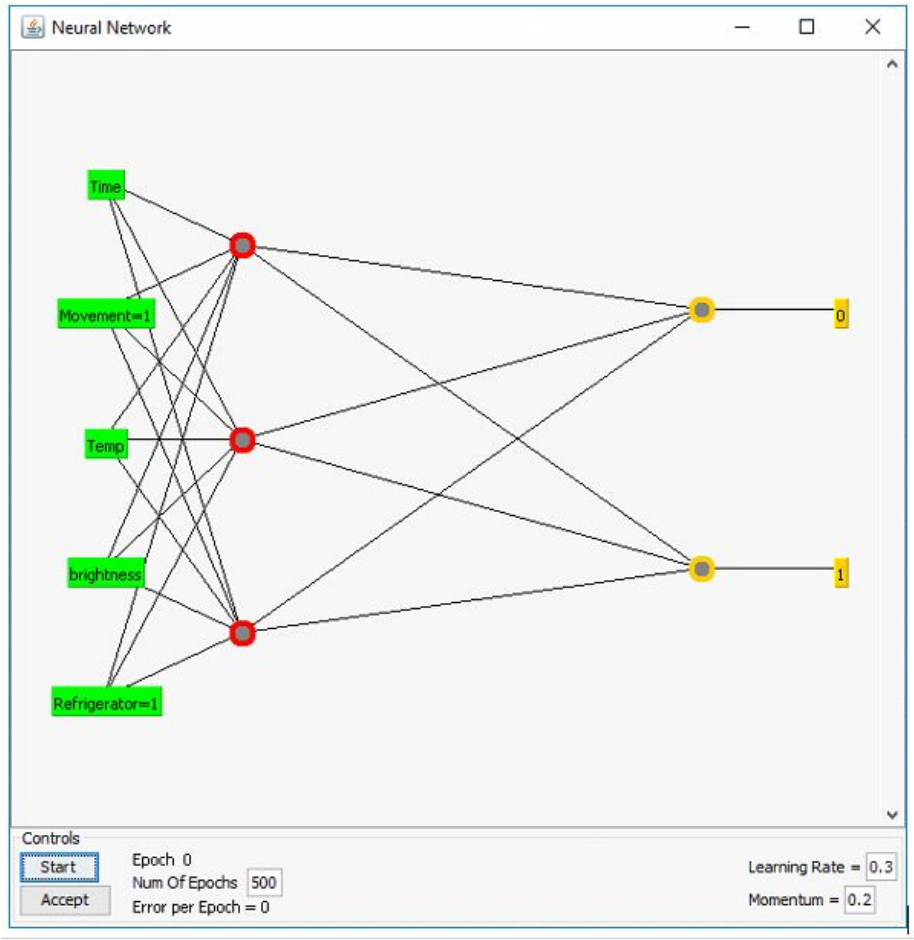
Percentage split: που δίνουμε εμείς ποιο θα είναι το ποσοστό του training και test συνόλου κάνοντας το σύστημα μια τυχαία επιλογή εγγραφών.

Στην συνέχεια εκτελουμε τον αλγόριθμο αλλάζοντας παραμέτρους του. Οι παράμετροι σε κάθε αλγόριθμο εμφανίζονται πατώντας πάνω στο όνομα του.

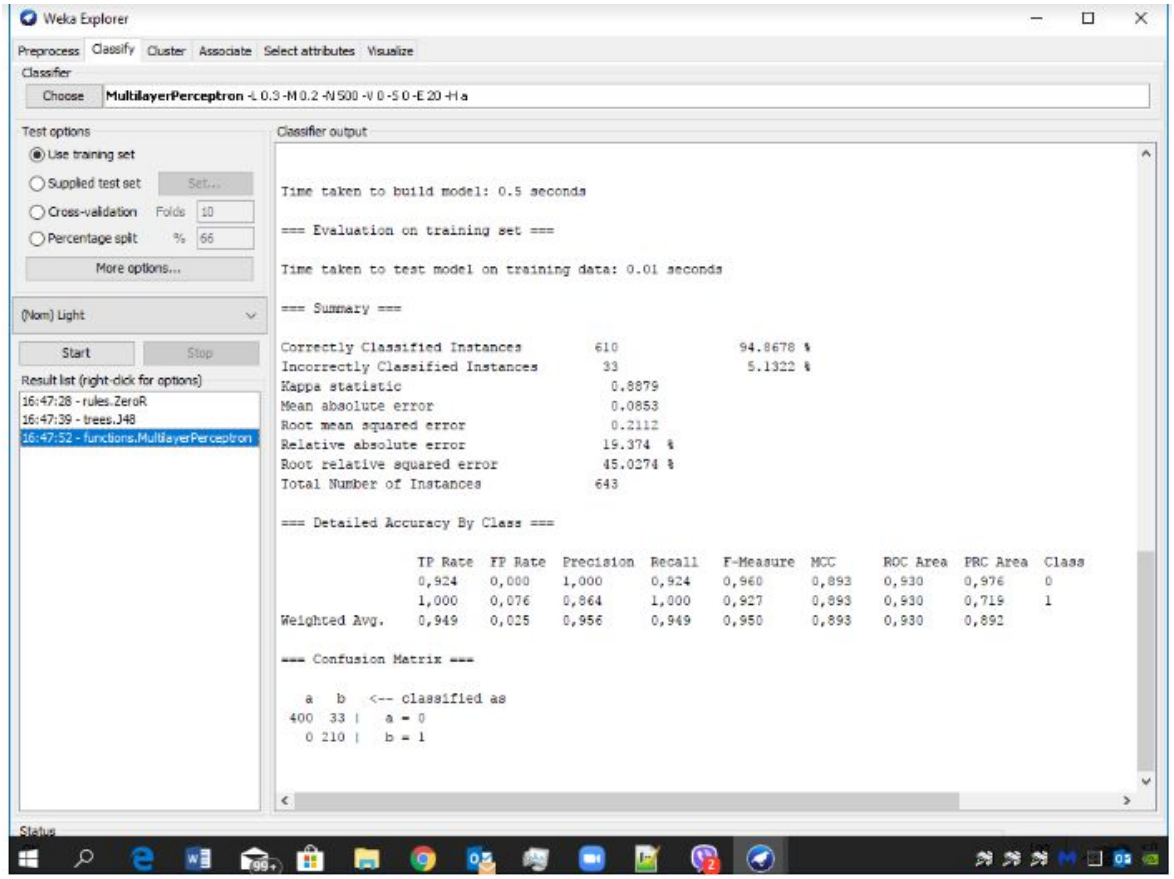
Αποτελέσματα

Στην περίπτωση μας έχουμε 3 νευρωνικά δίκτυα που αφορούν κάθε διακόπτη δηλαδή αυτό της τηλεόρασης, αυτό του θερμοσίφωνα και αυτό του φωτός.

Έτσι έχουμε πάρει ένα Νευρωνικό για το φως της παρακάτω μορφής:



Το αποτέλεσμα του Νευρωνικού είναι:



Οι συντελεστές του νευρωνικού είναι:

=== Classifier model (full training set) ===

Sigmoid Node 0
Inputs Weights
Threshold -2.0066490492104685
Node 2 3.006062757461911
Node 3 4.2402328052001375
Node 4 5.6167366256439015

Sigmoid Node 1
Inputs Weights
Threshold 2.006653397581374
Node 2 -2.9846809237898015
Node 3 -4.2688771622970805
Node 4 -5.603627381665341

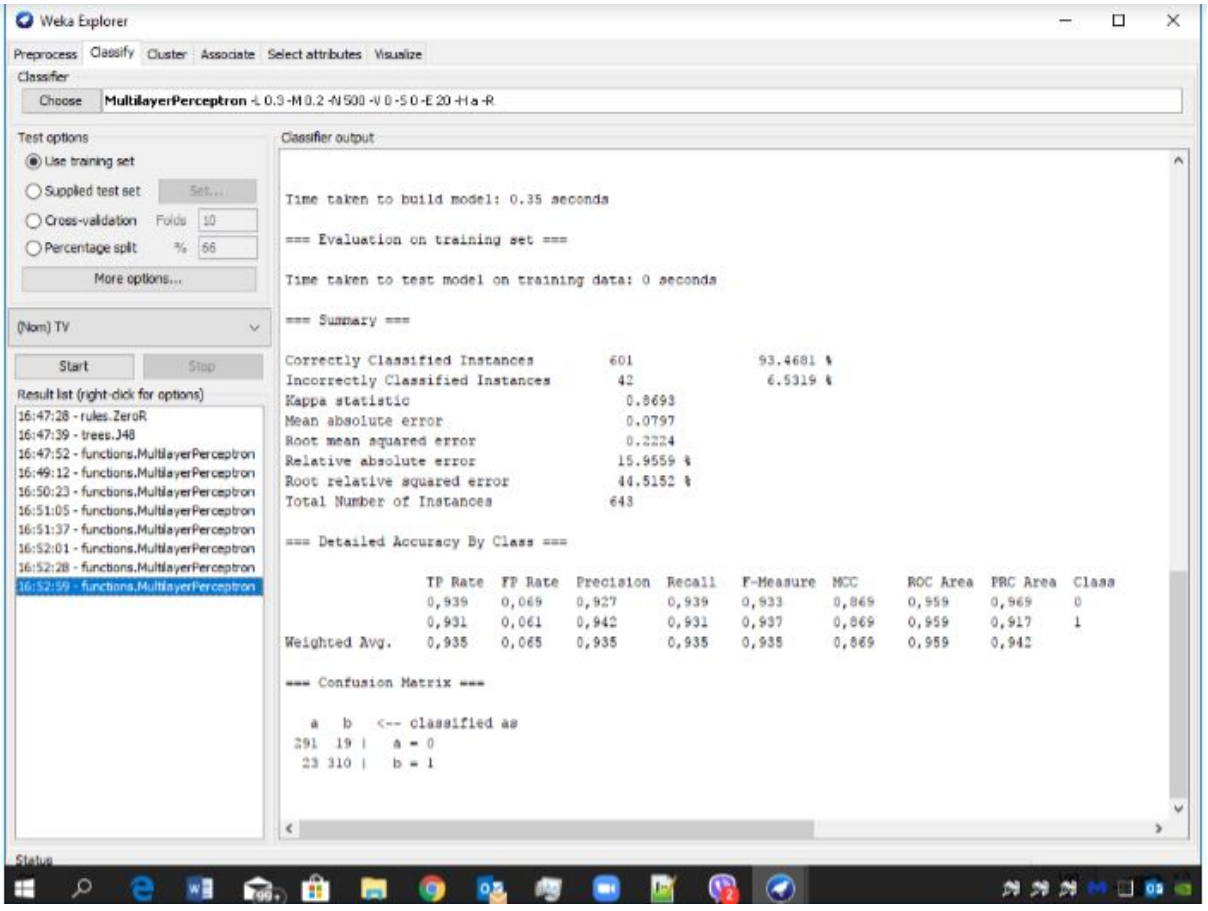
Sigmoid Node 2
Inputs Weights
Threshold -1.0393654006255888
Attrib Time -2.0456880481441186
Attrib Movement=1 -0.20798723551890386
Attrib Temp 0.4509098683295266
Attrib brightness 8.970500482149188
Attrib Refrigerator=1 -0.05337736780228678

Sigmoid Node 3
Inputs Weights
Threshold 0.08551309479693393
Attrib Time -2.28153266441175
Attrib Movement=1 -0.140008034677664
Attrib Temp 0.025700575914374357
Attrib brightness 11.188690361943149
Attrib Refrigerator=1 -0.13701306763814688

Sigmoid Node 4
Inputs Weights
Threshold 0.8203791807402567
Attrib Time -2.5172926000585125
Attrib Movement=1 -0.1280026236079387
Attrib Temp -0.11212737381900503
Attrib brightness 13.011877179866604
Attrib Refrigerator=1 -0.12928435871444433

Class 0
Input
Node 0
Class 1
Input
Node 1

Αντίστοιχα για την τηλεόραση έχουμε:



Οι συντελεστές του Νευρωνικού είναι:

Sigmoid Node 0
Inputs Weights
Threshold 1.043197830565829
Node 2 -5.222006526749453
Node 3 -4.840425985457626
Node 4 8.01887656294201

Sigmoid Node 1
Inputs Weights
Threshold -1.0431978290876078
Node 2 5.222006495812862

Node 3 4.840425979390269
Node 4 -8.018876505018673

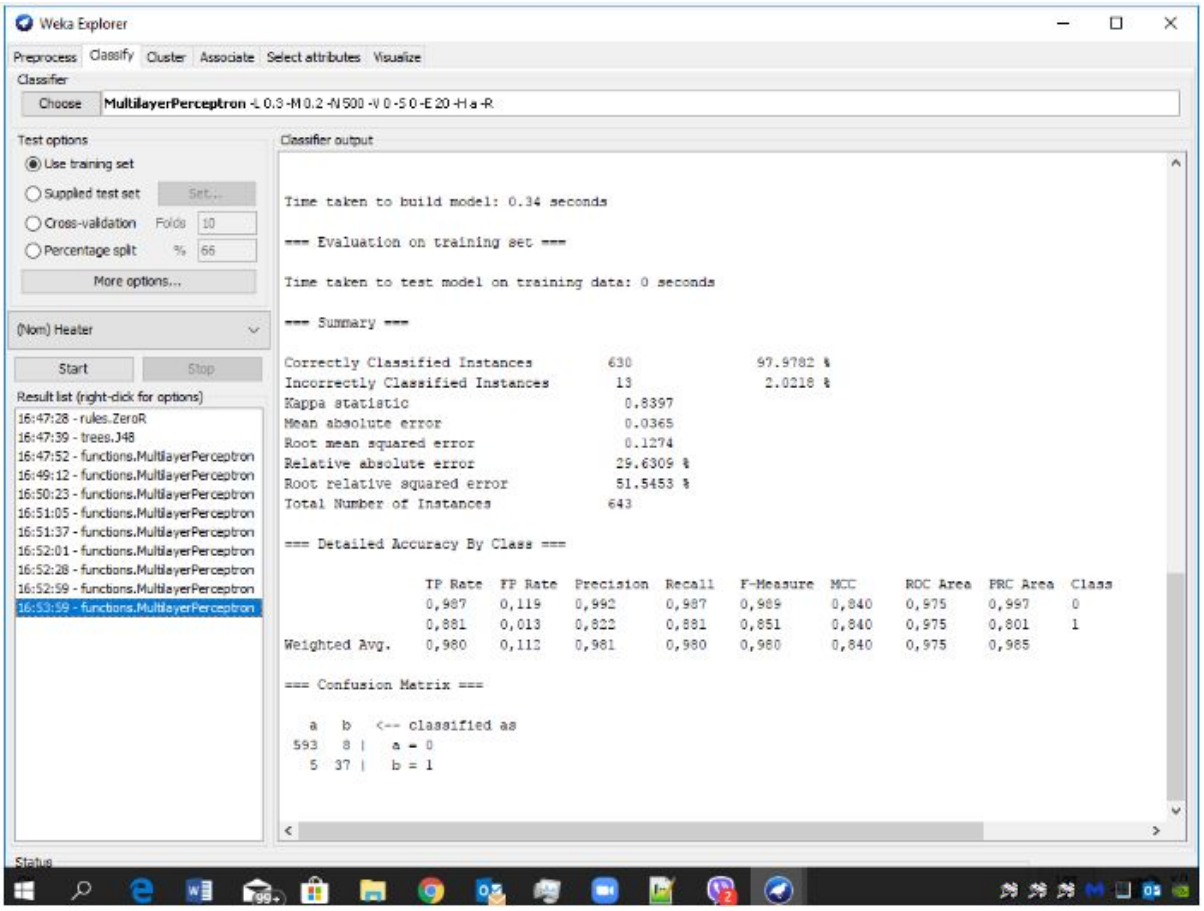
Sigmoid Node 2
Inputs Weights
Threshold 1.204729339766953
Attrib Time 17.213500116022477
Attrib Movement=1 0.19177398152106312
Attrib Temp 12.4654987781551
Attrib brightness -4.976419871438086
Attrib Refrigerator=1 0.21151356369437785

Sigmoid Node 3
Inputs Weights
Threshold -0.1189548939808462
Attrib Time 1.6516966554923198
Attrib Movement=1 0.6446321735481935
Attrib Temp -4.7829981526044
Attrib brightness -12.153149953236602
Attrib Refrigerator=1 0.13403374414348762

Sigmoid Node 4
Inputs Weights
Threshold -7.7509719603787115
Attrib Time -22.368945869172848
Attrib Movement=1 -0.12675062796469622
Attrib Temp 6.336599956732553
Attrib brightness -0.6995677109735561
Attrib Refrigerator=1 0.49108611048302886

Class 0
Input
Node 0
Class 1
Input
Node 1

Αντίστοιχα για τον θερμοσίφωνα έχουμε:



Sigmoid Node 0
Inputs Weights
Threshold -2.2651576163935583
Node 2 8.685629903650767
Node 3 3.353000549042919
Node 4 3.800313002615879

Sigmoid Node 1
Inputs Weights
Threshold 2.2651554529552747
Node 2 -8.68561312386247
Node 3 -3.3530021280610964
Node 4 -3.80030447938286

Sigmoid Node 2
Inputs Weights
Threshold -13.763039820470771
Attrib Time 5.575067920321341
Attrib Movement=1 -1.078046998270691
Attrib Temp -3.9980762007960284
Attrib brightness -12.884484109789721
Attrib Refrigerator=1 -0.39701453689535665

Sigmoid Node 3
Inputs Weights
Threshold 0.4411604388332498
Attrib Time -4.112412203226537
Attrib Movement=1 -0.07622682337950906
Attrib Temp 0.6243282068950987
Attrib brightness 8.482109150806489
Attrib Refrigerator=1 -1.4630767810116128

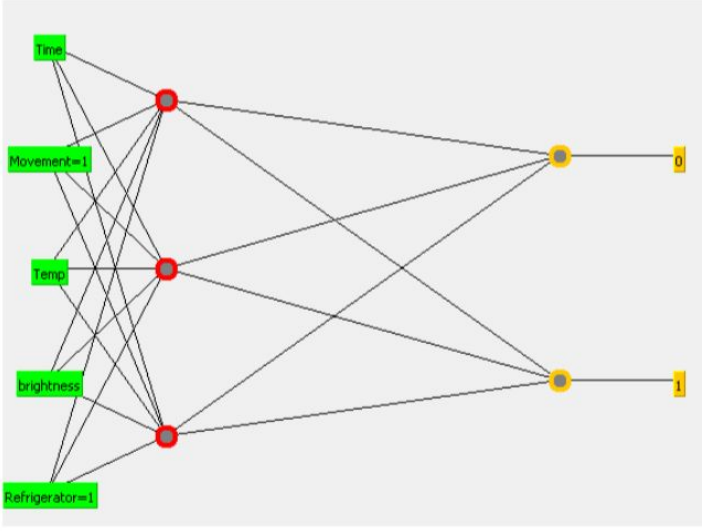
Sigmoid Node 4
Inputs Weights
Threshold -1.1667355522543597
Attrib Time -5.089650387986196
Attrib Movement=1 -1.5038460939140592
Attrib Temp -2.9378331043635075
Attrib brightness 9.794637362435576
Attrib Refrigerator=1 -0.2240207451008703

Class 0
Input
Node 0
Class 1
Input
Node 1

Δημιουργία Νευρώνα

Ένας νευρώνας ουσιαστικά είναι το στοιχείο εκείνο όπου έχει μια σειρά από εισόδους όπως ακριβώς προέκυψε στο νευρωνικό δίκτυο που εκπαιδεύσαμε προηγουμένως.

Το δίκτυο μας είναι το παρακάτω:



Στο επίπεδο της εισόδου έχουμε 3 νευρώνες όπου κάθε νευρώνας έχει 5 εισόδους και μια έξοδο. Η έξοδος από κάθε νευρώνα γίνεται είσοδος στους νευρώνες του επιπέδου της εξόδου. Στο επίπεδο εξόδου λοιπόν έχουμε 2 νευρώνες με 3 εισόδους και μία έξοδος που αποτελεί το αποτέλεσμα.

Ο κώδικας σε VHDL του πρώτου νεύρωνα είναι ο παρακάτω:

```
library ieee;

use ieee.numeric_std.all;

library ieee_proposed;
use ieee_proposed.fixed_pkg.all;
use IEEE.math_real.all;
USE ieee.numeric_std.ALL;


entity neuron1 is
	port(input1,input2,input3,input4,input5 : in  real ;
		weight1,weight2,weight3,weight4,weight5 : in real;
		threshold : in real;
		output : out real);
end entity neuron1;

architecture neuronFunction of neuron1 is
	signal i1,i2,i3,i4,i5 : real;
	signal w1,w2,w3,w4,w5 : real;
	signal th : real;
	signal p1,p2,p3,p4,p5 : real;
	signal sum : real;

	begin


				i1 <= input1;
				i2 <= input1;
				i3 <= input1;
				i4 <= input1;
				i5 <= input1;

				w1 <= weight1;
				w2 <= weight2;
				w3 <= weight3;
				w4 <= weight4;
```

```
                w5 <= weight5;

                th <= threshold;



        product : process (i1,i2,i3,i4,i5)
        begin
                p1 <= i1 * w1;
                p2 <= i2 * w2;
                p3 <= i3 * w3;
                p4 <= i4 * w4;
                p5 <= i5 * w5;

        end process product;


        summation : process (p1,p2,p3,p4,p5)
        begin
                sum <= p1+p2+p3+p4+p5;
        end process summation;


        sigmoid : process (sum)
        begin
                if sum>th then
                output <= real(1.0);
 else
 output <= real(0.0);
 end if ;
        end process sigmoid;

end neuronFunction;
```

Ο αντίστοιχος νεύρωνας δευτέρου επιπέδου είναι ο παρακάτω:

```
library ieee;

use ieee.numeric_std.all;
```

```
library ieee_proposed;
use ieee_proposed.fixed_pkg.all;
use IEEE.math_real.all;
USE ieee.numeric_std.ALL;


entity neuron2 is
	port(input1,input2,input3 : in  real ;
		weight1,weight2,weight3: in real;
		threshold : in real;
		output : out real);
end entity neuron2;

architecture neuronFunction of neuron2 is
	signal i1,i2,i3 : real;
	signal w1,w2,w3 : real;
	signal th : real;
	signal p1,p2,p3 : real;
	signal sum : real;

	begin


				i1 <= input1;
				i2 <= input1;
				i3 <= input1;


				w1 <= weight1;
				w2 <= weight2;
				w3 <= weight3;


				th <= threshold;



		product : process (i1,i2,i3)
		begin
			p1 <= i1 * w1;
			p2 <= i2 * w2;
			p3 <= i3 * w3;
```

```
        end process product;


        summation : process (p1,p2,p3)
        begin
                sum <= p1+p2+p3;
        end process summation;


        sigmoid : process (sum)
        begin
                if sum<th then
                output <= real(1);
    else
    output <= real(0);
    end if ;
        end process sigmoid;

    end neuronFunction;
```

Τελικώς το κύκλωμά μας είναι το παρακάτω:

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_arith.all;
use IEEE.std_logic_unsigned.all;
use IEEE.math_real.all;
USE ieee.numeric_std.ALL;
library ieee_proposed;
use ieee_proposed.fixed_pkg.all;

entity testbench is
-- empty
end testbench;

architecture tb of testbench is

component neuron1 is
    port(input1,input2,input3,input4,input5 : in  real ;
         weight1,weight2,weight3,weight4,weight5 : in real;
```

```
                threshold : in real;
                output : out real);
end component;

component neuron2 is
port(input1,input2,input3 : in  real ;
                weight1,weight2,weight3: in real;
                threshold : in real;
                output : out real);
end component;

signal i1,i2,i3,i4,i5: real;
signal w11,w12,w13,w14,w15,th1,o11: real;
signal w21,w22,w23,w24,w25,th2,o12: real;
signal w31,w32,w33,w34,w35,th3,o13: real;
signal w41,w42,w43,th4,o1: real;
signal w51,w52,w53,th5,o2: real;
begin

  -- Connect DUT
  D1: neuron1 port map(i1,i2,i3,i4,i5,w11,w12,w13,w14,w15,th1,o11);
  D2: neuron1 port map(i1,i2,i3,i4,i5,w21,w22,w23,w24,w25,th2,o12);
  D3: neuron1 port map(i1,i2,i3,i4,i5,w31,w32,w33,w34,w35,th3,o13);
  D4: neuron2 port map(o11,o12,o13,w41,w42,w43,th4,o1);
  D5: neuron2 port map(o11,o12,o13,w51,w52,w53,th5,o2);




  process
  begin


   i1<=1.0;
   i2<=13.68;
   i3<=0.8;
   i4<=1.0;
   i5<=1.0;

    w11<=-2.0456880481441186;
    w12<=-0.20798723551890386;
    w13<=0.4509098683295266;
    w14<=8.970500482149188;
    w15<=-0.05337736780228678;
```

```
        th1<=-1.0393654006255888;

    w21<=-2.28153266441175;
    w22<=-0.140008034677664;
    w23<=0.025700575914374357;
    w24<=11.188690361943149;
    w25<=-0.13701306763814688;
        th2<=0.08551309479693393;

    w31<=-2.5172926000585125;
    w32<=-0.1280026236079387;
    w33<=-0.11212737381900503;
    w34<=13.011877179866604;
    w35<=-0.12928435871444433;
        th3<=0.8203791807402567;



    w41<=3.006062757461911;
    w42<=4.2402328052001375;
    w43<=5.6167366256439015;

        th4<=0.8203791807402567;


    w51<=-2.9846809237898015;
    w52<=-4.2688771622970805;
    w53<=-5.603627381665341;

        th5<=2.006653397581374;

wait for 1ns;

   i1<=1.0;
   i2<=10.68;
   i3<=0.8;
   i4<=1.0;
   i5<=1.0;

  report "O1:" & to_string(o11);


    wait;
```

```
  end process;
end tb;
```

Παίρνοντας τις μετρήσεις μας παρατηρούμε όπως ήταν λογικό τις ίδιες τιμές με αυτές που δίνει το νευρωνικό δίκτυο που εξετάσαμε.

Βιβλιογραφία

► Norbert Wiener, *Κυβερνητική και κοινωνία, η ανθρώπινη χρησιμοποίηση των ανθρώπινων όντων*, μετ. Γιάννη Ιωαννίδη, Εκδόσεις Παπαζήση, Αθήνα 1970.
► Τεχνητά Νευρωνικά Δίκτυα, Διαμαντάρας Κωνσταντίνος, 2007
► Μηχανική Μάθηση, Μπότσης Δημήτριος, Διαμαντάρας Κωνσταντίνος, 2019
► J.A. Anderson, An Introduction to Neural Networks, MIT Press, Cambridge, 1995
► Churchland & Sejnowski (1992)
► (http://kelifos.physics.auth.gr/COURSES/neural/K4.pdf)
► Better Deep Learning: Train Faster, Reduce Overfitting, and Make Better Predictions, Jason Brownlee, 2018
► Master Machine Learning Algorithms: discover how they work and implement them from scratch, Jason Brownlee, 2016
► Deep Learning Tutorial Release 0.1 LISA lab, University of Montreal September 01, 2015
► Understanding Machine Learning: From Theory to Algorithms c 2014 by Shai Shalev-Shwartz and Shai Ben-David Published 2014 by Cambridge University Press.