Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών Ε.Μ.Π.

ΝΕΥΡΩΝΙΚΑ ΔΙΚΤΥΑ ΚΑΙ ΕΥΦΥΗ ΥΠΟΛΟΓΙΣΤΙΚΑ ΣΥΣΤΗΜΑΤΑ

2016-2017

ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ 1

Ευαγγελία-Σοφία Γεργατσούλη
ΑΜ: 03112064
Νικηφόρος Μανδηλαράς
ΑΜ: 03112012
Εξάμηνο:9ο

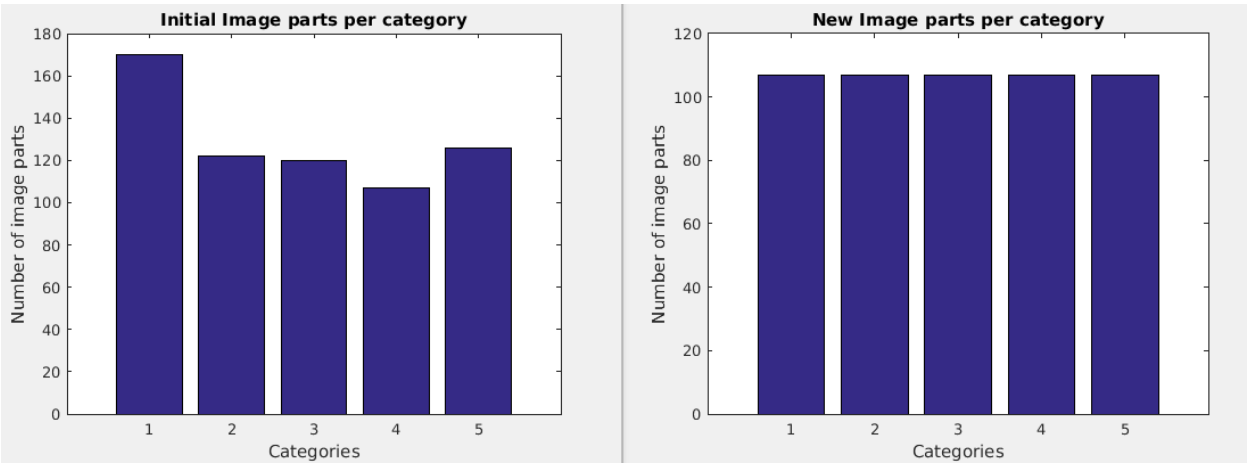
Ερώτημα 1 - Προεπεξεργασία των δεδομένων

Αρχικά γίνεται μια προεπεξεργασία των δεδομένων για να γίνει σωστότερα η εκπαίδευση του δικτύου μας. Πιο συγκεκριμένα, θέλουμε να κρατήσουμε ίδιο αριθμό δειγμάτων ανα κατηγορία για να γίνει ίση εκπαίδευση σε κάθε περίπτωση κατηγορίας. Επιπλέον χρειαζόμαστε κανονικοποίηση των δεδομένων ώστε να επιταχύνεται η διαδικασία μάθησης στον αλγόριθμο back propagation και να αποφεύγονται οι περιπτώσεις ”ταλάντωσης” κατα την διάδοση των αλλαγών στα βάρη. Για την επεξεργασία ακολουθήσαμε τα εξής βήματα:

1. Αφού φορτώθηκαν τα δεδομένα εισόδου, βρήκαμε με την εντολή sum πόσα τμήματα εικόνας περιέχει κάθε κατηγορία το οποίο είναι απλώς το άθροισμα της κάθε γραμμής του πίνακα TrainDataTargets για κάθε μια από τις 5 διαφορετικές γραμμές-κατηγορίες. Το αποτέλεσμα φαίνεται στο αριστερό bar διαγραμμα της εικόνας **??**. Όπως φαίνεται, δεν υπάρχει ο ίδιος αριθμός τμημάτων εικόνας για κάθε κατηγορία.

2. Στη συνέχεια, για την διευκόλυνση της εκπαίδευσης του δικτύου, περνάμε τα δεδομένα απο επεξεργασία με τις removeconstantrows, mapstd και processpca. Η πρώτη αφαιρεί από τον πίνακα τις γραμμές που έχουν σταθερή τιμή (οι οποίες ουσιαστικά είναι άχρηστες). Η mapstd κανονικοποιεί τα δεδομένα, έτσι ώστε κάθε γραμμή του πίνακα να έχει μέση τιμή 0 και variance 1 και η processpca κρατάει έναν μικρότερο αριθμό γραμμών του πίνακα σε περίπου 20 χωρίς να χάσει σημαντική πληροφορία (για να γίνει αυτό δίνουμε τιμή 0.01 στην παράμετρο maxfrac).

Τελικά μετά την επεξεργασία παίρνουμε το δεξί bar διαγραμμα της εικόνας **??**. Η συνάρτηση γι αυτό το κομμάτι είναι η *preproc.m*.

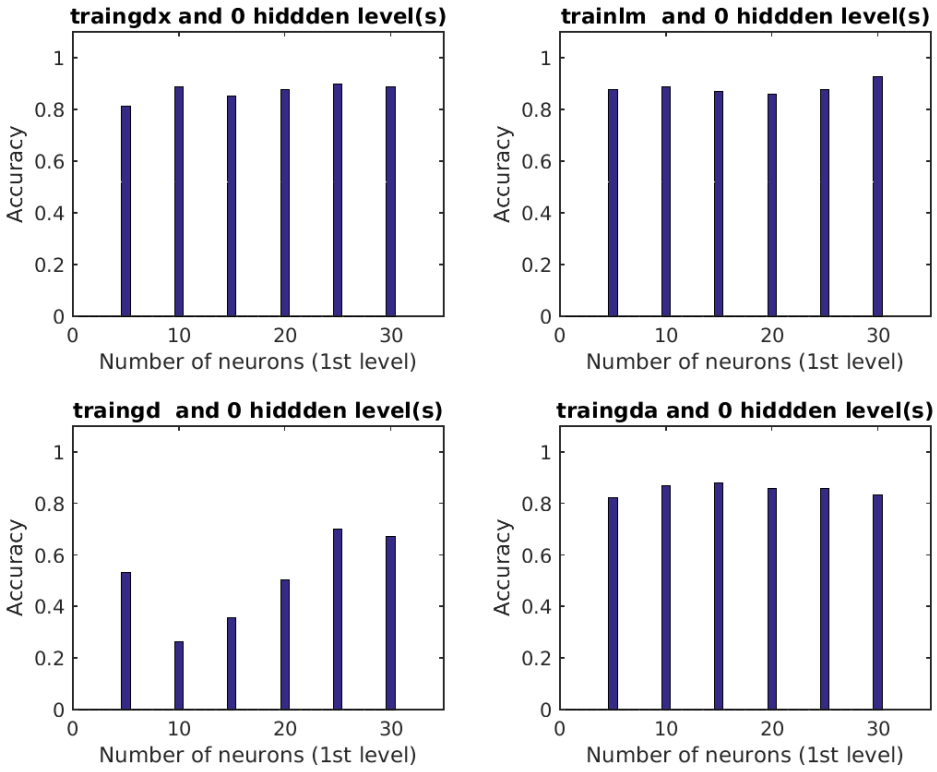


Εικόνα 1

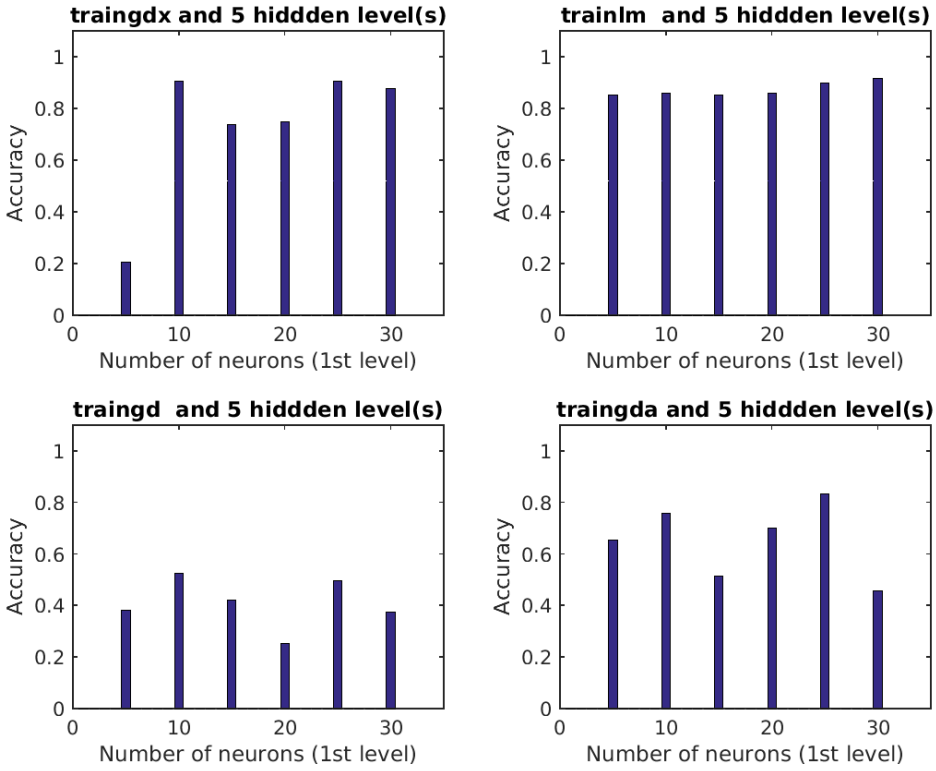
Ερώτημα 2 - Διαφορετικές συναρτήσεις εκπαίδευσης και αριθμός επιπέδων και νευρώνων

Καθώς η αρχικοποίηση γίνεται τυχαία όταν γίνεται η προεπεξεργασία των δεδομένων, κάθε φορά που φτιάχνουμε ένα νεο νευρωνικό δίκτυο, και μετράμε την απόδοση του δε θα παίρνουμε πάντα το ίδιο αποτέλεσμα. Μετά απο μετρήσεις σε πολλά διαφορετικά ”τρεξίματα” καταλήξαμε στο ότι η καλύτερη συνάρτηση εκπαίδευσης είναι η trainlm (η οποία υλοποιεί τον αλγόριθμο Levenberg-Marquardt), καθως είχε στην πλειοψηφία των περιπτώσεων καλύτερη απόδοση από όλες τις άλλες για κάθε συνδυασμό νευρώνων 1ου και 2ου επιπέδου. Η αμέσως επόμενη καλύτερη ήταν η traingdx και σπανίως έβγαινε η traingda. Η traingd είχε πάντα

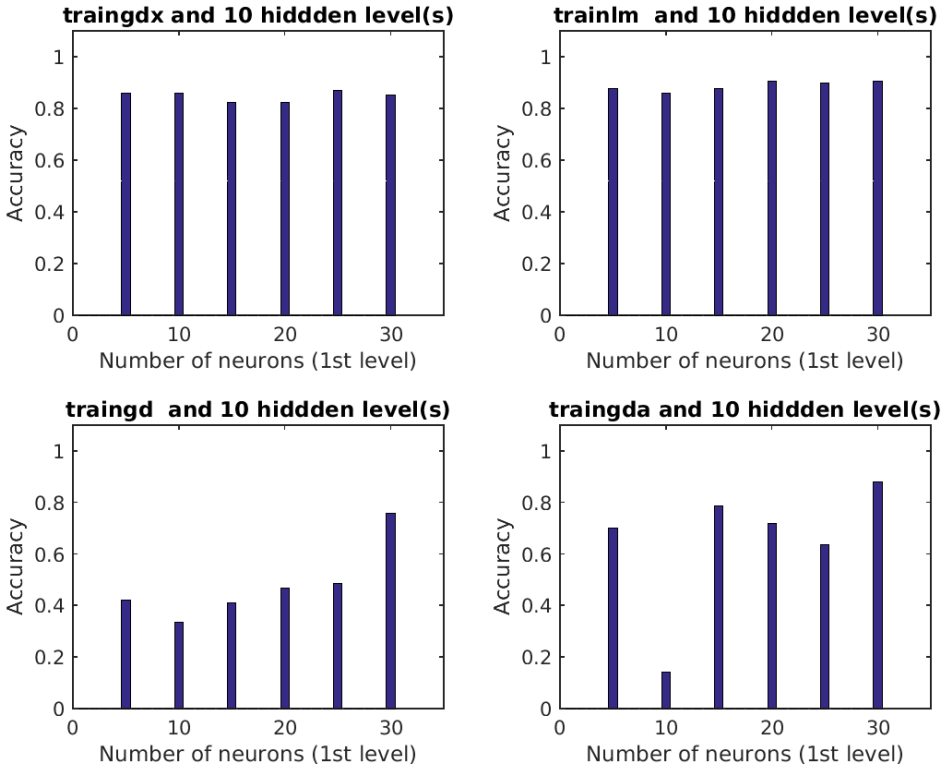
τη χειρότερη απόδοση απ' όλες. Στις εικόνες **??-??** φαίνεται η απόδοση για κάθε συνάρτηση εκπαίδευσης και για κάθε συνδυασμό νευρώνων 1ου και 2ου επιπέδου.
Αρχικά παρατηρούμε ότι περισσότεροι νευρώνες δεν συνεπάγονται και καλύτερη απόδοση καθώς όσο πιο πολύ "εξειδικεύεται" το δίκτυο υπάρχει ο κίνδυνος να μην μπορεί να γενικεύσει και απλώς να "θυμάται" τα δεδομένα εκπαίδευσης, δηλαδή να έχουμε overfitting.
Τέλος παρατηρήθηκε ότι τα καλύτερα αποτελέσματα τα παίρνουμε για λίγους εως καθόλου νευρώνες στο 2ο επίπεδο και μέτριο αριθμό νευρώνων 1ου επιπέδου. Συνγεκριμένα ο συνδυασμός 30 νευρώνες 1ου επιπέδου και 0 νευρώνες στο 2ο επίπεδο βγήκε νικητής τις περισσότερες φορές, άρα θα χρησιμοποιήσουμε αυτόν στα επόμενα ερωτήματα. Να σημειωθεί επίσης ότι η trainlm ήταν και αυτή που έτρεχε γρηγορότερα από τις υπόλοιπες, με πιο αργή την traingdx. Η συνάρτηση γι αυτό το κομμάτι είναι η *myTrain.m*.



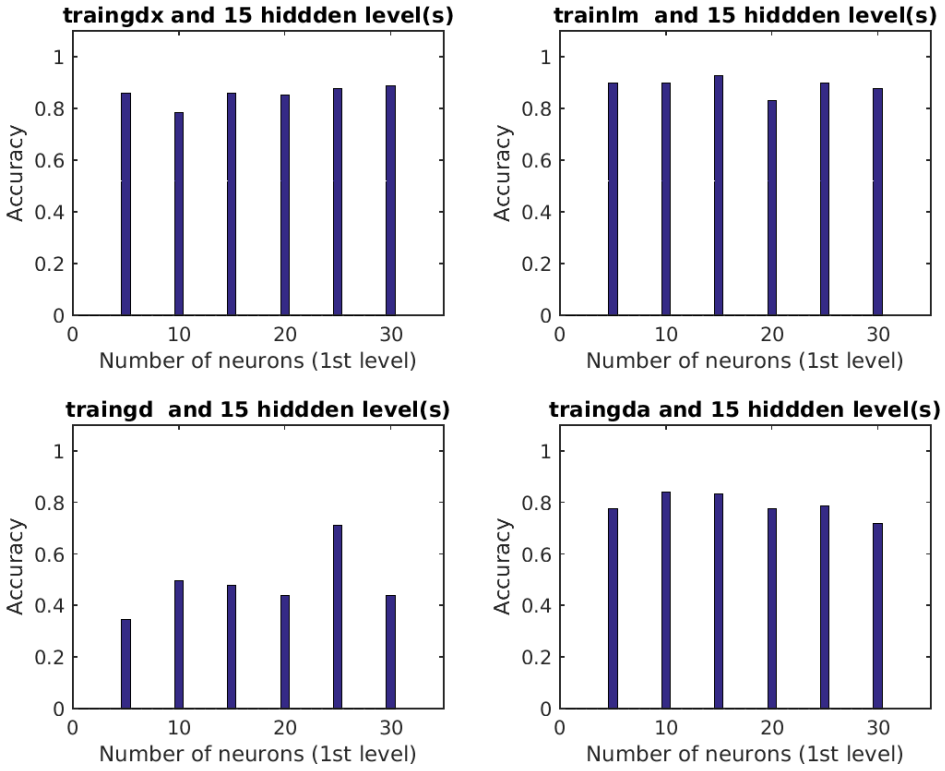
Εικόνα 2: Απόδοση χωρίς 2ο κρυφό επίπεδο



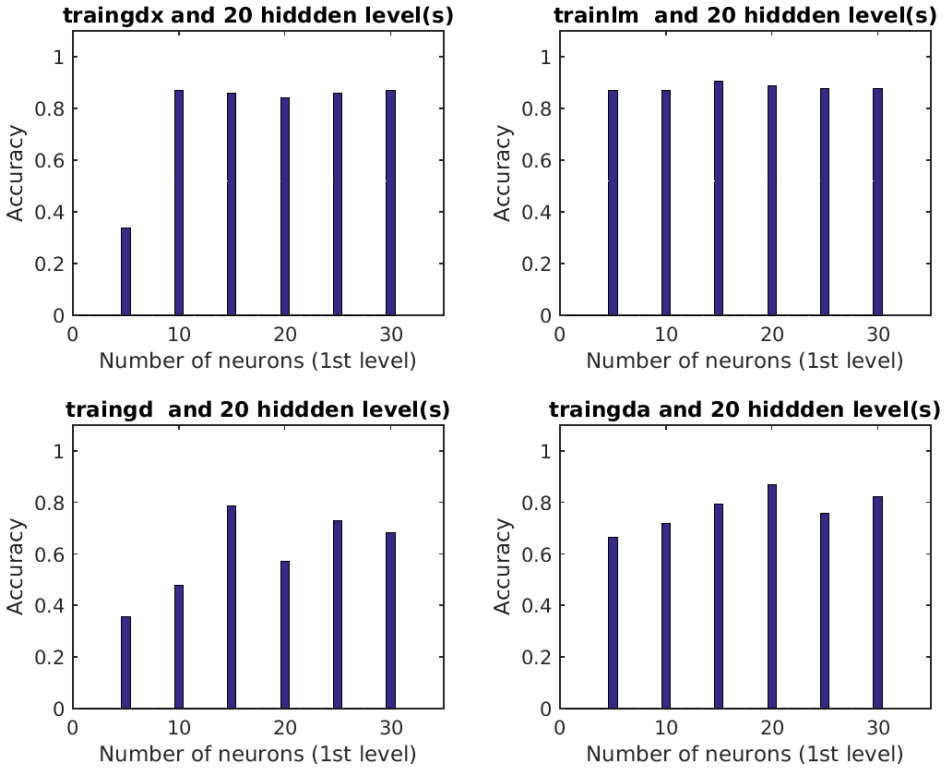
Εικόνα 3: Απόδοση με 5 νευρώνες στο 2ο κρυφό επίπεδο



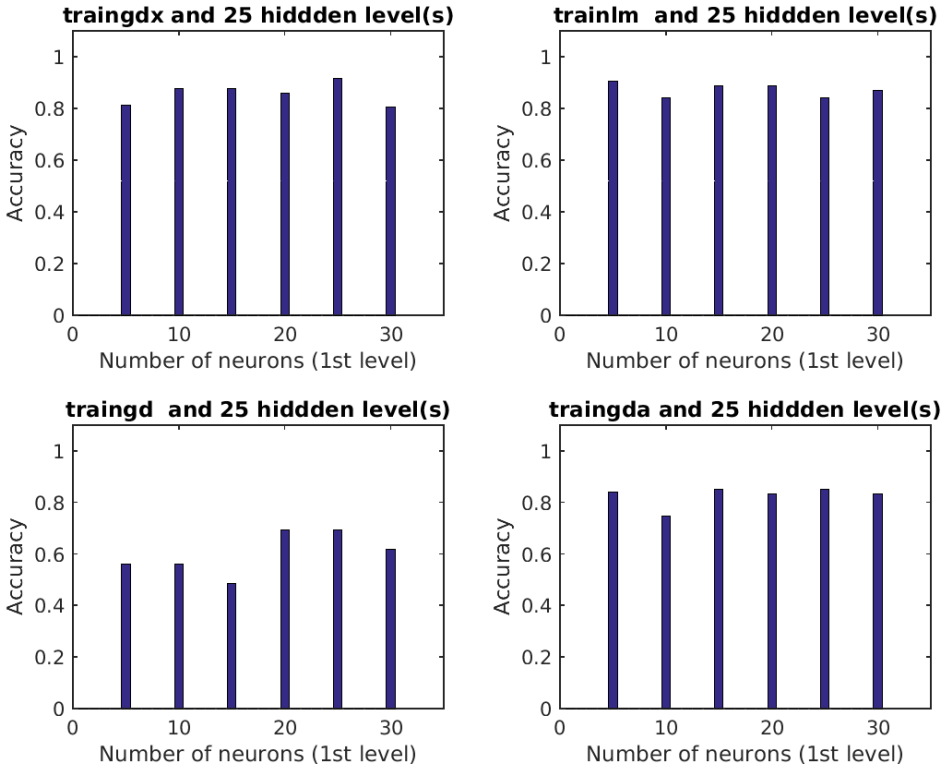
Εικόνα 4: Απόδοση με 10 νευρώνες στο 2ο κρυφό επίπεδο



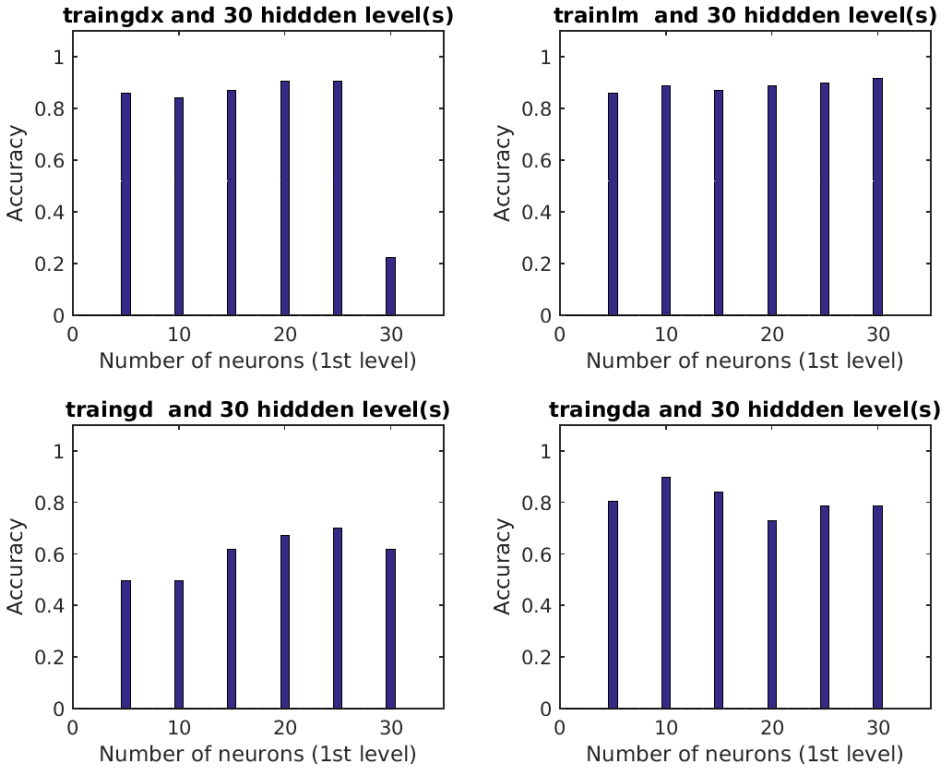
Εικόνα 5: Απόδοση με 15 νευρώνες στο 2ο κρυφό επίπεδο



Εικόνα 6: Απόδοση με 20 νευρώνες στο 2ο κρυφό επίπεδο



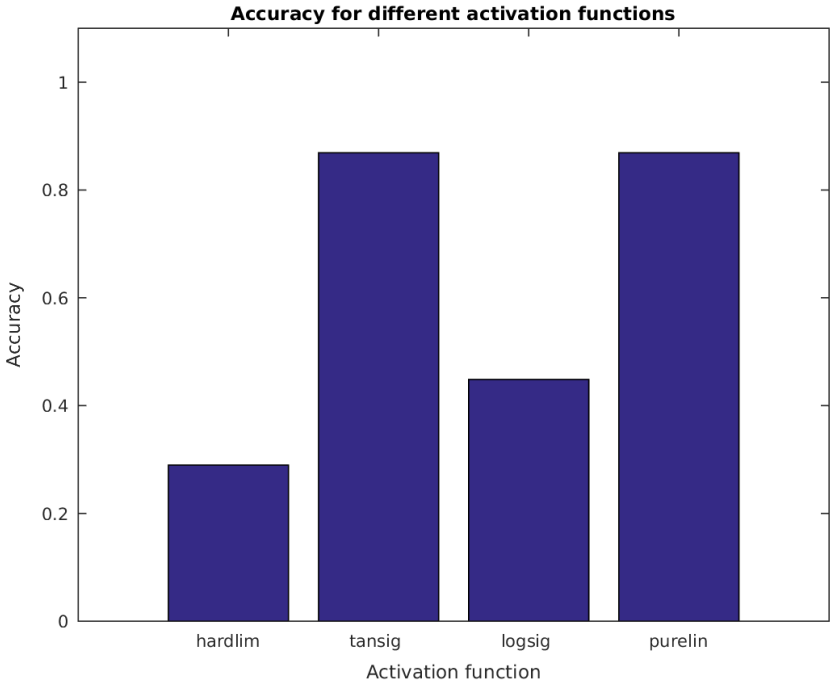
Εικόνα 7: Απόδοση με 25 νευρώνες στο 2ο κρυφό επίπεδο



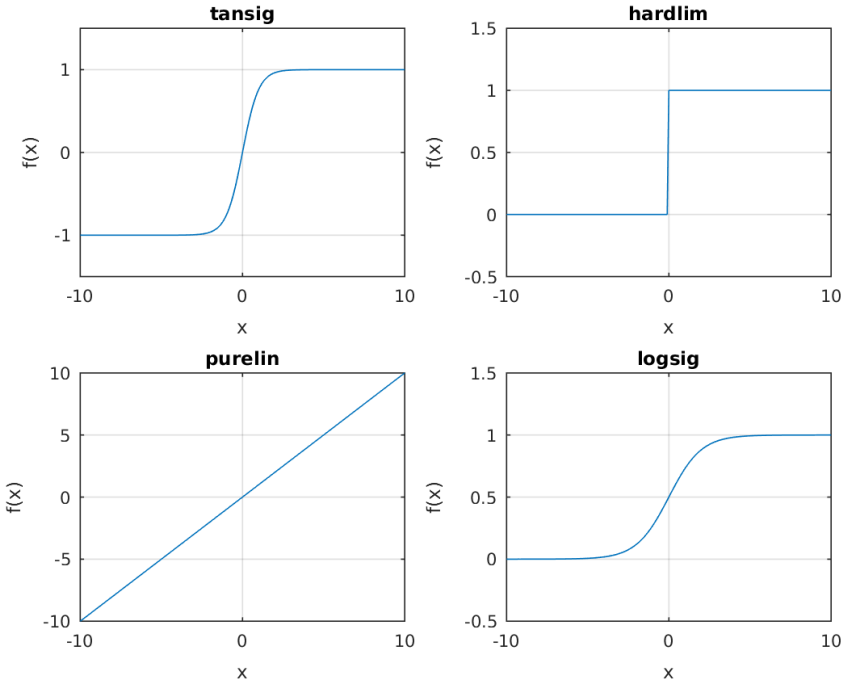
Εικόνα 8: Απόδοση με 30 νευρώνες στο 2ο κρυφό επίπεδο

Ερώτημα 3 - Συνάρτηση ενεργοποίησης

Στο ερώτημα αυτό συγκρίθηκε η απόδοση για 4 διαφορετικές συναρτήσεις ενεργοποίησης. Η purelin είναι απλώς μια γραμμική συνάρτηση, η tansig είναι συνάρτηση σιγμοειδούς υπερβολικής εφαπτομένης, η logsig λογαριθμική σιγμοειδής και η hardlim είναι hard limit. Οι γραφικές τους παραστάσεις φαίνονται στην εικόνα **??** και τα αποτελέσματα για το accuracy στην εικόνα **??**. Αρχικά παρατηρούμε ότι οι hardlim και logsig έχουν τη χειρότερη απόδοση από τις 4, κάτι που είναι λογικό καθώς τα δεδομένα μας έχουν μέση τιμή 0 αλλά οι δύο αυτές συναρτήσεις έχουν πεδίο τιμών το $[0, 1]$. Η συνάρτηση που κάνει τις γραφικές για τις διαφορετικές περιπτώσεις ειναι η *plotActFunc.m*.



Εικόνα 9: Αποδόση για διαφορετικες συναρτησεις ενεργοποίησης



Εικόνα 10: Οι συναρτήσεις ενεργοποίησης

Ερώτημα 4 - Αλγορίθμος μάθησης

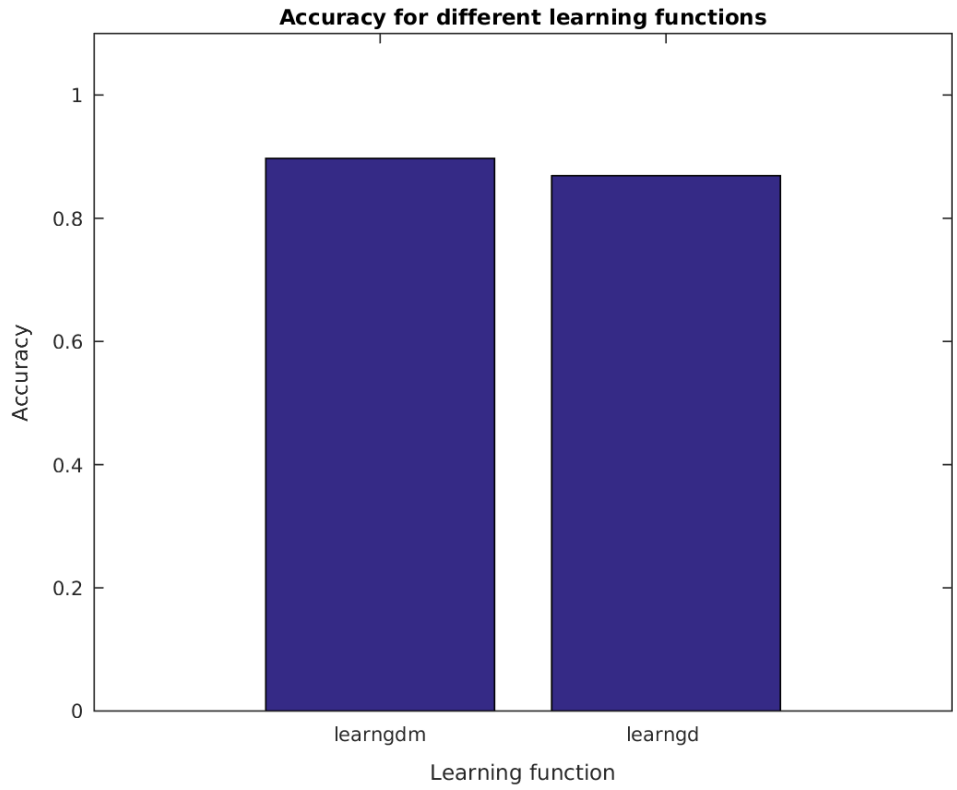
Συγκρίθηκαν οι αλγόριθμοι μάθησης learngd και learngdm. Ο πρώτος αποτελεί την κλασσική υλοποίηση του backpropagation και υπολογίζει το correction στα βάρη ως εξής:

$$\Delta w_{ij}(n) = \eta \delta_j(n) y_i(n) \tag{1}$$

όπου η είναι η παράμετρος μάθησης και δ_j είναι το local gradient, και $y_i(n)$ η είσοδος. Στην περίπτωση του learngdm έχουμε έναν επιπλέον όρο ορμής, οπότε το correction θα είναι:

$$\Delta w_{ij}(n) = \alpha \Delta w_{ij}(n-1) + (1-\alpha)\eta \delta_j(n) y_i(n) \tag{2}$$

όπου το α είναι η σταθερά ορμής.
Στην συνάρτηση learngd υπάρχει πρόβλημα όταν μειωθεί πολύ η παράμετρος μάθησης καθώς ενώ θα έχουμε ομαλές μεταβάσεις στον χώρο των βαρών, λόγω του ότι τα βάρη θα αυξάνονται με αργό ρυθμό, ο ρυθμός μάθησης θα είναι αργός. Αντίθετως, όταν ο ρυθμός αυξηθεί πολύ, οι αλλαγές στα βάρη θα είναι πολύ απότομες και υπάρχει περίπτωση να γίνεται ”ταλάντωση” γύρω απο το σημείο που θέλουμε να φτάσουμε και να έχουμε ένα ασταθές δίκτυο. Η learngdm δεν αφήνει τόσο απότομες αλλαγες, και ”μετριάζει” την ταλάντωση. Αυτό έχει ως αποτέλεσμα να συγκλίνει πιο αργά αλλά να φτάνει στο επιθυμητο αποτέλεσμα με καλύτερη ακρίβεια σε αντίθεση με την learngd που είναι αρκετά πιο γρήγορη. Στο δικό μας δίκτυο, η learngd είναι οριακά καλύτερη, αλλά λόγω όσων αναφέρθηκαν παραπάνω, θα χρησιμοποιούσαμε την learngdm. Το αποτέλεσμα στο accuracy φαίνεται στην εικόνα **??**.



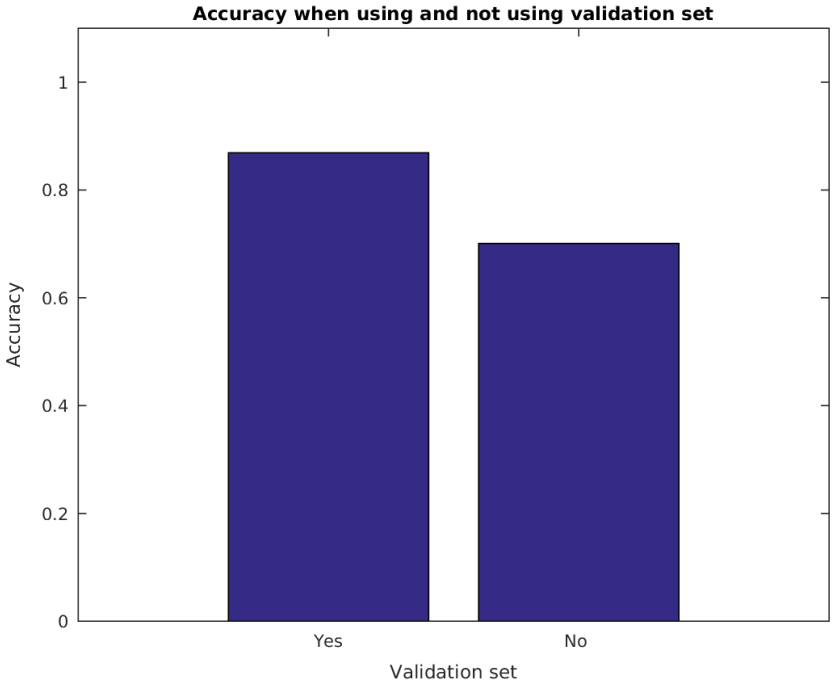
Εικόνα 11: Αποδόση για διαφορετικούς αλγορίθμους μάθησης

Η συνάρτηση για το συγκεκριμένο ερώτημα είναι η *plotLearnFunc.m*. Το τρέξαμε για τις default (1000) εποχές.

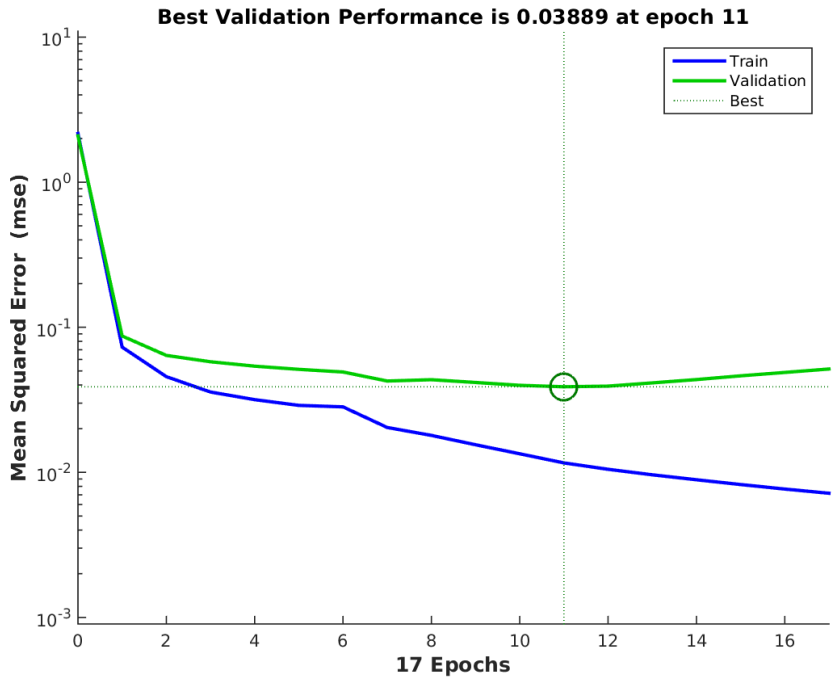
Ερώτημα 5 - Early Stopping

Σε αυτό το ερώτημα εξετάζεται η απόδοση του νευρωνικού με validation set και χωρίς αυτό. Η τεχνική του early stopping λειτουργεί ως εξής: τα δεδομένα εκπαίδευσης χωρίζονται σε 2 ομάδες, μια ομάδα με δεδομένα εκπαίδευσης και τα υπόλοιπα είναι το validation set. Κατα τη διάρκεια της εκπαίδευσης του δικτύου (χρησιμοποιώντας μόνο την πρώτη ομάδα δεδομένων), ελέγχεται συνεχώς το σφάλμα στη δευτερη ομάδα. Τυπικά, το σφάλμα στο validation set στην αρχή θα μειώνεται καθώς το δίκτυο εκπαιδεύεται, αλλά κάποια στιγμή θα φτάσει σε ένα minimum και θα αρχίσει να αυξάνεται πάλι. Αν συνεχίσει να αυξάνει για συγκεκριμένο αριθμό επαναλήψεων, η εκπαίδευση σταματάει, για να αποφευχθεί το overfitting, και επιστρέφονται τα βάρη που είχαν βρεθεί για το minimum του σφάλματος.

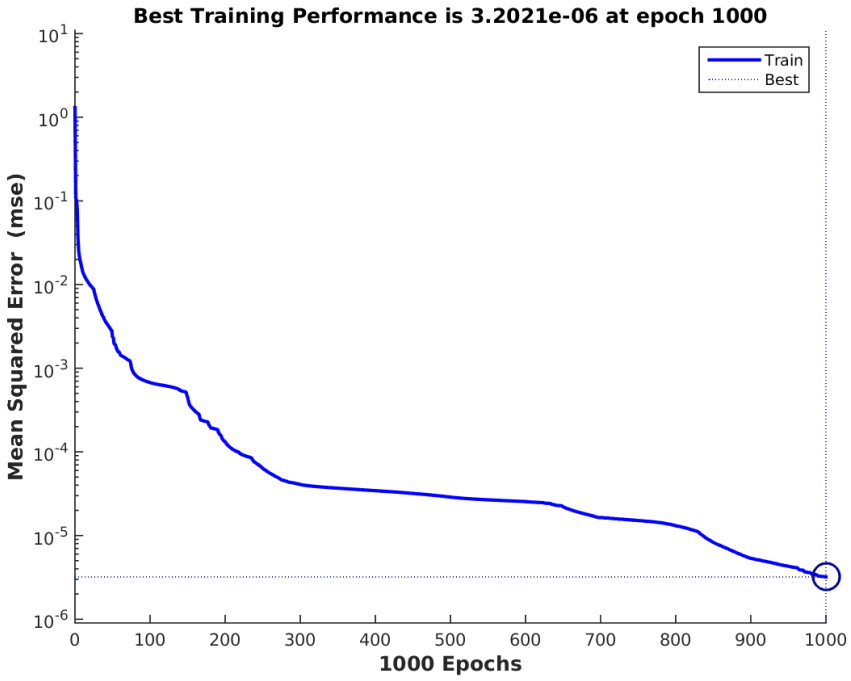
Στα μέχρι τώρα ερωτήματα, το 20% χρησιμοποιούνταν ως validation set. Στη δική μας περίπτωση βλέπουμε ότι με validation set σταματήσαμε στις 7 εποχές, ενώ τελικά το accuracy του δικτύου ήταν αρκετα καλύτερο. Παρατηρούμε επίσης ότι δε σταματάει αμέσως μόλις το σφάλμα στο Validation set γίνει πιο μεγάλο, καθώς μπορεί να είναι ενα τοπικό μέγιστο και να συνεχίσει να πέφτει μετά (όπως στην εικόνα **??**), ενώ ακριβώς επειδή σταματάει νωρίτερα, η μέθοδος αυτή θα είναι συνήθως αρκετά γρηγορότερη απο το να μην χρησιμοποιούσαμε validation set, όπου και θα έπρεπε να κάνει (εδω) 1000 εποχές για την πλήρη εκπαίδευση.



Εικόνα 12: Αποδόση με και χωρίς validation set



Εικόνα 13: Μέσο σφάλμα με validation set

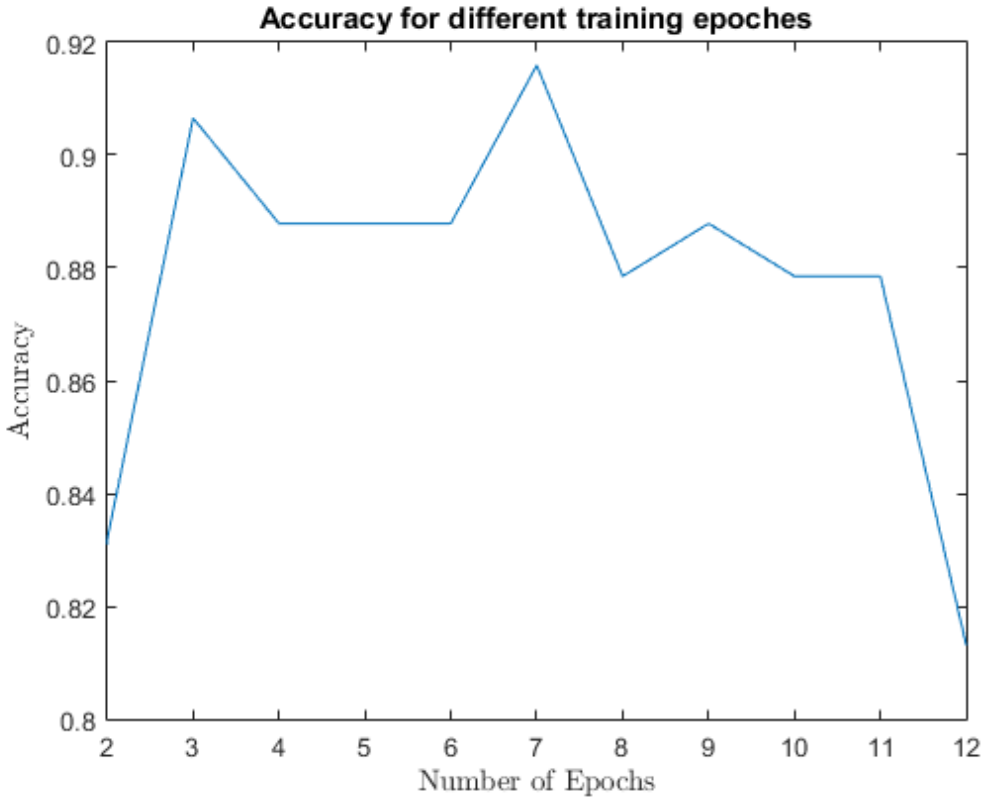


Εικόνα 14: Μέσο σφάλμα χωρίς validation set

Ο κώδικας για την παραγωγή των γραφικών του ερωτήματος είναι στο *plotValSet.m*.

Ερώτημα 6 - Αριθμός εποχών

Σε αυτό το ερώτημα θα προσπαθήσουμε να εκτιμήσουμε το πως επιδρά το διαφορετικό πλήθος εποχών στην απόδοση. Παίρνουμε τιμές σε ένα εύρος 5 γύρω από την τιμή που βρήκαμε στο προηγούμενο ερώτημα. Περιμένουμε καλύτερη απόδοση όσο το πλήθος των εποχών παραμένει κοντά στην επικρατέστερη τιμή που ήταν το 7. Παρ' όλα αυτά λόγω των αποκλίσεων των αποτελεσμάτων ανά εκτέλεση, περιμένουμε διαφοροποιήσεις. Επίσης λόγω της μικρής τιμής για το καλύτερο πλήθος εποχών εκπαίδευσης έχουμε αναγκαστικά και μικρό εύρος στο οποίο αναζητάμε τώρα διαφοροποιήσεις. Σ' αυτό το ερώτημα κριτήριο τερματισμού δεν υπήρχε και κάθε κύκλος εκπαίδευσης ολοκληρωνόταν έπειτα από το συγκεκριμένο αριθμό εποχών. Παραθέτουμε έπειτα το διάγραμμα Accuracy συναρτήσει του αριθμού των εποχών.

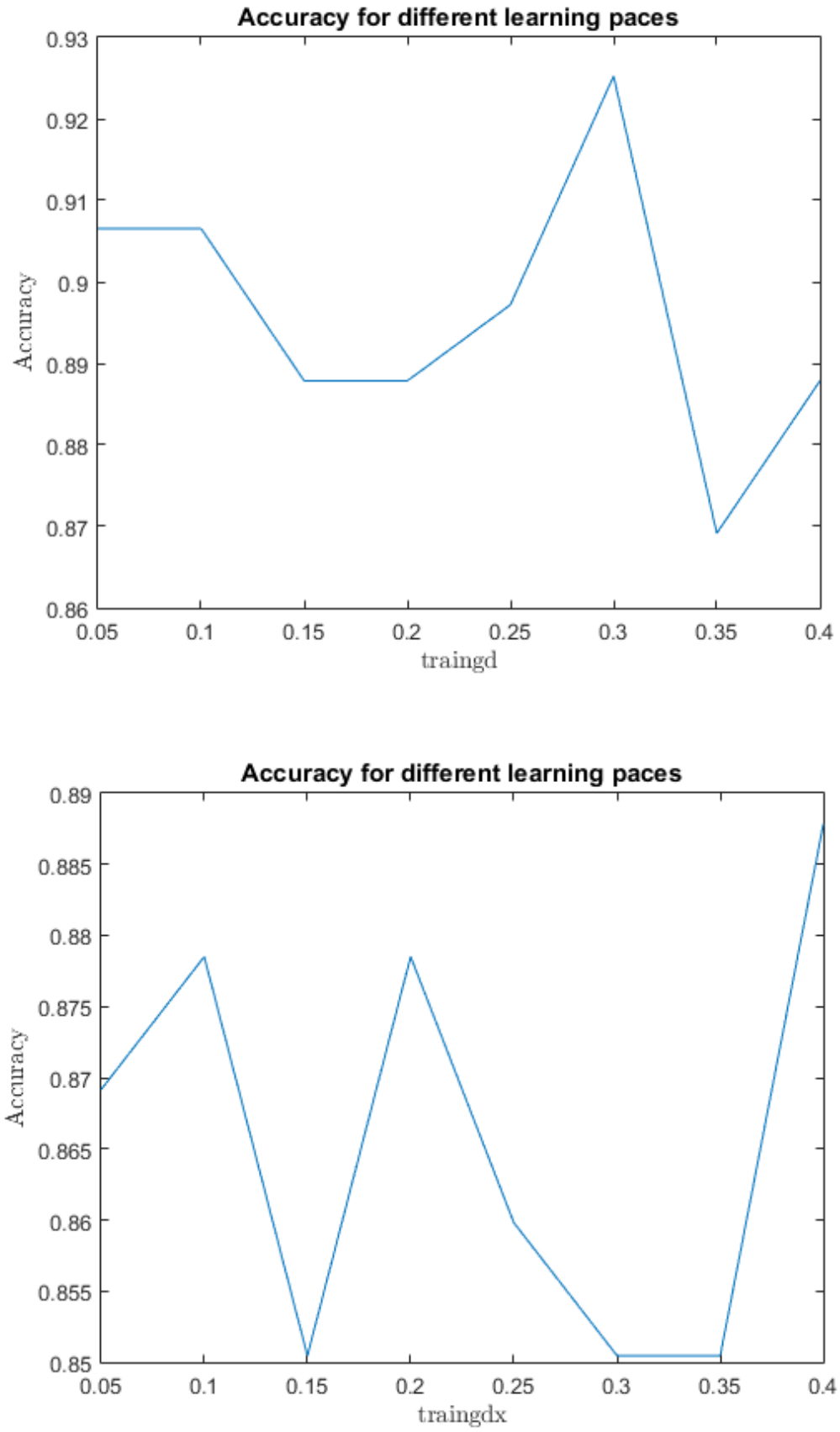


Εικόνα 15: Accuracy συναρτήσει του αριθμού εποχών

Πράγματι το παραπάνω διάγραμμα ακολουθεί την προσδοκόμενη συμπεριφορά. Η συνάρτηση για το συγκεκριμένο ερώτημα είναι η *epochRange.m*.

Ερώτημα 7 - Ρυθμός μάθησης

Μεγάλοι ρυθμοί μάθησης μπορεί να οδηγήσουν σε μη σωστή εκπαίδευση καθώς είναι πιθανό να αγνοηθούν τοπικά ελάχιστα είτε να παρατηρήσουμε 'ταλαντώσεις'. Απ' την άλλη όμως επιτυγχάνουμε καλύτερη ταχύτητα εκπαίδευσης. Ο μικρός ρυθμός αποτελεί πρόβλημα όταν είναι εξαιρετικά χαμηλός οπότε και παρατηρούνται φαινόμενα υπερεκπαίδευσης.



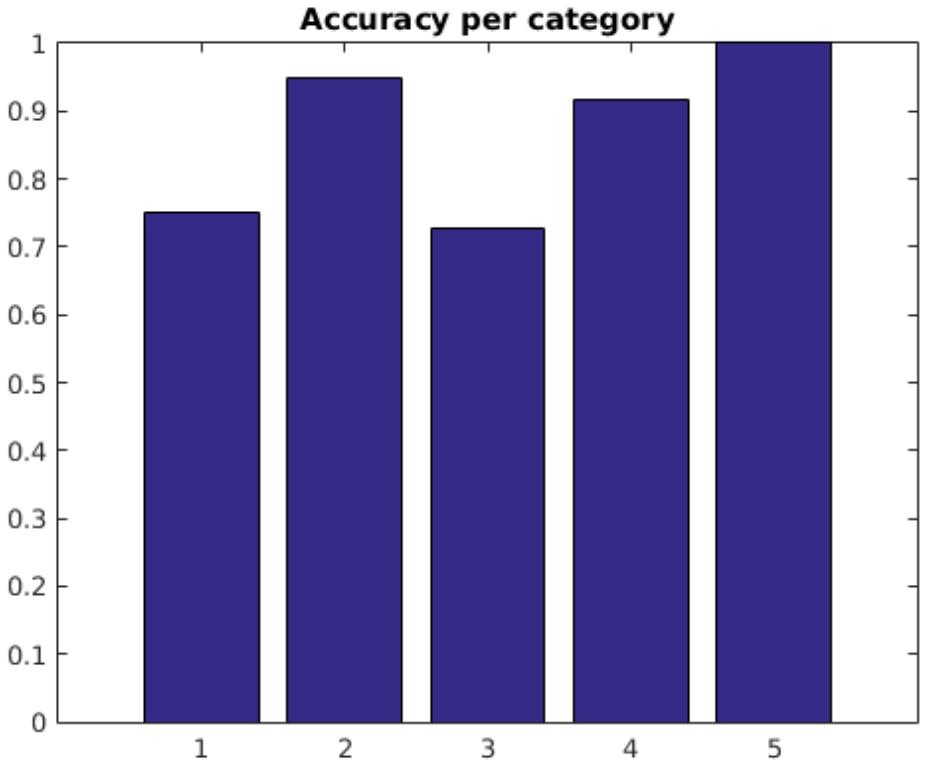
Εικόνα 16: Αποδόση για διαφορετικούς αλγορίθμους μάθησης

Η traingd όπως και αναμενόταν δεν επηρεάζεται ουσιαστικά από την αλλαγή του ρυθμού μάθησης αφού

δεν διαθέτει και όρο ορμής. Αντίθετα η traingdx φαίνεται να παρουσιάζει ασταθή συμπεριφορά ως προς αυτή την παράμετρο. Τέλος η απόδοση της traingd για τους διάφρορους ρυθμούς μάθησης είναι συνολικά μεγαλύτερη από αυτή της traingdx, συνεπώς μεταξύ των δύο θα επιλέγαμε την traingd. Για το συγκεκριμένο ερώτημα ο κώδικας βρίσκεται στο αρχείο *stepLearning.m*. Η εκπαίδευση του δικτύου έγινε για 30 χιλιάδες εποχές.

Ερώτημα 8- Απόδοση ανά κατηγορία

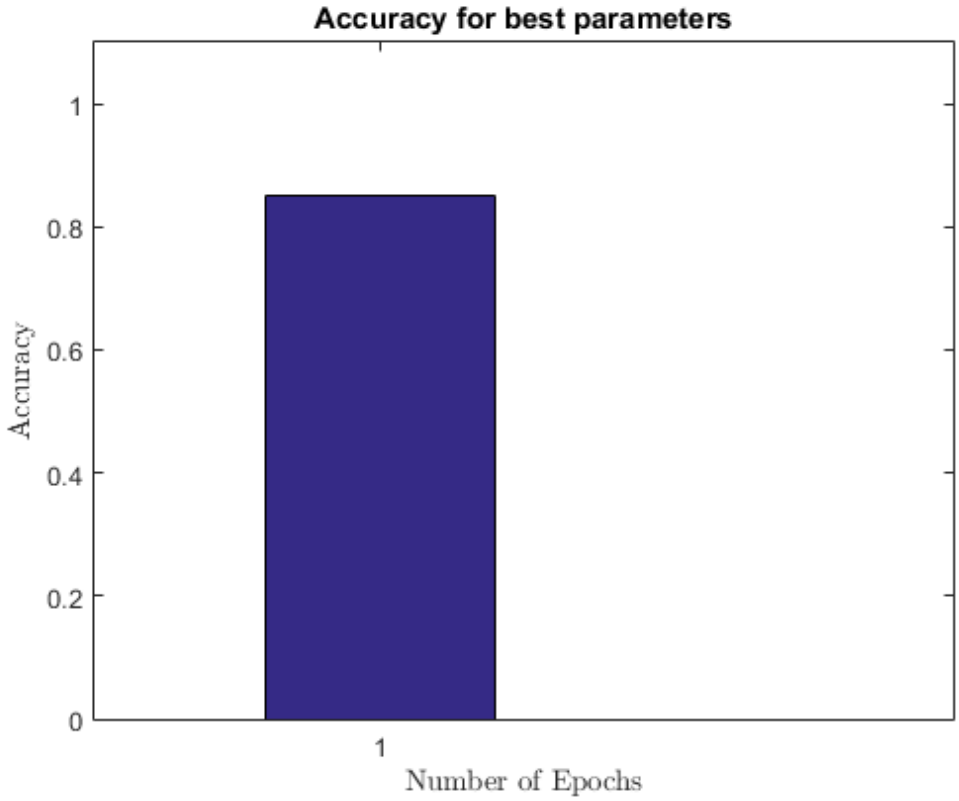
Είναι γεγονός πως παρατηρούμε αποκλίσεις μεταξύ των αποδόσεων, διαφορετικών κατηγοριών. Πιθανότατα οι συγκεκριμένες αποκλίσεις έχουν να κάνουν με το γεγονός ότι διαφορετικές κατηγορίες παρουσιάζουν και διαφορετική δυσκολία στην αναγνώριση τους. Ένας παράγοντας μπορεί να είναι ομοιότητες που παρουσιάζουν μεταξύ τους οι εικόνες δύο κατηγοριών. Ένας τρόπος αντιμετώπισης αυτού του γεγονότος θα ήταν η υπεραντιπροσώπευση αυτών των κατηγοριών στο δείγμα εκπαίδευσης με στόχο την καλύτερη αποσαφήνιση των μεταξύ τους διαφορών.



Εικόνα 17: Αποδόση για κάθε κατηγορία

Επιμύθιο

Συγκεντρωτικά με βάση τα συμπεράσματα των παραπάνω διερευνήσεων είμαστε πλέον σε θέση να προσδιορίσουμε τις κατάλληλες παραμέτρους για την βέλτιστη λειτουργία του νευρωνικού μας δικτύου. Επιλέξαμε ένα επίπεδο με 30 Νευρώνες, καλύτερη συνάρτηση εκπαίδευσης προέκυψε η trainglm, ενώ για συνάρτηση ενεργοποίησης επιλέξαμε την tansing. Ο αλγόριθμος μάθησης που θα χρησιμοποιήσουμε είναι ο learngd και τέλος θα κάνουμε χρήση του validation set. Παραθέτουμε στη συνέχεια την αποδοτικότητα της συγκεκριμένης υλοποίησης.

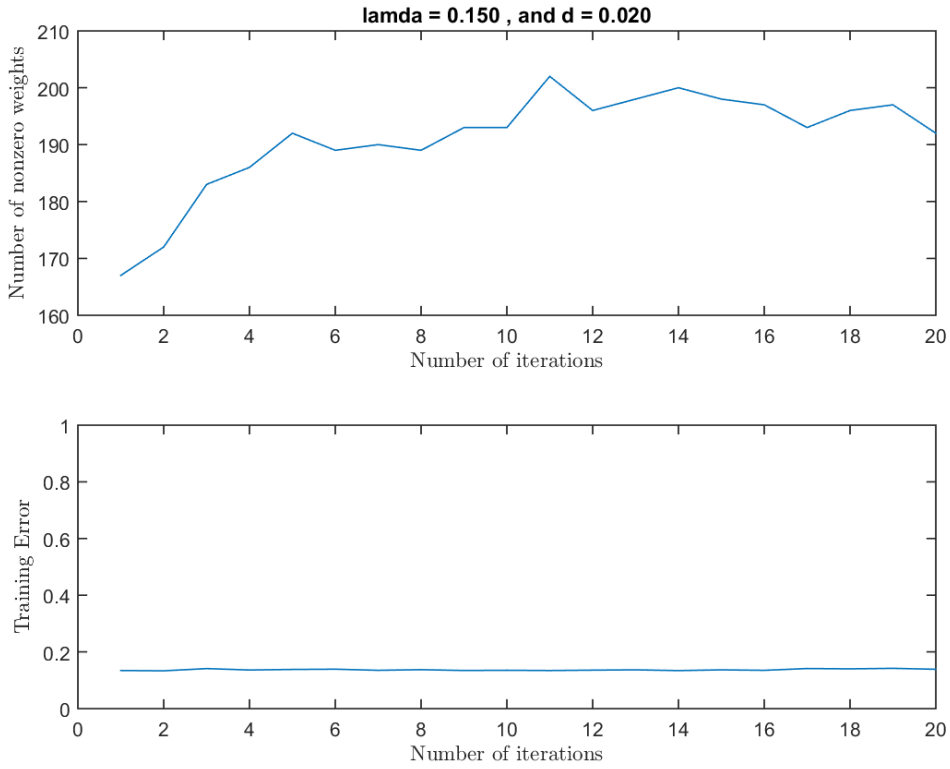


Εικόνα 18: Αποδόση Νευρωνικού Δικτύου με βέλτιστες παραμέτρους

Για το συγκεκριμένο ερώτημα ο κώδικας βρίσκεται στο αρχείο *bob.m*.

Η μέθοδος της αποσύνθεσης βαρών

Σκοπός μας με την μέθοδο της αποσύνθεσης των βαρών είναι να εξαλείψουμε βάρη του δικτύου που δεν υπερβαίνουν ένα όριο καθώς λειτουργούν περισσότερο ως θόρυβος. Για το σκοπό αυτό τρέξαμε τον τροποποιημένο αλγόριθμο που μας δόθηκε με διάφορες τιμές για την τιμή του λ,και του κατωφλίου d. Τελικά επιλέξαμε λ=0.15 και d=0.02



Εικόνα 19: Μη μηδενικά βάρη και μέσο τετραγωνικό σφάλμα - εκπαίδευση με weight decay

Η μέθοδος της αποσύνθεσης των βαρών υλοποιήθηκε στο αρχείο *weightDecay.m*.